

**ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO**

**FACULTAD DE INFORMÁTICA Y ELECTRÓNICA**

**ESCUELA DE INGENIERÍA EN SISTEMAS**

**“INYECCIÓN DE CONTENIDO EN UN VCMS (VIDEO CONTENT MANAGEMENT SYSTEM) Y LA AUTOMATIZACIÓN DE PUBLICACIONES EN MEDIOS SOCIALES (FACEBOOK, TWITTER)”**

Trabajo de titulación presentado para optar al grado académico de:

**INGENIERO EN SISTEMAS INFORMÁTICOS**

**AUTORES:** EDWIN ALEXANDER VILLA LÓPEZ

EDISON JAVIER RODRIGUEZ GUAMAN

**TUTOR:** DR. JULIO SANTILLÁN CASTILLO

Riobamba – Ecuador

2015

**ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO**

**FACULTAD DE INFORMÁTICA Y ELECTRÓNICA**

**ESCUELA DE INGENIERÍA EN SISTEMAS**

El Tribunal del Trabajo de Titulación certifica que: El trabajo de investigación “INYECCIÓN DE CONTENIDO EN UN VCMS (VIDEO CONTENT MANAGEMENT SYSTEM) Y LA AUTOMATIZACIÓN DE PUBLICACIONES EN MEDIOS SOCIALES (FACEBOOK, TWITTER)”, de responsabilidad de los señores Edwin Alexander Villa López, Edison Javier Rodriguez Guaman, ha sido minuciosamente revisado por los Miembros del Tribunal de Trabajo de Titulación, quedando autorizada su presentación.

**NOMBRE FIRMA FECHA**

Ing. Gonzalo Nicolay Samaniego Ph.D

DECANO DE LA FACULTAD DE

INFORMÁTICA Y ELECTRÓNICA …........................... .............................

Dr. Julio Santillán

DIRECTOR DE LA ESCUELA

DE INGENIERÍA EN SISTEMAS …........................... .............................

Ing. Julio Santillán

DIRECTOR DE TRABAJO DE

TITULACIÓN …........................... .............................

Ing. Fernando Proaño

MIEMBRO DEL TRIBUNAL …........................... .............................

DOCUMENTALISTA

SISBIB ESPOCH …........................... .............................

Nosotros, Edwin Alexander Villa López, Edison Javier Rodriguez Guaman somos responsables de las ideas, doctrinas y resultados expuestos en este trabajo, y el patrimonio intelectual del Trabajo de Titulación pertenece a la Escuela Superior Politécnica de Chimborazo”

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

EDWIN ALEXANDER VILLA LÓPEZ

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

EDISON JAVIER RODRIGUEZ GUAMAN

**DEDICATORIA**

Desde lo más profundo de mi corazón se lo dedico a Dios ya que gracias a su bendición he podido alcanzar este nuevo triunfo. Con mucho cariño a mis padres, por darme siempre su amor y comprensión todos los días. Por creer en mí y brindarme una carrera profesional para mi futuro. A mis hermanos Marcelo, Mary, Ligia y Letty por su apoyo incondicional y ayuda en todo momento. Este logro también lo comparto con ustedes. A mis maestros quien con su guía, confianza y ayuda colaboraron en el cumplimiento de esta meta, finalmente se lo dedico a mis amigos quienes me brindaron su confianza y apoyo para salir adelante.

Edwin

A Dios quien ha bendecido y ha guiado cada uno de mis pasos en este largo proceso compuesto de experiencias felices como tristes. A mis queridos padres que han depositado su confianza en mí, por sus sabios concejos que me han permitido no decaer a pesar de todas las adversidades que conlleva estar lejos del hogar, a mis hermanos José, Ángel por su apoyo y ayuda incondicional en todo momento, a mi hermanita Sofía quien es uno de mis principales motivos por los cuales intento superarme cada día. A mis maestros quienes con su sabiduría han sabido depositar en mi los conocimientos necesarios para poder seguir adelante en este camino que hoy no termina, sino al contrario acaba de empezar, por último pero no menos importante a mis amigos quienes me brindaron su amistad y cariño.

Edison

**AGRADECIMIENTO**

Primeramente damos gracias a Dios por todas las bendiciones brindadas durante toda esta etapa académica. A nuestras familias por su apoyo incondicional a lo largo de nuestras vidas estudiantiles, lo que nos ha permitido alcanzar uno de tantos anhelados sueños.

A la Escuela Superior Politécnica de Chimborazo y a nuestra querida escuela de Ingeniería en Sistemas, a todos los profesores quienes nos han inculcado los conocimientos necesarios para poder culminar esta etapa académica.

Agradecemos a nuestro director Dr. Julio Santillán por su tiempo y orientación para culminar con éxito nuestro trabajo de titulación, de la misma manera al Ing. Fernando Proaño por sus aportes y sugerencias que han sido de gran importancia.

No quisiéramos culminar sin agradecer a nuestros amigos quienes nos han brindado su amistad incondicional en los buenos y malos momentos.

Gracias totales.

Edwin y Edison

**ÍNDICE GENERAL**

**CERTIFICACIÓN………………………………………………………………………………….ii**

**DECLARACIÓN DE RESPONSABILIDAD…………………………………………………….iii**

**DEDICATORIA……………………………………………………………………………………iv**

**AGRADECIMIENTO………………………………………………………………………………v**

**ÍNDICE DE TABLAS……………………………………………………………………………..xii**

**ÍNDICE DE FIGURAS…………………………………………………………………………...xiv**

**ÍNDICE DE ANEXOS…………………………………………………………………………...xvii**

**ÍNDICE DE ABREVIATURAS………………………………………………………………...xviii**

[RESUMEN xix](#_Toc432579482)

[SUMARY xx](#_Toc432579483)

[INTRODUCCIÓN 1](#_Toc432579484)

[CAPITULO I 9](#_Toc432579485)

[1 MARCO TEÓRICO 9](#_Toc432579486)

[1.1 Gestión de Contenidos 9](#_Toc432579487)

[1.1.1 *Definición* 10](#_Toc432579488)

[1.1.2 *Procesos en la Gestión de Contenido* 11](#_Toc432579489)

[1.1.3 *Captura y Adquisición de Contenidos* 11](#_Toc432579491)

[1.2 Inyección de Contenido 13](#_Toc432579492)

[1.3 Sistemas de Gestión de Contenidos (CMS) 14](#_Toc432579494)

[1.3.1 *Breve Historia de los CMS* 15](#_Toc432579495)

[1.3.2 *Definición de un CMS* 15](#_Toc432579496)

[1.3.3 *Estructura y características* 17](#_Toc432579498)

[1.3.4 *Tipos de CMS* 18](#_Toc432579499)

[1.3.5 *Ventajas y Desventajas de los CMS* 19](#_Toc432579500)

[1.4 VCMS 20](#_Toc432579501)

[1.4.1 *Definición de un Sistema de Gestión de Contenidos de vídeo (VCMS)* 21](#_Toc432579503)

[1.5 JOOMLA 23](#_Toc432579505)

[1.5.1 *Definición* 24](#_Toc432579507)

[1.5.2 *Servicios de Joomla* 24](#_Toc432579508)

[1.5.3 *Datos Básicos sobre Joomla* 25](#_Toc432579509)

[1.6 Contus HD Video Share 26](#_Toc432579510)

[1.7 Marketing Online 27](#_Toc432579513)

[1.7.1 *Marketing en Redes Sociales (SMO-SMM).* 28](#_Toc432579514)

[**1.7.1.1** *Social Media Optimization (SMO)* 28](#_Toc432579515)

[**1.7.1.2** *Social Media Marketing (SMM)* 28](#_Toc432579516)

[**1.7.1.3** *Objetivos* 28](#_Toc432579517)

[**1.7.1.4** *Ventajas* 29](#_Toc432579518)

[**1.7.1.5** *Desventajas* 29](#_Toc432579519)

[1.8 Tráfico Web. 29](#_Toc432579520)

[1.8.1 *Fuentes de tráfico*. 30](#_Toc432579522)

[1.9 Frameworks 31](#_Toc432579523)

[1.9.1 *Definición*. 31](#_Toc432579524)

[1.9.2 *Características* 32](#_Toc432579525)

[1.9.3 *Ventajas* 33](#_Toc432579526)

[1.9.4 *Desventajas* 33](#_Toc432579527)

[1.10 Patrón de diseño MVC 33](#_Toc432579528)

[1.11 Estudio de tecnologías 34](#_Toc432579530)

[1.11.1 *Symfony* 38](#_Toc432579533)

[**1.11.1.1** *Características* 39](#_Toc432579534)

[1.11.2 *Laravel* 40](#_Toc432579535)

[**1.11.2.1** *Características* 40](#_Toc432579536)

[1.11.3 *CakePHP* 41](#_Toc432579537)

[**1.11.3.1** *Características* 41](#_Toc432579538)

[1.11.4 *Comparativa de Frameworks* 42](#_Toc432579540)

[1.12 Laravel 43](#_Toc432579542)

[1.12.1 *Filosofía de Laravel* 43](#_Toc432579543)

[1.12.2 *Requisitos del servidor* 43](#_Toc432579544)

[1.12.3 *Estructura* 44](#_Toc432579545)

[1.12.4 *Base datos* 45](#_Toc432579547)

[1.12.5 *Modelo* 46](#_Toc432579549)

[1.12.6 *Eloquent ORM* 47](#_Toc432579551)

[1.12.7 *Vista* 47](#_Toc432579552)

[1.12.8 *Blade* 48](#_Toc432579554)

[1.12.8.1 *Vista Master* 48](#_Toc432579555)

[1.12.8.2 *Vistas que Heredan* 49](#_Toc432579557)

[1.12.9 *Controlador* 50](#_Toc432579559)

[1.12.10 *Ruta* 51](#_Toc432579561)

[1.13 *Facebook* 52](#_Toc432579564)

[1.13.1 *Facebook para desarrolladores* 53](#_Toc432579565)

[1.13.2 *Graph API* 54](#_Toc432579566)

[**1.13.2.1** *Estructura de Graph API* 54](#_Toc432579567)

[**1.13.2.2** *Explorador de Graph API* 54](#_Toc432579569)

[1.13.3 *SDK para PHP* 55](#_Toc432579572)

[**1.13.3.1** *Instalación del SDK para PHP* 56](#_Toc432579573)

[1.13.4 *Permisos con el inicio de sesión*. 56](#_Toc432579575)

[1.13.5 *ConFiguración de la aplicación* 58](#_Toc432579577)

[1.13.6 *Autorización y autenticación* 58](#_Toc432579579)

[1.13.7 *Usar la Graph API* 59](#_Toc432579582)

[1.13.8 *Clases de SDK PHP* 61](#_Toc432579586)

[1.13.9 *Control de errores* 62](#_Toc432579588)

[1.13.10 *Fichas de acceso (Access Tokens)* 63](#_Toc432579590)

[1.14 Twitter 65](#_Toc432579593)

[1.14.1 *Twitter para desarrolladores* 65](#_Toc432579594)

[1.14.2 *Límites de petición a la API* 66](#_Toc432579595)

[1.14.3 *Explorador de API de Twitter* 67](#_Toc432579597)

[1.14.4 *Librerías de Twitter* 68](#_Toc432579599)

[1.14.5 *Métodos más relevantes* 70](#_Toc432579603)

[1.14.6 *Códigos de Error* 71](#_Toc432579610)

[1.15 YouTube 72](#_Toc432579612)

[1.15.1 *YouTube para desarrolladores* 73](#_Toc432579613)

[1.15.2 *YouTube Data API (v3)* 73](#_Toc432579614)

[1.15.3 *PHP Code* 73](#_Toc432579615)

[1.15.4 *Recursos y tipos de recursos* 74](#_Toc432579617)

[1.15.5 *Operaciones admitidas* 75](#_Toc432579619)

[1.15.6 *Uso de cuota* 76](#_Toc432579622)

[1.15.7 *Recursos parciales* 77](#_Toc432579623)

[1.15.8 *El parámetro part* 78](#_Toc432579624)

[1.15.9 *¿Qué es el parámetro fields?* 78](#_Toc432579625)

[1.15.10 *Ejemplo de solicitudes de la API* 79](#_Toc432579626)

[1.15.11 *YouTube Data API – Errors* 80](#_Toc432579628)

[**1.15.11.1** *Errores generales YouTube* 80](#_Toc432579629)

[1.16 Vimeo 81](#_Toc432579632)

[1.16.1 *Vimeo para desarrolladores* 82](#_Toc432579633)

[1.16.2 *Vimeo API* 82](#_Toc432579634)

[1.16.3 *Solicitudes a la API de Vimeo* 83](#_Toc432579635)

[CAPITULO II 87](#_Toc432579642)

[2 MARCO METODOLÓGICO 87](#_Toc432579643)

[2.1 Planeación 87](#_Toc432579644)

[2.1.1 *Requerimientos Específicos* 87](#_Toc432579645)

[2.1.2 *Factibilidad* 88](#_Toc432579647)

[2.1.3 *Estimaciones* 88](#_Toc432579648)

[2.1.4 *Historias de Usuario* 88](#_Toc432579649)

[2.1.5 *Iteraciones* 90](#_Toc432579651)

[2.1.6 *Cronograma* 91](#_Toc432579653)

[2.1.7 *Velocidad del Proyecto* 91](#_Toc432579655)

[2.1.8 *Reuniones* 92](#_Toc432579657)

[2.2 Diseño 93](#_Toc432579658)

[2.2.1 *Metáforas del sistema* 93](#_Toc432579659)

[**2.2.1.1** *Definición de la arquitectura del sistema* 93](#_Toc432579660)

[**2.2.1.2** *Diseño de la base de datos del sistema* 95](#_Toc432579663)

[**2.2.1.3** *Diccionario de datos* 96](#_Toc432579667)

[2.2.2 *Estándar de Codificación* 96](#_Toc432579669)

[2.2.3 *Estándar de Interfaces del sistema* 97](#_Toc432579670)

[2.2.4 *Tareas de ingeniería* 98](#_Toc432579672)

[2.2.5 *Tarjetas de clase, responsabilidad, colaboración (CRC cards)* 99](#_Toc432579674)

[2.2.6 *Soluciones puntuales (Spike Solution)* 99](#_Toc432579676)

[2.2.7 *Roles XP* 100](#_Toc432579677)

[2.3 Desarrollo 100](#_Toc432579679)

[2.3.1 *Iteración 1* 102](#_Toc432579681)

[2.3.2 *Iteración 2* 102](#_Toc432579683)

[2.3.3 *Iteración 3* 103](#_Toc432579685)

[2.3.4 *Iteración 4* 103](#_Toc432579687)

[2.3.5 *Iteración 5* 103](#_Toc432579689)

[2.3.6 *Iteración 6* 104](#_Toc432579691)

[2.4 Pruebas 104](#_Toc432579693)

[2.4.1 *Pruebas de Funcionalidad* 105](#_Toc432579695)

[2.4.2 *Ámbito de Pruebas* 106](#_Toc432579696)

[CAPITULO III 107](#_Toc432579700)

[3 MARCO DE RESULTADOS, ANÁLISIS Y DISCUSIÓN DE RESULTADOS 107](#_Toc432579701)

[CONCLUSIONES 120](#_Toc432579726)

[RECOMENDACIONES 121](#_Toc432579727)

[BIBLIOGRAFÍA 1](#_Toc432579728)

[ANEXOS 8](#_Toc432579729)

**ÍNDICE DE TABLAS**

[Tabla 1-1. Características de los VCMS 22](#_Toc432578038)

[Tabla 2-1. Cantidad de Paquetes de integración. 42](#_Toc432578065)

[Tabla 3-1. Comparación de Frameworks 42](#_Toc432578067)

[Tabla 4-1. Tipo de peticiones a la Graph API 54](#_Toc432578091)

[Tabla 5-1. Descripción principales permisos de Facebook 57](#_Toc432578097)

[Tabla 6-1. Funciones para la autorización y autentificación de la aplicación 59](#_Toc432578102)

[Tabla 7-1. Principales clases de la SDK de PHP 61](#_Toc432578108)

[Tabla 9-1. Límite de peticiones a la API 67](#_Toc432578117)

[Tabla 10-1. Códigos de error 72](#_Toc432578132)

[Tabla 11-1. Recursos y tipos de recursos 74](#_Toc432578139)

[Tabla 12-1. Métodos que admite la API de YouTube 75](#_Toc432578141)

[Tabla 13-1. Operaciones admitidas 75](#_Toc432578142)

[Tabla 14-1. Errores API de YouTube 80](#_Toc432578150)

[Tabla 15-1. Errores comunes de YouTube 80](#_Toc432578151)

[Tabla 16-1. Campos del API de Vimeo 83](#_Toc432578156)

[Tabla 17-1. Métodos más relevantes del API de Vimeo 83](#_Toc432578157)

[Tabla 18-1. Representación de colección 85](#_Toc432578160)

[Tabla 19-1. Errores comunes del API de Vimeo 86](#_Toc432578161)

[Tabla 1-2. Requerimientos específicos 87](#_Toc432578166)

[Tabla 2-2. Historias de usuario 89](#_Toc432578170)

[Tabla 3-2. Iteraciones 90](#_Toc432578172)

[Tabla 4-2. Velocidad del proyecto 92](#_Toc432578176)

[Tabla 5-2. Diccionario de datos Tabla Config 96](#_Toc432578185)

[Tabla 6-2. Modelo de las tareas de ingeniería 98](#_Toc432578190)

[Tabla 7-2. Modelo de la tarjetas CRC 99](#_Toc432578192)

[Tabla 8-2. Roles XP 100](#_Toc432578195)

[Tabla 9-2. Historias de usuario priorizadas 101](#_Toc432578197)

[Tabla 10-2. Iteración 1 102](#_Toc432578199)

[Tabla 11-2. Iteración 2 102](#_Toc432578201)

[Tabla 12-2. Iteración 3 103](#_Toc432578203)

[Tabla 13-2. Iteración 4 103](#_Toc432578205)

[Tabla 14-2. Iteración 5 103](#_Toc432578207)

[Tabla 15-2. Iteración 6 104](#_Toc432578209)

[Tabla 16-2. Recursos Cliente 106](#_Toc432578214)

[Tabla 17-2. Recursos Servidor 106](#_Toc432578215)

[Tabla 18-2. Recursos Red 106](#_Toc432578216)

[Tabla 1-3. Datos de prueba en YouTube 109](#_Toc432578223)

[Tabla 2-3. Datos de prueba en Vimeo 112](#_Toc432578229)

[Tabla 3-3. Datos de prueba en Facebook 115](#_Toc432578235)

[Tabla 4-3. Datos de prueba en Twitter 118](#_Toc432578241)

**ÍNDICE FIGURAS**

[Figura 1-1. Procesos de Gestión de Contenidos 11](#_Toc433049004)

[Figura 2-1. Proceso de Inyección de Contenido 14](#_Toc433049007)

[Figura 3-1. Anatomía de un Sistema de Gestión de Contenidos 16](#_Toc433049011)

[Figura 4-1. Cuadrante mágico de Gartner, La Gestión de Contenido 21](#_Toc433049016)

[Figura 5-1. Logotipo de Joomla 23](#_Toc433049020)

[Figura 6-1. Servicios de Joomla 25](#_Toc433049023)

[Figura 7-1. Logotipo Contus HD Video Share 26](#_Toc433049026)

[Figura 8-1. Funcionalidad de Joomla y Contus Video Share 27](#_Toc433049027)

[Figura 9-1. Trafico Web 30](#_Toc433049031)

[Figura 10-1. Interpretación de Framework 32](#_Toc433049035)

[Figura 11-1. Patrón de Diseño MVC 34](#_Toc433049040)

[Figura 12-1. Encuesta Php Frameworks más popular 2015 35](#_Toc433049042)

[Figura 13-1. Tendencia de búsquedas de Frameworks PHP 36](#_Toc433049043)

[Figura 14-1. Ranking de repositorios de GitHub 37](#_Toc433049044)

[Figura 15-1. Estructura de directorios en Laravel. 45](#_Toc433049056)

[Figura 16-1. ConFiguración para la base de datos. 45](#_Toc433049059)

[Figura 17-1. Modelo Utilizando Eloquent 46](#_Toc433049061)

[Figura 18-1. Código en la Vista 47](#_Toc433049064)

[Figura 19-1. Vista Master 49](#_Toc433049066)

[Figura 20-1. Vista heredada 50](#_Toc433049067)

[Figura 21-1. Controlador en Laravel 51](#_Toc433049069)

[Figura 22-1. Archivo de rutas en Laravel 51](#_Toc433049071)

[Figura 23-1. Proceso de petición al controlador 52](#_Toc433049072)

[Figura 24-1. Explorador de la Grahp API 55](#_Toc433049077)

[Figura 25-1. Respuesta de la Graph API 55](#_Toc433049078)

[Figura 26-1. ConFiguración para la instalación 56](#_Toc433049080)

[Figura 27-1. ConFiguración de la aplicación 58](#_Toc433049084)

[Figura 28-1. Autorización y autentificación 59](#_Toc433049086)

[Figura 29-1. Código para solicitar la información del usuario 60](#_Toc433049089)

[Figura 30-1. Código para realizar una publicación 60](#_Toc433049090)

[Figura 31-1. Código para publicar una foto de perfil 61](#_Toc433049091)

[Figura 32-1. Pasos para obtener token de acceso 63](#_Toc433049097)

[Figura 33-1. Proceso para solicitud de cambio de token 64](#_Toc433049098)

[Figura 34-1. Explorador de la API de Twitter 68](#_Toc433049104)

[Figura 35-1. Código para la instalación Twitter 69](#_Toc433049106)

[Figura 36-1. Código para hacer uso de la librería 69](#_Toc433049107)

[Figura 37-1. ConFiguración de la API 70](#_Toc433049108)

[Figura 38-1. Autorización de la aplicación 70](#_Toc433049110)

[Figura 39-1. Construcción de la URL de autenticación 70](#_Toc433049111)

[Figura 40-1. Llamado de peticiones en Twitter 71](#_Toc433049112)

[Figura 41-1. Construcción de la URL de autenticación 71](#_Toc433049113)

[Figura 42-1. Peticiones de tipo POST 71](#_Toc433049114)

[Figura 43-1. Actualización de token 71](#_Toc433049115)

[Figura 44-1. Instalación mediante Composer 74](#_Toc433049122)

[Figura 45-1. URL de solicitud de YouTube Api 80](#_Toc433049133)

[Figura 46-1. Parámetros de método GET 84](#_Toc433049143)

[Figura 47-1. Parámetros de método GET 84](#_Toc433049144)

[Figura 1-2. Cronograma de Actividades 91](#_Toc433049159)

[Figura 2-2. Arquitectura del sistema completo 94](#_Toc433049165)

[Figura 3-2. Arquitectura de la aplicación OpenPublish 94](#_Toc433049166)

[Figura 4-2. Modelo Entidad Relación de la base de datos 95](#_Toc433049167)

[Figura 5-2. Modelo Físico de la base de datos 95](#_Toc433049168)

[Figura 6-2. Tablas utilizadas de la base de datos de alavista.tv 96](#_Toc433049169)

[Figura 7-2. Interfaz Pantalla Principal 98](#_Toc433049173)

[Figura 8-2. Revista Perspectivas 105](#_Toc433049196)

[Figura 1-3. Interfaz de administración de alavista.tv para YouTube 108](#_Toc433049204)

[Figura 2-3. Interfaz de administración de OpenPublish para YouTube 108](#_Toc433049205)

[Figura 3-3. Tiempo manual para YouTube 109](#_Toc433049206)

[Figura 4-3. Tiempo obtenido con OpenPublish para YouTube 109](#_Toc433049207)

[Figura 5-3. Inyección de Contenido de YouTube 110](#_Toc433049209)

[Figura 6-3. Interfaz de administración de alavista.tv para Vimeo 111](#_Toc433049210)

[Figura 7-3. Interfaz de administración de OpenPublish para Vimeo 111](#_Toc433049211)

[Figura 8-3. Tiempo manual para Vimeo 112](#_Toc433049212)

[Figura 9-3. Tiempo obtenido con OpenPublish para Vimeo 112](#_Toc433049213)

[Figura 10-3. Inyección de Contenido de Vimeo 113](#_Toc433049215)

[Figura 11-3. Publicación en Facebook 114](#_Toc433049216)

[Figura 12-3. Publicación en Facebook con OpenPublish 114](#_Toc433049217)

[Figura 13-3. Tiempo manual para Facebook 115](#_Toc433049218)

[Figura 14-3. Tiempo obtenido con OpenPublish para Facebook 115](#_Toc433049219)

[Figura 15-3. Publicación de Contenido en Facebook 116](#_Toc433049221)

[Figura 16-3. Publicación en Twitter 117](#_Toc433049222)

[Figura 17-3. Publicación en Twitter con OpenPublish 117](#_Toc433049223)

[Figura 18-3. Tiempo manual para Twitter 118](#_Toc433049224)

[Figura 19-3. Tiempo obtenido con OpenPublish para Twitter 118](#_Toc433049225)

[Figura 20-3. Publicación de Contenido en Twitter 119](#_Toc433049227)

**ÍNDICE DE ANEXOS**

**ANEXO A.** Estudio de factibilidad

**ANEXO B.** Estimaciones

**ANEXO C.** Diccionario de datos

**ANEXO D.** Estandar de interfaces

**ANEXO E.** Detalle de historias de usuario, tareas de ingeniería, actividades, pruebas de aceptación

**ANEXO F.** Detalle de Tarjetas CRC

**ÍNDICE DE ABREVIATURAS**

**API** (Application Programming Interface) Interfaz de Programación de Aplicaciones

**SDK** (Software Development Kit) Kit de Desarrollo de Software

**SRL** Sociedad de Responsabilidad Limitada

**PHP** (Hypertext Preprocessor) Preporcesador de hipertexto

**SQL** (Structured Query Language) Lenguaje de Consulta Estructurado

**ESPOCH** Escuela Superior Politécnica de Chimborazo

**VCMS** (Video Content Management System) Sistema de Gestión de Contenido de Vídeo

**MVC** (Model View Controller) Modelo Vista Controlador

**CMS** (Content Management System) Sistema de Gestión de Contenido

**HTML** (HyperText Markup Language) Lenguaje de Marcas de Hipertexto

**CSS** (Cascading Style Sheets) Hoja de estilos de cascada

**JSON** (JavaScript Object Notation)

**AJAX** (Asynchronous JavaScript And XML) JavaScript Asíncrono y XML

**ONG** Organización no Gubernamental

**ISS** (Internet Information Services)

**RSS** (Really Simple Syndication)   
**MIT** (Massachusetts Institute of Technology)   
**ORM** (Object-relational mapping) Mapeo Objeto-Relacional  
**URL** (Uniform Resource Locator) Localizador de Recursos Uniforme

**TIC's** Tecnologías de la información y la comunicación

**XP** (Xtreme Programming) Programación Extrema  
**XML** (Extensible Markup Language) Lenguaje de marcas de hipertexto

**FTP** (File Transfer Protocol) Protocolo de Trasferencia de Archivos

**CSRF** (Cross-site request forgery) Falsificación de petición en sitios cruzados

**XSS** (Cross-site scripting)

**CRUD** (Create, Read, Update and Delete)  Crear, Obtener, Actualizar y Borrar

**SMM** (SOCIAL MEDIA MARKETING) Marketing en Medios Sociales

**SMO** (Social Media Optimization) Optimización en Medios Sociales

# RESUMEN

Este trabajo comprende el estudio de la inyección de contenido en un Video Content Management System (VCMS) basado en Joomla y la publicación en medios sociales como: Facebook y Twitter, mediante el desarrollo de una aplicación Web para el canal de videos Alavista.tv de la empresa GengisCan SRL, la cual se encarga de gestionar la inyección de contenido y la publicación del mismo en medios sociales. Este trabajo se ejecutó a través de las principales técnicas de recopilación de información como: observación y entrevista. Se desarrolló la aplicación Web OpenPublish con el Framework PHP Laravel y la Interfaz de Programacion de Aplicaciones (APIs) de desarrollo correspondiente a YouTube, Vimeo, Facebook y Twitter, la cual consta de los módulos de administración de usuarios, administración de videos, administración de redes sociales, galería de reproducción multimedia e información estadística. Se realizó la comparación entre la gestión de contenido ejecutada mediante OpenPublish y la gestión de contenido ejecutada manualmente, obteniendo como resultado que al inyectar la cantidad de 50 videos desde YouTube hay un ahorro de tiempo de 3312,193 segundos, de igual forma en Vimeo 3333,575 segundos, de la misma manera al realizar la difusión en redes sociales en Facebook se tiene un ahorro de tiempo de 1002,73 segundos, y en el caso de Twitter 130.39 segundos. Se concluye que se obtiene una mejora en la gestión de contenido mediante la utilización de la aplicación Web, obteniendo así un incremento en el tráfico de red en el canal de videos alavista.tv. Se recomienda realizar el control de las APIs utilizadas para conocer los posibles cambios que pueden tener y así evitar problemas en la conFiguración de OpenPublish.

**Palabras claves:** <INYECCIÓN DE CONTENIDO>, <REDES SOCIALES>, <SISTEMA DE GESTIÓN DE CONTENIDO DE VIDEO [VCMS]>, <FRAMEWORK [LARAVEL]>, <GESTIÓN DE CONTENIDO>, <APLICACIÓN WEB>, <CANALES DE VIDEO>

# SUMARY

This research has to do with content injection study in a (VCMS) Video Content Management System based on Joomla and its publishing in the social media such as: Facebook and Twitter through the implementation of Web application Alavista.tv video channel belonging to GengisCan SRL Company, which is in charge of managing the content and its publishing in the social media. This work was implemented within Laravel PHP Framework and (APIs) Application Programming Interface belonging to YouTube, Vimeo, Facebook and Twitter which contain modules for user´s administration, videos administration, social network administration, playback’s multimedia gallery and statistical information. It was necessary to make a comparison between the management of content using OpenPublish and the manual management of content using which showed that then sending 50 videos from YouTube, it is possible to save3312,193 seconds, the same as Vimeo which saves 3333,575 seconds. When diffusing the content via social networks, it is possible to save 1002,73 seconds in Facebook and 130,39 second in Twitter. It is concluded that is possible to get a better content management with the use of Web application, obtaining a network traffic increase in alavista.tv video channel; thus it is recommended to monitor the APIs used in order to know about the possible changes they can have to avoid problems in the OpenPublish conFiguration.

**Key words:** <CONTENT INJECTION>, <SOCIAL NETWORKS>, < [VCMS] VIDEO CONTENT MANAGEMENT SYSTEM>, < [LARAVEL] FRAMEWORK>, <CONTENT MANAGEMENT>, <WEB APPLICATION>, <VIDEO CHANNELS>

INTRODUCCIÓN

La empresa GengisCan SRL conformada con el fin de crear servicios innovadores de entrenamiento para la Web y dispositivos móviles, cuenta con varios productos que pone a disposición del público. Uno de estos productos es el sitio Web alavista.tv dedicada a gestionar videos, donde sus usuarios pueden publicar y compartir videos de manera fácil y sencilla.

El portal se encuentra desarrollado con uno de los sistemas de gestión de contenidos(CMS) de código abierto más utilizados actualmente llamado Joomla, el cual proporciona una gama de posibilidades para administrar un sitio Web de manera segura y escalable dependiendo las necesidades que se requiera. Gracias a su estructura, organización de contenido hace que su administración no sea complicada. (Joomla Organización, 2015, [http://ayuda.joomlaspanish.org](http://ayuda.joomlaspanish.org/que-es-joomla))

La empresa GengisCan SRL tiene como objetivo principal posicionar la marca de alavista.tv en el Ecuador, para poder cumplir con este objetivo se necesita contar con una aplicación Web, la cual estará conformada por dos componentes principales, inyección de contenido y publicación del contenido en medios sociales, para de esta manera dar solución a la necesidad presentada.

Como primer componente para dar solución al problema se necesita contar con un módulo que permita la automatización de carga de contenido, en este caso videos en el sitio Web, de esta manera mejorar la calidad del servicio que se pretende ofrecer al público, con el objetivo principal de proporcionar contenido que esté de acuerdo al interés social y cultural del público que se quiere captar.

Como segundo componente luego de tener contenido disponible para los usuarios es necesario contar con un módulo que permita dar a conocer el contenido que se dispone, haciendo uso del marketing online se llevará acciones de tipo Social Media Marketing o Marketing en Redes Sociales (SMM), considerando Facebook y Twitter las redes sociales más utilizadas en la actualidad, con el objetivo de captar mayor número de posibles visitantes al sitio Web. (Pes Rivas C, 2011, p.41)

El presente trabajo de titulación trata del desarrollo de una aplicación Web con el lenguaje de programación PHP, y la integración de las APIs de desarrollo necesarias para cumplir con el objetivo planteado por la empresa GengisCan SRL, todo el desarrollo se realizará con la ayuda de un Framework seleccionado previamente en base a un estudio entre los posibles candidatos.

El trabajo de titulación “INYECCIÓN DE CONTENIDO EN UN VCMS (VIDEO CONTENT MANAGEMENT SYSTEM) Y LA AUTOMATIZACIÓN DE PUBLICACIONES EN MEDIOS SOCIALES (FACEBOOK, TWITTER)” está dividido en los siguientes capítulos:

Capítulo I. Está relacionado con el marco teórico, detallando los conceptos de estudio necesarios que ayuden al desarrollo del trabajo de investigación. Abordando aspectos como definiciones, características, ventajas y desventajas. Ademas está relacionado con el estudio de las tecnologías utilizadas.

Capítulo II. Se enfoca en el estudio de la metodología de desarrollo de la aplicación Web detallando cada una de las fases empleadas.

Capítulo III. Describe el marco de resultados, análisis de los resultados obtenidos en el trabajo de titulación.

Finalmente se presentan las conclusiones y recomendaciones que se obtuvieron luego de realizar el trabajo de titulación, igualmente se muestran los respectivos anexos de cada capítulo.

A continuación se detallará la formulación general del trabajo de titulación dentro del cual se analizará los antecedentes, justificación y objetivos a alcanzar en presente trabajo de titulación.

**ANTECEDENTES**

En la actualidad se usan estrategias de comercialización, las cuales son guiadas y llevadas a cabo por canales de comunicación los cuales son conocidos como medios digitales. Existen varias técnicas entre las cuales podemos hablar de las del mundo off-line, dichas técnicas son llevadas a un nuevo mundo online en donde aparecen nuevas herramientas.

La primera estrategia se basa en la Web 1.0, que utiliza medios tradicionales, la principal característica es la imposibilidad de comunicación y la empresa era la única que tenía el control**.** La segunda estrategia es la Web 2.0, la cual brinda la posibilidad de compartir información de manera fácil gracias a las redes sociales y las nuevas herramientas tecnológicas de información que permiten el intercambio rápido de información como videos, imágenes, etc. Es así que se empieza a usar el Internet no solo para realizar búsquedas sino como un medio de comunicación con usuarios de diferentes partes del mundo.

La Empresa GengisCan SRL, se conforma con el objetivo de crear servicios innovadores de entretenimiento para la Web y Móviles. GengisCan es una empresa dedicada al desarrollo de software y asesoría en implementación de servicios de Streaming.

Sus marcas son las siguientes:

* **Ebetrix**: www.ebetrix.com
* **Alavista**: www.alavista.tv

Alavista.tv es una Web dedicada a videos “www.alavista.tv”, donde usuarios pueden compartir videos en formato digital, permite publicar una gran variedad de contenido como videos, documentales, filmaciones, tutoriales, programas, entre otros.



**Figura 1.** Logotipo de alavista.tv

**Fuente:** alavista.tv/images/alavista\_logo.png

Alavista.tv ofrece una plataforma de videos segura y escalable, con este VCMS hace que sea fácil ordenar, gestionar y organizar su biblioteca multimedia, brinda las características de edición de descripción, agregar una categoría, agregar tags, privacidad (público, privado), listas de reproducción entre otras, alavista.tv ofrece todo lo necesario para organizar y gestionar el contenido de video.

El portal se encuentra desarrollado con el sistema de gestión de contenidos más usados actualmente como lo es Joomla, el cual permite administrar el sitio de una manera fácil para el usuario, el CMS carece de un módulo que permite al usuario automatizar el proceso de publicación de videos en el portal, y a su vez realizar el proceso socialización en medios de comunicación como lo son Facebook y Twitter.

Muchas empresas han optado por optimizar sus tareas, ya que tienen la necesidad de adoptar y optimizar dichos procesos, pero en muchos de los casos carecen de conocimiento de cómo llevarla a cabo.

Alavista.tv es un canal de video donde la empresa GengisCan SRL brinda soporte a este producto, a petición del cliente se necesita crear un sistema de publicación de videos el cual permita realizar búsquedas por criterio (título, duración, usuario, fecha, entre otros) de canales similares como YouTube y Vimeo, una vez que se obtenga la información requerida se necesita un sistema que permita realizar una inyección de contenido a la base de datos de alavista.tv con los videos que se han obtenido de los canales antes mencionados, y a la ves la posibilidad de poder compartir estos videos en los medios sociales Facebook y Twitter.

Los problemas que se presentan al momento de gestionar contenidos multimedia, como por ejemplo la publicación vídeos en el sitio Web se lo realiza de forma manual, desde el proceso de búsqueda, obtener información de los videos desde los canales de YouTube y Vimeo, hasta el momento de la publicación genera gastos de recursos económicos, humanos así como también un mayor tiempo empleado para realizar las actividades antes mencionadas. De la misma manera la difusión de los videos publicados en las diferentes redes sociales (Facebook, Twitter) se los realiza de forma manual, produciendo mal uso de recursos.

Estos procesos realizados de forma manual contraen varios problemas, uno de ellos consiste en que el contenido publicado en el portal Web no es promocionado de una manera correcta y en el momento adecuado en las redes sociales, otro de los problemas más importantes a considerar en un portal Web es la pérdida de usuarios. Mediante la socialización del contenido publicado se pretende acaparar el mayor número de usuarios hacia la página Web de alavista.tv, de esta manera generar tráfico Web.

El tráfico Web hace referencia a la cantidad de usuarios que hacen uso de un sitio Web. Gracias al trafico Web se puede constatar la rentabilidad que posee una página Web, mientras más usuarios accedan se podrá obtener mayores oportunidades de ventas, publicidad, contactos, etc.

**JUSTIFICACIÓN DEL TRABAJO DE TITULACIÓN**

**Justificación Teórica**

Hoy en día la mayoría de procesos se realizan de manera automatizada gracias a la ayuda de las TICS (Tecnologías de la Información y Comunicación), optimizando recursos tanto económicos como humanos de una manera notable en diferentes áreas, tal es el caso del proceso de la gestión de contenidos de diferentes tipos.

Actualmente se cuenta con la ayuda de los CMS (Content Management Systems), los cuales son un software que permite gestionar contenidos, existen diferentes tipos de CMS, los más usados en la actualidad son:

● Contenidos Empresariales (ECM).

● Contenidos Web (WCM).

● Documentos y/o contenidos multimedia (DMS).

● Contenidos para el Aprendizaje (LCMS).

Centrándose principalmente en los CMS de contenidos Web (WCM), el cual permite crear, modificar, y publicar contenidos en portales Web dirigidos a una audiencia interna o externa. Joomla es un gestor de contenidos Web que permite crear sitios Web y poderosas aplicaciones online. Muchos aspectos, incluyendo su fácil uso y extensibilidad han convertido a Joomla en el software para la Web más popular. Joomla funciona con una base de datos MySQL y está escrito en PHP. (Valdés Pérez D, 2007, http://www.maestrosdelWeb.com)

Para poder dar solución a los problemas mencionados en los antecedentes e ir acorde con las tecnologías que utiliza alavista.tv, se hará uso de herramientas tales como:

MySQL que es un motor de base de datos relacional, es hoy en día uno de los más importantes en lo que es el diseño y gestión de base de datos.

PHP es un lenguaje de programación del lado del servidor más conocido en la actualidad gracias a que su sintaxis es fácil y sencilla, además de que su conFiguración es mínima especialmente utilizado para el desarrollo Web dinámico.

Para llevar a cabo el desarrollo Web se realizará un estudio de las APIs de desarrollo que proporcionan tanto los canales de video YouTube y Vimeo, así como también las redes sociales Facebook y Twitter.

**Justificación Aplicativa**

En el trabajo investigativo/aplicativo se desarrollará la aplicación Web denominada OpenPublish la cual permitirá administrar la publicación de videos, desde el proceso de búsqueda mediante varios parámetros y criterios de búsqueda en los canales de videos como YouTube y Vimeo, para posteriormente realizar una inyección de contenido en la base de datos de alavista.tv de los videos obtenidos anteriormente y seleccionados por el administrador.

OpenPublish también permitirá el anuncio de los videos publicados en alavista.tv mediante un módulo de publicación en medios sociales como Facebook tanto en grupos como en el perfil personal, además permitirá publicarlo en Twitter.

Así también contendrá módulos de administración de usuarios que permitirá gestionar la seguridad que debe tener una página de administración de contenido Web.

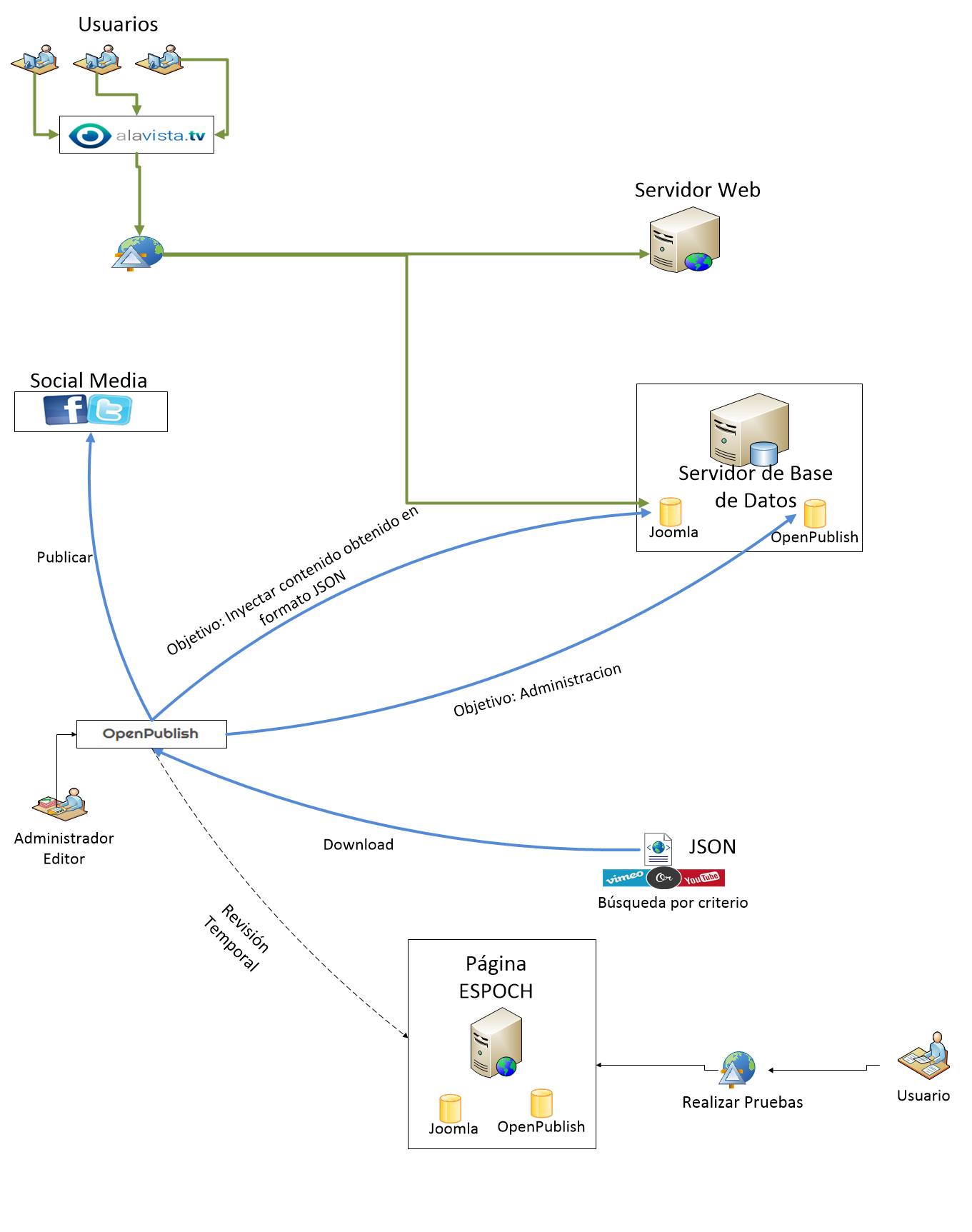
Además tendrá un módulo donde se podrá observar los videos que se han publicado en el portal alavista.tv, el cual permitirá dar un seguimiento del contenido que se encuentra disponible, OpenPublish se desarrollará como una aplicación Web, la cual además de ser una aplicación de escritorio será administrable desde dispositivos móviles mediante el uso de diseño Responsive, de esta manera será posible abarcar la mayoría de dispositivos tecnológicos con un solo producto, sin necesidad de llevar a cabo el desarrollo diferente para cada dispositivo.

Por otro lado, la investigación contribuye a la Política 11.3 que menciona “Democratizar la prestación de servicios públicos de telecomunicaciones y de tecnologías de información y comunicación (TIC), incluyendo radiodifusión, televisión y espectro radioeléctrico, y profundizar su uso y acceso universal.”, la cual se encuentra en el objetivo número 11 del Plan Nacional del Buen Vivir.

Además, aporta al “Programa para el desarrollo de aplicaciones de software para procesos de gestión y administración pública y privada. Educación.” De las áreas y líneas de Investigación de la ESPOCH.

Para poder comprobar el funcionamiento correcto de OpenPublish se lo utilizará sobre una aplicación Web de la ESPOCH creada en el CMS (Content Management System) Joomla, así se podrá realizar pruebas y estandarización del sistema previo a la entrega del mismo.

A continuación en la Figura 2, se presenta un esquema detallado de los componentes que contendrá la solución a aplicar.



**Figura 2.** Propuesta de la solución

**Realizado por**: Villa E., Rodriguez E., 2015

**OBJETIVOS**

**Objetivo General**

Inyectar contenido en un VCMS (Video Content Management System) y automatizar publicaciones en medios sociales (Facebook, Twitter).

**Objetivos Específicos**

* Estudiar la inyección de contenidos en los VCMS
* Estudiar las características principales de los Frameworks de desarrollo Web PHP y conocer las tecnologías que se utilizan como APIs de canales de video y de redes sociales para luego definir el adecuado para el desarrollo de la aplicación.
* Desarrollar la solución planteada para la aplicación Web OpenPublish
* Comprobar el funcionamiento correcto de OpenPublish sobre una aplicación Web de la ESPOCH creada en el CMS (Content Management System) Joomla
* Entregar la aplicación OpenPublish a GengisCan SRL

# CAPITULO I

# MARCO TEÓRICO

En este capítulo se analizan las bases y fundamentos, que servirá como base para entender los conceptos necesarios para el desarrollo de la investigación. En primer lugar se definirá lo que es Gestión de contenido, Inyección de contenido, que es un Sistema de Gestión de Contenido y tipos de sistemas, seguidamente se definirá los VCMS, características, funcionamiento y sus tipos con el fin de comprender el ámbito en el cual se trabajará. Posterior a esto se estudia de manera menos detallada el Marketing online, Marketing en Medios Sociales, como sus características, ventajas y desventajas, y por último se observa conceptos correspondientes a Frameworks y patrón de diseño MVC.

## Gestión de Contenidos

La gestión de contenidos apareció en la década de los noventa, esto como respuesta a la necesidad de poder crear, dar seguimiento y mantenimiento a sitios Web que requerían de la colaboración y organización de varios usuarios (autores, editores, diseñadores gráficos). La gestión de contenidos en la actualidad es una actividad que se lo utiliza en un entorno académico y también en un entorno profesional, el último entorno mencionado anteriormente ha sido desarrollado de manera tradicional por profesionales de informática.

El incremento de una sociedad de información, ha dado inicio a la incorporación de profesionales de la información en nuevos espacios relacionados a la gestión de información digital, la gestión de contenidos se caracteriza por traducir ideas, esto requiere de una variedad de capacidades que inician desde el conocimiento de las audiencias hasta la explotación de la convergencia tecnológica, incluyendo la dirección de personas o el dominio de las herramientas más eficaces de marketing. (Caraballo Pérez Y, 2007, http://bvs.sld.cu/revistas)

A continuación se contrastan algunas definiciones del término “gestión de contenido” con el fin de facilitar su comprensión.

### Definición

Para entender que es la Gestion de contenido se enuncian varias definiciones descritas por sus respectivos autores.

Ricardo Eíto-Brun, dice que las primeras definiciones del termino gestión de contenidostenía que ver con una aplicación informática y algunos procesos para crear, editar, controlar y publicar contenidos en la Web. Además da a conocer una definición más actual que dice: “En la actualidad se utiliza el término con un alcance más amplio y global que incluyen los procesos, recursos e infraestructura que hacen posible el control físico e intelectual de los contenidos digitales internos y externos, y la aplicación de las reglas que rigen su captura, almacenamiento, categorización y distribución a través de múltiples canales y dispositivos.” (Eiton Brun R, 2013, pp. 17-19)

Según el punto de vista de Boiko (2002), "la gestión de contenidos es el proceso de tratar de emparejar lo que usted tiene con lo que ellos desean, donde usted es una organización con información y funcionalidad como valor, y ellos son un sistema de audiencias definibles que desean ese valor. La gestión de contenidos puede parecer una manera de crear grandes sitios Web, pero en un examen más cercano es, en hecho, un proceso total para recoger, manejar y publicar contenido". (Yorbelis Rossel L,2011,http://scielo.sld.cu)

En palabras de Gerry Macgovern “la gestión de contenidos consiste en llevar el contenido correcto a la persona adecuada en su momento justo al coste idóneo”. (Serrano Cobos J.,2007,http://www.academia.edu). En el artículo “Gestión de Contenidos, un enfoque independiente” de Roberto Canales Mora da a conocer su propia definición, “Todos los procedimientos y procesos involucrados en el agregación, transformación, catalogación, agrupación, autorización, presentación y distribución de información útil para nuestros propósitos”. (Mora Canales R, 2013, http://www.adictosaltrabajo.com)

De lo expuesto anteriormente en la presente investigación se define como gestión de contenidos al uso de técnicas y herramientas que ayuden a crear, editar y dar mantenimiento a la información útil para nuestros propósitos.

### Procesos en la Gestión de Contenido

Se debe tener en cuenta que la gestión de contenidos nace de la necesidad de poder crear, gestionar y mantener un repositorio de contenidos, los cuales deben almacenarse, distribuirse, recuperarse atendiendo a las necesidades de cada uno de los usuarios. Entonces las organizaciones deben implantar procesos, los cuales pueden ser soportados, total o parcialmente, por las soluciones informáticas que se utilicen. Los procesos de gestión de contenidos serán comunes para diferentes contextos y áreas de negocio, y no dependerán del problema específico que se requiera resolver

En la Figura 1-1 se presenta un modo simple de procesos, cada proceso se divide en una o más actividades que se describen brevemente.

**Figura 1-1.** Procesos de Gestión de Contenidos

**Fuente:** (Eíton-Brun R., 2013, p. 28)

### Captura y Adquisición de Contenidos

Abarca todas las actividades para capturar contenido relevante, la selección de contenidos dependerá del ámbito del negocio particular. La captura y adquisición de contenidos incluye estas actividades.

* **Edición de contenido.-** la creación y edición de contenido debe poder realizarse con facilidad, usando editores de texto similares a los que utilizan los usuarios.
* **Adquisición de contenidos internos.-** se deben identificar contenidos generados internamente, estos contenidos pueden ser documentos creados mediante herramientas de ofimática.
* **Adquisición de contenidos externos.-** no solo se gestiona contenidos generados por el propio personal de una organización, esta adquisición puede consistir en la obtención de ficheros de distintos formatos que sean llevados al repositorio y tratándolos como recursos internos.
* **Revisión y aprobación de contribuciones.-** los contenidos internos y externos deben ser revisados antes de incorporarlos al repositorio, así se asegura la calidad de la información y se evitan posible errores.
* **Declaración de registros.-** se refiere a la necesidad de marcar ciertos documentos como registros, así se garantiza su almacenamiento durante un periodo establecido y no alterarlo accidental o voluntariamente.
* **Control del Repositorio.-** los contenidos que se obtienen mediante el proceso anterior se almacenarán y controlarán en un repositorio, el gestor de contenidos asegurará la integridad del repositorio y además interactuará con las aplicaciones de los clientes que desean tener acceso a los datos, este proceso tiene las siguientes actividades.
* **Almacenamiento seguro.-** la forma en la que se almacenan los contenidos en el repositorio debe ser transparente para los usuarios, los usuarios no deben tener conocimiento de la organización física de los contenidos para acceder a ellos.
* **Captura y gestión de metadatos.-** al incorporar contenidos al repositorio deben ser descritos y ordenados para facilitar su recuperación mediante identificador, titulo, descriptivo, fechas, palabras clave, entre otras.
* **Control de versiones.-** brinda información de cambios realizados en algún elemento contenido en el repositorio, se puede elegir crear una nueva versión y a su vez mantener la anterior o anteriores versiones
* **Bloqueo ítems de edición.-** trata de evitar la modificación de un elemento por múltiples usuarios a la vez, de esta manera evitar que los cambios realizados por un usuario sean sobrescritos por otros.
* **Auditoria y edición de accesos.-** realizar un control de los cambios que se hayan hecho en el repositorio, quién los realizo, y por qué los realizo.
* **Difusión y Distribución.-** los contenidos obtenidos mediante los procesos anteriormente mencionados generan y mantienen un repositorio para la organización, los cuales serán prestados como servicios y dichos servicios serán distribuidos como información, este proceso contiene las siguientes actividades.
* **Control de acceso según permisos.-** es importante brindar seguridad para que usuarios no autorizados accedan a la información que no se les permite el acceso
* **Mantener canales de distribución.-** los diferentes servicios que precisan ser leídos y obtener datos del repositorio actúan como canales de distribución.
* **Publicación en sitios Web.-** los sitios Web Internet e intranetson el principal mecanismo utilizado para el acceso y visibilidad a los contenidos del repositorio.
* **Publicación en sitios Web sociales.-** los sitios Web sociales tipo Facebook, Twitter, LinkedIn, etc., forman uno de los canales de distribución a los que debe atender la estrategia de la gestión de contenidos, ya que son herramientas claves para dar a conocer los contenidos de una organización. (Eiton Brun R, 2013, pp 28-33)

Existen varias opciones para crear un proyecto Web, este puede ser desarrollado desde cero o utilizando un entorno para crear una estructura que permita gestionar el contenido. Los gestores de contenidos brindan una interfaz que ayuda a controlar el sitio Web de forma fácil, segura y rápida.

## Inyección de Contenido

Las tecnologías que integran las aplicaciones de gestión de contenido son aquellas que permiten fácilmente integrar los datos que proceden de aplicaciones Web o fuentes externas.

Mientras la Web va creciendo, cada vez se torna más complejo gestionar contenido para los usuarios, teniendo en cuenta lo dicho anteriormente se puede decir que es una de las razones por la cual diferentes organizaciones buscan proveedores de contenido externos que brinden contenidos en forma de resúmenes, artículos, multimedia, etc., en un formato que les permita integrar el contenido necesario hacia su sitio Web.

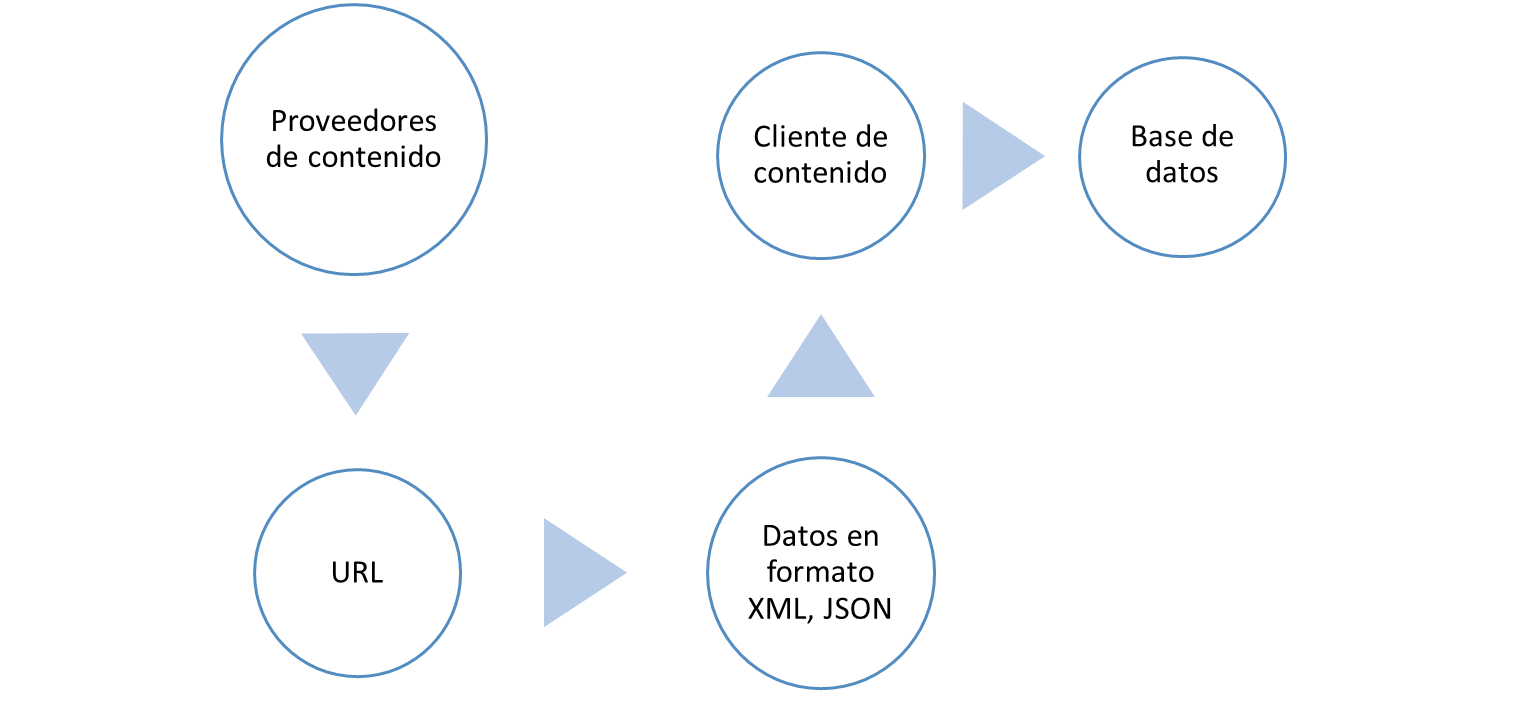
Para poder integrar contenido se necesita normalizar formatos que permitan transferir información mediante la red y tener un mecanismo que facilite a la empresa cliente obtener contenido de su interés. Al hablar de normalización de formatos se lo relaciona con el uso de eXtensible Markup Language o más comúnmente conocido por sus siglas XML el cual es una representación común para el intercambio de mensajes entre servidores y clientes, pero además se puede utilizar JSON para realizar lo mismo ya que es un formato basado en texto y que cualquier persona puede leer además de ser usado para intercambiar datos entre clientes y servidores. Una de las razones que justifica la adopción de JSON es que los objetos JSON son fáciles de manipular usando lenguajes de scripting.

Mediante la estructura de XML y JSON se define la estructura que se debe cumplir para poder transmitir contenidos y puedan ser utilizados fácilmente por la empresa consumidora y sean integrados a los diferentes sitios Web.

Una vez normalizados se debe contar con un mecanismo de descarga de contenidos, un mecanismo que brinda facilidades de acceder a funciones disponibles en un servidor Web remoto es la URL (Uniform Resource Locator) la cual devuelve un documento XML o un objeto JSON con los datos obtenidos después de realizar la ejecución de las funciones, las cuales son parte del servidor remoto.

La manera en que se ejecuta un servicio Web siempre es a través de una URL, pudiendo agregar a la solicitud un conjunto de parámetros que el servidor remoto usa para ejecutar el servicio.

Es necesario señalar que una aplicación de gestión de contenidos no debe estar forzadamente vinculada a una base de datos XML o JSON, la posibilidad de poder conseguir tanto documentos XML como objetos JSON y posteriormente procesarlos de una forma fácil para cualquier fin, ofrece una excelente flexibilidad y permite realizar cualquier tipo de integración como por ejemplo inyectar contenido en una base de datos. (Eito Brun R, 2013, http://www.upf.edu). En la Figura 2-1 se presenta el proceso que tiene la inyección de contenido.



**Figura 2-1.** Proceso de Inyección de Contenido

**Fuente:** Villa E., Rodriguez E., 2015.

## Sistemas de Gestión de Contenidos (CMS)

Desarrollar una Web a veces tiende a ser un trabajo complejo si no se disponen de herramientas adecuadas. En la actualidad se ha desarrollado el concepto de Sistema de Gestión de Contenido (CMS o Content Management System), estas herramientas permiten crear y dar mantenimiento a un sitio Web permitiendo ahorrar tiempo a los administradores de las Webs.

Los CMS ofrecen mayor facilidad de creación, actualización, y mantenimiento de una página o aplicación Web mediante una comunidad de usuarios que colaboran y participan de manera organizada.

### Breve Historia de los CMS

Históricamente nacen en los años 70 en el entorno de los mainframes, pero es hasta el año 1984 en donde se conocen los primeros sistemas que permiten compartir contenido desde un computador personal, y ya desde 1995 se habla de gestión de contenido en y para la Web.

Debido a su evolución y su reciente popularización gracias a herramientas Open Source el concepto de CMS ha ido ganando más seguidores cada día. (Serrano Cobos J, 2007, http://www.academia.edu/)

Uno de los primero CMS de código abierto fue Typo 3, que inicio a desarrollarse en el año 1997, tres años después en el año 2000 se empezó a desarrollar una de las herramientas que popularizo el uso de sistemas para comunidades de usuarios en Internet, esta fue conocida como PHPNuke (Cuerda Garcia X, 2004, http://mosaic.uoc.edu).

En la actualidad el concepto de CMS ha evolucionado, no es lo mismo una aplicación de hace 10 años que una aplicación de ahora.

### Definición de un CMS

Son las Siglas de Content Management System que se traduce al español como Sistema de Gestión de Contenidos, como su nombre lo indica es un sistema que permite gestionar o administrar contenidos.

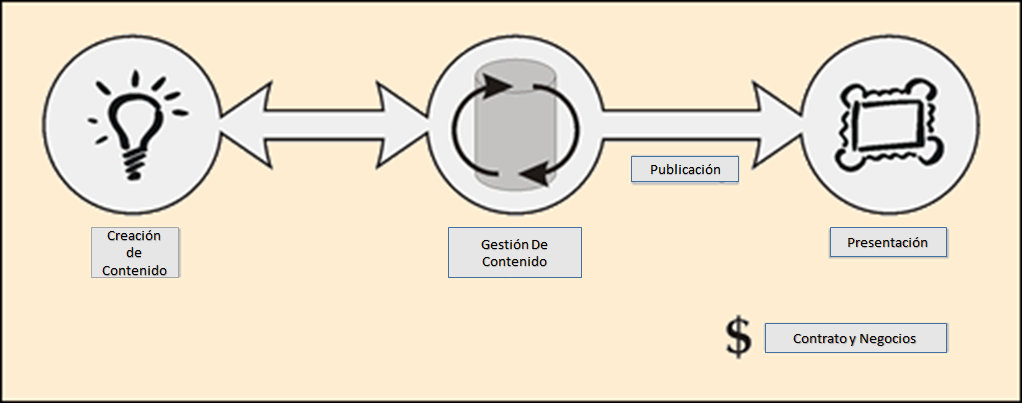
Jesús Niño Camazón (2010) da una definición de gestores de contenido, la cual dice “los gestores de contenidos son sistemas (software) que una vez instalados en el servidor nos permiten crear sitios Web; y solo hay que preocuparse de añadir textos, archivos, imágenes, etc., sin que sea necesario ser un experto en el sistema HTML para crear y administrar los contenidos de una Web”. (Niño J, 2010, pp 56,57)

Jesús Tramullas (2005) define a los gestores de contenidos como “herramientas que ofrecen las prestaciones necesarias para crear, editar, evaluar, publicar y archivar contenidos digitales, en el marco de un flujo de trabajo, y para administrar el ciclo de vida de la información digital”. (Tramullas J, Garrido P, Sanchez I, 2005, p.1295)

Tomando en cuenta una última definición tenemos la de Luis Enrique Baltazar Paz (2006), “Un CMS es un software que se encuentra en un servidor Web al que se le incorporan, comúnmente, una o varias bases de datos, lo que permite al desarrollador crear, administrar y actualizar páginas o sitios Web de manera simple y dinámica, a través de interfaces gráficas y plantillas de diseño predefinidas”. (Baltazar Paz L, 2008, http://www.enterate.unam.mx)

Entonces los gestores de contenidos facilitan la creación, la administración, la actualización, y el mantenimiento de sitios Web, pudiendo ser dicho sitio muy simple o considerablemente complejo.

James Robertson da a conocer una división de las funciones de los sistemas de gestión de contenidos, siendo estas las siguientes: creación de contenido, gestión de contenido, publicación y presentación. (Cuerda Garcia X, 2004, http://mosaic.uoc.edu). En la Figura 3-1 se presenta la anatomía de un sistema de gestión de contenidos.



**Figura 3-1.** Anatomía de un Sistema de Gestión de Contenidos

**Fuente:** http://www.steptwo.com.au/wp-content/uploads/cmsdiagram1.gif

**Creación de Contenido.-** un CMS ofrece varias herramientas mediante las cuales los creadores de contenido, sin necesidad de muchos conocimientos técnicos puedan utilizar diariamente según su necesidad, un editor de texto en el cual el usuario puede ver el resultado final que obtendrá mientras va generando su contenido.

Además los CMS ofrecen herramientas que permiten definir la estructura que debe tener un sitio Web, tomando en cuenta el formato de las páginas y su interfaz final.

**Gestión de Contenido.-** los contenidos creados se guardan en una base de datos centralizada, en la cual además se puede almacenar datos como: versiones, fechas, autor, etc., además permite modificaciones lo cual es indispensable para poder facilitar el tiempo de trabajo que inicia desde el autor y termina en la persona encargada de publicar el contenido final.

**Publicación.-** las páginas aceptadas se publican cuando llega su fecha de publicación siguiendo el patrón definido para todo el sitio Web, esto permite a los autores liberarse de tener que editar nuevamente la interfaz final de las páginas a publicarse.

**Presentación.-** un CMS puede administrar el acceso hacia una Web, adaptarse a las necesidades los usuarios y permitir mostrar el contenido en diferentes navegadores Web utilizados en todas las plataformas (Windows, Linux, Mac, etc.). (Cuerda Garcia X, 2004, http://mosaic.uoc.edu)

### Estructura y características

Generalmente los CMS tienen la siguiente estructura:

* **Base de Datos.-** Almacena todo el contenido del sitio Web, además de los parámetros de conFiguración, organización, usuarios, contraseñas**.** Normalmente los sistemas de base de datos en los CMS suelen ser Mysql o Postgresql.
* **Programación.-** en el gestor de contenidos sus ficheros están escritos en un lenguaje de programación, mediante estos ficheros se solicita información pedida por el usuario para visualizarla ordenadamente, el lenguaje más habitual es el PHP.
* **Diseño.-** define la maqueta o el dibujo de sobre el cual se colocara el contenido. Los lenguajes de programación y maquetación Webs son el HTML y CSS, complementados a veces con el JavaScript y AJAX. Normalmente tienen un diseño básico y disponen de plantillas de diversos diseños.

Las principales características de un CMS son:

* Permite interactuar con los usuarios mediante comentarios, encuestas, votaciones, etc
* La edición de los contenidos es sencilla, gracias a editores WYSIWYG (en inglés What You See Is What You Get, traducido al español “lo que vez es lo que obtienes”)
* Se puede programar fecha de publicación de los artículos
* El contenido se almacena en la base de datos y al generarse dinámicamente las páginas en el servidor el coste de gestión se reduce considerablemente
* Creación ilimitada de usuarios y grupos, se les puede asignar permisos a los usuarios y a los grupos para realizar funciones
* La maquetación solo se realiza en un primer momento, no hay que preocuparse del diseño cada vez que se va a crear un artículo
* Los gestores pueden hacer copias de seguridad y restaurarlas
* Miles de plantillas de diseño y plugins disponibles en Internet. (Niño J,2010, pp 56,57)

### Tipos de CMS

Generalmente los CMS tienen la siguiente estructura:

* **Base de Datos.-** Almacena todo el contenido del sitio Web, además de los parámetros de conFiguración, organización, usuarios, contraseñas**.** Normalmente los sistemas de base de datos en los CMS suelen ser Mysql o Postgresql.
* **Programación.-** en el gestor de contenidos sus ficheros están escritos en un lenguaje de programación, mediante estos ficheros se solicita información pedida por el usuario para visualizarla ordenadamente, el lenguaje más habitual es el PHP.
* **Diseño.-** define la maqueta o el dibujo de sobre el cual se colocara el contenido. Los lenguajes de programación y maquetación Webs son el HTML y CSS, complementados a veces con el JavaScript y AJAX. Normalmente tienen un diseño básico y disponen de plantillas de diversos diseños.

Las principales características de un CMS son:

* Permite interactuar con los usuarios mediante comentarios, encuestas, votaciones, etc
* La edición de los contenidos es sencilla, gracias a editores WYSIWYG (en inglés What You See Is What You Get, traducido al español “lo que vez es lo que obtienes”)
* Se puede programar fecha de publicación de los artículos
* El contenido se almacena en la base de datos y al generarse dinámicamente las páginas en el servidor el coste de gestión se reduce considerablemente
* Creación ilimitada de usuarios y grupos, se les puede asignar permisos a los usuarios y a los grupos para realizar funciones
* La maquetación solo se realiza en un primer momento, no hay que preocuparse del diseño cada vez que se va a crear un artículo
* Los gestores pueden hacer copias de seguridad y restaurarlas
* Miles de plantillas de diseño y plugins disponibles en Internet. (Niño J,2010, pp 56,57)

### Ventajas y Desventajas de los CMS

**Ventajas**

* No es necesario saber HTML y FTP para actualizar el contenido
* Cuando ya se sabe que CMS se va usar, ya se puede conFigurar los sitios Web rápidamente
* Las herramientas Web 2.0 son soportadas (muchas veces para funcionalidades adicionales) como Blogs, galería de fotos, calendarios, carritos de compras y muchos otras tecnologías
* El contenido está en una base de datos, por lo cual es portable para otros sistemas
* Posibilidad de cambiar el diseño y la personalización de la página Web de manera instantánea gracias a las plantillas o temas
* Cada CMS tiene su propia comunidad de usuarios, en la cual se puede resolver dudas fácilmente gracias a las aportaciones que estos mismos hacen

**Desventajas**

* Hospedaje Web más específico, los respaldos son importantes para tener la versión correcta de PHP y MySql así como el servidor Web
* Ralentización excesiva de la velocidad de carga, ya que tiene que recorrer mayor número de procesos
* Vulnerabilidades o agujeros conocidos que contribuyen a la ayuda de los hackers, por lo tanto menor seguridad.(Hernández J, 2014, https://books.google.com.ec)

## VCMS

En la actualidad, la mayoría de personas saben que un video es una herramienta muy amplia, la cual permite acelerar el aprendizaje, facilitar las comunicaciones, reducir costos, aumentar la productividad, etc., es por esto que debido a la demanda de los usuarios se van creando gestores de contenido, en este caso particular de estudio se toma en cuenta a los gestores de videos ya que muchas organizaciones están viendo un rápido aumento en la utilización de medios digitales para mejorar sus servicios.

Panopto es una compañía de software la cual brinda un sistema de gestión de contenidos de video que ha sido reconocido como líder en el “Gartner’s Magic Quadrant for Enterprise Video Content Management”

El cofundador y director de producto de Panopto Eric Burns dijo "En un futuro no muy lejano, el vídeo será un medio de comunicación de la empresa como el correo electrónico es hoy". De acuerdo con Gartner, los líderes en este mercado han desarrollado, productos extensibles flexibles que son eficaces en una variedad de casos de uso. En la Figura 4-1 se detalla el “cuadrante mágico” de Gartner sobre la gestión de contenido.



**Figura 4-1.** Cuadrante mágico de Gartner, La Gestión de Contenido

**Fuente:** (Panopto Blog, 2015, www.panopto.com/)

En la actualidad el video se convierte en una herramienta muy importante para la comunicación y la productividad, por lo cual las organizaciones deben descubrir una forma de manejar adecuadamente enormes cantidades de video y poder compartirlas mediante sus redes.

### Definición de un Sistema de Gestión de Contenidos de vídeo (VCMS)

VCMS son las siglas de Video Content Management System que se traduce al español como Sistema de Gestión de Contenidos de Video. Por lo tanto un VCMS es “un software que permite a una organización centralizar, administrar y entregar vídeo online.”(Panopto, 2015, http://panopto.com/blog)

Gartner define a los VCMS como accesorios o servicios de software (SaaS) destinados a gestionar y facilitar la entrega de uno-a-cualquier video bajo demanda a través de protocolos de Internet. (Panopto, 2014, pp 6,7).

Básicamente, un VCMS proporciona un repositorio de contenido de vídeo, en el que los archivos de vídeo se cargan en el sistema y son accesibles a las personas a través de un enlace.

En los últimos años, los VCMS han nacido como una nueva solución para las empresas, diseñados para gestionar y presentar videos de dichas organizaciones. Un VCMS proporciona una infraestructura económica y además utiliza los videos como un activo para estimular al crecimiento de la organización.

Al mismo tiempo de brindar un repositorio de videos, la mayoría de VCMS ofrece las siguientes características, detalladas en la Tabla 1-1.

**Tabla 1-1.** Características de los VCMS

|  |  |
| --- | --- |
| **Características** |  |
| **Upload** | La subida de vídeos pregrabados con soporte para una amplia gama de codecs y extensiones de archivo. Soporte de cargas mayores o basadas en API simplifica la ingestión de grandes videotecas. |
| **Capture** | El software de cliente que permite la grabación de vídeos y capturas de escritorio usando una gama de dispositivos de captura de video (cámaras Web USB, las cámaras de vídeo profesionales, fuentes de captura de HDMI, etc.). Una vez registrados, los vídeos se cargan automáticamente en el VCMS |
| **Transcoding** | La conversión de archivos de vídeo en una variedad de formatos para asegurar la compatibilidad con dispositivos móviles (por ejemplo, conversión de vídeo basado en Flash a MP4 para su reproducción en teléfonos inteligentes). |
| **Streaming** | En vivo y bajo demanda siendo transmitido en streaming de video con la capacidad de controlar el ancho de banda de los contenidos. |
| **Search** | La capacidad de buscar a través de las bibliotecas masivas de vídeo |
| **Editing** | Recorte de contenido de vídeo, combinar y remezclar vídeos, realizar cambios en las transcripciones de vídeo, edición de metadatos. |
| **Analytics** | Los informes que permiten conocer el consumo de ancho de banda, el comportamiento de visualización, realización de video y los porcentajes de abandono, de contenido popular, y la salud del sistema. |
| **Security** | Autenticación y autorización al contenido dentro del sistema de gestión de contenido de vídeo, el cifrado de la comunicación entre los clientes y los VCMS e inicio de sesión único con proveedores de identidad. |
| **Integration** | Por API o la conectividad basada en widgets a los sistemas de gestión de aprendizaje corporativa, sistemas tradicionales de gestión de contenidos como SharePoint y sistemas de gestión de relaciones con clientes (CRM) |
| **Hosting** | Opciones para instalar el VCMS detrás de su firewall en los servidores de TI o de acogida en la nube en los servidores del proveedor proporcionada. |
| **Playback** | Reproductores de vídeo basado en la Web interactivas que permiten a los espectadores a navegar dentro del video, tomar notas, la búsqueda interior de contenido de vídeo y vídeos de tasas. Podcasts de vídeo también deben estar disponibles para incrustar en páginas Web existentes. |
| **Movile** | Podcasts de vídeo compatibles con HTML5 para su visualización en tabletas y teléfonos inteligentes, así como aplicaciones móviles nativas para grabar y compartir video móvil desde dispositivos iOS y Android. |

**Fuente:** (Panopto White-Paper, 2015, pp. 6,7; *www.*panopto.com)

**Realizado por:** Villa E., Rodriguez E., 2015.

## JOOMLA



**Figura 5-1.** Logotipo de Joomla

**Fuente:** (Joomla, 2015, www.joomla.org)

Joomla es el gestor de contenidos líder en la creación de sitios Web, Joomla permite gestionar con mucha facilidad todo un sitio Web, crear un nuevo contenido, actualizar los existentes, añadir imágenes, crear nuevos menús, sin la necesidad de tener conocimientos técnicos, solamente conocer Word o editores de texto similares se puede administrar un sitio Web.

En todo el mundo existen alrededor de 30 millones de sitios Web creados con Joomla, teniendo a disposición más de 10 000 de componentes que permiten ampliar las funcionalidades de los diferentes sitios Web pudiendo así convertirse en tiendas virtuales, foros, galería de imágenes, y un sinfín de posibilidades más.

Joomla es el gestor de contenidos líder en la creación de sitios Web, Joomla permite gestionar con mucha facilidad todo un sitio Web, crear un nuevo contenido, actualizar los existentes, añadir imágenes, crear nuevos menús, sin la necesidad de tener conocimientos técnicos, solamente conocer Word o editores de texto similares se puede administrar un sitio Web.

En todo el mundo existen alrededor de 30 millones de sitios Web creados con Joomla, teniendo a disposición más de 10 000 de componentes que permiten ampliar las funcionalidades de los diferentes sitios Web pudiendo así convertirse en tiendas virtuales, foros, galería de imágenes, y un sinfín de posibilidades más.

### Definición

Es muy difícil poder sintetizar una definición exacta de Joomla, para esto se da a conocer varias definiciones individuales que en conjunto definen de mejor forma a Joomla.

* Joomla es un gestor de contenidos Web (Web CMS) de código abierto potente, flexible y fácil de usar, que permite crear virtualmente cualquier tipo de sitio Web
* Joomla es un Framework de desarrollo PHP que permite crear potentes aplicaciones Web de cualquier tipo
* Joomla es un Proyecto de software libre respaldado únicamente por una comunidad de desarrolladores y usuarios a lo largo y ancho de todo el mundo (Baquero I & Cámara M,2014, http://manualesjoomla.es)

Entonces Joomla al ser un proyecto de software libre, está respaldado únicamente por una comunidad de usuarios y desarrolladores, esta comunidad ha logrado desarrollar un CMS potente, flexible y de fácil utilización para poder crear sitios Web, además de ser un Framework de PHP que permite crear aplicaciones Web.

### Servicios de Joomla

Se lo utiliza para crear desde una página Web simple hasta aplicaciones Web más complejas como por ejemplo:

* Webs corporativas
* Web de Organizaciones o ONGs
* Comercio Electrónico
* Webs de Instituciones
* Canales de Video
* Revistas y Periódicos

En la Figura 6-1 se presenta los diferentes servicios que proporciona Joomla.



**Figura 6-1.** Servicios de Joomla

**Fuente:** Villa E., Rodriguez E., 2015.

### Datos Básicos sobre Joomla

Joomla está desarrollado usando PHP orientado a objetos, usando un patrón Modelo Vista Controlador

* Es un software capaz de usar distintos motores de base de datos: MySQL, SQL Server (Azure), PostgreSQL. Se está trabajando en otros controladores para futuras versiones
* Puede instalarse en distintos servidores Web: Apache, IIS y nginx
* Las versiones anteriores usaban Mootools como librería JavaScript. La versión 3.0 introdujo Bootstrap, y con ello jQuery, aunque aún mantenía bastantes dependencias con Mootools. La versión 3.3 prácticamente elimina todas las dependencias
* El core de Joomla es capaz de crear sitios multilingües sin necesidad de instalar extensiones adicionales, y esta traducido a 67 idiomas
* La interfaz que trae por defecto está adaptada a móviles tanto en la parte pública como en la parte administrativa (Baquero I & Cámara M,2014, http://manualesjoomla.es)

## Contus HD Video Share

Contus HD Video Share es una extensión eficiente para compartir videos en línea para Joomla, el cual proporciona soporte para Flash y HTML 5, mediante HD Video Share se puede incrustar videos de YouTube, Vimeo, Viddler y Daily Motion, esta extensión está disponible en un paquete de fácil instalación, una vez instalado permite compartir videos en un sitio Web en pocos minutos.



**Figura 7-1.** Logotipo Contus HD Video Share

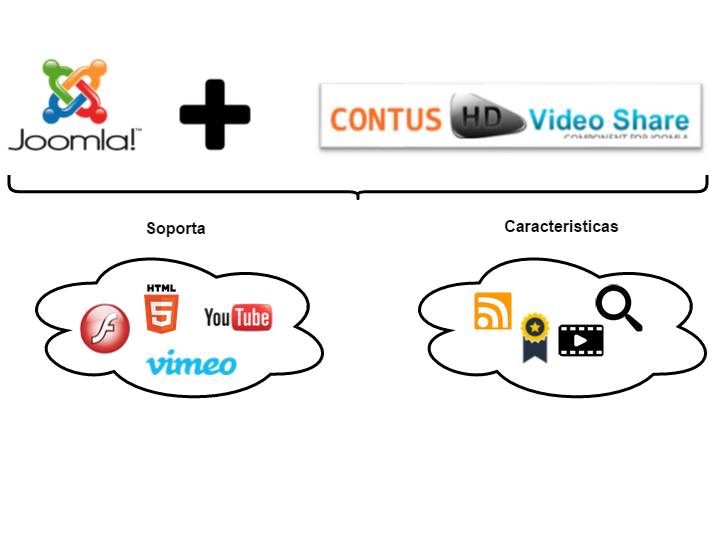
**Fuente:** (apptha, 2015, www.apptha.com)

Este componente viene con pocos módulos libres, utilizando estos módulos libres se puede se puede cambiar su sitio Web Joomla en un sitio Web para compartir videos como YouTube y Vimeo.

HD Video Share contiene los siguientes módulos:

* Buscar Videos
* Videos Destacados
* Video Categoría
* Videos Recientes
* Videos Relacionados
* Vídeos populares
* Módulo Categorías
* Módulo de RSS
* HD Video Share Player. (HD Video Share,2014, https://docs.google.com)

En la Figura 8-1 se representa las funcionalidades y características de Joomla con la integración de HD Video Share Player.



**Figura 8-1.** Funcionalidad de Joomla y Contus Video Share

**Fuente:** Villa E., Rodriguez E., 2015

## Marketing Online

Se puede definir como el conjunto de técnicas del tradicional marketing re direccionadas hacia el Internet, con el objetivo principal de aprovechar las oportunidades que brinda, de esta manera potenciar una marca o negocio de manera rápida, económica y eficiente, llegando a si a los potenciales clientes. (Big Eye, 2015, http://www.bigeyemarketing.com)

Para poder lograr este objetivo el marketing online debe tener las siguientes técnicas.

* Posicionamiento en Buscadores (SEO, SEM)
* Social Media Marketing. (SMO, SMM)
* Redes de afiliados
* E-mail Marketing
* Diseño y desarrollo Web (Big Eye, 2015, http://www.bigeyemarketing.com)

En el presente trabajo de titulación se involucra principalmente a una de estas técnicas, el marketing en redes sociales o social media marketing, por lo tanto se amplian los conceptos y aplicaciones en los siguientes capitulos.

### Marketing en Redes Sociales (SMO-SMM)

En la actualidad las redes sociales se han convertido en una herramienta necesaria para cualquier tipo de empresa, no importa su ámbito o tamaño, que tenga la necesidad de dar a conocer sus servicios de una manera más eficiente y económica. Las redes sociales hoy en día a más de ser un medio de comunicación entre personas es considerado como un medio publicitario importante, debido a su crecimiento en todo el mundo se han convertido en una herramienta clave de publicidad. (Big Eye, 2015, http://www.bigeyemarketing.com)

#### Social Media Optimization (SMO)

Esta técnica surge a partir de la aparición de la Web 2.0 con el objetivo de incrementar la interacción y viralización de los contenidos en los medios sociales de esta manera aumentar la visibilidad de un sitio Web. La diferencia principal de esta técnica comparada con las otras técnicas de marketing online, no va dirigida a los motores de búsqueda si no a los medios sociales.

Una de las estrategias que utiliza esta técnica es, poseer contenido de calidad, fácil y sencillo de compartir por el usuario, medios disponibles para compartir en redes sociales (opciones para compartir en Twitter Facebook, RSS, etc.), con el objetivo de conseguir tráfico a nuestro sitio Web, a partir de la recomendación de los propios usuarios, llegando así a generar publicidad viral. (Baquero I & Cámara M,2014, http://manualesjoomla.es)

#### Social Media Marketing (SMM)

Luego de usar un SMO, se debe tomar mecanismos para la distribución del contenido, para lograr esto se debe humanizar la empresa. Esto hace referencia a crear perfiles en redes sociales donde se encuentren los potenciales usuarios. Finalmente se debe tener estrategias para redes sociales.

#### Objetivos

* Comunicar contenido a los usuarios
* “Viralizar” (Expansión acelerada para producir incremento en el reconocimiento de una marca) el contenido que ofrece la empresa
* Trabajar en red con personas y empresas afines
* Crear una comunidad de seguidores
* Incentivar a los clientes, para que participen en el desarrollo de los productos y del negocio en general
* Ser una referencia en el sector para lograr un posicionamiento de un negocio
* Obtener información actualizada (desde el Internet) del concepto y aceptación del negocio (Panadero J,2012, p.6)

#### Ventajas

* Atraer tráfico al sitio Web de forma rápida
* Si el contenido a compartir es interesante para el público, la difusión resultara fácil
* Si se logra que el contenido se viralice, los resultados mejoraran notablemente
* El costo de inversión es mínima, comparado a otros medios de publicidad
* Gran alcance
* Posibilidad de segmentación

#### Desventajas

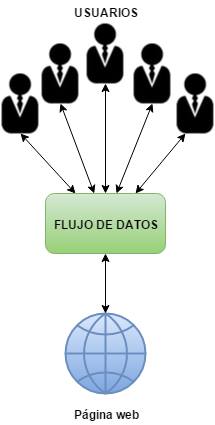
* Los resultados se consiguen a corto plazo, pero si no se lo realiza habitualmente el resultado no durara mucho tiempo
* Dependen exclusivamente de las redes sociales y de la difusión de otros usuarios
* Una publicación puede ser considerada como intrusiva, generalmente conocida como spam (Redes Pymes, 2014, http://redespymes.com)

## Tráfico Web

Una página Web a pesar de encontrarse desarrollada de manera óptima con un buen diseño, atractivo, con contenido de calidad si no tiene visitantes no será útil, por lo tanto se debe analizar si una página genera rentabilidad, para esto se debe analizar su tráfico Web.

Se considera como tráfico Web a la interacción que realiza el usuario con una página Web generando flujo de datos entre los operadores, por lo tanto el tráfico Web es determinado a partir del número de visitantes y las paginas internas que ingresa dentro del sitio Web. El tráfico Web es considerado como una de las variables más importantes dentro de las estrategias de marketing online, ya que a partir de ella se puede analizar el estado del negocio para mejorar el plan de marketing. (Codina L,2007,http://eprints.rclis.org)

En la Figura 9-1 se observa una representación del tráfico Web.



**Figura 9-1.** Trafico Web

**Fuente:** Villa E., Rodriguez E., 2015

### Fuentes de tráfico

Se considera fuente de tráfico a todo elemento externo que genere mínimo una visita hacia el sitio Web, entre los más considerables se tiene.

* Tráfico directo. Su visita la realiza de manera directa digitando la Url en el navegador, o lo tiene registrado como favorito
* Tráfico de búsqueda. Su vista proviene de la búsqueda de una palabra clave en los buscadores (Google, Bing, Yahoo)
* Tráfico de referencia. Trafico proveniente a partir de un sitio externo que de algún modo lo recomienda
* Tráfico de campaña. Tráfico proveniente publicidad pagada
* Tráfico desde redes sociales. Se puede considerar como tráfico de referencia, con la diferencia de que este tipo de tráfico puede ser viral (Argiles S, 2013 ,http://sergioargiles.com)

## Frameworks

Hoy en día el uso de Framework para el desarrollo de aplicaciones Web no es ninguna novedad, debido a que las aplicaciones deben cumplir estrictos niveles de calidad, seguridad y escalabilidad, además la necesidad de contar con la aplicación en corto tiempo, todo esto conlleva a hacer uso de los Framework de desarrollo, los cuales aportan de manera significativa una solución a los problemas presentados.

### Definición

Para dar una definición correcta se ha estudiado varias fuentes sobre el tema, las más importantes son las siguientes:

Se considera Framework a una estructura definida, reusable donde los componentes que lo componen facilitan la creación de aplicaciones Web, además provee una capa de abstracción sobre la arquitectura original. (Degiovannini M, 2007, http://static1.1.sqspcdn.com/ )

Javier J. Gutiérrez considera que un Framework es una estructura conjunta de componentes los cuales pueden ser personalizados e intercambiables, también se lo puede considerar como una aplicación genérica incompleta, la cual puede ser conFigurable de acuerdo a la necesidad del proyecto. (Gutiérrez J, 2005, pp. 1,2)

En conclusión para el presente trabajo se define como Framework a una estructura de software compuesta por componentes los cuales pueden ser conFigurables, conformando así el esqueleto principal de la aplicación, a la cual se debe ir agregando los componentes necesarios de acuerdo a la solución a desarrollar. La gran mayoría de Frameworks de desarrollo Web trabajan con la arquitectura Model View Controller (MVC). En la Figura 10-1 se representa la interpretación de un Framework.

C:\Users\Alex\Downloads\fra.png

**Figura 10-1.** Interpretación de Framework

**Fuente:** Villa E., Rodriguez E., 2015

### Características

Entre las características que presentan la mayoría de Framework son las siguientes.

* Abstracción de Urls y sesiones, el Framework realiza el trabajo de manejar las urls y las sesiones
* Acceso a datos, incluyen herramientas e interfaces necesarios para integrarse a herramientas de acceso a datos
* Controladores, facilitan la administración de eventos que se realizan en la aplicación
* Autenticación y control de acceso, poseen métodos de autenticación de usuario, y control de acceso a ciertas páginas dependiendo el tipo de usuario
* Internacionalización, Mecanismos que permiten que la aplicación se presente en diferentes idiomas que se considere necesarios
* Separación entre diseño y contenido, poseen herramientas en cuanto para el diseño y para su contenido

### Ventajas

* Uso de patrones de diseño. El Framework obliga a que se haga uso de ellos, evitando así el desarrollo conocido como “código espagueti” (codificación compleja e incomprensible) donde se agrega funcionalidades en capas que no corresponde
* Estructura predefinida de la aplicación. La estructura de los archivos de la aplicación ya se encuentra definida por el Framework, ahorrando tiempo para el desarrollador
* Código Altamente testeado. El código que proporcionada cada Framework se encuentra testeado por sus desarrolladores
* Comunidad de usuarios detrás de cada Framework. Existen comunidades de tras de cada uno de ellos, desarrollando en cada momento nuevas funcionalidades que se podrá utilizar sin perder el tiempo desarrollándolas personalmente
* Trabajo en equipo. Facilita el trabajo en equipo debido a que se conoce la estructura de los archivos (AcensTechnologies, 2014, pp. 4-6)

### Desventajas

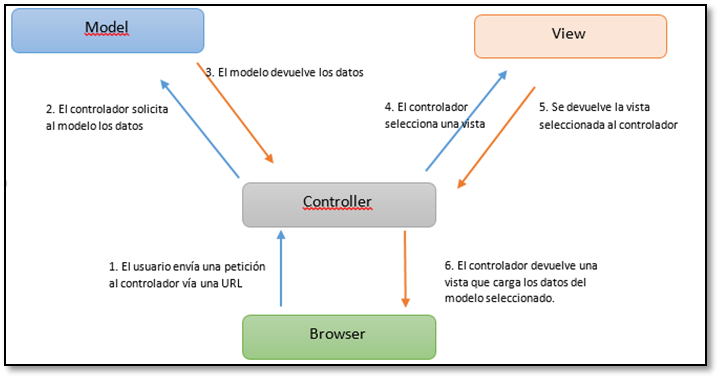
* Tiempo de aprendizaje. En situaciones algunos de los Framework poseen una curva de aprendizaje muy elevada
* Exceso de líneas de código. Algunos Framework pueden contener exceso de líneas de código para llevar acabo sus operaciones, los cuales pueden ser considerado como código basura por algunos expertos en el tema
* Limitaciones. Al momento de seleccionar un Framework existen componentes que no se pueden modificar, por lo tanto se debe elegir uno que se adapte a las posibles necesidades que presente el desarrollo de la aplicación
* Código público. El Framework al estar disponible para todo el público, puede ser estudiado por personas maliciosas y encontrar vulnerabilidades (AcensTechnologies, 2014, pp. 4-6)

## Patrón de diseño MVC

El modelo MVC es un patrón de arquitectura, desarrollado con el objetivo principal de separar la lógica del negocio con la interfaz del usuario, dividiendo el desarrollo en tres componentes principales: modelo, vista, controlador.

* **Modelo.** Representa los datos de la aplicación y las reglas de negocio
* **Vista.** Principalmente es la interfaz del usuario mediante la cual puede interactuar con el sistema
* **Controlador.** Controla las peticiones del usuario, y controla el flujo de ejecución del sistema (Degiovannini M, 2007, http://static1.1.sqspcdn.com)

En la Figura 11-1 se explica el funcionamiento del patrón de diseño MVC.



**Figura 11-1.** Patrón de Diseño MVC

**Fuente:** Villa E., Rodriguez E., 2015

## Estudio de tecnologías

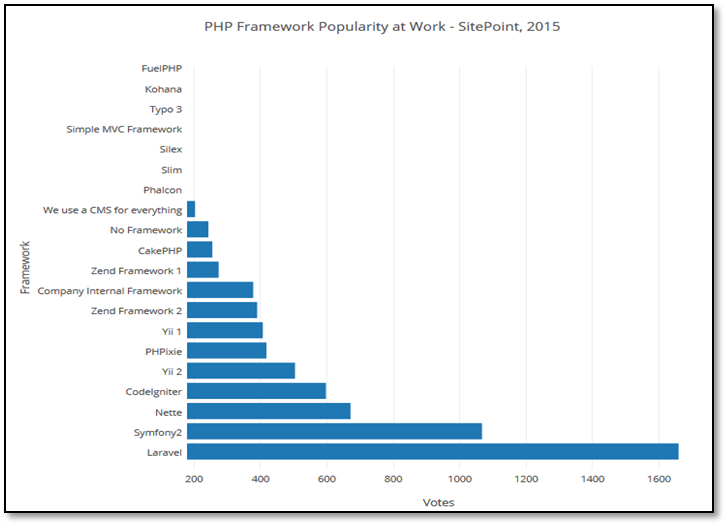
A continuación se detalla el estudio realizado para la selección del Framework para desarrollar la aplicación Web, así como el estudio de cada una de las APIs utilizadas, tal es el caso de los canales de video YouTube, y Vimeo, y en redes sociales se tiene Facebook, Twitter.

Selección de las herramientas de desarrollo

Para el siguiente trabajo de grado se ha tomado en cuenta estudiar los Framework que tienen mayor acogida por parte de los desarrolladores Web en la actualidad. Para esto se ha realizado una investigación en páginas Web relacionadas con tecnología, se ha seleccionado los tres mejores Framework, a continuación se detalla porque su selección.

En el portal sitepoint muestra los resultados obtenidos de la encuesta realizada: “Best PHP Framework for 2015 – SitePoint Survey Results”, con un total de 7800 encuestas, dando como resultado los cinco más populares, como se puede observar en la Figura 12-1. (Sitepoint, 2015, http://www.sitepoint.com)

* Laravel (1659 votos)
* Symfony2 (1067 votos)
* Nette (671 votos)
* CodeIgniter (597)
* Yii 2 (504)

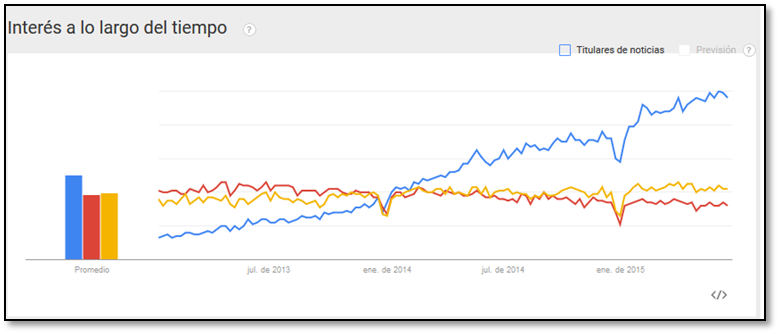


**Figura 12-1.** Encuesta Php Frameworks más popular 2015

**Fuente:** (Sitepoint, 2015, www.sitepoint.com/)

En el buscador más popular de Internet (google) la tendencia de los Frameworks php para desarrollo Web más buscados en los tres últimos años son los siguientes, como se puede observar en la Figura 13-1. (Google Trends, 2015, http://www.google.es)

* Laravel
* CakePHP
* Symfony



**Figura 13-1.** Tendencia de búsquedas de Frameworks PHP

**Fuente:** (Google Trends, 2015, www.google.es/trends)

El sitio Github, es una plataforma de desarrollo colaborativo con más de 9.9 millones de usuarios, donde se aloja repositorios de código tanto públicos como privados, utilizando el sistema de control de versiones Git. GitHub aloja un repositorio de código y brinda herramientas para el trabajo en equipo dentro de un proyecto

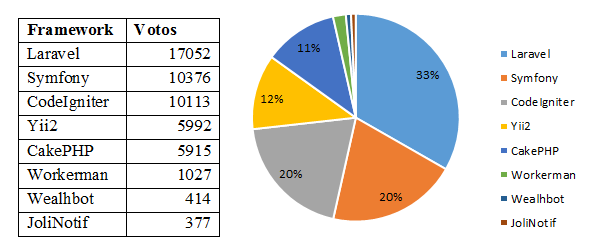
Github tiene una forma de realizar seguimiento a los proyectos que se encuentran interesantes, mediante la votación que puede ser realizada por un usuario que se encuentre o no asociado a dicho proyecto. Esta votación esta basada en estrellas, una ★ (estrella) representa un voto que da un usuario a un proyecto. Cuando se vota por un repositorio en realidad se realiza 2 acciones distintas:

* Crear un marcador para un acceso más fácil
* Mostrar agradecimiento al mantenedor del repositorio que utiliza en su trabajo

Muchos de los rankings de repositorio de GitHub dependen de la cantidad de estrellas que un repositorio tiene. Por ejemplo los repositorios se pueden clasificar y ser buscados en base a su conteo de estrellas, mediante este ranking se muestra los repositorios más populares.

Las mediciones recabadas desde el Internet, específicamente de GitHub son correspondientes a la fecha del 21 de marzo del 2015, con la participación de usuarios con el mismo conocimiento técnico, los cuales se pueden observar en la Figura 14-1.

Estos votos son uno de los parámetros considerados para la selección del Framework adecuado, además de los que se verán mas adelante.



**Figura 14-1.** Ranking de repositorios de GitHub año 2015

**Fuente:** Villa E., Rodriguez E., 2015

En el portal phpgang en el artículo: “Top 10 PHP Frameworks worth looking forward to in 2015”, presenta los diez Framework más destacados en el desarrollo de aplicaciones Web de alto rendimiento. (PHP Gang, 2015, http://www.phpgang.com)

* CakePHP 3.0
* Laravel
* Phalcon
* Symfony 2
* CodeIgniter
* Yii Framework
* Aura
* Zend
* FlightPHP
* FuelPHP

Según la página Beebon en su artículo: “15 Best Free PHP Frameworks of 2015”, da a conocer los 15 mejores Framework más destacados. (Beebon, 2015, http://beebom.com)

* Laravel
* Symfony
* CodeIgniter
* CakePHP
* Zend Framework
* Phalcon
* Slim
* Yii
* Fat-free
* Kohana
* FuelPHP
* Flight
* PHP-mini
* Simple PHP Framework
* Zikula

Para seleccionar cual es el mejor Framework para el desarrollo del trabajo de titulación se ha preseleccionado los siguientes Symfony, CakePHP, Laravel, por los siguientes motivos.

* De las referencias obtenidas en Internet los tres Framework se encuentran dentro de los cinco más populares
* Cada uno de ellos hacen uso para el desarrollo de aplicaciones Web, el patrón de diseño MVC
* Poseen herramientas para el manejo y abstracción de datos
* Los tres poseen documentación actualizada
* Comunidades de usuarios activas que se encuentran dando soporte a quien lo necesite

### Symfony

Es un completo Framework PHP de tipo full-stack Web creado con el objetivo de optimizar el desarrollo de aplicaciones Web. Gracias principalmente a que se encuentra basado en MVC como patrón de arquitectura, lo que le permite separar la lógica del negocio, los datos, y la presentación de la aplicación, desarrollando así aplicaciones fáciles de mantener, además posee varios componentes desarrollados de manera independiente por el proyecto Symfony. Este proyecto se encuentra liderado por Fabien Potencier creador del Framework y actual CEO de Sensio Labs patrocinador del proyecto. (Symfony.es, 2015, http://symfony.es)

En la actualidad el Framework se encuentra liberado bajo la licencia Open Source MIT, y además cuenta con un repositorio en una de las plataformas más grandes de desarrollo colaborativo del mundo llamada GitHub, liderado principalmente por Fabien Potencier. (Symfony.es, 2015, http://symfony.es)

La versión en la que se encuentra actualmente es Symfony 2.7 más conocida como Symfony2 desarrollada totalmente con PHP 5.3, evitando así problemas de seguridad que contraían versiones anteriores de PHP. La controversia que existe con Symfony2 es que a menudo se lo define como un Framework MVC como lo era las versiones anteriores, a diferencia de esto Symfony2 comparte conceptos de MVC, pero no en su totalidad, ya que solo proporciona herramientas para el controlador, y la vista, pero no para el modelo, esto dependerá del desarrollador, si lo realiza manualmente o hace uso de algún ORM. Su objetivo principal es otro, atender a peticiones HTTP de manera organizada y eficaz, por lo tanto se lo define como un Framework HTTP. (Potencier F, 2015, http://fabien.potencier.org)

#### Características

* Su código, como los componentes desarrollados son liberado bajo la licencia Open Sourse MIT (Massachusetts Institute of Technology)
* Su documentación, libros, tutoriales son libres
* Contiene una variedad de componentes para el desarrollo de sitios sencillos hasta aplicaciones PHP complejas
* EL proyecto Symfony es uno de los más activos en la plataforma GitHub, garantizando así su actualización continua y evitando que los usuarios se queden atrapados en un proyecto sin actividad
* Permite la creación de aplicaciones extensibles, mediante módulos (Blundles) permitiendo ser utilizados en cualquier proyecto

Posee una potente línea de comandos, lo que le permite realizar algunas tareas automáticamente, como la creación de proyectos, implementación de entidades o el borrado de cache. (Symfony.es, 2015, http://symfony.es)

### Laravel

Laravel es un Framework MVC de desarrollo Web hecho en PHP 5 por Taylor Otwell bajo la licencia MIT. Laravel fomenta el desarrollo rápido de aplicaciones. Ha sido diseñado para mejorar la calidad de software al reducir tanto el coste de desarrollo inicial como los costos de mantenimiento, y para mejorar la experiencia de trabajo al desarrollar aplicaciones, proporcionando una sintaxis clara y un conjunto básico de funciones que ahorran el tiempo de desarrollo. (McCool S, 2012, pp. 16-19)

De acuerdo a su documentación Laravel tiene todo para implementar soluciones mediante código ordenado, cortas líneas de codificación, esto permite una fácil interpretación por otro programador, evita la complejidad innecesaria y su sintaxis de codificación utiliza nombres significativos y expresivos que hacen que el código se documente así mismo. Se inició en el año 2011, y utiliza las mejoras de PHP 5.3 brindando una sintaxis clara. Laravel permite crear aplicaciones Web con muy pocas líneas de código que son fáciles de entender, incluso para un programador novato.

#### Características

* Control de ruteo
* Capacidad de RESTful
* Motor de plantillas Blade
* ORM Eloquent
* Utiliza Composer
* Soporte para cachéo
* Soporte para el patrón de diseño MVC
* Utiliza componentes de Symfony
* Licencia Open Sourse MIT
* Protección CSRF
* Poderoso administrador de extensiones (Bundles)
* Completa y concisa documentación que es muy sencilla de leer y comprender
* Comunidad creciente en el apoyo de este proyecto (Ríos D,2014, http://donaldorios.com)

### CakePHP

Es un Framework PHP 5.4 construido bajo los conceptos de Ruby on Rails con el objetivo principal de proporcionar a los desarrolladores una estructura organizada de herramientas para el desarrollo de aplicaciones robustas de manera rápida y sencilla, sin perder flexibilidad.

Sigue los fundamentos del patrón de arquitectura MVC, lo que garantiza desarrollar aplicaciones modulares y sostenibles, además de MVC posee otras clases, objetos, componentes que hacen que el desarrollo sea rápido y agradable.

En la actualidad la última versión estable es CakePHP 3.0 liberada el 22 de marzo del 2015, bajo la licencia Open Sourse MIT, la cual se encuentra sostenida por la comunidad que lleva el mismo nombre en la plataforma colaborativa GitHub, la cual se encuentra activa, garantizando así tener un Framework que se encuentra actualizando continuamente. (Cake Software Foundation, 2015, pp. 1,2)

#### Características

* Licencia flexible Open Sourse MIT que hace perfecto para uso en aplicaciones comerciales
* Comunidad y activa y amigable
* Scaffolding, técnica la cual permite crear CRUD de manera fácil, para la interacción de la base de datos
* Posee funciones para la generación de código
* Arquitectura Modelo Vista Controlador
* URLs personalizadas
* Sistema de plantillas rápidas y flexibles
* Ayudantes para AJAX, JavaScript, formularios HTML y más
* Componentes integrados para el acceso a datos, cache, validaciones, autenticación
* Contiene herramientas integradas para la validación de entrada de datos, protección CSRF, prevención de inyección SQL y XSS (Cake Software Foundation, 2015, pp. 1,2)

Para el desarrollo de la aplicación se realizará un estudio entre los Frameworks detallados anteriormente considerando algunos aspectos relevantes para el desarrollo de la solución planteada con anterioridad.

La información con la cual se llenó la Tabla 3-1 de comparación se la obtuvo desde la documentación oficial de cada Framework, para determinar el criterio de comparación sobre el número de paquetes para la integración. Se ha estudiado cada uno de los repositorios oficiales de cada Framework considerando las APIs a utilizar en el desarrollo, en la Tabla 2-1 se observa el resultado de la cantidad de repositorios por Framework.

**Tabla 2-1.** Cantidad de Paquetes de integración.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Framework** | **Repositorio oficial** | **YouTube** | **Vimeo** | **Facebook** | **Twitter** | **Total** |
| Symfony | Knp Bundles | 2 | 2 | 22 | 39 | 65 |
| Laravel | Packalyst | 13 | 10 | 50 | 50 | 123 |
| CakePHP | Plugins y packeges cakePHP.org | 3 | 2 | 10 | 20 | 35 |

**Fuente:** (KnpLabs, 2012), (CakePackages, 2015), (Packalyst, 2015)

**Realizado por:** Villa E., Rodriguez E., 2015.

### Comparativa de Frameworks

**Tabla 3-1.** Comparación de Frameworks

|  |  |  |  |
| --- | --- | --- | --- |
| **Característica/Framework** | **Symfony** | **Laravel** | **CakePHP** |
| Soporte PHP5 | SI | SI | SI |
| Licencia | MIT | MIT | MIT |
| Soporte para el patrón de arquitectura MVC | SI | SI | SI |
| Soporte para múltiples bases de datos | SI | SI | SI |
| Mapeo Objeto relacional (ORM) | SI | SI | SI |
| Autenticación de usuario incluida | SI | SI | SI |
| Manejo de caché | SI | SI | SI |
| Validaciones incluidas | SI | SI | SI |
| Motor de plantillas | SI | SI | SI |
| Integración de AJAX | SI | SI | SI |
| Operaciones CRUD | SI | SI | SI |
| Paquetes de integración | 65 | 123 | 35 |
| Popularidad del repositorio en GitHub | 10.376 | 17. 052 | 5915 |

**Realizado por:** Villa E., Rodriguez E., 2015.

De los tres Framework considerados para el estudio se puede evidenciar que todos cumplen con los criterios considerados para el desarrollo de la aplicación, sin embargo se considera de mayor prioridad para el presente trabajo de titulación los criterios como: paquetes de integración, popularidad del repositorio en GitHub. De los cuales se puede apreciar que Laravel sobresale a los demás, por lo tanto la aplicación que se seleccionara para el desarrollo de la aplicación es Laravel.

## Laravel

Laravel es un Framework de código abierto para el desarrollo de aplicaciones Web en PHP 5 que posee una sintaxis simple y de fácil comprención.

Características:

* Creado en 2011 por Taylor Otwell
* Está inspirado en Ruby and Rail y Symfony, de quien posee muchas dependencias
* Está diseñado para desarrollar bajo el patrón MVC
* Posee un sistema de mapeo de datos relacional llamado Eloquent ORM
* Utiliza un sistema de procesamiento de plantillas llamado Blade, el cual hace uso de la cache para darle mayor velocidad

### Filosofía de Laravel

Laravel es un Framework de aplicaciones Web con sintaxis expresiva. El desarrollo debe ser rápido, fácil y comprensible. Laravel intenta tomar las molestias del desarrollo facilitando las tareas comunes que tienen la mayoría de proyectos Web, como la autenticación, enrutamientos, sesiones, y caché. (Laravel, 2015, http://laravel.com/docs)

### Requisitos del servidor

El framework Laravel tiene pocos requerimientos del sistema

* PHP >= 5.4
* MCrypt PHP Extension

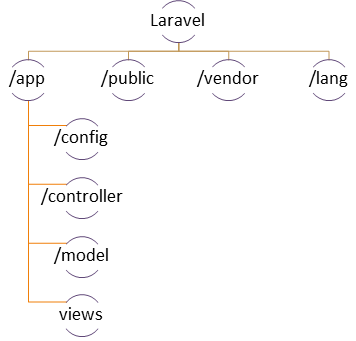
A partir de PHP 5.5, algunas distribuciones de sistemas operativos pueden requerir que instalar manualmente la extensión PHP JSON.

### Estructura

La estructura de Laravel de un proyecto contiene los siguientes directorios. (Codehero, 2013, <http://codehero.co>)

* **/app.-** contiene los controladores, modelos, vistas y conFiguraciones de la aplicación. En carpeta contiene la mayoría del código para que una aplicación funcione.
* **/public.-** es la única carpeta a la que los usuarios de la aplicación pueden acceder. Todas las peticiones y solicitudes a la aplicación pasan por esta carpeta, ya que en ella se encuentra el index.php, este archivo es el que inicia todo el proceso de ejecución del framework.
* **/vendor.-** en esta carpeta se alojan todas las librerías que conforman el framework y sus dependencias.
* **/lang.-** en esta carpeta se guardan archivos PHP que contienen Arrays con los textos de diferentes lenguajes, para poder traducir la aplicación.
* **/app/config.-** En esta carpeta se guardan archivos PHP que contienen Arrays con los textos de diferentes lenguajes, para poder traducir la aplicación.
* **/app/controller.-** en esta carpeta se guardan archivos PHP que contienen Arrays con los textos de diferentes lenguajes, para poder traducir la aplicación.
* **/app/model.-** en esta carpeta se guardan archivos PHP que contienen Arrays con los textos de diferentes lenguajes, para poder traducir la aplicación.
* **/app/views.-** en esta carpeta se guardan archivos PHP que contienen Arrays con los textos de diferentes lenguajes, para poder traducir la aplicación.

En la Figura 15-1 se presenta la extrucutra del directorio del Framework Laravel.

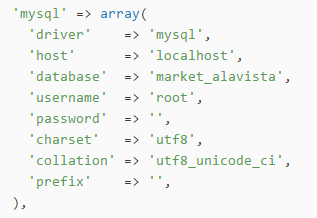


**Figura 15-1.** Estructura de directorios en Laravel.

**Fuente**: (Velasquez R, 2013, codehero.co)

### Base datos

Para manipular una base de datos, se debe realizar la conexión editando el archivo de conFiguración de Laravel, se edita el archivo /app/config/database.php y se edita los campos de ‘connections’ sqlite, mysql, pgsql, sqlsrv o nuevas conexiones creadas que se utilizaran.



**Figura 16-1.** ConFiguración para la base de datos.

**Fuente:** (Velasquez R, 2013, codehero.co)

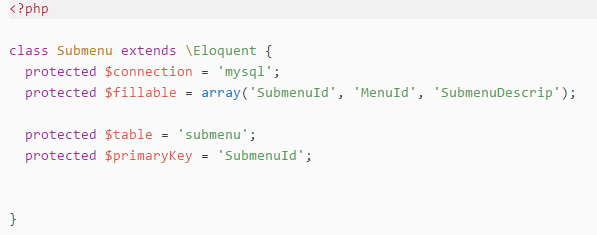
Como se puede ver en la Figura 16-1 se tiene un arreglo con los siguientes campos:

* **'mysql'.-** nombre del arreglo, al cual se refiere para realizar la conexión
* **'driver'.-** controlador que se utiliza para conectarse con el gestor de base de datos
* **'host'.-** equipo en el que se encuentra el servidor de base de datos
* **'database'.-** nombre de la base de datos
* **'username'.-** usuario que permite el ingreso al servidor de base de datos
* **'password'.-** credencial del usuario
* **'charset'.-** codificación de caracteres de la base de datos
* **'collation'.-** definición de codificación de base de datos o columnas de intercalación
* **'prefix'.-** prefijo que se antepone al nombre de las tablas, es opcional

### Modelo

Los modelos a utilizar se encuentran en la carpeta /app/models. Los modelos en Laravel usan el Eloquent ORM, el cual facilita interactuar con la base de datos, para esto es necesario que cada una de las tablas de la base de datos poseea un modelo correspondiente. Los modelos usan convenciones para poder saber que tabla de la base de datos está utilizando, esto se lo realiza mediante la variable $table, en la cual se identifica la tabla que se utiliza.

Laravel asume que todas las tablas de las base de datos poseen tres campos básicos 'id' (pk), 'created\_at', 'updated\_at' estos dos últimos campos son llenados de manera automática por el framework.



**Figura 17-1.** Modelo Utilizando Eloquent

**Fuente:** (Velasquez R, 2013, codehero.co)

Como se puede ver en la Figura 17-1 un modelo hereda de la clase Eloquent, y contiene las siguientes variables:

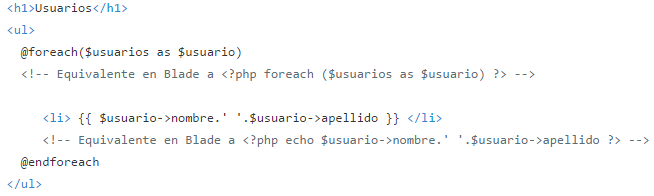
* **$connection.-** permite llamar a una conexión conFigurada en el archivo /app/config/database.php
* **$table.-** nombre de la tabla que utilizará el modelo
* **$fillable.-** arreglo en el cual se agrega los campos que se permiten modificar
* **$primaryKey.-** clave primaria de la tabla

### Eloquent ORM

Eloquent es un ORM que significa Mapeo Objeto-Relacional la cual es una técnica de programación que esta implementado en una clase de Laravel. Al aplicarse esta técnica se puede acceder a cada registro de la base de datos como si se tratara de objetos en PHP sin tener que ejecutar código SQL. Esto se lo puede realizar gracias a una clase en el proyecto (modelos), ya que cada tabla de la base de datos es manejada por una de las mencionadas clases, además esta técnica permite crear relaciones como propiedades de los objetos para poder relacionarlos de una manera mucho más sencilla. Finalmente nos permite independizarse de una base de datos, se puede cambiar de motor de base de datos en cualquier momento y el código seguirá funcionando correctamente.

### Vista

Las vistas se crean en la carpeta /app/views/, en la Figura 18-1 se observa un ejemplo lista.blade.php. En este archivo se escribe el contenido html, la extensión blade.php permite que el archivo pueda utilizar sentencias PHP y el sistema de plantillas Blade.



**Figura 18-1.** Código en la Vista

**Fuente:** (Velasquez R, 2013, codehero.co)

### Blade

Blade es un motor de plantillas proporcionado con Laravel. Blade es impulsado por la herencia de plantillas y secciones. Todas las plantillas de hoja deben utilizar la extensión .blade.php.

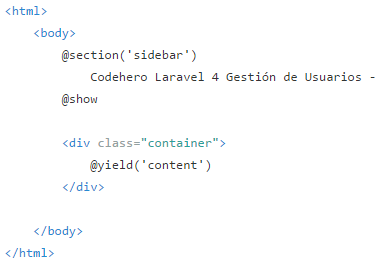
#### Vista Master

Como se observa en la Figura 14-1, contiene la estructura básica HTML de una página Web la cual tiene las etiquetas <html>, <body> y <div>, pero además se observa que hay funciones de Blade @section, @show, @yield, estas funciones ayudan a crear plantillas mediante herencia y secciones, las cuales ayudan a no repetir el código HTML en cualquier vista que se tenga que crear.

En este archivo es en el cual se debe crear la estructura básica del sitio Web, ejemplo menú, cabecera, pie de página, entre otros, se debe incluir todos los archivos javascript o css que se van a utilizar en todo el sitio.

En el ejemplo de se observa dos secciones, @section('sidebar') @show, en esta sección se puede completarla con más código del que posee, al momento que se crea la vista se puede llamar a esta sección (el cual está dentro de @section y @show) y se puede agregar más contenido si es necesario.

La siguiente sección es @yield ('content'), observar la Figura 19-1, esta sección no tiene contenido previo, cuando se llame a esa sección desde una vista se incluirá todo el contenido donde está declarada. Se puede pensar que estas secciones son como una especie de variables que se las pueden llenar desde las vistas con más HTML.



**Figura 19-1.** Vista Master

**Fuente:** (Velasquez R, 2013, codehero.co)

#### Vistas que Heredan

Estas vistas contienen diferentes funciones, que permitirán llamar al archivo padre.

La función @extends() permite llamar a la plantilla general, solo es necesario colocar las carpetas separándolas con punto (.).

@section('sidebar') y @yield('content'), se abre la sección mediante el nombre que corresponda, se agrega el código que se necesita incluir y se cierra con @stop

En las vistas se puede utilizar la clase Form, mediante la cual se puede crear elementos para ser empleados en un formulario.

* Form::open: permite abrir la etiqueta form
* Form::text: Esta función permite crear una etiqueta <input> la cual es del tipo text, el primer parámetro que recibe es el nombre de la etiqueta( propiedad name), el segundo parámetro es el valor que tiene el input(propiedad value). Finalmente se le puede pasar un tercer parámetro con el cual contendrá propiedades HTML o CSS
* Form::label: Esta función facilita crear una etiqueta <label>
* Form::submit: Esta función facilita crear una etiqueta <input> de tipo submit para enviar el formulario
* Form::close: Por último esta función facilita crear una etiqueta <form>

En la Figura 20-1 se presenta un ejemplo de una vista heredada.



**Figura 20-1.** Vista heredada

**Fuente:** (Velasquez R, 2013, codehero.co)

### Controlador

El controlador es el tercer componente de MVC, este permite unir el modelo y la vista, los controladores se crean en el directorio /app/controllers

Todos los controladores deben extender del controlador BaseController. El nombre de la nueva clase debe terminar en Controller y debe ser igual al nombre del archivo en donde se encuentra implementada.

En la Figura 21-1 se presenta un ejemplo de un controlador en Laravel.

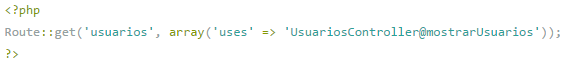


**Figura 21-1.** Controlador en Laravel

**Fuente:** (Velasquez R, 2013, codehero.co)

### Ruta

Mediante esta manera Laravel nos permite llegar hasta la acción y el controlador que se desea, para crear una ruta se la debe agregar en el archivo /app/routes.php, obvervar la Figura 22-1.

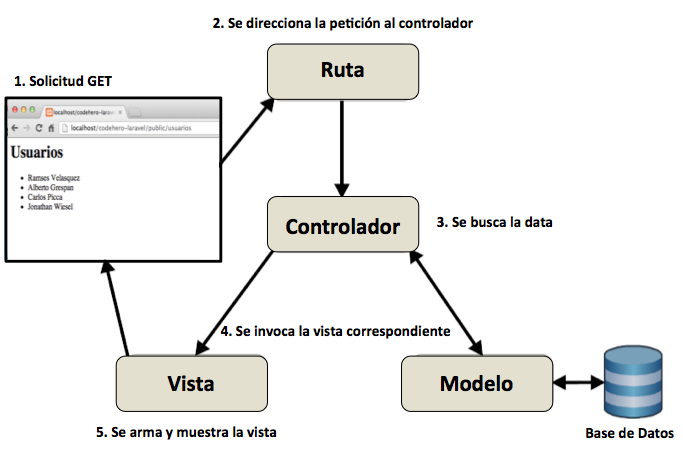


**Figura 22-1**. Archivo de rutas en Laravel

**Fuente:** (Velasquez R, 2013, codehero.co)

Mediante este código se declara que al realizar una petición GET o POST a la ruta ‘usuarios’, la petición será atendida por el controlador UsuariosController con la acción mostrarUsuarios. Hay que decir que todas las acciones que se deseen publicar para que el usuario pueda acceder deben declararlas mediante rutas, si no el usuario no podrá acceder a ellas.

En la siguiente Figura 23-1 se puede observar gráficamente el proceso que se lleva a cabo cuando se realiza una petición al servidor.



**Figura 23-1.** Proceso de petición al controlador

**Fuente:** (Velasquez R, 2013, codehero.co)

## Facebook

Facebook es una red social fundada en el 2004, diseñada con el objetivo de conectar personas para que se encuentren informados de lo que está sucediendo en el mundo sin importar la distancia entre ellos. Permite crear perfiles en los cuales se puede agregar información personal, publicar eventos, estados, noticias, etc. De esta manera la interacción entre usuarios es sencilla y de manera directa. (Facebook Newsroom, 2015, https://newsroom.fb.com)

Hoy en día Facebook es una de las redes sociales más populares en todo el mundo, en el 2014 Facebook cubre más del 69% de la audiencia global conectada en redes sociales, con un aproximado de 59% de usuarios activos. Cuenta con un promedio hasta la fecha de marzo del 2015 de 936 millones de usuarios activos diariamente, 798 millones de usuarios activos desde dispositivos móviles. (Facebook Newsroom, 2015, https://newsroom.fb.com)

Considerando que Facebook es una de las redes sociales con mayor crecimiento de manera global, se observa desde la perspectiva del mercadeo como una potente plataforma de publicidad en línea, debido a esto las empresas cada vez hacen uso de ella para sus estrategias de marketing, esto es evidente ya que hoy en día Facebook cuenta con 40 millones de páginas activas de negocios.

### Facebook para desarrolladores

Facebook posee varias herramientas que pone a disposición para que los desarrolladores de software puedan integrar esta red social a sus proyectos. Pone a disposición sus nuevas APIs y SDKs.

Las herramientas utilizadas en esta investigación son

* Graph API
* SDK para PHP

Adicionalmente se tiene:

* SDK para iOS
* SDK para Android
* SDK para JavasCript
* SDK para Unity

El **SDK para iOS** es una biblioteca de código abierto que permite integrar Facebook con aplicaciones para IOS, de igual manera el **SDK para Android** permite integrar Facebook en aplicaciones Android

El **SDK para JavasCript** proporciona un conjunto de funcionalidades al client, las culaes permiten utilizar el botón Me gusta u otros plugins sociales en un sitio Web

El **SDK para Unity** ofrece una colección de las características sociales de Facebook mediante las cuales los jugadores de un Juego de Unity pueden compartit contenido con sus amigos

### Graph API

Es la principal manera de obtener datos de la plataforma de Facebook ya sea desde el interior o exterior de ella, basada en HTTP de bajo nivel, se puede utilizar para realizar consultas, gestionar estados, gestionar anuncios, trabajar con fotos, etc.

#### Estructura de Graph API

La estructura de petición es sencilla, únicamente cada objeto posee un identificador para hacer uso de la API, existen diferentes tipos de peticiones, en la siguiente Tabla 4-1 se detalla cada una de ellas.

**Tabla 4-1.** Tipo de peticiones a la Graph API

|  |  |
| --- | --- |
| **Peticiones** | **Descripción** |
| HTTP GET | Para obtener datos del usuario que se encuentre con una sesión activa. |
| HTTP POST | Para realizar publicaciones en el perfil del usuario que se encuentre con una sesión activada, o para realizar operaciones en nombre del usuario. |
| HTTP DELETE | Para realizar operaciones de eliminación. |

**Fuente:** (Facebook Developers, 2015)

**Realizado por:** Villa E., Rodriguez E., 2015.

La información presentada por Graph API tiene una estructura la cual posee la siguiente información de manera básica.

* **Nodes**. Los datos de la consulta, ya sea fotos, páginas, comentarios
* **Edges**. Representa la conexión que existe entre los nodos, como puede ser comentarios de una foto, fotos de una página
* **Fields**. Información de las consultas realizadas, como por ejemplo, cumpleaños de un usuario, nombre de la página, etc

#### Explorador de Graph API

Facebook facilita hacer pruebas de su Graph API mediante su explorador, servicio que se encuentra disponible en la página para desarrolladores, en este servicio se puede emular la solicitud de datos a la API de manera rápida y sencilla. El la Figura 24-1 se observa un ejemplo de una solicitud de información de un usuario activo.



**Figura 24-1.** Explorador de la Grahp API

Fuente: (Facebook Developers, 2015)

La información obtenida a partir de la Graph API si fue exitosa, arroja el presente resultado.



**Figura 25-1.** Respuesta de la Graph API

Fuente: (Facebook Developers, 2015)

### SDK para PHP

Contrastando a que el desarrollo de la aplicación en el presente trabajo de titulación se lo realizara con el lenguaje de programación PHP, se estudiara la SDK correspondiente para este lenguaje, estudiando los métodos más importes que dispone.

Es una biblioteca que facilita a los desarrolladores hacer uso de funcionalidades con el objetivo de acceder a la Graph API, utilizado principalmente para realizar operaciones como administrador de la aplicación, así también como realizar operaciones en nombre del usuario quien haya iniciado sesión, o que se encuentre con una sesión activada.

Actualmente se encuentra en la versión 4.0 para la cual se requiere como mínimo la versión de PHP 5.4. A continuación se detallara las funciones primordiales para integrar Facebook a la aplicación Web. (Facebook Developers, 2015, https://developers.facebook.com)

#### Instalación del SDK para PHP

Existen dos maneras de instalación, la más sencilla es hacer uso de composer, el cual es un gestor de paquetes PHP, únicamente se debe agregar una línea de código en el archivo de composer donde se encuentra todas las dependencias de la aplicación.

En la siguiente Figura 26-1 se puede observar un ejemplo de conFiguración para la instalación.



**Figura 26-1.** ConFiguración para la instalación

Fuente: (Facebook Developers, 2015)

En el caso de no hacer uso de composer como administrador de dependencias se puede descargar el SDK desde el repositorio oficial de Facebook en GitHub. Antes de empezar a desarrollar la aplicación Web que se integrará con Facebook se debe crear una aplicación de tipo Web, en la cual se realizara conFiguraciones importantes como:

* URL de la aplicación Web
* El dominio de la aplicación Web
* Agregar los permisos que poseerá la aplicación sobre los datos del usuario que haya iniciado sesión con la aplicación

### Permisos con el inicio de sesión.

En el momento en que una persona se registra en una aplicación mediante el inicio de sesión con Facebook, la aplicación tiene acceso a los datos del usuario, así también a realizar operaciones en nombre del usuario. Para controlar estas acciones Facebook al momento de registrar la aplicación realiza preguntas al usuario si está de acuerdo con los permisos que pretende acceder la aplicación.

Existen diferentes tipos de permisos, los que no requieren autorización por el usuario y los que requieren autorización de los usuarios.

**No requieren autorización**

* Perfil público por defecto. Obtiene la información básica del usuario, como nombre, apellido, fecha de nacimiento, etc
* Lista de amigos. Obtiene la lista de amigos de los usuarios registrados
* Permiso de correo electrónico. Permite el acceso a la dirección de correo electrónico principal del usuario

**Requieren autorización**

* Propiedades de perfil extendidos. Permite tener acceso a propiedades sensibles del perfil del usuario que no se encuentren públicos
* Permisos extendidos. Constituye a información más sensible del perfil, por ejemplo realizar operaciones a nombre del usuario
* Permisos Open Graph. Para acceder a los datos almacenados en Open Graph de un usuario
* Permisos de página. Permite administrar páginas que tenga registrado el usuario

En la Tabla 5-1 se detalla los principales permisos de Facebook.

**Tabla 5-1.** Descripción principales permisos de Facebook

|  |  |
| --- | --- |
| **Permiso** | **Descripción** |
| public\_profile  (por defecto) | Proporciona información a una lista de elementos del perfil público como por ejemplo:   * identificación * nombre * primer nombre * apellido * rango de edad * enlace * género * local * zona horaria * updated\_time * verificado |
| user\_friends | Permite el acceso a la lista de amigos del usuario. Siempre y cuando los dos amigos hayan otorgado este permiso a la aplicación. |
| email | Proporciona el acceso a la dirección de correo electrónico principal del usuario. |
| user\_about\_me | Proporciona acceso a la información personal de una persona (sobre mí, sección del perfil). |
| user\_groups | Permite obtener la lista de grupos en la cual el usuario se encuentre asociado. |
| manage\_pages | Permite a su aplicación administrar las páginas de una persona en la cuales sea administrador. |
| publish\_pages | Permite realizar publicaciones en las páginas que administre una persona. |
| publish\_actions | Proporciona acceso para publicar mensajes, acciones Open Graph, logros, resultaros, y otras actividades que haya realizado una persona en una aplicación. |
| user\_events | Proporciona acceso de lectura a los acontecimientos de una persona está organizando o ha sido invitada. |

**Fuente:** (Facebook Developers, 2015)

### ConFiguración de la aplicación

Para empezar a utilizar el SDK, previamente se debe conFigurar una aplicación de Facebook, para obtener el identificador de la aplicación (app ID), y el identificador secreto (app secret). Estos identificadores deben ser agregados en el código del proyecto para hacer uso de la aplicación registrada, como se puede observar en la Figura 27-1.

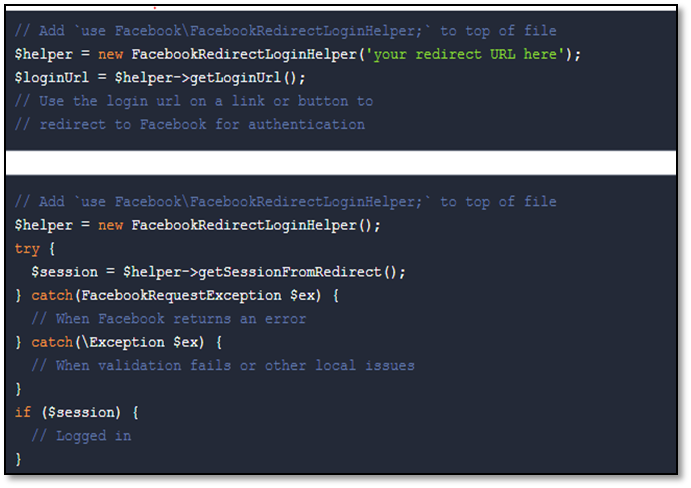


**Figura 27-1.** ConFiguración de la aplicación

**Fuente:** (Facebook Developers, 2015)

### Autorización y autenticación

Las aplicaciones de Facebook para que puedan acceder a la información de los usuarios, realizar operaciones a nombre del usuario, primeramente es necesario autorizar a la aplicación. Para esto se debe agregar las siguientes líneas de código en la aplicación a desarrollar. En la Figura 28-1 se detalla el código para la autorización y autentificación de la aplicación.



**Figura 28-1.** Autorización y autentificación

**Fuente:** (Facebook Developers, 2015)

Entre las funciones más importantes para llevar a cabo esta solicitud de información, se detallan en la Tabla 6-1.

**Tabla 6-1.** Funciones para la autorización y autentificación de la aplicación

|  |  |
| --- | --- |
| **Funciones** | **Descripción de lo que realiza.** |
| **getLoginUrl()** | Genera la URL para redireccionar al usuario desde Facebook hacia su aplicación para que inicie sesión. |
| **getSessionFromRedirect()** | Procesa el redireccionamiento de Facebook, si la sesión se inició exitosamente retorna a la aplicación una sesión de Facebook, caso contrario retorna una sesión nula. |

**Fuente:** (Facebook Developers, 2015)

### Usar la Graph API

Una vez generada la sesión, se podrá hacer llamadas a la Graph API con el método FacebookRequest, dependiendo la información que se desee obtener se estructura el código, para ayuda de los desarrolladores Facebook cuenta con documentación donde proporciona ejemplos básicos de cómo hacer llamadas a la API.

**Obtener información del usuario**

En el siguiente Figura 29-1 se puede apreciar el código haciendo uso de la Graph API para obtener la información de un usuario, este caso el nombre del usuario.

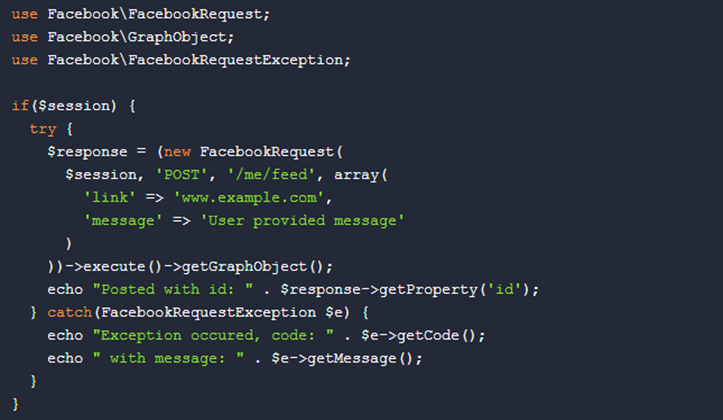


**Figura 29-1.** Código para solicitar la información del usuario

**Fuente:** (Facebook Developers, 2015)

**Publicar Enlaces con mensaje**

En la Figura 30-1 se observa el código para realizar una publicación en la línea del tiempo del usuario que haya iniciado sesión haciendo uso de la Graph API.



**Figura 30-1.** Código para realizar una publicación

**Fuente:** (Facebook Developers, 2015)

**Publicar Fotos de perfil**

En el siguiente Figura 31-1 se observa el código base para subir una foto como es el perfil de usuario haciendo uso de Graph API.

****

**Figura 31-1.** Código para publicar una foto de perfil

**Fuente:** (Facebook Developers, 2015)

### Clases de SDK PHP

En la siguiente Tabla 7-1 se detalla las clases de la SDK para PHP, para hacer uso de la Graph API de Facebook.

**Tabla 7-1.** Principales clases de la SDK de PHP

|  |  |
| --- | --- |
| **Clase** | **Descripción** |
| Facebook\FacebookSession | Genera una sesión mediante un token de acceso, para hacer uso de la Graph API. |
| Facebook\FacebookRequest | Realiza peticiones a la Graph API. |
| Facebook\FacebookResponse | Obtiene una respuesta exitosa desde la Grahp API resultante de FacebookRequest |
| Facebook\  FacebookRequestException | Error devuelto por la Graph API luego de hacer una petición con FacebookRequest, los diferentes tipos de error son: FacebookClientException, FacebookServerException, FacebookAuthorizationException, FacebookPermissionException, FacebookThrottleException, FacebookOtherException. |
| Facebook\GraphObject | Es un objeto devuelto por la Graph API, sus subclases son: GraphUser, GraphLocation, GraphSessionInfo. |
| Facebook\FacebookRedirectLoginHelper | Clase que permite generar la URL de inicio de sesión para obtener la sesión de Facebook utilizando el protocolo OAuth, mediante los métodos getLoginUrl(), getSessionFromRedirect() que devuelve un FacebookSession en caso de éxito. |
| Facebook\FacebookCanvas LoginHelper | Clase de ayuda para conseguir una sesión de Facebook desde Facebook Canvas. |
| Facebook\FacebookJavaScriptLoginHelper | Clase de ayuda para conseguir una sesión de Facebook desde la SDK para JavaScript. |

**Fuente:** (Facebook Developers, 2015)

### Control de errores

No todas las solicitudes realizadas a la API son exitosas, por alguna razón se puede presentar un error, por lo tanto se debe controlar estos posibles errores, para esto Facebook proporciona una lista de los códigos de errores con sus nombres y la posible razón del porqué del problema, los cuales se pueden apreciar en la Tabla 8-1.

**Tabla 8-1.** Control de errores de Facebook

|  |  |  |
| --- | --- | --- |
| Código | Nombre | Razón |
| OAuthException |  | Sesión o token de acceso caducados. |
| 102 | Sesión API | Sesión o token de acceso caducados. |
| 1 | API Desconocido | Posiblemente un problema temporal debido al tiempo de inactividad. |
| 2 | Servicio API | Cuestión temporal debido al tiempo de inactividad. |
| 4 | Demasiadas llamadas a la API | Debido a exceder los límites de solicitud a la API, vuelva a intentarlo luego de esperar un momento. |
| 17 | API Usuario demasiadas llamadas | Debido a exceder los límites de solicitud a la API, vuelva a intentarlo luego de esperar un momento. |
| 10 | Permiso denegado API | Permiso denegado o eliminado. |
| 200 - 299 | Permiso API (múltiples valores dependiendo los permisos) | Permiso denegado o eliminado. |
| 341 | Límite de la aplicación alcanza | Debido al tiempo de inactividad. |
| 506 | Publicación duplicada | Mensajes duplicados no se pueden publicar de forma consecutiva |
| 1609005 | Publicación error en el enlace | Error en el enlace proporcionado, compruebe la dirección de URL. |

**Fuente:** (Facebook Developers, 2015)

### Fichas de acceso (Access Tokens)

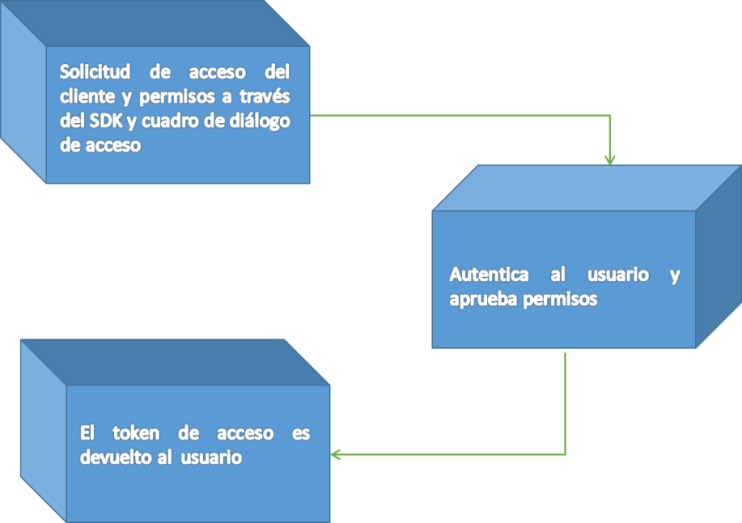
Para hacer uso de la aplicación Web se debe generar una ficha de acceso, la cual es generada al momento de iniciar sesión en Facebook, proporcionando un acceso temporal y seguro a la API.

Una ficha de acceso es una cadena de caracteres que identifica a un usuario, aplicación, página, para que pueda ser utilizado por la aplicación para hacer uso de la Graph API. La ficha posee información, como fecha de expiración, que aplicación genera el token. Existen diferentes tipos de fichas de acceso de acuerdo a la aplicación a utilizar.

**Ficha de acceso de usuario**

Ficha de acceso mayormente usado por las aplicaciones, se hace uso de una ficha de acceso cada vez que la aplicación necesita hacer una llamada a la API para leer, modificar, escribir datos en Facebook.

Se obtiene mediante el cuadro de dialogo de inicio de sesión en Facebook, luego de que haya dado los permisos a la aplicación, el método para iniciar este proceso sin importar la plataforma que se utilice sigue el mismo proceso que se detalla en el Figura 32-1.



**Figura 32-1.** Pasos para obtener token de acceso

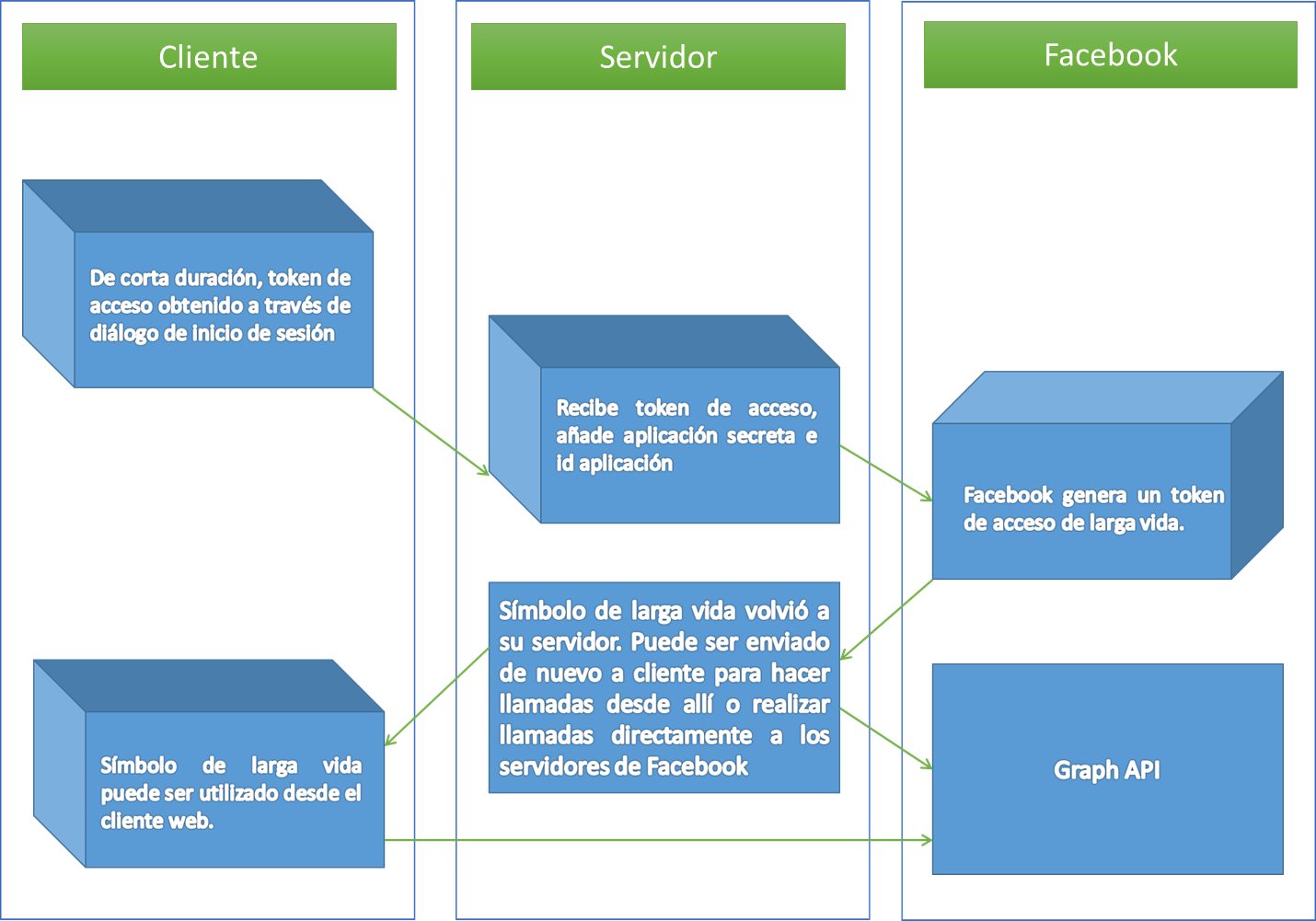
**Fuente:** (Facebook Developers, 2015)

**Ficha de acceso de aplicación.** Ficha de acceso para modificar y leer la conFiguración de aplicaciones. También se puede hacer uso para publicar acciones Open Graph.

**Ficha de acceso de página.** Ficha de acceso similar a las fichas de acceso de usuario, con la diferencia que da permisos a la API para escribir, leer, modificar datos pertenecientes a una página de Facebook.

**Ficha Cliente.** Identificador usado para incrustar archivos binarios móviles nativas o aplicaciones de escritorio para identificar su aplicación. Utilizado para acceder a la API de nivel de aplicación pero con limitaciones.

Las fichas de acceso en la Web por lo general tienen un tiempo de vida útil de 2 horas pero se actualizan automáticamente si es necesario, también existen fichas de larga duración las cuales tienen una vida útil de 60 días. Para generar estas fichas se debe seguir el siguiente proceso detallado en la Figura 33-1.



**Figura 33-1.** Proceso para solicitud de cambio de token

**Fuente:** (Facebook Developers, 2015)

## Twitter

Es una red social de tipo microblogging creada en el 2006, con el objetivo de que las personas puedan enterarse de lo que está sucediendo en el mundo en ese preciso momento, compartiendo información de manera instantánea y resumida. Una de sus características principales son los tweets, los cuales son mensajes cortos con un máximo de 140 caracteres, estos pueden contener texto, fotos, enlaces y videos, considerando que todo enlace ocupara únicamente 22 caracteres de manera directa. (Omofonmwan N, 2012, p. 4)

Desde sus inicios su popularidad ha ido incrementando de manera impresionante, debido a este crecimiento es una de las redes sociales con mayor audiencia en el mundo. En el 2014 ocupa un 44% de la audiencia global conectada a redes sociales, ubicándose únicamente por debajo de Facebook, Google+ y YouTube. Con una cantidad mayor a 271 millones de usuarios activos mensualmente produciendo un total de 500 millones de Tweets al día. (Online Busine School, 2014,pp. 7,8). Para la fecha de Marzo 31 del 2015 cuenta con un aproximado de 302 millones de usuarios activos mensuales, con más 500 millones de Tweets al día, donde un 80% de usuarios activos se conectan mediante dispositivos móviles. (Twitter, 2015, https://about.twitter.com/company)

### Twitter para desarrolladores

Twitter posee varias herramientas para integrar ya sea una aplicación o página Web con twitter, estas herramientas pueden ser utilizadas por los desarrolladores dependiendo el tipo de proyecto que se encuentren elaborando. Las herramientas que pone a disposición son las siguientes.

**Fabric**. Es un conjunto de herramientas para el desarrollo de aplicaciones móviles flexibles y modulares, el cual ayuda a que las aplicaciones sean estables, añadiendo características sociales.

**Twitter para sitios Web.** Es un conjunto de widgets embebidos, botones, herramientas de scripting en el lado del cliente, con el objetivo de integrar Twitter con una página, para mostrar los tweet de un usuario, presentar botones como: Tweet Button, the Follow Button, Embedded Tweets, andEmbedded Timelines.

**Cards.** Proporcionan contenido adicional a un tweet al momento de compartir un enlace, por ejemplo dar mayor información de una fotografía, video, o resumen de una página.

**OAuth.** Utiliza los OAuth para conectar a los usuarios de Twitter y enviar solicitudes autorizadas de manera segura a la API de Twitter.

**REST APIs.** Permite leer y escribir los datos de Twitter, asi también como realizar operaciones para tweetear, obtener información de un perfil, etc. Identifica las aplicaciones de Twitter y usuarios haciendo uso de OAuth. Las respuestas están disponibles en el formato JSON.

**Streaming APIs.** Mantiene una conexión HTTP de una consulta realizada mediante la API REST, lo que le permite recibir continuamente respuestas de la consulta realizada, sin tener la necesidad de realizar nuevas consultas, usado principalmente para mantener sincronizados los tweet de un perfil, solución ideal si se cuenta con una aplicación que tiene límites de solicitud a la API.

**Ads API.** Proporciona a los socios la manera de integrar la gestión de la publicidad de Twitter en sus productos. Los socios seleccionados tienen la posibilidad de crear herramientas de acuerdo a sus necesidades de gestión de campañas publicitarias.

**MoPub.** Es una plataforma de publicidad en aplicaciones móviles más grande del mundo, permite la integración de anuncios publicitarios de manera rápida y sencilla. Es considerada como una de las principales plataformas de monetización utilizada por los desarrolladores. A diferencia de las demás plataformas permite conectarse a miles de fuentes de publicidad móvil con mayor intercambio en el mundo.

### Límites de petición a la API

Para hacer uso de la API versión 1.1 es importante tener en cuenta los límites de peticiones permitidas, para tener acceso a los datos de Twitter es necesario realizar peticiones mediante un token de acceso, el token de acceso puede ser generado por un usuario o por la aplicación, existen límites diferentes de acuerdo al tipo de token.

En la Tabla 9-1 se presenta de manera detallada los límites de peticiones a la API de acuerdo al tipo de token generado.

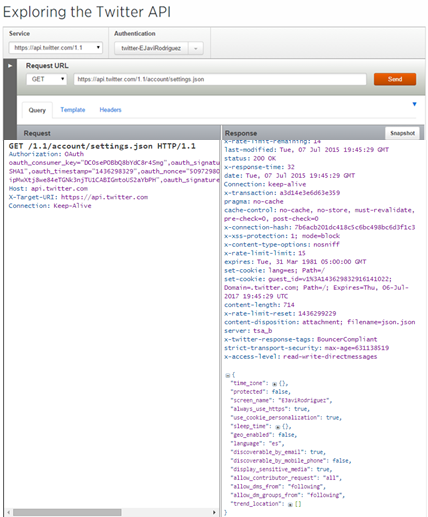
**Tabla 9-1.** Límite de peticiones a la API

|  |  |  |  |
| --- | --- | --- | --- |
| **Operación** | **Tipo recurso** | **Solicitudes / 15 min ventana (autenticación de usuario)** | **Solicitudes / 15 min ventana (aplicación de autenticación)** |
| GET application/rate\_limit\_status | Solicitud | 180 | 180 |
| GET favorites/list | Favoritos | 15 | 15 |
| GET followers/ids | Favoritos | 15 | 15 |
| GET followers/list | Favoritos | 15 | 30 |
| GET friends/ids | Amigos | 15 | 15 |
| GET friends/list | Amigos | 15 | 30 |
| GET friendships/show | Amistades | 180 | 15 |
| GET search/tweets | Búsqueda | 180 | 450 |
| GET statuses/retweeters/ids | Estados | 15 | 60 |
| GET statuses/retweets/:id | Estados | 15 | 60 |
| GET statuses/show/:id | Estados | 180 | 180 |
| GET statuses/user\_timeline | Estados | 180 | 300 |
| GET trends/available | Tendencia | 15 | 15 |
| GET trends/closest | Tendencia | 15 | 15 |
| GET trends/place | Tendencia | 15 | 15 |
| GET users/show | Usuarios | 180 | 180 |
| GET users/suggestions | Usuarios | 15 | 15 |

**Fuente:** (Twitter developers, 2015)

### Explorador de API de Twitter

Twitter pone a disposición para los desarrolladores el explorador de la API, donde se puede realizar pruebas de cómo realizar peticiones de manera sencilla, mediante el cual permite realizar peticiones de manera gráfica, como se puede observar en la Figura 33-1.



**Figura 34-1.** Explorador de la API de Twitter

**Fuente:** (Twitter developers, 2015)

En el ejemplo de petición se puede apreciar al lado izquierdo el formato de solicitud mediante GET /1.1/account/setings.json HTTP/1.1, mientras que en el lado derecho la respuesta por la API en formato json.

### Librerías de Twitter

Twitter pone a disposición varias librerías de acuerdo a la plataforma de desarrollo que se vaya a utilizar, entre las plataformas soportadas son las siguientes:

* Java
* ASP
* C++
* Closure
* ColdFusion
* .NET
* Go
* Javascript/ node.js
* Lua/Corona SDK
* Objective-C
* Perl
* PHP
* Python
* Ruby

Para el desarrollo con el lenguaje de programación PHP, Twitter cuenta con varias librerías, entre ellas se encuentra la librería TwitterOAuth para el uso de la API OAuth REST construida por Abraham Williams, y se encuentra en la plataforma GitHub. Cuenta con una documentación detallada en su página oficial.

**La instalación**

Para agregar la librería al proyecto se puede realizar mediante composer, como se puede observar en la Figura 35-1, únicamente se debe conFigurar el archivo composer.json agregando la línea de código que haga referencia a la librería.



**Figura 35-1.** Código para la instalación Twitter

**Fuente: (**Williams A., 2015, twitteroauth.com)

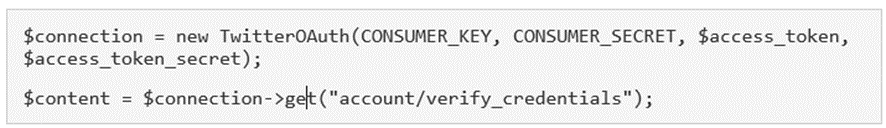
Ahora solamente se debe actualizar el proyecto con el comando composer update en la consola de símbolo de sistemas en Windows. Posterior a esto, ya se puede hacer uso de la librería, solamente se debe importar la librería para empezar a utilizarla, como se observa en la Figura 36-1.



**Figura 36-1.** Código para hacer uso de la librería

**Fuente:** (Williams A., 2015, twitteroauth.com)

Para empezar a realizar llamadas a la API, previamente a esto se debe conFigurar una aplicación en la página de twitter Gestión de Aplicaciones, para obtener las credenciales que se agregaran en el proyecto para integrarlo, como se observa en la Figura 37-1.



**Figura 37-1.** ConFiguración de la API

**Fuente:** (Williams A., 2015, twitteroauth.com)

### Métodos más relevantes

**OAuth**

En la Figura 38-1 se observa como se utiliza el método para autorizar el acceso a una cuenta de usuario, incluye algunos métodos como  POST oauth/request\_token y POST oauth/access\_token.

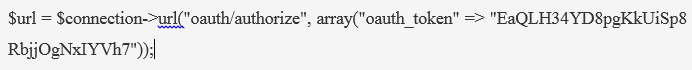


**Figura 38-1.** Autorización de la aplicación

**Fuente:** (Williams A., 2015, twitteroauth.com)

**URL**

Método que construye la URL para que los usuarios realicen la autorización de la aplicación para el acceso a la cuenta. Se utiliza únicamente para GET oauth/authorize and GET oauth/authenticate.



**Figura 39-1.** Construcción de la URL de autenticación

**Fuente:** (Williams A., 2015, twitteroauth.com)

Método para realizar peticiones a la API de tipo HTTP GET, ejemplo GET search/tweets.

HTTP



**Figura 40-1.** Llamado de peticiones en Twitter

**Fuente:** (Williams A., 2015, twitteroauth.com)

TwitterOAuth



**Figura 41-1.** Construcción de la URL de autenticación

**Fuente:** (Williams A., 2015, twitteroauth.com)

**POST**

Método para realizar peticiones a la API de tipo HTTP POST, ejemplo POST statuses/update.

HTTP



**Figura 42-1**. Peticiones de tipo POST

Fuente: (Williams A., 2015, twitteroauth.com)

TwitterOAuth



**Figura 43-1.** Actualización de token

**Fuente: (Williams A., 2015, twitteroauth.com)**

### Códigos de Error

No todas las solicitudes realizadas a la API de Twitter son exitosas, por alguna razón se puede presentar un error, por lo tanto se debe controlar estos posibles errores. Cuando se presenta un error la respuesta de la API es un mensaje de error con su respectivo código en formato JSON. Los errores más comunes se detallan en la Tabla 10‑1

**Tabla 10-1.** Códigos de error

|  |  |  |
| --- | --- | --- |
| **Código** | **Mensaje** | **Descripción** |
| 32 | No se pudo autenticar | La llamada no se puede completar, por error de autenticación. |
| 34 | Lo siento, esa página no existe | Corresponde con un HTTP 404 - No se encuentra el recurso especificado. |
| 64 | Su cuenta está suspendida y no se le permite acceder a esta función. | Error de tipo HTTP 403, el token de acceso corresponde a un usuario suspendido por lo tanto no se puede completar la acción. |
| 88 | Excede el límite de peticiones por tiempo específico. | Se ha alcanzado el límite de solicitud de este recurso. |
| 89 | Token no válido o caducado | El token de acceso utilizado en la solicitud no es correcta o ha expirado. |
| 131 | Error interno | Corresponde con un HTTP 500 - Error interno desconocido. |
| 135 | No se pudo autenticar | Error de tipo HTTP 401, significa que el oauth\_timestamp ha sobrepasado el tiempo admitido. |
| 187 | Estado duplicado | El texto ya ha sido publicado en Twitter por el usuario autentificado. |
| 215 | Malos datos de autenticación | Error de tipo HTTP 400, significa que el método requiere autenticación, y la información ingresada es incorrecta. |
| 261 | La aplicación no puede realizar acciones de escritura. | Error de tipo HTTP 403, significa que la aplicación tiene restringido el uso de acciones POST, PUT o DELETE |

**Fuente:** (Twitter developers, 2015)

## YouTube

YouTube es una comunidad virtual, basada en tecnología de videoblog que nacio en el año 2005, Este permite a sus usuarios ofrecer, compartir, ver y recomendar herramientas audiovisuales en Internet, usando el lenguaje de videos formateados en flash en diversas extensiones y tamaños, en octubre de 2006 fue adquirido por Google Inc.

YouTube aloja gran variedad de videoclips de películas, programas de televisión, videos musicales, videos caseros, además esta plataforma brinda la posibilidad de insertar un videoclip a un sitio Web externo a sus servidores, lo cual permite a bloggers y creadores de sitios Web disponer de sus videos en sus páginas.

Los espacios de exposición de recursos audiovisuales han ganado un nuevo espacio desde la Web 2.0, especialmente con los Blogs por lo cual ahora es posible que una persona tenga un canal propio de comunicación. El uso de este medio genera nuevas maneras de comunicación. (Marañon O., 2012, pp. 119-121)

### YouTube para desarrolladores

YouTube brinda varias herramientas a los desarrolladores de software para que puedan integrarlo a sus proyectos. Pone a disposición su nueva API.

* YouTube Data API (v3)

### YouTube Data API (v3)

Mediante la API de Datos de YouTube, es posible añadir una variedad de características de YouTube a un sitio Web o aplicación propia. Se utiliza la API para buscar videos mediante términos de búsqueda específicos, temas, lugares, fechas de publicación.

**Antes de comenzar**

* Se necesita una cuenta de Google para acceder a la Consola de la API de Google, y solicitar una clave de API para registrar una aplicación.
* Registrar la aplicación en Google para poder enviar solicitudes de API.
* Después de registrar la aplicación, seleccionar la API de datos de YouTube como uno de los servicios que utilizara una aplicación:
* Aprender conceptos básicos del formato de datos JSON (JavaScript Object Notation). JSON es un formato de datos común e independiente del lenguaje que proporciona una representación de texto simple de estructuras de datos arbitrarias.

### PHP Code

La Biblioteca cliente de API de Google para PHP, está disponible para YouTube Data API. La cual se la puede instalar utilizando el gestor de dependencias Composer, como se observa en la Figura 44-1.



**Figura 44-1**. Instalación mediante Composer

Fuente: (Google Developers, 2015)

### Recursos y tipos de recursos

Un recurso es una entidad de datos individual con un identificador propio. En la Tabla 11-1 se describen los distintos tipos de recursos mediante los cuales se puede interactuar con la API.

**Tabla 11-1.** Recursos y tipos de recursos

|  |  |
| --- | --- |
| **Recursos** | |
| Activity | Contiene información sobre una acción que un usuario realizo en el sitio de YouTube, las cuales incluyen a: calificar videos, compartir videos, marcar un video como favorito, publicar el boletín de un canal, etc. |
| Cannel | Contiene información sobre un solo canal de YouTube. |
| channelBanner | Identifica la dirección URL que se va a usar para conFigurar una imagen recién cargada como la imagen de banner de un canal. |
| guideCategory | Identifica una categoría que YouTube asocia con los canales según su contenido u otros indicadores, como la popularidad. El objetivo de las categorías es organizar los canales de una manera que facilite a los usuarios de YouTube encontrar el contenido que buscan. |
| Playlist | Representa una lista de reproducción de YouTube. Una lista de reproducción es una colección de videos que se pueden ver de manera sucesiva y compartir con otros usuarios. |
| playlistItem | Identifica un recurso, como por ejemplo un video, que forma parte de una lista de reproducción. El recurso playlistItem también contiene detalles que explican cómo se utiliza el recurso incluido en la lista de reproducción. |
| search result | Un resultado de búsqueda contiene información acerca de un video, canal o lista de reproducción de YouTube que coincide con los parámetros de búsqueda especificados en una solicitud de la API. |
| subscription | Contiene información sobre la suscripción de un usuario de YouTube. Una suscripción notifica al usuario cuando se agregan nuevos videos a un canal o cuando otro usuario realiza una de varias acciones en YouTube. |
| thumbnail | Identifica imágenes en miniatura asociadas a un recurso. |
| Video | Representa un video de YouTube. |
| videoCategory | Identifica una categoría que se asoció o podría asociarse a videos subidos. |

**Fuente:** (Google Developers, 2015)

### Operaciones admitidas

En la Tabla 12-1 que se muestra a continuación se visualizan los métodos más comunes que admite la API de YouTube.

**Tabla 12-1.** Métodos que admite la API de YouTube

|  |  |
| --- | --- |
| **Operaciones** | |
| List | Recupera (GET) una lista de cero o más recursos. |
| Insert | Crea (POST) un nuevo recurso. |
| update | Modifica (PUT) un recurso existente para reflejar los datos de tu solicitud. |
| Delete | Elimina un (DELETE) recurso específico. |
| List | Recupera (GET) una lista de cero o más recursos. |

**Fuente:** (Google Developers, 2015)

Actualmente, la API admite métodos para enumerar cada uno de los tipos de recursos admitidos y, del mismo modo, admite operaciones de escritura para muchos recursos.

En la Tabla 13-1 que se muestra en la parte inferior se identifican las diferentes operaciones admitidas para los diferentes tipos de recursos. Operaciones como insertar, actualizar, o eliminar recursos necesitan la autorización del usuario. En varios casos, los métodos list admiten solicitudes tanto autorizadas como no autorizadas, las solicitudes sin autorización recuperan únicamente datos públicos, en cambio las solicitudes autorizadas obtienen información pública o privada de usuario autenticado.

**Tabla 13-1.** Operaciones admitidas

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Operaciones Admitidas** | | | | |
|  | List | Insert | Update | Delete |
| activity | S | S | N | N |
| caption | S | S | S | S |
| channel | S | N | N | N |
| channelBanner | N | S | N | N |
| channelSection | S | S | S | S |
| comment | S | S | S | S |
| commentThread | S | S | S | N |
| guideCategory | S | N | N | N |
| i18nLanguage | S | N | N | N |
| i18nRegion | S | N | N | N |
| playlist | S | S | S | S |
| playlistItem | S | S | S | S |
| search result | S | N | N | N |
| subscription | S | N | N | N |
| thumbnail | N | N | N | N |
| video | S | S | S | S |
| videoCategory | S | N | N | N |
| watermark | N | N | N | N |

**Fuente:** (Google Developers, 2015)

**Realizado por:** Villa E., Rodriguez E., 2015.

S = Si Admite N = No Admite

### Uso de cuota

La YouTube Data API utiliza una cuota para garantizar que los programadores usen el servicio según lo previsto y no creen aplicaciones que reduzcan injustamente la calidad del servicio o limiten el acceso de las demás personas. Se puede encontrar la cuota disponible para tu aplicación en el panel de API console's Cuotas.

Google calcula el uso de cuota mediante la asignación de un costo a cada solicitud, pero no se toma el mismo costo para cada solicitud. Hay dos factores principales que influyen en el costo de la cuota de una solicitud:

Diferentes tipos de operaciones tienen diferentes costos de cuotas.

* Una operación de lectura simple que solo recupera el ID de cada recurso devuelto tiene un costo de aproximadamente 1 unidad
* Una operación de escritura tiene un costo de aproximadamente 50 unidades
* Subir un video tiene un costo de aproximadamente 1600 unidades

Las operaciones de lectura y escritura utilizan diferentes cantidades de cuota, según el número de partes de los recursos que recupera cada solicitud. Las operaciones insert y update escriben datos y también muestran un recurso. Entonces, por ejemplo, la inserción de una lista de reproducción tiene un costo de cuota de 50 unidades por la operación de escritura más el costo del recurso mostrado de lista de reproducción.

Como se explica en la sección que continua en la parte inferior, cada recurso de API se divide en partes. Por ejemplo, un recurso de playlist tiene dos partes, snippet y status, mientras que un recurso de channel tiene seis partes y un recurso de video tiene 10. Cada parte contiene un grupo de propiedades relacionadas, y los grupos están diseñados de manera que tu aplicación solo necesita recuperar los tipos de datos que realmente utiliza.

Una solicitud de API que muestra datos de recursos debe especificar las partes de los recursos que la solicitud recupera. Cada parte agrega aproximadamente 2 unidades al costo de la cuota de la solicitud. Como tal, una solicitud de videos.list que solo recupera la parte snippet de cada video podría tener un costo de 3 unidades. Sin embargo, una solicitud de videos.list que recupera todas las partes de cada recurso podría tener un costo aproximado de 21 unidades de cuota.

Por lo tanto, se puede estimar el número de solicitudes de lectura, escritura o subida que una aplicación podría enviar al día sin exceder la cuota. Por ejemplo, se tiene una cuota diaria de 5.000.000 de unidades, una aplicación podría tener cualquiera de los siguientes límites aproximados:

* 1.000.000 de operaciones de lectura, donde cada una recupera dos partes de recursos
* 50.000 operaciones de escritura y 450.000 operaciones de lectura adicionales, donde cada una recupera dos partes de recursos
* 2.000 subidas de video, 7.000 operaciones de escritura y 200.000 operaciones de lectura, donde cada una recupera tres partes de recursos

### Recursos parciales

La API permite, y requiere, la recuperación de recursos parciales para que las aplicaciones eviten la transferencia, el análisis y el almacenamiento de datos innecesarios. Este enfoque también garantiza que la API utilice los recursos de red, CPU y memoria de manera más eficiente.

La API admite dos parámetros de solicitud, que se explican en las siguientes secciones, que permiten identificar las propiedades de los recursos que se deben incluir en las respuestas de la API.

* El parámetro part identifica grupos de propiedades que se deben mostrar para un recurso
* El parámetro fields filtra la respuesta de la API para que devuelva solo las propiedades específicas dentro de las partes de los recursos solicitados

### El parámetro part

El parámetro part es obligatorio para cualquier solicitud de API que recupera o muestra un recurso. El parámetro identifica una o más propiedades de recursos de nivel superior (no anidada) que se deben incluir en una respuesta de la API. Por ejemplo, un recurso de video tiene las siguientes partes:

* snippet
* contentDetails
* fileDetails
* player
* processingDetails
* recordingDetails
* statistics
* status
* suggestions
* topicDetails

Todas estas partes son objetos que contienen propiedades anidadas, y se puede considerar estos objetos como grupos de campos de metadatos que el servidor de la API podría (o no) recuperar. Como tal, el parámetro part requiere que se seleccione los componentes de recursos que una aplicación realmente usa. Este requisito tiene varios propósitos:

* Permite administrar el uso de cuota de API. Cuando aumenta el número de partes que se recuperan en las respuestas de la API, el uso de la API aumenta en consecuencia, y la cuota disponible disminuye.
* Se reduce la latencia al impedir que el servidor de la API se dedique a recuperar campos de metadatos que la aplicación no utiliza.
* Se reduce el uso de ancho de banda al reducir la cantidad de datos innecesarios que la aplicación puede recuperar.

### ¿Qué es el parámetro fields?

El parámetro fields filtra la respuesta de la API, que solo contiene las partes de los recursos identificados en el valor del parámetro part, de modo que la respuesta solo incluye un conjunto específico de campos. El parámetro fields permite eliminar propiedades anidadas de una respuesta de la API para reducir aún más el uso de ancho de banda. El parámetro part no se puede utilizar para filtrar propiedades anidadas de una respuesta.

Las siguientes reglas explican la sintaxis admitida para el valor del parámetro fields, que se basa libremente en la sintaxis de XPath:

* Se usa una lista separada por comas (fields=a,b) para seleccionar varios campos
* Se usa un asterisco (fields=\*) como comodín para identificar todos los campos
* Se usa paréntesis (fields=a(b,c)) para especificar un grupo de propiedades anidadas que se incluirá en la respuesta de la API
* Se usa una barra diagonal (fields=a/b) para identificar una propiedad anidada

En la práctica, a menudo estas normas permiten que diferentes valores del parámetro fields recuperen la misma respuesta de la API. Por ejemplo, recuperar la ID de elemento de lista de reproducción, el título y la posición de todos los elementos de una lista de reproducción, se puede utilizar cualquiera de los siguientes valores:

* fields=items/id,playlistItems/snippet/title,playlistItems/snippet/position
* fields=items(id,snippet/title,snippet/position)
* fields=items(id,snippet(title,position))

### Ejemplo de solicitudes de la API

Se utiliza la API de datos de YouTube para recuperar y manipular los recursos como videos de YouTube, canales y listas de reproducción, la URL base para el host que procesa las solicitudes del API de datos es https://www.googleAPIs.com/youtube/v3. Para hacer la solicitud, es necesario incluir la dirección URL completa. (Google Developers, 2015, https://developers.google.com)

La dirección URL completa para esta solicitud es:

GET



**Figura 45-1.** URL de solicitud de YouTube Api

**Fuente:** (Google Developers, 2015)

### YouTube Data API – Errors

A continuación se identifican los diferentes tipos de errores que pueden mostrar las operaciones YouTube Data API.

#### Errores generales YouTube

En la Tabla 14-1 se muestra el listado de errores del core API

**Tabla 14-1.** Errores API de YouTube

|  |  |  |
| --- | --- | --- |
| **Tipo de error** | **Detalle del error** | **Descripción** |
| forbidden | forbidden | Acceso prohibido. Puede que la solicitud no esté debidamente autorizada. |
| quotaExceeded | quotaExceeded | La solicitud no se puede completar porque se superó la cuota. |

**Fuente:** (Google Developers, 2015)

**Errores Comunes**

En la Tabla 15-1 se muestra el listado de errores comunes que se pueden presentar en YouTube.

**Tabla 15-1.** Errores comunes de YouTube

|  |  |  |
| --- | --- | --- |
| **Tipo de error** | **Detalle del error** | **Descripción** |
| badRequest | incompatibleParameters | La solicitud especifica dos o más parámetros que no se pueden utilizar en la misma solicitud. |
| badRequest | invalidFilters | La solicitud especifica un parámetro de filtro no válido. |
| badRequest | invalidPageToken | La solicitud especifica un token de página no válido. |
| badRequest | missingRequiredParameter | A la solicitud le falta un parámetro obligatorio. |
| badRequest | unexpectedParameter | La solicitud especifica un parámetro inesperado. |
| forbidden | accountDelegationForbidden | El usuario autenticado no puede actuar en nombre de la cuenta de Google especificada. |
| forbidden | authenticatedUserAccountClosed | La cuenta de YouTube del usuario autenticado está cerrada. En caso de que el usuario autenticado esté actuando en nombre de otra cuenta de Google, este error se refiere a este último punto. |
| forbidden | authenticatedUserAccountSuspended | La cuenta de YouTube del usuario autenticado está suspendida. En caso de que el usuario autenticado esté actuando en nombre de otra cuenta de Google, este error se refiere a este último punto. |
| forbidden | authenticatedUserNotChannel | En esta solicitud, el usuario autenticado debe resolverse en un canal, pero no lo hace. Si la solicitud se autentica y utiliza el parámetro de delegación onBehalfOfContentOwner, entonces también se debe establecer el parámetro onBehalfOfContentOwnerChannel. |
| forbidden | channelClosed | El canal identificado en la solicitud se cerró. |
| forbidden | channelNotFound | No se puede encontrar el canal identificado en la solicitud. |
| forbidden | channelSuspended | El canal identificado en la solicitud se suspendió. |
| forbidden | cmsUserAccountNotFound | No se permite al usuario de CMS actuar en nombre del propietario de contenido especificado. |
| forbidden | insufficientCapabilities | El usuario CMS tiene capacidades insuficientes. |
| notFound | contentOwnerAccountNotFound | No se encontró la cuenta del propietario del contenido especificada. |

**Fuente**: (Google Developers, 2015)

## Vimeo

Vimeo es una red social donde se publica diferentes videos realizados por sus usuarios, para que otros puedan verlos. Vimeo es un sitio en Internet para intercambiar videos que existe desde el 2004, fue creado por un grupo de cineastas y cuenta con gran cantidad de usuarios mayormente artistas de cine, animación, música, y otras obras de arte, los cuales buscan dar a conocer y compartir su arte.

Vimeo es uno de los sitios líderes de video gracias a varias diferencias, tales como:

* **Apariencia Profesional.-** muestra videos a los usuarios, sin publicidad, mostrar publicidad antes de un video puede hacer que los usuarios abandonen el sitio sin ver un video
* **Es gratis.-** si se necesita solo unos cuantos videos o
* **Muy barato.-** si se necesita más espacio
* **Es rápido.-** los videos cargan muy rápido aun siendo de alta definición (HD)

Vimeo es considerado una red de video profesional, por lo cual se encuentra mucho apoyo en la comunidad, la cual está hecha por otros miembros similares. Aunque Vimeo es gratis para un uso básico, también permite realizar una suscripción para usuarios más activos, los mismos que necesitan mejoras en el servicio.

### Vimeo para desarrolladores

Vimeo brinda varias herramientas a los desarrolladores de software para que puedan integrarlo a sus proyectos. Pone a disposición su nueva API.

### Vimeo API

Mediante la API de Vimeo, es posible añadir una variedad de características de Vimeo a un sitio Web o aplicación propia. Se utiliza la API para buscar videos mediante términos de búsqueda específicos, temas, fechas de publicación, y mucho más.

**Para Empezar**

* **Obtener clave de API.- s**e necesita un Client ID y Client Secret para cada aplicación que se cree, para crear una clave, se necesita proporcionar información básica acerca de la aplicación. No es necesario estar seguro de toda la información ya que se puede editar después.
* **Token de Acceso.- p**ara poder realizar peticiones a la API de Vimeo se necesita un token de acceso, si se desea hacer peticiones a nombre de un usuario, de estar autenticado.se redirige a una aplicación en donde se cambia un código por un token de acceso.
* **Librerias Oficiales.-** posee librerías para PHP, Python, Node, iOS, iOS – Player, iOS - Deeplink
* **Librerias No Oficiales.-** .Net, Java
* **Llamar a la API de Vimeo**

Una vez que se tenga un token de acceso, se puede hacer llamada a la API. La API tiene acceso a la mayoría de las características de Vimeo.

### Solicitudes a la API de Vimeo

En la Tabla 16-1 se observa como construir una consulta, cada petición HTTP consiste en un método HTTP, un endpoint y una lista de parámetros. A lo largo de la documentación, estas solicitudes se formatean como el siguiente ejemplo.

{METHOD}https://api.vimeo.com{endpoint}

Este método requiere un token con ámbito de aplicación “{scope\_name}”

**Tabla 16-1.** Campos del API de Vimeo

|  |  |  |
| --- | --- | --- |
| **Campo** | **Necesario** | **Descripción** |
| {field1} | Sí | Descripción del campo1 |
| {field2} | No | Descripción del campo2 |

**Fuente:** (Vimeo developer, 2015)

**Método.** La mayor parte de los puntos finales del API de Vimeo apoyan múltiples acciones.

Cada acción se define por un método HTTP. Estos métodos generalmente siguen diferentes pautas detalladas en la Tabla 17-1.

**Tabla 17-1.** Métodos más relevantes del API de Vimeo

|  |  |  |
| --- | --- | --- |
| **MÉTODO** | **Descripción** | **Ejemplo** |
| GET | Recuperar la representación JSON de un recurso | GET / me Devoluciones información del usuario autenticado |
| POST | Añadir nuevos recursos para una colección. Este método generalmente requiere parámetros adicionales | Poste /me/albums al usuario autenticado crea un nuevo álbum |
| PUT | Reemplazar un recurso o conectar dos recursos juntos | PUT /me/likes/[video\_id] agrega un "me gusta" a [video\_id] en nombre del usuario autenticado |
| PATCH | Modificar un recurso. El cuerpo contendrá la nueva representación de recursos. | PATCH /videos/[video\_id] le permite cambiar los campos del video [video\_id] |
| DELETE | Eliminar un recurso existente, o desconectar dos recursos | BORRAR /videos/[video\_id] elimina el video [video\_id] |

**Fuente:** (Vimeo developer, 2015)

* **Endpoint**

Un endpoint de la API es un camino que identifica de forma exclusiva un recurso Vimeo. Para hacer una petición HTTP a un endpoint, agregue este camino hasta el final de https://api.vimeo.com. La respuesta desde un endpoint será JSON, y contiene uno o más recursos.

* **Parámetros**

La mayoría de los endpoints de la API soportan parámetros de solicitud adicionales. Estos parámetros dependen del método HTTP y endpoint de la API.

GET

Todas las solicitudes GET deben anexar los parámetros para el endpoint que es para cualquier estándar de componente de consulta.



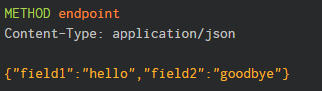
**Figura 46-1**. Parámetros de método GET

**Fuente:** (Vimeo developer, 2015)

**POST, PUT, DELETE, PATCH**

Todas las demás solicitudes deben proporcionar los parámetros a través del cuerpo. Por defecto se asume que está enviando el valor como application/x-www-form-urlencoded.

También se acepta JSON si se proporciona un encabezado Content-Type de application/json, como se puede observar en la Figura 47-1.



**Figura 47-1.** Parámetros de método GET

Fuente: (Vimeo developer, 2015)

* **Establecer Versión**

La API de Vimeo ofrece versiones para seguir mejorando la API a través del tiempo sin romper las aplicaciones de los usuarios existentes. Mientras se proporciona explícitamente un número de versión con cada solicitud de API.

Se indica la versión de la API a través de la cabecera Accept.

Accept: application/vnd.vimeo.\*+json;version=3.2

La segunda sección de la cabecera accept define el formato de respuesta. Actualmente, sólo JSON es compatible. La tercera sección de la cabecera accept define la versión de la API que espera utilizar. Como la API crece, los cambios no afectarán a las versiones anteriores de la API.

Mediante la combinación del parámetro q y versión se puede solicitar diferentes versiones de diferentes recursos con la misma cabecera Accept.

El siguiente ejemplo volverá vídeos versión 3, y los usuarios en la versión 2 application/vnd.vimeo.user+json;version=2.0,application/vnd.vimeo.\*+json;version=3.0

* **Autenticación**

Todas las solicitudes requieren alguna forma de autenticación. La autenticación se maneja a través de OAuth 2.0, ya sea utilizando tokens de acceso de cliente o tokens de acceso de usuario.

* **Representación**

Si la solicitud API devuelve más de un recurso, el API Vimeo los envuelve en la representación colección. Este envoltorio JSON incluye información útil adicional, como la paginación, contadores y más, detallados en la siguiente Tabla 18-1.

**Tabla 18-1.** Representación de colección

|  |  |  |
| --- | --- | --- |
| **Campo** | **Tipo** | **Descripción** |
| Data | array | Una serie de representaciones de recursos. Este es el contenido principal de la colección. |
| total | number | El monto total de los recursos para este endpoint, resumido en todas las páginas. |
| page | number | La página actual de los recursos. |
| per\_page | number | La cantidad de recursos a incluir en cada página de datos. |
| pagination | object | Un objeto que detalla cómo solicitar páginas adicionales en esta colección. |

**Fuente:** (Vimeo developer, 2015)

* **Errores comunes**

A continuación en la Tabla 19-1 se identifican los diferentes tipos de errores que pueden mostrar las operaciones de Vimeo API.

**Tabla 19-1.** Errores comunes del API de Vimeo

|  |  |
| --- | --- |
| **Código de estado HTTP** | **Explicación** |
| 304 Not Modified | Si ningún elemento se ha seguido desde el proporcionado If-Modified-Since encabezado |
| 400 | Si uno de los parámetros no es válido. |
| 401 Unauthorized | Si no se proporciona ningún token. |
| 403 | Si el elemento no tiene los alcances adecuados. |
| 403 | Si el usuario autenticado no es dueño del elemento |
| 404 No Encontrado | Si el elemento no existe |
| 500 Server Error | Si había un error interno con waldo/solr |

**Fuente:** (Vimeo developer, 2015)

# CAPITULO II

# Marco metodológico

En este capítulo se llevara a cabo el desarrollo de la aplicación Web, seleccionando la metodología de desarrollo ágil denominada Extreme Programming (XP), la misma que cuenta con cuatro etapas que son, planeación, diseño, desarrollo y pruebas. Gracias a que la metodología no es estrictamente fija, las etapas pueden variar de acuerdo al desarrollo de la aplicación, a continuación se detallara cada una de las etapas realizadas para obtener un software de calidad.

## Planeación

La planeación es la etapa inicial, en este punto se identifican las historias de usuario, se llega a un acuerdo en la duración del proyecto y el esquema general de trabajo que se realizara.

### Requerimientos Específicos

En la siguiente Tabla 1-2 se detalla cada uno de los requerimientos del proyecto.

**Tabla 1-2.** Requerimientos específicos

|  |  |  |
| --- | --- | --- |
| **IDENTIFICACIÓN** | **DESCRIPCIÓN** | **TIPO** |
| 1. Ingreso. | Permite el ingreso de datos de usuario a OpenPublish.  Permite el ingreso de videos a OpenPublish y Alavista.tv  Permite el ingreso de cuentas de redes sociales en OpenPublish.  Permite el ingreso de credenciales de las APIs utilizadas. | Entrada |
| 2. Modificación. | Modificar datos de usuario, videos, cuentas de redes sociales, credenciales de las APIs utilizadas. | Entrada |
| 3. Eliminar | Eliminar usuarios, videos, cuentas de redes sociales. | Consulta |
| 4. Buscar | Buscar usuarios, videos, cuentas de redes sociales en OpenPublish.  Buscar videos de acuerdo a diferentes parámetros de búsqueda en los canales de video. | Consulta |
| 5. Salida | Publicar contenido de Alavista.tv en redes sociales.  Publicar páginas Web en redes sociales.  Presentar datos de reproducción en forma de galería multimedia de los contenido de Alavista.tv | Salida |

**Realizado por:** Villa E., Rodriguez E., 2015.

### Factibilidad

En el siguiente estudio de factibilidad se determina los recursos económicos, técnicos y operativos los cuales ayudaran a definir si el proyecto es útil para llevar al negocio hacia el cumplimiento de los objetivos planteados.

De acuerdo al estudio de factibilidad que se realizó sobre el proyecto y los motivos que llevaron a la toma de decisión sobre el proyecto se detallan en el **Anexo A**.

La conclusión que se tomo es iniciar con el desarrollo del proyecto ya que **es** **factible** realizarlo debido a que existen los recursos tanto humanos, materiales y económicos que facilitan la implementación del sistema.

### Estimaciones

En las estimaciones se valora la duración, el tamaño, el esfuerzo así como también el costo los cuales son una referencia que puede o no ser asumidos para la planificación del proyecto.

Las estimaciones se han llevado a cabo en la aplicación **COCOMO** y la cual se puede corroborar en el **Anexo B.** Aunque el tiempo, personal y costos son valores nominales pero sirven de guía para la proyección del sistema.

Con las estimaciones nominales concluimos que el resultado del costo total del proyecto es de $ **7852.46** aunque es un monto solamente de referencia debido a que el personal que desarrollará el proyecto es definido de 2 personas y **COCOMO** nos da un personal nominal de una persona con un tiempo aproximado de 8 meses.

### Historias de Usuario

Las historias de usuario son usadas como una herramienta para dar a conocer los requerimientos del sistema al equipo de desarrollo, mediante pequeños textos, en los cuales el cliente describe las actividades que realizará el sistema, la redacción de las historias de usuario se las realiza bajo la terminología del cliente de forma clara y sencilla, como se detalla en la Tabla 2-2.

**Tabla 2-2.** Historias de usuario

|  |  |  |
| --- | --- | --- |
| **Número** | **Descripción** | **Tipo** |
| HU1 | Permite el ingreso de datos de usuarios a OpenPublish. | Entrada |
| HU2 | Permite el ingreso de credenciales de las APIs utilizadas en OpenPublish. | Entrada |
| HU3 | Permite el ingreso de datos de cuentas de redes sociales en OpenPublish. | Entrada |
| HU4 | Permite buscar videos de acuerdo a diferentes parámetros de búsqueda en YouTube | Consulta |
| HU5 | Permite buscar videos de acuerdo a diferentes parámetros de búsqueda en Vimeo | Consulta |
| HU6 | Permite el ingreso de datos de videos a OpenPublish y Alavista. | Entrada |
| HU7 | Permite modificar los datos de usuarios en OpenPublish. | Entrada |
| HU8 | Permite modificar los datos de las credenciales de las APIs utilizadas en OpenPublish. | Entrada |
| HU9 | Permite modificar los datos de cuentas de la red social Facebook en OpenPublish. | Entrada |
| HU10 | Permite modificar los datos de videos en OpenPublish y Alavista. | Entrada |
| HU11 | Permite buscar usuarios, cuentas de redes sociales en OpenPublish. | Consulta |
| HU12 | Permite eliminar usuarios y videos de OpenPublish y Alavista.tv. | Consulta |
| HU13 | Permite eliminar cuentas de redes sociales de OpenPublish. | Consulta |
| HU14 | Permite publicar contenido de Alavista.tv en la red social Facebook | Salida |
| HU15 | Permite publicar contenido de Alavista.tv en la red social Twitter | Salida |
| HU16 | Permite publicar páginas Web en redes sociales. | Salida |
| HU17 | Permite presentar datos de reproducción en forma de galería multimedia de los contenidos de Alavista.tv. | Salida |

**Realizado por:** Villa E., Rodriguez E., 2015.

### Iteraciones

En la Tabla 3-2 se presenta las iteraciones en las cuales se ha dividido el proyecto.

**Tabla 3-2.** Iteraciones

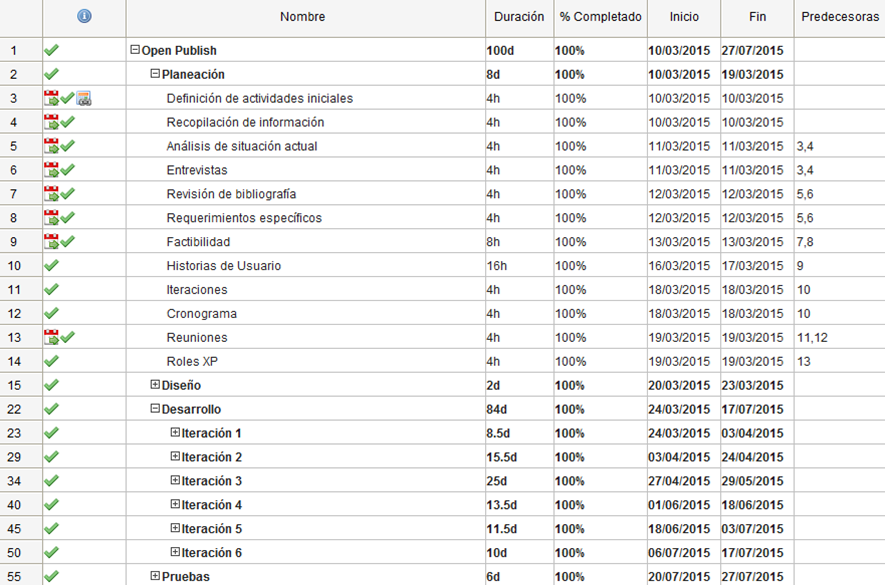
|  |  |  |
| --- | --- | --- |
| Número | Descripción | Tipo |
| Iteración 1 | | |
| HT1 | Diseño de la Arquitectura del sistema | Técnica |
| HT2 | Diseño de la base de datos del sistema | Técnica |
| HT3 | Definir el estándar de interfaz del sistema | Técnica |
| HU1 | Permite el ingreso de datos de usuarios a OpenPublish. | Entrada |
| Iteración 2 | | |
| HU2 | Permite el ingreso de credenciales de las APIs utilizadas en OpenPublish. | Entrada |
| HU3 | Permite el ingreso de datos de cuentas de redes sociales en OpenPublish. | Entrada |
| HU4 | Permite buscar videos de acuerdo a diferentes parámetros de búsqueda en YouTube | Consulta |
| Iteración 3 | | |
| HU5 | Permite buscar videos de acuerdo a diferentes parámetros de búsqueda en Vimeo | Consulta |
| HU6 | Permite el ingreso de datos de videos a OpenPublish y Alavista. | Entrada |
| HU7 | Permite modificar los datos de usuarios en OpenPublish. | Entrada |
| HU8 | Permite modificar los datos de las credenciales de las APIs utilizadas en OpenPublish. | Entrada |
| Iteración 4 | | |
| HU9 | Permite modificar los datos de cuentas de la red social Facebook en OpenPublish. | Entrada |
| HU10 | Permite modificar los datos de videos en OpenPublish y Alavista. | Entrada |
| HU11 | Permite buscar usuarios, cuentas de redes sociales en OpenPublish. | Consulta |
| Iteración 5 | | |
| HU12 | Permite eliminar usuarios y videos de OpenPublish y Alavista.tv. | Consulta |
| HU13 | Permite eliminar cuentas de redes sociales de OpenPublish. | Consulta |
| HU14 | Permite publicar contenido de Alavista.tv en la red social Facebook | Salida |
| Iteración 6 | | |
| HU15 | Permite publicar contenido de Alavista.tv en la red social Twitter. | Salida |
| HU16 | Permite publicar páginas Web en redes sociales. | Salida |
| HU17 | Permite presentar datos de reproducción en forma de galería multimedia de los contenidos de Alavista.tv. | Salida |

**Realizado por:** Villa E., Rodriguez E., 2015.

### Cronograma

En el cronograma se detalla las iteraciones a desarrollar con sus respectivas fechas determinadas, esta planificación se la considerará como tiempo de trabajo ocho horas diarias, días laborales de lunes a viernes, equivalente a trabajar 40 horas en los días mencionados.

Para esta planificación se ha utilizado el servicio Web gratuito Gannteren donde se detalla las fases del sistema a desarrollar de acuerdo a las fases del modelo XP, como se puede observar en la Figura 1-2. De esta manera concluyendo que el sistema tendrá inicio el **10 de marzo del 2015** y culminaráel **27 de julio de 2015 (aproximadamente 4 meses).**



**Figura 1-2.** Cronograma de Actividades

**Realizado por:** Villa E., Rodriguez E., 2015

### Velocidad del Proyecto

La velocidad del proyecto sirve de ayuda para estimar la cantidad de historias de usuario a implementar en una iteración, y estimar el tiempo que se podrá tomar en desarrollar la siguiente iteración, para esto se tomará en cuenta que un punto estimado tendrá el valor de 4 horas.

Calculo de un punto estimado

* Horas diarias laborables (HDL) = 8
* Dias Laborables (DL) = 5
* Total Horas Laborables Semanales (THLS)
* Hora Minima Laborable (HmL) = 1
* Horas Maximas Laborables (HML) = 8
* Punto Medio (PM)

Un punto estimado debe ser representado por una cantidad entera por lo tanto se acuerda que cada punto estimado debe tener la cantidad de 4 horas, en consecuencia cada dia se laboraran 2 puntos y 10 puntos en una semana, como se detalla en la Tabla 4-2.

**Tabla 4-2.** Velocidad del proyecto

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Iteración 1** | **Iteración 2** | **Iteración 3** | **Iteración 4** | **Iteración 5** | **Iteración 6** |
| Historias de Usuario | 4 | 3 | 4 | 3 | 3 | 3 |
| Puntos Estimados | 17 | 31 | 50 | 27 | 23 | 20 |
| Velocidad del Proyecto | **4/17** | **3/31** | **4/50** | **3/27** | **3/23** | **3/20** |

**Realizado por:** Villa E., Rodriguez E., 2015.

### Reuniones

La planificación es importante en el desarrollo del sistema, por lo tanto es importante realizar reuniones continuas durante todo el ciclo de desarrollo, estas reuniones se llevaran a cabo conjuntamente con el cliente y el equipo de desarrollo.

La primera reunión con el cliente y el equipo de desarrollo se realizó con el objetivo de determinar los requerimientos del sistema. La segunda reunión con el cliente se lo realizo para culminar de analizar los requerimientos del sistema, y fijar las fechas de entrega para cada iteración.

De la misma manera se realizara reuniones con el cliente al momento de la entrega de cada iteración para realizar las pruebas correspondientes, estas reuniones se las realizará mediante Skype debido a la complejidad de realizar una reunión personalmente.

Las reuniones diarias del equipo de desarrollo se las realizara para dar a conocer el estado en el que se encuentra el desarrollo de la iteración, así también para solucionar algún problema que pueda surgir, esta reunión no debe tomar demasiado tiempo.

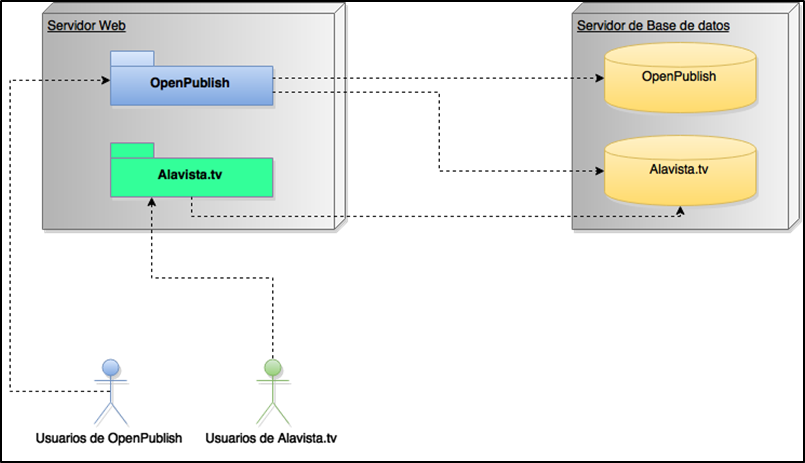
## Diseño

### Metáforas del sistema

Las metáforas del sistema hacen referencia al funcionamiento que tendrá el sistema, detallando la arquitectura del sistema, diseño de la base de datos, definición de la interfaz de usuario y el estándar de codificación.

#### Definición de la arquitectura del sistema

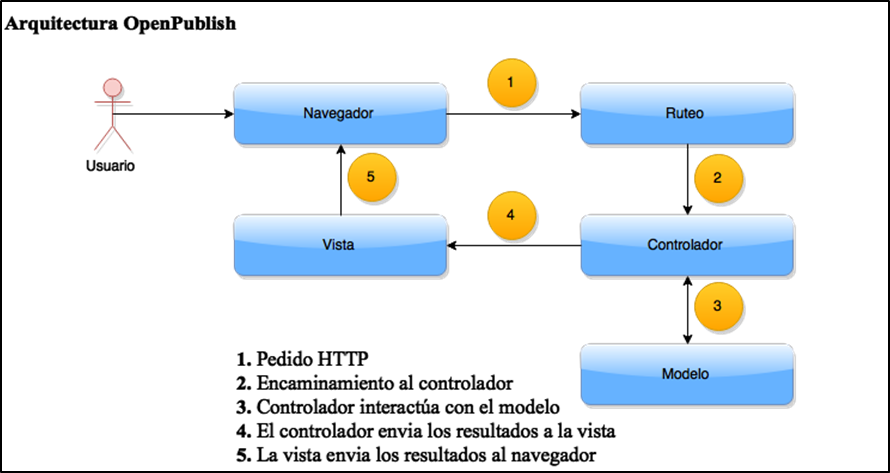
Para el desarrollo de la aplicación se ha considerado la arquitectura cliente servidor, en la Figura 2-2 se detalla además como se relacionara la aplicación con la base de datos de la página Web de Alavista.tv.



**Figura 2-2.** Arquitectura del sistema completo

**Realizado por:** Villa E., Rodriguez E., 2015

Detallando el funcionamiento de la aplicación OpenPublish, en la Figura 3-2 se describe la manera de interacción entre el modelo, vista y el controlador. Arquitectura que se implementa con la ayuda del Framework de desarrollo.



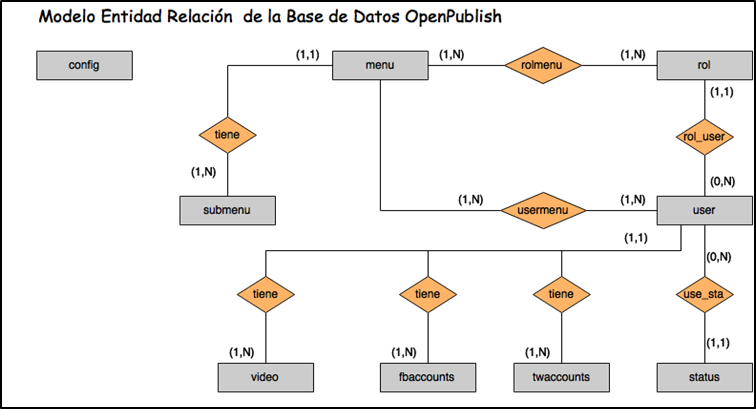
**Figura 3-2.** Arquitectura de la aplicación OpenPublish

**Realizado por:** Villa E., Rodriguez E., 2015

#### Diseño de la base de datos del sistema

En la Figura 4-2 se presenta el diseño de la base de datos OpenPublish

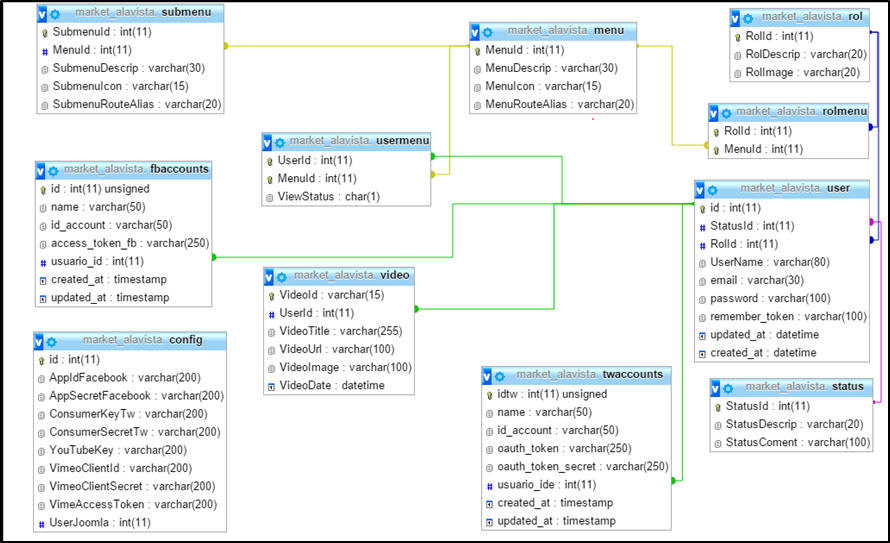
Modelo Entidad relación.



**Figura 4-2.** Modelo Entidad Relación de la base de datos

**Realizado por:** Villa E., Rodriguez E., 2015

Modelo Fisico.



**Figura 5-2.** Modelo Físico de la base de datos

**Realizado por:** Villa E., Rodriguez E., 2015

Tablas utilizadas de la base de datos de Alavista.tv



**Figura 6-2.** Tablas utilizadas de la base de datos de alavista.tv

**Realizado por:** Villa E., Rodriguez E., 2015

#### Diccionario de datos

El diccionario de datos es un conjunto de metadatos donde se detalla las características lógicas y puntuales de los datos que se van a utilizar en el sistema, se detalla también las entidades de la base de datos y sus atributos correspondientes con una pequeña descripción del mismo.

A continuación en la Tabla 5-2 se detalla el diccionario de datos de la tabla Config de la Base de datos OpenPublish, en el **Anexo C** se detallara las demás tablas.

**Tabla 5-2.** Diccionario de datos Tabla Config

|  |  |  |  |
| --- | --- | --- | --- |
| **Columna** | **Tipo** | **Nulo** | **Comentario** |
| Id | int(11) | No | Clave única. Identificación de las APIs |
| AppIdFacebook | varchar(200) | No | Identificador de la aplicación de Facebook. |
| AppSecretFacebook | varchar(200) | No | Identificador secreto de la aplicación de Facebook. |
| ConsumerKeyTw | varchar(200) | No | Identificador de la aplicación de Twitter. |
| ConsumerSecretTw | varchar(200) | No | Identificador secreto de la aplicación de Twitter. |
| YouTubeKey | varchar(200) | No | Identificador de la aplicación de YouTube. |
| VimeoClientId | varchar(200) | No | Identificador de la aplicación de Vimeo. |
| VimeoClientSecret | varchar(200) | No | Identificador secreto de la aplicación de Vimeo. |
| VimeAccessToken | varchar(200) | No | Identificador de acceso de la aplicación de Vimeo. |
| UserJoomla | int(11) | No | Identificador del usuario de Joomla(Alavista.tv) |

**Realizado por:** Villa E., Rodriguez E., 2015.

### Estándar de Codificación

Para el desarrollo del proyecto se ha considerado tomar en cuenta como estándar de codificación PSR-0 y PSR-1, los cuales se encuentran integrados en el Framework Laravel. (Laravel, 2015, http://laravel.com/docs/4.2/)

Entre las características principales de estos estándares son:

* Funciones y estructuras de control deben utilizar llaves estilo Allman
* Sangría con pestañas, alinee con espacios
* Caracteres alfabéticos para los nombres de clases, pueden combinar mayúsculas y minúsculas
* Los archivos deben utilizar solamente <? php y <? = etiquetas
* Los archivos deben utilizar solamente UTF-8 para el código PHP
* Para nombrar los controladores se agregara al final la palabra Controller
* Los métodos se declaran con el estilo CamelCase
* Los nombres de las clases deben ser declarados en StudlyCaps
* Los modelos se declaran de acuerdo al nombre de la tabla a la que hacen referencia con la letra principal en mayúscula

### Estándar de Interfaces del sistema

El estándar de interfaces permite establecer estándares generales para el diseño de los componentes gráficos en la aplicación a desarrollar, en la Figura 7-2 se detalla la pantalla principal.

**Pantalla Principal**

* Logo.- contiene el logo de OpenPublish, se en la parte superior izquierda, la cual permitirá dirigirse a la pantalla de inicio
* Des.-permite desplazar el menú para mostrarlo u ocultarlo, su ubicación estará al lado derecho del Logo
* Menú.- contiene el menú del sistema, su ubicación será en la parte izquierda bajo Logo
* Contenido.- será el área en la cual se mostrará las diferentes acciones que se pueden realizar
* ID.- Identificación del usuario que ingresa al sistema, su ubicación será la parte superior derecha



**Figura 7-2**.Interfaz Pantalla Principal

**Realizado por:** Villa E., Rodriguez E., 2015

En el **Anexo D** se muestra el estándar de interfaces de Ingreso y Modificación, Búsqueda, Eliminación, Publicación, Galería.

### Tareas de ingeniería

Son utilizadas para describir las tareas que se realizaran en el proyecto, estas tareas se encuentran relacionadas con cada una de las historias de usuario, así también cada una de estas tareas cuenta con las actividades que se deben llevar a cabo para cumplir con la tarea.

En una tarea se especifica la historia de usuario a la que corresponde, nombre de la tarea, tipo, puntos estimados, fecha de inicio y fin de la tarea, nombre del programador responsable, descripción de lo que realizara la tarea, y las pruebas de aceptación.

El modelo para redactar cada una de las tareas se presenta a continuación en la Tabla 6-2, las tareas se especificaran de acuerdo a cada una de las historias de usuario en el **Anexo E**

**Tabla 6-2.** Modelo de las tareas de ingeniería

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario**: | |
| **Número tarea:** | **Nombre de la tarea:** |
| **Tipo de tarea:** | **Puntos estimados:** |
| **Fecha de inicio:** | **Fecha fin:** |
| **Programador responsable:** | |
| **Descripción:** | |
| **Pruebas de aceptación:** | |

**Realizado por:** Villa E., Rodriguez E., 2015.

### Tarjetas de clase, responsabilidad, colaboración (CRC cards)

Las clases tienen una funcionalidad indispensable en el desarrollo del proyecto, proporcionar un enfoque orientado a objetos de todas las clases que se utilizaran en el desarrollo del mismo. Una tarjeta CRC está compuesta de un nombre, funciones, métodos que constaran la clase, y si es el caso de ser necesario el nombre de clases que depende para poder llevar acabo esas funciones.

En el **Anexo F** se puede evidenciar cada una de las clases necesarias para el desarrollo, en la Tabla 7-2 se presenta el modelo que se utilizara para redactar las tarjetas.

**Tabla 7-2.** Modelo de la tarjetas CRC

|  |  |
| --- | --- |
| Nombre de la Clase | |
| Responsabilidades | Dependencias, Colaboradores |

**Realizado por:** Villa E., Rodriguez E., 2015.

### Soluciones puntuales (Spike Solution)

Se trata de una solución rápida a un problema que se presenta en el desarrollo de la aplicación, esta investigación para resolver el problema presentado se lo realiza de manera aislada al proyecto, con el objetivo de evitar retrasos en el desarrollo. (Extreme Programming.org, 1999, http://www.extremeprogramming.org)

En el proyecto se realizó ocho spike solution los cuales se delegaron de manera alterna al equipo de desarrollo, estas se encuentran detalladas en las actividades correspondientes a cada tarea de ingeniería.

### Roles XP

Tomando en cuenta que el equipo de desarrollo del proyecto solo consta de dos personas, los cargos recayeron sobre estas dos personas y también el asesor del proyecto, detalladas en la Siguiente Tabla 8-2.

**Tabla 8-2.** Roles XP

|  |  |
| --- | --- |
| **Cliente** | Ing. Mario Murillo MSC. |
| **Líder de proyecto** | Edwin Villa |
| **Administrador Base de Datos** | Edison Rodriguez |
| **Tester** | Edison Rodriguez |
| **Desarrolladores** | Edison Rodriguez  Edwin Villa |
| **Asesor** | Dr. Julio Santillán  Ing. Fernando Proaño |

**Realizado por:** Villa E., Rodriguez E., 2015.

## Desarrollo

Con el análisis de los requerimientos adquiridos en la etapa de planificación se realiza unadefinición general de la aplicación y se organiza el desarrollo con laestimación de la duración del proyecto.

A partir del plan de entregas inicial, se define en cada iteración una planificaciónespecífica paralas historias de usuario asignadas, actualizando la estimación deltiempo de implementación y prioridad en caso de requerirse.

**Prioridad**

Laprioridad es el nivel de importancia de la historia de usuario se calificara como 1 la prioridad más alta y como 4 la prioridad más baja

En la Tabla 9-2 se presenta el resumen del análisis realizado por parte del equipo de desarrollo (estimación del tiempo) y el cliente (prioridad), de las historias de usuario, para esto se tomará en cuenta que un punto estimado tendrá el valor de 4 horas.

**Tabla 9-2.** Historias de usuario priorizadas

|  |  |  |  |
| --- | --- | --- | --- |
| Número | Descripción | Prioridad | Estimación |
| Iteración 1 | | |  |
| HT1 | Diseño de la Arquitectura del sistema | 1 | 1 |
| HT2 | Diseño de la base de datos del sistema | 1 | 1 |
| HT3 | Definir el estándar de interfaz del sistema | 1 | 4 |
| HU1 | Permite el ingreso de datos de usuarios a OpenPublish. | 1 | 10 |
| Iteración 2 | | |  |
| HU2 | Permite el ingreso de credenciales de las APIs utilizadas en OpenPublish. | 1 | 3 |
| HU3 | Permite el ingreso de datos de cuentas de redes sociales en OpenPublish. | 1 | 8 |
| HU4 | Permite buscar videos de acuerdo a diferentes parámetros de búsqueda en YouTube | 1 | 19 |
| Iteración 3 | | |  |
| HU5 | Permite buscar videos de acuerdo a diferentes parámetros de búsqueda en Vimeo | 1 | 20 |
| HU6 | Permite el ingreso de datos de videos a OpenPublish y Alavista. | 1 | 20 |
| HU7 | Permite modificar los datos de usuarios en OpenPublish. | 2 | 5 |
| HU8 | Permite modificar los datos de las credenciales de las APIs utilizadas en OpenPublish. | 2 | 4 |
| Iteración 4 | | |  |
| HU9 | Permite modificar los datos de cuentas de la red social Facebook en OpenPublish. | 2 | 8 |
| HU10 | Permite modificar los datos de videos en OpenPublish y Alavista. | 2 | 10 |
| HU11 | Permite buscar usuarios, cuentas de redes sociales en OpenPublish. | 2 | 8 |
| Iteración 5 | | |  |
| HU12 | Permite eliminar usuarios y videos de OpenPublish y Alavista.tv. | 2 | 8 |
| HU13 | Permite eliminar cuentas de redes sociales de OpenPublish. | 2 | 4 |
| HU14 | Permite publicar contenido de Alavista.tv en la red social Facebook | 2 | 10 |
| Iteración 6 | | |  |
| HU15 | Permite publicar contenido de Alavista.tv en la red social Twitter. | 2 | 10 |
| HU16 | Permite publicar páginas Web en redes sociales. | 3 | 4 |
| HU17 | Permite presentar datos de reproducción en forma de galería multimedia de los contenidos de Alavista.tv. | 4 | 4 |

**Realizado por:** Villa E., Rodriguez E., 2015.

**Control**

El control de la planificación, permite el monitoreo del desarrollo del proyecto, mediante la comparación de la planificación de cada iteración con su tiempo de desarrollo ideal. Permitiendo que el cliente y el equipo de desarrollo cuente con información actualizada, al realizar la revisión de estimaciones y priorizaciones de las historias de usuario.

### Iteración 1

**Tabla 10-2.** Iteración 1

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Número | Prioridad | Estimación | Real | Estimación vs Real | Estado |
| HT1 | 1 | 1 | 1 | --- | Aceptada |
| HT2 | 1 | 1 | 1 | --- | Aceptada |
| HT3 | 1 | 4 | 4 | --- | Aceptada |
| HU1 | 1 | 10 | 10 | --- | Aceptada |

**Realizado por:** Villa E., Rodriguez E., 2015.

La iteración 1 cuenta con 3 Historias Técnicas y una Historia de Usuario, las cuales tuvieron una buena estimación, teniendo cada una de las historias el estado final de Aceptada, presentadas en la Tabla 10-2.

### Iteración 2

**Tabla 11-2.** Iteración 2

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Número | Prioridad | Estimación | Real | Estimación vs Real | Estado |
| HU2 | 1 | 3 | 3 | --- | Aceptada |
| HU3 | 1 | 8 | 10 | Subestimado | Aceptada |
| HU4 | 1 | 19 | 17 | Sobreestimado | Aceptada |

**Realizado por:** Villa E., Rodriguez E., 2015.

La iteración 2 cuenta con 3 Historias de Usuario, de las cuales la HU2 tuvo una buena estimación, la HU3 y HU4 fueron subestimada y sobreestimada respectivamente, cada una de las historias tiene el estado final de Aceptada, presentadas en la Tabla 11-2.

### Iteración 3

**Tabla 12-2.** Iteración 3

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Número | Prioridad | Estimación | Real | Estimación vs Real | Estado |
| HU5 | 1 | 20 | 18 | Sobreestimado | Aceptada |
| HU6 | 1 | 20 | 19 | Sobreestimado | Aceptada |
| HU7 | 2 | 5 | 5 | --- | Aceptada |
| HU8 | 2 | 4 | 4 | --- | Aceptada |

**Realizado por:** Villa E., Rodriguez E., 2015.

La iteración 3 cuenta con 4 Historias de Usuario, de las cuales las HU5 y HU6 fueron sobrestimadas, la HU7 y HU8 tuvieron una buena estimación, cada una de las historias tiene el estado final de Aceptada, como se puede observar en la Tabla 12-2.

### Iteración 4

**Tabla 13-2.** Iteración 4

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Número | Prioridad | Estimación | Real | Estimación vs Real | Estado |
| HU9 | 2 | 8 | 8 | --- | Aceptada |
| HU10 | 2 | 10 | 8 | Sobreestimado | Aceptada |
| HU11 | 2 | 8 | 6 | Sobreestimado | Aceptada |

**Realizado por:** Villa E., Rodriguez E., 2015.

La iteración 4 cuenta con 3 Historias de Usuario, de las cuales la HU9 tuvo una buena estimación, la HU10 y HU11 fueron sobreestimadas, cada una de las historias tiene el estado final de Aceptada, como se observa en la Tabla 13-2.

### Iteración 5

**Tabla 14-2.** Iteración 5

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Número | Prioridad | Estimación | Real | Estimación vs Real | Estado |
| HU12 | 2 | 8 | 7 | Sobreestimado | Aceptada |
| HU13 | 2 | 4 | 5 | Subestimado | Aceptada |
| HU14 | 2 | 10 | 12 | Subestimado | Aceptada |

**Realizado por:** Villa E., Rodriguez E., 2015.

La iteración 5 cuenta con 3 Historias de Usuario, de las cuales la HU12 fue sobreestimada, la HU13 y HU14 tuvieron fueron subestimadas, cada una de las historias tiene el estado final de Aceptada, como se observa en la Tabla 14-2.

### Iteración 6

**Tabla 15-2.** Iteración 6

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Número | Prioridad | Estimación | Real | Estimación vs Real | Estado |
| HU15 | 2 | 10 | 12 | Subestimado | Aceptada |
| HU16 | 3 | 4 | 4 | --- | Aceptada |
| HU17 | 4 | 4 | 3 | Sobreestimado | Aceptada |

**Realizado por:** Villa E., Rodriguez E., 2015.

La iteración 6 cuenta con 3 Historias de Usuario, de las cuales la HU15 fue subestimada, la HU16 tuvo una buena estimación y HU17 fue sobrestimada, cada una de las historias tiene el estado final de Aceptada, como se observa en la Tabla 15-2.

## Pruebas

Para poder realizar las pruebas de funcionalidad de OpenPublish se realizó la integración con la aplicación Web Perspectivas.

Perspectivas es la revista científica de la Facultad de Informática y Electrónica de la Escuela Superior Politécnica de Chimborazo la cual tiene como Presidente Editorial de la FIE al Ing. Ms.C. Fernando Proaño, en esta revista se puede ver artículos científicos, investigaciones de docentes que tienen el auspicio de Instituciones del Estado y privadas, además se ha incursionado en revisar y aprobar los mejores trabajos de tesis de grado en todas las áreas de estudio de la FIE, en la Figura 8-2 se puede observar su página principal.

La revista Perspectivas se encuentra administrada mediante en gestor de contenidos Joomla, la versión de dicho gestor de contenido es la 1.5 la cual no permite instalar Contus HD Video Share.

Contus HD Video Share se la puede instalar en Joomla 1.6, 1.7, 2.5 y 3.x, por lo cual se procedió a actualizar a la versión 2.5, la cual brinda mejor seguridad, una mejora estructura en su base de datos, entre otro beneficios.



**Figura 8-2.** Revista Perspectivas

**Fuente:** (Perspectivas, 2015)

Las pruebas de funcionalidad realizadas con la integración antes mencionada se realizó satisfactoriamente y cuyas pruebas se a continuación.

### Pruebas de Funcionalidad

Para llevar a cabo las pruebas se realizaron las siguientes tareas:

* Ingreso al sistema
* Administración de Usuarios
* Selección de un estado, rol, funciones
* Editar usuarios
* ConFiguración de credenciales
* Administración de videos
* Buscar videos
* Editar datos de videos
* Selección de videos a guardar
* Eliminar videos
* Enviar notificación
* Administración Redes sociales
* Agregar cuentas
* Publicar videos
* Publicar páginas
* Galería
* Listar videos
* Reproducir videos

Las pruebas de aceptación se las realiza por cada historia de usuario realizadas, las cuales permiten confirmar que la historia de usuario se ha implementado correctamente, en el caso de presentarse algún problema se realiza refactorización de la misma hasta que se obtenga la aceptación de la prueba.

### Ámbito de Pruebas

A continuación se detalla en las siguientes tablas el equipo cliente que se utilizó para realizar las pruebas de funcionamiento de la aplicación.

**Hardware**

**Tabla 16-2.** Recursos Cliente

|  |  |
| --- | --- |
| **Característica** | **Descripción** |
| Procesador | Intel(R) Core(TM) i5-2410M CPU @ 2.30GHz |
| **Memoria** | **6 GB** |

**Realizado por:** Villa E., Rodriguez E., 2015.

**Tabla 17-2.** Recursos Servidor

|  |  |
| --- | --- |
| **Característica** | **Descripción** |
| Procesador | Intel(R) Core(TM) i7-2410M 2.30GHz |
| Memoria | 8 GB |
| Servidor Web | IIS |
| Servidor Base de Datos | MySql |

**Realizado por:** Villa E., Rodriguez E., 2015.

Los datos de la velocidad de Internet fueron obtenidos mediante la herramienta Web SpeedTest, detallada en la Tabla 18-2.

**Tabla 18-2.** Recursos Red

|  |  |
| --- | --- |
| **Característica** | **Descripción** |
| Velocidad de Subida | 2.11 Mbps |
| Velocidad de Bajada | 0.33 Mbps |

**Realizado por:** Villa E., Rodriguez E., 2015.

# CAPITULO III

# Marco de resultados, anÁlisis y discusiÓn de resultados

En este capítulo se describe los resultados obtenidos y se analiza la aplicación Web si mejoró la inyección de contenido y la publicación en medios sociales.

**Tiempo de Inyección de Contenido**

A continuación se muestra los criterios de evaluación para realizar el proceso de inyección de contenido al VCMS, donde se tiene los siguientes campos

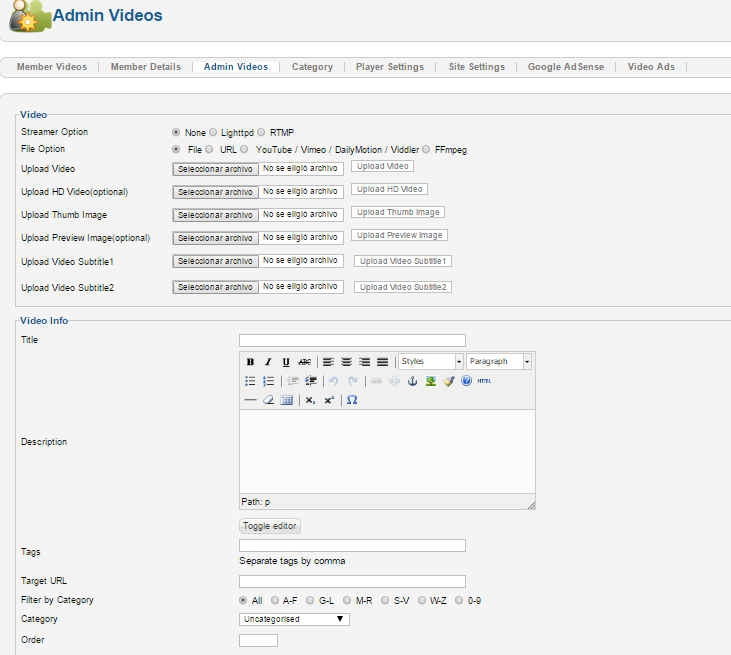
* Cantidad de Videos.- número de videos a inyectar en el VCMS
* Cantidad de Publicaciones.- número de publicaciones realizadas en los medios sociales
* OpenPublish.- tiempo que se utiliza, haciendo uso de la aplicación Web
* Manualmente.-tiempo que se utiliza sin la aplicación Web

Para poder obtener informacion del tiempo utilizado al realizar los procesos manualmente se utilizo la herramienta Web Cronometro.co localizada en la URL http://www.cronometro.co/, y para obtener el tiempo utilizado con OpenPublish se utilizo la herramienta para desarrolladores de Google Chrome

**YouTube**

Proceso manual para Youtube:

* Buscar un video en youtube.com
* Copiar el URL
* Pegar la URL del video en el VCMS
* Digitar la informacion del video

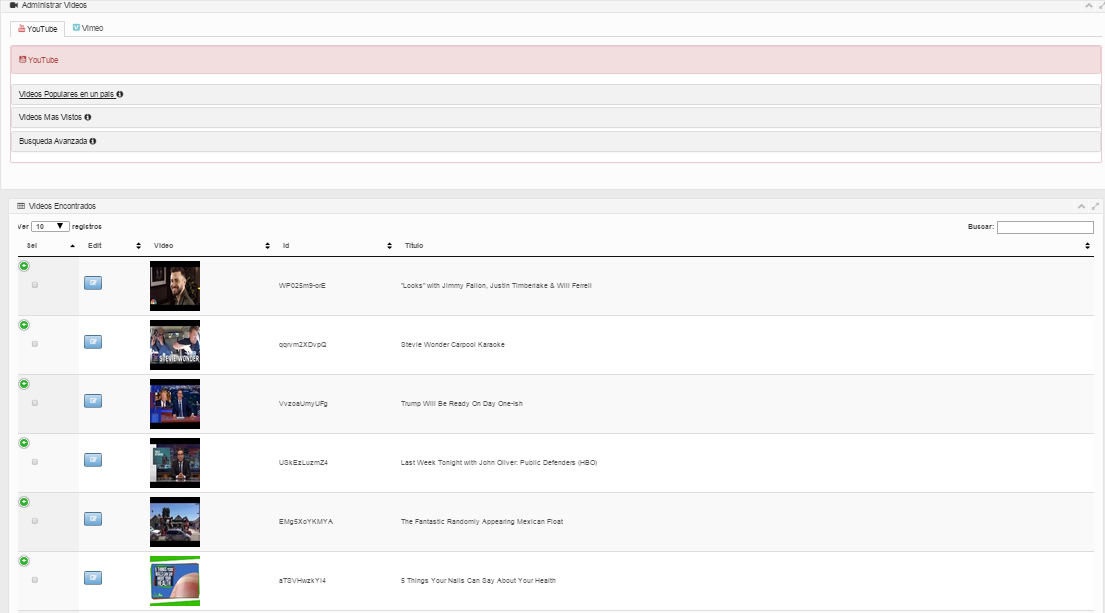


**Figura 1-3.** Interfaz de administración de alavista.tv para YouTube

**Fuente:** Villa E., Rodriguez E., 2015

Open Publish - Youtube:

* Realizar la busqueda mediante los parametros deseados
* Seleccionar los videos deseados y Guardar



**Figura 2-3.** Interfaz de administración de OpenPublish para YouTube

**Fuente:** Villa E., Rodriguez E., 2015

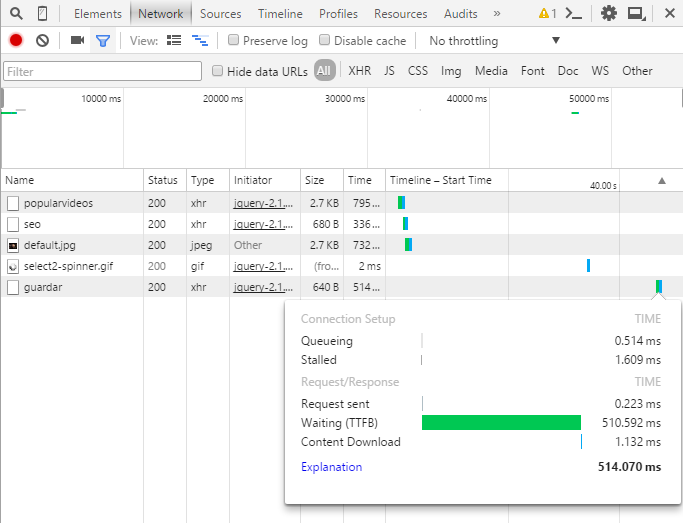
**Toma de tiempo**

En las Figuras 3-3, 4-3,5-3 se observa los tiempos obtenidos para 1 video.



**Figura 3-3.** Tiempo manual para YouTube

**Fuente:** Villa E., Rodriguez E., 2015



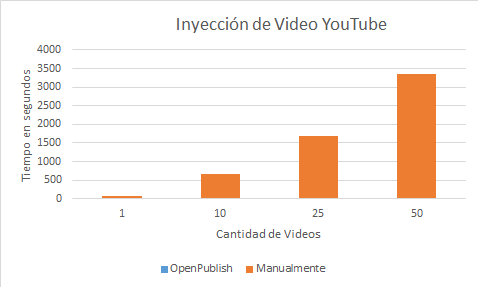
**Figura 4-3.** Tiempo obtenido con OpenPublish para YouTube

**Fuente:** Villa E., Rodriguez E., 2015

**Tabla 1-3.** Datos de prueba en YouTube

|  |  |  |
| --- | --- | --- |
| **Cantidad de Videos** | **OpenPublish** | **Manualmente** |
| 1 | 1,898 s | 66,582 s |
| 10 | 5,125 s | 665,82 s |
| 25 | 9,774 s | 1664,55 s |
| 50 | 16,907 s | 3329,1 s |

**Realizado por:** Villa E., Rodriguez E., 2015.



**Figura 5-3.** Inyección de Contenido de YouTube

**Fuente:** Villa E., Rodriguez E., 2015

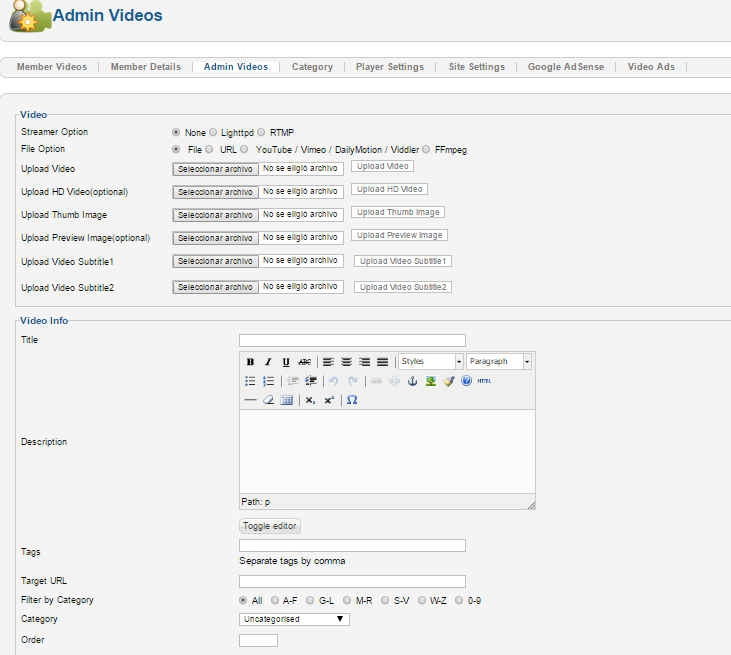
**Interpretación de Datos**

Al utilizar OpenPublish se realiza el proceso de inyección de contenido desde la búsqueda, organización y preparación del contenido obtenido de YouTube y la inyección de contenido en el VCMS dando como resultado un tiempo de 1,898 seg, mientras que al realizarse manualmente conlleva a utilizar un tiempo de 66,582 seg. Obteniendo un ahorro de tiempo de 65,664 seg, esto se puede evidenciar más ampliamente al inyectar un mayor número de contenidos como por ejemplo la cantidad de 50 videos en el cual se obtuvo un ahorro de 3312,19 seg que equivalen a 55,203 min

**Vimeo**

Proceso manual para Vimeo:

* Buscar un video en youtube.com
* Copiar el URL
* Pegar la URL del video en el VCMS
* Digitar la información del video

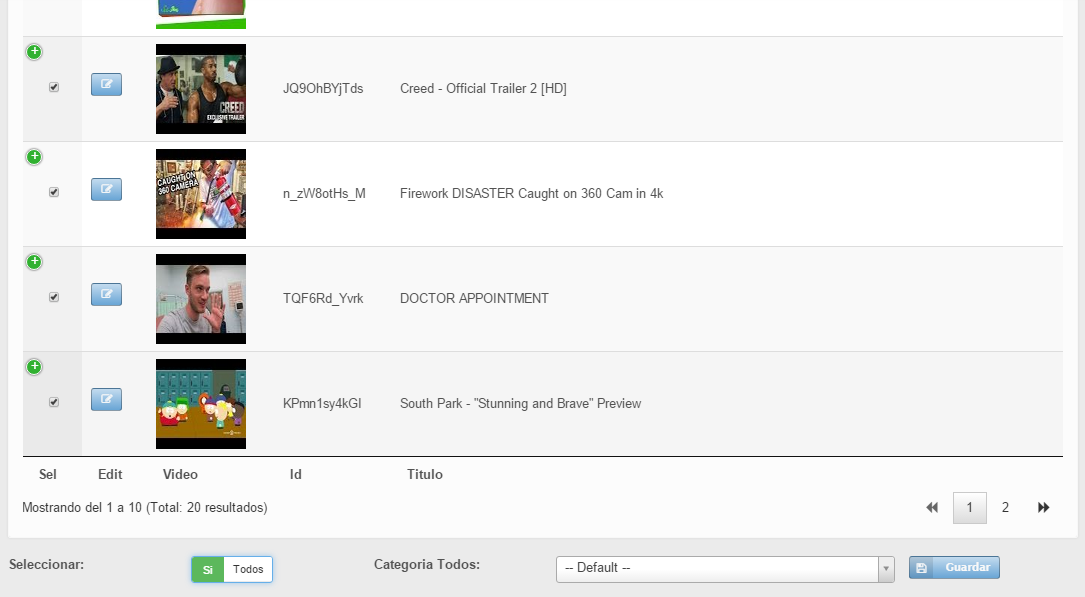


**Figura 6-3.** Interfaz de administración de alavista.tv para Vimeo

**Fuente:** Villa E., Rodriguez E., 2015

Open Publish – Vimeo:

* Realizar la busqueda mediante los parametros deseados
* Seleccionar los videos deseados y Guardar

****

**Figura 7-3.** Interfaz de administración de OpenPublish para Vimeo

**Fuente:** Villa E., Rodriguez E., 2015

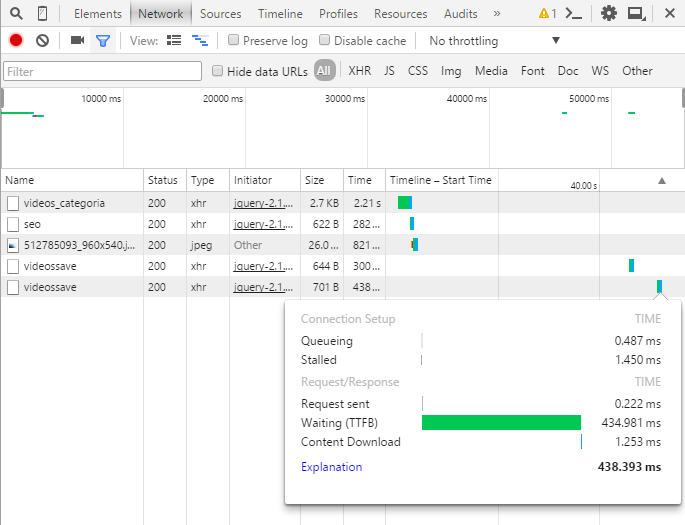
**Toma de tiempo**

En las Figuras 8-3, 9-3,10-3 se observa los tiempos obtenidos para 1 video.



**Figura 8-3.** Tiempo manual para Vimeo

**Fuente:** Villa E., Rodriguez E., 2015

****

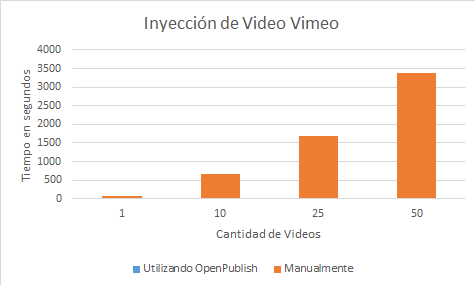
**Figura 9-3.** Tiempo obtenido con OpenPublish para Vimeo

**Fuente:** Villa E., Rodriguez E., 2015

**Tabla 2-3.** Datos de prueba en Vimeo

|  |  |  |
| --- | --- | --- |
| **Cantidad de Videos** | **Utilizando OpenPublish** | **Manualmente** |
| 1 | 2,375 s | 67,028 s |
| 10 | 6,187 s | 670,28 s |
| 25 | 10,001 s | 1675,7 s |
| 50 | 17,825 s | 3351,4 s |

**Realizado por:** Villa E., Rodriguez E., 2015.

****

**Figura 10-3.** Inyección de Contenido de Vimeo

**Fuente:** Villa E., Rodriguez E., 2015

**Interpretación de Datos**

Al utilizar OpenPublish se realiza el proceso de inyección de contenido desde la búsqueda, organización y preparación del contenido obtenido de Vimeo y la inyección de contenido en el VCMS dando como resultado un tiempo de 2,375 seg, mientras que al realizarse manualmente conlleva a utilizar un tiempo de 67,028 seg. Obteniendo un ahorro de tiempo de 64,653 seg, esto se puede evidenciar más ampliamente al inyectar un mayor número de contenidos como por ejemplo la cantidad de 50 videos en el cual se obtuvo un ahorro de 3333,575 seg que equivalen a 55,59 min

**Facebook**

Proceso manual para Facebook:

* Buscar un video en alavista.tv
* Copiar el URL
* Pegar la URL del video en un perfil, grupo, página, o evento
* Digitar la información de la publicación y publicar

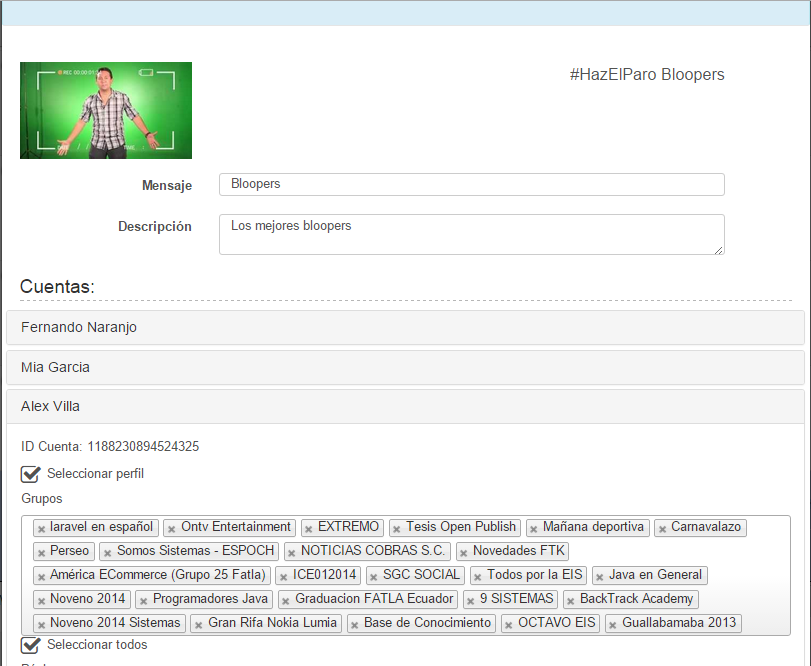


**Figura 11-3.** Publicación en Facebook

**Fuente:** Villa E., Rodriguez E., 2015

Open Publish – Facebook:

* Seleccionar el contenido a publicar
* Seleccionar perfil,grupos, paginas y eventos
* Digitar la información de la publicación y publicar



**Figura12-3.** Publicación en Facebook con OpenPublish

**Fuente:** Villa E., Rodriguez E., 2015

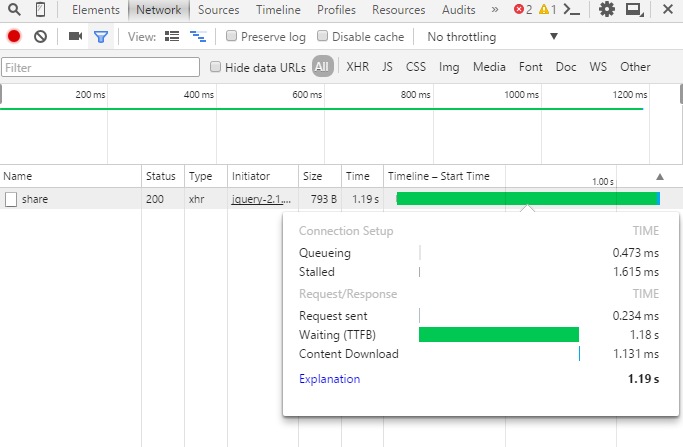
**Toma de tiempo**

En las Figuras 13-3, 14-3,15-3 se observa los tiempos obtenidos para 1 publicación.



**Figura 13-3.** Tiempo manual para Facebook

**Fuente:** Villa E., Rodriguez E., 2015



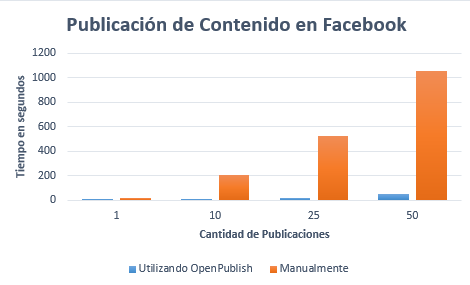
**Figura 14-3.** Tiempo obtenido con OpenPublish para Facebook

**Fuente:** Villa E., Rodriguez E., 2015

**Tabla 3-3.** Datos de prueba en Facebook

|  |  |  |
| --- | --- | --- |
| **Cantidad de Publicaciones** | **Utilizando OpenPublish** | **Manualmente** |
| 1 | 3,95 s | 21,022 s |
| 10 | 10,81 s | 210,22 s |
| 25 | 23,53 s | 525,55 s |
| 50 | 48,28 s | 1051,01 s |

**Realizado por:** Villa E., Rodriguez E., 2015.



**Figura 15-3.** Publicación de Contenido en Facebook

**Fuente:** Villa E., Rodriguez E., 2015

**Interpretación de Datos**

Al utilizar OpenPublish se realiza el proceso de publicación de contenido desde la selección y preparación del contenido a compartirse en la red social Facebook, dando como resultado un tiempo de 3,95 seg, mientras que al realizarse manualmente conlleva a utilizar un tiempo de 21,022 seg. Obteniendo un ahorro de tiempo de 17,072 seg, esto se puede evidenciar más ampliamente al publicar en un mayor número de páginas de Facebook (perfil, grupos, páginas, eventos) como por ejemplo la cantidad de 50 publicaciones en el cual se obtuvo un ahorro de 1002,73 seg que equivalen a 16,712 min

**Twitter**

Proceso manual para Twitter:

* Buscar un video en alavista.tv
* Copiar el URL
* Pegar la URL del video en un perfil o tendencia
* Digitar la información de la publicación y publicar

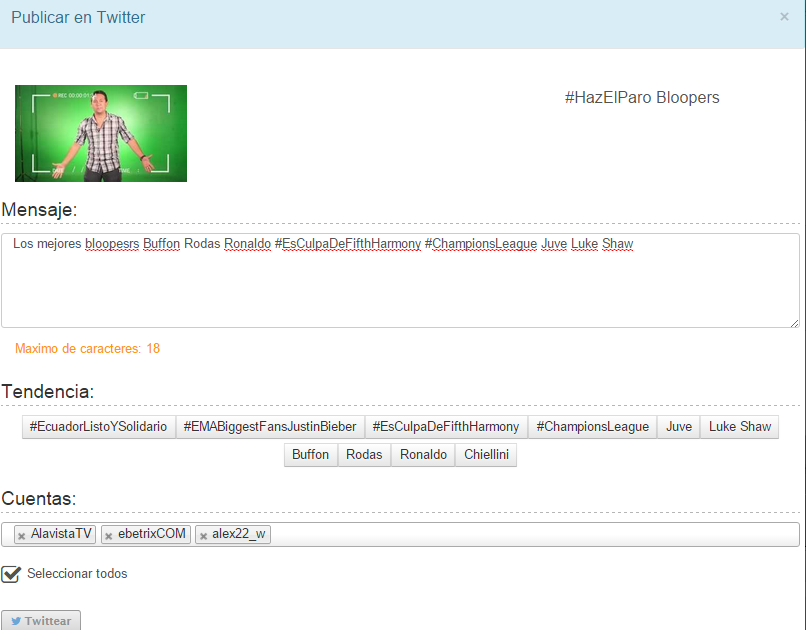
****

**Figura 16-3.** Publicación en Twitter

**Fuente:** Villa E., Rodriguez E., 2015

**Open Publish – Twitter:**

* Seleccionar el contenido a publicar
* Seleccionar tendencias y cuentas
* Digitar la información de la publicación y publicar

****

**Figura 17-3.** Publicación en Twitter con OpenPublish

**Fuente:** Villa E., Rodriguez E., 2015

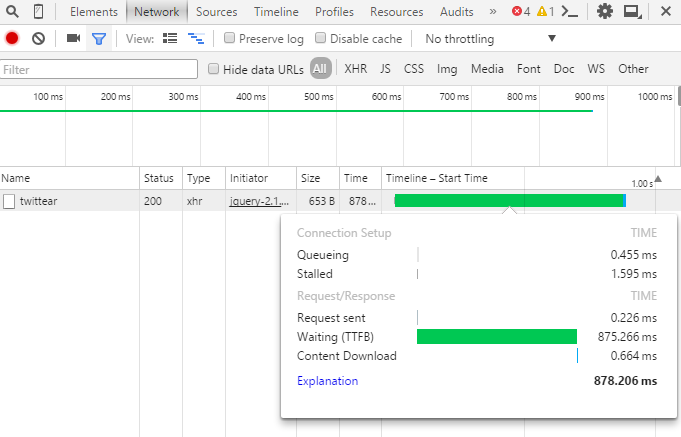
**Toma de tiempo**

En las Figuras 18-3, 19-3,20-3 se observa los tiempos obtenidos para 1 publicación.



**Figura 18-3.** Tiempo manual para Twitter

**Fuente:** Villa E., Rodriguez E., 2015

****

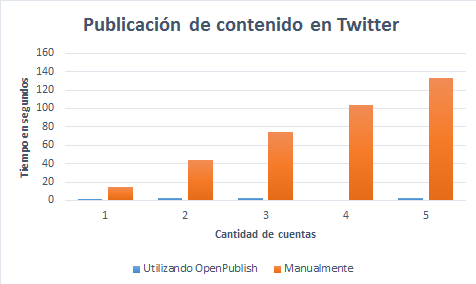
**Figura 19-3.** Tiempo obtenido con OpenPublish para Twitter

**Fuente:** Villa E., Rodriguez E., 2015

**Tabla 4-3.** Datos de prueba en Twitter

|  |  |  |
| --- | --- | --- |
| **Cantidad de Cuentas** | **Utilizando OpenPublish** | **Manualmente** |
| 1 | 0,899 s | 14,607 s |
| 2 | 2,16 s | 44,214 s |
| 3 | 2,24 s | 73,821 s |
| 4 | 2.31 s | 103,428 s |
| 5 | 2,97 s | 133,36 s |

**Realizado por:** Villa E., Rodriguez E., 2015.

****

**Figura 20-3.** Publicación de Contenido en Twitter

**Fuente:** Villa E., Rodriguez E., 2015

**Interpretación de Datos**

Al utilizar OpenPublish se realiza el proceso de publicación de contenido desde la selección y preparación del contenido a compartirse en la red social Twitter, dando como resultado un tiempo de 0,899 seg, mientras que al realizarse manualmente conlleva a utilizar un tiempo de 14,607 seg. Obteniendo un ahorro de tiempo de 13,708 seg, esto se puede evidenciar más ampliamente al publicar en un mayor número de cuentas como por ejemplo la cantidad de 5 en el cual se obtuvo un ahorro de 130,39 seg que equivalen a 2,173 min

Es importante mencionar que al tiempo de publicación manual se le agrego el tiempo de 15,1 seg, correspondientes al tiempo que se ocupa para cambiar de cuentas.

Además la publicación de videos mediante la aplicación ha generado un incremento de visitas hacia el canal Alavista.tv evidenciándose en el número de reproducciones en los videos que han sido publicados a diferencia de los que no han sido compartidos

# CONCLUSIONES

Al finalizar el desarrollo de la aplicación y realizar las pruebas correspondientes se ha obtenido las siguientes conclusiones.

* Al realizar la selección del Framework Laravel para el desarrollo de la aplicación no se pretende desplazar a los demás, la selección depende de los necesidades de los desarrolladores y requisitos de la aplicación a desarrollar.
* Para realizar la inyección de contenido en un VCMS basado en Joomla, se necesita consumir los servicios que brindan las APIs de canales de video externos de los cuales se obtiene datos en un formato JSON.
* Al hacer uso de la aplicación se puede evidenciar una reducción de tiempo al inyectar videos, por ejemplo en YouTube al tener la cantidad de 50 videos se tiene una diferencia de 3312,193 segundos con respecto a realizarlo manualmente, de igual manera en Vimeo se tiene una diferencia de 3333,575 segundos
* Utilizando OpenPublish se puede evidenciar una reducción de tiempo al publicar videos, por ejemplo en Facebook al realizar la cantidad de 50 publicaciones se tiene una diferencia de 1002,73 segundos con respecto a realizarlo manualmente, de igual manera en Twitter al realizar una publicación en 5 cuentas se tiene una diferencia de 130.39 segundos
* Luego de la integración de OpenPublish con Perspectivas y Alavista.tv se concluye que la aplicación Web creada se puede integrar con cualquier VCMS basado en Joomla.
* Al utilizar la aplicación OpenPublish se puede ver que se ha mejorado el tráfico Web hacia Alavista.tv mediante la publicación de su contenido en redes sociales.

* Tanto Facebook y Twitter poseen un mayor numero de usuarios en referencia a otras redes sociales, por lo tanto el presente proyecto hace uso de ellas para lograr una mayor difusión de contenido.

# RECOMENDACIONES

* Se recomienda realizar un control continuo a las APIs utilizadas para conocer el estado y los posibles cambios que se podrían realizar para evitar tener problemas en la conFiguración de OpenPublish.
* Para poder construir un VCMS basado en Joomla se recomienda contar con las versiones 1.6, 1.7, 2.5 y 3.x, las cuales permiten agregar el complemento para gestionar videos.
* Se recomienda realizar un análisis previo de requerimientos con el fin de optimizar el tiempo de desarrollo.
* Al publicar contenido en redes sociales se recomienda compartir contenido con intereses afines al público que se pretende alcanzar para así evitar sanciones a la aplicación.
* Para realizar trabajos de manera colaborativa se recomienda utilizar la plataforma de desarrollo colaborativo GitHub.
* Para realizar la gestión de contenidos se ha seleccionado los canales de video de YouTube y Vimeo así como las redes sociales Facebook y Twitter, por ser los medios sociales con mayor aceptación en el público, pero se recomienda estudiar otros posibles candidatos para su integración, utilizando las mismas técnicas u otras que se puedan generar.
* Realizar busqueda de contenidos mediante la integracion de Web semantica, es decir la busqueda por contenido interno del video, utilizando nuevos plugins.

# BIBLIOGRAFÍA

**AcensTechnologies.** *Framework para el desarrollo ágil de aplicaciones.* [En línea], Marzo 2014, pp. 4-6 [Consulta: 18 de Junio de 2015] Disponible en: <https://www.acens.com/wp-content/images/2014/03/frameworks-white-paper-acens-.pdf>

**Argiles S,** *Tráfico: Qué es y de qué forma influye en el éxito de mi Web.* [En línea]. Enero 2013, [Consulta: 18 de Junio de 2015] Disponible en: [http://sergioargiles.com/2013/01/trafico-que-es-y-de-que-forma-influye-en-el-exito-de-mi-Web/](http://sergioargiles.com/2013/01/trafico-que-es-y-de-que-forma-influye-en-el-exito-de-mi-web/)

**Baltazar Paz L.** *Sistemas Gestores de Contenidos, SGC.*  [En línea], 27 de Noviembre 2008, [Consulta: 18 de Junio de 2015] Disponible en: <http://www.enterate.unam.mx/Articulos/2006/octubre/sistemas.htm>

**Baquero I & Cámara M.** *Introducción Joomla!* [En línea], 2014 [Consulta: 20 de Junio de 2015] Disponible en: http://manualesjoomla.es/es/descarga/descarga-patrocinada-1-0.

**Beebon,** *15 Best Free PHP Frameworks of 2015.* [En línea]. Irfan Fauzi, Febrero 16 2015, [Consulta: 19 de Junio de 2015] Disponible en: <http://beebom.com/2015/02/best-free-php-frameworks>.

**Big Eye.** *¿Qué es el Marketing Online?* [En línea], 2015, [Consulta: 28 de Junio de 2015] Disponible en: <http://www.bigeyemarketing.com/empresa/marketing-digital/>

**Cake Software Foundation,** *CakePHP Cookbook Documentation.* [En línea], 01 de Octubre 2005, pp. 8, 2 [Consulta: 20 de Marzo de 2015] Disponible en: <http://book.cakephp.org/2.0/_downloads/es/CakePHPCookbook.pdf>

**Cake Software Foundation.** P*lugins & packages CakePHP.* [En línea], 2015, [Consulta: 12 de Marzo de 2015] Disponible en: <http://plugins.cakephp.org/packages>

**Caraballo Pérez Y.** *La gestión de contenidos en portales Web.*  [En línea], 29 de Enero 2007, [Consulta: 18 de Junio de 2015] Disponible en: <http://bvs.sld.cu/revistas/aci/vol15_3_07/aci07307.htm>

**Centro de Apoyo Tecnologico a Emprendedores, Fundación Parque Científico y Tecnológico de Albacete,** *Estudio de los sistemas de gestión de contenidos Web.*[En línea]. 13 de Julio 2012, pp. 8, 9 [Consulta: 19 de Junio de 2015] Disponible en: [http://www.bilib.es/uploads/media/estudio\_sistemas\_gestion\_contenidos\_Web\_cms.pdf](http://www.bilib.es/uploads/media/estudio_sistemas_gestion_contenidos_web_cms.pdf).

**Codehero,** *Laravel 4 desde Cero: Estructura del Proyecto.* [En línea], Ramses Velazques, 06 de Agosto 2013, [Consulta: 25 de Marzo de 2015] Disponible en: <http://codehero.co/laravel-4-desde-cero-estructura-del-proyecto/>

**Codina L.** *Posicionamiento Web: Conceptos y Ciclo de Vida*. [En línea], 12 de Febrero, p. 3, [Consulta: 18 de Junio de 2015] Disponible en: <http://eprints.rclis.org/9008/1/Posicionamiento_Web_Conceptos_y_Ciclo_de_Vida.pdf>

**Cuerda Garcia X.** *Introducción a los Sistemas de Gestión de Contenidos (CMS) de código abierto.* [En línea], 29 de Noviembre de 2004 [Consulta: 17 de Junio de 2015] Disponible en: <http://mosaic.uoc.edu/2004/11/29/introduccion-a-los-sistemas-de-gestion-de-contenidos-cms-de-codigo-abierto/>

**Degiovannini M.** *Comparativa de Frameworks WEB.* [En línea], Java Hispano, Diciembre 2007, [Consulta: 18 de Junio de 2015] Disponible en: [http://static1.1.sqspcdn.com/static/f/923743/15025206/1320739503647/frameworks\_Web.pdf?token=OARc5pz9pnixJpQ73IcKo%2BBwbLQ%3D](http://static1.1.sqspcdn.com/static/f/923743/15025206/1320739503647/frameworks_web.pdf?token=OARc5pz9pnixJpQ73IcKo%2BBwbLQ%3D)

**Eito Brun R.,** HIPERTEX.NET. *XML y la gestión de contenidos.* [En línea] [Consulta: 29 de Junio de 2015] Disponible en: <http://www.upf.edu/hipertextnet/numero-3/xml.html>.

**Eiton Brun R.** *Gestion de Contenidos.* [En línea], Editorial UOC, Enero de 2013, [Consulta: 17 de Junio de 2015] Disponible en: <https://books.google.com.ec/books?id=XzbaAwAAQBAJ&printsec=frontcover&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false>

**Extreme Programming.org.** *Create a Spike Solution.* [En línea], Don Wells, 1999, [Consulta: 01 de Abril de 2015] Disponible en: <http://www.extremeprogramming.org/rules/spike.html>

**Facebook Developers.** *Facebook Documentación PHP SDK.* [En línea], 2015, [Consulta: 22 de Marzo de 2015] Disponible en: <https://developers.facebook.com/docs/php/gettingstarted/4.0.0>

**Facebook Newsroom.** *Facebook company information. Newsroom.* [En línea], 2015, [Consulta: 28 de Marzo de 2015] Disponible en: <https://newsroom.fb.com/company-info>

**Github .** *Trending repositories.* [En línea], 2015, [Consulta: 21 de Marzo de 2015] Disponible en: <https://github.com/trending?l=PHP&since=monthly>

**Google Developers.** *Getting Started with the YouTube Data API.* [En línea], 2015, [Consulta: 25 de Marzo de 2015] Disponible en: <https://developers.google.com/youtube/v3/getting-started>

**Google Trends.** *Tendencia de busquedas de Framework php.* [En línea], Junio 2015, [Consulta: 19 de Junio de 2015] Disponible en: <http://www.google.es/trends/explore#q=laravel%2C%20CakePHP%2C%20Symfony&date=1%2F2013%2034m&cmpt=q&tz=Etc%2FGMT%2B5>

**Gutiérrez J.** *¿Qué es un framework Web?.* [En línea], 2015, pp. 1, 2, [Consulta: 19 de Junio de 2015] Disponible en: <http://www.lsi.us.es/~javierj/investigacion_ficheros/Framework.pdf>

**Hernández J.** *Análisis y Desarrollo Web*. [En línea], 14 de Diciembre 2014, [Consulta: 17 de Junio de 2015] Disponible en: <https://books.google.com.ec/books?id=nYDVBQAAQBAJ&hl=es&source=gbs_navlinks_s>

**HD Video Share.** *HD Video Share User Manual.* [En línea], Diciembre 2014, [Consulta: 17 de Junio de 2015] Disponible en: https://docs.google.com/viewer?url=https://www.apptha.com/downloadable/download/sample/sample\_id/9/HD%20Video%20Share%20-%20Joomla

**Joomla Organización.** *ayuda.joomlaspanish*. [En línea], [Consulta: 15 de Junio de 2015] Disponible en: <http://ayuda.joomlaspanish.org/que-es-joomla>

**KnpLabs.** KnpBundles. [En línea], 2012 [Consulta: 12 de Marzo de 2015] Disponible en: <http://knpbundles.com>

**Laravel.** *Documentación Laravel*. [En línea], [Consulta: 11 de Junio de 2015] Disponible en: http://laravel.com/docs/4.2/introduction

**Marañon O.** *Comunicación 2.0, visibilidad e interactividad: Fundamentos de la imagen corporativa de las universidades públicas de dmdrid en Youtube* [En línea], 16 de Abril 2012, pp. 119-121 [Consulta: 28 de Junio de 2015] Disponible en: <http://revistas.usal.es/index.php/2172-9077/article/view/12084/12435>

**McCool S.** *Laravel Starter* [En línea], 22 de Noviembre de 2012, pp. 16, 17 [Consulta: 25 de Marzo de 2015] Disponible en: <https://books.google.com.ec/books?id=pIBevFHNX3oC&printsec=frontcover&dq=Laravel&hl=es-.#v=onepage&q=Laravel&f=false>

**Mora Canales R.** *Gestión de contenidos y errores comunes.* [En línea] 23 de Noviembre de 2013, [Consulta: 17 de Junio de 2015] Disponible en: <http://www.adictosaltrabajo.com/tutoriales/visioncms/>

**Niño J.** *Aplicaciones Webs.* [En línea] 2010, pp 56, 57 [Consulta: 18 de Junio de 2015] Disponible en: <https://books.google.com.ec/books?id=UKjEAwAAQBAJ&dq=clasificacion+de+los+gestores+decontenidos&source=gbs_navlinks_s>

**Online Busine School.** *Social Media 2014* [En línea], 2014, pp. 7,8 [Consulta: 28 de Marzo de 2015] Disponible en: <http://mba.americaeconomia.com/sites/mba.americaeconomia.com/files/investigacion_obs._social_media_2014.pdf>

**Omofonmwan N.** *What is Twitter, a social network or a news media?* [En línea] Septiembre 2012, p. 4 [Consulta: 29 de Marzo de 2015] Disponible en: [http://Webuser.hs-furtwangen.de/~heindl/ebte-2012ss/What%20is%20Twitter.pdf](http://webuser.hs-furtwangen.de/~heindl/ebte-2012ss/What%20is%20Twitter.pdf)

**Panadero J.** *Social Marketing: Las Redes Sociales en la Empresa.* [En línea] 22 de Marzo 2012, p. 6 [Consulta: 28 de Junio de 2015] Disponible en: <http://mba.americaeconomia.com/sites/mba.americaeconomia.com/files/social_marketing.pdf>

**Panopto.** *Panopto Recognized as a “Leader” in Gartner’s Magic Quadrant for Enterprise Video Content Management.* [En línea] 17 de Octubre 2014 [Consulta: 17 de Junio de 2015] Disponible en: <http://panopto.com/blog/panopto-recognized-as-a-leader-in-gartners-magic-quadrant-for-enterprise-video-content-management/>

**Panopto.** *Five reasons why eviry cio needs a video content managent system.* [En línea] Noviembre 2014, pp. 6, 7 [Consulta: 17 de Junio de 2015] Disponible en: <http://panopto.com/wp-content/uploads/2014/11/White-Paper-5-Reasons-Why-Every-CIO-Needs-a-Video-Content-Management-System-Panopto-Enterprise-Video.pdf>

**Panopto.** *What is a Video Content Management System (VCMS)?* [En línea], 2015, [Consulta: 17 de Junio de 2015] Disponible en: <http://panopto.com/blog/what-is-a-video-content-management-system-vcms/>

**Pes Rivas C.** *Apuntes del Curso de SEO y Marketing Online*. [En línea], Mayo 2011, p.41 [Consulta: 15 de Junio de 2015] Disponible en: [http://Web.uchile.cl/DctosIntranet/08SEO/BuenasPracticas/CursoSEOyMarketing.pdf](http://web.uchile.cl/DctosIntranet/08SEO/BuenasPracticas/CursoSEOyMarketing.pdf)

**PHP Gang.** *Top 10 PHP frameworks worth looking forward to in 2015.*  [En línea], 13 de Febrero 2015, [Consulta: 19 de Junio de 2015] Disponible en: <http://www.phpgang.com/top-10-php-frameworks-worth-looking-forward-to-in-2015_937.html>

**Potencier F.** *What is Symfony2?.*  [En línea], 25 de Octubre 2011, [Consulta: 20 de Junio de 2015] Disponible en: <http://fabien.potencier.org/what-is-symfony2.html>

**Redes Pymes.** *Publicidad en Redes Sociales; ventajas y desventajas para las empresas.* [En línea], 04 de Diciembre 2014. [Consulta: 18 de Junio de 2015] Disponible en: <http://redespymes.com/2013/12/04/publicidad-en-redes-sociales-ventajas-y-desventajas-para-las-empresas/>

**Ríos D.** *Laravel PHP framework.* [En línea], 22 de Julio 2014, [Consulta: 25 de Junio de 2015] Disponible en: <http://donaldorios.com/2014/07/22/laravel-php-framework/>

**Serrano Cobos J.** *Evolución de los sistemas de gestión de contenidos (CMS), Del mainframe al open source*. [En línea], Junio de 2007 [Consulta: 17 de Junio de 2015] Disponible en: <http://www.academia.edu/471944/Evoluci%C3%B3n_de_los_sistemas_de_gesti%C3%B3n_de_contenidos_CMS_._Del_mainframe_al_open_source>

**Sitepoint.** *Best PHP Framework for 2015 – SitePoint Survey Results.*  [En línea], 28 de Marzo 2015, [Consulta: 19 de Junio de 2015] Disponible en: <http://www.sitepoint.com/best-php-framework-2015-sitepoint-survey-results/>

**Symfony.es.** *¿Qué es Symfony?.* [En línea], 2015. [Consulta: 23 de Junio de 2015] Disponible en: <http://symfony.es/pagina/que-es-symfony/>

**Tramullas J, Garrido P, Sanchez I.** *Funcionalidades de la clasificación en sistemas de gestión de contenidos*. [En línea], p. 1295 [Consulta: 20 de Junio de 2015] Disponible en: <http://eprints.rclis.org/20618/1/isko13.pdf>

**Twitter.** *Twitter documentation.* [En línea], [Consulta: 05 de Abril 2015] Disponible en: <https://dev.twitter.com/overview/documentation>

**Twitter.***Twitter authentication.* [En línea], [Consulta: 06 de Abril 2015] Disponible en: <https://twitteroauth.com/>

**Twitter.***Twitter developers.* [En línea], [Consulta: 23 de Marzo 2015] Disponible en: <https://dev.twitter.com/overview/api>

**Twitter.***Twitter about.* [En línea], [Consulta: 23 de Marzo 2015] Disponible en: https://about.twitter.com/company

**Valdés Pérez D.***Los Frameworks de PHP agilizan tu trabajo.* [En línea], 31 de Julio 2007 [Consulta: 15 de Junio 2015] Disponible en: [http://www.maestrosdelWeb.com/los-frameworks-de-php-agilizan-tu-trabajo/](http://www.maestrosdelweb.com/los-frameworks-de-php-agilizan-tu-trabajo/)

**Vimeo.***Vimeo documentación.*  [En línea], 2005 [Consulta: 26 de Marzo 2015] Disponible en: <https://developer.vimeo.com/api/endpoints>

**Yorbelis Rossel L.** *Sistemas gestores de contenidos: una mirada desde las ciencias de la información.*  [En línea], Marzo 2011[Consulta: 17 de Junio 2015] Disponible en: <http://scielo.sld.cu/scielo.php?pid=S1024-94352011000100002&script=sci_arttext>

# ANEXOS

# ANEXO A. Estudio de factibilidad

* Factibilidad técnica

Hardware existente

**Tabla 1.** Hardware existente

|  |  |  |
| --- | --- | --- |
| **CANTIDAD** | **DESCRIPCION** | **ESTADO** |
| 1 | Servidor de Base de Datos | Buen estado |
| 1 | Servidor Web IIS | Buen estado |
| 1 | Laptop HP Pavilion dm4 | Buen estado |
| 1 | Laptop Dell Inspiron 14z | Buen estado |

**Realizado por:** Villa E., Rodriguez E., 2015.

**Hardware requerido**

**Tabla 2.** Hardware Requerido

|  |  |  |
| --- | --- | --- |
| **CANTIDAD** | **DESCRIPCION** | **OBSERVACIONES** |
| 1 | Impresora | Buen estado  Impresiones |
| 1 | Dispositivo USB | Buen estado  Trasladar información necesaria |

**Realizado por:** Villa E., Rodriguez E., 2015.

Software

Incluye: sistemas operativos, DBMS, programas de desarrollo, herramientas CASE.

Software existente

**Tabla 3.** Software Existente

|  |  |  |
| --- | --- | --- |
| **NOMBRE** | **DESCRIPCION** | **OBSERVACIONES** |
| Windows 7 Professional | Sistema Operativo | Vigente |
| Linux, Ubuntu Server 14.04 LTS | Sistema Operativo | Vigente |
| Internet Information Service 7 | Servidor Web | Vigente |
| Apache 2.4.10 Server PHP 5.4 | Servidor Web | Vigente, PHP >= 5.4 |
| Mysql 5.6.20 | Servidor de Base de Datos | Vigente |
| Sublime Text 3 | Editor de Texto | Versión Gratuita |
| Firefox, Chrome, Internet Explorer 9, Opera, Safari | Navegadores Web | Vigente |

**Realizado por:** Villa E., Rodriguez E., 2015.

Software requerido

**Tabla 4.** Software Requerido

|  |  |  |
| --- | --- | --- |
| **NOMBRE** | **DESCRIPCION** | **OBSERVACIONES** |
| NetBeans 7.2 | Entorno de desarrollo integrado | 1 (se pueden adquirir más licencias GPL2) |
| Gantter | Herramientas de [gestión](http://www.monografias.com/trabajos15/sistemas-control/sistemas-control.shtml) de proyectos | Programación de proyectos basado en la nube gratis |
| Draw.IO | Herramientas para construir diagramas | Herramienta en la nube gratis |
| Laravel | Framework de desarrollo | Framework de código abierto para el desarrollo de aplicaciones Web en PHP 5 |
| YouTube API | Interfaz de programación de aplicaciones de YouTube | API para ser utilizado por otro software |
| Vimeo API | Interfaz de programación de aplicaciones Vimeo | API para ser utilizado por otro software |
| Facebook API | Interfaz de programación de aplicaciones Facebook | API para ser utilizado por otro software |
| Twitter API | Interfaz de programación de aplicaciones Twitter | API para ser utilizado por otro software |

**Realizado por:** Villa E., Rodriguez E., 2015.

Personal técnico

**Tabla 5.** Personal técnico existente

|  |  |
| --- | --- |
| **NOMBRE** | **FUNCION** |
| Ing. Mario Murillo | Mantenimiento de la Operación del sistema, Administrador de Base de Datos |

**Realizado por:** Villa E., Rodriguez E., 2015.

**Tabla 6.** Personal técnico requerido

|  |  |  |
| --- | --- | --- |
| **FUNCION** | **FORMACION ACADEMICA** | **EXPERIENCIA EN:** |
| **Mantenimiento de la Operación del sistema** | Ingeniero en sistemas informáticos. |  |
| **Diseño del Sistema** | Ingeniero en sistemas informáticos. | Experiencia en diseño y  Creación sistemas informáticos |
| **Programadores** | Ingeniero en sistemas informáticos. | Creación de sistemas  Manejo de lenguaje PHP |
| **Creación de Base de Datos** | Ingeniero en sistemas informáticos. | Creación de Base de Datos  Manejo de DBMS, Mysql |

**Realizado por:** Villa E., Rodriguez E., 2015.

* Factibilidad económica

*Costos De Desarrollo*

* Consto de Personal

* + Costo de Programadores. Se tendrá 2 programadores

Costo por programador Mes 450.00 $

Costo de programadores 900.00 $

4 Meses 3600.00 $

* Costo de Jefe de Proyecto: Se tendrá 1 jefe

Costo Jefe de Proyecto: 450.00 $

* Costo de Administrador de Base de Datos: Se tendrá 1 DBA

Costo DBA: 300$

**Total Costo Personal 4350.00 $**

* Costo de Hardware y Software
* Costo estaciones de trabajo
  + Equipos existentes
* Costo software de trabajo de desarrollo
  + Licencias 0.00

**Total costo de software 0.00 $**

* Otros costos
  + Viajes 25.00 $

**Total costo de Desarrollo 1675.00 $**

*COSTOS DE INSTALAR EL SISTEMA*

* Costo de capacitación a usuarios.- existen 2 usuarios
  + Costo de capacitación por usuario 50.00 $
    - Costo Software
* Software nuevo(operación).- necesitará licencias nuevas para el hardware nuevo

Licencias 0.00 $

**Total costo de instalación: 100.00 $**

*COSTOS DE OPERACIÓN*

* Costo de mantenimiento (al año)
  + Manteamiento de Pc cliente 100.00 $

* Costo de materiales y suministros
  + Resmas de papel 2.50 $
  + Carpetas 3.00 $
  + Tinta para impresora 30.00 $
  + Suministros de oficina 20.00 $
    - Otros costos
      * Internet (hora) 0.50 $

**Total costo de operación: 156.00 $**

* Factibilidad operativa

El personal que realizará el manejo del sistema será el usuario previamente capacitado.

# ANEXO B. Estimaciones

Estimaciones

Se realizó utilizando los 15 factores de Cocomo y se ubicó el número de personas reales en las fórmulas matemáticas quedando de la siguiente manera

ConFiguración COCOMO

Modelo de desarrollo: Post Arquitectura

Lenguaje de Programación: Orientado a Objetos

Factores de Escala

**Tabla 7.** Factores de Escala

|  |  |  |
| --- | --- | --- |
| **FACTOR** | **DESCRIPCION** | **Valor** |
| *Precedentedness* | Precedencia. Experiencia en trabajo con sistemas de software relacionados | NORMAL |
| *Development flexibility* | Flexibilidad del desarrollo. Necesidad de conformidad del software con requisitos pre establecidos | NORMAL |
| *Architecture/risk resolution* | Resolución de arquitectura y riesgos. El Plan de gestión de riesgos identifica todos los ítems de riesgos críticos | NORMAL |
| *Team cohesión* | Cohesión del equipo de trabajo. Experiencia de los desarrolladores en trabajos en equipo. | BAJO |
| *Process maturity* | Madurez del proceso (cuestionario de madurez CMM). Se obtiene a través del Modelo de Madurez de Capacidad del Instituto de Ingeniería de Software CMM | NORMAL |

**Realizado por:** Villa E., Rodriguez E., 2015.

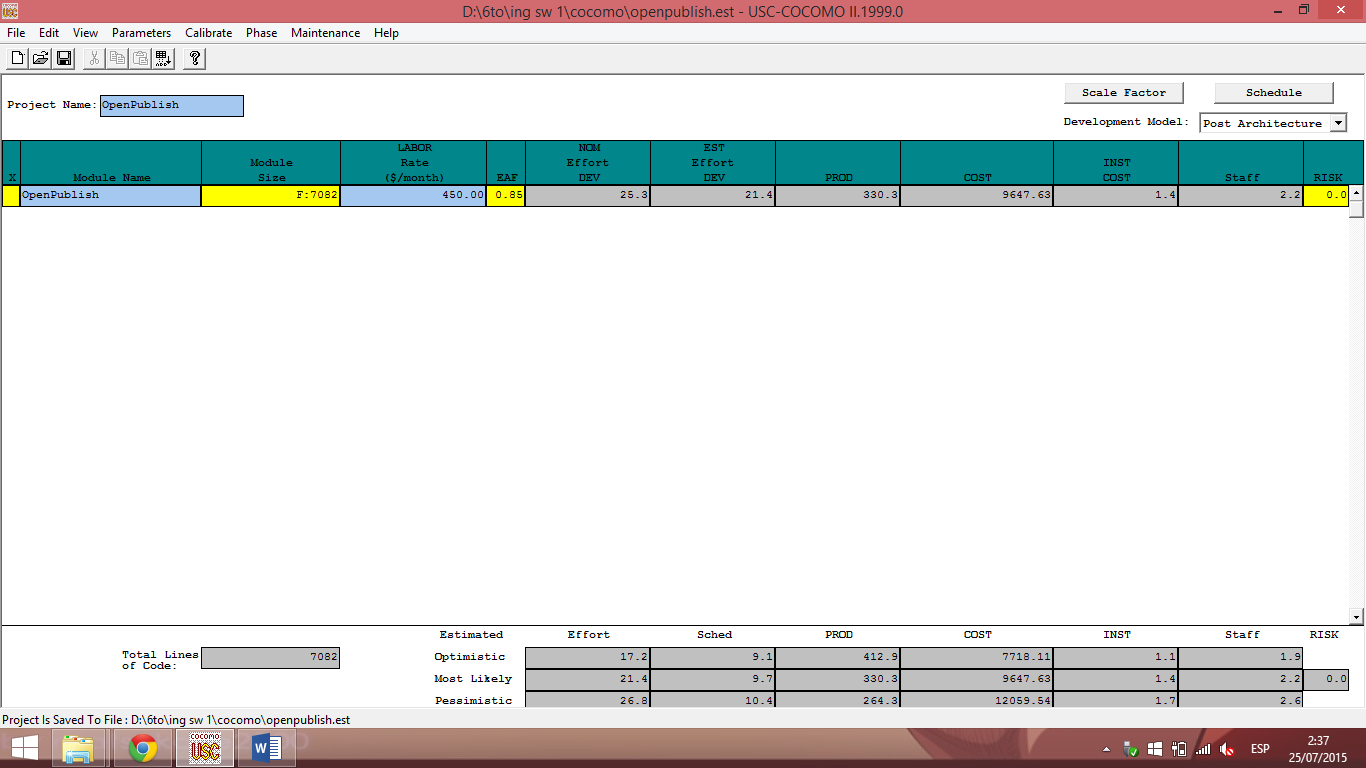
**Multiplicadores de Esfuerzo**

**Tabla 8.** Multiplicadores de Esfuerzo

| **CATEGORÍA** | **MULTIPLICADOR** | **DESCRIPCIÓN** | **VALOR** |
| --- | --- | --- | --- |
| *Product* | *RELAY* | Fiabilidad requerida del software | NORMAL |
| *DATA* | Medida de volumen de datos | NORMAL |
| *CPLX* | Complejidad del producto | ALTO |
| *DOCU* | Documentación requerida | ALTO |
| *RUSE* | Reutilización requerida | NORMAL |
| *Platform* | *TIME* | Restricción del tiempo de ejecución | NORMAL |
| *STOR* | Restricción de almacenamiento principal | NORMAL |
| *PVOL* | Volatilidad de la plataforma | NORMAL |
| *Personnel* | *ACAP* | Habilidad del analista | BAJO |
| *AEXP* | Experiencia en las aplicaciones | BAJO |
| *PCAP* | Habilidad del programador | NORMAL |
| *PEXP* | Experiencia en la plataforma | MUY BAJO |
| *LTEX* | Experiencia en la herramienta y el lenguaje | BAJO |
| *PCON* | Continuidad del personal | NORMAL |
| *Project* | *TOOL* | Uso de herramientas CASE | NORMAL |
| *SITE* | Desarrollo multilugar | NORMAL |

**Realizado por:** Villa E., Rodriguez E., 2015.

**Estimación Nominal**



**Figura 1.** Estimación nominal

**Fuente:** Villa E., Rodriguez E., 2015

Costo del proyecto: 7718.11 dólares americanos.

Número de personas: 1.9 personas.

Tiempo de desarrollo del software: 9.1 meses.

**Cálculo de Estimación Real**

Donde E es el esfuerzo, P es el número de personas y es el tiempo de desarrollo.

Extrayendo de COCOMO el indicador E (esfuerzo)=17.2 y el número de personas en el grupo de desarrollo es 2, por lo tanto remplazando los valores:

Concluyendo que el tiempo de desarrollo () es igual a 8,6 meses.

Estimación Real

Costo de proyecto: 7852.46 dólares americanos.

Número de personas: 6 personas.

Tiempo de desarrollo: 8,6 meses.

# ANEXO C. Diccionario de datos

Tabla fbaccount

**Tabla 9.** Diccionario de datos tabla fbaccount

|  |  |  |  |
| --- | --- | --- | --- |
| **Columna** | **Tipo** | **Nulo** | **Comentario** |
| Id | int(11) | No | Clave única. Identificador de la cuenta. |
| Name | varchar(50) | No | Nombre de la cuenta. |
| id\_account | varchar(50) | No | Identificador de la cuenta. |
| access\_token\_fb | varchar(250) | No | Ficha de acceso de la aplicación de Facebook |
| usuario\_id | int(11) | No | Clave foránea. Identifica usuario de OpenPublish. |
| created\_at | timestamp | No | Fecha de creación de la cuenta. |
| updated\_at | timestamp | No | Fecha de actualización de la ficha de acceso. |

**Realizado por:** Villa E., Rodriguez E., 2015.

Tabla menú

**Tabla 10.** Diccionario de datos tabla menú

|  |  |  |  |
| --- | --- | --- | --- |
| **Columna** | **Tipo** | **Nulo** | **Comentario** |
| MenuId | int(11) | No | Clave única. Identificador del menú. |
| MenuDescrip | varchar(30) | No | Descripción del menú. |
| MenuIcon | varchar(15) | Sí | Icono del menú |
| MenuRouteAlias | varchar(20) | Sí | Nombre del menú para el ruteo. |

**Realizado por:** Villa E., Rodriguez E., 2015.

Tabla Rol

**Tabla 11.** Diccionario de datos tabla rol

|  |  |  |  |
| --- | --- | --- | --- |
| **Columna** | **Tipo** | **Nulo** | **Comentario** |
| RolId | int(11) | No | Clave única. Identificador del rol. |
| RolDescrip | varchar(20) | No | Descripción del rol. |
| RolImage | varchar(20) | Sí | Imagen del rol. |

**Realizado por:** Villa E., Rodriguez E., 2015.

Tabla rolmenu

**Tabla 12.** Diccionario de datos tabla rolmenu

|  |  |  |  |
| --- | --- | --- | --- |
| **Columna** | **Tipo** | **Nulo** | **Comentario** |
| RolId | int(11) | No | Clave única. Identificador del rol. |
| RolDescrip | varchar(20) | No | Descripción del rol. |
| RolImage | varchar(20) | Sí | Imagen del rol. |

**Realizado por:** Villa E., Rodriguez E., 2015.

Tabla Status

**Tabla 13.** Diccionario de datos tabla status

|  |  |  |  |
| --- | --- | --- | --- |
| **Columna** | **Tipo** | **Nulo** | **Comentario** |
| StatusId | int(11) | No | Clave única. Identificador del estado. |
| StatusDescrip | varchar(20) | No | Descripción del estado. |
| StatusComent | varchar(100) | Sí | Comentario del estado. |

**Realizado por:** Villa E., Rodriguez E., 2015.

Tabla Submenu

**Tabla 14.** Diccionario de datos tabla submenu

|  |  |  |  |
| --- | --- | --- | --- |
| **Columna** | **Tipo** | **Nulo** | **Comentario** |
| SubmenuId | int(11) | No | Clave única. Identificador del submenú. |
| MenuId | int(11) | Sí | Clave foránea. Identificador del menú. |
| SubmenuDescrip | varchar(30) | No | Descripción del submenú. |
| SubmenuIcon | varchar(15) | Sí | Imagen del submenú |
| SubmenuRouteAlias | varchar(20) | Sí | Nombre del submenú para el ruteo. |

**Realizado por:** Villa E., Rodriguez E., 2015.

Tabla Twaccounts

**Tabla 15.** Diccionario de datos tabla twaccounts

|  |  |  |  |
| --- | --- | --- | --- |
| **Columna** | **Tipo** | **Nulo** | **Comentario** |
| Idtw | int(11) | No | Clave única. Identificador de la cuenta de twitter. |
| Name | varchar(50) | No | Nombre del perfil de la cuenta. |
| id\_account | varchar(50) | No | Identificador de la cuenta. |
| oauth\_token | varchar(250) | No | Ficha de autentificación. |
| oauth\_token\_secret | varchar(250) | No | Ficha de autenticación secreta. |
| usuario\_ide | int(11) | No | Clave Foránea. Identificador de usuario de OpenPublish. |
| created\_at | timestamp | No | Fecha de registro de la cuenta. |
| updated\_at | timestamp | No | Fecha de actualización de la cuenta. |

**Realizado por:** Villa E., Rodriguez E., 2015.

Tabla User.

**Tabla 16.** Diccionario de datos tabla user

|  |  |  |  |
| --- | --- | --- | --- |
| **Columna** | **Tipo** | **Nulo** | **Comentario** |
| Id | int(11) | No | Clave única. Edificador del usuario. |
| StatusId | int(11) | Sí | Clave Foranea. IDenficador del estado. |
| RolId | int(11) | Sí | Clave foránea. Identificador del rol |
| UserName | varchar(80) | No | Nombre del usuario. |
| email | varchar(30) | No | Correo electrónico. |
| password | varchar(100) | No | Contraseña del usuario. |
| remember\_token | varchar(100) | Sí | Ficha de acceso para recordar la sesión del usuario. |
| updated\_at | datetime | Sí | Fecha de actualización del usuario. |
| created\_at | datetime | Sí | Fecha de creación del usuario. |

**Realizado por:** Villa E., Rodriguez E., 2015.

Tabla de Usermenu

**Tabla 17.** Diccionario de datos tabla usermenu

|  |  |  |  |
| --- | --- | --- | --- |
| **Columna** | **Tipo** | **Nulo** | **Comentario** |
| UserId | int(11) | No | Clave primaria, foránea. Identificador del usuario. |
| MenuId | int(11) | No | Clave primaria, foránea. Identificador del menú. |
| ViewStatus | char(1) | No | Observación si el menú está habilitado al usuario. |

**Realizado por:** Villa E., Rodriguez E., 2015.

Tabla de Video.

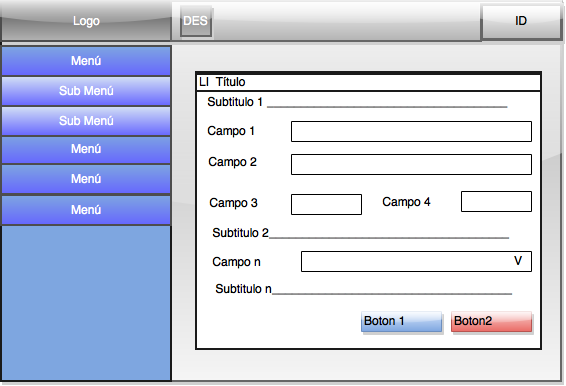
**Tabla 18.** Diccionario de datos tabla video

|  |  |  |  |
| --- | --- | --- | --- |
| **Columna** | **Tipo** | **Nulo** | **Comentario** |
| Id | varchar(15) | No | Clave única. Identificador del video. |
| UserId | int(11) | Sí | Clave foránea. Identificador del usuario. |
| VideoTitle | varchar(255) | Sí | Título del video. |
| VideoUrl | varchar(100) | Sí | Url del video. |
| VideoImage | varchar(100) | Sí | Imagen del video. |
| VideoDate | datetime | No | Fecha del registro del video. |

**Realizado por:** Villa E., Rodriguez E., 2015.

# ANEXO D. Estandar de interfaces

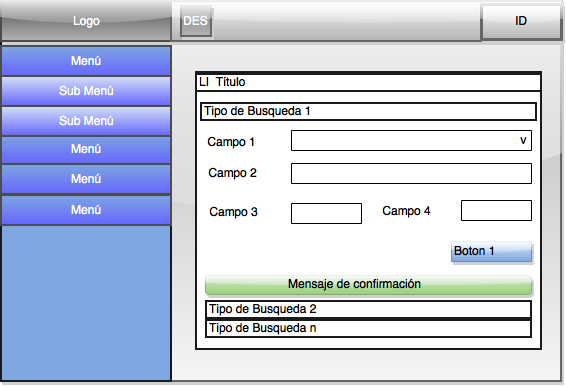
Ingreso y Modificación

****

**Figura 2.** Interfaz de ingreso y modificación

**Fuente:** Villa E., Rodriguez E., 2015

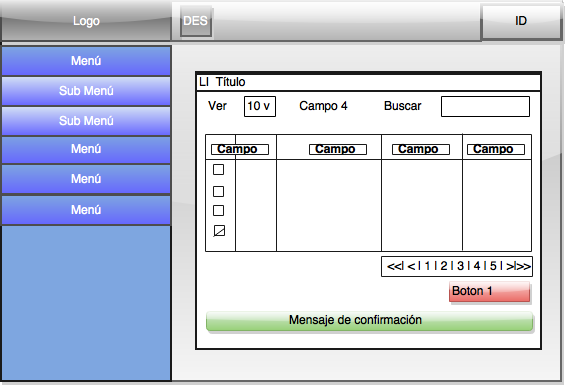
Búsqueda

****

**Figura 3.** Interfaz de búsqueda

**Fuente:** Villa E., Rodriguez E., 2015

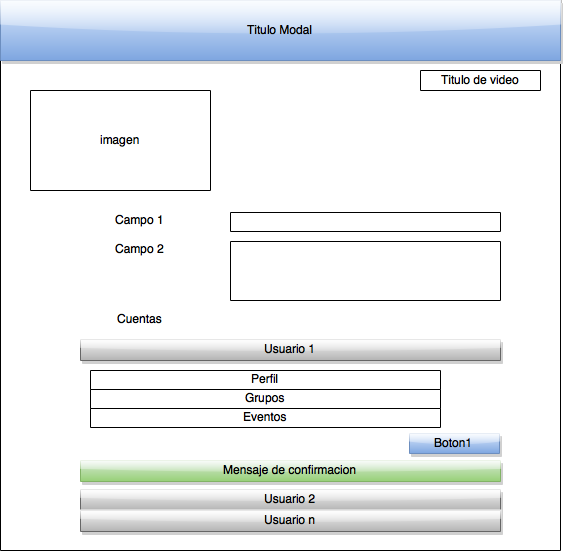
Eliminar

****

**Figura 4.** Interfaz de eliminación

**Fuente:** Villa E., Rodriguez E., 2015

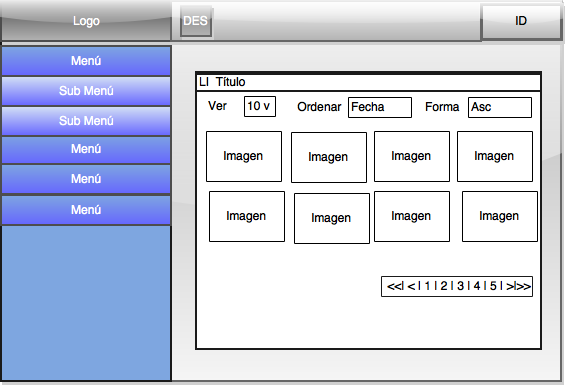
Publicación

****

**Figura 5.** Interfaz de publicación

**Fuente:** Villa E., Rodriguez E., 2015

Galería

****

**Figura 6.** Interfaz de galería

**Fuente:** Villa E., Rodriguez E., 2015

Especificación de Colores

Los colores que se utilizarán en los formularios estarán basados en el siguiente esquema:

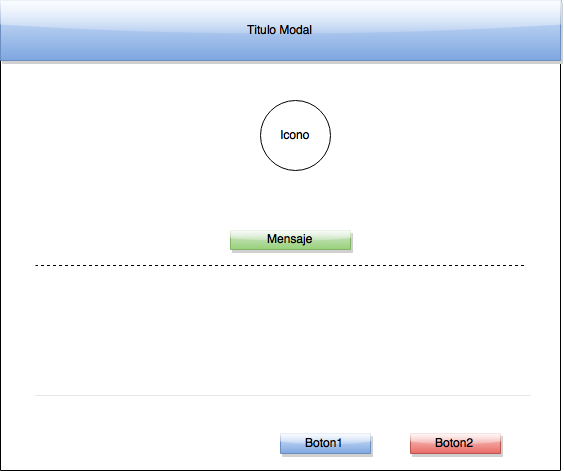
**Tabla 19.** Especificación de colores

|  |  |
| --- | --- |
| **Especificación de Colores** | |
| **Característica** | **Estándar** |
| Color de Logo | Código de color HTML 525252 |
| Color de contenedor de Menú | Código de color HTML 6AA6D6 |
| Color de Menú, Submenú | Código de color HTML 428bca |
| Color de DES, ID | Código de color HTML ebebeb |
| Color de Área de contenido | Código de color HTML ebebeb |
| Color de Mensaje de Información errónea | Código de color HTML D15E5E |
| Color de Mensaje Información | Código de color HTML 70A3D0 |
| Color de Mensaje Información de confirmación | Código de color HTML 3c763d |

**Realizado por:** Villa E., Rodriguez E., 2015.

Mensajes de Información, Error, Confirmación de acciones

Los mensajes de información, error, confirmación de acciones tendrán la siguiente estructura

****

**Figura 7.** Interfaz de mensajes de información, confirmación, errores

**Fuente:** Villa E., Rodriguez E., 2015

**Botones**

**Tabla 20.** Identificación de botones de administración

|  |  |
| --- | --- |
| Menú | |
|  | Administración |
|  | Administrar Videos |
|  | Redes Sociales |
|  | Galería |
|  | Crear Usuario |
|  | Editar Usuario |
|  | ConFiguración |
|  | Agregar Videos |
|  | Eliminar Videos |
|  | Administrar Facebook |
|  | Administrar Twitter |
|  | Facebook |
|  | Twitter |
|  | Galería Simple |
|  | YouTube |
|  | Vimeo |

**Realizado por:** Villa E., Rodriguez E., 2015.

**Tabla 21.** Identificación de botones de acción

|  |  |
| --- | --- |
| Botones de acción | |
|  | Buscar |
|  | Eliminar |
|  | Editar |
|  | Guardar |
|  | Desplegar |
|  | Esperar |
|  | Anterior |
|  | Siguiente |
|  | Cerrar |
|  | Play |
|  | Pause |

**Realizado por:** Villa E., Rodriguez E., 2015.

# ANEXO E. Detalle de historias de usuario, tareas de ingeniería, actividades y pruebas de aceptación

**Tabla 21.** HT1 Diseño de la Arquitectura del sistema

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario:**  Diseño de la Arquitectura del sistema | | |
| **Número:** HT1 | **Nombre de la historia:** Diseño de la Arquitectura del sistema | |
| **Modificación de historia de usuario:** Número y Nombre | | |
| **Usuario:** Grupo de desarrollo | | **Iteración Asignada:** 1 |
| **Prioridad en el Negocio:** Alta | | **Puntos Estimados:** 1 |
| **Riesgo en el Desarrollo:** Alto | | **Puntos Reales:** 1 |
| **Descripción:** Para realizar el desarrollo del proyecto se debe definir primeramente la arquitectura de la solución. | | |
| **Observaciones:** La arquitectura se implementara en un servidor proporcionado por la empresa donde se realiza el proyecto. | | |
| **Pruebas de Aceptación:**   * La arquitectura planteada debe ser evaluada por el cliente, conjuntamente con el equipo de desarrollo. | | |
| **Realizado por:** Villa E., Rodriguez E., 2015. | | |

**Tabla 22.** Tarea de ingeniería HT1

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario**: HT1. Arquitectura del sistema | |
| **Número tarea:** 1 | **Nombre de la tarea:** Diseño de la arquitectura del sistema. |
| **Tipo de tarea:** Desarrollo | **Puntos estimados:** 1 |
| **Fecha de inicio:** 24-03-2015 | **Fecha fin:** 24-03-2015 |
| **Programador responsable:** Edison Rodriguez | |
| **Descripción:** Diseñar la arquitectura adecuada para el sistema OpenPublish | |
| **Pruebas de aceptación:**  Que exista la debida comunicación entre las capas diseñadas. | |

**Realizado por:** Villa E., Rodriguez E., 2015.

**Tabla 23.** Actividades HT1

|  |  |  |
| --- | --- | --- |
| **FECHA** | **ACTIVIDAD** | **TIEMPO** |
| Inicio: 24-03-2015  Fin: 24-03-2015 | **Tarea: Diseñar la arquitectura del sistema**   * Consultar sobre la arquitectura MVC * Implementar la arquitectura * Documentación | 1h:00  3h:00  1h:00 |

**Realizado por:** Villa E., Rodriguez E., 2015.

**Tabla 24.** Prueba de aceptación HT1

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HT1.1 | **Nombre y número de la Historia de Usuario:** Diseño de la Arquitectura del sistema. |
| **Nombre:** La arquitectura planteada debe ser evaluada por el cliente, conjuntamente con el equipo de desarrollo. | |
| **Responsable:** Edison Rodriguez | **Fecha:**24-03-2015 |
| **Descripción:** Se detallara el diagrama de funcionamiento de la aplicación para su aprobación. | |
| **Condiciones de ejecución:**   * Que se encuentre diseñado la arquitectura del sistema para realizar el análisis. | |
| **Pasos de ejecución:**   * Revisar la arquitectura y su interacción entre cada uno de los componentes. | |
| **Resultado esperado:**   * La arquitectura debe cumplir con los requerimientos del cliente, así como su aprobación por parte del cliente. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015.

**Tabla 25.** HT2: Diseño de la base de datos

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario:** Diseño de la base de datos. | | |
| **Número:** HT2 | **Nombre de la historia:** Diseño de la base de datos | |
| **Modificación de historia de usuario:** Número y Nombre | | |
| **Usuario:** Grupo de desarrollo | | **Iteración Asignada:** 1 |
| **Prioridad en el Negocio:** Alta | | **Puntos Estimados:** 1 |
| **Riesgo en el Desarrollo:** Alto | | **Puntos Reales:** 1 |
| **Descripción:** Diseñar la base de datos. | | |
| **Observaciones:** Realizar correctamente el diseño de la base de datos, realizando una correcta normalización de las tablas. | | |
| **Pruebas de Aceptación:**   * Establecer correctamente las tablas. * Declarar correctamente las claves para la base de datos * Normalización correcta | | |
| **Realizado por:** Villa E., Rodriguez E., 2015. | | |

**Tabla 26.** Tarea de ingeniería HT2

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario:** HT2. Diseño de la base de datos | |
| **Número tarea:** 1 | **Nombre de la tarea:** Creación de la base de datos |
| **Tipo de tarea:** Desarrollo | **Puntos estimados:** 1 |
| **Fecha de inicio:** 24-03-2015 | **Fecha fin:** 24-03-2015 |
| **Programador responsable**: Edison Rodriguez | |
| **Descripción:** Crear la base de datos, desde el diseño hasta la creación de la misma, además realizar el diccionario de datos. | |
| **Pruebas de aceptación:**   * Los tipos de atributos deben ser acorde a los especificados en la base de datos. * Declarar correctamente las relaciones entre las tablas. | |

**Realizado por:** Villa E., Rodriguez E., 2015.

**Tabla 27.** Actividades HT2

|  |  |  |
| --- | --- | --- |
| **FECHA** | **ACTIVIDAD** | **TIEMPO** |
| Inicio: 24-03-2015  Fin: 24-03-2015 | **Tarea: Diseño de la base de datos**   * Diseñar el modelo lógico * Diseñar el modelo físico * Crear la base de datos en MySQL * Redactar el diccionario de datos * Documentar | 1h:00  30 min  30 min  1h:00  1h:00 |

**Realizado por:** Villa E., Rodriguez E., 2015.

**Tabla 28.** Prueba de aceptación HT2.1

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HT2.1 | **Nombre y número de la Historia de Usuario:** HT2. Diseño de la base de datos. |
| **Nombre:** Los tipos de atributos deben ser acorde a los especificados en la base de datos. | |
| **Responsable:** Edison Rodriguez | **Fecha:**24-03-2015 |
| **Descripción:** Se verificara que los atributos sean los adecuados para cada entidad. | |
| **Condiciones de ejecución:**   * Que se encuentre definido cada una de las entidades y atributos de la base de datos. | |
| **Pasos de ejecución:**   * Establecer las entidades que contendrá base de datos. * Establecer los atributos de las entidades. * Verificar que los atributos son los adecuados. | |
| **Resultado esperado:**   * Los atributos especificados son los necesarios para soportar las necesidades del cliente. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015.

**Tabla 29.** Prueba de aceptación HT2.2

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HT2.2 | **Nombre y número de la Historia de Usuario:** HT2. Diseño de la base de datos. |
| **Nombre:** Declarar correctamente las relaciones entre las tablas. | |
| **Responsable:** Edison Rodriguez | **Fecha:**24-03-2015 |
| **Descripción:** Se verificara cada una de las relaciones son correctas. | |
| **Condiciones de ejecución:**   * Que se encuentre definido cada una de las entidades y atributos de la base de datos. | |
| **Pasos de ejecución:**   * Establecer las entidades que contendrá base de datos. * Establecer los atributos de las entidades. * Estables las claves primarias y claves foráneas. | |
| **Resultado esperado:**   * La relación entre las tablas son correctas. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015.

**Tabla 30.** HT3: Definir el estándar de interfaz del sistema

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario:**  Definir el estándar de interfaz del sistema | | |
| **Número:** HT3 | **Nombre de la historia:** Definir el estándar de interfaz del sistema | |
| **Modificación de historia de usuario:** Número y Nombre | | |
| **Usuario:** Grupo de desarrollo | | **Iteración Asignada:** 1 |
| **Prioridad en el Negocio:** Alta | | **Puntos Estimados:** 4 |
| **Riesgo en el Desarrollo:** Alto | | **Puntos Reales:** 4 |
| **Descripción:** Diseñar y desarrollar la interfaz sencilla y eficiente para el usuario. | | |
| **Observaciones:** Estandarizar la interfaz del sistema. | | |
| **Pruebas de Aceptación:**   * Obtener una aprobación por parte del usuario. | | |
| **Realizado por:** Villa E., Rodriguez E., 2015. | | |

**Tabla 31.** Tarea de ingeniería HT3

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario:** HT3. Definir el estándar de interfaz del sistema | |
| **Número tarea:** 1 | Nombre de la tarea: Definir el estándar de interfaz del sistema. |
| **Tipo de tarea:** Desarrollo | Puntos estimados: 4 |
| **Fecha de inicio:** 25-03-2015 | Fecha fin: 26-03-2015 |
| **Programador responsable:** Edwin Villa | |
| Descripción: Especificar y desarrollar la interfaz de la aplicación. | |
| Pruebas de aceptación:   * Los colores de la interfaz deben ser cómodos para el usuario. * La interfaz no debe contar con elementos que el usuario no haya solicitado. | |

**Realizado por:** Villa E., Rodriguez E., 2015.

**Tabla 32.** Actividades HT3

|  |  |  |
| --- | --- | --- |
| **FECHA** | **ACTIVIDAD** | **TIEMPO** |
| Inicio: 25-03-2015  Fin: 26-03-2015 | **Tarea: Establecer el estándar de la interfaz del sistema.**   * Escoger los colores adecuados para la interfaz. * Escoger el tipo de letra y tamaño de letra. * Escoger los colores para las notificaciones que tendrá el sistema. * Desarrollar la interfaz de la aplicación. * Desarrollar el menú dinámico. * Desarrollar páginas de errores. * Diseñar la interfaz para los módulos de ingreso, modificación, búsqueda, etc. * Documentar. | 30 min  30 min  30 min  4h:00  4h:00  1h:00  4h:30  1h:00 |

**Realizado por:** Villa E., Rodriguez E., 2015.

**Tabla 33.** Prueba de aceptación HT3.1

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HT3.1 | **Nombre y número de la Historia de Usuario:** HT3. Definir el estándar de interfaz del sistema |
| **Nombre:** Obtener una aprobación por parte del usuario. | |
| **Responsable:** Edison Rodriguez | **Fecha:**26-03-2015 |
| **Descripción:** Se verificara cada una de las características que tendrá la interfaz de la aplicación. | |
| **Condiciones de ejecución:**   * Contar con las características que posee la interfaz planteada para su aprobación por parte del cliente. | |
| **Pasos de ejecución:**   * Analizar las características de la interfaz. * Analizar el formato de presentación. * Analizar los colores de la aplicación. | |
| **Resultado esperado:**   * La aprobación por parte del cliente la interfaz de la aplicación. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015.

**Tabla 34.** HU1 Permite el ingreso de datos de usuarios a OpenPublish.

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario:** Permite el ingreso de datos de usuarios a OpenPublish. | | |
| **Número:** HU1 | **Nombre de la historia:** Ingresar datos de usuario a OpenPublish. | |
| **Modificación de historia de usuario:** Número y Nombre | | |
| **Usuario:** Administrador | | **Iteración Asignada:** 1 |
| **Prioridad en el Negocio:** Alta | | **Puntos Estimados:** 10 |
| **Riesgo en el Desarrollo:** Alto | | **Puntos Reales:** 10 |
| **Descripción:** Como administrador deseo crear nuevos usuarios indicando el estado, el rol, y las funciones que deben cumplir. | | |
| **Observaciones:** Las funciones del usuario deben variar de acuerdo al rol seleccionado. | | |
| **Pruebas de Aceptación:**   * Introducir un nombre de usuario con caracteres especiales y emita el mensaje correspondiente. * Introducir un nombre de usuario con un Número de caracteres menor a 6 y mayor que 80, y emita el mensaje correspondiente * Introducir el e-mail del usuario con dirección de correo valida y un número de caracteres menor a 10 y mayor a 30 y emita el correspondiente mensaje. * Introducir la clave de usuario con un Número de caracteres menor a 6 y mayor que 30, y emita el mensaje correspondiente * Seleccionar el estado del usuario en el sistema, y emita el mensaje correspondiente * Seleccionar el rol del usuario en el sistema, y emita el mensaje correspondiente * Seleccionar las funciones del usuario en el sistema, y emita el mensaje correspondiente * Introducir todos los datos correctos sin campos vacíos y emita el mensaje correspondiente * Guardar correctamente los datos de un usuario y emitir un mensaje de confirmación | | |
| **Realizado por:** Villa E., Rodriguez E., 2015. | | |

**Tabla 35.** Tarea de ingeniería 1 HU1

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario:** HU1, Permite el ingreso de datos de usuarios a OpenPublish. | |
| **Número tarea:** **1** | **Nombre de la tarea:** Crear el modelo de Usuario |
| **Tipo de tarea:** **Desarrollo** | **Puntos estimados:** 1 |
| **Fecha de inicio:** 27-03-2015 | **Fecha fin:** 27-03-2015 |
| **Programador responsable:** Edwin Villa | |
| **Descripción**: Se crea el archivo User.php, en el cual se conFigura la conexión y tabla de la base de datos de OpenPublish | |
| **Pruebas de aceptación:**   * Verificar la conFiguración correcta del Modelo User | |

**Realizado por:** Villa E., Rodriguez E., 2015.

**Tabla 36.** Tarea de ingeniería 2 HU1

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario:** HU1, Permite el ingreso de datos de usuarios a OpenPublish. | |
| **Número tarea:** 2 | **Nombre de la tarea:** Crear el método de relación con las funciones de un usuario |
| **Tipo de tarea:** Desarrollo | **Puntos estimados:** 2 |
| **Fecha de inicio:** 27-03-2015 | **Fecha fin:** 30-03-2015 |
| **Programador responsable:** Edwin Villa | |
| Descripción: **Se crea el método que permite realizar la relación con los menús que al usuario se le permitirá usar** | |
| **Pruebas de aceptación:**   * Verificar la correcta relación entre user y menu, pasando por la tabla pivote Status | |

**Realizado por:** Villa E., Rodriguez E., 2015.

**Tabla 37.** Tarea de ingeniería 3 HU1

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario:** HU1, Permite el ingreso de datos de usuarios a OpenPublish. | |
| **Número tarea:** 3 | **Nombre de la tarea:** Crear el controlador y ruteo del proceso para ingresar un nuevo usuario. |
| **Tipo de tarea:** Desarrollo | **Puntos estimados:** 4 |
| **Fecha de inicio:** 30-03-2015 | **Fecha fin:** 31-03-2015 |
| **Programador responsable:** Edwin Villa | |
| **Descripción:** Se crea el método y su respectivo ruteo que permite ingresar un nuevo usuario | |
| **Pruebas de aceptación:**   * Cumplir con el estándar de codificación. * Control de del tipo de ruteo correcto. | |

**Realizado por:** Villa E., Rodriguez E., 2015.

**Tabla 38.** Tarea de ingeniería 4 HU1

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario:** HU1, Permite el ingreso de datos de usuarios a OpenPublish. | |
| **Número tarea:** 4 | **Nombre de la tarea:** Crear la interfaz para la vista de registro de cuentas. |
| **Tipo de tarea:** Desarrollo | **Puntos estimados:** 3 |
| **Fecha de inicio**: 01-04-2015 | **Fecha fin:** 02-04-2015 |
| **Programador responsable:** Edwin Villa | |
| **Descripción:** Se crea un formulario para poder permitir registrar nuevos usuarios | |
| **Pruebas de aceptación:**   * Cumplir con el estándar de interfaz. * Listar los estados de un usuario * Listar los roles de un usuario * Listar las funciones de un rol * Emitir un mensaje específico dependiendo el resultado de la solicitud. | |

**Realizado por:** Villa E., Rodriguez E., 2015.

**Tabla 39.** Actividades HU1

|  |  |  |
| --- | --- | --- |
| **FECHA** | **ACTIVIDAD** | **TIEMPO** |
| Inicio: 27-03-2015  Fin: 27-03-2015  Inicio: 27-03-2015  Fin: 30-03-2015  Inicio: 30-03-2015  Fin: 31-03-2015  Inicio: 01-04-2015  Fin: 02-04-2015 | **Tarea: Crear el modelo de Usuario**   * Crear y conFigurar el modelo User * Crear y conFigurar el modelo Status * Crear y conFigurar el modelo Rol * Crear y conFigurar el modelo Menu   **Tarea: Crear el método de relación con las funciones de un usuario**   * Spike Solution: Investigar como Relacionar varios modelos mediante el ORM Eloquent * Crear el Método basado en Eloquent que permita relacionar los usuarios con las funciones que podrán realizar   **Tarea: Crear el controlador y ruteo del proceso para ingresar un nuevo usuario.**   * Crear y conFigurar el Archivo UserController * Crear el Métodos para listar los estados que puede tener un nuevo usuario * Crear el Métodos para listar los roles que puede tener un nuevo usuario * Crear el Métodos para listar los menús que puede tener un rol * Crear el Métodos para ingresar un nuevo usuario * Crear los métodos de ruteo post y get que se utilizaran * Spike Solution: Investigar como autenticar usuarios mediante MultiAuth * Crear y conFigurar el Archivo AuthController * Crear el método login para que ingrese un usuario * Crear el método logout para que salga un usuario * Crear los métodos de ruteo post y get que se utilizaran   Tarea: Crear la interfaz para la vista de registro de cuentas.   * Crear el formulario para ingresar un nuevo usuario * Realizar el código Ajax para listar los estados que puede tener un usuario. * Realizar el código Ajax para listar los roles que puede tener un usuario. * Realizar el código Ajax para listar los menús que puede tener un usuario. * Realizar el código Ajax para guardar un nuevo usuario * Crear el formulario para que un usuario ingrese al sistema * Realizar el código Ajax para que ingrese un usuario del sistema * Realizar el código Ajax para que salga un usuario del sistema | 30 min  45 min  30 min  30 min  4h:00  4h:00  1h:00  1h:00  1h:00  1h:00  3h:00  30 min  2h:00  2h:00  1h:00  1h:00  30 min  1h:00  1h:00  1h:00  1h:30  1h:00  1h:00  1h:00  1h:00 |

**Realizado por:** Villa E., Rodriguez E., 2015.

**Tabla 40.** Prueba de aceptación HT1.1

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU1.1 | **Nombre y número de la Historia de Usuario:** HU1, Permite el ingreso de datos de usuarios a OpenPublish. |
| **Nombre:** Introducir un nombre de usuario con caracteres especiales y emita el mensaje correspondiente. | |
| **Responsable:** Edwin Villa. | **Fecha:**02-04-2015 |
| **Descripción:** Se verificara el ingreso de los nombres de usuario sin caracteres especiales. | |
| **Condiciones de ejecución:**   * Contar con el modelo del usuario. * Contar con los métodos, funciones para el ingreso del usuario. * Contar con el controlador y sus funciones para el ingreso del usuario. * Contar con la interfaz para ingresar datos de usuario. | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador de la aplicación. * Seleccionar la opción de agregar un nuevo usuario. * Ingresar el nombre del usuario. | |
| **Resultado esperado:**   * Emitir un mensaje de notificación en el caso de introducir caracteres especiales. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015.

**Tabla 41.** Prueba de aceptación HT1.2

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU1.2 | **Nombre y número de la Historia de Usuario:** HU1, Permite el ingreso de datos de usuarios a OpenPublish. |
| **Nombre:** Introducir un nombre de usuario con un Número de caracteres menor a 6 y mayor que 80, y emita el mensaje correspondiente | |
| **Responsable:** Edwin Villa. | **Fecha:**02-04-2015 |
| **Descripción:** Se verificara el ingreso de los nombres de usuario que se encuentren en un rango de caracteres válidos. | |
| **Condiciones de ejecución:**   * Contar con el modelo del usuario. * Contar con los métodos, funciones para el ingreso del usuario. * Contar con el controlador y sus funciones para el ingreso del usuario. * Contar con la interfaz para ingresar datos de usuario. | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador de la aplicación. * Seleccionar la opción de agregar un nuevo usuario. * Ingresar el nombre del usuario. | |
| **Resultado esperado:**   * Emitir un mensaje de notificación en el caso de introducir un nombre que no se encuentre dentro del rango esperado. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015.

**Tabla 42.** Prueba de aceptación HT1.3

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU1.3 | **Nombre y número de la Historia de Usuario:** HU1, Permite el ingreso de datos de usuarios a OpenPublish. |
| **Nombre:** Introducir el e-mail del usuario con dirección de correo valida y un número de caracteres menor a 10 y mayor a 30 y emita el correspondiente mensaje. | |
| **Responsable:** Edwin Villa. | **Fecha:**02-04-2015 |
| **Descripción:** Se verificara el ingreso del correo electrónico valido. | |
| **Condiciones de ejecución:**   * Contar con el modelo del usuario. * Contar con los métodos, funciones para el ingreso del usuario. * Contar con el controlador y sus funciones para el ingreso del usuario. * Contar con la interfaz para ingresar datos de usuario. | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador de la aplicación. * Seleccionar la opción de agregar un nuevo usuario. * Ingresar el nombre del usuario. * Ingresar el correo electrónico. | |
| **Resultado esperado:**   * Emitir un mensaje de notificación en el caso de introducir un correo no valido. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015.

**Tabla 43.** Prueba de aceptación HT1.4

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU1.4 | **Nombre y número de la Historia de Usuario:** HU1, Permite el ingreso de datos de usuarios a OpenPublish. |
| **Nombre:** Introducir la clave de usuario con un Número de caracteres menor a 6 y mayor que 30, y emita el mensaje correspondiente | |
| **Responsable:** Edwin Villa. | **Fecha:**02-04-2015 |
| **Descripción:** Se verificara el ingreso de la clave de usuario correctamente. | |
| **Condiciones de ejecución:**   * Contar con el modelo del usuario. * Contar con los métodos, funciones para el ingreso del usuario. * Contar con el controlador y sus funciones para el ingreso del usuario. * Contar con la interfaz para ingresar datos de usuario. | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador de la aplicación. * Seleccionar la opción de agregar un nuevo usuario. * Ingresar la clave del usuario. | |
| **Resultado esperado:**   * Emitir un mensaje de notificación en el caso de introducir una clave incorrecta. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015.

**Tabla 44.** Prueba de aceptación HT1.5

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU1.5 | **Nombre y número de la Historia de Usuario:** HU1, Permite el ingreso de datos de usuarios a OpenPublish. |
| **Nombre:** Seleccionar el estado del usuario en el sistema, y emita el mensaje correspondiente. | |
| **Responsable:** Edwin Villa. | **Fecha:**02-04-2015 |
| **Descripción:** Seleccionar el estado con el cual se va a crear el cliente. | |
| **Condiciones de ejecución:**   * Contar con el modelo del usuario. * Contar con los métodos, funciones para el ingreso del usuario. * Contar con el controlador y sus funciones para el ingreso del usuario. * Contar con la interfaz para ingresar datos de usuario. | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador de la aplicación. * Seleccionar la opción de agregar un nuevo usuario. * Ingresar la información requerida. * Seleccionar el estado del usuario. | |
| **Resultado esperado:**   * Emitir un mensaje de notificación en el caso de no haber seleccionado ningún estado. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015.

**Tabla 45.** Prueba de aceptación HT1.6

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU1.6 | **Nombre y número de la Historia de Usuario:** HU1, Permite el ingreso de datos de usuarios a OpenPublish. |
| **Nombre:** Seleccionar el rol del usuario en el sistema, y emita el mensaje correspondiente | |
| **Responsable:** Edwin Villa. | **Fecha:**02-04-2015 |
| **Descripción:** Se verificara la selección del rol de un usuario. | |
| **Condiciones de ejecución:**   * Contar con el modelo del usuario. * Contar con los métodos, funciones para el ingreso del usuario. * Contar con el controlador y sus funciones para el ingreso del usuario. * Contar con la interfaz para ingresar datos de usuario. | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador de la aplicación. * Seleccionar la opción de agregar un nuevo usuario. * Ingresar la información requerida. * Seleccionar el rol del usuario a cumplir. | |
| **Resultado esperado:**   * Emitir un mensaje de notificación en el caso de no haber seleccionado ningún rol. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015.

**Tabla 46.** Prueba de aceptación HT1.7

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU1.7 | **Nombre y número de la Historia de Usuario:** HU1, Permite el ingreso de datos de usuarios a OpenPublish. |
| **Nombre:** Seleccionar las funciones del usuario en el sistema, y emita el mensaje correspondiente | |
| **Responsable:** Edwin Villa. | **Fecha:**02-04-2015 |
| **Descripción:** Se verificara la selección de por lo mínimo una función. | |
| **Condiciones de ejecución:**   * Contar con el modelo del usuario. * Contar con los métodos, funciones para el ingreso del usuario. * Contar con el controlador y sus funciones para el ingreso del usuario. * Contar con la interfaz para ingresar datos de usuario. | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador de la aplicación. * Seleccionar la opción de agregar un nuevo usuario. * Ingresar la información requerida. * Seleccionar una o varias funciones del nuevo usuario. | |
| **Resultado esperado:**   * Emitir un mensaje de notificación en el caso de no haber seleccionado ninguna función. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015.

**Tabla 47.** Prueba de aceptación HT1.8

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU1.8 | **Nombre y número de la Historia de Usuario:** HU1, Permite el ingreso de datos de usuarios a OpenPublish. |
| **Nombre:** Introducir todos los datos correctos sin campos vacíos y emita el mensaje correspondiente. | |
| **Responsable:** Edwin Villa. | **Fecha:**02-04-2015 |
| **Descripción:** Se verificara que los campos cumplan con los requerimientos. | |
| **Condiciones de ejecución:**   * Contar con el modelo del usuario. * Contar con los métodos, funciones para el ingreso del usuario. * Contar con el controlador y sus funciones para el ingreso del usuario. * Contar con la interfaz para ingresar datos de usuario. | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador de la aplicación. * Seleccionar la opción de agregar un nuevo usuario. * Ingresar la información requerida. | |
| **Resultado esperado:**   * Emitir un mensaje de notificación en el caso de contar con espacios vacíos en el formulario de ingreso. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015.

**Tabla 48.** Prueba de aceptación HT1.9

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU1.9 | **Nombre y número de la Historia de Usuario:** HU1, Permite el ingreso de datos de usuarios a OpenPublish. |
| **Nombre:** Guardar correctamente los datos de un usuario y emitir un mensaje de confirmación | |
| **Responsable:** Edwin Villa. | **Fecha:**02-04-2015 |
| **Descripción:** Se verificara el registro de los nuevos usuarios correctamente en la base de datos de OpenPublish. | |
| **Condiciones de ejecución:**   * Contar con el modelo del usuario. * Contar con los métodos, funciones para el ingreso del usuario. * Contar con el controlador y sus funciones para el ingreso del usuario. * Contar con la interfaz para ingresar datos de usuario. * Ingreso de los datos correctamente. | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador de la aplicación. * Seleccionar la opción de agregar un nuevo usuario. * Ingresar la información requerida. * Proceder a guardar la información en la Base de datos. | |
| **Resultado esperado:**   * Emitir un mensaje de confirmación si la operación se realizó satisfactoriamente. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015.

**Tabla 49.** HU 2 Permite el ingreso de credenciales de las APIS utilizadas en OpenPublish

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario:** Permite el ingreso de credenciales de las APIS utilizadas en OpenPublish. | | |
| **Número:** HU2 | **Nombre de la historia:** Ingresar las credenciales de las APIS utilizadas en OpenPublish | |
| **Modificación de historia de usuario:** Número y Nombre | | |
| **Usuario:** Administrador, Editor | | **Iteración Asignada:** 2 |
| **Prioridad en el Negocio:** Alta | | **Puntos Estimados:** 3 |
| **Riesgo en el Desarrollo:** Alta | | **Puntos Reales:** 3 |
| **Descripción:** Como administrador, editor deseo ingresar las credenciales de autenticación de las APIS, para poder obtener información | | |
| **Observaciones:** Las credenciales de cada API deben ser ingresadas de manera independiente.  Además de ingresar las credenciales de las APIS, se debe seleccionar un usuario de Alavista.tv para agregar contenido a su nombre | | |
| **Pruebas de Aceptación:**   * Introducir todos los datos correctos sin campos vacíos y emita el mensaje correspondiente * Seleccionar un usuario de Alavista.tv y emita el mensaje correspondiente * Guardar correctamente los datos de conFiguración de una API y emitir un mensaje de confirmación | | |
| **Realizado por:** Villa E., Rodriguez E., 2015. | | |

**Tabla 50.** Tarea de ingeniería 1 HU2

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario:** HU2. Permite el ingreso de credenciales de las APIS utilizadas en OpenPublish. | |
| **Número tarea:** 1 | **Nombre de la tarea:** Crear el modelo de conFiguración de credenciales |
| **Tipo de tarea:** Desarrollo | **Puntos estimados:** 1 |
| **Fecha de inicio:** 03-04-2015 | **Fecha fin:** 03-04-2015 |
| **Programador responsable:** Edwin Villa | |
| **Descripción:** Se crea el archivo ConfigApp.php, en el cual se conFigura la conexión y tabla de la base de datos de OpenPublish | |
| **Pruebas de aceptación:**  Verificar la conFiguración correcta del Modelo ConfigApp | |

**Realizado por:** Villa E., Rodriguez E., 2015.

**Tabla 51.** Tarea de ingeniería 2 HU2

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario:** HU2, Permite el ingreso de credenciales de las APIS utilizadas en OpenPublish. | |
| **Número tarea:** 2 | **Nombre de la tarea:** Crear el método de relación con las credenciales de un usuario |
| **Tipo de tarea: Desarrollo** | **Puntos estimados:** 1 |
| **Fecha de inicio:** 03-04-2015 | **Fecha fin:** 03-04-2015 |
| **Programador responsable:** Edwin Villa | |
| **Descripción:** Se crea el método que permite realizar la relación con las credenciales de un usuario | |
| **Pruebas de aceptación:**   * Verificar la correcta relación entre las credenciales y el usuario | |

**Realizado por:** Villa E., Rodriguez E., 2015.

**Tabla 52.** Tarea de ingeniería 3 HU2

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario:** HU2. Permite el ingreso de credenciales de las APIS utilizadas en OpenPublish. | |
| **Número tarea:** 3 | **Nombre de la tarea:** Crear el controlador y ruteo del proceso para ingresar las credenciales de las APIS. |
| **Tipo de tarea:** Desarrollo | **Puntos estimados:** 1 |
| **Fecha de inicio:** 06-04-2015 | **Fecha fin:** 06-04-2015 |
| **Programador responsable:** Edwin Villa | |
| **Descripción:** Se crea el método y su respectivo ruteo que permite ingresar las credenciales de las APIS | |
| **Pruebas de aceptación:**   * Cumplir con el estándar de codificación. * Control de del tipo de ruteo correcto. | |

**Realizado por:** Villa E., Rodriguez E., 2015.

**Tabla 53.** Tarea de ingeniería 4 HU2

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario:** HU2. Permite el ingreso de credenciales de las APIS utilizadas en OpenPublish. | |
| **Número tarea:** 4 | **Nombre de la tarea:** Crear la interfaz para la vista de registro de credenciales. |
| **Tipo de tarea:** Desarrollo | **Puntos estimados:** 1 |
| **Fecha de inicio:** 06-04-2015 | **Fecha fin:** 06-04-2015 |
| **Programador responsable:** Edwin Villa | |
| **Descripción:** Se crea un formulario para poder permitir ingresar las credenciales de las APIS | |
| **Pruebas de aceptación:**   * Cumplir con el estándar de interfaz. * Emitir un mensaje específico dependiendo el resultado de la solicitud. | |

**Realizado por:** Villa E., Rodriguez E., 2015.

**Tabla 54.** Actividades HU2

|  |  |  |
| --- | --- | --- |
| **FECHA** | **ACTIVIDAD** | **TIEMPO** |
| Inicio: 03-04-2015  Fin: 03-04-2015  Inicio: 03-04-2015  Fin: 03-04-2015  Inicio: 06-04-2015  Fin: 06-04-2015  Inicio: 06-04-2015  Fin: 06-04-2015 | **Tarea: Crear el modelo de conFiguración de credenciales**   * Crear y conFigurar el modelo ConfigApp * Crear y conFigurar el modelo UserJoomla   **Tarea: Crear el método de relación con las credenciales de un usuario**   * Crear el Método basado en Eloquent que permita relacionar las credenciales con las funciones que podrán realizar   **Tarea: Crear el controlador y ruteo del proceso para ingresar las credenciales de las APIS.**   * Crear y conFigurar el Archivo ConfigController * Crear el Método para guardar las credenciales de YouTube * Crear el Método para guardar las credenciales de Vimeo * Crear el Método para guardar las credenciales de Facebook * Crear el Método para guardar las credenciales de Twitter * Crear el Método para mostrar la lista de usuarios de Joomla * Crear los métodos de ruteo post y get que se utilizaran   **Tarea: Crear la interfaz para la vista de registro de credenciales.**   * Crear el formulario para ingresar las credenciales de YouTube * Crear el formulario para ingresar las credenciales de Vimeo * Crear el formulario para ingresar las credenciales de Facebook * Crear el formulario para ingresar las credenciales de Twitter * Realizar el código Ajax para guardar cada una de las credenciales de las Apis * Realizar el código Ajax para mostrar los usuarios de Joomla * Realizar el código Ajax para guardar el usuario de Joomla que guardara los videos | 30 min  30 min  1h:00  30 min  30 min  30 min  30 min  30 min  30 min  30 min  1h:00  1h:00  1h:00  1h:00  1h:00  30 min  1h:00 |

**Realizado por:** Villa E., Rodriguez E., 2015.

**Tabla 55.** Prueba de aceptación HU2.1

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU2.1 | **Nombre y número de la Historia de Usuario:** HU2. Permite el ingreso de credenciales de las APIS utilizadas en OpenPublish. |
| **Nombre:** Introducir todos los datos correctos sin campos vacíos y emita el mensaje correspondiente. | |
| **Responsable:** Edwin Villa. | **Fecha:**06-04-2015 |
| **Descripción:** Se verificara el ingreso de los datos sin contener espacios vacíos. | |
| **Condiciones de ejecución:**   * Contar con el modelo de conFiguración de credenciales. * Contar con el controlador y sus funciones correspondientes para realizar el ingreso de las credenciales. * Contar con la interfaz para el ingreso de las credenciales. | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador de la aplicación. * Seleccionar la opción de conFiguración de la aplicación. * Ingresar las credenciales correspondientes a cada API. | |
| **Resultado esperado:**   * Emitir un mensaje de notificación si se dejó campos sin completar. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015.

**Tabla 56.** Prueba de aceptación HU2.2

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU2.2 | **Nombre y número de la Historia de Usuario:** HU2. Permite el ingreso de credenciales de las APIS utilizadas en OpenPublish. |
| **Nombre:** Seleccionar un usuario de Alavista.tv y emita el mensaje correspondiente | |
| **Responsable:** Edwin Villa. | **Fecha:**06-04-2015 |
| **Descripción:** Se verificara la selección de un usuario de Alavista.tv emitiendo un mensaje de notificación. | |
| **Condiciones de ejecución:**   * Contar con el modelo de conFiguración de credenciales. * Contar con el controlador y sus funciones correspondientes para realizar el ingreso de las credenciales. * Contar con la interfaz para el ingreso de las credenciales. | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador de la aplicación. * Seleccionar la opción de conFiguración de la aplicación. Seleccionar un usuario de Alavista.tv | |
| **Resultado esperado:**   * Emitir un mensaje de notificación de usuario seleccionado. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015.

**Tabla 57.** Prueba de aceptación HU2.3

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU2.3 | **Nombre y número de la Historia de Usuario:** HU2. Permite el ingreso de credenciales de las APIS utilizadas en OpenPublish. |
| **Nombre:** Guardar correctamente los datos de conFiguración de una API y usuario seleccionado emitir un mensaje de confirmación | |
| **Responsable:** Edwin Villa. | **Fecha:**06-04-2015 |
| **Descripción:** Se verificara que la información agregada se guarda correctamente en la base de datos, emitiendo un mensaje de confirmación. | |
| **Condiciones de ejecución:**   * Contar con el modelo de conFiguración de credenciales. * Contar con el controlador y sus funciones correspondientes para realizar el ingreso de las credenciales. * Contar con la interfaz para el ingreso de las credenciales. * Ingresar la información de las credenciales y selección de usuario. | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador de la aplicación. * Ingresar las credenciales de las APIs. * Seleccionar la opción de conFiguración de la aplicación. Seleccionar un usuario de Alavista.tv | |
| **Resultado esperado:**   * Emitir un mensaje de confirmación que se guardó los datos en la base. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015.

**Tabla 58.** HU3. Permite el ingreso de datos de cuentas de redes sociales en OpenPublish.

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario:** Permite el ingreso de datos de cuentas de redes sociales en OpenPublish. | | |
| **Número:** HU3 | **Nombre de la historia:** Permite el ingreso de datos de cuentas de redes sociales en OpenPublish. | |
| **Modificación de historia de usuario:** Número y Nombre | | |
| **Usuario:** Administrador, Editor | | **Iteración Asignada:** 2 |
| **Prioridad en el Negocio:** Alta | | **Puntos Estimados:** 8 |
| **Riesgo en el Desarrollo:** Media | | **Puntos Reales:** 10 |
| **Descripción:** Como administrador, editor deseo contar con la opción de guardar cuentas de usuarios de Facebook y Twitter en OpenPublish. | | |
| **Observaciones:** El sistema utilizara la API de las redes sociales para obtener datos de las cuentas. | | |
| **Pruebas de Aceptación:**   * Redireccionamiento correcto para dar permisos a la aplicación a las redes sociales. * Obtener los datos necesarios de la cuenta mediante las APIs. * Guardar los datos obtenidos en la base de datos y emitir un mensaje de confirmación de la acción realizada. | | |
| **Realizado por:** Villa E., Rodriguez E., 2015. | | |

**Tabla 59.** Tarea de ingeniería 1 HU3

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario:** HU3. Permite el ingreso de datos de cuentas de redes sociales en OpenPublish | |
| **Número tarea:** 1 | **Nombre de la tarea:** Instalar las APIs de redes sociales. |
| **Tipo de tarea:** Desarrollo | **Puntos estimados:** 1 |
| **Fecha de inicio:** 07-04-2015 | **Fecha fin:** 07-04-2015 |
| **Programador responsable:** Edison Rodriguez | |
| **Descripción:** Se realiza la instalación de las APIs de redes sociales para consumir los servicios de redes sociales. | |
| **Pruebas de aceptación:**   * Verificar la instalación correcta de las APIs de redes sociales. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 60.** Tarea de ingeniería 2 HU3

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario:** HU3. Permite el ingreso de datos de cuentas de redes sociales en OpenPublish | |
| **Número tarea:** 2 | **Nombre de la tarea:** Crear el archivo de conFiguración para las APIs de redes sociales. |
| **Tipo de tarea:** Desarrollo | **Puntos estimados:** 1 |
| **Fecha de inicio:** 07-04-2015 | **Fecha fin:** 07-04-2015 |
| **Programador responsable:** Edison Rodriguez | |
| **Descripción:** Se agregara la conFiguración de las IDs de las redes sociales Facebook y Twitter.  Se agregara los permisos que tendrá la aplicación al momento de usar las APIs. | |
| **Pruebas de aceptación:**   * Verificar que las credenciales sean correctas. * Verificar que se cuenta con los permisos necesarios para la aplicación. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 61.** Tarea de ingeniería 3 HU3

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario:** HU3. Permite el ingreso de datos de cuentas de redes sociales en OpenPublish | |
| **Número tarea:** 3 | **Nombre de la tarea:** Crear el archivo de ayuda (Helper) para cada API de redes sociales |
| **Tipo de tarea:** Desarrollo | **Puntos estimados:** 2 |
| **Fecha de inicio:** 08-04-2015 | **Fecha fin:** 08-04-2015 |
| **Programador responsable:** Edison Rodriguez | |
| **Descripción:** Se crea los archivos helper para hacer uso de las APIs de redes sociales, en las cuales se codificaran los diferentes métodos que se utilizaran en la aplicación. | |
| **Pruebas de aceptación:**   * Cumplir con el estándar de codificación. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 62.** Tarea de ingeniería 4 HU3

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario:** HU3. Permite el ingreso de datos de cuentas de redes sociales en OpenPublish | |
| **Número tarea:** 4 | **Nombre de la tarea:** Crear el modelo de la cuenta tanto para twitter como para Facebook. |
| **Tipo de tarea:** Desarrollo | **Puntos estimados:** 1 |
| **Fecha de inicio**: 09-04-2015 | **Fecha fin:** 09-04-2015 |
| **Programador responsable:** Edison Rodriguez | |
| **Descripción:** Se crea el modelo cuenta de acuerdo a cada red social, especificando cada uno de sus atributos. | |
| **Pruebas de aceptación:**   * El modelo especifique todos los atributos se hacen uso. * Especificar relaciones. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 63.** Tarea de ingeniería 5 HU3

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario:** HU3. Permite el ingreso de datos de cuentas de redes sociales en OpenPublish. | |
| **Número tarea:** 5 | **Nombre de la tarea:** Crear el controlador y ruteo del proceso a realizar. |
| **Tipo de tarea:** Desarrollo | **Puntos estimados:** 2 |
| **Fecha de inicio:** 09-04-2015 | **Fecha fin:** 10-04-2015 |
| **Programador responsable:** Edison Rodriguez | |
| **Descripción:** Agregar la dirección de la operación en el archivo de ruteo, y realizar las funciones necesarias para guardar en la base de datos las cuentas de redes sociales. | |
| **Pruebas de aceptación:**   * Cumplir con el estándar de codificación. * Control de del tipo de ruteo correcto. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 64.** Tarea de ingeniería 6 HU3

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario:** HU3. Permite el ingreso de datos de cuentas de redes sociales en OpenPublish. | |
| **Número tarea:** 6 | **Nombre de la tarea:** Crear la vista para la interfaz de registro de cuentas. |
| **Tipo de tarea:** Desarrollo | **Puntos estimados:** 1 |
| **Fecha de inicio:** 10-04-2015 | **Fecha fin:** 10-04-2015 |
| **Programador responsable:** Edison Rodriguez | |
| **Descripción:** Se creara un formulario para poder permitir al usuario registrar cuentas de redes sociales. | |
| **Pruebas de aceptación:**   * Cumplir con el estándar de interfaz. * Emitir un mensaje específico dependiendo el resultado de la solicitud. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 65.** Actividades HU3

|  |  |  |
| --- | --- | --- |
| **FECHA** | **ACTIVIDAD** | **TIEMPO** |
| Inicio: 07-04-2015  Fin: 07-04-2015  Inicio: 07-04-2015  Fin: 07-04-2015  Inicio: 08-04-2015  Fin: 08-04-2015  Inicio: 09-04-2015  Fin: 09-04-2015  Inicio: 09-04-2015  Fin: 10-04-2015  Inicio: 10-04-2015  Fin: 10-04-2015 | **Tarea: Instalar las APIs de redes sociales**   * ConFigurar el archivo composer para realizar la instalación. * Realizar la instalación mediante la actualización de composer.   **Tarea: Crear el archivo de conFiguración para las APIs de redes sociales.**   * Agregar las credenciales de las APIs de redes sociales. * Agregar los permisos requeridos por la aplicación.   **Tarea: Crear el archivo de ayuda (Helper) para cada API de redes sociales**   * Importar los archivos de cada API. * Realizar la función para generar el link para iniciar sesión en las redes sociales. * Realizar la función para generar el link de redireccionamiento. * Realizar la función para realizar el cambio de token temporal por uno de larga duración. * Realizar la función para obtener la información de la cuenta asociada.   **Tarea: Crear el modelo de la cuenta para la conexión con la base de datos.**   * Especificar la información necesaria para la conexión a la base de datos para obtener los datos.   **Tarea: Crear el controlador y ruteo del proceso a realizar.**   * Importar los paquetes del archivo helper. * Realizar el método para registrar una nueva cuenta en la base de datos. * Agregar el ruteo para el controlador en el archivo route.   **Tarea: Crear la vista para presentar la interfaz para ingresar una cuenta.**   * Realizar la interfaz para realizar el registro de una nueva cuenta. * Confirmación de ingreso de las cuentas mediante un mensaje. | 2h:00  2h:00  2h:00  2h:00  30 min  30 min  1h:30 min  3h:00  2h:30  4h:00  30 min  6h:00  1h:30 min  2h:00  2h:00 |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 66.** Prueba de aceptación HT3.1

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU3.1 | **Nombre y número de la Historia de Usuario:** HU3. Permite el ingreso de datos de cuentas de redes sociales en OpenPublish. |
| **Nombre:** Redireccionamiento correcto para dar permisos a la aplicación a las redes sociales. | |
| **Responsable:** Edison Rodriguez | **Fecha:**10-04-2015 |
| **Descripción:** Se verificara el redireccionamiento correcto a la red social Facebook o Twitter para dar los permisos necesarios a la aplicación. | |
| **Condiciones de ejecución:**   * Tener instaladas las APIs de las redes sociales * Tener creado y conFigurado el archivo de las APIs de redes sociales. * Tener creado el archivo de ayuda para cada API de las redes sociales. * Tener codificado las funciones para realizar el redireccionamiento a las redes sociales. * Tener creado el modelo de la cuenta para Twitter como para Facebook. * Tener creado el controlador y sus funciones. * Haber realizado el ruteo correspondiente para el controlador. | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador o editor. * Seleccionar la opción de conFiguración de redes sociales. * Seleccionar la opción de agregar nueva cuenta de redes sociales. | |
| **Resultado esperado:**   * Redireccionamiento a la red social seleccionada para iniciar sesión de la cuenta. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 67.** Prueba de aceptación HT3.2

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU3.2 | **Nombre y número de la Historia de Usuario:** HU3. Permite el ingreso de datos de cuentas de redes sociales en OpenPublish. |
| **Nombre:** Obtener los datos necesarios de la cuenta mediante las APIs. | |
| **Responsable:** Edison Rodriguez | **Fecha:**10-04-2015 |
| **Descripción:** Se obtiene la información de la cuenta a registrar en OpenPublish luego de haber sido aprobada por el usuario en las redes sociales. | |
| **Condiciones de ejecución:**   * Tener instaladas las APIs de las redes sociales * Tener creado y conFigurado el archivo de las APIs de redes sociales. * Tener creado el archivo de ayuda para cada API de las redes sociales. * Tener codificado las funciones para realizar el redireccionamiento a las redes sociales. * Tener creado el modelo de la cuenta para Twitter como para Facebook. * Tener creado el controlador y sus funciones. * Haber realizado el ruteo correspondiente para el controlador. | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador o editor. * Seleccionar la opción de conFiguración de redes sociales. * Seleccionar la opción de agregar nueva cuenta de redes sociales. * Iniciar sesión en la red social. * Aprobar la aplicación para que haga uso de los permisos solicitados. | |
| **Resultado esperado:**   * Obtener la información de la cuenta que haya aprobado la aplicación. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 68.** Prueba de aceptación HT3.3

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU3.3 | **Nombre y número de la Historia de Usuario:** HU3. Permite el ingreso de datos de cuentas de redes sociales en OpenPublish. |
| **Nombre:** Guardar los datos obtenidos en la base de datos y emitir un mensaje de confirmación de la acción realizada. | |
| **Responsable:** Edison Rodriguez | **Fecha:**10-04-2015 |
| **Descripción:** Verificar que los datos obtenidos se guarden correctamente en la base de datos. | |
| **Condiciones de ejecución:**   * Tener instaladas las APIs de las redes sociales * Tener creado y conFigurado el archivo de las APIs de redes sociales. * Tener creado el archivo de ayuda para cada API de las redes sociales. * Tener codificado las funciones para realizar el redireccionamiento a las redes sociales. * Tener creado el modelo de la cuenta para Twitter como para Facebook. * Tener creado el controlador y sus funciones. * Haber realizado el ruteo correspondiente para el controlador. | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador o editor. * Seleccionar la opción de conFiguración de redes sociales. * Seleccionar la opción de agregar nueva cuenta de redes sociales. * Iniciar sesión en la red social. * Aprobar la aplicación para que haga uso de los permisos solicitados. * Obtener los datos de la cuenta. * Guardar los datos obtenidos en la base de datos. | |
| **Resultado esperado:**   * Registro de los datos en la base de datos de manera correcta. * Emitir un mensaje de confirmación que los datos fueron guardados correctamente. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 69.** HU4 Permite buscar videos de acuerdo a diferentes parámetros de búsqueda en YouTube

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario:**  Permite buscar videos de acuerdo a diferentes parámetros de búsqueda en YouTube | | |
| **Número:** HU4 | **Nombre de la historia:** Buscar videos de acuerdo a diferentes parámetros de búsqueda en YouTube | |
| **Modificación de historia de usuario:** Número y Nombre | | |
| **Usuario:** Administrador, Editor | | **Iteración Asignada:** 2 |
| **Prioridad en el Negocio:** Alta | | **Puntos Estimados:** 19 |
| **Riesgo en el Desarrollo:** Media | | **Puntos Reales:** 17 |
| **Descripción:** Como administrador, editor deseo buscar videos de YouTube | | |
| **Observaciones:** El sistema utilizara la API de YouTube para permitir buscar los videos más populares en un país, los videos más vistos, y una búsqueda avanzada | | |
| **Pruebas de Aceptación:**   * Ingresar el campo Max Result que debe ser menor o igual a 50 y mayor o igual a 1. * Introducir datos sin campos vacíos y permita emitir el mensaje correspondiente * Al obtener una lista sin resultados de búsqueda se emita un mensaje * Obtener los resultados de búsqueda mediante una paginación de 10 elementos cada página | | |
| **Realizado por:** Villa E., Rodriguez E., 2015. | | |

**Tabla 70.** Tarea de ingeniería 1 HU4

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario:** HU4, Permite buscar videos de acuerdo a diferentes parámetros de búsqueda en YouTube | |
| **Número tarea:** 1 | **Nombre de la tarea:** Instalar la API de YouTube. |
| **Tipo de tarea:** Desarrollo | **Puntos estimados:** 1 |
| **Fecha de inicio:** 13-04-2015 | **Fecha fin:** 13-04-2015 |
| **Programador responsable:** Edwin Villa | |
| **Descripción:** Se realiza la instalación de las API de YouTube para consumir los servicios | |
| **Pruebas de aceptación:**   * Verificar la instalación correcta de la API | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 71.** Tarea de ingeniería 2 HU4

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario:** HU4, Permite buscar videos de acuerdo a diferentes parámetros de búsqueda en YouTube | |
| **Número tarea:** 2 | **Nombre de la tarea:** Crear el archivo de conFiguración para la API de YouTube. |
| **Tipo de tarea:** Desarrollo | **Puntos estimados:** 1 |
| **Fecha de inicio:** 13-04-2015 | **Fecha fin:** 13-04-2015 |
| **Programador responsable:** Edwin Villa | |
| **Descripción:** Se agregara la conFiguración de las ID de YouTube | |
| **Pruebas de aceptación:**   * Verificar que las credenciales sean correctas. * Verificar que se cuenta con los permisos necesarios para la aplicación. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 72.** Tarea de ingeniería 3 HU4

|  |  |
| --- | --- |
| TAREA DE INGENIERIA | |
| **Historia de Usuario:** HU4, Permite buscar videos de acuerdo a diferentes parámetros de búsqueda en YouTube | |
| **Número tarea:** 3 | **Nombre de la tarea:** Crear métodos de búsqueda en el la API de YouTube |
| **Tipo de tarea:** Desarrollo | **Puntos estimados:** 2 |
| **Fecha de inicio:** 14-04-2015 | **Fecha fin:** 14-04-2015 |
| **Programador responsable:** Edwin Villa | |
| Descripción: Se crea los métodos que permitirán buscar los Videos Populares en un país, Videos más Vistos, Búsqueda Avanzada en el archivo YouTube.php | |
| **Pruebas de aceptación:**   * Cumplir con el estándar de codificación. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 73.** Tarea de ingeniería 4 HU4

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario:** HU4, Permite buscar videos de acuerdo a diferentes parámetros de búsqueda en YouTube | |
| **Número tarea:** 4 | **Nombre de la tarea:** Crear el modelo de videos para OpenPublish |
| **Tipo de tarea:** Desarrollo | **Puntos estimados:** 1 |
| **Fecha de inicio:** 15-04-2015 | **Fecha fin:** 15-04-2015 |
| **Programador responsable:** Edwin Villa | |
| **Descripción:** Se crea el modelo para almacenar los videos en OpenPublish, especificando cada uno de sus atributos. | |
| **Pruebas de aceptación:**   * El modelo especifique que todos los atributos se hacen uso. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 74.** Tarea de ingeniería 5 HU4

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario:** HU4, Permite buscar videos de acuerdo a diferentes parámetros de búsqueda en YouTube | |
| **Número tarea:** 5 | **Nombre de la tarea:** Crear el modelo de videos para Alavista.tv |
| **Tipo de tarea:** Desarrollo | **Puntos estimados:** 1 |
| **Fecha de inicio:** 15-04-2015 | **Fecha fin:** 15-04-2015 |
| **Programador responsable:** Edwin Villa | |
| **Descripción:** Se crea el modelo para almacenar los videos en Alavista.tv, especificando cada uno de sus atributos. | |
| **Pruebas de aceptación:**  El modelo especifique que todos los atributos se hacen uso. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 75.** Tarea de ingeniería 6 HU4

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario:** HU4, Permite buscar videos de acuerdo a diferentes parámetros de búsqueda en YouTube. | |
| **Número tarea:** 6 | **Nombre de la tarea:** Crear el controlador y ruteo del proceso a realizar. |
| **Tipo de tarea:** Desarrollo | **Puntos estimados:** 6 |
| **Fecha de inicio:** 15-04-2015 | **Fecha fin:** 20-04-2015 |
| **Programador responsable:** Edwin Villa | |
| **Descripción:** Agregar la dirección de la operación en el archivo de ruteo, y realizar las funciones necesarias para guardar en la base de datos los datos de los videos | |
| **Pruebas de aceptación:**   * Cumplir con el estándar de codificación. * Control de del tipo de ruteo correcto. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 76.** Tarea de ingeniería 7 HU4

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario:** HU4, Permite buscar videos de acuerdo a diferentes parámetros de búsqueda en YouTube. | |
| **Número tarea:** 7 | **Nombre de la tarea:** Crear interfaz para la vista de búsquedas de videos |
| **Tipo de tarea:** Desarrollo | **Puntos estimados:** 5 |
| **Fecha de inicio:** 20-04-2015 | **Fecha fin:** 24-04-2015 |
| **Programador responsable:** Edwin Villa | |
| **Descripción:** Se creara un formulario para poder permitir al usuario buscar videos. | |
| **Pruebas de aceptación:**   * Cumplir con el estándar de interfaz. * Emitir un mensaje específico dependiendo el resultado de la solicitud. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 77.** Actividades HU4

|  |  |  |
| --- | --- | --- |
| **FECHA** | **ACTIVIDAD** | **TIEMPO** |
| Inicio: 13-04-2015  Fin: 13-04-2015  Inicio: 13-04-2015  Fin: 13-04-2015  Inicio: 14-04-2015  Fin: 14-04-2015  Inicio: 15-04-2015  Fin: 15-04-2015  Inicio: 15-04-2015  Fin: 15-04-2015  Inicio: 15-04-2015  Fin: 20-04-2015  Inicio: 20-04-2015  Fin: 24-04-2015 | **Tarea: Instalar la API de YouTube.**   * ConFigurar el archivo composer para realizar la instalación. * Realizar la instalación mediante la actualización de composer.   **Tarea: Crear el archivo de conFiguración para la API de YouTube.**   * Agregar la credencial de las API. * Agregar los permisos requeridos por la aplicación. * Publicar el archivo de conFiguración   **Tarea: Crear métodos de búsqueda en el la API de YouTube**   * Crear el Método para buscar los videos más vistos * Crear el Método para obtener las categorías de los videos * Crear el Método para buscar por regiones * Crear el Método para buscar los videos mediante una búsqueda avanzada   **Tarea: Crear el modelo de videos para OpenPublish**   * Crear y conFigurar el modelo VideoOpenPub   **Tarea: Crear el modelo de videos para Alavista.tv**   * Crear y conFigurar el modelo Video   **Tarea: Crear el controlador y ruteo del proceso a realizar.**   * Crear el archivo YouTubeController * Crear el Método para buscar los videos más vistos * Crear el Método para obtener las categorías de los videos * Crear el Método para buscar por regiones * Crear el Método para buscar los videos mediante una búsqueda avanzada * Agregar el ruteo para el controlador en el archivo route.   **Tarea: Crear interfaz para la vista de búsquedas de videos**   * Crear el formulario para buscar los videos más vistos * Crear el formulario para buscar por regiones * Crear el formulario para buscar los videos mediante una búsqueda avanzada * Investigar cómo crear un objeto json, un array en js * Realizar el código Ajax para realizar cada una de las búsquedas * Crear la tabla de visualización de los videos encontrados * Realizar el código Ajax para mostrar videos encontrados | 15 min  30 min  15 min  15 min  30 min  2h:00  2h:00  2h:00  2h:00  1h:00  1h:00  30 min  5h:00  5h:00  5h:00  6h:00  1h:00  2h:00  2h:00  4h:00  3:h00  3h:00  3h:00  1h:00  1h:00  4h:00  3h:00 |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 78.** Prueba de aceptación HT4.1

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU4.1 | **Nombre y número de la Historia de Usuario:** HU4, Permite buscar videos de acuerdo a diferentes parámetros de búsqueda en YouTube. |
| **Nombre:** Ingresar el campo Max Result que debe ser menor o igual a 50 y mayor o igual a 1. | |
| **Responsable:** Edwin Villa | **Fecha:**24-04-2015 |
| **Descripción:** Verificar que el número de resultados sea mayor o igual a 1 y menor o igual a 50. | |
| **Condiciones de ejecución:**   * Instalar la API de YouTube. * Contar con el archivo de conFiguración para la API de YouTube. * Contar con los métodos de búsqueda en el la API de YouTube * Contar con el modelo de videos para OpenPublish * Contar con el modelo de videos para Alavista.tv * Contar con la interfaz de búsqueda de videos. | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador o editor. * Seleccionar la opción de agregar videos. * Escoger la opción de YouTube. * Seleccionar cualquier opción de búsqueda. * Agregar la cantidad de resultados desea obtener. | |
| **Resultado esperado:**   * Que se liste una cantidad de videos entre 1-50 del resultado de la busqueda. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 79.** Prueba de aceptación HT4.2

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU4.2 | **Nombre y número de la Historia de Usuario:** HU4, Permite buscar videos de acuerdo a diferentes parámetros de búsqueda en YouTube. |
| **Nombre:** Introducir datos sin campos vacíos y permita emitir el mensaje correspondiente | |
| **Responsable:** Edwin Villa | **Fecha:**24-04-2015 |
| **Descripción:** Verificar que los criterios de búsquedas no queden en blanco. | |
| **Condiciones de ejecución:**   * Instalar la API de YouTube. * Contar con el archivo de conFiguración para la API de YouTube. * Contar con los métodos de búsqueda en el la API de YouTube * Contar con el modelo de videos para OpenPublish * Contar con el modelo de videos para Alavista.tv * Contar con la interfaz de búsqueda de videos. | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador o editor. * Seleccionar la opción de agregar videos. * Escoger la opción de YouTube. * Seleccionar cualquier opción de búsqueda. * Agregar la cantidad de resultados desea obtener. * Agregar un criterio de búsqueda. | |
| **Resultado esperado:**   * Emitir un mensaje de notificación, si ha dejado en blanco algún criterio de búsqueda. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 80.** Prueba de aceptación HT4.3

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU4.3 | **Nombre y número de la Historia de Usuario:** HU4, Permite buscar videos de acuerdo a diferentes parámetros de búsqueda en YouTube. |
| **Nombre:** Al obtener una lista sin resultados de búsqueda se emita un mensaje | |
| **Responsable:** Edwin Villa | **Fecha:**24-04-2015 |
| **Descripción:** Al obtener como resultado de la búsqueda una lista vacía se presente un mensaje de notificación. | |
| **Condiciones de ejecución:**   * Instalar la API de YouTube. * Contar con el archivo de conFiguración para la API de YouTube. * Contar con los métodos de búsqueda en el la API de YouTube * Contar con el modelo de videos para OpenPublish * Contar con el modelo de videos para Alavista.tv * Contar con la interfaz de búsqueda de videos. | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador o editor. * Seleccionar la opción de agregar videos. * Escoger la opción de YouTube. * Seleccionar cualquier opción de búsqueda. * Agregar la cantidad de resultados desea obtener. * Agregar un criterio de búsqueda. | |
| **Resultado esperado:**   * Emitir un mensaje de notificación si no se ha obtenido resultados de la búsqueda. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 81.** Prueba de aceptación HT4.4

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU4.4 | **Nombre y número de la Historia de Usuario:** HU4, Permite buscar videos de acuerdo a diferentes parámetros de búsqueda en YouTube. |
| **Nombre:** Obtener los resultados de búsqueda mediante una paginación de 10 elementos cada página | |
| **Responsable:** Edwin Villa | **Fecha:**24-04-2015 |
| **Descripción:** Obtener resultados mediante paginación de 10 elementos por pagina. | |
| **Condiciones de ejecución:**   * Instalar la API de YouTube. * Contar con el archivo de conFiguración para la API de YouTube. * Contar con los métodos de búsqueda en el la API de YouTube * Contar con el modelo de videos para OpenPublish * Contar con el modelo de videos para Alavista.tv * Contar con la interfaz de búsqueda de videos. | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador o editor. * Seleccionar la opción de agregar videos. * Escoger la opción de YouTube. * Seleccionar cualquier opción de búsqueda. * Agregar la cantidad de resultados desea obtener. * Agregar un criterio de búsqueda. | |
| **Resultado esperado:**   * Presentar los resultados obtenidos de forma paginada con 10 elementos por cada página. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 82.** HU5 Permite buscar videos de acuerdo a diferentes parámetros de búsqueda en Vimeo

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario:**  Permite buscar videos de acuerdo a diferentes parámetros de búsqueda en Vimeo | | |
| **Número:** HU5 | **Nombre de la historia:** Buscar videos de acuerdo a diferentes parámetros de búsqueda en Vimeo | |
| **Modificación de historia de usuario:** Número y Nombre | | |
| **Usuario:** Administrador, Editor | | **Iteración Asignada:** 3 |
| **Prioridad en el Negocio:** Alta | | **Puntos Estimados:** 20 |
| **Riesgo en el Desarrollo:** Media | | **Puntos Reales:** 18 |
| **Descripción:** Como administrador, editor deseo buscar videos de YouTube | | |
| **Observaciones:** El sistema utilizara la API de Vimeo para permitir buscar los videos por categorías, por etiquetas, y mediante palabras de búsqueda | | |
| **Pruebas de Aceptación:**   * Ingresar el campo Max Result que debe ser menor o igual a 50 y mayor o igual a 1. * Introducir datos sin campos vacíos y permita emitir el mensaje correspondiente * Al obtener una lista sin resultados de búsqueda se emita un mensaje * Obtener los resultados de búsqueda mediante una paginación de 10 elementos cada página | | |
| **Realizado por:** Villa E., Rodriguez E., 2015. | | |

**Tabla 83.** Tarea de ingeniería 1 HU5

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario:** HU5, Permite buscar videos de acuerdo a diferentes parámetros de búsqueda en Vimeo | |
| **Número tarea:** 1 | **Nombre de la tarea:** Instalar la API de Vimeo. |
| **Tipo de tarea:** Desarrollo | **Puntos estimados:** 1 |
| **Fecha de inicio:** 27-04-2015 | **Fecha fin:** 27-04-2015 |
| **Programador responsable:** Edwin Villa | |
| **Descripción:** Se realiza la instalación de las API de Vimeo para consumir los servicios | |
| **Pruebas de aceptación:**   * Verificar la instalación correcta de la API | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 84.** Tarea de ingeniería 2 HU5

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario:** HU5, Permite buscar videos de acuerdo a diferentes parámetros de búsqueda en Vimeo | |
| **Número tarea:** 2 | **Nombre de la tarea:** Crear el archivo de conFiguración para la API de Vimeo. |
| **Tipo de tarea:** Desarrollo | **Puntos estimados:** 1 |
| **Fecha de inicio:** 27-04-2015 | **Fecha fin:** 28-04-2015 |
| **Programador responsable:** Edwin Villa | |
| **Descripción:** Se agregara la conFiguración de las ID de Vimeo | |
| **Pruebas de aceptación:**   * Verificar que las credenciales sean correctas. * Verificar que se cuenta con los permisos necesarios para la aplicación. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 85.** Tarea de ingeniería 3 HU5

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario:** HU5, Permite buscar videos de acuerdo a diferentes parámetros de búsqueda en Vimeo | |
| **Número tarea:** 3 | **Nombre de la tarea:** Crear métodos de búsqueda en el la API de Vimeo |
| **Tipo de tarea:** Desarrollo | **Puntos estimados:** 4 |
| **Fecha de inicio:** 28-04-2015 | **Fecha fin:** 30-04-2015 |
| **Programador responsable:** Edwin Villa | |
| **Descripción:** Se crea los métodos que permitirán buscar los Videos Populares en un país, Videos más Vistos, Búsqueda Avanzada en el archivo Vimeo.php | |
| **Pruebas de aceptación:**   * Cumplir con el estándar de codificación. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 86.** Tarea de ingeniería 4 HU5

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario:** HU5, Permite buscar videos de acuerdo a diferentes parámetros de búsqueda en Vimeo. | |
| **Número tarea:** 4 | **Nombre de la tarea:** Crear el controlador y ruteo del proceso a realizar. |
| **Tipo de tarea:** Desarrollo | **Puntos estimados:** 8 |
| **Fecha de inicio:** 30-04-2015 | **Fecha fin:** 05-05-2015 |
| **Programador responsable:** Edwin Villa | |
| **Descripción:** Agregar la dirección de la operación en el archivo de ruteo, y realizar las funciones necesarias para buscar videos | |
| **Pruebas de aceptación:**   * Cumplir con el estándar de codificación. * Control de del tipo de ruteo correcto. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 87.** Tarea de ingeniería 5 HU5

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario:** HU5, Permite buscar videos de acuerdo a diferentes parámetros de búsqueda en Vimeo. | |
| **Número tarea:** 5 | **Nombre de la tarea:** Crear interfaz para la vista de búsquedas de videos |
| **Tipo de tarea:** Desarrollo | **Puntos estimados:** 6 |
| **Fecha de inicio:** 05-05-2015 | **Fecha fin:** 08-05-2015 |
| **Programador responsable:** Edwin Villa | |
| **Descripción:** Se creara un formulario para poder permitir al usuario buscar videos. | |
| **Pruebas de aceptación:**   * Cumplir con el estándar de interfaz. * Emitir un mensaje específico dependiendo el resultado de la solicitud. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 88.** Actividades HU5

|  |  |  |
| --- | --- | --- |
| **FECHA** | **ACTIVIDAD** | **TIEMPO** |
| Inicio: 27-04-2015  Fin: 27-04-2015  Inicio: 27-04-2015  Fin: 28-04-2015  Inicio: 28-04-2015  Fin: 30-04-2015  Inicio: 30-04-2015  Fin: 05-05-2015  Inicio: 05-05-2015  Fin: 08-05-2015 | **Tarea: Instalar la API de Vimeo.**   * ConFigurar el archivo composer para realizar la instalación. * Realizar la instalación mediante la actualización de composer.   **Tarea: Crear el archivo de conFiguración para la API de Vimeo.**   * Agregar la credencial de las API. * Agregar los permisos requeridos por la aplicación. * Publicar el archivo de conFiguración   **Tarea: Crear métodos de búsqueda en el la API de Vimeo**   * Crear el Método para buscar los videos por categorías * Crear el Método para obtener las categorías de los videos * Crear el Método para buscar por etiquetas * Crear el Método para buscar los videos mediante palabras   **Tarea: Crear el controlador y ruteo del proceso a realizar.**   * Crear el archivo VimeoController * Crear el Método para buscar los videos por categorías * Crear el Método para buscar por etiquetas * Crear el Método para buscar los videos mediante palabras * Crear el Método para generar los Tags de video * Agregar el ruteo para el controlador en el archivo route.   **Tarea: Crear interfaz para la vista de búsquedas de videos**   * Crear el Método para buscar los videos por categorías * Crear el Método para buscar por etiquetas * Crear el Método para buscar los videos mediante palabras * Realizar el código Ajax para realizar cada una de las búsquedas * Crear la tabla de visualización de los videos encontrados * Realizar el código Ajax para mostrar videos encontrados | 1h:00  1h:00  30 min  1h:00  1h:00  4h:00  4h:00  4h:00  4h:00  4h:00  4h:00  4h:00  4h:00  4h:00  2h:00  4h:00  4h:00  4h:00  4h:00  4h:00  4h:00 |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 89.** Prueba de aceptación HU5.1

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU5.1 | **Nombre y número de la Historia de Usuario:** HU5, Permite buscar videos de acuerdo a diferentes parámetros de búsqueda en Vimeo. |
| **Nombre:** Ingresar el campo Max Result que debe ser menor o igual a 50 y mayor o igual a 1. | |
| **Responsable:** Edwin Villa | **Fecha:**08-05-2015 |
| **Descripción:** Verificar que el número de resultados sea mayor o igual a 1 y menor o igual a 50. | |
| **Condiciones de ejecución:**   * Instalar la API de Vimeo. * Contar con el archivo de conFiguración para la API de Vimeo. * Contar con los métodos de búsqueda en el la API de Vimeo. * Contar con el modelo de videos para OpenPublish * Contar con el modelo de videos para Alavista.tv * Contar con la interfaz de búsqueda de videos. | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador o editor. * Seleccionar la opción de agregar videos. * Escoger la opción de Vimeo. * Seleccionar cualquier opción de búsqueda. * Agregar la cantidad de resultados desea obtener. | |
| **Resultado esperado:**   * Que se liste una cantidad de videos entre 1-50 del resultado de la búsqueda. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 90.** Prueba de aceptación HU5.2

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU5.2 | **Nombre y número de la Historia de Usuario:** HU5, Permite buscar videos de acuerdo a diferentes parámetros de búsqueda en Vimeo. |
| **Nombre:** Introducir datos sin campos vacíos y permita emitir el mensaje correspondiente | |
| **Responsable:** Edwin Villa | **Fecha:**08-05-2015 |
| **Descripción:** Verificar que los criterios de búsquedas no queden en blanco. | |
| **Condiciones de ejecución:**   * Instalar la API de Vimeo. * Contar con el archivo de conFiguración para la API de Vimeo. * Contar con los métodos de búsqueda en el la API de Vimeo. * Contar con el modelo de videos para OpenPublish * Contar con el modelo de videos para Alavista.tv * Contar con la interfaz de búsqueda de videos. | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador o editor. * Seleccionar la opción de agregar videos. * Escoger la opción de Vimeo. * Seleccionar cualquier opción de búsqueda. * Agregar la cantidad de resultados desea obtener. * Agregar un criterio de búsqueda. | |
| **Resultado esperado:**   * Emitir un mensaje de notificación, si ha dejado en blanco algún criterio de búsqueda. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 91.** Prueba de aceptación HU5.3

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU5.3 | **Nombre y número de la Historia de Usuario:** HU5, Permite buscar videos de acuerdo a diferentes parámetros de búsqueda en Vimeo. |
| **Nombre:** Al obtener una lista sin resultados de búsqueda se emita un mensaje | |
| **Responsable:** Edwin Villa | **Fecha:**08-05-2015 |
| **Descripción:** Al obtener como resultado de la búsqueda una lista vacía se presente un mensaje de notificación. | |
| **Condiciones de ejecución:**   * Instalar la API de Vimeo. * Contar con el archivo de conFiguración para la API de Vimeo. * Contar con los métodos de búsqueda en el la API de Vimeo. * Contar con el modelo de videos para OpenPublish * Contar con el modelo de videos para Alavista.tv * Contar con la interfaz de búsqueda de videos. | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador o editor. * Seleccionar la opción de agregar videos. * Escoger la opción de Vimeo. * Seleccionar cualquier opción de búsqueda. * Agregar la cantidad de resultados desea obtener. * Agregar un criterio de búsqueda. | |
| **Resultado esperado:**   * Emitir un mensaje de notificación si no se ha obtenido resultados de la búsqueda. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 92.** Prueba de aceptación HU5.4

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU5.4 | **Nombre y número de la Historia de Usuario:** HU5. Permite buscar videos de acuerdo a diferentes parámetros de búsqueda en Vimeo. |
| **Nombre:** Obtener los resultados de búsqueda mediante una paginación de 10 elementos cada página | |
| **Responsable:** Edwin Villa | **Fecha:**08-05-2015 |
| **Descripción:** Obtener resultados mediante paginación de 10 elementos por página. | |
| **Condiciones de ejecución:**   * Instalar la API de Vimeo. * Contar con el archivo de conFiguración para la API de Vimeo. * Contar con los métodos de búsqueda en el la API de Vimeo. * Contar con el modelo de videos para OpenPublish * Contar con el modelo de videos para Alavista.tv * Contar con la interfaz de búsqueda de videos. | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador o editor. * Seleccionar la opción de agregar videos. * Escoger la opción de Vimeo. * Seleccionar cualquier opción de búsqueda. * Agregar la cantidad de resultados desea obtener. * Agregar un criterio de búsqueda. | |
| **Resultado esperado:**   * Presentar los resultados obtenidos de forma paginada con 10 elementos por cada página. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 93.** HU6. Permite el ingreso de datos de videos a OpenPublish y Alavista.

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario:** Permite el ingreso de datos de videos a OpenPublish y Alavista. | | |
| **Número:** HU6 | **Nombre de la historia:** Guardar datos de videos en OpenPublish y Alavista. | |
| **Modificación de historia de usuario:** Número y Nombre | | |
| **Usuario:** Administrador, Editor | | **Iteración Asignada:** 3 |
| **Prioridad en el Negocio:** Alta | | **Puntos Estimados:** 20 |
| **Riesgo en el Desarrollo:** Media | | **Puntos Reales:** 19 |
| **Descripción:** Como administrador, editor deseo guardar los videos encontrados | | |
| **Observaciones:** La información de los videos se debe guardar en la base de datos de OpenPublish y Alavista.tv.  Debe permitir seleccionar todos los videos o uno por uno.  Se debe poder asignar una categoría de videos de Alavista.tv por defecto y permitir asignar una categoría a todos los videos.  Crear tags para cada video, basados en el título del mismo | | |
| **Pruebas de Aceptación:**   * Seleccionar todos los videos encontrados con un Check * Cambiar la categoría de todos los videos con un Select * Al no seleccionar ningún video se emite el mensaje correspondiente * Guardar correctamente los datos de los videos y emitir un mensaje de confirmación | | |
| **Realizado por:** Villa E., Rodriguez E., 2015. | | |

**Tabla 94.** Tarea de ingeniería 1 HU6

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario:** HU6, Permite el ingreso de datos de videos a OpenPublish y Alavista. | |
| **Número tarea:** 1 | **Nombre de la tarea:** Crear métodos en el modelo de videos para OpenPublish |
| **Tipo de tarea:** Desarrollo | **Puntos estimados:** 1 |
| **Fecha de inicio:** 11-05-2015 | **Fecha fin:** 11-05-2015 |
| **Programador responsable:** Edwin Villa | |
| **Descripción:** Se crea el modelo para almacenar los videos en OpenPublish, especificando cada uno de sus atributos. | |
| **Pruebas de aceptación:**   * El modelo especifique que todos los atributos se hacen uso. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 95.** Tarea de ingeniería 2 HU6

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario:** HU6, Permite el ingreso de datos de videos a OpenPublish y Alavista. | |
| **Número tarea**: 2 | **Nombre de la tarea:** Crear métodos en el modelo de videos para Alavista.tv |
| **Tipo de tarea:** Desarrollo | **Puntos estimados:** 1 |
| **Fecha de inicio:** 11-05-2015 | **Fecha fin:** 11-05-2015 |
| **Programador responsable:** Edwin Villa | |
| **Descripción:** Se crea el modelo para almacenar los videos en Alavista.tv, especificando cada uno de sus atributos. | |
| **Pruebas de aceptación:**   * El modelo especifique que todos los atributos se hacen uso. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 96.** Tarea de ingeniería 3 HU6

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario:** HU6, Permite el ingreso de datos de videos a OpenPublish y Alavista. | |
| **Número tarea:** 3 | **Nombre de la tarea:** Crear el controlador y ruteo del proceso a realizar. |
| **Tipo de tarea:** Desarrollo | **Puntos estimados:** 13 |
| **Fecha de inicio:** 12-05-2015 | **Fecha fin:** 20-05-2015 |
| **Programador responsable:** Edwin Villa | |
| **Descripción:** Agregar la dirección de la operación en el archivo de ruteo, y realizar las funciones necesarias para guardar los videos | |
| **Pruebas de aceptación:**   * Cumplir con el estándar de codificación. * Control de del tipo de ruteo correcto. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 97.** Tarea de ingeniería 1 HU6

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario:** HU6, Permite el ingreso de datos de videos a OpenPublish y Alavista. | |
| **Número tarea:** 4 | **Nombre de la tarea:** Crear interfaz para la vista que ingresa nuevos videos |
| **Tipo de tarea:** Desarrollo | **Puntos estimados:** 5 |
| **Fecha de inicio:** 20-05-2015 | **Fecha fin:** 22-05-2015 |
| **Programador responsable:** Edwin Villa | |
| **Descripción:** Se creara un formulario para poder permitir al usuario buscar videos. | |
| **Pruebas de aceptación:**   * Cumplir con el estándar de interfaz. * Emitir un mensaje específico dependiendo el resultado de la solicitud. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 98.** Actividades HU6

|  |  |  |
| --- | --- | --- |
| **FECHA** | **ACTIVIDAD** | **TIEMPO** |
| Inicio: 11-05-2015  Fin: 11-05-2015  Inicio: 11-05-2015  Fin: 11-05-2015  Inicio: 12-05-2015  Fin: 20-05-2015  Inicio: 20-05-2015  Fin: 22-05-2015 | **Tarea: Crear métodos en el modelo de videos para OpenPublish**   * Agregar métodos en el Modelo VideoOpenPub   **Tarea: Crear métodos en el modelo de videos para Alavista.tv**   * Agregar métodos en el Modelo Video   **Tarea: Crear el controlador y ruteo del proceso a realizar**.   * Crear el método para generar el SEO título del video * Crear el método para verificar si conjunto de videos a guardar está vacío * Spike Solution: Investigar cómo hacer transacciones mediante Laravel * Crear el método para guardar los videos en OpenPublish y Alavista.tv * Agregar el ruteo para el controlador en el archivo route.   **Tarea: Crear interfaz para la vista que ingresa nuevos videos**   * Crear el método javascript para modificar la categoría de todos los videos * Crear el método javascript para seleccionar todos los videos encontrados * Realizar el código Ajax para guardar los videos seleccionados | 1h:00  1h:00  18h:00  4h:00  4h:00  16h:00  1h:00  8h:00  8h:00  3h:00 |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 99.** Prueba de aceptación HT6.1

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU6.1 | **Nombre y número de la Historia de Usuario:** HU6, Permite el ingreso de datos de videos a OpenPublish y Alavista. |
| **Nombre:** Seleccionar todos los videos encontrados con un Check. | |
| **Responsable:** Edwin Villa | **Fecha:**22-05-2015 |
| **Descripción:** Seleccionar todos los videos mediante los check para realizar acciones como editar información, guardar datos. | |
| **Condiciones de ejecución:**   * Debe existir los métodos en el modelo de videos para OpenPublish * Debe existir los métodos en el modelo de videos para Alavista.tv * Debe existir el controlador y ruteo del proceso a realizar. * Debe existir la interfaz para la vista que ingresa nuevos videos | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador o editor. * Seleccionar la opción de agregar videos. * Seleccionar uno de los canales de video. * Seleccionar cualquier opción de búsqueda. * Listar el resultado de la búsqueda. | |
| **Resultado esperado:**   * Seleccionar todos los videos del resultado obtenido de la búsqueda. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 100.** Prueba de aceptación HT6.2

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU6.2 | **Nombre y número de la Historia de Usuario:** HU6, Permite el ingreso de datos de videos a OpenPublish y Alavista. |
| **Nombre:** Cambiar la categoría de todos los videos con un Select | |
| **Responsable:** Edwin Villa | **Fecha:**22-05-2015 |
| **Descripción:** Poder realizar el cambio de categoría mediante un select. | |
| **Condiciones de ejecución:**   * Debe existir los métodos en el modelo de videos para OpenPublish * Debe existir los métodos en el modelo de videos para Alavista.tv * Debe existir el controlador y ruteo del proceso a realizar. * Debe existir la interfaz para la vista que ingresa nuevos videos | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador o editor. * Seleccionar la opción de agregar videos. * Seleccionar uno de los canales de video. * Seleccionar cualquier opción de búsqueda. * Listar el resultado de la búsqueda. * Seleccionar por lo menos un video. * Seleccionar la categoría de video para realizar la actualización. | |
| **Resultado esperado:**   * Realizar el cambio de categoría de un video o todos los videos. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 101.** Prueba de aceptación HT6.3

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU6.3 | **Nombre y número de la Historia de Usuario:** HU6, Permite el ingreso de datos de videos a OpenPublish y Alavista. |
| **Nombre:** Al no seleccionar ningún video se emite el mensaje correspondiente. | |
| **Responsable:** Edwin Villa | **Fecha:**22-05-2015 |
| **Descripción:** Para realizar el registro de un video se debe seleccionar mínimo un video de la busqueda realizada. | |
| **Condiciones de ejecución:**   * Debe existir los métodos en el modelo de videos para OpenPublish * Debe existir los métodos en el modelo de videos para Alavista.tv * Debe existir el controlador y ruteo del proceso a realizar. * Debe existir la interfaz para la vista que ingresa nuevos videos | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador o editor. * Seleccionar la opción de agregar videos. * Seleccionar uno de los canales de video. * Seleccionar cualquier opción de búsqueda. * Listar el resultado de la búsqueda. * Seleccionar por lo menos un video. * Seleccionar la categoría de video para realizar la actualización. * Seleccionar uno o varios videos para guardar en la Base de dados. | |
| **Resultado esperado:**   * Notificar mediante un mensaje que no se ha seleccionado ningún video para que sea guardado. | |
| **Evaluación de la prueba:**  Satisfactoria. | |
| **Realizado por:** Villa E., Rodriguez E., 2015. | |

**Tabla 102.** Prueba de aceptación HT6.4

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU6.4 | **Nombre y número de la Historia de Usuario:** HU6, Permite el ingreso de datos de videos a OpenPublish y Alavista. |
| **Nombre:** Guardar correctamente los datos de los videos y emitir un mensaje de confirmación. | |
| **Responsable:** Edwin Villa | **Fecha:**22-05-2015 |
| **Descripción:** Controlar que los datos se han guardado correctamente y emitir un mensaje de confirmación. | |
| **Condiciones de ejecución:**   * Debe existir los métodos en el modelo de videos para OpenPublish * Debe existir los métodos en el modelo de videos para Alavista.tv * Debe existir el controlador y ruteo del proceso a realizar. * Debe existir la interfaz para la vista que ingresa nuevos videos | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador o editor. * Seleccionar la opción de agregar videos. * Seleccionar uno de los canales de video. * Seleccionar cualquier opción de búsqueda. * Listar el resultado de la búsqueda. * Seleccionar por lo menos un video. * Seleccionar la categoría de video para realizar la actualización. * Seleccionar uno o varios videos para guardar en la Base de dados. * Guardar los datos en la base de datos. | |
| **Resultado esperado:**   * Notificar mediante un mensaje que los datos han sido guardados correctamente. | |
| **Evaluación de la prueba:**  Satisfactoria. | |
| **Realizado por:** Villa E., Rodriguez E., 2015. | |

**Tabla 103.** HU7 Permite modificar los datos de usuarios en OpenPublish.

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario:** Permite modificar los datos de usuarios en OpenPublish. | | |
| **Número:** HU7 | **Nombre de la historia:** Modificar los datos de usuarios en OpenPublish. | |
| **Modificación de historia de usuario:** Número y Nombre | | |
| **Usuario:** Administrador | | **Iteración Asignada:** 3 |
| **Prioridad en el Negocio:** Alta | | **Puntos Estimados:** 5 |
| **Riesgo en el Desarrollo:** Media | | **Puntos Reales:** 5 |
| **Descripción:** Como administrador deseo modificar la información de los usuarios de OpenPublish | | |
| **Observaciones:** Los usuarios registrados deben ser listados en una tabla con paginación, cuando se selecciona un usuario se muestra su información en una ventana modal cargando información del usuario seleccionado para editar | | |
| **Pruebas de Aceptación:**   * Introducir un nombre de usuario con caracteres especiales y emita el mensaje correspondiente. * Introducir un nombre de usuario con un Número de caracteres menor a 6 y mayor que 80, y emita el mensaje correspondiente * Introducir el e-mail del usuario con dirección de correo valida y emita el correspondiente mensaje. * Introducir el e-mail del usuario con un Número de caracteres menor a 10 y mayor a 30, y emita el mensaje correspondiente. * Introducir la clave de usuario con un Número de caracteres menor a 6 y mayor que 30, y emita el mensaje correspondiente * Seleccionar el estado del usuario en el sistema, y emita el mensaje correspondiente * Seleccionar el rol del usuario en el sistema, y emita el mensaje correspondiente * Seleccionar las funciones del usuario en el sistema, y emita el mensaje correspondiente * Introducir todos los datos correctos sin campos vacíos y emita el mensaje correspondiente * Guardar correctamente los datos de un usuario y emitir un mensaje de confirmación | | |
| **Realizado por:** Villa E., Rodriguez E., 2015. | | |

**Tabla 104.** Tarea de ingeniería 1 HU7

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario:** HU7, Permite modificar los datos de usuarios en OpenPublish | |
| **Número tarea:** 1 | **Nombre de la tarea:** Crear métodos en el controlador y ruteo del proceso a realizar. |
| **Tipo de tarea:** Desarrollo | **Puntos estimados:** 3 |
| **Fecha de inicio:** 25-05-2015 | **Fecha fin:** 26-05-2015 |
| **Programador responsable:** Edwin Villa | |
| **Descripción:** Agregar la dirección de la operación en el archivo de ruteo, y realizar las funciones necesarias para guardar los videos | |
| **Pruebas de aceptación:**   * Cumplir con el estándar de codificación. * Control de del tipo de ruteo correcto. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 105.** Tarea de ingeniería 2 HU7

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario:** HU7, Permite modificar los datos de usuarios en OpenPublish | |
| **Número tarea:** 2 | **Nombre de la tarea:** Crear interfaz para la vista que modifica usuarios |
| **Tipo de tarea:** Desarrollo | **Puntos estimados:** 2 |
| **Fecha de inicio:** 26-05-2015 | **Fecha fin:** 27-05-2015 |
| **Programador responsable:** Edwin Villa | |
| **Descripción:** Se creara un formulario para poder permitir modificar datos de los usuarios | |
| **Pruebas de aceptación:**   * Cumplir con el estándar de interfaz. * Emitir un mensaje específico dependiendo el resultado de la solicitud. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 106.** Actividades HU7

|  |  |  |
| --- | --- | --- |
| **FECHA** | **ACTIVIDAD** | **TIEMPO** |
| Inicio: 25-05-2015  Fin: 26-05-2015  Inicio: 26-05-2015  Fin: 27-05-2015 | **Tarea: Crear el controlador y ruteo del proceso para ingresar un nuevo usuario.**   * Crear el Método para listar los usuarios de OpenPublish a editar * Crear el Método para modificar un usuario * Crear los métodos de ruteo post y get que se utilizaran   **Tarea: Crear interfaz para la vista que modifica usuarios.**   * Listar los usuarios de OpenPublish * Crear el formulario para editar a los usuarios de OpenPublish * Realizar el código Ajax para listar los estados que puede tener un usuario. * Realizar el código Ajax para listar los roles que puede tener un usuario. * Realizar el código Ajax para listar los menús que puede tener un usuario. * Realizar el código Ajax para guardar un nuevo usuario * Crear el formulario para que un usuario modifique su clave * Realizar el código Ajax para que un usuario edite su clave | 4h:00  5h:00  1h:00  1h:00  1h:00  1h:00  1h:00  1h:00  1h:00  1h:00  1h:00 |

**Tabla 107.** Prueba de aceptación HU7.1

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU7.1 | **Nombre y número de la Historia de Usuario:** HU7, Permite modificar los datos de usuarios en OpenPublish |
| **Nombre:** Introducir un nombre de usuario con caracteres especiales y emita el mensaje correspondiente. | |
| **Responsable:** Edwin Villa. | **Fecha:**27-05-2015 |
| **Descripción:** Se verificara el ingreso de los nombres de usuario sin caracteres especiales. | |
| **Condiciones de ejecución:**   * Contar con el modelo del usuario. * Contar con los métodos, funciones para la modificación del usuario. * Contar con el controlador y sus funciones para la modificación del usuario. * Contar con la interfaz para modificar datos de usuario. | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador de la aplicación. * Seleccionar la opción de editar usuario. * Seleccionar el cliente a editar. * Ingresar nuevos datos. | |
| **Resultado esperado:**   * Emitir un mensaje de notificación en el caso de introducir caracteres especiales. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 108.** Prueba de aceptación HU7.2

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU7.2 | **Nombre y número de la Historia de Usuario:** HU7, Permite modificar los datos de usuarios en OpenPublish |
| **Nombre:** Introducir un nombre de usuario con un Número de caracteres menor a 6 y mayor que 80, y emita el mensaje correspondiente | |
| **Responsable:** Edwin Villa. | **Fecha:**27-05-2015 |
| **Descripción:** Se verificara el ingreso de los nombres de usuario que se encuentren en un rango de caracteres válidos. | |
| **Condiciones de ejecución:**   * Contar con el modelo del usuario. * Contar con los métodos, funciones para la modificación del usuario. * Contar con el controlador y sus funciones para la modificación del usuario. * Contar con la interfaz para modificar datos de usuario. | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador de la aplicación. * Seleccionar la opción de editar usuario. * Seleccionar el cliente a editar. * Ingresar nuevos datos. | |
| **Resultado esperado:**   * Emitir un mensaje de notificación en el caso de introducir un nombre que no se encuentre dentro del rango esperado. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 109.** Prueba de aceptación HU7.3

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU7.3 | **Nombre y número de la Historia de Usuario:** HU7, Permite modificar los datos de usuarios en OpenPublish |
| **Nombre:** Introducir el e-mail del usuario con dirección de correo valida y un número de caracteres menor a 10 y mayor a 30 y emita el correspondiente mensaje. | |
| **Responsable:** Edwin Villa. | **Fecha:**27-05-2015 |
| **Descripción:** Se verificara el ingreso del correo electrónico valido. | |
| **Condiciones de ejecución:**   * Contar con el modelo del usuario. * Contar con los métodos, funciones para la modificación del usuario. * Contar con el controlador y sus funciones para la modificación del usuario. * Contar con la interfaz para modificar datos de usuario. | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador de la aplicación. * Seleccionar la opción de editar usuario. * Seleccionar el cliente a editar. * Ingresar nuevos datos. | |
| **Resultado esperado:**   * Emitir un mensaje de notificación en el caso de introducir un correo no valido. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 110.** Prueba de aceptación HU7.4

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU7.4 | **Nombre y número de la Historia de Usuario:** HU7, Permite modificar los datos de usuarios en OpenPublish |
| **Nombre:** Introducir la clave de usuario con un Número de caracteres menor a 6 y mayor que 30, y emita el mensaje correspondiente | |
| **Responsable:** Edwin Villa. | **Fecha:**27-05-2015 |
| **Descripción:** Se verificara el ingreso de la clave de usuario correctamente. | |
| **Condiciones de ejecución:**   * Contar con el modelo del usuario. * Contar con los métodos, funciones para la modificación del usuario. * Contar con el controlador y sus funciones para la modificación del usuario. * Contar con la interfaz para modificar datos de usuario. | |
| **Pasos de ejecución:**   * Iniciar sesión como usuario de la aplicación. * Seleccionar la opción de editar usuario. * Seleccionar el cliente a editar. * Ingresar la nueva clave de usuario. | |
| **Resultado esperado:**   * Emitir un mensaje de notificación en el caso de introducir una clave incorrecta. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 111.** Prueba de aceptación HU7.5

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU7.5 | **Nombre y número de la Historia de Usuario:** HU7, Permite modificar los datos de usuarios en OpenPublish |
| **Nombre:** Seleccionar el estado del usuario en el sistema, y emita el mensaje correspondiente. | |
| **Responsable:** Edwin Villa. | **Fecha:**27-05-2015 |
| **Descripción:** Seleccionar el estado con el cual se va a crear el cliente. | |
| **Condiciones de ejecución:**   * Contar con el modelo del usuario. * Contar con los métodos, funciones para la modificación del usuario. * Contar con el controlador y sus funciones para la modificación del usuario. * Contar con la interfaz para modificar datos de usuario | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador de la aplicación. * Seleccionar la opción de editar usuario. * Seleccionar el cliente a editar. * Seleccionar el estado del usuario. | |
| **Resultado esperado:**   * Emitir un mensaje de notificación en el caso de no haber seleccionado ningún estado. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 112.** Prueba de aceptación HU7.6

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU7.6 | **Nombre y número de la Historia de Usuario:** HU7, Permite modificar los datos de usuarios en OpenPublish |
| **Nombre:** Seleccionar el rol del usuario en el sistema, y emita el mensaje correspondiente | |
| **Responsable:** Edwin Villa. | **Fecha:**27-05-2015 |
| **Descripción:** Se verificara la selección del rol de un usuario. | |
| **Condiciones de ejecución:**   * Contar con el modelo del usuario. * Contar con los métodos, funciones para modificar el usuario. * Contar con el controlador y sus funciones para modificar el usuario. * Contar con la interfaz para ingresar datos de usuario. | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador de la aplicación. * Seleccionar la opción de editar usuarios. * Ingresar los datos a actualizados. * Seleccionar el rol del usuario a cumplir. | |
| **Resultado esperado:**   * Emitir un mensaje de notificación en el caso de no haber seleccionado ningún rol. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 113.** Prueba de aceptación HU7.7

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU7.7 | **Nombre y número de la Historia de Usuario:** HU7, Permite modificar los datos de usuarios en OpenPublish |
| **Nombre:** Seleccionar las funciones del usuario en el sistema, y emita el mensaje correspondiente | |
| **Responsable:** Edwin Villa. | **Fecha:**27-05-201 |
| **Descripción:** Se verificara la selección de por lo mínimo una función. | |
| **Condiciones de ejecución:**   * Contar con el modelo del usuario. * Contar con los métodos, funciones para modificar el usuario. * Contar con el controlador y sus funciones para modificar el usuario. * Contar con la interfaz para modificar los datos de usuario. | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador de la aplicación. * Seleccionar la opción para modificar usuarios. * Ingresar la información requerida. * Seleccionar una o varias funciones a actualizar. | |
| **Resultado esperado:**   * Emitir un mensaje de notificación en el caso de no haber seleccionado ninguna función. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 114.** Prueba de aceptación HU7.8

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU7.8 | **Nombre y número de la Historia de Usuario:** HU7, Permite modificar los datos de usuarios en OpenPublish |
| **Nombre:** Introducir todos los datos correctos sin campos vacíos y emita el mensaje correspondiente. | |
| **Responsable:** Edwin Villa. | **Fecha:**27-05-2015 |
| **Descripción:** Se verificara que los campos cumplan con los requerimientos. | |
| **Condiciones de ejecución:**   * Contar con el modelo del usuario. * Contar con los métodos, funciones para modificar el usuario. * Contar con el controlador y sus funciones para modificar el usuario. * Contar con la interfaz para modificar los datos de usuario. | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador de la aplicación. * Seleccionar la opción de editar usuario. * Ingresar la información a actualizar. | |
| **Resultado esperado:**   * Emitir un mensaje de notificación en el caso de contar con espacios vacíos en el formulario de modificación. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 115.** Prueba de aceptación HU7.9

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU7.9 | **Nombre y número de la Historia de Usuario:** HU7, Permite modificar los datos de usuarios en OpenPublish |
| **Nombre:** Guardar correctamente los datos de un usuario y emitir un mensaje de confirmación | |
| **Responsable:** Edwin Villa. | **Fecha:**27-05-2015 |
| **Descripción:** Se verificara el registro de los nuevos usuarios correctamente en la base de datos de OpenPublish. | |
| **Condiciones de ejecución:**   * Contar con el modelo del usuario. * Contar con los métodos, funciones para modificar el usuario. * Contar con el controlador y sus funciones para modificar el usuario. * Contar con la interfaz para modificar los datos de usuario. * Ingreso de los datos correctamente. | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador de la aplicación. * Seleccionar la opción de agregar un nuevo usuario. * Ingresar la información requerida. * Proceder a guardar la información en la Base de datos. | |
| **Resultado esperado:**   * Emitir un mensaje de confirmación si la operación se realizó satisfactoriamente. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 116.** HU8: Permite modificar los datos de las credenciales de las APIs utilizadas en OpenPublish.

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario:** Permite modificar los datos de las credenciales de las APIs utilizadas en OpenPublish. | | |
| **Número:** HU8 | **Nombre de la historia:** Permite modificar los datos de las credenciales de las APIs utilizadas en OpenPublish. | |
| **Modificación de historia de usuario:** Número y Nombre | | |
| **Usuario:** Administrador, Editor | | **Iteración Asignada:** 3 |
| **Prioridad en el Negocio:** Alta | | **Puntos Estimados:** 4 |
| **Riesgo en el Desarrollo:** Media | | **Puntos Reales:** 4 |
| **Descripción:** Como administrador, deseo contar con la opción de poder modificar las credenciales de las APIs utilizadas para la integración de la aplicación. | | |
| **Observaciones:** La credencial defectuosa será remplazada por una nueva ingresada por el administrador. | | |
| **Pruebas de Aceptación:**   * No permitir el ingreso de campos vacíos. * Guardar las credenciales de las APIs de manera correcta en la base de datos, y emitir un mensaje de confirmación de la acción efectuada. | | |

**Tabla 117.** Tarea de ingeniería 1 HU8

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario:** HU8. Permite modificar los datos de las credenciales de las APIs utilizadas en OpenPublish. | |
| **Número tarea:** 1 | **Nombre de la tarea:** Crear métodos en el controlador y ruteo del proceso a realizar. |
| **Tipo de tarea:** Desarrollo | **Puntos estimados:** 2 |
| **Fecha de inicio:** 27-05-2015 | **Fecha fin:** 28-05-2015 |
| **Programador responsable:** Edison Rodriguez | |
| **Descripción:** Agregar la dirección de la operación en el archivo de ruteo, y realizar las funciones necesarias para modificar las credenciales de las APIS | |
| **Pruebas de aceptación:**   * Cumplir con el estándar de codificación. * Control de del tipo de ruteo correcto**.** | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 118.** Tarea de ingeniería 2 HU8

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario:** HU8. Permite modificar los datos de las credenciales de las APIs utilizadas en OpenPublish. | |
| **Número tarea:** 2 | **Nombre de la tarea:** Crear interfaz para la vista que modifica credenciales de las APIs |
| **Tipo de tarea:** Desarrollo | **Puntos estimados:** 2 |
| **Fecha de inicio:** 28-05-2015 | **Fecha fin:** 29-05-2015 |
| **Programador responsable:** Edison Rodriguez | |
| **Descripción:** Se creara un formulario para poder permitir modificar datos de las credenciales | |
| **Pruebas de aceptación:**   * Cumplir con el estándar de interfaz. * Emitir un mensaje específico dependiendo el resultado de la solicitud. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 119.** Actividades HU8

|  |  |  |
| --- | --- | --- |
| **FECHA** | **ACTIVIDAD** | **TIEMPO** |
| Inicio:27-05-2015  Fin: 28-05-2015  Inicio:28-05-2015  Fin: 29-05-2015 | **Tarea: Crear métodos en el controlador y ruteo del proceso a realizar.**   * Crear el Método para listar los usuarios de Alavista.tv a seleccionar * Crear el Método para modificar cada una de las credenciales de cada API * Crear los métodos de ruteo post y get que se utilizaran   **Tarea: Crear interfaz para la vista que modifica credenciales de las APIs**   * Crear el código para Listar los usuarios de Alavista.tv * Crear el formulario para editar las credenciales de las Apis * Realizar el código Ajax para listar las credenciales anteriores * Realizar el código Ajax para listar los usuarios de Alavista.tv * Realizar el código Ajax para guardar las nuevas credenciales | 2h:00  5h:00  1h:00  1h:00  1h:00  2h:00  2h:00  2h:00 |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 120.** Prueba de aceptación HU8.1

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU8.1 | **Nombre y número de la Historia de Usuario** HU8. Permite modificar los datos de las credenciales de las APIs utilizadas en OpenPublish. |
| **Nombre:** No permitir el ingreso de campos vacíos. | |
| **Responsable:** Edison Rodriguez | **Fecha:**29-05-2015 |
| **Descripción:** Se verificara que no se permita campos vacíos. | |
| **Condiciones de ejecución:**   * Contar con los métodos en el controlador y ruteo del proceso a realizar. * Crear interfaz para la vista que modifica credenciales de las APIs. | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador de la aplicación. * Seleccionar la opción de administrador, y conFiguración. * Ingresar las credenciales de las APIs. | |
| **Resultado esperado:**   * Emitir un mensaje de notificación informando que no ha ingresado un campo requerido. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 121.** Prueba de aceptación HU8.2

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU8.2 | **Nombre y número de la Historia de Usuario** HU8. Permite modificar los datos de las credenciales de las APIs utilizadas en OpenPublish. |
| **Nombre:** Guardar las credenciales de las APIs de manera correcta en la base de datos, y emitir un mensaje de confirmación de la acción efectuada. | |
| **Responsable:** Edison Rodriguez | **Fecha:**29-05-2015 |
| **Descripción:** Confirmar el registro correcto de la actualización de las credenciales de la APIs en la base de datos de OpenPublish. | |
| **Condiciones de ejecución:**   * Contar con los métodos en el controlador y ruteo del proceso a realizar. * Crear interfaz para la vista que modifica credenciales de las APIs. | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador de la aplicación. * Seleccionar la opción de administrador, y conFiguración. * Ingresar las credenciales de las APIs. * Guardar los nuevos datos de las credenciales. | |
| **Resultado esperado:**   * Emitir un mensaje de confirmación de la actualización realizada. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 122.** HU9: Permite modificar los datos de cuentas de la red social Facebook en OpenPublish.

HU9: Permite modificar los datos de cuentas de la red social Facebook en OpenPublish.

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario:** Permite modificar los datos de cuentas de la red social Facebook en OpenPublish. | | |
| **Número:** HU9 | **Nombre de la historia:** Permite modificar los datos de cuentas de la red social Facebook en OpenPublish. | |
| **Modificación de historia de usuario:** Número y Nombre | | |
| **Usuario:** Administrador, Editor | | **Iteración Asignada:** 4 |
| **Prioridad en el Negocio:** Alta | | **Puntos Estimados:** 8 |
| **Riesgo en el Desarrollo:** Alta | | **Puntos Reales:** 8 |
| **Descripción:** Como administrador deseo contar con la opción de poder modificar los token de autenticación de todas las cuentas de redes sociales Facebook y Twitter.  El editor contara con la misma opción, con la diferencia que solo podrá actualizar sus cuentas registradas. | | |
| **Observaciones:** Realizar un cambio de token de corta duración por uno de larga duración haciendo uso de la API de las redes sociales. | | |
| **Pruebas de Aceptación:**   * Obtener los datos correctos de la API de Facebook. * Guardar la información correctamente en la base de datos y emitir un mensaje de confirmación de la acción efectuada. | | |
| **Realizado por:** Villa E., Rodriguez E., 2015. | | |

**Tabla 123.** Tarea de ingeniería 1 HU9

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario:** HU9. Permite modificar los datos de cuentas de la red social Facebook en OpenPublish. | |
| **Número tarea:** 1 | **Nombre de la tarea:** Crear el método para actualizar token en el archivo FacebookHelper. |
| **Tipo de tarea:** Desarrollo | **Puntos estimados:** 3 |
| **Fecha de inicio:** 01-06-2015 | **Fecha fin:** 02-06-2015 |
| **Programador responsable:** Edison Rodriguez | |
| **Descripción:** Se creara la función en el archivo helper para realizar el actualización de un token de larga duración, haciendo uso de la API de Facebook | |
| **Pruebas de aceptación:**   * Cumplir con el estándar de codificación. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 124.** Tarea de ingeniería 2 HU9

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario:** HU9. Permite modificar los datos de cuentas de la red social Facebook en OpenPublish. | |
| **Número tarea:** 2 | **Nombre de la tarea:** Crear las funciones en el controlador, y el ruteo del mismo. |
| **Tipo de tarea:** Desarrollo | **Puntos estimados:** 3 |
| **Fecha de inicio:** 02-06-2015 | **Fecha fin:** 03-06-2015 |
| **Programador responsable:** Edison Rodriguez | |
| **Descripción:** Se creara la función en el controlador para hacer uso de la función en el archivo helper para la actualización del token. | |
| **Pruebas de aceptación:**   * Cumplir con el estándar de codificación. * Verificar la actualización del token de larga duración. * Guardar el token en la base de datos y emitir un mensaje de confirmación. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 125.** Tarea de ingeniería 3 HU9

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario:** HU9. Permite modificar los datos de cuentas de la red social Facebook en OpenPublish. | |
| **Número tarea:** 3 | **Nombre de la tarea:** Crear la interfaz para la actualización del token. |
| **Tipo de tarea:** Desarrollo | **Puntos estimados:** 2 |
| **Fecha de inicio:** 04-06-2015 | **Fecha fin:** 04-06-2015 |
| Programador responsable: Edison Rodriguez | |
| **Descripción:** Se creara la interfaz para el proceso de actualización del token de la cuenta de Facebook. | |
| **Pruebas de aceptación:**   * Cumplir con el estándar de la interfaz. * Emitir un mensaje específico dependiendo el resultado de la solicitud. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 126.** Actividades HU9

|  |  |  |
| --- | --- | --- |
| **FECHA** | **ACTIVIDAD** | **TIEMPO** |
| Inicio:01-06-2015  Fin: 02-06-2015  Inicio:02-06-2015  Fin: 03-06-2015  Inicio:04-06-2015  Fin: 04-06-2015 | **Tarea: Crear el método para actualizar token en el archivo helper.**   * Spike solution: Realizar las pruebas para el cambio de token en un archivo independiente para luego aplicarlo al proyecto. * Crear el método para iniciar sesión a partir del token almacenado en la Base de datos. * Crear el método para realizar el cambio de token temporal por el de larga duración.   **Tarea: Crear las funciones en el controlador, y el ruteo del mismo.**   * Crear la función para listar las cuentas asociadas. * Crear la función para realizar la actualización de token.   **Tarea: Crear la interfaz para la actualización del token.**   * Creación y codificación del botón para realizar la actualización en la interfaz. * Realizar el código Ajax para listar las cuentas asociadas. * Realizar el código Ajax para realizar la petición de actualización. | 3h:00  3h:00  6h:00  8h:00  4h:00  2h:00  4h:00  2h:00 |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 127.** Prueba de aceptación HU9.1

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU9.1 | **Nombre y número de la Historia de Usuario** HU9. Permite modificar los datos de cuentas de la red social Facebook en OpenPublish. |
| **Nombre:** Obtener los datos correctos de la API de Facebook. | |
| **Responsable:** Edison Rodriguez | **Fecha:**04-06-2015 |
| **Descripción:** Verificar que los datos solicitados por la aplicación, mediante el uso de la API. | |
| **Condiciones de ejecución:**   * Contar con el método para la actualización del token en el archivo FacebookHelper. * Contar las funciones en el controlador, y el ruteo del mismo. * Contar con la interfaz para la actualización del token. | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador, editor de la aplicación. * Seleccionar la opción redes sociales. * Seleccionar la opción conFiguración de Facebook. * Seleccionar el botón de actualización de token. | |
| **Resultado esperado:**   * Verificar que los datos enviados por la API son los correctos. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 128.** Prueba de aceptación HU9.2

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU9.2 | **Nombre y número de la Historia de Usuario** HU9. Permite modificar los datos de cuentas de la red social Facebook en OpenPublish. |
| **Nombre:** Guardar la información correctamente en la base de datos y emitir un mensaje de confirmación de la acción efectuada. | |
| **Responsable:** Edison Rodriguez | **Fecha:**04-06-2015 |
| **Descripción:** Verificar que los datos de la cuenta se guardan correctamente en la base de datos. | |
| **Condiciones de ejecución:**   * Contar con el método para la actualización del token en el archivo FacebookHelper. * Contar las funciones en el controlador, y el ruteo del mismo. * Contar con la interfaz para la actualización del token. | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador, editor de la aplicación. * Seleccionar la opción redes sociales. * Seleccionar la opción conFiguración de Facebook. * Seleccionar el botón de actualización de token. * Guardar los datos obtenidos. | |
| **Resultado esperado:**   * Verificar que los datos enviados por la API son guardados correctamente. * Emitir un mensaje de confirmación de la acción realizada. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 129.** HU10 Permite modificar los datos de videos en OpenPublish y Alavista.

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario:** Permite modificar los datos de videos en OpenPublish y Alavista. | | |
| **Número:** HU10 | **Nombre de la historia:** Modificar los datos de videos en OpenPublish y Alavista. | |
| **Modificación de historia de usuario:** Número y Nombre | | |
| **Usuario:** Administrador, Editor | | **Iteración Asignada:** 4 |
| **Prioridad en el Negocio:** Alta | | **Puntos Estimados:** 10 |
| **Riesgo en el Desarrollo:** Media | | **Puntos Reales:** 8 |
| **Descripción:** Como administrador, editor deseo editar la información de los videos que se ingresarán en OpenPublish y Alavista.tv | | |
| **Observaciones:** Editar un datos de un video individualmente, cambiar la categoría de un video individualmente | | |
| **Pruebas de Aceptación:**   * Cambiar la categoría de video al cambiar un elemento de selección. * Guardar correctamente los datos de un video y emitir un mensaje de confirmación | | |
| **Realizado por:** Villa E., Rodriguez E., 2015. | | |

**Tabla 130.** Tarea de ingeniería 1 HU10

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario:** HU10. Permite modificar los datos de videos en OpenPublish y Alavista. | |
| **Número tarea:** 1 | **Nombre de la tarea:** Crear métodos en el controlador y ruteo del proceso a realizar. |
| **Tipo de tarea:** Desarrollo | **Puntos estimados:** 6 |
| **Fecha de inicio:** 05-06-2015 | **Fecha fin:** 09-06-2015 |
| **Programador responsable:** Edwin Villa | |
| **Descripción:** Agregar la dirección de la operación en el archivo de ruteo, y realizar las funciones necesarias para modificar los datos de un video a guardar | |
| **Pruebas de aceptación:**   * Cumplir con el estándar de codificación. * Control de del tipo de ruteo correcto. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 130.** Tarea de ingeniería 1 HU10

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario:** HU10, Permite modificar los datos de videos en OpenPublish y Alavista. | |
| **Número tarea:** 2 | **Nombre de la tarea:** Crear interfaz para la vista que modifica videos encontrados |
| **Tipo de tarea:** Desarrollo | **Puntos estimados:** 4 |
| **Fecha de inicio:** 09-06-2015 | **Fecha fin: 11**-06-2015 |
| **Programador responsable:** Edwin Villa | |
| **Descripción:** Se creara un formulario para poder permitir modificar datos de un video | |
| **Pruebas de aceptación:**   * Cumplir con el estándar de interfaz. * Emitir un mensaje específico dependiendo el resultado de la solicitud. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 131.** Actividades HU10

|  |  |  |
| --- | --- | --- |
| **FECHA** | **ACTIVIDAD** | **TIEMPO** |
| Inicio: 05-06-2015  Fin: 09-06-2015  Inicio: 09-06-2015  Fin: 11-06-2015 | **Tarea: Crear métodos en el controlador y ruteo del proceso a realizar.**   * Crear el método para modificar los videos a seleccionar * Crear los métodos de ruteo post y get que se utilizaran   **Tarea: Crear interfaz para la vista que modifica videos encontrados**   * Crear el formulario para editar los datos de un video * Realizar el código Ajax para listar las categorías de los videos * Realizar el código Ajax generar los Tags de videos * Realizar el código Ajax para guardar los nuevos datos | 12h:00  1h:00  4h:00  4h:00  4h:00  4h:00 |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 132.** Prueba de aceptación HU10.1

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU10.1 | **Nombre y número de la Historia de Usuario** HU10. Permite modificar los datos de videos en OpenPublish y Alavista. |
| **Nombre:** Cambiar la categoría de video al cambiar un elemento de selección. | |
| **Responsable:** Alex Villa | **Fecha:**11-06-2015 |
| **Descripción:** Verificar el cambio de categoría de videos al seleccionar el elemento. | |
| **Condiciones de ejecución:**   * Contar con los métodos en el controlador y ruteo del proceso a realizar. * Contar con la interfaz para la vista que modifica videos encontrados. | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador, editor de la aplicación. * Seleccionar la opción Administrar videos. * Seleccionar la opción de Agregar videos. * Seleccionar uno de los canales de videos. * Realizar búsqueda. * Editar el resultado de la búsqueda. | |
| **Resultado esperado:**   * Cambio de categoría de los videos seleccionados, realizando el cambio en el campo de selección. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 133.** Prueba de aceptación HU10.2

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU10.2 | **Nombre y número de la Historia de Usuario** HU10. Permite modificar los datos de videos en OpenPublish y Alavista. |
| **Nombre:** Guardar correctamente los datos de un video y emitir un mensaje de confirmación | |
| **Responsable:** Alex Villa | **Fecha:**11-06-2015 |
| **Descripción:** Verificar el cambio de categoría de videos al seleccionar el elemento. | |
| **Condiciones de ejecución:**   * Contar con los métodos en el controlador y ruteo del proceso a realizar. * Contar con la interfaz para la vista que modifica videos encontrados. | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador, editor de la aplicación. * Seleccionar la opción Administrar videos. * Seleccionar la opción de Agregar videos. * Seleccionar uno de los canales de videos. * Realizar búsqueda. * Editar el resultado de la búsqueda. * Enviar los datos para el almacenamiento en la base de datos. | |
| **Resultado esperado:**   * Datos guardados correctamente en la base de datos. * Emitir un mensaje de confirmación del registro de los videos en la base de datos. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 134.** HU11 Permite buscar usuarios, cuentas de redes sociales en OpenPublish.

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario:** Permite buscar usuarios, cuentas de redes sociales en OpenPublish. | | |
| **Número:** HU11 | **Nombre de la historia:** Permite buscar usuarios, cuentas de redes sociales en OpenPublish. | |
| **Modificación de historia de usuario:** Número y Nombre | | |
| **Usuario:** Administrador, Editor | | **Iteración Asignada:** 4 |
| **Prioridad en el Negocio:** Alta | | **Puntos Estimados:** 8 |
| **Riesgo en el Desarrollo:** Media | | **Puntos Reales:**6 |
| **Descripción:** Como administrador deseo contar con la opción de poder buscar videos, usuarios, cuentas de redes sociales en OpenPublish.  Como editor deseo contar con la opción de poder buscar videos, cuentas de redes sociales en OpenPublish. | | |
| **Observaciones:** El administrador podrá realizar búsquedas de usuarios, videos, cuentas en OpenPublish, mientras que al editor se le restringirá observar a los usuarios. | | |
| **Pruebas de Aceptación:**   * Ingresar el nombre ya sea de video, usuario, cuenta a buscar en el campo de búsqueda, posterior se mostrara el resultado de la búsqueda. | | |
| **Realizado por:** Villa E., Rodriguez E., 2015. | | |

**Tabla 135.** Tarea de ingeniería 1 HU11

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario:** HU11. Permite buscar usuarios, cuentas de redes sociales en OpenPublish. | |
| **Número tarea**: 1 | **Nombre de la tarea:** Crear la interfaz para la búsqueda de usuarios y cuentas de redes sociales. |
| **Tipo de tarea:** Desarrollo | **Puntos estimados:** 8 |
| **Fecha de inicio:** 12-06-2015 | **Fecha fin:** 17-06-2015 |
| **Programador responsable:** Edwin Villa | |
| **Descripción:** Se hará uso de la tabla disponible para presentar los datos, en la cual se agregara el botón de búsqueda de los registros presentes en la tabla. | |
| **Pruebas de aceptación:**   * Cumplir con el estándar de la interfaz. * Emitir un mensaje de resultado de la búsqueda. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 136.** Actividades HU11

|  |  |  |
| --- | --- | --- |
| **FECHA** | **ACTIVIDAD** | **TIEMPO** |
| Inicio: 18-06-2015  Fin: 19-06-2015 | Tarea: **Crear la interfaz para la búsqueda de usuarios y cuentas de redes sociales.**   * Creación y codificación del campo de texto para ingresar el texto a buscar. * Realizar el código Ajax para presentar los resultados. | 10h:00  10h:00 |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 137.** Prueba de aceptación HU11.1

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU11.1 | **Nombre y número de la Historia de Usuario** HU11 Permite buscar usuarios, cuentas de redes sociales en OpenPublish. |
| **Nombre:** Ingresar el nombre ya sea de video, usuario, cuenta a buscar en el campo de búsqueda, posterior se mostrara el resultado de la búsqueda. | |
| **Responsable:** Edwin Villa | **Fecha:**19-06-2015 |
| **Descripción:** Revisar el resultado obtenido a partir del texto ingresado para la consulta. | |
| **Condiciones de ejecución:**   * Contar con la interfaz para la búsqueda de usuarios y cuentas de redes sociales. * Listar el contenido disponible de la base de datos, ya sean usuarios o cuentas. | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador, editor de la aplicación. * Seleccionar la opción administrar cuentas o administrar usuarios. * Agregar el texto a buscar en el campo de búsqueda. * Realizar búsqueda. | |
| **Resultado esperado:**   * Datos obtenidos de acuerdo a la búsqueda realizada. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 138.** HU12 Permite eliminar usuarios y videos de OpenPublish y Alavista.tv.

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario:** Permite eliminar usuarios y videos de OpenPublish y Alavista.tv. | | |
| **Número:** HU12 | **Nombre de la historia:** Eliminar usuarios y videos de OpenPublish y Alavista.tv. | |
| **Modificación de historia de usuario:** Número y Nombre | | |
| **Usuario:** Administrador, Editor | | **Iteración Asignada:** 5 |
| **Prioridad en el Negocio:** Alta | | **Puntos Estimados:** 8 |
| **Riesgo en el Desarrollo:** Media | | **Puntos Reales:7** |
| **Descripción:** Como administrador deseo eliminar usuarios de OpenPublish, como administrador, editor deseo eliminar videos de OpenPublish y Alavista.tv, | | |
| **Observaciones:** Se debe listar los videos almacenados y poder identificar los videos que se encuentran tanto en OpenPublish como en Alavista.tv | | |
| **Pruebas de Aceptación:**   * Listar los todos los videos almacenados e identificar que videos se encuentran en Alavista.tv y OpenPublish * Mostrar una lista de videos con paginación, realizar una nueva petición al cambiar de página * Al no seleccionar ningún video se emitirá el mensaje correspondiente * Guardar correctamente los cambios realizados y emitir un mensaje de confirmación | | |
| **Realizado por:** Villa E., Rodriguez E., 2015. | | |

**Tabla 139.** Tarea de ingeniería 1 HU12

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario:** HU12, Permite eliminar usuarios y videos de OpenPublish y Alavista.tv. | |
| **Número tarea:** 1 | **Nombre de la tarea**: Crear métodos en el controlador y ruteo del proceso a realizar. |
| **Tipo de tarea:** Desarrollo | **Puntos estimados:** 4 |
| **Fecha de inicio:** 18-06-2015 | **Fecha fin:** 19-06-2015 |
| **Programador responsable**: Edwin Villa | |
| **Descripción:** Agregar la dirección de la operación en el archivo de ruteo, y realizar las funciones necesarias para eliminar usuarios y videos de OpenPublish y Alavista.tv. | |
| **Pruebas de aceptación:**   * Cumplir con el estándar de codificación. * Control de del tipo de ruteo correcto. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 140.** Tarea de ingeniería 2 HU12

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario**: HU12, Permite eliminar usuarios y videos de OpenPublish y Alavista.tv. | |
| **Número tarea:** 2 | **Nombre de la tarea:** Crear interfaz para la vista que elimina usuarios y videos de OpenPublish y Alavista.tv. |
| **Tipo de tarea:** Desarrollo | **Puntos estimados:** 4 |
| **Fecha de inicio:** 22-06-2015 | **Fecha fin:** 24-06-2015 |
| **Programador responsable:** Edwin Villa | |
| **Descripción:** Se creara un formulario para poder permitir eliminar datos videos y usuarios de OpenPublish y Alavista.tv. | |
| **Pruebas de aceptación:**   * Cumplir con el estándar de interfaz. * Emitir un mensaje específico dependiendo el resultado de la solicitud. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 141.** Actividades HU12

|  |  |  |
| --- | --- | --- |
| **FECHA** | **ACTIVIDAD** | **TIEMPO** |
| Inicio: 18-06-2015  Fin: 19-06-2015  Inicio: 22-06-2015  Fin: 24-06-2015 | **Tarea: Crear métodos en el controlador y ruteo del proceso a realizar.**   * Crear la función para eliminar los usuarios de OpenPublish * Crear la función para listar los videos de OpenPublish y Alavista.tv. * Crear la función para eliminar los usuarios de OpenPublish * Crear la función para eliminar los videos de OpenPublish y Alavista.tv   **Tarea: Crear interfaz para la vista que elimina usuarios y** **videos de OpenPublish y Alavista.tv.**   * Crear la interfaz de la ventana modal donde se presenta información del usuario a eliminar. * Realizar el código AJAX para la eliminación del usuario seleccionado y presentación de resultados. * Realizar el código javascript seleccionar todos los videos presentados en una página. * Realizar el código AJAX para la eliminar los videos seleccionados | 4h:00  2h:00  4h:00  4h:00  4h:00  4h:00  4h:00  2h:00 |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 142.** Prueba de aceptación HU12.1

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU12.1 | **Nombre y número de la Historia de Usuario** HU12 Permite eliminar usuarios y videos de OpenPublish y Alavista.tv. |
| **Nombre:** Listar todos los videos almacenados e identificar que videos se encuentran en Alavista.tv y OpenPublish. | |
| **Responsable:** Edwin Villa | **Fecha:**24-06-2015 |
| **Descripción:** Revisar el listado de los videos almacenados en la base de datos. | |
| **Condiciones de ejecución:**   * Contar con la conexión a las bases de datos necesarias. * Contar con los métodos del controlador y su ruteo correspondiente. * Contar con la interfaz de la vista para realizar la eliminación de Videos. * Contar con la interfaz de la vista para realizar la eliminación de usuarios. | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador, editor de la aplicación. * Seleccionar la opción administrar videos o administrar usuarios. | |
| **Resultado esperado:**   * Presentar los videos, usuarios en la tabla de manera dinámica | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 143.** Prueba de aceptación HU12.2

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU12.2 | **Nombre y número de la Historia de Usuario** HU12 Permite eliminar usuarios y videos de OpenPublish y Alavista.tv. |
| **Nombre:** Mostrar una lista de videos con paginación, realizar una nueva petición al cambiar de página | |
| **Responsable:** Edwin Villa | **Fecha:**24-06-2015 |
| **Descripción:** Revisar el listado de los videos de manera dinámica mediante la paginación. | |
| **Condiciones de ejecución:**   * Contar con la conexión a las bases de datos necesarias. * Contar con los métodos del controlador y su ruteo correspondiente. * Contar con la interfaz de la vista para realizar la eliminación de Videos. * Contar con la interfaz de la vista para realizar la eliminación de usuarios. | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador, editor de la aplicación. * Seleccionar la opción administrar videos o administrar usuarios. | |
| **Resultado esperado:**   * La presentación de los registros en las tablas de manera dinámica, de acuerdo a la paginación. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 144.** Prueba de aceptación HU12.3

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU12.3 | **Nombre y número de la Historia de Usuario** HU12 Permite eliminar usuarios y videos de OpenPublish y Alavista.tv. |
| **Nombre:** Al no seleccionar ningún video se emitirá el mensaje correspondiente | |
| **Responsable:** Edwin Villa | **Fecha:**24-06-2015 |
| **Descripción:** Se notificara mediante un mensaje que no ha seleccionado un video a eliminar. | |
| **Condiciones de ejecución:**   * Contar con la conexión a las bases de datos necesarias. * Contar con los métodos del controlador y su ruteo correspondiente. * Contar con la interfaz de la vista para realizar la eliminación de Videos. * Contar con la interfaz de la vista para realizar la eliminación de usuarios. | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador, editor de la aplicación. * Seleccionar la opción administrar videos o administrar usuarios. * Seleccionar el video, o usuario a eliminar. | |
| **Resultado esperado:**   * Presentar un mensaje de notificación que no ha seleccionado elementos a eliminar. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 145.** Prueba de aceptación HU12.4

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU12.4 | **Nombre y número de la Historia de Usuario** HU12 Permite eliminar usuarios y videos de OpenPublish y Alavista.tv. |
| **Nombre:** Guardar correctamente los cambios realizados y emitir un mensaje de confirmación | |
| **Responsable:** Edwin Villa | **Fecha:**24-06-2015 |
| **Descripción:** Se notificara mediante un mensaje que la acción realizada se ejecutó correctamente. | |
| **Condiciones de ejecución:**   * Contar con la conexión a las bases de datos necesarias. * Contar con los métodos del controlador y su ruteo correspondiente. * Contar con la interfaz de la vista para realizar la eliminación de Videos. * Contar con la interfaz de la vista para realizar la eliminación de usuarios. | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador, editor de la aplicación. * Seleccionar la opción administrar videos o administrar usuarios. * Seleccionar el video, o usuario a eliminar. * Realizar la petición de eliminación. | |
| **Resultado esperado:**   * Guardar el cambio realizado en la base de datos y proporcionar un mensaje de confirmación de la eliminación. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 146.** HU13: Permite eliminar cuentas de redes sociales de OpenPublish.

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario:** Permite eliminar cuentas de redes sociales de OpenPublish. | | |
| **Número:** HU13 | **Nombre de la historia:** Permite eliminar cuentas de redes sociales de OpenPublish. | |
| **Modificación de historia de usuario:** Número y Nombre | | |
| **Usuario:** Administrador, Editor | | **Iteración Asignada:** 5 |
| **Prioridad en el Negocio:** Media | | **Puntos Estimados:** 4 |
| **Riesgo en el Desarrollo:** Media | | **Puntos Reales:** 5 |
| **Descripción:** Como administrador deseo contar con la opción de poder eliminar cualquier cuenta de redes sociales en OpenPublish.  Como editor deseo contar con la opción de poder eliminar cuentas de redes sociales que haya registrado en OpenPublish. | | |
| **Observaciones:** El administrador podrá eliminar cualquier cuenta de redes sociales en OpenPublish, mientras que al editor podrá eliminar cuentas que haya registrado posteriormente. | | |
| **Pruebas de Aceptación:**   * Presentar el listado de las cuentas de Facebook o Twitter. * Realizar la eliminación de la cuenta de redes sociales en la Base de datos, y emitir un mensaje de confirmación de la acción efectuada. | | |
| **Realizado por:** Villa E., Rodriguez E., 2015. | | |

**Tabla 147.** Tarea de ingeniería 1 HU13

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario:** HU13. Permite eliminar cuentas de redes sociales de OpenPublish. | |
| **Número tarea:** 1 | **Nombre de la tarea:** Crear las funciones para eliminar cuentas de redes sociales en el controlador de Facebook y Twitter y realizar el ruteo correspondiente. |
| **Tipo de tarea:** Desarrollo | **Puntos estimados:** 3 |
| **Fecha de inicio:** 24-06-2015 | **Fecha fin:** 25-06-2015 |
| Programador responsable: Edison Rodriguez | |
| **Descripción:** Se creara la función para eliminar cuentas en cada uno de los controladores correspondientes a Facebook y Twitter. | |
| **Pruebas de aceptación:**   * Cumplir con el estándar de la interfaz. * La función retorna mensaje de resultado. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 148.** Tarea de ingeniería 2 HU13

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario:** HU13. Permite eliminar cuentas de redes sociales de OpenPublish. | |
| **Número tarea:** 2 | **Nombre de la tarea:** Crear la interfaz para eliminar las cuentas de redes sociales. |
| **Tipo de tarea:** Desarrollo | **Puntos estimados:** 1 |
| **Fecha de inicio:** 26-06-2015 | **Fecha fin:** 26-06-2015 |
| **Programador responsable:** Edison Rodriguez | |
| **Descripción:** Se agregara la funcionalidad en el botón de eliminar cuentas en la tabla donde se presenta la información de las cuentas, además la ventana modal donde se observa detalladamente la cuenta a eliminar y la opción de confirmación. | |
| **Pruebas de aceptación:**   * Cumplir con el estándar de la interfaz. * Presentar mensaje de confirmación de la eliminación. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 149.** Actividades HU13

|  |  |  |
| --- | --- | --- |
| **FECHA** | **ACTIVIDAD** | **TIEMPO** |
| Inicio: 24-06-2015  Fin:25-06-2015  Inicio: 26-06-2015  Fin: 26-06-2015 | **Tarea: Crear las funciones para eliminar cuentas de redes sociales en el controlador de Facebook y Twitter.**   * Crear la función para eliminar las cuentas de Facebook en el controlador FacebookController. * Crear la función para eliminar las cuentas de Twitter en el controlador TwitterController. * Realizar el ruteo correspondiente para el controlador.   **Tarea: Crear la interfaz para eliminar las cuentas de redes sociales.**   * Crear la interfaz de la ventana modal donde se presenta información de la cuenta a eliminar. * Realizar el código AJAX para la eliminación de la cuenta y presentación de resultados. | 5h:00  5h:00  2h:00  2h:00  2h:00 |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 150.** Prueba de aceptación HU13.1

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU13.1 | **Nombre y número de la Historia de Usuario**. HU13. Permite eliminar cuentas de redes sociales de OpenPublish. |
| **Nombre:** Presentar el listado de las cuentas de Facebook o Twitter. | |
| **Responsable:** Edison Rodriguez | **Fecha:**26-06-2015 |
| **Descripción:** Observar la lista de las cuentas de Facebook o Twitter. | |
| **Condiciones de ejecución:**   * Contener cuentas de redes sociales en las bases de datos. * Contar con las funciones para eliminar cuentas de Facebook * Contar con las funciones para eliminar cuentas de Twitter. * Contar con el ruteo correspondiente para el controlador. * Contar con la interfaz para eliminar las cuentas de redes sociales. | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador, editor de la aplicación. * Seleccionar la opción de redes sociales. * Seleccionar administrar una red social. | |
| **Resultado esperado:**   * Listado de las cuentas de red social de manera dimamica la paginación. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 151.** Prueba de aceptación HU13.2

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU13.2 | **Nombre y número de la Historia de Usuario**. HU13. Permite eliminar cuentas de redes sociales de OpenPublish. |
| **Nombre:** Realizar la eliminación de la cuenta de redes sociales en la Base de datos, y emitir un mensaje de confirmación de la acción efectuada. | |
| **Responsable:** Edison Rodriguez | **Fecha:**26-06-2015 |
| **Descripción:** Realizar la eliminación correcta de una cuenta de red social, y presentar un mensaje de la confirmación de la acción realizada. | |
| **Condiciones de ejecución:**   * Contener cuentas de redes sociales en las bases de datos. * Contar con las funciones para eliminar cuentas de Facebook * Contar con las funciones para eliminar cuentas de Twitter. * Contar con el ruteo correspondiente para el controlador. * Contar con la interfaz para eliminar las cuentas de redes sociales. | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador, editor de la aplicación. * Seleccionar la opción de redes sociales. * Seleccionar administrar una red social. * Eliminar una red social. | |
| **Resultado esperado:**   * Eliminación correcta de la cuenta en la base de datos. * Presentar un mensaje de confirmación de la acción realizada. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 152.** HU14 Permite publicar contenido de Alavista.tv en la red social Facebook.

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario:** Permite publicar contenido de Alavista.tv en la red social Facebook | | |
| **Número:** HU14 | **Nombre de la historia:** Permite publicar contenido de Alavista.tv en la red social Facebook | |
| **Modificación de historia de usuario:** Número y Nombre | | |
| **Usuario:** Administrador, Editor | | **Iteración Asignada:** 5 |
| **Prioridad en el Negocio:** Alta | | **Puntos Estimados:** 10 |
| **Riesgo en el Desarrollo:** Alta | | **Puntos Reales:** 12 |
| **Descripción:** Como administrador, editor deseo realizar publicaciones del contenido de Alavista.tv en redes sociales, Facebook ya sea con una o varias cuentas registradas. | | |
| **Observaciones:** En el caso de publicar en Facebook tendrá la posibilidad de publicar en el perfil de la cuenta, grupos, eventos, páginas. | | |
| **Pruebas de Aceptación:**   * Hacer uso de la APIs de redes sociales para obtener la información de las cuentas para realizar la publicación, en el caso de error se presentara el mensaje correspondiente. * Seleccionar por lo mínimo un identificador de la cuenta ya sea un perfil, grupo, evento, pagina, si no realiza esta acción se presentara un mensaje de notificación. * Agregar un mensaje a la publicación, si no realiza esta acción se presentara un mensaje de notificación. * El contenido para la publicación se envía a la API para que realice la publicación, y se emitirá un mensaje de notificación si la acción se realizó correctamente o si hubo algún error. | | |
| **Realizado por:** Villa E., Rodriguez E., 2015. | | |

**Tabla 153.** Tarea de ingeniería 1 HU14

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario:** HU14. Permite publicar contenido de Alavista.tv en la red social Facebook | |
| **Número tarea:** 1 | **Nombre de la tarea:** Instalar el complemento datatables. |
| **Tipo de tarea:** Desarrollo | **Puntos estimados:** 1 |
| **Fecha de inicio:** 26-06-2015 | **Fecha fin:** 26-06-2015 |
| **Programador responsable:** Edison Rodriguez | |
| **Descripción:** Se realiza la instalación del plugin datatables para realizar el despliegue de los videos disponibles en Alavista.tv de manera dinámica, para mejorar experiencia del usuario. | |
| **Pruebas de aceptación:**   * Plugin instalado y conFigurado listo para utilizarlo. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 154.** Tarea de ingeniería 2 HU14

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario:** HU14. Permite publicar contenido de Alavista.tv en la red social Facebook | |
| **Número tarea:** 2 | **Nombre de la tarea:** Crear la función en el VideoController para listar los videos dinámicamente de Alavista.tv |
| **Tipo de tarea:** Desarrollo | **Puntos estimados:** 1 |
| **Fecha de inicio:** 29-06-2015 | **Fecha fin:** 29-06-2015 |
| **Programador responsable:** Edison Rodriguez | |
| **Descripción:** Se agregara la función para listar los videos de Alavista.tv de manera dinámica en el controlador VideoController, haciendo uso del plugin instalado. | |
| **Pruebas de aceptación:**   * Cumplir con el estándar de conFiguración. * Obtener los videos de acuerdo a la acción efectuada por el usuario. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 155.** Tarea de ingeniería 3 HU14

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario**: HU14. Permite publicar contenido de Alavista.tv en la red social Facebook | |
| **Número tarea:** 3 | **Nombre de la tarea:** Crear los métodos para obtener la información de las cuentas, método para realizar la publicación en el archivo FacebookHelper. |
| **Tipo de tarea:** Desarrollo | **Puntos estimados:** 3 |
| **Fecha de inicio:** 29-06-2015 | **Fecha fin:** 30-06-2015 |
| **Programador responsable:** Edison Rodriguez | |
| **Descripción:** Se crea los métodos para obtener la información de la cuenta como, grupos pertenecientes, páginas, eventos, así también como el método para realizar la publicación. | |
| **Pruebas de aceptación:**   * Cumplir con el estándar de codificación. * Verificar la información de las cuentas son las correctas. * Verificar el proceso de publicación es correcto. * Emitir un mensaje de respuesta de la API de las acciones realizadas. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 156.** Tarea de ingeniería 4 HU14

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario:** HU14. Permite publicar contenido de Alavista.tv en la red social Facebook. | |
| **Número tarea: 4** | **Nombre de la tarea**: Crear las funciones en el controlador FacebookController y el ruteo correspondiente. |
| **Tipo de tarea**: Desarrollo | **Puntos estimados:** 3 |
| **Fecha de inicio:** 01-07-2015 | **Fecha fin:** 02-07-2015 |
| **Programador responsable:** Edison Rodriguez | |
| Descripción: Se creara las funciones en el controlador para obtener la información de la cuenta, como son grupos asociados, eventos, páginas, así como la función para realizar la publicación del video. | |
| **Pruebas de aceptación:**   * Cumplir con el estándar de conFiguración. * Verificar la información obtenida es la correcta. * Emitir mensajes correspondientes a la respuesta de la API * Enviar mensajes específicos para errores comunes que se presentan al momento de consumir recursos de la API. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 157.** Tarea de ingeniería 5 HU14

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario:** HU14. Permite publicar contenido de Alavista.tv en la red social Facebook. | |
| **Número tarea:** 5 | **Nombre de la tarea:** Crear la interfaz de publicaciones en Facebook. |
| **Tipo de tarea:** Desarrollo | **Puntos estimados:** 2 |
| **Fecha de inicio:** 02-07-2015 | **Fecha fin:** 03-07-2015 |
| **Programador responsable:** Edison Rodriguez | |
| **Descripción:** Crear la vista para realizar la publicación de los videos en Facebook. | |
| **Pruebas de aceptación:**   * Verificar la carga de videos para realizar la publicación. * Verificar la información de cada cuenta se cargue correctamente. * Emitir mensajes de error en el caso que exista al momento de cargar las cuentas de Facebook. * Verificar que se seleccione por lo menos una de las opciones para realizar la publicación. * Obtener mensajes de confirmación de la publicación realizada. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 158.** Actividades HU14

|  |  |  |
| --- | --- | --- |
| **FECHA** | **ACTIVIDAD** | **TIEMPO** |
| Inicio:26-06-2015  Fin: 26-06-2015  Inicio:29-06-2015  Fin: 29-06-2015  Inicio:29-06-2015  Fin: 30-07-2015  Inicio:01-07-2015  Fin:02-07-2015  Inicio:02-07-2015  Fin: 03-07-2015 | **Tarea: Instalar el complemento datatables.**   * ConFigurar el archivo composer para realizar la instalación. * Realizar la instalación mediante la actualización de composer. * Spike solution: Realizar prueba para generar una tabla dinámicamente en archivos independientes.   **Tarea: Crear la función para en el VideoController para listar los videos dinámicamente de Alavista.tv.**   * Crear la función para listar los videos de acuerdo a la acción realizada por el usuario. * Agregar el ruteo para el controlador en el archivo route.   **Tarea: Crear los métodos para obtener la información de las cuentas, método para realizar la publicación en el archivo FacebookHelper.**   * Spike solution: Realizar pruebas para consumir los servicios de la API de Facebook. * Crear el método para obtener la información del perfil de la cuenta. * Crear el método para obtener los grupos asociados a la cuenta. * Crear el método para obtener las páginas administradas por la cuenta. * Crear el método para obtener los eventos a los cuales se encuentre asociada una cuenta. * Crear el método para realizar la publicación del video en Facebook.   **Tarea: Crear las funciones en el controlador FacebookController y el ruteo correspondiente.**   * Spike solution: Realizar pruebas para consumir las funciones realizadas en el archivo FacebookHelper. * Crear la función para listar las cuentas de Facebook conjuntamente con grupos, paginas, eventos, datos personales asociadas al usuario de OpenPublish * Crear la función para publicar los videos en Facebook. * Agregar el ruteo para el controlador en el archivo route.   **Tarea: Crear la interfaz de publicaciones en Facebook.**   * Crear las funciones AJAX para listar los videos. * Crear las funciones AJAX para la publicación de videos. * Realizar la interfaz para listar los videos disponibles para publicar. * Realizar la interfaz para la publicación de los videos. * Presentar mensajes del resultado de la publicación. | 2h:00  1h:00  1h:00  3h:00  1h:00  2h:00  2h:00  2h:00  2h:00  2h:00  2h:00  2h:00  6h:00  3h:00  1h:00  1h:00  1h:00  2h:00  2h:00  2h:00 |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 159.** Prueba de aceptación HU14.1

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU14.1 | **Nombre y número de la Historia de Usuario**. HU14 Permite publicar contenido de Alavista.tv en la red social Facebook |
| **Nombre:** Hacer uso de la APIs de redes sociales para obtener la información de las cuentas para realizar la publicación, en el caso de error se presentara el mensaje correspondiente. | |
| **Responsable:** Edison Rodriguez | **Fecha:**03-07-2015 |
| **Descripción:** Verificar el uso correcto de la API de Facebook para obtener los datos de las cuentas. | |
| **Condiciones de ejecución:**   * Contener datos de cuentas en la base de datos. * Contener videos en la base de datos de OpenPublish * Tener las funciones del VideoController para listar los videos de manera dinámica. * Tener los métodos para obtener la información de las cuentas, método para realizar la publicación en el archivo FacebookHelper. * Tener las funciones en el controlador FacebookController y el ruteo correspondiente. * Tener la interfaz de publicaciones en Facebook. | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador o editor de la aplicación. * Seleccionar la opción de redes sociales. * Seleccionar la opción de publicar en Facebook. | |
| **Resultado esperado:**   * Presentar la información de las cuentas asociadas al usuario de OpenPublish. * Presentar los grupos, eventos, páginas de acuerdo a cada cuenta de Facebook. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 160.** Prueba de aceptación HU14.2

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU14.2 | **Nombre y número de la Historia de Usuario**. HU14 Permite publicar contenido de Alavista.tv en la red social Facebook |
| **Nombre:** Seleccionar por lo mínimo un identificador de la cuenta ya sea un perfil, grupo, evento, pagina, si no realiza esta acción se presentara un mensaje de notificación. | |
| **Responsable:** Edison Rodriguez | **Fecha:**03-07-2015 |
| **Descripción:** Verificar que se selección por lo mínimo un identificador para realizar la publicación. | |
| **Condiciones de ejecución:**   * Contener datos de cuentas en la base de datos. * Contener videos en la base de datos de OpenPublish * Tener las funciones del VideoController para listar los videos de manera dinámica. * Tener los métodos para obtener la información de las cuentas, método para realizar la publicación en el archivo FacebookHelper. * Tener las funciones en el controlador FacebookController y el ruteo correspondiente. * Tener la interfaz de publicaciones en Facebook. | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador o editor de la aplicación. * Seleccionar la opción de redes sociales. * Seleccionar la opción de publicar en Facebook. * Seleccionar un identificador de cuenta para realizar la publicación. | |
| **Resultado esperado:**   * Presentar mensaje de notificación que no se ha seleccionado ninguna cuenta para realizar la publicación. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 161.** Prueba de aceptación HU14.3

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU14.3 | **Nombre y número de la Historia de Usuario**. HU14 Permite publicar contenido de Alavista.tv en la red social Facebook |
| **Nombre:** Agregar un mensaje a la publicación, si no realiza esta acción se presentara un mensaje de notificación. | |
| **Responsable:** Edison Rodriguez | **Fecha:**03-07-2015 |
| **Descripción:** Verificar que se ha agregado un mensaje a la publicación. | |
| **Condiciones de ejecución:**   * Contener datos de cuentas en la base de datos. * Contener videos en la base de datos de OpenPublish * Tener las funciones del VideoController para listar los videos de manera dinámica. * Tener los métodos para obtener la información de las cuentas, método para realizar la publicación en el archivo FacebookHelper. * Tener las funciones en el controlador FacebookController y el ruteo correspondiente. * Tener la interfaz de publicaciones en Facebook. | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador o editor de la aplicación. * Seleccionar la opción de redes sociales. * Seleccionar la opción de publicar en Facebook. * Seleccionar un identificador de cuenta para realizar la publicación. * Agregar un mensaje a la plucación. | |
| **Resultado esperado:**   * Presentar mensaje de notificación que no se ha agregado ningún mensaje a la publicación que se realizara en Facebook. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 162.** Prueba de aceptación HU14.4

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU14.4 | **Nombre y número de la Historia de Usuario**. HU14 Permite publicar contenido de Alavista.tv en la red social Facebook |
| **Nombre:** El contenido para la publicación se envía a la API para que realice la publicación, y se emitirá un mensaje de notificación si la acción se realizó correctamente o si hubo algún error. | |
| **Responsable:** Edison Rodriguez | **Fecha:**03-07-2015 |
| **Descripción:** Verificar que la publicación se realizó correctamente en Facebook. | |
| **Condiciones de ejecución:**   * Contener datos de cuentas en la base de datos. * Contener videos en la base de datos de OpenPublish * Tener las funciones del VideoController para listar los videos de manera dinámica. * Tener los métodos para obtener la información de las cuentas, método para realizar la publicación en el archivo FacebookHelper. * Tener las funciones en el controlador FacebookController y el ruteo correspondiente. * Tener la interfaz de publicaciones en Facebook. | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador o editor de la aplicación. * Seleccionar la opción de redes sociales. * Seleccionar la opción de publicar en Facebook. * Seleccionar un identificador de cuenta para realizar la publicación. * Agregar un mensaje a la aplicación. * Realizar la publicación. | |
| **Resultado esperado:**   * Presentar mensaje de notificación donde se evidencie el identificador de la publicación realizada en Facebook. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 163.** HU15 Permite publicar contenido de Alavista.tv en la red social Twitter

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario:** Permite publicar contenido de Alavista.tv en rede social Twitter | | |
| **Número:** HU15 | **Nombre de la historia:** Permite publicar contenido de Alavista.tv en rede social Twitter. | |
| **Modificación de historia de usuario:** Número y Nombre | | |
| **Usuario:** Administrador, Editor | | **Iteración Asignada:** 6 |
| **Prioridad en el Negocio:** Alta | | **Puntos Estimados:** 10 |
| **Riesgo en el Desarrollo:** Alta | | **Puntos Reales:** 12 |
| **Descripción:** Como administrador, editor deseo realizar publicaciones del contenido de Alavista.tv en redes sociales en twitter ya sea con una o varias cuentas registradas. | | |
| **Observaciones:** En el caso de publicar en Twitter tendrá la posibilidad de realizar un Tweet añadiendo elementos de las tendencias en el País. | | |
| **Pruebas de Aceptación:**   * Hacer uso de la APIs de redes sociales para obtener la información de las cuentas para realizar la publicación, en el caso de error se presentara el mensaje correspondiente. * Seleccionar por lo mínimo un identificador de la cuenta, seleccionar si se desea las tendencias del país, si no realiza esta acción se presentara un mensaje de notificación. * Agregar un mensaje a la publicación, si no realiza esta acción se presentara un mensaje de notificación. * El contenido para la publicación se envía a la API para que realice la publicación, y se emitirá un mensaje de notificación si la acción se realizó correctamente o si hubo algún error. | | |
| **Realizado por:** Villa E., Rodriguez E., 2015 | | |

**Tabla 164.** Tarea de ingeniería 1 HU15

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario:** HU15. Permite publicar contenido de Alavista.tv en la red social Twitter | |
| Número tarea: **1** | **Nombre de la tarea:** Crear los métodos para obtener la información de las cuentas, método para realizar la publicación en el archivo TwitterHelper. |
| **Tipo de tarea:** Desarrollo | **Puntos estimados:** 4 |
| **Fecha de inicio:** 06-07-2015 | **Fecha fin:** 07-07-2015 |
| **Programador responsable:** Edison Rodriguez | |
| **Descripción:** Se crea los métodos para obtener la información de la cuenta como nombre, apellidos, edad, correo, tendencias en el País, así también como el método para realizar la publicación. | |
| **Pruebas de aceptación:**   * Cumplir con el estándar de codificación. * Verificar la información de las cuentas son las correctas. * Verificar el proceso de publicación es correcto. * Emitir un mensaje de respuesta de la API de las acciones realizadas. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 165.** Tarea de ingeniería 2 HU15

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario:** HU15. Permite publicar contenido de Alavista.tv en la red social Twitter. | |
| **Número tarea:** 2 | **Nombre de la tarea:** Crear las funciones en el controlador TwitterController y el ruteo correspondiente. |
| **Tipo de tarea:** Desarrollo | **Puntos estimados:** 4 |
| **Fecha de inicio:** 08-07-2015 | **Fecha fin:** 09-07-2015 |
| **Programador responsable:** Edison Rodriguez | |
| **Descripción:** Se creara las funciones en el controlador para obtener la información de la cuenta, como son nombre, apellido, edad, correo, tendencias en el País, así como la función para realizar la publicación del video. | |
| **Pruebas de aceptación:**   * Cumplir con el estándar de conFiguración. * Verificar la información obtenida es la correcta. * Emitir mensajes correspondientes a la respuesta de la API * Enviar mensajes específicos para errores comunes que se presentan al momento de consumir recursos de la API. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 166.** Tarea de ingeniería 3 HU15

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario:** HU15. Permite publicar contenido de Alavista.tv en la red social Twitter. | |
| **Número tarea:** 3 | **Nombre de la tarea:** Crear la interfaz de publicaciones en Twitter. |
| **Tipo de tarea:** Desarrollo | **Puntos estimados:** 2 |
| **Fecha de inicio:** 10-07-2015 | **Fecha fin:** 10-07-2015 |
| **Programador responsable:** Edison Rodriguez | |
| **Descripción:** Crear la vista para realizar la publicación de los videos en Twitter. | |
| Pruebas de aceptación:   * Verificar la carga de videos para realizar la publicación. * Verificar la información de cada cuenta se cargue correctamente. * Emitir mensajes de error en el caso que exista al momento de cargar las cuentas de Twitter. * Verificar que se seleccione por lo menos una de las opciones para realizar la publicación. * Obtener mensajes de confirmación de la publicación realizada. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 167.** Actividades HU15

|  |  |  |
| --- | --- | --- |
| **FECHA** | **ACTIVIDAD** | **TIEMPO** |
| Inicio: 06-07-2015  Fin: 07-07-2015  Inicio: 08-07-2015  Fin: 09-07-2015  Inicio:10-07-2015  Fin:10-07-2015 | **Tarea: Crear los métodos para obtener la información de las cuentas, método para realizar la publicación en el archivo TwitterHelper.**   * Spike solution: Realizar pruebas para consumir los servicios de la API de Twitter. * Crear el método para obtener la información del perfil de la cuenta. * Crear el método para obtener las tendencias del país. * Crear el método para realizar la publicación del video en Twitter   **Tarea: Crear las funciones en el controlador TwitterController y el ruteo correspondiente.**   * Spike Solutión: Realizar pruebas para obtener información del archivo TwitterHelper. * Crear la función para listar las cuentas de Twitter asociadas al usuario de OpenPublish. * Crear la función para publicar los videos en Twitter. * Agregar el ruteo para el controlador en el archivo * route.   **Tarea: Crear la interfaz de publicaciones en Twitter.**   * Crear las funciones AJAX para listar los videos. * Crear las funciones AJAX para la publicación de videos. * Realizar la interfaz para listar los videos disponibles para publicar. * Realizar la interfaz para la publicación de los videos. * Presentar mensajes del resultado de la publicación. | 4h:00  4h:00  4h:00  4h:00  4h:00  4h:00  4h:00  4h:00  2h:00  2h:00  1h:00  2h:00  1h:00 |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 168.** Prueba de aceptación HU15.1

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU15.1 | **Nombre y número de la Historia de Usuario**. HU15 Permite publicar contenido de Alavista.tv en la red social Twitter |
| **Nombre:** Hacer uso de la APIs de redes sociales para obtener la información de las cuentas para realizar la publicación, en el caso de error se presentara el mensaje correspondiente. | |
| **Responsable:** Edison Rodriguez | **Fecha:**10-07-2015 |
| **Descripción:** Verificar el uso correcto de la API de Twitter para obtener los datos de las cuentas. | |
| **Condiciones de ejecución:**   * Contener datos de cuentas en la base de datos. * Contener videos en la base de datos de OpenPublish * Tener las funciones del VideoController para listar los videos de manera dinámica. * Tener los métodos para obtener la información de las cuentas, método para realizar la publicación en el archivo TwitterHelper. * Tener las funciones en el controlador TwitterController y el ruteo correspondiente. * Tener la interfaz de publicaciones en Twitter. | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador o editor de la aplicación. * Seleccionar la opción de redes sociales. * Seleccionar la opción de publicar en Twitter. | |
| **Resultado esperado:**   * Presentar la información de las cuentas asociadas al usuario de OpenPublish. * Presentar los grupos, eventos, páginas de acuerdo a cada cuenta de Twitter. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 169.** Prueba de aceptación HU15.2

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU15.2 | **Nombre y número de la Historia de Usuario**. HU15 Permite publicar contenido de Alavista.tv en la red social Twitter |
| **Nombre:** Seleccionar por lo mínimo un identificador de la cuenta, seleccionar si se desea las tendencias del país, si no realiza esta acción se presentara un mensaje de notificación. | |
| **Responsable:** Edison Rodriguez | **Fecha:**10-07-2015 |
| **Descripción:** Verificar que se selección por lo mínimo un identificador para realizar la publicación. | |
| **Condiciones de ejecución:**   * Contener datos de cuentas en la base de datos. * Contener videos en la base de datos de OpenPublish * Tener las funciones del VideoController para listar los videos de manera dinámica. * Tener los métodos para obtener la información de las cuentas, método para realizar la publicación en el archivo TwitterHelper. * Tener las funciones en el controlador TwitterController y el ruteo correspondiente. * Tener la interfaz de publicaciones en Twitter. | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador o editor de la aplicación. * Seleccionar la opción de redes sociales. * Seleccionar la opción de publicar en Twitter. * Seleccionar un identificador de cuenta para realizar la publicación. * Si lo desea selecciona las tendencias del país. | |
| **Resultado esperado:**   * Presentar el mensaje de notificación que no se ha seleccionado ninguna cuenta para realizar la publicación. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 170.** Prueba de aceptación HU15.3

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU15.3 | **Nombre y número de la Historia de Usuario**. HU15 Permite publicar contenido de Alavista.tv en la red social Twitter |
| **Nombre:** Agregar un mensaje a la publicación, si no realiza esta acción se presentara un mensaje de notificación. | |
| **Responsable:** Edison Rodriguez | **Fecha:**10-07-2015 |
| **Descripción:** Verificar que se ha agregado un mensaje a la publicación. | |
| **Condiciones de ejecución:**   * Contener datos de cuentas en la base de datos. * Contener videos en la base de datos de OpenPublish * Tener las funciones del VideoController para listar los videos de manera dinámica. * Tener los métodos para obtener la información de las cuentas, método para realizar la publicación en el archivo TwitterHelper. * Tener las funciones en el controlador TwitterController y el ruteo correspondiente. * Tener la interfaz de publicaciones en Twitter. | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador o editor de la aplicación. * Seleccionar la opción de redes sociales. * Seleccionar la opción de publicar en Twitter. * Seleccionar un identificador de cuenta para realizar la publicación. * Agregar un mensaje a la publicación. * Agregar tendencias del país si se lo desea. | |
| **Resultado esperado:**   * Presentar mensaje de notificación que no se ha agregado ningún mensaje a la publicación que se realizara en Twitter. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 171.** Prueba de aceptación HU15.4

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU15.4 | **Nombre y número de la Historia de Usuario**. HU15 Permite publicar contenido de Alavista.tv en la red social Twitter |
| **Nombre:** El contenido para la publicación se envía a la API para que realice la publicación, y se emitirá un mensaje de notificación si la acción se realizó correctamente o si hubo algún error. | |
| **Responsable:** Edison Rodriguez | **Fecha:**10-07-2015 |
| **Descripción:** Verificar que la publicación se realizó correctamente en Twitter. | |
| **Condiciones de ejecución:**   * Contener datos de cuentas en la base de datos. * Contener videos en la base de datos de OpenPublish * Tener las funciones del VideoController para listar los videos de manera dinámica. * Tener los métodos para obtener la información de las cuentas, método para realizar la publicación en el archivo TwitterHelper. * Tener las funciones en el controlador TwitterController y el ruteo correspondiente. * Tener la interfaz de publicaciones en Twitter. | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador o editor de la aplicación. * Seleccionar la opción de redes sociales. * Seleccionar la opción de publicar en Twitter. * Seleccionar un identificador de cuenta para realizar la publicación. * Agregar un mensaje a la aplicación. * Realizar la publicación. | |
| **Resultado esperado:**   * Presentar mensaje de notificación donde se evidencie el identificador de la publicación realizada en Twitter. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 172.** HU 16 Permite publicar páginas Web en redes sociales.

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario:** Permite publicar páginas Web en redes sociales. | | |
| **Número:** HU16 | **Nombre de la historia:** Permite publicar páginas Web en redes sociales. | |
| **Modificación de historia de usuario:** Número y Nombre | | |
| **Usuario:** Administrador, Editor | | **Iteración Asignada:** 6 |
| **Prioridad en el Negocio:** Alta | | **Puntos Estimados:** 4 |
| **Riesgo en el Desarrollo:** Alta | | **Puntos Reales:**4 |
| **Descripción:** Como administrador, editor deseo realizar publicaciones de páginas de importancia para Alavista.tv en redes sociales, Facebook y Twitter ya sea con una o varias cuentas registradas. | | |
| **Observaciones:** En el caso de publicar en Facebook tendrá la posibilidad de publicar en el perfil de la cuenta, grupos, eventos, páginas.  En el caso de publicar en Twitter tendrá la posibilidad de realizar un Tweet añadiendo elementos de las tendencias en el País. | | |
| **Pruebas de Aceptación:**   * Hacer uso de la APIs de redes sociales para obtener la información de las cuentas para realizar la publicación, en el caso de error se presentara el mensaje correspondiente. * Seleccionar por lo mínimo un identificador de la cuenta ya sea un perfil, grupo, evento, pagina, si no realiza esta acción se presentara un mensaje de notificación. * Listar las tendencias que tiene el país en el momento actual de realizar la publicación. * Agregar un mensaje a la publicación, si no realiza esta acción se presentara un mensaje de notificación. * El contenido para la publicación se envía a la API para que realice la publicación, y se emitirá un mensaje de notificación si la acción se realizó correctamente o si hubo algún error. | | |
| **Realizado por:** Villa E., Rodriguez E., 2015 | | |

**Tabla 173.** Tarea de ingeniería 1 HU16

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario:** HU16. Permite publicar páginas Web en redes sociales. | |
| **Número tarea:** 1 | **Nombre de la tarea:** Crear las funciones en el controlador TwitterController y FacebookController y el ruteo correspondiente. |
| **Tipo de tarea:** Desarrollo | **Puntos estimados:** 2 |
| **Fecha de inicio:** 13-07-2015 | **Fecha fin:** 13-07-2015 |
| **Programador responsable:** Edison Rodriguez | |
| **Descripción:** Se creara las funciones en el controlador para obtener la información de la cuenta, como son nombre, apellido, edad, correo, tendencias. Así como la función para realizar la publicación del video. | |
| **Pruebas de aceptación:**   * Cumplir con el estándar de conFiguración. * Verificar la información obtenida es la correcta. * Emitir mensajes correspondientes a la respuesta de la API * Enviar mensajes específicos para errores comunes que se presentan al momento de consumir recursos de la API. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 174.** Tarea de ingeniería 2 HU16

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario:** HU16. Permite publicar páginas Web en redes sociales. | |
| **Número tarea:** 2 | **Nombre de la tarea:** Crear la interfaz de publicaciones en redes sociales. |
| **Tipo de tarea:** Desarrollo | **Puntos estimados:** 2 |
| **Fecha de inicio:** 14-07-2015 | **Fecha fin:** 14-07-2015 |
| **Programador responsable:** Edison Rodriguez | |
| **Descripción:** Crear la vista para realizar la publicación de los videos en redes sociales | |
| **Pruebas de aceptación:**   * Verificar la información de cada cuenta se cargue correctamente. * Emitir mensajes de error en el caso que exista al momento de cargar las cuentas de las redes sociales. * Verificar que se seleccione por lo menos una de las opciones para realizar la publicación. * Obtener mensajes de confirmación de la publicación realizada. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 175.** Actividades HU16

|  |  |  |
| --- | --- | --- |
| **FECHA** | **ACTIVIDAD** | **TIEMPO** |
| Inicio:13-07-2015  Fin:13-07-2015  Inicio:14-07-2015  Fin:14-07-2015 | **Tarea: Crear las funciones en el controlador TwitterController y FacebookController y el ruteo correspondiente.**   * Crear la función para listar las cuentas de las redes sociales asociadas al usuario de OpenPublish. * Crear la función para publicar la página en redes sociales. * Agregar el ruteo para el controlador en el archivo route.   **Tarea: Crear la interfaz de publicaciones en Twitter.**   * Crear las funciones AJAX para la publicación de las páginas en redes sociales. * Realizar la interfaz para la publicación de las páginas en redes sociales. * Presentar mensajes del resultado de la publicación. | 4h:00  3h:00  1h:00  4h:00  3h:00  1h:00 |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 176.** Prueba de aceptación HU16.1

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU16.1 | **Nombre y número de la Historia de Usuario**. HU 16 Permite publicar páginas Web en redes sociales. |
| **Nombre:** Hacer uso de la APIs de redes sociales para obtener la información de las cuentas para realizar la publicación, en el caso de error se presentara el mensaje correspondiente. | |
| **Responsable:** Edison Rodriguez | **Fecha:**14-07-2015 |
| **Descripción:** Verificar el uso correcto de la API de redes sociales para obtener los datos de las cuentas. | |
| **Condiciones de ejecución:**   * Contener datos de las cuentas de redes sociales en la base de datos. * Contener videos en la base de datos de OpenPublish * Tener las funciones del VideoController para listar los videos de manera dinámica. * Tener los métodos para obtener la información de las cuentas, método para realizar la publicación. * Tener las funciones en el controlador y el ruteo correspondiente. * Tener la interfaz de publicaciones de Facebook y Twitter. | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador o editor de la aplicación. * Seleccionar la opción de redes sociales. * Seleccionar la opción de publicar en Facebook o Twitter. | |
| **Resultado esperado:**   * Presentar la información de las cuentas asociadas al usuario de OpenPublish. * Presentar los grupos, eventos, páginas de acuerdo a cada cuenta de Facebook. * Presentar las tendencias de Twitter en el momento de la publicación. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 177.** Prueba de aceptación HU16.2

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU16.2 | **Nombre y número de la Historia de Usuario**. HU 16 Permite publicar páginas Web en redes sociales. |
| **Nombre:** Seleccionar por lo mínimo un identificador de la cuenta ya sea un perfil, grupo, evento, pagina, si no realiza esta acción se presentara un mensaje de notificación. | |
| **Responsable:** Edison Rodriguez | **Fecha:**14-07-2015 |
| **Descripción:** Verificar que se selección por lo mínimo un identificador para realizar la publicación. | |
| **Condiciones de ejecución:**   * Contener datos de cuentas en la base de datos. * Contener videos en la base de datos de OpenPublish * Tener las funciones del VideoController para listar los videos de manera dinámica. * Tener los métodos para obtener la información de las cuentas, método para realizar la publicación en el archivo Facebook y Twitter. * Tener las funciones en el controlador TwitterController y FacebookController y el ruteo correspondiente. * Tener la interfaz de publicaciones en Twitter y Facebook. | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador o editor de la aplicación. * Seleccionar la opción de redes sociales. * Seleccionar la opción de publicar en Twitter o en Facebook. * Seleccionar un identificador de cuenta para realizar la publicación. | |
| **Resultado esperado:**   * Presentar mensaje de notificación que no se ha seleccionado ninguna cuenta para realizar la publicación. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 178.** Prueba de aceptación HU16.3

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU16.3 | **Nombre y número de la Historia de Usuario**. HU 16 Permite publicar páginas Web en redes sociales. |
| **Nombre:** Listar las tendencias que tiene el país en el momento actual de realizar la publicación. | |
| **Responsable:** Edison Rodriguez | **Fecha:**14-07-2015 |
| **Descripción:** Verificar que se listen las tendencias actuales en el país. | |
| **Condiciones de ejecución:**   * Contener datos de cuentas en la base de datos. * Contener videos en la base de datos de OpenPublish * Tener las funciones del VideoController para listar los videos de manera dinámica. * Tener los métodos para obtener la información de las cuentas, método para realizar la publicación en el archivo Facebook y Twitter. * Tener las funciones en el controlador TwitterController y FacebookController y el ruteo correspondiente. * Tener la interfaz de publicaciones en Twitter y Facebook | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador o editor de la aplicación. * Seleccionar la opción de redes sociales. * Seleccionar la opción de publicar en Twitter o en Facebook. * Seleccionar un identificador de cuenta para realizar la publicación. | |
| **Resultado esperado:**   * Presentar las tendencias actuales del país de forma correcta, para que pueda seleccionar para agregarlos en la publicación. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 179.** Prueba de aceptación HU16.4

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU16.4 | **Nombre y número de la Historia de Usuario**. HU 16 Permite publicar páginas Web en redes sociales. |
| **Nombre:** Agregar un mensaje a la publicación, si no realiza esta acción se presentara un mensaje de notificación. | |
| **Responsable:** Edison Rodriguez | **Fecha:**14-07-2015 |
| **Descripción:** Verificar que se ha agregado un mensaje a la publicación. | |
| **Condiciones de ejecución:**   * Contener datos de cuentas en la base de datos. * Contener videos en la base de datos de OpenPublish * Tener las funciones del VideoController para listar los videos de manera dinámica. * Tener los métodos para obtener la información de las cuentas, método para realizar la publicación en el archivo Facebook y Twitter. * Tener las funciones en el controlador TwitterController y FacebookController y el ruteo correspondiente. * Tener la interfaz de publicaciones en Twitter y Facebook. | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador o editor de la aplicación. * Seleccionar la opción de redes sociales. * Seleccionar la opción de publicar en Twitter o en Facebook. * Seleccionar un identificador de cuenta para realizar la publicación. * Agregar el link de la página a publicar. | |
| **Resultado esperado:**   * Presentar mensaje de notificación que no se ha agregado ningún mensaje a la publicación que se realizara. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 180.** Prueba de aceptación HU16.5

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU16.5 | **Nombre y número de la Historia de Usuario**. HU 16 Permite publicar páginas Web en redes sociales. |
| **Nombre:** El contenido para la publicación se envía a la API para que realice la publicación, y se emitirá un mensaje de notificación si la acción se realizó correctamente o si hubo algún error. | |
| **Responsable:** Edison Rodriguez | **Fecha:**14-07-2015 |
| **Descripción:** Verificar que la publicación se realizó correctamente en Twitter. | |
| **Condiciones de ejecución:**   * Contener datos de cuentas en la base de datos. * Contener videos en la base de datos de OpenPublish * Tener las funciones del VideoController para listar los videos de manera dinámica. * Tener los métodos para obtener la información de las cuentas, método para realizar la publicación en el archivo Facebook y Twitter. * Tener las funciones en el controlador TwitterController y FacebookController y el ruteo correspondiente. * Tener la interfaz de publicaciones en Twitter y Facebook. | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador o editor de la aplicación. * Seleccionar la opción de redes sociales. * Seleccionar la opción de publicar en Twitter o en Facebook. * Seleccionar un identificador de cuenta para realizar la publicación. * Agregar el link de la página a publicar. * Realizar la publicación. | |
| **Resultado esperado:**   * Presentar mensaje de notificación donde se evidencie el identificador de la publicación realizada. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 181.** HU17 Permite presentar datos de reproducción en forma de galería multimedia de los contenidos de Alavista.tv.

|  |  |  |
| --- | --- | --- |
| **Historia de Usuario:** Permite presentar datos de reproducción en forma de galería multimedia de los contenidos de Alavista.tv. | | |
| **Número:** HU17 | **Nombre de la historia:** Presentar datos de reproducción en forma de galería multimedia de los contenidos de Alavista.tv. | |
| **Modificación de historia de usuario:** Número y Nombre | | |
| **Usuario:** Administrador, Editor, Invitado | | **Iteración Asignada:** 6 |
| **Prioridad en el Negocio:** Alta | | **Puntos Estimados:** 4 |
| **Riesgo en el Desarrollo:** Alta | | **Puntos Reales:** 3 |
| **Descripción:** Como administrador, editor, invitado deseo poder visualizar y reproducir los videos que han sido guardados con OpenPublish | | |
| **Observaciones:** Los videos pueden ordenarse de acuerdo al id, titulo, calificación, fecha  Se pueden visualizar los videos mediante una galería multimedia | | |
| **Pruebas de Aceptación:**   * Listar correctamente los videos mediante los filtros de búsqueda * Mostrar una ventana modal para reproducir un video, la ventana modal debe contener controles de reproducción. | | |
| **Realizado por:** Villa E., Rodriguez E., 2015 | | |

**Tabla 182.** Tarea de ingeniería 1 HU17

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario:** HU17. Permite presentar datos de reproducción en forma de galería multimedia de los contenidos de Alavista.tv. | |
| **Número tarea:** 1 | **Nombre de la tarea:** Crear el controlador Gallery y ruteo del proceso a realizar. |
| **Tipo de tarea:** Desarrollo | **Puntos estimados:** 2 |
| **Fecha de inicio**: 15-07-2015 | **Fecha fin:** 15-07-2015 |
| **Programador responsable:** Edwin Villa | |
| **Descripción**: Agregar la dirección de la operación en el archivo de ruteo, y realizar las funciones necesarias para reproducir videos de Alavista.tv. | |
| **Pruebas de aceptación:**   * Cumplir con el estándar de codificación. * Control de del tipo de ruteo correcto**.** | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 183.** Tarea de ingeniería 2 HU17

|  |  |
| --- | --- |
| **TAREA DE INGENIERIA** | |
| **Historia de Usuario:** HU17. Permite presentar datos de reproducción en forma de galería multimedia de los contenidos de Alavista.tv. | |
| **Número tarea:** 2 | **Nombre de la tarea:** Crear interfaz para la vista que permite reproducir videos de Alavista.tv. |
| **Tipo de tarea:** Desarrollo | **Puntos estimados:** 2 |
| **Fecha de inicio:** 16-07-2015 | **Fecha fin:** 16-07-2015 |
| **Programador responsable:** Edwin Villa | |
| **Descripción:** Se creara un reproductor que permita visualizar los videos de Alavista.tv. | |
| **Pruebas de aceptación:**   * Cumplir con el estándar de interfaz. * Emitir un mensaje específico dependiendo el resultado de la solicitud. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 184.** Actividades HU17

|  |  |  |
| --- | --- | --- |
| **FECHA** | **ACTIVIDAD** | **TIEMPO** |
| Inicio:15-07-2015  Fin:15-07-2015  Inicio:16-07-2015  Fin:16-07-2015 | **Tarea: Crear el controlador Gallery y ruteo del proceso a realizar.**   * Crear y conFigurar el archivo GalleryController * Crear la función para listar los videos ingresados por el usuario seleccionado en la conFiguración de credenciales * Crear método de paginación de búsqueda de videos * Agregar el ruteo para el controlador en el archivo route.   **Tarea: Crear interfaz para la vista que permite reproducir videos de Alavista.tv.**   * Crear las funciones AJAX para reproducir el video seleccionado | 1h:00  2h:00  2h:00  1h:00  4h:00 |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 185.** Prueba de aceptación HU17.1

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU17.1 | **Nombre y número de la Historia de Usuario**. HU17 Permite presentar datos de reproducción en forma de galería multimedia de los contenidos de Alavista.tv. |
| **Nombre:** Listar correctamente los videos mediante los filtros de búsqueda | |
| **Responsable:** Edison Rodriguez | **Fecha:**16-07-2015 |
| **Descripción:** Verificar que se listen los videos en la galería de manera correcta. | |
| **Condiciones de ejecución:**   * Contener videos en la base de datos de OpenPublish * Crear el controlador de galería y ruteo del proceso a realizar. * Crear interfaz para la vista que permite reproducir videos de Alavista.tv. | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador o editor de la aplicación. * Seleccionar la opción de galería. | |
| **Resultado esperado:**   * Obtener el listado de videos en formato de galería, para observar información relevante, y con la opción de reproducción. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 186.** Prueba de aceptación HU17.2

|  |  |
| --- | --- |
| **PRUEBA DE ACEPTACIÓN** | |
| **Código:** HU17.2 | **Nombre y número de la Historia de Usuario**. HU17 Permite presentar datos de reproducción en forma de galería multimedia de los contenidos de Alavista.tv. |
| **Nombre:** Mostrar una ventana modal para reproducir un video, la ventana modal debe contener controles de reproducción. | |
| **Responsable:** Edison Rodriguez | **Fecha:**16-07-2015 |
| **Descripción:** Observar la ventana modal con la opción de reproducción. | |
| **Condiciones de ejecución:**   * Contener videos en la base de datos de OpenPublish * Crear el controlador de galería y ruteo del proceso a realizar. * Crear interfaz para la vista que permite reproducir videos de Alavista.tv. | |
| **Pasos de ejecución:**   * Iniciar sesión como administrador o editor de la aplicación. * Seleccionar la opción de galería. * Seleccionar un video de la galería. | |
| **Resultado esperado:**   * Reproducción del video seleccionado conjuntamente con su información. | |
| **Evaluación de la prueba:**  Satisfactoria. | |

**Realizado por:** Villa E., Rodriguez E., 2015

# ANEXO F. Detalle de Tarjtas CRC.

**Tabla 187.** Modelo Cuenta de Facebook.

|  |  |
| --- | --- |
| **Class Account** | |
| Parámetros de conFiguración.   * Conexión. * Tabla * Clave primaria. * Atributos de la tabla. | ORM Eloquent |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 188.** Modelo Cuenta de Twitter.

|  |  |
| --- | --- |
| **Class Accounttw** | |
| Parámetros de conFiguración.   * Conexión. * Tabla * Clave primaria. * Atributos de la tabla. | ORM Eloquent |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 189.** Modelo Categoría.

|  |  |
| --- | --- |
| **Class Accounttw** | |
| Parámetros de conFiguración.   * Conexión. * Tabla * Clave primaria. * Atributos de la tabla. | ORM Eloquent |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 190.** Modelo ConFiguración de la aplicación.

|  |  |
| --- | --- |
| **Class ConfigApp** | |
| Parámetros de conFiguración.   * Conexión. * Tabla * Clave primaria. * Atributos de la tabla. | ORM Eloquent |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 191.** Modelo Menú.

|  |  |
| --- | --- |
| **Class Menu** | |
| Parámetros de conFiguración.   * Conexión. * Tabla * Clave primaria. * Atributos de la tabla.   submenús()  users() | ORM Eloquent |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 192.** Modelo Rol

|  |  |
| --- | --- |
| **Class Rol** | |
| Parámetros de conFiguración.   * Conexión. * Tabla * Clave primaria. * Atributos de la tabla.   submenús()  users()  menuspivot() | ORM Eloquent |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 193.** Modelo Estado

|  |  |
| --- | --- |
| **Class Status** | |
| Parámetros de conFiguración.   * Conexión. * Tabla * Clave primaria. * Atributos de la tabla. | ORM Eloquent |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 194.** Modelo Submenú

|  |  |
| --- | --- |
| **Class Submenu** | |
| Parámetros de conFiguración.   * Conexión. * Tabla * Clave primaria. * Atributos de la tabla. | ORM Eloquent |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 195.** Modelo Usuario

|  |  |
| --- | --- |
| **Class User** | |
| Parámetros de conFiguración.   * Conexión. * Tabla * Clave primaria. * Atributos de la tabla.   submenús()  users()  cartasnopivot()  cartas() | ORM Eloquent |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 196.** Modelo Usuario Joomla

|  |  |
| --- | --- |
| **Class UserJoomla** | |
| Parámetros de conFiguración.   * Conexión. * Tabla * Clave primaria. | ORM Eloquent |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 197.** Modelo Usuario grupo de joomla

|  |  |
| --- | --- |
| **Class UserUserGroupJoomla** | |
| Parámetros de conFiguración.   * Conexión. * Tabla * Clave primaria. | ORM Eloquent |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 198.** Modelo Videos

|  |  |
| --- | --- |
| **Class Video** | |
| Parámetros de conFiguración.   * Conexión. * Tabla * Clave primaria. | ORM Eloquent |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 199.** Modelo Videos categoría

|  |  |
| --- | --- |
| **Class VideoCategory** | |
| Parámetros de conFiguración.   * Conexión. * Tabla * Clave primaria. | ORM Eloquent |

**Realizado por:** Villa E., Rodriguez E., 2015

**Tabla 200.** Modelo Video OpenPublish

|  |  |
| --- | --- |
| **Class VideoOpenpublish** | |
| Parámetros de conFiguración.   * Conexión. * Tabla * Clave primaria. | ORM Eloquent |

**Realizado por:** Villa E., Rodriguez E., 2015