

**ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO**

**FACULTAD DE INFORMÁTICA Y ELECTRÓNICA**

**ESCUELA DE INGENIERÍA EN SISTEMAS**

**Estudio comparativo de productividad de frameworks PHP orientados a objetos para desarrollar el Sistema de Seguimiento de Incidentes de la Infraestructura de Red en la ESPOCH**

Tesis de grado previo a la obtención del Título de:

**INGENIERÍA EN SISTEMAS INFORMÁTICOS**

**AUTOR:** SAMANIEGO LARREA MARÍA JOSÉ

**TUTOR:** ING. JORGE MENÉNDEZ

Riobamba – Ecuador

2015

**ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO**

**FACULTAD DE INFORMÁTICA Y ELECTRÓNICA**

**ESCUELA DE INGENIERÍA EN SISTEMAS**

El Tribunal de Tesis certifica que: El trabajo de investigación: ESTUDIO COMPARATIVO DE PRODUCTIVIDAD DE FRAMEWORKS PHP ORIENTADOS A OBJETOS PARA DESARROLLAR EL SISTEMA DE SEGUIMIENTO DE INCIDENTES DE LA INFRAESTRUCTURA DE RED EN LA ESPOCH*,* de responsabilidad de la señorita María José Samaniego Larrea, ha sido minuciosamente revisado por los Miembros del Tribunal de Tesis, quedando autorizada su presentación.

Ing. Gonzalo Samaniego, Ph.D

**DECANO DE LA FACULTAD DE**

**INFORMÁTICA Y ELECTRÓNICA**  \_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_

Dr. Julio Santillán

**DIRECTOR DE LA ESCUELA DE**

**INGENIERÍA EN SISTEMAS** \_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_

Ing. Jorge Menéndez

**DIRECTOR DE TESIS** \_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_

Ing. Danilo Pástor

**MIEMBRO DEL TRIBUNAL** \_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_

**DOCUMENTALISTA**

**SISBIB ESPOCH**  \_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_

Yo, María José Samaniego Larrea soy responsable de las ideas, doctrinas y resultados expuestos en esta tesis, y el patrimonio intelectual de la Tesis de Grado pertenece a la Escuela Superior Politécnica de Chimborazo

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

MARÍA JOSÉ SAMANIEGO LARREA

DEDICATORIA

A mis padres, por su amor, trabajo y sacrificio en todos estos años para poder llegar hasta aquí, además de siempre darme las fuerzas y palabras de aliento necesarias para seguir adelante ante toda circunstancia.

AGRADECIMIENTO

Al culminar una nueva etapa de mi vida quiero agradecer a mis padres por el apoyo incondicional que me han brindado en mi formación académica y creer siempre en mí; a mi familia y amigos por brindarme esas palabras de aliento para seguir adelante sin desistir. A mi tutores de tesis, Ing. Jorge Menéndez e Ing. Danilo Pastor por su valioso tiempo y conocimiento compartido para la realización de esta investigación para culminar la carrera.

María José

**CONTENIDO**

**CONTENIDO** Páginas

[DEDICATORIA iv](#_Toc425344852)

[AGRADECIMIENTO v](#_Toc425344853)

[RESUMEN xiii](#_Toc425344854)

[SUMMARY xiv](#_Toc425344855)

[INTRODUCCIÓN 1](#_Toc425344856)

[CAPÍTULO I](#_Toc425344857)

[MARCO TEÓRICO](#_Toc425344858)

[1.1 **Aplicaciones web** 8](#_Toc425344859)

[1.1.1 ***Historia de las Aplicaciones Web***. 8](#_Toc425344860)

[1.1.2 ***Características de las Aplicaciones Web.*** 9](#_Toc425344861)

[1.2 **Lenguajes de programación**. 10](#_Toc425344862)

[1.2.1 ***Lenguajes de programación para desarrollo web.*** 10](#_Toc425344863)

[1.3  **Lenguaje PHP** 11](#_Toc425344864)

[1.3.1 ***Historia.*** 11](#_Toc425344865)

[1.3.2 ***Características***. 12](#_Toc425344866)

[1.3.3 ***Sintaxis.*** 13](#_Toc425344867)

[1.4 **Framework web** 15](#_Toc425344868)

[1.4.1 ***Definición de Framework Web.*** 15](#_Toc425344869)

[1.4.2 ***Objetivos de los Frameworks Web.*** 16](#_Toc425344870)

[1.4.3 ***Características de los Frameworks Web.*** 16](#_Toc425344871)

[1.4.4 ***Ventajas de usar Frameworks Web***. 17](#_Toc425344872)

[1.4.5 ***Desventajas de usar Frameworks Web.*** 17](#_Toc425344873)

[1.5 **Frameworks web para PHP** 17](#_Toc425344874)

[1.6 **CakePHP** 19](#_Toc425344875)

[1.6.1 ***Definición***. 19](#_Toc425344876)

[1.6.2 ***Historia***. 19](#_Toc425344877)

[1.6.3 ***Filosofía.*** 20](#_Toc425344878)

[1.6.4 ***Características***. 20](#_Toc425344879)

[1.6.5 ***Arquitectura MVC en CakePHP***. 21](#_Toc425344880)

[1.6.6 ***Convenciones en CakePHP***. 22](#_Toc425344881)

[1.6.7 ***Requerimientos***. 24](#_Toc425344882)

[1.7 **Laravel** 25](#_Toc425344883)

[1.7.1 ***Definición.*** 25](#_Toc425344884)

[1.7.2 ***Historia***. 25](#_Toc425344885)

[1.7.3 ***Características***. 26](#_Toc425344886)

[1.7.4 ***Filosofía.*** 26](#_Toc425344887)

[1.7.5 ***Arquitectura***. 27](#_Toc425344888)

[1.7.6 ***Eloquent ORM***. 28](#_Toc425344889)

[1.7.7 ***Requerimientos***. 29](#_Toc425344890)

[1.8 **Métricas de software** 30](#_Toc425344891)

[1.8.1 ***Definición***. 30](#_Toc425344892)

[1.8.2 ***Importancia de medir Software***. 31](#_Toc425344893)

[1.8.3 ***Características***. 31](#_Toc425344894)

[1.8.4 ***Métricas del Proceso y del Proyecto***. 32](#_Toc425344895)

[1.8.5 ***Clasificación de las Medidas de Software*** 33](#_Toc425344896)

[1.9 **Productividad** 33](#_Toc425344897)

[1.9.1 ***Definición***. 33](#_Toc425344898)

[1.9.2 ***Factores que afectan a la Productividad.*** 34](#_Toc425344899)

[1.10 **Métricas de productividad** 35](#_Toc425344900)

[1.10.1 ***Métricas Orientadas al Tamaño.*** 35](#_Toc425344901)

[1.10.2 ***Métricas Orientadas a la Función***. 37](#_Toc425344902)

[1.10.3 ***Métricas Orientadas a la Documentación***. 39](#_Toc425344903)

[1.11 **Líneas de código vs. Puntos de función vs. Documentación** 40](#_Toc425344904)

[1.12 **Desarrollo ágil de aplicaciones web** 41](#_Toc425344905)

[1.12.1 ***Ventajas de las Metodologías Ágiles*** 41](#_Toc425344906)

[1.12.2 ***Principales Metodologías Ágiles***. 42](#_Toc425344907)

[1.12.3 ***Metodología SCRUM.*** 43](#_Toc425344908)

[CAPITULO II](#_Toc425344909)

[MARCO METODOLÓGICO](#_Toc425344910)

[2.1 **Introducción** 46](#_Toc425344911)

[2.2 **Tipo de investigación** 46](#_Toc425344912)

[2.3 **Definición del parámetro de productividad para la comparación** 46](#_Toc425344913)

[2.4 **Definición del índice de productividad física** 48](#_Toc425344914)

[2.4.1 ***Indicador 1: Líneas de código (LDC)***. 49](#_Toc425344915)

[2.4.2 ***Indicador 2: Tiempo Empleado (TE)***. 49](#_Toc425344916)

[2.5 **Diseño del experimento** 49](#_Toc425344917)

[2.5.1 ***Método de muestreo***. 50](#_Toc425344918)

[2.5.2 ***Tipo de muestreo***. 50](#_Toc425344919)

[2.5.3 ***Determinación del escenario de prueba***. 50](#_Toc425344920)

[2.5.4 ***Herramientas para el estudio comparativo*** 52](#_Toc425344921)

[2.5.5 ***Elección de la Prueba Estadística***. 52](#_Toc425344922)

[CAPITULO III](#_Toc425344923)

[RESULTADOS, DISCUSIÓN Y ANÁLISIS DE RESULTADOS](#_Toc425344924)

[3.1 **Estudio comparativo** 54](#_Toc425344925)

[3.1.1 ***Índice de Productividad Física (IPF)***. 54](#_Toc425344926)

[3.2 **Resultado del parámetro de productividad** 65](#_Toc425344927)

[3.2.1 ***Evaluación de resultados del indicador líneas de código (LDC)***. 65](#_Toc425344928)

[3.2.2 ***Evaluación de resultados del indicador tiempo empleado (TE***). 66](#_Toc425344929)

[3.2.3 ***Evaluación de resultados del índice de productividad física (IPF***). 67](#_Toc425344930)

[CAPITULO IV](#_Toc425344931)

[SISTEMA DE SEGUIMIENTO DE INCIDENTES DE LA INFRAESTRUCTURA DE RED EN LA ESPOCH](#_Toc425344932)

[4.1 **Introducción** 70](#_Toc425344933)

[4.2 **Información general** 71](#_Toc425344934)

[4.3 **Definición del ámbito** 72](#_Toc425344935)

[4.4 **Estudio de factibilidad** 73](#_Toc425344936)

[4.5 **Características de usuario** 75](#_Toc425344937)

[4.6 **Definición del Product Backlog** 75](#_Toc425344938)

[4.7 **Planificación Scrum** 77](#_Toc425344939)

[4.7.1 ***Especificación de actividades del Product Backlog*** 78](#_Toc425344940)

[4.8 **Sprints** 80](#_Toc425344941)

[4.9  **Sistema de seguimiento de incidentes con Burn up Chart** 95](#_Toc425344942)

[CONCLUSIONES 97](#_Toc425344943)

[RECOMENDACIONES 98](#_Toc425344944)

**GLOSARIO**

**BIBLIOGRAFÍA**

**ANEXOS**

**ÍNDICE DE ILUSTRACIONES**

[Figura. 1-1. Etiquetas de PHP 13](#_Toc425188947)

[Figura. 2-1. Frameworks Web 16](#_Toc425188948)

[Figura. 3-1. Distribución de Frameworks en Sitios Top. 18](#_Toc425188949)

[Figura. 4- 1. Logo de CakePHP 19](#_Toc425188950)

[Figura. 5-1. Petición típica MVC en CakePHP 22](#_Toc425188951)

[Figura. 6-1. Logo de Laravel 25](#_Toc425188952)

[Figura. 7-1. Petición típica MVC en Laravel 28](#_Toc425188953)

[Figura. 8-1. Script CLOC 36](#_Toc425188954)

[Figura. 9-1. Understand Tool 37](#_Toc425188955)

[Figura. 10-1. Cálculo de cuenta total de Puntos de Función 38](#_Toc425188956)

[Figura. 11-1. Controlador Embebido Con Sistema de Sonar 40](#_Toc425188957)

[Figura. 12-1. Proceso de la metodología Ágil Scrum 45](#_Toc425188958)

[Figura. 1-2 Modelo Simple de Productividad 47](#_Toc425188959)

[Figura. 1-3. Promedios líneas de código para CakePHP y Laravel 66](#_Toc425188960)

[Figura. 2-3. Promedios tiempo empleado (horas) para CakePHP y Laravel 67](#_Toc425188961)

[Figura. 3-3. Promedios indice de productividad fisica para CakePHP y Laravel 68](#_Toc425188962)

[Figura. 1-4: Organigrama estructura DTIC 72](#_Toc425188963)

[Figura. 2-4. Planificación previa de desarrollo 78](#_Toc425188964)

[Figura. 3-4. Diagrama Entidad Relación SDSI-DTIC 82](#_Toc425188965)

[Figura. 4-4. Arquitectura General del SDSI-DTIC 84](#_Toc425188966)

[Figura. 5-4. Diseño de interfaz del sistema SDSI-DTIC 88](#_Toc425188967)

[Figura. 6-4. Interfaz de Autentificación 88](#_Toc425188968)

[Figura. 7-4. Interfaz de Ingreso de Datos 89](#_Toc425188969)

[Figura. 8-4. Interfaz de Seguimiento de Incidentes 90](#_Toc425188970)

[Figura. 9-4. Interfaz de Reportes 91](#_Toc425188971)

[Figura. 10-4. Burn-Up Chart SDSI-DTIC 96](#_Toc425188972)

**ÍNDICE DE TABLAS**

[Tabla 1-1: Tipos de datos primitivos en PHP 14](#_Toc426542881)

[Tabla 2-1: Tipos de datos especiales en PHP 15](#_Toc426542882)

[Tabla 3-1: Tipos de datos compuestos en PHP 15](#_Toc426542883)

[Tabla 1-2: Parámetros de comparación a valorar 47](#_Toc426542884)

[Tabla 2-2: Índice de Productividad física 48](#_Toc426542885)

[Tabla 3-2: Indicador líneas de código 49](#_Toc426542886)

[Tabla 4-2: Índice de tiempo empleado 49](#_Toc426542887)

[Tabla 5-2: Pruebas de normalidad 53](#_Toc426542888)

[Tabla 3-1: Resultados obtenidos líneas de código CakePHP 55](#_Toc426542889)

[Tabla 3-2: Resultados obtenidos líneas de código Laravel 55](#_Toc426542890)

[Tabla 3-3: Estadística descriptiva líneas de código 56](#_Toc426542891)

[Tabla 4-3: Estadísticos de contrastes Líneas de código 58](#_Toc426542892)

[Tabla 5-3: Resultados indicador tiempo empleado CakePHP 58](#_Toc426542893)

[Tabla 6-3: Resultados indicador tiempo empleado Laravel 59](#_Toc426542894)

[Tabla 7-3: Estadística descriptiva tiempo empleado 59](#_Toc426542895)

[Tabla 8-3: Estadísticos de contrastes Tiempo empleado 61](#_Toc426542896)

[Tabla 9-3: Resultados Índice Productividad Física CakePHP 62](#_Toc426542897)

[Tabla 10-3: Resultados Índice Productividad Física Laravel 62](#_Toc426542898)

[Tabla 11-3: Estadística descriptiva índice de productividad física 63](#_Toc426542899)

[Tabla 12-3: Estadísticos de contrastes Índice de Productividad Física 65](#_Toc426542900)

[Tabla 1-4: Información de contacto 71](#_Toc426542901)

[Tabla 2-4: Product Backlog proyecto SDSI-DTIC 76](#_Toc426542902)

[Tabla 3-4: Planificación de historias de usuario y técnicas 78](#_Toc426542903)

[Tabla 4-4: Historia técnica diseño de la base de datos 80](#_Toc426542904)

[Tabla 5-4: Tarea de ingeniería de historia técnica 1 81](#_Toc426542905)

[Tabla 6-4: Prueba de aceptación historia técnica 1 81](#_Toc426542906)

[Tabla 7-4: Historia técnica diseño de la arquitectura del sistema 83](#_Toc426542907)

[Tabla 8-4: Tarea de ingeniería de historia técnica 2 83](#_Toc426542908)

[Tabla 9-4: Prueba de aceptación historia técnica 2 84](#_Toc426542909)

[Tabla 10-4: Historia técnica diseño de las interfaces de usuario SDSI-DTIC 85](#_Toc426542910)

[Tabla 11-4: Prueba de aceptación historia técnica 3 86](#_Toc426542911)

[Tabla 12-4: Historia técnica codificación del SDSI-DTIC 92](#_Toc426542912)

[Tabla 13-4: Prueba de aceptación historia técnica 4 93](#_Toc426542913)

[Tabla 14-4: Historia de usuario autentificación 94](#_Toc426542914)

[Tabla 15-4: Tarea de ingeniería de autentificación 94](#_Toc426542915)

[Tabla 16-4: Prueba de aceptación autentificación 95](#_Toc426542916)

**ÍNDICE DE ABREVIATURAS**

**ACL:**  (Access Control Lists), Listas de Control de Acceso.

**BSD:** (Berkeley Software Distribution), Distribución de Software de Berkeley.

**CGI:** (Common Gateway Interface), Interfaz de entrada común.

**CLOC:** (Count Lines of Code), Conteo de líneas de Código.

**CRUD:** (Create, Read, Update and Delete), Crear, Leer, Modificar, y Eliminar.

**DHTML:** (Dynamic HTML), Dinámico Lenguaje de Marcado de Hipertexto.

**DTIC:** Dirección de Tecnologías de la Información y Comunicación.

**GNU:** (GNU's Not UNIX), GNU no es UNIX.

**HE:** Tiempo empleado.

**HTML:** (HyperText Markup Language), Lenguaje de Marcado de Hipertexto.

**HTTP:** (Hypertext Transfer Protocol), Protocolo de Transferencia de Hipertexto.

**IEC:** (International Electrotechnical Commission), Comisión Electrotécnica Internacional.

**IEEE:** (Institute of Electrical and Electronics Engineers), Instituto de Ingenieros Eléctricos y Electrónicos.

**ISO:** (International Organization for Standardization), Organización Internacional de Estandarización.

**LDC:** Líneas de Código.

**MIT:** (Massachusetts Institute of Technology), Instituto de Tecnología de Massachusetts.

**MVC**: Modelo Vista Controlador.

**ORM:** (Object-Relational Mapping), Mapeo Objeto-Relacional.

**OSL:** (Open Software License), Licencia de Software Abierto.

**PHP:** (Hypertext Preprocessor), Procesador de Hipertexto

**SDSI-DTIC**: Sistema de Seguimiento de Incidentes de la Dirección de Tecnologías de la Información y Comunicación.

**SQL:** (Structured Query Language), Lenguaje de Consultas Estructuradas.

**TE:** Tiempo empleado.

**URLs:** (Uniform Resource Locator), Localizador Uniforme de Recursos.

**XML:** (Extensible Markup Language), Lenguaje de Marcas

RESUMEN

Se desarrolló el Sistema de Seguimiento de Incidentes para la infraestructura de red (SDSI-DTIC) de la Escuela Superior Politécnica de Chimborazo, con el objetivo de crear una aplicación web que permita gestionar y dar un seguimiento a las incidencias que se reportan diariamente en la Dirección de Tecnologías de la Información y Comunicaciones de la institución con el fin de reducir los tiempos de respuesta en la atención al cliente, se planteó la hipótesis de investigación en base a la problemática, para a continuación dar paso al diseño del experimento con la creación del prototipo de prueba creado con los dos frameworks PHP. Se procedió a recolectar la información con la ayuda del software Understand Tools y los tiempos de creación y modificación de los archivos, para así calcular el índice de productividad física por medio de la medición de las líneas de código ejecutable(LDC) y el tiempo empleado(TE) para producirlas(horas). La comparativa de los frameworks permitió conocer que CakePHP cuenta con un promedio de líneas de código por tiempo empleado de 92.903 LDC/TE y Laravel con 39.102 LDC/TE. Se concluye que el framework CakePHP permite aumentar la productividad en el desarrollo de aplicaciones web con un mayor nivel que Laravel, el Sistema de Seguimiento de Incidentes para la infraestructura de red se desarrolló con el framework ganador. Se recomienda a la Dirección de Tecnologías de la Información y Comunicación crear la funcionalidad de notificaciones vía correo electrónico para acceder a la información desde fuera de la ESPOCH.

**Palabras Claves: <**PRODUCTIVIDAD DE SOFTWARE><FRAMEWORK CAKEPHP><FRAMEWORK LARAVEL><SEGUIMIENTO DE INCIDENCIAS><LÍNEAS DE CÓDIGO><METODOLOGÍA ÁGIL SCRUM>

SUMMARY

An Incident monitoring system for network infrastructure (SDSI-DTIC) was developed in the Escuela Superior Politécnica de Chimborazo (Higher education). In order to create a web application allowing managing and monitoring to the incidents which is reported daily at the Information and Communication Technology Department of the education institution for shorter response times in the customer service. The problem-based research hypothesis is presented, after giving way to the experiment design with the prototype test development with two PHP frameworks. For collection information with help of Understand Tools software and the time of creation and modification of the files, in order to calculate the physical productivity index with executable lines of code measurement (LDC) and time spent (TE) to develop them (hours). The code of lines for time spent of 92.903 LDC/TE and Laravel with 39.102 LDC/TE. It is concluded that the CakePHP framework allows increasing the productivity in the web application development with a higher level than Laravel, the Incident Monitoring System for the network infrastructure was developed by the framework winner. It is recommended to the Information and Communication Technology Department to create the notification functionality via e-mail in order to access the information from outside of the ESPOCH.

**Keywords: <**SOFTWARE PRODUCTIVITY><CAKEPHP FRAMEWORK><LARAVEL FRAMEWORK><INCIDENCE MONITORING><LINES OF CODE><SCRUM METHODOLOGY>

INTRODUCCIÓN

Con la necesidad de mejorar el servicio de atención al cliente prestado a los usuarios de la Escuela Superior Politécnica de Chimborazo, se desea automatizar la gestión y seguimiento de los incidentes reportados por medio de una aplicación web que presente una interfaz amigable. El desarrollo de este sistema implica la utilización de nuevas tecnologías que permitan aumentar la productividad del programador en la Dirección de Tecnologías de la Información y Comunicación (DTIC).

La DTIC es la encargada de proporcionar servicios integrales de calidad en las áreas de desarrollo organizacional y sistemas de información a la ESPOCH y entidades externas, utilizando tecnología de punta, con personal capacitado, estándares de calidad y una participación activa y eficaz del usuario.

Esta dirección tiene como objetivos brindar asistencia y mantenimiento a los sistemas informáticos administrativos, académicos, y de la organización, como también proporcionar servicios de mantenimiento de Hardware, Software y redes computacionales. Esto involucra el seguimiento de los incidentes ocurridos dentro de la ESPOCH, los cuales son atendidos por llamadas telefónicas, correo electrónico o a su vez atendidos en la misma dirección.

La información recolectada se encuentra en bases de datos heterogenias, es decir que es almacenada en documentos en la nube con GoogleDrive lo que causa una descentralización, además de no tener un seguimiento de los mismos. Por esta razón se desea crear un sistema web que permita proveer soporte y asistencia a sus clientes o empleados por medio de un seguimiento de incidentes con el fin de mejorar su servicio de atención y regularizar el volumen de trabajo del personal de soporte.

En la actualidad se ha creado un gran debate sobre la utilización de frameworks, siendo estos una ayuda para incrementar la productividad en el desarrollo de aplicaciones web; o al contrario, posicionarlos como un impedimento para llegar al objetivo de un proyecto de una forma más rápida.

Al programar en lenguajes puros como PHP se llega a tener más control y conocimiento de cómo está estructurado un proyecto de software, pero a la final el trabajo no se encuentra separado o clasificado de acuerdo a sus funcionalidades.

Todo lo descrito anteriormente es lo que lleva a requerir más tiempo en el proceso de programación, pero a nivel de tiempo y productividad la magnitud del proyecto va a variar dependiendo de las necesidades del usuario y con qué rapidez desee hacer uso ya del producto.

La necesidad de incrementar la productividad en el proceso de ingeniería del software y de reducir drásticamente los tiempos de desarrollo y el esfuerzo del programador al momento de proponer una solución al problema en un entorno web, ha cambiado el modelo de construcción tradicional, convirtiendo en una necesidad principal la adopción de frameworks web en vez de la utilización de lenguajes de programación puros.

Estas herramientas que ayudan en el desarrollo de software, se encargan de simplificar las tareas de programación, promoviendo la reutilización del código, fomentado el uso de estándares o convenciones y reduciendo los costes asociados al mantenimiento. Los frameworks son “aplicaciones o conjunto de módulos que permiten, o tienen por objetivo, el desarrollo ágil de aplicaciones mediante la aportación de librerías y/o funcionalidades ya creadas para que nosotros las usemos directamente” (¿Deberías usar un framework para tu proyecto en Internet?,2011).

Hoy en día se cuenta con diversas tecnologías para crear soluciones y productos de alto nivel, y la variedad de frameworks PHP es muy extensa por lo que en la mayoría de problemas se requiere realizar análisis previo que nos permita determinar cuál es el más adecuado para la construcción de un producto de software adaptándose a las necesidades del cliente.

Entre los más populares a: Laravel, Codelgniter, CakePHP, Symphony entre otros, los cuales tienen funciones distintas y como objetivo agilizar el trabajo en términos de productividad, y a la vez ganar tiempo en el desarrollo, dejando a las tareas más simples a cargo del framework y las más complejas a cargo del desarrollador(Smith, 2014).

La decisión de escoger que framework PHP satisfaga las necesidades del desarrollador depende del momento de crear aplicaciones web de calidad, dando solución a problemas comunes como el diseño de arquitectura, codificación, reutilización, internacionalización, plantillas, entre otros, debe estar sustentada por sólidos argumentos como la productividad que presenta cada uno.

Por lo tanto se propone el estudio comparativo de productividad de los frameworks, CakePHP y Laravel, para el desarrollo del Sistema de Seguimiento de Incidentes de la Infraestructura de Red en la ESPOCH para la Dirección de Tecnologías de la Información y Comunicación.

**Justificación**

A continuación se definirán las razones del porqué del tema propuesto, tanto de una forma teórica, metodológica y práctica señalando la importancia de nuestra investigación.

**Justificación teórica.**

El Decreto 1014 “Software Libre en el Ecuador” señala la utilización de software libre en entidades de administración pública tales como la ESPOCH es obligatoria. PHP es un lenguaje de código abierto con diversos frameworks que presentan diferentes características, unos más fáciles de aprender que otros, brindando un aporte para los desarrolladores que deseen escoger con cual trabajar según sus necesidades.

Estos frameworks se crearon como una colaboración directa en el desarrollo permitiendo incrementar la productividad siendo “una capa intermedia entre un lenguaje de programación crudo y el programador”(Usar o no un Framework, 2012), evitando que se invierta tiempo en programar soluciones ya escritas, permitiendo escribir menos código lo que se traduce en menos errores, más funcionalidad, uso de estándares o convenciones, o mejorar la arquitectura del sistema(Alten, 2009, p.1).

De esta manera los frameworks permiten al desarrollador pasar más tiempo identificando requerimientos de software, que tratando con los detalles de bajo nivel para obtener un sistema funcional.

Cabe mencionar investigaciones sobre la medición de productividad en el desarrollo que motivo la presente investigación, tal como “Analyzing PHP Frameworks for Use in a Project-Based Software Engineering Course”(Lancor y Samyukta, s.f. p.4),donde se realiza un estudio comparativo sobre el mismo prototipo desarrollado con diferentes frameworks PHP. Los resultados fueron obtenidos por medio de las medidas de líneas de código y el esfuerzo empleado, llegando a la conclusión que la utilización de Codelgniter permite un desarrollo más rápido en comparación con CakePHP.

Y el caso de estudio empírico “Empirical Case Study of Measuring Productivity of Programming Language Ruby and Ruby on Rails”(Noda y Jia, 2011, p2), que realiza un estudio comparativo entre un lenguaje puro como Ruby y el framework basado en este lenguaje como es Ruby on Rails. Los resultados indican que la productividad medida bajo las mismas condiciones de desarrollo resulta difícil, y que se recomienda realizar más estudios para realizar una comparación de manera científica y empírica.

El estudio Fuente Clever Age sobre los frameworks más usados en PHP, titulado “Libro blanco sobre frameworks PHP para empresas”, se realiza un exhaustivo estudio de todas funcionalidades de dichos frameworks, clasificando a CakePHP entre uno de los frameworks más populares(Comparación y Rendimiento de Frameworks PHP,2008).

La misma investigación menciona a Laravel como “el framework más popular a inicios del presente año, con un porcentaje de 25,85%”(Top PHP Frameworks 2014,2014). Y por último el sitio de Google Trends, clasifica a CakePHP entre los primeros puestos con un promedio de 55% de búsquedas a nivel mundial, y a Laravel en un ascenso en un 56% en el 2014(Google Trends - Web Search interest: Laravel, CodeIgniter, CakePHP, Symfony - Worldwide, Jun 2011 - Feb 2014,2014).Presentándose estos dos frameworks como objetivos principales para este estudio comparativo de productividad.

Este estudio comparativo de los frameworks CakePHP y Laravel, la norma ISO/IEC 9126 define tres vistas diferenciadas en el estudio de la calidad de un producto, clasificando a la productividad como una vista interna de un sistema, la cual se ocupa de medir o estimar las propiedades del software producido como: el tamaño, la complejidad o la conformidad con las normas de orientación a objetos. (International Standard ISO/IEC FDIS 9126-1, 2000, p.3)

Cabe recalcar la existencia del estudio previo, “ANÁLISIS DEL RENDIMIENTO DE FRAMEWORKS PHP PARA DESARROLLAR APLICACIONES WEB ÓPTIMAS. CASO PRÁCTICO: PORTAL ACADEMIA LINUX ESPOCH”, el cual, según la norma ISO/IEC 9126, realiza una vista externa de un sistema, analizando el comportamiento del software en producción (el rendimiento de un software en una máquina determinada, el uso de memoria de un programa o el tiempo de funcionamiento entre fallos.).

**Justificación metodológica.**

Existen 3 comunes formas basadas en el tamaño para definir y medir la productividad, los cuales son: productividad física, funcional y económica. Donde la física trata sobre la relación de la cantidad del producto o líneas de código, a los recursos consumidos o esfuerzo. La funcional es la relación entre cantidad de funcionalidad, es decir puntos de función reales con el esfuerzo. Y por último la económica que relaciona el valor del producto producido con los recursos usados para producirlo. (Card, 2006, p.7)

El estándar IEEE 1045-1992, define un marco para la medición de productividad de software con el objetivo de mejorar la precisión en la recolección y reporte de información relacionada con la productividad del proceso de desarrollo de software, a través un cálculo en términos de esfuerzo combinado con la cantidad de líneas de código (medida de tamaño física, salidas) o puntos de función (medida de tamaño funcional, entradas)(IEEE Standard for Software Productivity Metrics, s.f., pp.6-11).

Por medio del método científico, la investigación se llevará a cabo con el estudio individual de los frameworks CakePHP y Laravel. De esta forma se procederá a crear 2 aplicaciones web con equivalente funcionalidad, con cada una se recolectará datos, los cuales se evaluarán siguiendo las métricas mencionadas anteriormente. Para de esta manera probar la hipótesis de investigación planteada. El framework que presente mayor productividad, será utilizado para el desarrollo del Sistema de Seguimiento de Incidentes de la Infraestructura de Red en la ESPOCH.

La metodología utilizada para el desarrollo del sistema será Scrum debido a que permite “asegurar la calidad del producto de manera sistemática y objetiva, a nivel de satisfacción del cliente, requisitos listos para ser utilizados y calidad interna del producto”(Beneficios de Scrum, s.f.).

**Justificación práctica.**

Hoy en día los sistemas web informáticos son cada vez más complejos y cada vez las empresas o instituciones dependen más de servicios tecnológicos, por lo que un mal funcionamiento o la interrupción de servicios de éstos pueden llegar a tener importantes consecuencias en la consecución de los objetivos de esas entidades.

El porqué de construir una aplicación web para la Dirección de Tecnologías de la Información y Comunicación, viene de la necesidad de mejorar la atención al cliente y así dar un seguimiento a la información sobre los incidentes que son reportados, sin los requerimientos de instalar, descargar, tan solo poseer una conexión a internet y un navegador web y de esta manera empezar a trabajar.

La calidad que brinden a la hora de solucionar incidencias puede ser un hecho determinante para captar o mantener a sus clientes en este caso a los funcionarios de ESPOCH. De ahí también la importancia de que conozcan la valoración de sus clientes en cuánto a la rapidez y eficacia en la solución de los incidentes.

El Sistema de Seguimiento de Incidentes de la Infraestructura de Red en la ESPOCH será desarrollado para DTIC, con el objetivo de proveer soporte y asistencia a los diferentes departamentos o empleados por medio de reportes de incidentes y la administración de los mismos a través de una aplicación web, y valorar la carga de trabajo de cada técnico o funcionario que use el sistema web, reduciendo el tiempo de dedicado a la gestión y planificación de las incidencias según recursos.

El Sistema de Seguimiento de Incidentes contendrá: el módulo de autentificación y administración de usuarios, el módulo de administración de roles, módulo de administración de las áreas de trabajo, el módulo de administración de información de tickets, módulo de seguimiento de incidentes (tickets), y el módulo de reportes.

**Objetivos**

A continuación se detalla el objetivo general y los objetivos específicos de esta investigación.

**Objetivo general.**

Comparar la productividad de los Frameworks orientados a objetos, CakePHP y Laravel para el desarrollo del Sistema de Seguimiento de Incidentes de la Infraestructura de Red en la ESPOCH.

**Objetivos específicos**.

* Estudio individual de los Frameworks CakePHP y Laravel.
* Definir los parámetros de comparación de productividad para el análisis de los frameworks CakePHP y Laravel.
* Construir los prototipos de pruebas que permitan determinar el framework con mayor productividad.
* Comparar los frameworks CakePHP y Laravel con los parámetros establecidos.
* Desarrollar el Sistema de Seguimiento de Incidentes de la Infraestructura de Red en la ESPOCH con el framework ganador.

**Hipótesis**

La productividad del Framework CakePHP es superior a Laravel, en el desarrollo del Sistema de Seguimiento de Incidentes de la Infraestructura de Red en la ESPOCH.

En el Capítulo 1, Marco Teórico se detalla los aspectos teóricos motivo de la investigación, conceptos, terminologías, referenciadas al objeto de estudio.

Continuando con el Capítulo 2, Marco Metodológico, para el estudio comparativo entre los frameworks CakePHP y Laravel se establecen los parámetros de productividad con su respectivo indicador, la definición de herramientas de medición además de la recolección de información. El análisis de los datos del prototipo de prueba se encuentran en el Capítulo 3, siguiendo además con la evaluación de resultados permitiendo determinar que framework brinda una mejor productividad al momento del desarrollo y finalizando con la demostración de la hipótesis en base a los resultados obtenidos.

En el Capítulo 4 se detalla la parte aplicativa de la tesis la cual es el desarrollo del Sistema de Seguimiento de Incidentes para la Escuela Superior Politécnica de Chimborazo con el framework ganador por medio de la metodología ágil Scrum.

CAPÍTULO I

MARCO TEÓRICO

* 1. Aplicaciones web

Una aplicación web es definida como “un programa o conjunto de programas para ayudar al usuario de un ordenador para procesar una tarea específica”(Menéndez, 2012). Cumpliendo así todas las necesidades del usuario, interactuando a través de un navegador para el manejo de datos o información, tan solo con una conexión a internet.

A continuación se describirá y definirá los aspectos correspondientes a las aplicaciones web tal como su historia, características, y cómo funcionan.

* + 1. Historia de las Aplicaciones Web.

En los inicios de la informática en el modelo cliente-servidor, las aplicaciones tenían su propio programa cliente y funcionaba como una interfaz de usuario y está a la vez necesitaba estar instalada en cada computador personal del usuario. Donde las páginas web eran sencillamente una colección de páginas estáticas mostradas al cliente con el único fin de consultar o descargar información que se encontraba de forma permanente.

Viéndose en la necesidad de evolucionar y mejorar el contenido estático de las aplicaciones a un contenido rico e interactivo, se introdujo un método para confeccionar aplicaciones web dinámicas conocido como CGI (Common Gateway interface), el cual definía un “mecanismo mediante el cual podíamos pasar información entre el servidor HTTP y programas externos”(Mateu, 2004, p.20).

El punto débil de este método es que cada vez que se recibía una petición, el servidor web lanzaba un proceso que ejecutaba el programa CGI lo que quiere decir que actuaban bajo demanda del cliente. Por otra parte la mayoría de estos CGI estaban escritos en lenguajes interpretados o en algún lenguaje que requería run-time environment provocando problemas en el rendimiento del servidor, es por eso que a partir de este problema se desarrollaron alternativas como nuevas arquitecturas y lenguajes de programación para el desarrollo web.

El primer lenguaje de desarrollo de aplicaciones web antes de que el internet se convirtiera de acceso general al público, en 1987 conocido como “Perl” inventado por Larry Wall. En 1995 el programador Rasmus Lerdorf puso a disposición el lenguaje PHP, siendo el despegue para la creación de estas aplicaciones web dinámicas(Mateu, 2004, p.20).

El siguiente avance ocurrió meses después cuando el navegador más antiguo y popular conocido como Netscape anuncio una nueva tecnología, “JavaScript” creada por uno de sus programadores Brendan Eich, con el fin de que se ejecutara en el lado del usuario a través del navegador y con un tiempo de respuesta más rápido. En 1996, Macromedia introdujo flash, utilizado para añadir contenido interactivo a tus aplicaciones Web.

En 1999 se introduce el concepto de Java como lenguaje, integrada con dos componentes tanto como al lado del servidor y el cliente, Java Servlet y Java server faces respectivamente. En 2005, el término Ajax fue introducido, y aplicaciones como Gmail se volvieron más interactivas para el usuario. Así como otros lenguajes de programación creados para el desarrollo de aplicaciones web(Mateu, 2004, p.20).

* + 1. Características de las Aplicaciones Web.

Las Aplicaciones web tienen una serie de atributos comunes que las diferencia de otro tipo de aplicaciones de software, y que son las siguientes(eNubes, s.f.):

* **Accesibles:** El usuario puede acceder fácilmente a estas aplicaciones empleando un navegador web.
* **Disponibles:** El usuario puede acceder a estas en cualquier momento y desde cualquier lugar del mundo donde disponga de una conexión a internet.
* **Mantenible:** Se puede actualizar y mantener una única aplicación y todos sus miles de usuarios verán los resultados inmediatamente. Los usuarios no tienen que actualizarla.
* **Portables:** Emplean tecnologías como Java, JavaFX, JavaScript, DHTML, Flash, Ajax entre otras, que dan gran potencia a la interfaz de usuario.
* **Multiplataforma:** Emplean tecnologías que permiten una gran portabilidad entre diferentes plataformas. Se reducen los problemas de compatibilidad.
* **Livianas:** Requieren poco o nada de espacio en disco.
* **Interactivas**: Las aplicaciones web pueden ejecutar audio, vídeo, mostrar animaciones,
  1. Lenguajes de programación.

Un lenguaje de programación es un conjunto de comandos, instrucciones y otros tipos de sintaxis para crear programas de escritorio, aplicaciones web, software entre otros. Existiendo dos tipos de lenguajes: los de alto nivel, que son utilizados por los programadores para escribir código de una manera entendible y fácil de leer, ejemplo de estos están C, Java y PHP en el lado del servidor; y los lenguajes de bajo nivel son conocidos como lenguajes de maquina o lenguaje ensamblador(Programming Language Definition, 2011).

* + 1. Lenguajes de programación para desarrollo web.

Al mismo tiempo de que surgió la necesidad de mejorar el contenido de las aplicaciones web de estáticas a dinámicas, los lenguajes de programación han evolucionado a la par con la interacción de los datos con el usuario a través de la aplicación web. A continuación se describen los más destacados:

* **PHP:** Es un “lenguaje de código abierto muy popular especialmente adecuado para el desarrollo web y que puede ser incrustado en HTML”(PHP: ¿Qué es PHP? - Manual, s.f.). Publicado bajo la licencia PHP, que es incompatible con la Licencia Pública General de GNU.

Además de tener las características de ser multiplataforma, libre por lo cual es de fácil acceso para todos, fácil aprendizaje y tener una capacidad de conexión con la mayoría de las bases de datos que se usan en la actualidad. Siendo este un lenguaje de programación centrado en el lado del servidor por medio de scripts, también realiza algo innovador al lado del cliente, ejecutando el código en el servidor, generando HTML y enviándolo al cliente.(PHP: ¿Qué es PHP? - Manual, s.f.)

* **Java Server Pages(JSP):** Es una tecnología para generar páginas web en el servidor, basado en scripts de igual manera que PHP y ASP, el cual puede incrustar código HTML o tenerlo contenido en su interior (¿Qué es JSP?, s.f.), introduciendo a lo que se conoce como eventos ejecutados por los usuarios. Publicado bajo la Licencia Pública General de GNU, siendo así libre para desarrollar. Entre sus características es que es multiplataforma, permite a los programadores generar dinámicamente HTML, XML o algún otro tipo de página web, permite la utilización de servlets. (¿Qué es JSP?, s.f.)
* **Active Server Page (ASP):** Este es un lenguaje comercializado por Microsoft lo que hace que este bajo la licencia de esta empresa. Siendo ASP.NET el sucesor de la tecnología ASP, siendo definido como “un modelo de desarrollo Web unificado que incluye los servicios necesarios para que usted pueda crear aplicaciones Web de clase empresarial con un mínimo de codificación” (Microsoft, s.f.). Este software es compatible en todos los servidores de Windows ya que tanto los productos son propiedad de Microsoft y se pueden utilizar junto con Microsoft SQL.

Entre estos 3 lenguajes más usados para el desarrollo web, se puede concluir que a nivel de licencias JSP y PHP son libres lo cual facilita el acceso a estas, mientras que ASP.NET necesita de una licencia paga como la de Microsoft. A nivel de curva de aprendizaje, la de PHP es mucho menor que JSP y ASP.NET según los expertos.

Otro punto no mencionado en las descripciones anteriores es el hosting, es más costoso ofrecer entornos de alojamiento para ASP.NET y JSP, principalmente porque ASP.NET y JSP tienden a demandar más recursos de CPU y memoria. Siendo así PHP la mejor opción para el desarrollo web, descrito de mejor manera en el siguiente punto.

* 1. Lenguaje PHP

PHP (Hypertext Pre-processor) es un lenguaje interpretado para la creación de sitios web dinámicos o aplicaciones web alojados en servidores. Permitiendo que  los [scripts](http://www.alegsa.com.ar/Dic/script.php) en PHP se embeben en otros [códigos](http://www.alegsa.com.ar/Dic/codigo.php) como [HTML](http://www.alegsa.com.ar/Dic/html.php) ampliando las posibilidades de diseño web(¿Cuál es la definición de PHP?, s.f.).

* + 1. Historia.

La primera versión de PHP fue creada en el año 1994 por Rasmus Lerdorf, que utilizaba el método CGI, con un conjunto de ficheros de lenguaje C, con el objetivo de rastrear visitas de su currículum online, llamando al conjunto de scripts "Personal Home Page Tools", más frecuentemente referenciado como "PHP Tools". En 1995, Rasmus publicó el código fuente con las mejoras de funcionalidad e interacción con base de datos, proporcionando a los desarrolladores crear aplicaciones web dinámicas sencillas.

En 1996 el código fue hecho de nuevo dándole un nuevo nombre PHP/FI, con nuevas características propias como: soporte interno para DBM, mSQL, y bases de datos Postgres95, cookies, soporte para funciones definidas por el usuario. Entre los años 1997 y 1998 PHP/FI 2.0, la cantidad de usuarios habría incrementado alrededor del mundo con 60,000 dominios reportados con cabeceras que contenían "PHP", indicando en efecto que el servidor host lo tenía instalado(PHP: Historia de PHP - Manual, s.f.).

En 1998, se anunció la versión de PHP 3.0, instalado en más de 70,000 dominios de todo el mundo albergados en servidores que ejecutaban Windows 95, 98, y NT, y Macintosh. En el 2000 fue oficialmente publicado PHP 4.0, utilizando el Motor Zend, soporte para la mayoría de los servidores Web, sesiones HTTP, buffers de salida, formas más seguras de controlar las entradas de usuario y muchas nuevas construcciones de lenguaje.

En el 2004 PHP 5.0 fue lanzado, impulsado por su núcleo, Zend Engine 2.0 que contiene un nuevo modelo de objetos y docenas de nuevas opciones (PHP: Historia de PHP - Manual, s.f.).

* + 1. Características.

A continuación se listan las características más importantes con respecto a este lenguaje (PHP , s.f.):

* Multiplataforma.
* Completamente orientado al desarrollo de aplicaciones web dinámicas con acceso a información almacenada en una Base de Datos.
* Seguro y confiable.
* Capacidad de conexión con la mayoría de los motores de base de datos que se utilizan en la actualidad, destaca su conectividad con MySQL y PostgreSQL.
* Capacidad de expandir su potencial utilizando la enorme cantidad de módulos o extensiones.
* Buena Documentación.
* Es libre.
* Permite aplicar técnicas de programación orientada a objetos.
* Biblioteca nativa de funciones sumamente amplia e incluida.
* No requiere definición de tipos de variables.
* Tiene manejo de excepciones (desde PHP5).
  + 1. Sintaxis.

La sintaxis del lenguaje PHP es similar a la de C y Perl. PHP nos permite embeber sus fragmentos de código dentro de la página HTML, es decir, es posible incluir lenguaje PHP en un código HTML, delimitando nuestro código por etiquetas.

Existen 4 tipos de etiquetas, que indican a PHP dónde empezar y finalizar la interpretación del código ya que todo lo que esté fuera de las etiquetas de PHP será ignorado por el intérprete. En la Figura II.1 se muestran los tipos de etiquetas, de las cuales solo la primera y la cuarta están siempre disponibles y las otras dos pueden ser configuradas en el archivo *php.ini*(Curso de PHP, s.f.).

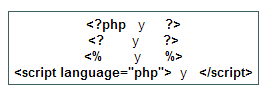


Figura. 1-1. Etiquetas de PHP

**Fuente**: Maria José Samaniego

Como podemos observar el tipo de etiqueta número tres, es un método creado para que tenga compatibilidad con el lenguaje ASP.(Curso de PHP, s.f.)

Variables

Las variables aquí son representadas con un signo de dólar “$”, seguido por el nombre de la variable, las cuales ya quedan declaradas en su primera aparición del código. Estas pueden ser de tipo entero, números en punto flotante, cadenas, array y de tipo objetos. Sin embargo hay muchas formas de declarar variables, pero la forma correcta es que tienen que empezar con una letra (o si no empieza con una letra, tendrá que hacerlo con un carácter de subrayado), seguido de cualquier carácter.

Los tipos de datos que soporta PHP son: Integer, float o double, String y boolean, tomando en cuenta que al asignar las variables no es necesario definir el tipo de dato al que pertenece.(Tipos de variables en PHP. Declaración y asignación. La función echo: mostrar texto en pantalla (CU00816B), s.f.)

Comentarios

El uso de comentarios en PHP y en otro lenguaje es bastante recomendado para documentar el código que se esté haciendo, e indicando que hace cada línea y que hace cada función. PHP soporta comentarios C y C++ existiendo tres tipos los cuales son(“PHP: Comentarios - Manual”, s.f.):

* /\* Comentarios de varias líneas \*/
* // Comentarios de una sola línea
* # Comentarios estilo Bash / Perl (Hasta el final de línea)

Tipos de Datos

El lenguaje PHP posee cuatro tipos de datos primitivos mostrados en la siguiente tabla:

Tabla 1-1: Tipos de datos primitivos en PHP

|  |  |
| --- | --- |
| **Boolean** | Un valor que puede ser verdadero(true) o falso (false) |
| **Int** | Un valor numérico con signo |
| **Float** | Un valor numérico de punto flotante con signo |
| **String** | Una colección binaria de datos que puede ser un texto |

**Fuente:** María José Samaniego

Además, cuenta con dos tipos de datos especiales:

Tabla 2-1: Tipos de datos especiales en PHP

|  |  |
| --- | --- |
| **null** | Representando la ausencia de un valor para una variable |
| **resource** | Recursos representados de forma no nativa por PHP, por ejemplo conexiones a bases de datos y manejadores de archivo. |

**Fuente**: María José Samaniego

PHP también posee tipos de datos compuestos:

Tabla 3-1: Tipos de datos compuestos en PHP

|  |  |
| --- | --- |
| **Array** | Matriz o vector que puede contener diversos valores a la vez. |
| **object** | Estructura con atributos y métodos. |

**Fuente:** María José Samaniego

Constantes

Las constantes son identificadoras o nombres para expresar un valor simple que no puede variar en la ejecución de un script. En PHP las constantes se declaran con la siguiente instrucción:

*Define (‘NOMBRE\_DE\_CONSTANTE’), valor\_de\_constante);*

El nombre de la constante va entre comillas simples o dobles y el valor puede ser cualquier tipo primitivo. Además se recomienda asignar nombres en mayúsculas a las constantes y separar las palabras con un guion bajo, omitiendo el símbolo $(Tipos de datos, variables y constantes en PHP”, s.f.).

* 1. Framework web

A continuación se definirá el concepto de framework web, los objetivos que persiguen, sus características, ventajas y desventajas que presentan.

* + 1. Definición de Framework Web.

El termino framework se lo define como “un conjunto de herramientas, librerías, convenciones y buenas prácticas que pretenden encapsular las tareas repetitivas en módulos genéricos fácilmente reutilizables”( Frameworks (CSS avanzado), s.f.). De tal forma que un framework web, un conjunto de componentes que ayuda a los lenguajes puros con desarrollo de aplicaciones o sistemas web.



Figura. 2-1. Frameworks Web

**Fuente:** <http://www.emprenderalia.com/deberias-usar-un-framework-para-tu-proyecto-web/>

* + 1. Objetivos de los Frameworks Web.

Entre los principales objetivos que persiguen los frameworks web están(Saavedra, 2009, p.4):

* Acelerar el proceso de desarrollo.
* Reutilizar código ya existente.
* Promover buenas prácticas de desarrollo como el uso de patrones de diseño.
* Disminuir el esfuerzo en el desarrollo.
* Tener como aliado a las metodologías de desarrollo Ágiles (XP, Scrum, AD, etc.).
* Mejorar la organización de carpetas y estructura del proyecto, para promover una mantenibilidad fácil del código.
  + 1. Características de los Frameworks Web.

Los frameworks web han ido creando un conjunto de características que los identifica, entre las principales están(Gutiérrez, s.f., p.1):

* Abstracción de URLs y sesiones.
* Acceso a Datos
* Controladores
* Autentificación y control de acceso
* Internacionalización
* Separación entre diseño y estilo.
  + 1. Ventajas de usar Frameworks Web.

Al momento de usar un framework, la velocidad de desarrollo aumenta considerablemente, incrementando así la productividad de desarrollo creando aplicaciones o sitios web con una mayor velocidad, y todo esto con el debido conocimiento de los aspectos básicos de utilización que presenta cualquier framework web.

La reducción de costos es otra ventaja importante con respecto al trabajo que se debe realizar, ya que el framework permitirá al desarrollador centrar el mayor esfuerzo hacia los aspectos esenciales que se debe realizar en un producto de software. La utilización casi obligatoria de estándares y convenciones de código de los frameworks, ayuda a mejorar aplicaciones o sitios web al momento de trabajar en equipo, permitiendo una mejor comprensión del código y un trabajo más organizado(Usar o no un Framework, 2012).

* + 1. Desventajas de usar Frameworks Web.

Obviamente los frameworks web ayudan a desarrollo rápido de aplicaciones, pero aquí viene el inconveniente del aprendizaje ya que primero se debe invertir tiempo en conocer la herramienta y sus aspectos básicos para así proceder a probarlos y utilizarlos. Por otro lado, el conocimiento que se tiene sobre la utilización de un framework, no es el mismo que se tiene sobre el lenguaje sobre el que esta creado(Usar o no un Framework, 2012).

* 1. Frameworks web para PHP

Existiendo una amplia gama de frameworks para la creación de aplicaciones web, siendo unos mejores que otros para la organización de proyectos de acuerdo a su magnitud, y a la vez con funcionalidades distintas, el objetivo común a cumplir es la simplicidad en crear proyectos de calidad en el menor tiempo posible sin la tediosa tarea de volver a escribir código repetitivamente.

Debido a la popularidad del Lenguaje PHP, es que existen muchos frameworks que están disponibles en la web que actualmente capturan al menos el 40% de los sitios top del internet como se muestra en la Figura. 3-1.

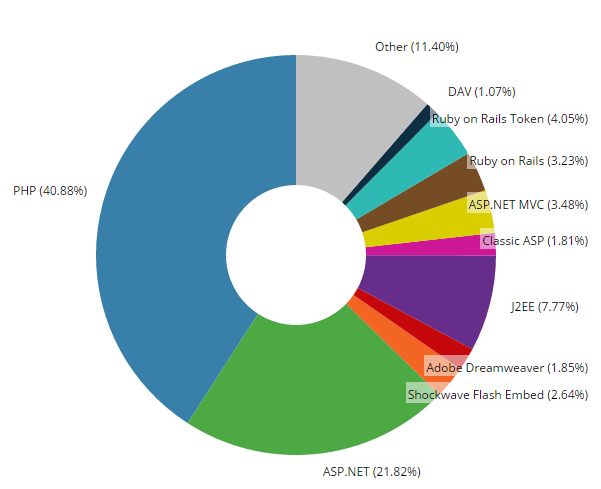


Figura 3-1. Distribución de Frameworks en Sitios Top.

**Fuente:** <http://trends.builtwith.com/framework>

Entre los frameworks más populares para PHP se encuentran(Best PHP MVC frameworks of 2013, 2013):

* **CakePHP.** Es un framework de desarrollo rápido, licenciado bajo MIT License, bajo el patrón de arquitectura MVC, y utiliza funciones solidas de CakePHP para las queries de SQL.
* **Yii.** Es un framework enfocado en el rendimiento bajo el patrón MVC, pudiendo automatizar las funciones básicas CRUD. Licenciado bajo BSD.
* **Laravel.** Es un framework ligero licenciado bajo MIT, también funciona bajo el patrón de arquitectura MVC. Con consultas SQL de tipo plain SQL.
* **Codeigniter.** Este framework es la base para un exitoso sistema de gestor de contenidos, además de ser ligero y rápido. Licenciado bajo la categoría de propietario OSL License.
* **Symfony.** Siendo unos de los frameworks con perspectiva para proyectos empresariales, licenciado bajo MIT License.
  1. CakePHP
     1. Definición.

CakePHP es un framework creado para el desarrollo rápido de aplicaciones en PHP comprendido por librerías, clases, e infraestructuras en tiempo de ejecución, brindando flexibilidad mientras se trabaja de manera estructurada. “Si Ruby tiene su Rails y Java su Trails, PHP tiene todo un pastel: CakePHP es el Rails killer para el desarrollo rápido de servicios web en PHP”(Ramírez, s.f., p.2). En resumen CakePHP hace que la construcción de aplicaciones web sea más fácil, rápida y con menos código.

****

Figura 4- 1. Logo de CakePHP

**Fuente:** Cakephp.org

* + 1. Historia.

El 2005 fue el año de despegue de este framework, cuando Michael Tatarynowicz creó una versión muy básica de plataforma de desarrollo rápido para aplicaciones PHP, dándose cuenta que este sería el comienzo de una nueva solución web. Publicando esta versión bajo la licencia MIT la cual permite no solo usar, copiar, modificar código sino también publicar, sublicenciar o vender copiar de las aplicaciones; apodándola Cake para la comunidad de desarrolladores la conozca. Actualmente su nombre lo mantienen como CakePHP, debido a un juego de palabras referente a los ingredientes que añades a tu gusto a un pastel, y que al crear una aplicación web con este framework en la comunidad de desarrolladores lo conocen como bake (cocinar)(Ramírez, s.f., p.3).

* + 1. Filosofía.

Debido a que la creación de CakePHP, fue bajo la inspiración de Ruby On Rails, comparten la misma filosofía de “Convention over Configuration”, que traducido al español seria Convenciones sobre Configuraciones. Gracias al uso de este simple concepto se obtiene una libre funcionalidad, es decir que el desarrollador se libera del tedioso trabajo del seguimiento del mantenimiento de los ficheros de configuraciones, además de que se estandariza el trabajo en el despliegue, permitiendo a otros conocer de una manera más fácil la estructura de la aplicación.(Convenciones en CakePHP — documentación de CakePHP Cookbook - 2.x, s.f.)

* + 1. Características.

CakePHP posee varias características(What is CakePHP? Why use it? — CakePHP Cookbook 2.x documentation, s.f.), que lo convierten en una buena opción como framework de desarrollo rápido de aplicaciones web PHP, entre las principales están:

* Comunidad amigable.
* Licencia flexible.
* Compatible con las versiones de PHP 5.2.6 y superiores.
* CRUD (Create, Read, Update, Delete) para la interacción de la base de datos.
* Plataforma de código.
* Generación automática de código.
* Arquitectura MVC.
* URLs personalizadas
* Función de Validación.
* Plantillas rápidas y flexibles (La sintaxis de PHP, con ayudantes).
* Ayudantes para AJAX, JavaScript, formularios HTML y más.
* Componentes de Email, Cookie, Seguridad, Sesión y otros.
* ACL (Access Control Lists) flexible.
* Sanitización de Datos.
* Poderoso Caché.
* Localización e Internacionalización.
  + 1. Arquitectura MVC en CakePHP.

CakePHP sigue el patrón de diseño MVC, que obliga a seguir una estructura determinada que ayudará a ampliar y mantener las funcionalidades del código de la aplicación. MVC separa a la aplicación en tres partes: modelo, la vista y el controlador(Entendiendo el Modelo - Vista - Controlador — documentación de CakePHP Cookbook - 2.x, s.f.), las cuales se describen a continuación.

Capas MVC en CakePHP

**Capa del Modelo**

Esta capa es la responsable de la recuperación de datos, donde intervienen las acciones de procesamiento, validación, asociación y cualquier otra tarea relativa a la manipulación de datos. En CakePHP los modelos están representados por una base de datos, tabla o registro, así como sus relaciones con otras tablas o registros. Siendo aquí en esta capa donde reside la conexión a la base de datos(Entendiendo el Modelo - Vista - Controlador — documentación de CakePHP Cookbook - 2.x, s.f.).

**Capa de la Vista**

En cambio esta capa es encargada de la presentación de los datos manipulados por el modelo, a través de una interfaz gráfica con el objetivo de que esta información sea usada por el usuario a partir de peticiones. Las vistas en CakePHP están representadas generalmente por los archivos “view” de tipo HTML con código PHP embebido, aunque no se limitan únicamente a este formato para representar datos, ya que dependen de las necesidades del usuario tales como video, audio, música, documentos, entre otros(Entendiendo el Modelo - Vista - Controlador — documentación de CakePHP Cookbook - 2.x, s.f.).

**Capa del Controlador**

Esta capa es la responsable de dotar de lógica a nuestra aplicación, conocida como la lógica de negocios. Además de gestionar a las peticiones de los clientes al servidor comprobando su validez de acuerdo a las normas de autentificación o Autorización, respondiendo con información solicitada con ayuda de la capa de modelo para leer y escribir en base de datos y otras rutas, y mandar la salida apropiada de datos por la capa de vistas(Entendiendo el Modelo - Vista - Controlador — documentación de CakePHP Cookbook - 2.x, s.f.).

Ciclo de Petición MVC en CakePHP

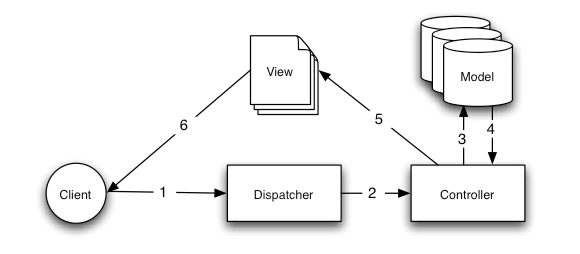


Figura 5-1. Petición típica MVC en CakePHP

**Fuente:** CakePHP.org

El ciclo de petición MVC en CakePHP tal como se muestra en la Figura 5-1, comienza cuando el cliente realiza una petición a la aplicación, la cual pasa por un despachador que las convierte en acciones de controladores pasan al controlador el cual se conecta con el modelo para la manipulación de datos solicitada, al finalizar la comunicación el controlador delega el objeto a la vista correcta para generar una presentación de información correcta al cliente(Entendiendo el Modelo - Vista - Controlador — documentación de CakePHP Cookbook - 2.x, s.f.).

* + 1. Convenciones en CakePHP.

Como se mencionó anteriormente la filosofía de CakePHP, las convenciones son importantes para un desarrollo de aplicaciones uniforme. Debido a que este framework está escrito en inglés es recomendable utilizar los nombres en este idioma, debido a que las palabras son más cortas, y evitar problemas de plurales que ocasionan las palabras en español.

Convenciones de Controladores

Los nombres de las clases de los controladores son en plural, con formato *CamelCased*, y terminan en Controller. *PeopleController* y *LatestArticlesController* son ejemplos que siguen esta convención.

Además de mencionar que el primer método que se debe crear para un controlador sea el *index ()*, ya que este será por defecto el primer método en ejecutarse. Por ejemplo una petición de [*http://www.example.com/cakes/*](http://www.example.com/cakes/) corresponderá con la llama al método *index ()* del controlador *CakesController*.(CakePHP Conventions — CakePHP Cookbook 2.x documentation, s.f.)

Convenciones sobre nombres de fichero y nombres de clases

En general, los nombres de fichero y los nombres de clase serán iguales, en formato *CamelCased*, por ejemplo la clase *CakesController* es encontrada en un archivo llamado *CakesControlller.php*, de la misma manera con los modelos, vistas, componentes, helpers. (CakePHP Conventions — CakePHP Cookbook 2.x documentation, s.f.)

Convenciones para modelos y bases de datos

Los nombres de las clases de modelos están en singular y en formato *CamelCase*. Entre los ejemplos convencionales de nombres de los modelos pueden ser *Person*, *BigPerson*, y *ReallyBigPerson*. Los nombres de las tablas correspondientes a los modelos deben ir en plural y usar el símbolo underscore "\_" o guion bajo.

Por ejemplo *people, big\_people,* y *really\_big\_people*.Los nombres de los campos con más de una palabra se escriben en minúscula y subrayado, por ejemplo *first\_name*. Otra recomendación es al utilizar el atributo id seguido del nombre de la tabla en singular utilizando guion bajo, por ejemplo si la tabla es cakes, su atributo seria *cake\_id*. (CakePHP Conventions — CakePHP Cookbook 2.x documentation, s.f.)

Convenciones en las Vistas

Los nombres de las vistas utilizan el mismo método del controlador al que hacen referencia, en formato subrayado. Siguiendo el patrón para nombrar las vistas:

*/app/View/Controller/underscored\_function\_name.ctp.*

Por ejemplo el método *add ()* del controlador *CakesController* buscará el fichero de vista en la ruta */app/View/Cakes/add.ctp*.(CakePHP Conventions — CakePHP Cookbook 2.x documentation, s.f.)

* + 1. Requerimientos.

CakePHP es rápido y fácil de instalar, los requerimientos mínimos son:

* **Servidor Web**.

Por ejemplo Apache, preferiblemente con *mod\_rewrite*, pero no requerido. Aunque este framework corre sobre la mayoría de servidores web, tales como, LightHTTPD o bien Microsoft IIS.

* **Lenguaje de Programación.**

PHP 4.3.2 o superior. Se puede añadir que CakePHP funciona de una manera mejor y optima en PHP 4 y 5.

* **Servidor de Base de Datos.**

CakePHP soporta una gran variedad de motores de almacenamiento como: MySQL (4 o superior), PostgreSQL, Firebird DB2, Microsoft SQL Server, Oracle, SQLite, ODBC, ADOdb.

* 1. Laravel
     1. Definición.

Laravel, es un framework de código abierto creado por Taylor Otwell, para desarrollar aplicaciones y servicios web con el lenguaje de programación PHP en su versión 5. Que ha venido tomando mucha fuerza en los últimos meses largos y con el lanzamiento de su versión número 4 se posiciona como una opción más que interesante para explorar.(Laravel - The PHP framework for web artisans, s.f.)

****

Figura 6-1. Logo de Laravel

**Fuente:** Laravel

* + 1. Historia.

En el año 2011, el creador de Laravel Taylor Otwell vio la carencia de una importante funcionalidad en el framework CodeIgniter, la cual pensó que era indispensable en la construcción de aplicaciones. Es por eso que para completar esta característica lanzo la primera versión beta de Laravel en junio del mismo año, aun no siendo MVC, poseía otras funcionalidades como Eloquent ORM para operación en la base de datos, localización, modelos, sesiones, vistas, sesiones y mucho más.

La versión 2 de Laravel apareció seis meses después, con una sintaxis limpia y simple para el desarrollo y la adición de convertirse en un framework MVC completo. En Febrero del 2012 Laravel 3 fue lanzado enfocándose en integración de pruebas unitarias, la interfaz de línea de comandos Artisan, migración de base de datos, eventos, drivers de sesiones y bases de datos. Llegando a la actual versión 4 apodada Illuminate, presentando una diferente arquitectura del núcleo del framework (Surguy, 2013).

* + 1. Características.

Entre las principales características están(Laravel Dominicana: Que es Laravel? s.f.):

* Sistema de ruteo.
* Curva de aprendizaje es corta.
* Código Abierto.
* Basado en el framework Ruby y componentes de Symfony.
* Blade, Motor de plantillas.
* Eloquent ORM.
* Basado en Composer.
* Soporte para el cache.
* Buena documentación.
* Soporte para MVC.
* Usa componentes de Symfony.
  + 1. Filosofía.

Laravel se centra en la escritura de código PHP de forma elegante y simple, evitando lo que se conoce como “código espagueti”, es decir evitar que el código sea incomprensible y complejo. El resultado de aplicaciones creadas con Laravel son aplicaciones basadas en PHP 5 con un código sencillo de entender. Es, sobre todo, interesante para leer código, que al contrario cuando no se usa un framework, la comprensión toma tiempo.

Siendo así el propósito de Laravel, de hacer agradable el proceso de desarrollo para el programador sin sacrificar funcionalidad. “Desarrolladores felices escriben el mejor código.” (Documentación de Laravel en Español versión 4.1, s.f.)

* + 1. Arquitectura.

Laravel sigue el patrón de arquitectura MVC tradicional, el cual hace cumplir una separación entre la lógica de negocios, de la lógica de entradas y presentación asociada con una interfaz gráfica de usuario (GUI). Mientras tanto en las aplicaciones web creadas por Laravel, la lógica de negocios típicamente consiste de los modelos de datos como por ejemplo usuarios, publicaciones en blogs, y se podría decir que la interfaz gráfica es solo una página web en el navegador.

Capas MVC en Laravel

**Capa del Modelo**

El modelo en cualquier otro MVC framework, es el componente responsable de la comunicación de la base de datos. Es aquí donde interviene Eloquent que es una clase que se hereda a los modelos en Laravel, para trabajar de manera simple con los objetos de la base de datos. Obviamente el uso de Eloquent es opcional, pues también este framework dispone de otros recursos que facilita interactuar con los datos, o específicamente la creación de modelos(Cogneau, 2012, p1).

**Capa de Vista**

Laravel, usa su propio motor de plantillas llamado Blade, con el objetivo de dar un código limpio y fácil de comprender en las Vistas. Utiliza código PHP plano y además incluye un sistema de Caché que lo hace mucho más rápido. Para la creación de estas plantillas se debe crear el archivo dentro de la carpeta /views con la extensión *nombrevista.blade.php*(Cogneau,2012, p.1)*.*

**Capa del Controlador**

El controlador es donde la lógica de negocio está ubicada y gracias a esto se tienen funcionalidades como: recuperar todas las entradas de la base de datos para enumerarlas, actualizar, eliminar o realizar búsquedas de la información en las tablas de las base de datos dado un identificador, añadir nuevos registros a la base de datos y crear mensajes de confirmación de las funciones básicas. Esta capa permite organizar el código en clases sin tener que escribirlo todo en las rutas. Todos los controladores deben extenderse de la clase BaseController(Cogneau,2012,p.1).

Ciclo de petición MVC en Laravel

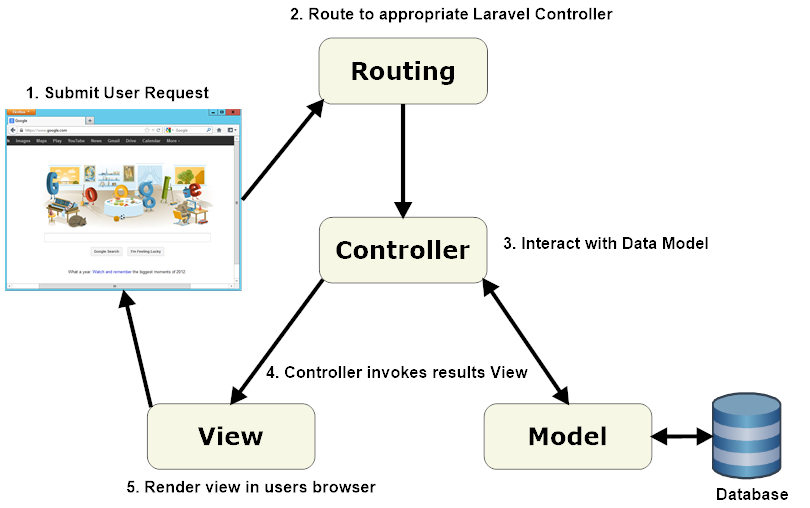


Figura 7-1. Petición típica MVC en Laravel

**Fuente:** <http://xn--diseorapido-4db.es/laravel-que-es/>

Al momento de interactuar con una aplicación desarrollada por Laravel como se muestra en la Figura.7-1, la primera acción que se realiza es cuando el navegador envía una petición al servidor web, y este se la envía al motor de enrutamiento. El siguiente paso es que este router re direcciona esta petición hacia la clase del controlador apropiado según el patrón URL del mismo.

Es aquí donde la clase del controlador realiza su trabajo, siendo el tercer paso la interacción con el modelo, el cual es el encargado de comunicarse con la base de datos. Después de ser invocado el modelo, el controlador interpreta la vista final y regresa la página completa al navegador el usuario con toda la información solicitada. (Architecture of Laravel Applications - Laravel Book, s.f.)

* + 1. Eloquent ORM.

Esta característica es el Object-Relational Mapping de Laravel, que proporciona una manera sencilla de interactuar con la base de datos y relacionar los modelos de la aplicación. Entre los usos del Eloquent ORM están:

* Las funciones básicas CRUD (Create, Read, Update, Delete).
* Configuración y uso de la relación uno a uno.
* Configuración y uso de la relación uno a muchos.
* Configuración y uso de la relación muchos a muchos.(A Guide to Using Eloquent ORM in Laravel, 2014)

El eloquent ORM proporciona una gran implementación con Active Record para trabajar con la base de datos. Es decir que al momento de crear un modelo bajo la estructura MVC, este se corresponde directamente a una tabla en la base de datos. Por ejemplo un modelo llamado Email corresponde a la tabla emails, permitiendo así la manipulación de información de una manera fácil.(A Guide to Using Eloquent ORM in Laravel, 2014)

* + 1. Requerimientos.

Los requerimientos mínimos para el uso de Laravel en su última versión son:

* **Lenguaje de Programación.**

Obviamente para el uso de este framework, se debe tener instalado el lenguaje PHP en su versión 5.3.7.

* **Extensión de MCrypt PHP.**

Es una interfaz para la biblioteca mcrypt, que admite una gran variedad de algoritmos de bloques. En la mayoría de servidores web, como apache esto ya viene incluido.

* **Composer.**

Es un administrador de dependencias para tus proyectos.

* **Servidor Web.**

Por ejemplo Apache, preferiblemente con mod\_rewrite, pero no requerido, otro servidor compatible esta Nginx el cual genera URLs amigables.

* **Servidor de Bases de Datos**

Actualmente Laravel soporta cuatro sistemas de bases de datos los cuales son: MySQL, Postgres, SQLite, y SQL Server.

* 1. Métricas de software
     1. Definición.

Las métricas son aquella aplicación continua en todo proceso o etapa de la ingeniería de software, que relaciona o compara mediciones cuantitativas con un atributo que posee un sistema, proceso o componente. Con el objetivo de tener una mejor visión del proyecto, de la evaluación del producto y a su vez el nivel de aceptación, por medio de la cuantificación y gestión de una forma más efectiva de cada una de las variables a las que se necesita hacer un seguimiento(Salcedo, Gil, y García,2011, p.3). Las magnitudes objeto de valoración son tres: la calidad, fiabilidad y productividad.

Por ende, las métricas de software utilizadas en un proyecto sirven para “ayudar en la estimación, control de calidad, evaluación y medición de la productividad y control de proyectos”(Navarro, s.f.,p2). Pero existen diferentes términos que componen este concepto como, medidas, indicadores, métricas que a continuación se definen cada una de ellas y sus diferencias:

* **Medida:** Proporciona una indicación cuantitativa de la extensión, cantidad, dimensiones, capacidad o tamaño de algunos atributos de un proceso o producto. Existen 4 razones para medir: caracterizar, evaluar, predecir, mejorar. Las medidas no sirven para comparar, ya que solo capturan una característica individual, la medición es la encargada de capturar esta medida.(Navarro, s.f., p.2), por ejemplo 35.000 líneas de código (LDC).
* **Medición:** Es el acto de determinar una medida. Ejemplo: Ana será la encargada de medir las LDC de cada módulo del sistema.
* **Métrica:** En cambio las métricas son una medida cuantitativa que relacionan un valor numérico con una característica, es decir estas sirven para relacionar o comparar medidas. Son el fundamento para los indicadores.(Navarro, s.f.,p.3) Ejemplo: La productividad de este proyecto fue de 500 líneas (LDC/persona-mes).
* **Indicadores:** Es una métrica o una combinación de métricas que proporcionan una visión profunda del proceso o proyecto del software, evaluando su estado y dar un seguimiento a los riesgos potenciales. (Navarro, s.f., p.4)

Ejemplo: La productividad media de nuestra empresa es de 500 (LDC/pm).

* + 1. Importancia de medir Software.

El uso de métricas en el desarrollo de software es una de las prácticas que menos beneficio emocional proporciona a corto plazo, pero mayores ventajas aportan a medio y largo plazo(Mena, 2013).

Tomando en cuenta que lo que no se puede medir no se puede mejorar, y en este caso es una tarea difícil la toma de datos por medio de métricas ya que la tarea de homogeneizarlos influye en muchos aspectos como si son varios o un solo proyecto, varios equipos de trabajo o las diferentes tecnologías que se utilizan.

Los resultados cuantitativos siempre ofrecen el mejor resultado de una actividad y el uso de métricas sirve de ayuda para obtener estos resultados.

* + 1. Características.

Una buena métrica debe poseer las siguientes características(Calero, 2006, p2):

* Ser simples y definidas con precisión.
* Ser objetivas.
* De fácil obtención.
* Válidas.
* Robustas.
* Además es importante que tengan una escala de medición apropiada.
* En especial se preocupan por evaluar la funcionalidad y el presupuesto de un proyecto.

Sin embargo existen otras 4 importantes características que afirman la utilidad de las métricas:

1. Cuantificables, las cuales deben basarse en hechos, no en opiniones.
2. Independientes, los recursos no deben poder ser alterados por los miembros que las apliquen o utilicen.
3. Explicable, debe documentarse información acerca de la métrica y de su uso.
4. Precisas, debe de conocerse un nivel de tolerancia permitido cuando se mide.(Ince, Sharp, y Woodman, 1993., pp.80-81)
   * 1. Métricas del Proceso y del Proyecto.

Se afirma que existen dos tipos de métricas: las métricas de producto y las métricas de proceso, y cabe recalcar que las métricas del proceso se extraen de las del proyecto en una recopilación. Refiriéndose como parte de las métricas de software a las métricas de proceso y producto. (Navarro, s.f., p.4)

Métricas del Proceso

La medición del proceso implica la medición de las actividades relacionadas con el software como el esfuerzo, el coste y los defectos encontrados, con el objetivo de cuantificar el comportamiento del proceso. (Navarro, s.f., p.5).

Usualmente son objetivas, absolutas, explícitas y dinámicas y basadas en opinión de expertos, es decir que estas métricas estarán soportadas empíricamente. Siendo típicas las métricas que cuentan eventos como por ejemplo: número de defectos detectados en inspección, número de requisitos modificados, Número de defectos de diseño detectados durante las pruebas. Se debe considerar además que las métricas del proceso son estratégicas ya que ayudan a determinar el curso del proceso de producción de software. (Navarro, s.f., p.5)

Métricas del Producto

A estas métricas se las definen como una medición del tamaño o cantidad de un producto que se produjo o se piensa producir. Desde los requerimientos hasta el sistema instalado. Hay varias razones para medir un producto(Calero,2006, p.2):

Para indicar la calidad del producto.

* Para evaluar la productividad de la gente que desarrolla el producto.
* Par evaluar los beneficios en términos de productividad y de calidad, derivados del uso de nuevos métodos y herramientas de la ingeniería de software.
* Para establecer una línea de base para la estimación.
* Para ayudar a justificar el uso de nuevas herramientas entre estas los frameworks por ejemplo.

De esta manera las métricas del producto deben medir en términos de calidad o productividad: la complejidad del diseño del software, el tamaño del programa final, o el número de páginas producidas por documentación.

* + 1. Clasificación de las Medidas de Software.

El dominio de esta clasificación de estas medidas es de Proceso, Producto y Proyecto. La medición del software da una clasificación habitual la cual consiste en dividir las medidas de un atributo en dos tipos, directas e indirectas(Calero, 2006, p.2):

* **Medidas Directas:** Estas medidas se obtienen sin la necesidad de utilizar otras medidas de otro atributo. Ejemplos de medidas directas en la ingeniería del software serían la longitud del código, costo, esfuerzo entre otras.
* **Medidas Indirectas:** Son las medidas que se definen y calculan a partir de otras medidas directas de atributos. Ejemplo de estas medidas indirectas están funcionalidad, calidad, complejidad, eficiencia, fiabilidad, productividad.
  1. Productividad
     1. Definición.

Productividad, es definida como las salidas producidas en relación a las entradas o recursos consumidos. Donde se considera a las salidas como el valor entregado y las entradas como los recursos gastados como por ejemplo el esfuerzo, o la influencia de factores externos como herramientas, complejidad, constantes de tiempo entre otras. Se cabe recalcar, que es mucho más fácil medir las entradas, que las salidas y su entorno(Card, 2006, p.2).

La productividad del software es la relación entre el valor funcional del software producido y la mano de obra o gastos para producirlo(Sidler, 2002).En si el concepto de productividad de software viene a ser el resultado de la relación de lo producido por el programador y el esfuerzo empleado para producirlo, ambas siendo medidas directas de software. De acuerdo a la clasificación de medidas de software mencionada anteriormente, la productividad viene a ser parte de una medida indirecta, convirtiéndose en una métrica debido a que depende de otras medidas directas las cuales son las líneas de código y el esfuerzo o tiempo.

* + 1. Factores que afectan a la Productividad.

Existen 4 factores importantes que inciden en la productividad del software(Visconti, s.f., p.2):

* **Factor humano:** Este factor va enfocado en el tamaño del equipo de trabajo, la capacidad individual, años de experiencia en cierto lenguaje de programación o en problemas similares entre otros.
* **Factor del proceso:** dentro de este factor se incluye los lenguajes de programación, ambientes de trabajo, estrategias de programación, revisiones de diseño y código, herramientas case o frameworks, entre otros.
* **Factor del producto:** incluye aspectos de fiabilidad y rendimiento del sistema como por ejemplo tamaño del proyecto y base de datos individualmente, requerimientos de tiempo real, complejidad, cantidad de código reusado, cantidad de documentación, entre otros.
* **Factor del recurso:** Disponibilidad de herramientas CASE, y recursos.

Este estudio comparativo se centrará justamente en el factor del proceso, ya que los frameworks de estudio utilizan funcionalidades diferentes y al medir su productividad se obtendrá los resultados.

* 1. Métricas de productividad

La importancia de medir la productividad de productos de software, viene de la necesidad de identificar que recursos se están sobre utilizando y causan perdida para la empresa(Nwelih y Amadin, 2008, p.1).

Las métricas de productividad de software (IEEE Standard for Software Productivity Metrics, 1993,p.5), definen 3 categorías de métricas las cuales son: métricas orientadas al tamaño, orientadas a la función, y orientadas a documentación. Aunque existen otros modelos de métricas propuestas, las mencionadas anteriormente son las más usadas en cuanto al producto de software. Por lo tanto la productividad se mide la siguiente manera:

**Ecuación N° 1**

Donde:

**P**= Productividad en el desarrollo de aplicaciones web

**SP**= Salidas producidas

**RC**= Recursos consumidos

* + 1. Métricas Orientadas al Tamaño.

Estas métricas se calculan dividiendo la medición del software desarrollado o entregado de conformidad, por el tiempo empleado siendo otra forma para medir la productividad de un producto de software. Donde las salidas utilizan como medida de tamaño al número de líneas de código, aunque existen otras medidas propuestas como número de clases, numero de métodos, número de páginas de documentación, número de casos de uso, o requerimientos(IEEE Standard for Software Productivity Metrics, 1993, p.5).

**Líneas de Código (LDC)**

Las líneas de código es una medida física del producto de software, sin embargo se debe definir qué se debe contar y que no. Es aquí donde aparecen dos opciones, las cuales son 1) si contar líneas de comentarios o no y 2) si contar líneas de código o sentencias. Definiendo a esta medida física como una medida de líneas de código no en blanco(IEEE Standard for Software Productivity Metrics, 1993, p.5).

* **Líneas no comentadas:** son las líneas de software que contienen ya sea código ejecutable, declaración de variables o sentencias de compilación.
* **Líneas comentadas:** líneas que tan solo contienen declaraciones de comentarios.

Entre las ventajas de esta medida física esta que es fácil de calcular, y los inconvenientes que presenta es que dependen del lenguaje de programación y perjudican a los programas cortos pero bien diseñados, ya que mientras más detallado y organizado sea el programador, la productividad será mayor(Ebert, 2006, p.6).

**Herramientas de medición Líneas de Código**

**CLOC**

En la actualidad existe un script escrito en Perl capaz de sacar informe, a partir del directorio raíz de un proyecto, que permite conocer las líneas reales de código fuente, su tipo (código, comentario o línea en blanco) y su distribución (SQL, XML, ASP.NET o C#) empleo, llamado CLOC(“CLOC -- Count Lines of Code”, s.f.). Además de reconocer cientos de lenguajes de programación.

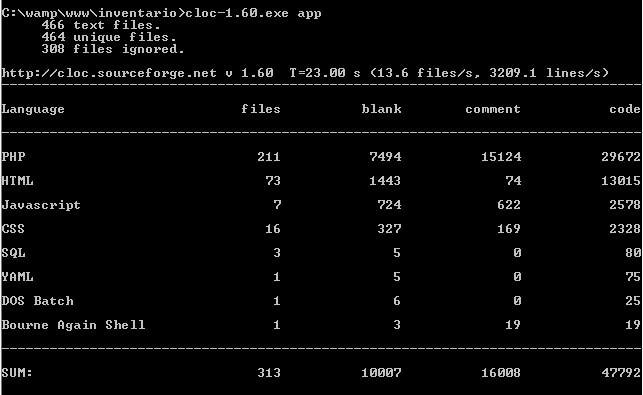


Figura 8-1. Script CLOC

**Fuente:** [http://jerodsanto.net/2013/02/cloc-counts-lines-of-code**/**](http://jerodsanto.net/2013/02/cloc-counts-lines-of-code/)

**Understand**

Existe además otra herramienta con licencia llamada Understand creada por Scitools, que permite realizar un análisis pertinente de la información del código. Brindando información sobre funciones, clases, variables, etc.(“UnderstandTM Static Code Analysis Tool | SciTools.com”, s.f.). Por medio de la opción de reportes de métricas, se provee información estadística sobre el proyecto y entidades tales como líneas de código como se muestra en la Figura. 9-1.

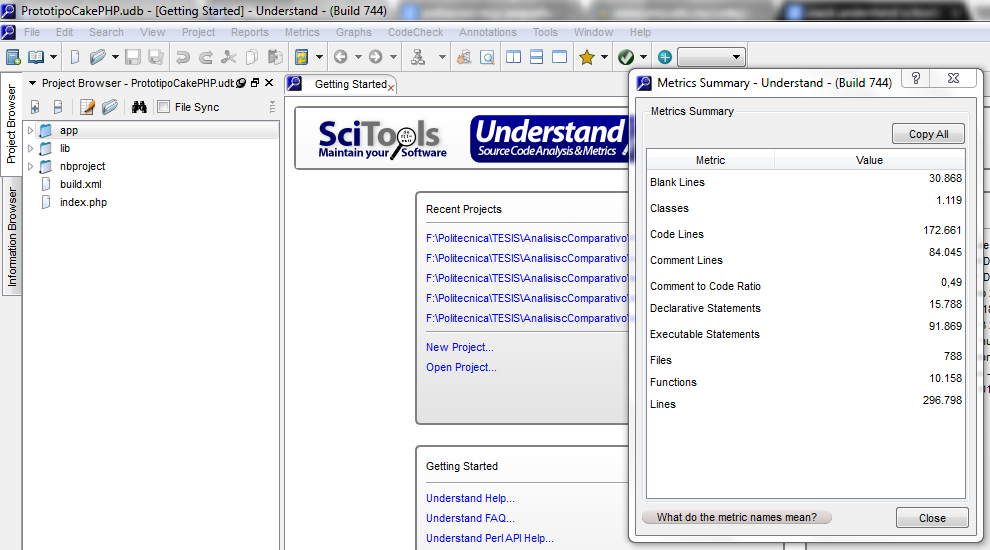


Figura 9-1. Understand Tool

**Fuente**: María José Samaniego

**Esfuerzo.**

Respecto a las entradas en la medición de productividad, la unidad de medida es el esfuerzo en persona-mes, donde se debe incluir a todos los que intervinieron directamente en la creación del producto de software. Para obtener un mejor resultado en la productividad, se debe tener más precisión en capturar el esfuerzo empleado en un producto(IEEE Standard for Software Productivity Metrics, 1993, p.9).

* + 1. Métricas Orientadas a la Función.

Se encarga de medir la funcionalidad del proyecto, y debido a que no se puede medir directamente, se convierte en una medida indirecta. Se considera según algunos expertos que “la funcionalidad captura de mejor manera el valor entregado que las métricas orientadas al tamaño”(Ebert, 2006, p.2).

Utilizando como unidad de medida de salida producida a los llamados puntos de función, y las entradas el esfuerzo. El estándar IEEE 1015-1992 de medición de productividad de software no muestra como requerimiento utilizar esta métrica, pero los desarrolladores o grupos de trabajo que deciden utilizarla deben seguir lo establecido con las métricas lógicas orientadas al tamaño o instrucciones lógicas de software.

**Puntos de Función**

Está basada en una combinación de características de un programa como se muestra en la Figura. 10-1, y cada una asociada con un peso, el total de los puntos de función es modificado de acuerdo a la complejidad que presentan los proyectos(IEEE Standard for Software Productivity Metrics, 1993, p.9).

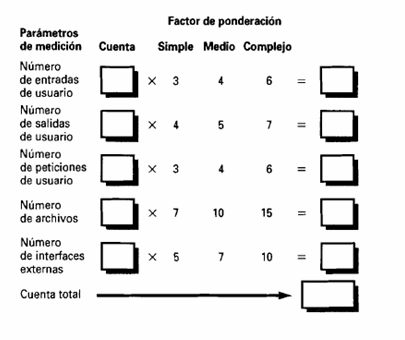


Figura 10-1. Cálculo de cuenta total de Puntos de Función

**Fuente**: <http://samuelartwaltz.blogspot.com/2010/10/metricas.html>

* **Entradas de Usuario:** son los datos que proporcionan los usuario y lo ingresan en la aplicación o sistema web.
* **Salidas de Usuario:** es la información que la aplicación ofrece al usuario como mensajes de error.
* **Peticiones de Usuario:** estas son las salidas de información producidas por las peticiones que realiza el usuario a la aplicación o sistema web por medio de entradas interactivas.
* **Archivos:** son los archivos que presenta el producto, por ejemplo de configuración de base de datos o sistema de archivos.
* **Interfaces Externas.** Estas vienen a ser las interfaces que se utilizan para comunicar y transmitir información entre sistemas web diferentes.(Navarro, s.f., p.12)

En fin el uso de puntos de función para medir la productividad se convierte en una estimación real que presenta el proyecto y son independientes del lenguaje de programación, el problema es que esta métrica es más subjetiva que las líneas de código. El cálculo de la medida de esfuerzo viene a ser el mismo establecido en las métricas de tamaño.

* + 1. Métricas Orientadas a la Documentación.

El estándar IEEE 1045-1992 define a la medición de productividad de acuerdo a todos los documentos que son utilizados como recursos para el desarrollo de un proyecto. Donde los documentos incluyen copias impresas, captura de pantallas, texto y gráficos los cuales son utilizados para brindar y trasmitir información a los usuarios.

Páginas de Documentación

Hay que tener en cuenta que el número de página viene a ser la medida para las salidas en cálculo de la productividad, y el esfuerzo las entradas, de la misma manera que las otras métricas descritas anteriormente. La manera de contar las páginas de documentos impresos es por medio de las páginas que no estén en blanco, las que no son impresas por medio de páginas por pantalla(IEEE Standard for Software Productivity Metrics, 1993, p.10).

Tokens de Documentación

Esta medida de salida por medio de la documentación cuenta de tres tokens o piezas fundamentales: palabras, ideogramas, y gráficos.

* **Número de Palabras:** estas siguen reglas como por ejemplo palabras simples, numeraciones, acrónimos son contadas como una sola palabra, además de que los signos de puntuación son ignorados en la cuenta.
* **Número de ideogramas:** aquí se cuenta cada símbolo del ideograma por separado.
* **Número de gráficos:** se toman en cuenta el número de gráficos, tablas, figuras, imágenes por separado(IEEE Standard for Software Productivity Metrics, 1993, p.10).
  1. Líneas de código vs. Puntos de función vs. Documentación

El cálculo de medición de productividad, como se mencionó anteriormente tiene 3 enfoques como el tamaño, la funcionalidad y documentación, donde el mismo concepto es aplicado a todos que es salidas sobre las entradas. Pero como diferenciar cada uno de ellos, que diferencias tienen, o cual métrica es la apropiada para un proyecto de software.

Tomando en cuenta que la medición de las salidas es algo difícil, ya que este valor solo existe en la perspectiva del desarrollador se presentara a continuación un ejemplo como se puede observar en la Figura 11-1, que mostrara dos diferentes programas pero con la misma funcionalidad exacta, del cual se medirá la productividad con las métricas orientadas al tamaño y orientadas a la función y sus variantes.

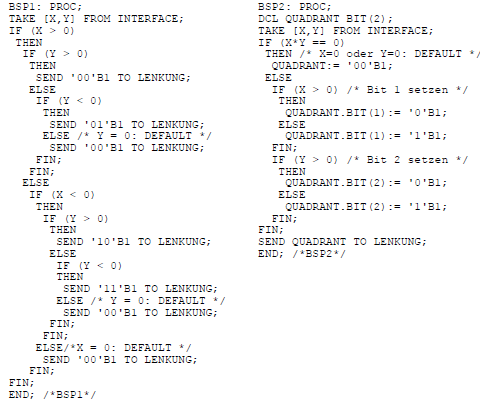


Figura 11-1. Controlador Embebido Con Sistema de Sonar

**Fuente:** (Ebert, 2006)

Para el correspondiente análisis del ejemplo propuesto en la Figura 11-1**,** se asume que el esfuerzo empleado es el mismo para desarrollar estos dos códigos ejecutables.

**Líneas de Código.**

Al contar las líneas de código de ambos programas, tenemos el que se encuentra a la izquierda es equivalente a 34 líneas de código y el de la derecha 22 líneas de código. De esta manera se llega a calcular que el código 2 es 50 veces más productivo que el primero, debido a la menor cantidad de líneas escritas en el mismo tiempo.

Al analizar los factores que afectan a la productividad como en esta investigación seria el factor del proceso donde intervienen los frameworks, en un mismo proyecto, las líneas de código, u otra medida de tamaño no serán iguales ya que dependen de cómo trabaja la herramienta, debido a sus diferentes maneras de ayudar a reducir el desarrollo y gastos que causan las pruebas en las aplicaciones web creadas.

**Puntos de Función:** En cambio con puntos de función, la funcionalidad será la misma en los segmentos del código 1 y 2, dando como resultado la misma productividad. Esta métrica es apropiada para comparar diferentes proyectos, ya que contienen diferentes funcionalidades y requerimientos. Al usar esta métrica en esta investigación, el resultado será el mismo.

**Documentación:** Lo mismo sucederá con esta métrica, ya que al utilizar como medida el número de páginas de documentación o número de elementos, el resultado no variara, dando por consecuente la misma productividad ya que se trata del mismo proyecto.

* 1. Desarrollo ágil de aplicaciones web

Las aplicaciones web deben cumplir con la característica principal de trabajar con requisitos o variables desconocidas para de esta manera crear rápidamente prototipos y de versiones previas a la entrega final del producto, lo cual será de agrado al cliente permitiendo de esta manera tener un menor riesgo ante la posibilidad de cambios en los requisitos funcionales.

Tomando en cuenta además de la reciente aparición de las metodologías denominadas ágiles, las cuales se adaptan al proceso de desarrollo al desarrollo software específico y que a su vez permiten fácilmente reunir y contemplar cambios en los requisitos(Marcos y Cáceres , s.f., p.2).

Permitiendo así una visión clara, dinámica y en tiempo real de lo que se está trabajando dentro del equipo, y a su vez tener una participación del cliente en cada momento para obtener los resultados que este desee.

* + 1. Ventajas de las Metodologías Ágiles.

Las metodologías ágiles presentan diversas ventajas tales como (“Metodologías Agiles de Desarrollo”, 2009):

* Respuesta rápida a cambios de requisitos a lo largo del desarrollo y entrega continua y en plazos cortos de software funcional.
* Trabajo conjunto entre el cliente y el equipo de desarrollo.
* Minimiza los costos frente a cambios.
* Importancia de la simplicidad, al eliminar el trabajo innecesario.
* Atención constante la excelencia técnica y al buen diseño.
* Mejora continua de los procesos y el equipo de desarrollo.
* Evita malentendidos de requerimientos entre el cliente y el equipo.
* El equipo de desarrollo no malgasta el tiempo y dinero del cliente desarrollando soluciones innecesariamente generales y complejas que en realidad no son un requisito del cliente.
* Cada componente del producto final ha sido probado y satisface los requerimientos.
  + 1. Principales Metodologías Ágiles.

Aunque los creadores e impulsores de las metodologías ágiles más populares han suscrito el manifiesto ágil y coinciden con las ventajas mencionadas anteriormente, cada metodología tiene características propias y hace hincapié en algunos aspectos más específicos. A continuación se listan dichas metodologías ágiles( Letelier y Penadés, s.f.):

* SCRUM
* Crystal Methodologies
* Dynamic Systems Development Method (DSDM)
* Adaptive Software Development (ASD)
* Feature-Driven Development (FDD)
* Lean Development (LD)
  + 1. Metodología SCRUM.

Origen

Scrum es una metodología ágil de desarrollo de proyectos que toma su nombre y principios de los estudios realizados sobre nuevas prácticas de producción por Hirotaka Takeuchi e Ikujijo Nonaka a mediados de los 80 en su artículo” The New Product Developroent Game”, donde dan a conocer una nueva forma de gestionar proyectos donde la agilidad es el elemento principal.

A partir de esta investigación, Jeff Sutherland aplicó el modelo Scrum al desarrollo de software en 1993 en Easel Corporation (Empresa que en los macro-juegos de compras y fusiones se integraría en VMARK, luego en Informix y finalmente en Ascential Software Corporation). En 1996 junto con Ken Schwaber presentaron como formal este proceso para el desarrollo de software y que pasaría a incluirse en la lista del Agile Alliance(Palacio, 2006, p.1).

Introducción a SCRUM

Scrum incrementa significativamente la productividad y reduce el tiempo de espera para ver los beneficios del proyecto, así como facilitar la adaptación continua de los sistemas desarrollados en su evolución.

Parte de la base de que los procesos definidos funcionan bien, sólo si las entradas están perfectamente definidas y el ruido, ambigüedad o cambio es muy pequeño. Por lo tanto, resulta ideal para proyectos con requerimientos inestables, ya que fomenta el surgimiento de los mismos.

**Principales Características de SCRUM**

Entre las principales características que presenta Scrum se encuentran(Peralta, 2003, p.2):

* Equipos auto dirigidos
* Utiliza reglas para crear un entorno ágil de administración de proyectos
* No prescribe prácticas específicas de ingeniería
* Los requerimientos se capturan como ítems de la lista Product Backlog
* El producto se construye en una serie de Sprints de un mes de duración

**Roles SCRUM**

Existen tres roles para esta metodología los cuales son(Peralta, 2003, p.9):

* **Product Owner:** Este rol representa a todas las personas interesadas en el producto final. Con el fin de encontrar la visión del producto y representarla en el Backlog List.
* **Scrum Master:** Es un rol de administración que debe asegurar que el proyecto funcione según lo planeado y reducir riesgos del producto.
* **Scrum Team:** Este es el equipo que tiene la autoridad para decidir cómo organizarse para cumplir los objetivos del Sprint.

**Herramientas SCRUM**

Según Peralta, SCRUM especifica herramientas de gerencia que se aplican en sus distintas fases para evitar el caos originado por la complejidad e imposibilidad de realizar predicciones(Peralta, 2003, p.4), las cuales se definen a continuación :

* **Product Backlog List:** Este es el listado de los requisitos del sistema, el cual puede crecer y modificarse a medida que se obtiene más conocimiento acerca del producto y del cliente.
* **Sprints:** Son ciclos iterativos en los cuales se desarrolla o mejora una funcionalidad para producir nuevos incrementos, pero no se puede introducir cambios durante un sprint. Las actividades que se desarrollan durante un sprint son: Sprint Planning Meeting, Sprint Backlog, Daily Scrum Meetings y Sprint Review Meeting.
* **Burn Down Chart:** Es una gráfica que ayuda a medir o recalcular la velocidad del trabajo de manera diaria.
* **Sprint Backlog:** Es el listado de tareas en el que subdivide las historias de usuario que describen las funcionalidades que componen un proyecto. Este listado se define y estima en la reunión de Planificación del Sprint al inicio de la iteración. Las tareas deben ser pequeñas y poco acopladas para poder estimarlas. Se añaden tantas tareas como velocidad tenga el equipo.

**Proceso SCRUM**

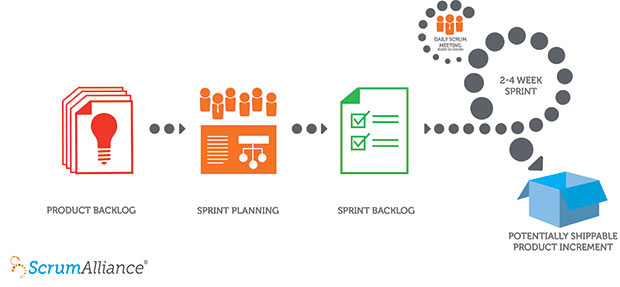


Figura 12-1. Proceso de la metodología Ágil Scrum

**Fuente:** (What is Scrum? An Agile Framework for Completing Complex Projects - Scrum Alliances, s.f.).

En la Figura. 12-1 podemos ver que el proceso empieza cuando el Product Owner crea una lista de deseos priorizada llamada Product Backlog. Durante el Scrum Planning el equipo extrae una pequeña porción de tareas de la parte superior de la lista de deseos, creando así el Sprint Backlog, y de esta manera deciden como implementarlas.

El equipo o Scrum Team tiene cierta cantidad de tiempo, un sprint (usualmente 2 a 4 semanas), para completar su trabajo, pero existen reuniones diarias para evaluar su progreso (Daily Scrum). En el Camino el Scrum Master mantiene al equipo enfocado en su meta. Al final de este Sprint el trabajo debe ser potencialmente entregable es decir, listo para entregar al cliente o mostrar al stakeholder(What is Scrum? An Agile Framework for Completing Complex Projects - Scrum Alliances, s.f.).

Siendo de esta manera el fin del sprint con una revisión y una retrospectiva del mismo. Al comenzar el siguiente sprint, el equipo elije otra pequeña porción de la lista de tareas del Product Backlog y empieza a trabajar nuevamente.

CAPITULO II

MARCO METODOLÓGICO

**ANÁLISIS COMPARATIVO DE PRODUCTIVIDAD DE LOS FRAMEWORKS CAKEPHP Y LARAVEL**

* 1. Introducción

Los frameworks son creados con el fin de simplificar tareas repetitivas a los desarrolladores, permitiendo de esta manera centrarse en las actividades de mayor prioridad en la creación de una aplicación web. Debido a la aparición de varios frameworks, es necesario realizar un estudio de los mismos con el fin de elegir los más aptos y a su vez los que se adapten de mejor manera a las necesidades que demanda un proyecto.

* 1. Tipo de investigación

El método científico utilizado para esta investigación será de carácter descriptivo e inferencial, ya que estará dirigido a determinar la como es la situación de la información recolectada dentro de una población, y a su vez a la toma de decisiones por medio de estas. En consideración de lo expuesto anteriormente, se puede definir que la investigación será de tipo inductiva.

Es por ello que para la realización del estudio comparativo de productividad entre CakePHP y Laravel, se definirá el parámetro de productividad por medio del índice de productividad física, el cual contiene sus respectivos indicadores. En el término de la comparación se selecciona el framework con mayor puntuación para el desarrollo del sistema SDSI-DTIC.

* 1. Definición del parámetro de productividad para la comparación

Para la comparación de los Frameworks CakePHP y Laravel, se ha tomado como referencia la investigación sobre la medición de productividad del software realizada por David N. Card, denominada *The Challenge of Productivity Measurement* (Card, 2006, p.3)**,** donde se define un modelo simple de productividad demostrada en la Figura. 1-2como la relación entre las salidas producidas y los recursos consumidos para realizarlas.

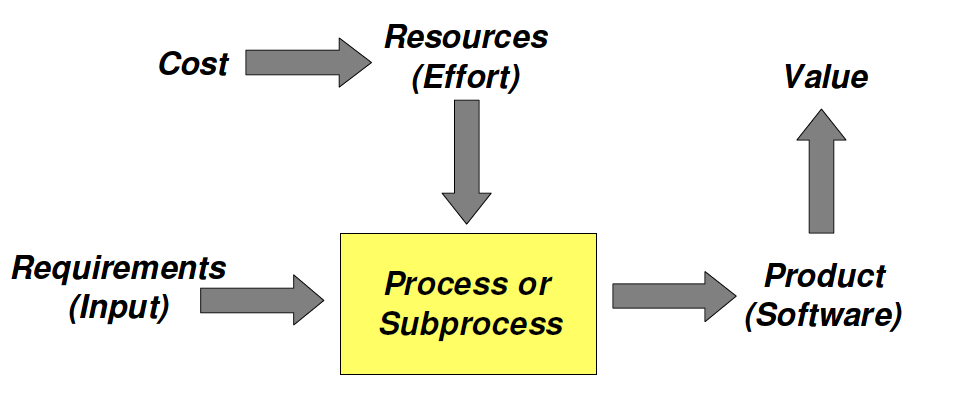


Figura 1-2 Modelo Simple de Productividad

**Fuente:** (Card, 2006)

En la terminología del estándar de métricas de productividad de software del Instituto de Ingenieros Eléctricos y Electrónicos (IEEE Standard for Software Productivity Metrics, 1993, p.5-11), se habla de 2 Primitivas las cuales hacen referencia al nivel más bajo en el que se recogen los datos:

* Las de salida, referenciándose a las características finales de software que pueden ser medidas por la cantidad de líneas de código ejecutable escritas por el desarrollador o páginas de documentación.
* Las de Entrada que miden lo que ha sido necesario utilizar para construir el software.

Se ha considerado definir el índice e indicadores de productividad física por medio de la adaptación del estándar IEEE 1045-1192 y el modelo simple de productividad de Card, para proceder con el estudio comparativo.

Se describe a continuación el índice e indicadores para la determinación del mejor framework con mayor productividad en el desarrollo.

Tabla 1-2: Parámetros de comparación a valorar

|  |  |  |
| --- | --- | --- |
| **Parámetro** | **Índice** | **Indicadores** |
| Productividad  del desarrollo | Índice de Productividad  Física | Líneas de código |
| Tiempo empleado |

**Fuente:** María José Samaniego

* 1. Definición del índice de productividad física

Este índice responderá a la relación entre los indicadores de las salidas producidas y los recursos consumidos en el desarrollo de una aplicación web. De modo que estos no se comportarán linealmente, sino que las variaciones de cada indicador afectarán al valor final del índice de forma dependiente a los valores de indicadores. Cabe mencionar que este índice se lo cataloga como una medida indirecta de software.

La toma de decisión general será tomada en base a este indicador, por medio de conclusiones estadísticas. Este índice se encuentra descrito en la Tabla 2-2.

Tabla 2-2: Índice de Productividad física

|  |  |
| --- | --- |
| **ÍNDICE** | **DESCRIPCIÓN** |
| Índice de Productividad Física | Es la relación definida por la cantidad de líneas de código ejecutable y el tiempo empleado para su desarrollo. Mientas el valor promedio sea mayor, mayor será la productividad en el desarrollo. |

**Fuente:** María José Samaniego

Este índice se calculara por medio de la siguiente ecuación:

**Ecuación N°2**

Donde:

**IPF**= Índice de Productividad Física

**LDC**= Líneas de Código

**TE**= Tiempo Empleado

A continuación se definirá cada uno de los indicadores correspondientes al índice de productividad física establecidos en la Tabla 1-2.

* + 1. Indicador 1: Líneas de código (LDC).

El indicador líneas de código o LDC catalogado como una medida directa de software, se encarga de medir las salidas producidas de un producto de software.

Con este indicador se determinará la diferencia de tamaño de un proyecto a otro. En la Tabla 3-2 se describe este indicador.

Tabla 3-2: Indicador líneas de código

|  |  |
| --- | --- |
| **INDICADOR** | **DESCRIPCIÓN** |
| Líneas de código | El número de líneas de código ejecutable no comentado, que ha sido escrito por el programador. Mientras este número sea menor, mayor será la productividad. |

**Fuente:** María José Samaniego

* + 1. Indicador 2: Tiempo Empleado (TE).

Este indicador se refiere a los recursos consumidos en el desarrollo de un producto de software dentro de la fase de codificación. De tal modo que este indicador determinara la cantidad de tiempo invertido en el desarrollo. La Tabla 4-2 describe este indicador.

Tabla 4-2: Índice de tiempo empleado

|  |  |
| --- | --- |
| **INDICADOR** | **DESCRIPCIÓN** |
| Tiempo empleado | Es el número de horas que se dedicaron para la codificación de un producto de software por parte del desarrollador, entre la fecha de creación y su última modificación dentro de un horario de trabajo de 8 horas diarias de lunes a viernes. A menor valor, mayor será el índice de productividad física, lo que implica mayor velocidad de desarrollo de una aplicación web y además menos coste para la empresa. |

**Fuente:** María José Samaniego

* 1. Diseño del experimento

El estudio comparativo se basará en la recolección de datos contenidos en los archivos que han sido creados por el programador dentro del prototipo de prueba desarrollado con cada uno de los frameworks establecidos para esta investigación.(Lancor y Samyukta, s.f., p.4).

Cabe recalcar que la toma de información estará dividida en tres elementos, el Modelo, la Vista y el Controlador (Noda y Jia, 2011, p.2).

* + 1. Método de muestreo.

El método de muestreo que será utilizado en esta investigación será el no probabilístico. Se aplicará para los archivos creados en el prototipo de prueba con cada uno de los frameworks a comparar. Debido a que estos archivos fueron los que se crearon y codificaron por parte del desarrollador.

* + 1. Tipo de muestreo.

Para la investigación se utilizará el muestreo de juicio o discrecional, donde se toma la muestra seleccionando los elementos que al investigador le parece representativos. Se selecciona los 11 archivos correspondientes al prototipo de prueba por cada framework creados por el desarrollador dentro del patrón de arquitectura Modelo-Vista-Controlador.

* + 1. Determinación del escenario de prueba.

Con el objetivo de que las pruebas se realicen en un entorno de total igualdad para CakePHP y Laravel se diseñó un prototipo con funcionalidades exactas (insertar, modificar, eliminar, y listar) correspondiente a la Administración y Autenticación de Usuarios; el cual será utilizado para las pruebas de tamaño, esfuerzo y de esta manera una vez obtenida la información proceder a calcular el índice de productividad física.

El prototipo de prueba constará de las siguientes acciones:

1. Autentificación de Usuarios
2. Creación de Roles
3. Modificación de Roles
4. Eliminación de Roles
5. Listar Roles
6. Creación de Grupos de Trabajo
7. Modificación de Grupos de Trabajo
8. Eliminación de Grupos de Trabajo
9. Listar Grupos de Trabajo
10. Creación de Usuarios
11. Modificación de Usuarios
12. Eliminación de Usuarios
13. Listar Usuarios

El prototipo constará con 11 elementos los mismos que serán creados y codificados por el programador (Ver Anexo A1).

**CakePHP**

1. /app/Model/Group.php
2. /app/Model/Role.php
3. /app/Model/User.php
4. /app/View/groups/index.ctp
5. /app/View/roles/index.ctp
6. /app/View/users/index.ctp
7. /app/View/users/login.ctp
8. /app/View/Pages/home.ctp
9. /app/Controller/GroupsController.php
10. /app/Controller/RolesController.php
11. /app/Controller/UsersController.php

**Laravel**

1. /app/models/Group.php
2. /app/models/Role.php
3. /app/models/User.php
4. /app/views/groups/index.blade.php
5. /app/views/roles/index.blade.php
6. /app/views/users/index.blade.php
7. /app/views/users/login.blade.php
8. /app/views/home.blade.php
9. /app/controllers/GroupsController.php
10. /app/controllers/RolesController.php
11. /app/controllers/UsersController.php
    * 1. Herramientas para el estudio comparativo

Se describirá a continuación las herramientas que serán utilizadas para la medición de los indicadores y además para su correspondiente procesamiento.

**Herramientas para la medición de indicadores.**

Para la recolección de datos de los indicadores establecidos se utilizará herramientas que permitirán determinar de manera eficiente el código que fue escrito por el programador, y a su vez el tiempo que se tardó en producirlo.

* Para el indicador correspondiente al número de líneas de código se utilizará la herramienta Understand de SciTools, que emitirá un reporte del total de líneas de código de ejecución no comentadas por cada archivo de la muestra.
* Para el tiempo empleado por cada archivo de la muestra, se utilizará los tiempos MAC dentro de las propiedades de Windows, tales como el ctime (created time) y mtime (modify time).

**Herramienta para procesamiento de datos**

La herramienta para procesar los datos obtenidos será SPSS 20 en español. La cual me permitirá realizar la estadística descriptiva e inferencial.

* + 1. Elección de la Prueba Estadística.

La elección de la prueba estadística para la prueba de hipótesis, tomará en cuenta los siguientes supuestos:

* Los datos se distribuyan normalmente.
* Que las muestras sean independientes.

Se determina un valor de significancia (α) del 5% equivalente a 0.05, ya que dicho valor es establecido por defecto para las pruebas de hipótesis. Siendo así el nivel de confianza con un 95%.

A continuación se realiza la prueba de normalidad Shapiro-Wilk con la muestra para esta investigación con cada framework:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Tabla 5-2: Pruebas de normalidad | | | | | |
|  | **FRAMEWORK** | Shapiro-Wilk | | |
|  | Estadístico | gl | Sig. |
| IPF | CakePHP | 0.919 | 11 | 0.308 |
| Laravel | 0.831 | 11 | 0.024 |

**Fuente:** Maria José Samaniego

Como se puede observar el nivel de significación o P-Valor de los frameworks en la prueba de normalidad, se llega a la conclusión de que los datos no se distribuyen normalmente en cada una de las muestras independientes. Por tal motivo se descarta la opción de utilizar la prueba T de Student.

La mejor opción para la prueba de hipótesis es utilizar la prueba no paramétrica de Mann-Whitney, debido a que el tema de tesis expuesto es un estudio transversal entre dos grupos (CakePHP y Laravel) y al realizar inferencias estadísticas, se acostumbra adoptar un modelo de decisión determinando diferencias significativas por medio de hipótesis alterna y nula.

La estadística descriptiva se utilizará para recopilar, organizar, presentar, analizar e interpretar los datos correspondientes a los indicadores del índice de productividad física de manera tal que describa fácil y rápidamente las características esenciales de dicha información.

La estadística inferencial será utilizada para la toma de decisión en base a los resultados del índice de productividad física, probando de esta manera si se cumple la hipótesis planteada para la presente investigación.

CAPITULO III

RESULTADOS, DISCUSIÓN Y ANÁLISIS DE RESULTADOS

* 1. Estudio comparativo

A continuación se realizará el estudio comparativo para medir la productividad de desarrollo entre los frameworks CakePHP y Laravel por medio del índice de productividad física y sus indicadores dentro de un mismo escenario de pruebas.

Para el cálculo del índice de productividad física primero se realizará la toma de datos de los indicadores correspondientes a esta relación para los frameworks CakePHP y Laravel respectivamente. Una vez obtenido los datos, se procederá a realizar el cálculo del índice de productividad física.

* + 1. Índice de Productividad Física (IPF).

Para calcular el índice de productividad física se realiza una relación entre el indicador de líneas de código y tiempo empleado(Card, 2006, p.7). Por medio del estándar IEEE 1042-1992 se miden las primitivas o indicadores que dan resultado a este índice. La ventaja de los valores obtenidos de la productividad depende de la exactitud de los datos utilizados para calcularlos, por medio de la siguiente formula:

**Ecuación N°3**

Donde:

**IPF**= Índice de Productividad Física

**LDC**=Líneas de Código

**TE**= Tiempo Empleado

El procesos para proceder al cálculo de este índice será: primero la medición de los indicadores y su correspondiente descripción estadística. Luego se tomara la información para ser calculados por medio de la Ecuación N°3. Y así proceder a la prueba de hipótesis por medio de la prueba U de Mann-Whitney.

**Indicador 1: Líneas de código (LDC).**

Este indicador mide directamente la cantidad de líneas de código ejecutables no comentadas del prototipo de prueba de CakePHP y Laravel. Estas se encuentran dentro de las carpetas correspondientes a la arquitectura MVC. El número de líneas de código, se lo obtuvo por medio del programa Understand de SciTools (Ver Anexo A2).

Los resultados obtenidos se muestran a continuación.

Tabla 3-1: Resultados obtenidos líneas de código CakePHP

|  |  |
| --- | --- |
| **Archivo** | **# LDC** |
| /app/Model/Group.php | 15 |
| /app/Model/Role.php | 15 |
| /app/Model/User.php | 41 |
| /app/View/groups/index.ctp | 480 |
| /app/View/roles/index.ctp | 474 |
| /app/View/users/index.ctp | 511 |
| /app/View/users/login.ctp | 37 |
| /app/View/Pages/home.ctp | 319 |
| /app/Controller/GroupsController.php | 44 |
| /app/Controller/RolesController.php | 44 |
| /app/Controller/UsersController.php | 73 |

**Fuente**: María José Samaniego

Tabla 3-2: Resultados obtenidos líneas de código Laravel

|  |  |
| --- | --- |
| **Archivo** | **#LDC** |
| /app/models/Group.php | 9 |
| /app/models/Role.php | 9 |
| /app/models/User.php | 6 |
| /app/views/groups/index.blade.php | 513 |
| /app/views/roles/index.blade.php | 513 |
| /app/views/users/index.blade.php | 556 |
| /app/views/users/login.blade.php | 62 |
| /app/views/home.blade.php | 323 |
| /app/controllers/GroupsController.php | 41 |
| /app/controllers/RolesController.php | 41 |
| /app/controllers/UsersController.php | 52 |

**Fuente**: María José Samaniego

La Tabla 3-3 contiene la estadística descriptiva cuyos datos son:

Tabla 3-3: Estadística descriptiva líneas de código

|  |  |  |
| --- | --- | --- |
|  | **CakePHP** | **Laravel** |
| Media | 186,636 | 193,182 |
| Error típico | 66,895 | 73,443 |
| Mediana | 44 | 52 |
| Moda | 15 | 9 |
| Desviación estándar | 211,541 | 232,250 |
| Varianza de la muestra | 44749,455 | 53939,964 |
| Rango | 496 | 550 |
| Mínimo | 15 | 6 |
| Máximo | 511 | 556 |
| Suma | 2053 | 2125 |
| Cuenta | 11 | 11 |

**Fuente:** María José Samaniego

Para CakePHP se tiene:

El número de líneas de código promedio con este framework es de 186,636, es decir que tiene tendencia a variar por debajo o por encima de dicho número con 66,895.

El 50% de número de líneas de código no sobrepasa el valor de 44 correspondiente a los archivos a continuación:

* /app/models/Group.php
* /app/models/Role.php
* /app/views/users/login.php
* /app/models/User.php
* /app/controllers/GroupsController.php

Mientras que el otro 50% sobrepasa las 44 líneas de código siendo estos los archivos:

* /app/controllers/UsersController.php
* /app/views/home.php
* /app/views/roles/index.php
* /app/views/groups/index.php
* /app/views/users/index.php

La cantidad de líneas de código que más se repite es igual a 15 LDC, correspondiente a los archivos /app/models/Group.php, /app/models/Role.php. Existe una mayor dispersión de datos con respecto al promedio de líneas de código con 211,541 de diferencia, debido a que existen valores extremos de un archivo a otro entre 15 y 511, es decir que los datos no están distribuidos normalmente.

Para Laravel se tiene:

El número de líneas de código promedio con este framework es de 193,182, es decir que tiene tendencia a variar por debajo o por encima de dicho número con 73,443.

El 50% de número de líneas de código no sobrepasa el valor de 53 correspondiente a los archivos a continuación:

* /app/models/User.php
* /app/models/Group.php
* /app/models/Role.php
* /app/controllers/GroupsController.php
* /app/controllers/RolesController.php

Mientras que el otro 50% sobrepasa las 52 líneas de código siendo estos los archivos:

* /app/views/users/login.blade.php
* /app/views/home.blade.php
* /app/views/groups/index.blade.php
* /app/views/roles/index.blade.php
* /app/views/users/index.blade.php

La cantidad de líneas de código que más se repite es igual a 9 líneas de código correspondiente a los archivos /app/models/Group.php, /app/models/Role.php. Existe una mayor dispersión de datos con respecto al promedio de líneas de código con 232,250 de diferencia, debido a que existen valores extremos de un archivo a otro entre 6 y 556.

La cantidad promedio de líneas de código para CakePHP es significativamente mayor al de Laravel, con una mínima diferencia de aproximadamente 7 LDC.

Para el análisis estadístico de este indicador se tiene:

**Ho**= No existe diferencia significativa entre el número de líneas de código de CakePHP con el número de líneas de código de Laravel.

**H1**= Existe diferencia significativa entre el número de líneas de código de CakePHP con el número de líneas de código de Laravel.

En la Tabla 4-3 se muestra los datos obtenidos de la prueba U de Mann-Whitney:

Tabla 4-3: Estadísticos de contrastes Líneas de código

|  |  |
| --- | --- |
|  | LDC |
| U de Mann-Whitney | 60.000 |
| W de Wilcoxon | 126.000 |
| Z | -.033 |
| Sig. Asintót. (bilateral) | .974 |

**Fuente:** María José Samaniego

El P-Valor para la prueba U de Mann-Whitney como se observa en la Tabla 10-3 es de 0.974, por lo tanto se deduce que el valor de significancia asintótica es mayor al valor alfa (0.974>0.05), por lo tanto se acepta la hipótesis nula y se rechaza la hipótesis alternativa, es decir que no existe una diferencia significativa entre las LDC de Cakephp con respecto a Laravel.

**Indicador 2: Tiempo empleado (TE).**

Este indicador mide directamente el tiempo empleado en horas para la producción de las líneas de código por cada archivo de la muestra. La Tabla 5- y Tabla 6-3 contiene los datos resultantes del prototipo de prueba para este indicador (Ver Anexo A3).

Tabla 5-3: Resultados indicador tiempo empleado CakePHP

|  |  |
| --- | --- |
| **Archivo** | **# Horas** |
| /app/Model/Group.php | 2,32 |
| /app/Model/Role.php | 0,99 |
| /app/Model/User.php | 0,92 |
| /app/View/groups/index.ctp | 1,95 |
| /app/View/roles/index.ctp | 2,63 |
| /app/View/users/index.ctp | 5,58 |
| /app/View/users/login.ctp | 1,27 |
| /app/View/Pages/home.ctp | 2,97 |
| /app/Controller/GroupsController.php | 1,96 |
| /app/Controller/UsersController.php | 0,6 |
| /app/Controller/RolesController.php | 0,28 |

**Fuente:** María José Samaniego

Tabla 6-3: Resultados indicador tiempo empleado Laravel

|  |  |
| --- | --- |
| **Archivo** | **# Horas** |
| /app/models/Group.php | 3,55 |
| /app/models/Role.php | 2,17 |
| /app/models/User.php | 0,84 |
| /app/views/groups/index.blade.php | 8,61 |
| /app/views/roles/index.blade.php | 4,07 |
| /app/views/users/index.blade.php | 7,82 |
| /app/views/users/login.blade.php | 4,48 |
| /app/views/home.blade.php | 3,59 |
| /app/controllers/GroupsController.php | 2,78 |
| /app/controllers/RolesController.php | 2,28 |
| /app/controllers/UsersController.php | 1,98 |

**Fuente:** María José Samaniego

La Tabla 7-3 contiene la estadística descriptiva realizada con los datos obtenidos en la muestra de 11 archivos.

Tabla 7-3: Estadística descriptiva tiempo empleado

|  |  |  |
| --- | --- | --- |
|  | **CakePHP** | **Laravel** |
| Media | 1.952 | 3.879 |
| Error típico | 0.446 | 0.718 |
| Mediana | 1.950 | 3.550 |
| Moda | NA | 2.780 |
| Desviación estándar | 1.479 | 2.381 |
| Varianza de la muestra | 2.187 | 5.668 |
| Rango | 5.300 | 7.770 |
| Mínimo | 0.280 | 0.840 |
| Máximo | 5.580 | 8.610 |
| Suma | 21.470 | 42.670 |
| Cuenta | 11,000 | 11,000 |

**Fuente:** María José Samaniego

Para CakePHP se tiene:

La cantidad promedio de tiempo empleado para la programación con este framework es de 1,952, con la tendencia a variar por debajo o por encima de dicho número con 0,446.

El 50% de Tiempo empleado no sobrepasa las 1,950 horas correspondientes a los archivos a continuación:

* /app/controllers/RolesController.php
* /app/controllers/UsersController.php
* /app/models/User.php
* /app/models/Role.php
* /app/views/users/login.php

Mientras que el otro 50% sobrepasa las 1,950 horas:

* / app/controllers/GroupsController.php
* /app/models/Group.php
* /app/views/roles/index.php
* /app/views/home.php
* /app/views/users/index.php

El tiempo empleado en la programación es diferente en cada archivo de muestra. Los datos de este indicador se encuentran más concentrados ya que su desviación estándar es menor, en un rango entre 0.280 y 5.580.

Para Laravel se tiene:

La cantidad promedio de Tiempo empleado para la programación con este framework es de 3.879, con la tendencia a variar por debajo o por encima de dicho número con 0.718.

El 50% de Tiempo empleado no sobrepasa las 3,550 horas correspondientes a los archivos a continuación:

* /app/models/User.php
* /app/controllers/UsersController.php
* /app/models/Role.php
* /app/controllers/GroupsController.php
* /app/controllers/RolesController.php

Mientras que el otro 50% sobrepasa las 3,550 horas:

* /app/views/home.blade.php
* /app/views/roles/index.blade.php
* /app/views/users/login.blade.php
* /app/views/users/index.blade.php
* /app/views/groups/index.blade.php

El Tiempo empleado correspondiente a 2.780 horas es para los archivos /app/controllers/GroupsController.php y /app/controllers/RolesController.php debido a las exactas funcionalidades que contienen este modelo. Los datos de este indicador se encuentran más concentrados ya que su desviación estándar es menor, en un rango entre 0.840 y 8.610.

Para el análisis estadístico de este indicador se tiene:

**Ho**= No existe diferencia significativa entre el tiempo empleado en CakePHP con respecto al tiempo empleado en Laravel.

**H1**= Existe diferencia significativa entre el tiempo empleado en CakePHP con respecto al tiempo empleado en Laravel.

En la Tabla 8-3 se muestra los datos obtenidos de la prueba U de Mann-Whitney:

Tabla 8-3: Estadísticos de contrastes Tiempo empleado

|  |  |
| --- | --- |
|  | TE |
| U de Mann-Whitney | 25.000 |
| W de Wilcoxon | 91.000 |
| Z | -2.332 |
| Sig. Asintót. (bilateral) | .020 |

**Fuente:** María José Samaniego

El P-Valor para la prueba U de Mann-Whitney como se observa en la Tabla 8-3 es de 0.020, por lo tanto se deduce que el valor de significancia asintótica es menor al valor alfa (0.020<0.05), por lo tanto se acepta la hipótesis alternativa y se rechaza la hipótesis nula, es decir que si existe una diferencia significativa entre el tiempo empleado en desarrollo de Cakephp con respecto a Laravel.

En la Tabla 9-3 y Tabla 10-3 se muestran los resultados obtenidos para el índice de productividad física.

Tabla 9-3: Resultados Índice Productividad Física CakePHP

|  |  |  |  |
| --- | --- | --- | --- |
| **Archivo** | **LDC** | **TE** | **IPF** |
| /app/Model/Group.php | 15 | 2,32 | 6,465 |
| /app/Model/Role.php | 15 | 0,99 | 15,151 |
| /app/Model/User.php | 41 | 0,92 | 44,565 |
| /app/View/groups/index.ctp | 480 | 1,95 | 246,153 |
| /app/View/roles/index.ctp | 474 | 2,63 | 180,228 |
| /app/View/users/index.ctp | 511 | 5,58 | 91,577 |
| /app/View/users/login.ctp | 37 | 1,27 | 29,133 |
| /app/View/Pages/home.ctp | 319 | 2,97 | 107,407 |
| /app/Controller/GroupsController.php | 44 | 1,96 | 22,448 |
| /app/Controller/RolesController.php | 44 | 0.28 | 157,142 |
| /app/Controller/UsersController.php | 73 | 0,6 | 121,666 |

**Fuente:** María José Samaniego

Tabla 10-3: Resultados Índice Productividad Física Laravel

|  |  |  |  |
| --- | --- | --- | --- |
| **Archivo** | **LDC** | **TE** | **IPF** |
| /app/models/Group.php | 9 | 3,55 | 2,535 |
| /app/models/Role.php | 9 | 2,17 | 4,147 |
| /app/models/User.php | 6 | 0,84 | 7,142 |
| /app/views/groups/index.blade.php | 513 | 8,61 | 59,581 |
| /app/views/roles/index.blade.php | 513 | 4,07 | 126,044 |
| /app/views/users/index.blade.php | 556 | 7,82 | 71,099 |
| /app/views/users/login.blade.php | 62 | 4,48 | 13,839 |
| /app/views/home.blade.php | 323 | 3,59 | 89,972 |
| /app/controllers/GroupsController.php | 41 | 2,78 | 14,748 |
| /app/controllers/RolesController.php | 41 | 2,28 | 14,748 |
| /app/controllers/UsersController.php | 52 | 1,98 | 26,262 |

**Fuente:** María José Samaniego

Para el análisis estadístico de este índice se tiene:

**Ho**= No existe diferencia significativa entre el índice de productividad física de CakePHP con el índice de productividad física de Laravel.

**H1**= Existe diferencia significativa entre el índice de productividad física de CakePHP con el índice de productividad física de Laravel.

La Tabla 11-3 contiene la estadística descriptiva realizada por el software SPSS 20.

Tabla 11-3: Estadística descriptiva índice de productividad física

|  |  |  |
| --- | --- | --- |
|  | **CakePHP** | **Laravel** |
| Media | 92.903 | 39.102 |
| Error típico | 23.580 | 12.490 |
| Mediana | 91.577 | 14.748 |
| Moda | NA | 14.748 |
| Desviación estándar | 78.206 | 41.424 |
| Varianza de la muestra | 6116.129 | 1715.962 |
| Rango | 239.688 | 123.509 |
| Mínimo | 6.465 | 2.535 |
| Máximo | 246.153 | 126.044 |
| Suma | 1021.935 | 430.117 |
| Cuenta | 11.000 | 11,000 |

**Fuente**: María José Samaniego

Para CakePHP se tiene:

El promedio de índice de productividad física es de 92.903, con la tendencia a variar por debajo o por encima de dicho número con 23.580.

El 50% de archivos no sobrepasa un índice de productividad física de 91.577 siendo estos:

* /app/Model/Group.php
* /app/Model/Role.php
* /app/Controller/GroupsController.php
* /app/View/users/login.ctp
* /app/Model/User.php

Mientras que el otro 50% sobrepasa el 91.577:

* /app/View/Pages/home.ctp
* /app/Controller/UsersController.php
* /app/Controller/RolesController.php
* /app/View/roles/index.ctp
* /app/View/groups/index.ctp

Los índices de productividad física obtenidos por cada elemento de la muestra se encuentran más dispersos ya que su desviación estándar es menor a la media obtenida, los datos se encuentran en un rango entre 6,465 y 246,153.

Para Laravel se tiene:

La índice de productividad física con este framework es de 39.102, con la tendencia a variar por debajo o por encima de dicho número con 12.490.

El 50% de archivos no sobrepasa un índice de productividad física de 14,748 siendo estos:

* /app/models/Group.php
* /app/models/Role.php
* /app/models/User.php
* /app/views/users/login.blade.php
* /app/controllers/RolesController.php

Mientras que el otro 50% sobrepasa el índice de 14,748:

* /app/controllers/UsersController.php
* /app/views/groups/index.blade.php
* /app/views/users/index.blade.php
* /app/views/home.blade.php
* /app/views/roles/index.blade.php

Los índices de productividad física son distintos en los 11 archivos, ya que varían en tamaño y esfuerzo de acuerdo a que funciones cumple cada uno de ellos. Estos índices se encuentran más concentrados con respecto al promedio ya que su desviación estándar es mayor, en un rango entre 2.535 y 126.044 líneas de código por tiempo empleado.

A continuación en la Tabla 12-3se muestra los datos obtenidos de la prueba U de Mann-Whitney por SPSS 20:

Tabla 12-3: Estadísticos de contrastes Índice de Productividad Física

|  |  |
| --- | --- |
|  | IPF |
| U de Mann-Whitney | 30.000 |
| W de Wilcoxon | 96.000 |
| Z | -2.003 |
| Sig. Asintót. (bilateral) | .045 |

**Fuente:** María José Samaniego

Mediante la prueba no paramétrica U de Mann-Whitney se rechaza la hipótesis nula, es decir que si existe una diferencia significativa entre el índice de productividad física entre CakePHP y Laravel, debido a que el P-valor es menor al de alpha (0,045<0.05). El índice de productividad física de CakePHP es diferente al de Laravel, todo esto establecido con un nivel de confianza del 95%.

* 1. Resultado del parámetro de productividad

Para la obtención de resultados se procederá a la calificación del parámetro de productividad, a través del indicador de productividad física. Y de esta mónera proceder a probar la hipótesis establecida en esta investigación.

* + 1. Evaluación de resultados del indicador líneas de código (LDC).

Por medio de la conclusión estadística obtenida a través de la prueba U de Mann-Whitney de la Tabla 4-3 con respecto al indicador de líneas de código entre CakePHP y Laravel, no existe una diferencia significativa.

Figura 1-3. Promedios líneas de código para CakePHP y Laravel

**Fuente:** Maria Jose Samaniego

**Interpretación de Resultados**

Debido a que este valor es inversamente proporcional se concluye que a menor promedio de líneas de código, mayor será el resultado del índice de productividad, pero estadísticamente no existe una diferencia debido a que el tamaño de la muestra es pequeña. La figura a continuación muestra los promedios obtenidos en la muestra de 11 archivos.

* + 1. Evaluación de resultados del indicador tiempo empleado (TE).

Por medio de la conclusión estadística obtenida a través de la prueba U de Mann-Whitney de la Tabla 8-3 con respecto al indicador de tiempo empleado (horas) entre CakePHP y Laravel, si existe una diferencia significativa.

La Figura 2-3 muestra la diferencia de promedios obtenidos por cada framework.

Figura 2-3. Promedios tiempo empleado (horas) para CakePHP y Laravel

**Fuente:** Maria Jose Samaniego

**Interpretación de Resultados**

Debido a que este valor es inversamente proporcional se concluye que a menor promedio de líneas de código, mayor será el resultado del índice de productividad. De tal forma que el tiempo promedio empleado para programar con CakePHP es menor con respecto a Laravel con una diferencia de 1.927 horas.

* + 1. Evaluación de resultados del índice de productividad física (IPF).

Por medio de la conclusión estadística obtenida a través de la prueba U de Mann-Whitney de la Tabla 12-3 con respecto al índice de productividad física entre CakePHP y Laravel, si existe una diferencia significativa.

La Figura 3-3 muestra la diferencia de promedios obtenidos por cada framework.

Figura 3-3. Promedios índice de productividad física CakePHP y Laravel

**Fuente:** Maria Jose Samaniego

**Interpretación de los Resultados**

Debido a que este valor es obtenido por medio de la relación entre las líneas de código y el tiempo empleado si existe una diferencia significativa entre CakePHP con 92,903 LDC/TE y Laravel con 39.102 LDC/TE.

Como se puede observar que a menores líneas de código escritas por el programador en menor tiempo, la productividad de este se verá en aumento debido a la utilización del framework que ofrece estas ventajas.

* 1. **Demostración de la hipótesis**

Para el planteamiento de la hipótesis se tiene:

**Ho:** La productividad del Framework CakePHP no es superior a Laravel, en el desarrollo del Sistema de Seguimiento de Incidentes de la Infraestructura de Red en la ESPOCH.

**H1:** La productividad del Framework CakePHP es superior a Laravel, en el desarrollo del Sistema de Seguimiento de Incidentes de la Infraestructura de Red en la ESPOCH.

µc = productividad de CakePHP

µl = productividad de Laravel

Ho: µc < µl

H1: µc >= µl

Donde la variable dependiente de esta investigación es:

* Productividad

Y las variables independientes son:

* CakePHP
* Laravel

En base a los resultados de la Figura 3-3 se procede a demostrar la hipótesis planteada en esta investigación. Debido que el índice de productividad es un valor promediado de líneas de código escritas por el desarrollador y el tiempo empleado para su desarrollo, el framework CakePHP ha obtenido un mejor resultado con un promedio de 92,903 LDC/TE frente a Laravel, por lo cual se procede a aceptar la hipótesis alternativa haciendo uso de la estadística inferencial.

CAPITULO IV

SISTEMA DE SEGUIMIENTO DE INCIDENTES DE LA INFRAESTRUCTURA DE RED EN LA ESPOCH

* 1. Introducción

En la Escuela Superior Politécnica de Chimborazo ocurren diariamente incidencias que causan interrupción en el servicio. Al no existir un sistema informático que permita automatizar la gestión y seguimiento de incidentes se desea crear una aplicación web que optimice el servicio de atención al cliente.

La Dirección de Tecnologías de la Información y Comunicación tiene como objetivos brindar asistencia y mantenimiento a los sistemas informáticos administrativos, académicos, y de la organización, como también proporcionar servicios de mantenimiento de Hardware, Software y redes computacionales. Dicha asistencia involucra el seguimiento de los incidentes que ocurren dentro de la ESPOCH, los cuales son reportados por llamadas telefónicas, correo electrónico o a su vez personalmente en la misma dirección.

La información reportada es almacenada en documentos en la nube con GoogleDrive permitiendo que sea expuesta a la inconsistencia y redundancia de esta, dificultando su acceso de manera rápida por medio del personal de DTIC y además de que no exista un seguimiento.

Por esta razón se desea crear un sistema web que permita proveer soporte y asistencia a sus clientes o empleados por medio de un seguimiento de incidentes con el fin de mejorar su servicio de atención y regularizar el volumen de trabajo del personal de soporte. Se ha escogido la Metodología SCRUM para el desarrollo del Sistema de seguimiento de incidentes de la infraestructura de red en la ESPOCH.

A continuación se detallará la fase de análisis con los diferentes requisitos de software, que debe cumplir el Sistema de Seguimiento de Incidentes para el área de Infraestructura de red de la Dirección de Tecnologías de la Información y Comunicación de la Escuela Superior Politécnica de Chimborazo; además de definir los límites del mismo.

* 1. Información general

La Escuela Superior Politécnica de Chimborazo, en el mes de julio del 2003 aprobó mediante resolución del H. Consejo Politécnico la reestructuración orgánica funcional de la institución, misma que involucró a las diferentes dependencias administrativas y académicas con la finalidad de lograr una administración moderna y eficiente en sus diferentes ámbitos(Escuela Superior Politécnica de Chimborazo, s.f.).

Este cambio determino que las tareas académicas encargadas al Departamento de Cómputo y Sistemas se vinculen directamente a las diferentes facultades y las funciones técnicas de asesoría, desarrollo de soluciones tecnológicas en el área informática se integren en el Departamento de Sistemas y Telemática mismas que se encontraban divididas en el comité Informático y el Departamento de Cómputo y Sistemas.

A partir del mes de septiembre del 2013 según el nuevo Estatuto Politécnico 2013, aprobado por el Consejo de Educación Superior CES, mediante Resolución RPC-SO-278-2013 cambia su nominación oficial a: Dirección de Tecnologías de la Información y Comunicación (DTIC comúnmente), la cual es ampliamente aceptada y utilizada en instituciones públicas y privadas a nivel mundial(Escuela Superior Politécnica de Chimborazo, s.f.). En la Tabla 1-4se puede observar la información principal de la Institución como la dirección y sus teléfonos de contacto.

Tabla 1-4: Información de contacto

|  |  |  |  |
| --- | --- | --- | --- |
| **F:\Politecnica\wwwwqqt (1).jpgC:\Users\Majos\Desktop\Logo_DiTIC_77548.pngESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO**  **DIRECCIÓN DE TECNOLOGÍAS DE LA INFORMACIÓN Y COMUNICACIÓN** | | | |
| **Director** | **Datos de Contacto** | | |
| **Dirección** | **Página Web** | **Teléfono** |
| Ing. Daniel Haro Mendoza | Panamericana Sur Km. 1.5 | http://dtic.espoch.edu.ec  dtic@espoch.edu.ec | +593 (3) 2998 200 ext. 271 |

**Fuente:** María José Samaniego

El organigrama estructural de DTIC se detalla en la Figura 1-5 a continuación:

Figura 1-4: Organigrama estructura DTIC

**Fuente:** María Jose Samaniego

Correspondiente al nivel directivo está la Dirección DTIC, al nivel asesor la secretaría, y al nivel operativo las áreas de desarrollo, infraestructuras y redes, y soporte y mantenimiento.

* 1. Definición del ámbito

El Sistema de Seguimiento de Incidentes tiene como objetivo la administración, el tratamiento y seguimiento de incidentes que son reportados diariamente en la ESPOCH. De esta manera se necesita crear una solución tecnológica por medio de una aplicación web accesible por la intranet de la ESPOCH, que permitirá automatizar el proceso de gestión de incidentes que causen una interrupción en el servicio que brinda DTIC de la manera más rápida y eficaz posible.

Con la creación de usuarios se desea asignar un rol y un grupo de trabajo que permita ejecutar las tareas asignadas respectivamente, tales como administración de la gestión de usuarios, de información de los tickets y seguimiento de los mismos. También se generará reportes que facilitarán al Director de DTIC, ver la cantidad de trabajo asignado a cada técnico, y poder de esta manera tener una mejor distribución de tareas entre el personal técnico.

Los roles SCRUM implicados para el desarrollo del Sistema de Seguimiento de Incidentes DTIC-ESPOCH serán los siguientes:

* **Product Owner:** Escuela Superior Politécnica de Chimborazo, responsable de obtener el mayor valor de producto para los usuarios y resto de implicados.
* **Equipo de Desarrollo:** María José Samaniego Larrea, encargada del desarrollo del producto.
* **Scrum Manager:** Ing. Jorge Menéndez, gestor del equipo, es responsable del funcionamiento y de la productividad del equipo de desarrollo.
  1. Estudio de factibilidad

Después de definir la problemática presente y causas que ameritan el desarrollo del nuevo sistema, es pertinente realizar un estudio de factibilidad para determinar la capacidad técnica e infraestructura tecnológica que implica para la implantación del SDSI-DTIC. Los aspectos tomados en cuenta para este estudio de factibilidad fueron clasificados en dos áreas, las cuales se describen a continuación:

Factibilidad Técnica

La factibilidad técnica consistió en un estudio de las tecnologías existentes en la DTIC, permitiendo evaluar la posibilidad de uso de los mismos para la implantación y además de la tecnología necesaria para el desarrollo del sistema. Para esto se evaluó dos enfoques: Recurso de Hardware y de Software

1. **Recurso de Hardware**

En cuanto a Hardware, específicamente el servidor donde debe estar instalado el sistema cuenta con las siguientes características:

**Servidor Físico**: Marca IBM, Modelo: x3200 M2, Procesador Intel Xeon E3110 32 3 GHz, Memoria RAM: 1.79 GB RAM

Lo que respecta a Hardware para el desarrollo del sistema, las características son las siguientes:

**Computadora portátil**: Toshiba AMD Athlon(tm) P320 Dual-Core Processor 2.10 GHz, Memoria RAM de 4GB, 250 GB en disco duro.

Evaluando el Hardware existente, la Dirección de Tecnologías de Información y Comunicación no requirió realizar ninguna inversión, ya que cumplen con los requisitos mínimos para la implantación y desarrollo.

1. **Recurso de Software**

En cuanto a software, la Dirección cuenta con todas las tecnologías para el desarrollo del sistema, lo cual no amerita inversión alguna.

* Servidor Base de Datos: MySQL 5.1.73 Source distribution.
* Servidor Web: Apache 2.0
* PHP: Versión 5.3.3

Para el desarrollo se requiere las siguientes tecnologías:

* Sistema Operativo: Microsoft Windows 7 / Ultímate 32 bit
* Lenguaje de Programación: PHP 5.5.12
* Tecnología de desarrollo: CakePHP 2.4.2
* IDE: NetBeans 7.3
* Servidor Web: Apache 2.4.9
* Gestor de Base de Datos: MySQL 5.6.17
* DomPDF para reportes.
* Bootstrap 3.2.0 para el front-end de la aplicación.
* Font Awesome 4.2.0 para iconos de las vistas.

Las estaciones de trabajo cuentan con ambiente Windows, mientras que el servidor cuenta con el sistema operativo Linux. Como resultado de este estudio técnico se concluyó que en la actualidad la Dirección de Tecnologías de Información y Comunicación cuenta con toda la infraestructura tecnológica tal como hardware y software necesario para el desarrollo e implantación del mismo para su funcionamiento.

Factibilidad Operativa

Para un mejor alcance como Dirección se desarrollará una sistema web lo más amigable posible con los usuarios, de tal forma que sin mucha dificultad este pueda adaptarse y aprovechar al máximo las facilidades que este brinde, traduciéndose a una herramienta de fácil manejo y comprensión, tanto como las interfaces y reportes.

El SDSI-DTIC que se desarrollará, está destinado a ser operado por el personal técnico y administrativo de DTIC, así como de otras dependencias y facultades dentro de la ESPOCH, haciendo de forma más fácil el reporte y seguimiento de incidentes.

* 1. Características de usuario

El sistema cuenta con seis tipos de usuarios:

* **Administrador:** persona que se encarga de la creación de usuarios, contraseñas, y asignar permisos conforme lo crea conveniente. Además es el encargado de la administración del sistema por medio del panel de control.
* **Director DTIC:** persona encargada de DTIC que recepta el ticket o caso de parte de alguna dependencia o facultad de la ESPOCH y lo delega al jefe de área correspondiente.
* **Director Dependencia:** Persona encargada de generar tickets, de acuerdo a los incidentes ocasionados en esa dependencia, enviándolos a la Dirección DTIC.
* **Jefe de Área:** persona responsable de cada grupo o área dentro de DTIC que se encargan de generar o asignar los tickets a los técnicos DTIC.
* **Técnico DTIC:** persona encargada de la gestión del caso por medio del ticket abierto.
* **Técnico Facultad:** Persona encargada de generar tickets, de acuerdo a los incidentes ocasionados en su respectiva facultad. Permitiendo asignarse el caso, o enviar directamente al área correspondiente DTIC.
  1. Definición del Product Backlog

El Product Backlog listará todos los requisitos o funcionalidades de manera  priorizada, las cuales serán necesarias para el desarrollo exitoso del SDSI-DTIC. Siendo el único recurso de requerimientos que pueden cambiar para la realización del producto. El Product Owner, será el responsable de la lista y de su contenido, tanto como su prioridad y orden.

Los requisitos específicos para el funcionamiento del SDSI-DTIC, están representados en forma de historias de usuario o técnicas en el Product Backlog en la Tabla 2-4.

Cada historia contiene un identificador compuesto por las abreviaturas *“HU”* corresponden a una historia de usuario y *“HT”* a historias técnicas seguido de un número según el orden; también contiene el enunciado de la historia descrita usando la terminología del cliente, la prioridad de cada una de ellas entre Muy Alto, Alto, Normal, Baja.

Por último la estimación de puntos de historia utilizando la técnica *T-Shirt Sizing,* con las habituales tallas de las camisetas: pequeña (S) igual a 2 puntos, mediana (M) igual a 3 puntos, grande (L) con 5 puntos y muy grande (XL) con 8 puntos, esta última medida no es recomendable usarla ya que equivale a toda la duración del sprint por una historia de usuario.

Tabla 2-4: Product Backlog proyecto SDSI-DTIC

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Nombre de la Historia** | **Prioridad** | **Puntos** |
| HT1 | Diseño del esquema de la Base de Datos | Muy Alta | 5 |
| HT2 | Diseño de la Arquitectura del SDSI-DTIC | Muy Alta | 5 |
| HT3 | Diseño de las Interfaces de Usuario | Muy Alta | 5 |
| HT4 | Codificación del sistema. |  |  |
| HT5 | Construcción del Product Backlog | Muy Alta | 5 |
| HU1 | Autentificación de usuarios | Muy Alta | 3 |
| HU2 | Registro y Visualización de usuarios | Muy Alta | 3 |
| HU3 | Modificación de usuarios | Alta | 3 |
| HU4 | Validar usuarios | Alta | 3 |
| HU5 | Modificación de información de los roles de trabajo de los usuarios | Alta | 3 |
| HU6 | Validar roles de trabajo de los usuarios. | Alta | 3 |
| HU7 | Modificación la información de los grupos o áreas de trabajo. | Alta | 3 |
| HU8 | Validez a los grupos o áreas de trabajo. | Alta | 3 |
| HU9 | Registro y Visualización de información de las facultades o dependencias existentes en la ESPOCH | Alta | 3 |
| HU10 | Modificación de información de las facultades o dependencias | Alta | 3 |
| HU11 | Eliminar a una facultad o dependencia de la ESPOCH | Alta | 3 |
| HU12 | Registro y Visualización de un nuevo tipo de ticket | Alta | 3 |
| HU13 | Modificación de información de un tipo de ticket | Alta | 3 |
| HU14 | Visualización de la información correspondiente a los estados posibles que tendrá un ticket. | Alta | 3 |
| HU15 | Registro y Visualización de una nueva prioridad de ticket. | Alta | 3 |
| HU16 | Modificación de la información de una prioridad de ticket | Alta | 3 |
| HU17 | Visualización de la lista de todos los tickets asignados. | Alta | 3 |
| HU18 | Generar un ticket o caso. | Alta | 5 |
| HU19 | Visualización del ticket generado | Alta | 2 (continuará) |
| HU20 | Atención de un ticket. | Alta | (continuación)5 |
| HU21 | Transferir un ticket. | Alta | 2 |
| HU22 | Asignación de un ticket. | Alta | 3 |
| HU23 | Transferir un ticket | Alta | 3 |
| HU24 | Creación de bitácora de ticket. | Alta | 3 |
| HU25 | Cambio de estado del ticket. | Alta | 2 |
| HU26 | Búsqueda de tickets. | Alta | 2 |
| HU27 | Visualización de la lista de todos los tickets asignados. | Normal | 2 |
| HU28 | Reporte con la lista de todos tickets creados. | Normal | 3 |
| HU29 | Reporte con la lista de todos tickets nuevos. | Normal | 3 |
| HU30 | Reporte con la lista de todos tickets abiertos. | Normal | 3 |
| HU31 | Reporte con la lista de todos tickets cerrados. | Normal | 3 |
| HU32 | Reporte con la lista de tickets por área DTIC. | Normal | 3 |
| HU33 | Reporte con la lista de tickets del área. | Normal | 3 |
| HU34 | Reporte con la lista de tickets por Dependencia o Facultad ESPOCH | Normal | 3 |
| HU35 | Reporte de la bitácora del ticket. | Normal | 3 |
| HU36 | Reporte del ticket por técnico. | Normal | 3 |
| HT6 | Elaboración del Manual de Usuario del SDSI-DTIC | Normal | 3 |
| HT7 | Elaboración del Manual de Instalación del SDSI-DTIC | Normal | 3 |
| HT8 | Implantación del Sistema | Normal | 3 |

**Fuente:** María José Samaniego

El Product Backlog contiene en total 44 historias, donde 8 corresponden a las historias técnicas correspondientes al diseño de base de datos, arquitectura, interfaces de usuario del sistema, la construcción del Product Backlog y los manuales e instalación y usuario.

También existen 36 a historias de usuario que definen los requerimientos específicos del SDSI-DTIC. Con un total de 136 puntos estimados en conjunto con todas las historias.

* 1. Planificación Scrum

A continuación se realizará la planificación de todas las historias de usuario y técnicas descritas en el Product Backlog de la Tabla 2-4.

El tiempo estimado de realización de este proyecto es de 4 meses entre las siguientes fechas:

**Fecha de inicio:** 05 de Enero de 2015

**Fecha de Fin:** 01 de Mayo de 2015

La Figura 2-4 presenta la planificación general de los Sprint y su duración.

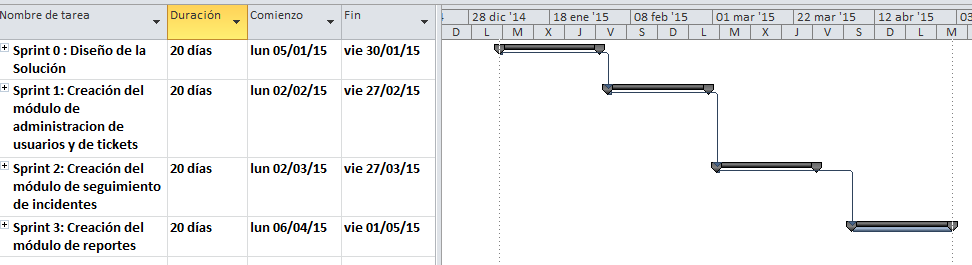


Figura. 2-4. Planificación previa de desarrollo

**Fuente:** María José Samaniego

La duración de todo el proyecto será de 4 meses, asignando 20 días laborables a cada uno de los Sprints.

* + 1. Especificación de actividades del Product Backlog

Para el desarrollo del sistema, se han establecido cuatro productos entregables (sprint), la Tabla 3-4 contiene la lista de todas las historias de usuario y técnicas asignadas a cada uno de los Sprints.

Tabla 3-4: Planificación de historias de usuario y técnicas

|  |  |
| --- | --- |
| **SPRINT 0** | |
| HT1 | Diseño del esquema de la Base de Datos |
| HT2 | Diseño de la Arquitectura del SDSI-DTIC |
| HT3 | Diseño de las Interfaces de Usuario |
| HT4 | Codificación del Sistema |
| HT5 | Construcción del Product Backlog |
| **SPRINT 1** | |
| HU1 | Autentificación de usuarios |
| HU2 | Registro y Visualización de usuarios |
| HU3 | Modificación de usuarios |
| HU4 | Validar usuarios |
| HU5 | Modificación de información de los roles de trabajo de los usuarios |
| HU6 | Validar roles de trabajo de los usuarios. |
| HU7 | Modificación la información de los grupos o áreas de trabajo. |
| HU8 | Validez a los grupos o áreas de trabajo. |
| HU9 | Registro y Visualización de información de las facultades o dependencias existentes en la ESPOCH |
| HU10 | Modificación de información de las facultades o dependencias |
| HU11 | Eliminar a una facultad o dependencia de la ESPOCH |
| HU12 | Registro y Visualización de un nuevo tipo de ticket |
| HU13 | Modificación de información de un tipo de ticket. (continuará) |
| HU14 | (continuación) Visualización de la información correspondiente a los estados posibles que tendrá un ticket. |
| HU15 | Registro y Visualización de una nueva prioridad de ticket. |
| HU16 | Modificación de la información de una prioridad de ticket |
| **SPRINT 2** | |
| HU17 | Visualización de la lista de todos los tickets asignados. |
| HU18 | Generar un ticket o caso. |
| HU19 | Visualización del ticket generado |
| HU20 | Atención de un ticket. |
| HU21 | Transferir un ticket. |
| HU22 | Asignación de un ticket. |
| HU23 | Transferir un ticket |
| HU24 | Creación de bitácora de ticket. |
| HU25 | Cambio de estado del ticket. |
| HU26 | Búsqueda de tickets. |
| **SPRINT 3** | |
| HU27 | Reporte con la lista de todos tickets creados. |
| HU28 | Reporte con la lista de todos tickets nuevos. |
| HU29 | Reporte con la lista de todos tickets abiertos. |
| HU30 | Reporte con la lista de todos tickets cerrados. |
| HU31 | Reporte con la lista de tickets por área DTIC. |
| HU32 | Reporte con la lista de tickets del área. |
| HU33 | Reporte con la lista de tickets por Dependencia o Facultad ESPOCH |
| HU34 | Reporte de la bitácora del ticket. |
| HU35 | Reporte del ticket generado o atendido. |
| HU36 | Reporte de tickets por técnico. |
| HT6 | Elaboración del Manual de Usuario del SDSI-DTIC |
| HT7 | Elaboración del Manual de Instalación del SDSI-DTIC |
| HT8 | Implantación del Sistema |

**Fuente**: María José Samaniego

El Sprint 0, contiene 5 historias técnicas las cuales corresponden al diseño de la solución del Sistema de Seguimiento de Incidentes para la infraestructura de redes de la ESPOCH. El Sprint 1 contiene 16 historias de usuario, que en conjunto forman el módulo de administración de usuarios y de información de los tickets.

El Sprint 2 contiene 11 historias de usuario, que forman el módulo de seguimiento de incidentes.

Y por último el Sprint 3 con 12 de historias, divididas en 3 historias técnicas y 9 historias de usuario para el módulo de reportes del sistema, además de la implantación del sistema en la Dirección de Tecnologías de la Información y Comunicación de la ESPOCH, con sus correspondientes manuales.

A continuación se detalla el primer sprint del sistema, dentro del cual se asignó las historias técnicas y a ellas las tareas de ingeniería que se realizaron en el mismo.

* 1. Sprints

El Sprint Planning meeting realizado el día 05 de enero del 2015 se definieron cada una de las tareas de ingeniería dentro de cada historia técnica de esta iteración para dar comienzo al proyecto. El Sprint Backlog de las 4 iteraciones contiene la información de todas las tareas programadas (Ver Anexo B3).

La Tabla 4-5 detalla la primera historia correspondiente al sprint 1 del sistema formada por: el identificador HU1, donde las primeras siglas identifican con orden ascendente; el usuario que tendrá acceso a esta funcionalidad, un nombre breve y descriptivo, la prioridad del negocio la cual está establecida en el Product Backlog, el riesgo en el desarrollo establecida por el programador, los puntos asignados a la historia, la iteración a la que está asignada, el programador responsable, además de una descripción de lo que se desea realizar. Y las observaciones en el caso de que las hubiera.

Tabla 4-4: Historia técnica diseño de la base de datos

|  |  |
| --- | --- |
| **Historia Técnica** | |
| **Identificador:** HT1 | |
| **Nombre de la Historia:** Diseño del esquema de la Base de Datos | |
| **Prioridad en Negocio:** Muy Alta  (Muy Alta/Alta / Media / Baja) | **Riesgo en Desarrollo:** Alta  (Alta / Media / Baja) |
| **Puntos Estimados:** 5 | **Iteración Asignada:** 0 |
| **Programador Responsable:** María Jose Samaniego Larrea | |
| **Descripción:**  Como equipo de desarrollo se necesita crear el diseño de la base de datos. | |
| **Observaciones:** | |

**Fuente**: María José Samaniego

La Tabla 5-4 detalla la primera tarea de ingeniería correspondiente a la historia de técnica 1, contiene el identificador de la tarea, el número de la historia a la cual pertenece, el tipo de tarea, los puntos estimados, el responsable de la tarea y una breve descripción sobre cómo se llevó a cabo la misma.

Tabla 5-4: Tarea de ingeniería de historia técnica 1

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia Técnica que pertenece:** HT1 |
| **Identificador:** TI1 |
| **Nombre de la Tarea:** Creación del diseño conceptual |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**  Se diseñará la base de datos por medio del diagrama entidad relación notación UML (diagrama de clases de los modelos), tomando en cuenta los requisitos iníciales del cliente. |

**Fuente:** María José Samaniego

La Tabla 6-4 contiene la prueba de aceptación correspondiente a la historia técnica 1, en la cual se detalla el identificador de la prueba, el nombre de la prueba, el identificador de la historia de usuario y tarea a la cual pertenece, el responsable de la prueba, una breve descripción sobre cómo se llevó a cabo la misma, las precondiciones, la ejecución del mismo, las post condiciones y los resultados obtenidos.

Tabla 6-4: Prueba de aceptación historia técnica 1

|  |  |
| --- | --- |
| **Prueba de Aceptación** | |
| **Identificador:** PA1 | |
| **Nombre de la Prueba:** Test de normalización del esquema conceptual | |
| **N° de Historia que pertenece:** HT1 | |
| **Programador Responsable:** María José Samaniego Larrea | |
| **Descripción:**  Se verifica que las tablas creadas en la base de datos, estén normalizadas hasta la tercera forma. | |
| **Precondiciones:** Tener creado el diagrama de entidad relación con notación UML | **Ejecución:**   1. Revisar que no exista asociaciones binarias entre las clases. 2. Revisar que no existan clases asociativas. |
| **Post condiciones:**  Obtención de un diagrama entidad relación con notación UML o diagrama de clases normalizado. | **Evaluación de Prueba:**  Exitosa |

**Fuente:** María José Samaniego

Al culminar las pruebas de aceptación de la historia técnica 1, se obtuvo un total de 2 pruebas las cuales resultaron exitosas con un 100 % de aceptación cumpliendo con los requerimientos del cliente, procediendo a crear la base de datos y sus respectivas tablas.

**Diseño de la Base de Datos**

A partir del diseño conceptual normalizado de la base de datos (Ver Anexo B3), se crea el diseño de la base de datos con las tablas relacionales representado en la Figura 3-4.

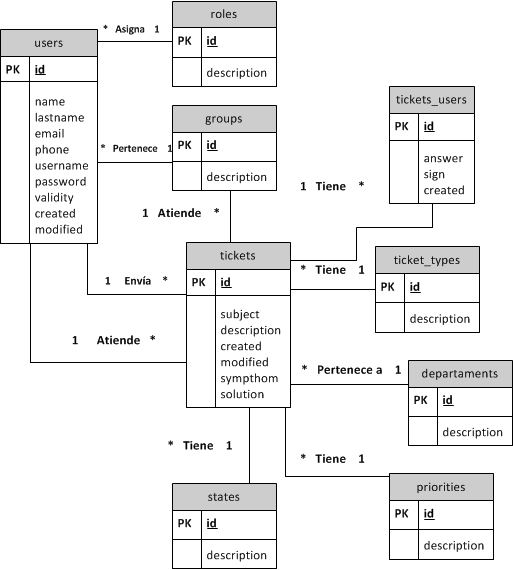


Figura. 3-4. Diagrama Entidad Relación SDSI-DTIC

**Fuente:** María José Samaniego

El SDSI-DTIC contendrá 9 entidades, las cuales corresponden a Roles, Grupos, Usuarios, Tipo de Ticket, Prioridades, Estados, Departamentos, Respuestas, y Tickets, cada una de ellas con su respectiva clave primaria.

Cada entidad será representada como un modelo en el sistema. Todas estas entidades sirven en conjunto para la generación de un ticket por parte de los usuarios registrados para el uso del sistema y a su vez dar un seguimiento del incidente reportado.

La Tabla 7-4 detalla la segunda historia técnica, siguiendo el modelo de la historia técnica 1.

Tabla 7-4: Historia técnica diseño de la arquitectura del sistema

|  |  |
| --- | --- |
| **Historia Técnica** | |
| **Identificador:** HT2 | |
| **Nombre de la Historia:** Diseño de la Arquitectura del Sistema. | |
| **Prioridad en Negocio:** Muy Alta  (Muy Alta/Alta / Media / Baja) | **Riesgo en Desarrollo:** Alta  (Alta / Media / Baja) |
| **Puntos Estimados:** 5 | **Iteración Asignada:** 0 |
| **Programador Responsable:** María José Samaniego Larrea | |
| **Descripción:**  Como equipo de desarrollo se necesita crear el diseño de la arquitectura del SDSI-DTIC. | |
| **Observaciones:** | |

**Fuente:** María José Samaniego

La Tabla 8-4describe la primera tarea de ingeniería correspondiente a la historia de técnica 2, que corresponde al diagrama de despliegue del sistema representando la topología de hardware del sistema sobre el que funcionara el SDSI-DTIC, además del diagrama de componentes representando las tres capas de Modelo Vista y Controlador.

Tabla 8-4: Tarea de ingeniería de historia técnica 2

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia Técnica que pertenece:** HT2 |
| **Identificador:** TI2 |
| **Nombre de la Tarea:** Creación del diagrama de despliegue y componentes. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**  Se creará el diagrama de despliegue y componentes para el sistema SDSI-DTIC. |

**Fuente:** María José Samaniego

La Tabla 9-4contiene la prueba de aceptación correspondiente a la historia técnica 2.

Tabla 9-4: Prueba de aceptación historia técnica 2

|  |  |
| --- | --- |
| **Prueba de Aceptación** | |
| **Identificador:** PA2 | |
| **Nombre de la Prueba:** Test de Arquitectura del Sistema | |
| **N° de Historia que pertenece:** HT2 | |
| **Programador Responsable:** María José Samaniego Larrea | |
| **Descripción:**  Se verifica la conexión a la base de datos con la aplicación web.(continuación…) | |
| **Precondiciones:** Dar los permisos necesarios a la base de datos para su conexión. | **Ejecución:**   1. Configuración del archivo database.php de la aplicación con los datos de la base de datos. |
| **Post condiciones:**  Mensaje de confirmación de la conexión. | **Evaluación de Prueba:**  Exitosa |

**Fuente:** María José Samaniego

Al culminar las pruebas de aceptación de la historia técnica 2, se obtuvo un total de 1 prueba las cual resultó exitosa con un 100 % de aceptación cumpliendo con los requerimientos del cliente, procediendo a crear el proyecto para su desarrollo.

**Arquitectura General del Sistema**

La arquitectura general que se utilizará para el desarrollo del SDSI-DTIC será una arquitectura de tres capas MVC (Modelo-Vista-Controlador) la cual es utilizada por el framework CakePHP. La Figura 4-4demuestra la arquitectura general del SDSI-DTIC, donde el servidor físico funcionará como servidor de base de datos, y de aplicación.

En cuanto al usuario, cada dependencia o facultad de la ESPOCH cuenta con una máquina y una impresora. Los diagramas de componentes y despliegue se encuentran en el Anexo B1.

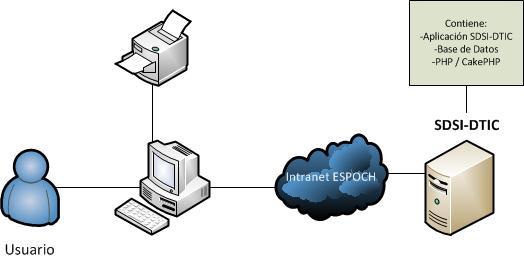


Figura. 4-4. Arquitectura General del SDSI-DTIC

**Fuente:** María José Samaniego

Para la implantación del Sistema de Seguimiento de Incidentes DTIC-ESPOCH, la Dirección de Tecnologías de la Información y Comunicación cuenta con un servidor físico con las siguientes características:

* Marca IBM
* Modelo: x3200 M2
* Procesador Intel Xeon E3110 32 3 GHz
* Memoria RAM: 1.79 GB RAM

Se puede observar en el escenario mostrado en la Figura 4-4 que se utilizará una red privada con la tecnología Internet para acceder a SDSI-DTIC.

La Tabla 5-4 detalla la tercera historia técnica, correspondiente al diseño de interfaces del sistema.

Tabla 10-4: Historia técnica diseño de las interfaces de usuario SDSI-DTIC

|  |  |
| --- | --- |
| **Historia Técnica** | |
| **Identificador:** HT3 | |
| **Nombre de la Historia:** Diseño de las Interfaces de Usuario del SDSI-DTIC. | |
| **Prioridad en Negocio:** Muy Alta  (Muy Alta/Alta / Media / Baja) | **Riesgo en Desarrollo:** Alta  (Alta / Media / Baja) |
| **Puntos Estimados:** 5 | **Iteración Asignada:** 0 |
| **Programador Responsable:** María José Samaniego Larrea | |
| **Descripción:**  Como equipo de desarrollo se necesita crear el diseño de interfaces para el SDSI-DTIC. | |
| **Observaciones:** | |

**Fuente:** María José Samaniego

La Tabla 11-4contiene la prueba de aceptación correspondiente al diseño de interfaces de usuario.

Tabla 11-4: Prueba de aceptación historia técnica 3

|  |  |
| --- | --- |
| **Prueba de Aceptación** | |
| **Identificador:** PA3 | |
| **Nombre de la Prueba:** Diseño de Interfaces | |
| **N° de Historia que pertenece:** HT3 | |
| **Programador Responsable:** María José Samaniego Larrea | |
| **Descripción:**  Se verifica que todas las interfaces de usuario cumplan la plantilla de diseño de la interfaz del sistema. | |
| **Precondiciones:** Tener la plantilla de diseño de interfaces de acuerdo a los requerimientos del cliente. | **Ejecución:**   1. Crear el diseño de las demás vistas del sistema de acuerdo a la plantilla de diseño. |
| **Post condiciones:**  Ninguna | **Evaluación de Prueba:**  Exitosa |

**Fuente:** María José Samaniego

Al culminar las pruebas de aceptación de la historia técnica 3, se obtuvo un total de 1 prueba las cual resultó exitosa con un 100 % de aceptación cumpliendo con los requerimientos del cliente, procediendo a crear el proyecto para su desarrollo.

Diseño de Interfaces del Sistema

Las interfaces de usuario se diseñaron en base a los requerimientos previamente establecidos. El SDSI-DTIC contiene una estructura jerárquica, permitiendo de esta manera al usuario saber en qué punto de la aplicación se encuentra y así poder pasar a otro sin dificultad.

Para la realización de las vistas se utilizó las siguientes características:

**Banner:** Contiene una imagen en la parte superior izquierda, dentro de la misma se encuentra el logo de la ESPOCH, en la parte derecha se encuentra una imagen con el logo de DTIC, en el centro se coloca tanto el nombre del sistema con el logo del mismo.

**Submenú:** En la parte izquierda se encuentra la fecha actual, la parte derecha se encuentra un menú desplegable correspondiente a la sesión.

**Menú Principal:** Permiten agrupar opciones de navegación para el sistema. Se realizó mediante listas ordenadas desplegables.

**Pie de Página:** Contiene la información de la institución, además de dos link con los términos de uso y él acerca de, del sistema.

**Campos Requeridos**: son cuadros de texto necesarios para el ingreso de información para las funcionalidades de ingreso y modificación.

**Modales:** Son las ventanas de dialogo que se utilizaran para el ingreso y modificación de información.

**Listas desplegables:** son utilizadas en la inserción o actualización de datos, como por ejemplo roles de usuario, grupos de trabajo, facultad o dependencia, prioridad o estado del ticket, y técnicos.

**Botones de Comando:** Estos se utilizan para dar finalización a las tareas de insertar, actualizar información, además para cancelar la acción.

**Tablas:** Estas son utilizadas para listar la información ingresada.

**Tabs de Navegación**: Estas son utilizadas para separar acciones o información.

**Mensajes de alerta**: Estos contendrán información de éxito o fallo al momento de realizar una acción de inserción, actualización o eliminación de datos.

**Paneles**: Contendrán información del ticket que se ingrese o actualice.

La Figura 5-4 muestra un ejemplo de la interfaz principal de usuario, la cual servirá de plantilla para las demás vistas del sistema.



Figura. 5-4. Diseño de interfaz del sistema SDSI-DTIC

**Fuente:** María José Samaniego

La pantalla de Autentificación de usuarios se muestra en la Figura 6-4, donde se validará el nombre de usuario y contraseña contenidos en campos requeridos y estos serán enviados por medio del botón de comando Iniciar Sesión.



Figura. 6-4. Interfaz de Autentificación

**Fuente:** María José Samaniego

Un ejemplo de interfaz para el ingreso de datos se muestra en la Figura 7-4 Esta se encuentra contenida dentro de una ventana modal. Cabe mencionar que la misma ventana se utilizara para la actualización de información.

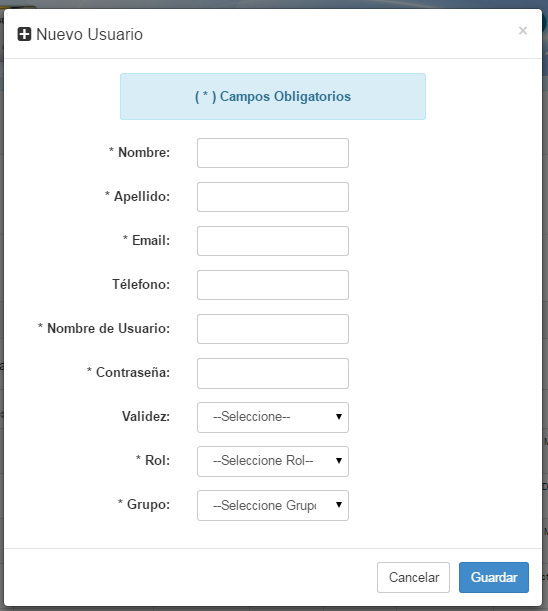


Figura. 7-4. Interfaz de Ingreso de Datos

**Fuente:** María José Samaniego

Para el reporte de incidentes se utilizará la misma estructura de la figura anterior, a excepción de que no estará contenida dentro de una ventana modal.

Los tickets separan la información correspondiente a este por paneles, tal como información general, historial de respuestas con hora exacta a la que se envió la respuesta y el destinatario, y además con una navegación por medio de Tabs.

Cada Tab contendrá campos de ingreso de datos, listas desplegables y botones de comando, tal y como se muestra en la figura a continuación en la siguiente página.

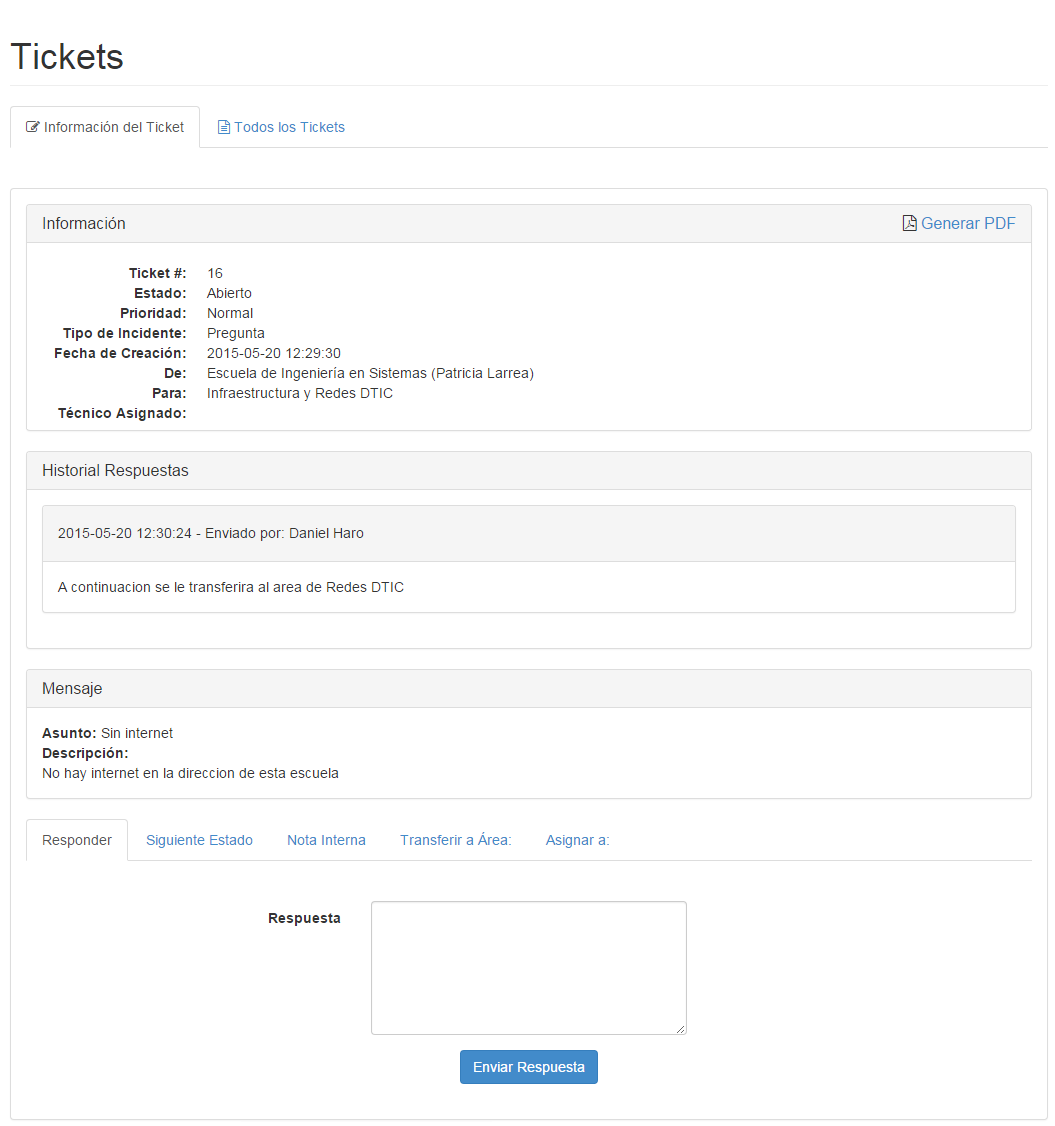


Figura. 8-4. Interfaz de Seguimiento de Incidentes

**Fuente:** María José Samaniego

Los reportes que se generan dentro del sistema están estructurados por un banner, con los logos de la Escuela Superior Politécnica de Chimborazo a la izquierda, a la derecha el de la Dirección de Tecnologías de la Información y Comunicación, en el medio con la información de la institución.

Contará con un banner, con el nombre de la institución y un logo al lado izquierdo y derecho, de la ESPOCH y DTIC respectivamente. Además de un título identificando el tipo de reporte genera y una tabla que contiene la información solicitada tal y como se muestra en la a continuación.



Figura. 9-4. Interfaz de Reportes

**Fuente**: María José Samaniego

Las interfaces de usuario se diseñaron en base a la Figura 5-4 por tal motivo para la implementación del sistema SDSI-DTIC se considera las siguientes interfaces:

* **Interfaz de Autentificación:** Esta funcionalidad permitirá realizar la autenticación del usuario con el nombre de usuario y la contraseña.
* **Interfaz Principal:** La funcionalidad de esta interfaz es dar a conocer las opciones del menú principal según el rol del usuario. Además de brindar información sobre la estructura de servicio del sistema.
* **Interfaz de Administración de roles de usuario**: Permite visualizar los roles que se le asignaran a un usuario en su creación, además de la opción de editar.
* **Interfaz de Administración de grupos de trabajo**: Permite visualizar grupos de trabajo existentes, además de la opción de editar.
* **Interfaz de Administración de usuarios**: Permite visualizar los usuarios existentes, además de la opción de ingresar, editar y validación de cada uno de ellos.
* **Interfaz de Administración de estados de un ticket**: Permite visualizar los estados existentes.
* **Interfaz de Administración de prioridades de un ticket**: Permite visualizar las prioridades existentes, además de la opción de ingresar una nueva y editar.
* **Interfaz de Administración de tipos de tickets:** Permite visualizar los tipos de tickets existentes, además de la opción de ingresar y editar.
* **Interfaz de Administración de facultades o dependencias ESPOCH**: Permite visualizar las facultades o dependencias existentes, además de la opción de ingresar una nueva y editar.
* **Interfaz de Gestión de Tickets**: Permite ver la lista de todos los tickets, abiertos y cerrados, la generación de un nuevo ticket y la búsqueda de un ticket por contenido del mismo.
* **Interfaz de Visualización del Ticket**: Permite ver la información de un ticket seleccionado y dar tratamiento al mismo.
* **Interfaz de Reportes**: Esta funcionalidad permite escoger el tipo de reporte que se desee generar.

La Tabla 12-4 detalla la tercera historia técnica, correspondiente a la codificación del sistema, la cual corresponde a las convenciones establecidas por el framework para el desarrollo.

Tabla 12-4: Historia técnica codificación del SDSI-DTIC

|  |  |
| --- | --- |
| **Historia Técnica** | |
| **Identificador:** HT4 | |
| **Nombre de la Historia:** Codificación del Sistema | |
| **Prioridad en Negocio:** Muy Alta  (Muy Alta/Alta / Media / Baja) | **Riesgo en Desarrollo:** Alta  (Alta / Media / Baja) |
| **Puntos Estimados:** 5 | **Iteración Asignada:** 0 |
| **Programador Responsable:** María José Samaniego Larrea | |
| **Descripción:**  Como equipo de desarrollo se necesita seleccionar un estándar de convenciones para proceder a la codificación del sistema | |
| **Observaciones:** | |

**Fuente:** María José Samaniego

La Tabla 13-4 contiene la prueba de aceptación correspondiente a la codificación del sistema.

Tabla 13-4: Prueba de aceptación historia técnica 4

|  |  |
| --- | --- |
| **Prueba de Aceptación** | |
| **Identificador:** PA4 | |
| **Nombre de la Prueba:** Codificación del sistema | |
| **N° de Historia que pertenece:** HT4 | |
| **Programador Responsable:** María José Samaniego Larrea | |
| **Descripción:**  Se elabora un documento resumen de las convenciones para la codificación del sistema presentado por el CakePHP. | |
| **Precondiciones:** Entendimiento de las convenciones de CakePHP. | **Ejecución:**   1. Creación de los archivos del proyecto siguiendo las convenciones recomendadas por el framework, los cuales corresponden al modelo, vista controlador |
| **Post condiciones:**  Ninguna | **Evaluación de Prueba:**  Exitosa |

**Fuente:** María José Samaniego

Al culminar las pruebas de aceptación de la historia técnica 3, se obtuvo un total de 1 prueba las cual resultó exitosa con un 100 % de aceptación cumpliendo con los requerimientos del cliente, procediendo a crear el proyecto para su desarrollo.

**Codificación del Sistema**

Para el desarrollo del sistema, se seguirán las convenciones(CakePHP Conventions — CakePHP Cookbook 2.x documentation, s.f.), establecidas por el framework CakePHP, para los distintos nombres archivo de los modelos, controladores y vistas.

Las historias de usuario describen una funcionalidad que, por sí misma, aporta valor al usuario y a la vez están escritas en lenguaje del cliente. La Tabla 14-4contiene una historia de usuario modelo, correspondiente a la autentificación de usuarios.

La estructura de la historia de usuario está formada por: el identificador HU1, donde las primeras siglas identifican que es una historia de usuario y seguido del número siguiendo un orden ascendente; el usuario que tendrá acceso a esta funcionalidad, un nombre breve y descriptivo, la prioridad del negocio la cual está establecida en el Product Backlog, el riesgo en el desarrollo establecida por el programador, los puntos estimados de la historia, la iteración a la que está asignada, el programador responsable de que esta historia se cumpla, además de una descripción de lo que se desea realizar. Y las observaciones en el caso de que las hubiera.

Tabla 14-4: Historia de usuario autentificación

|  |  |
| --- | --- |
| **Historia de Usuario** | |
| **Identificador:** HU1 | **Usuario:** Técnico DTIC |
| **Nombre de la Historia:** Autentificar Usuario | |
| **Prioridad en Negocio:** Muy Alta  (Muy Alta/Alta / Media / Baja) | **Riesgo en Desarrollo:** Alta  (Alta / Normal/Medio/ Bajo) |
| **Puntos Estimados:** 3 | **Iteración Asignada:** 1 |
| **Programador Responsable:** María José Samaniego Larrea | |
| **Descripción:**  Como Técnico DTIC deseo autentificarme al sistema, para poder acceder a las funcionalidades asignadas. | |
| **Observaciones:** | |

**Fuente:** María José Samaniego

La Tabla 15-4 contiene una de las tareas de ingeniería, correspondiente a la autentificación de usuarios, con la conexión a la base de datos desde la aplicación.

Tabla 15-4: Tarea de ingeniería de autentificación

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU1 |
| **Identificador:** TI3 |
| **Nombre de la Tarea:** Conexión con la base de datos desde la aplicación. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**  Se configurara el archivo database.php, y se procederá a ingresar los datos respectivos a la base de datos creada para el sistema. |

**Fuente:** María José Samaniego

La Tabla 16-4 contiene la prueba de aceptación correspondiente a la autentificación de usuarios, la cual resulto exitosa, en el ingreso de nombre de usuario y contraseña.

Tabla 16-4: Prueba de aceptación autentificación

|  |  |
| --- | --- |
| **Prueba de Aceptación** | |
| **Identificador:** PA5 | |
| **Nombre de la Prueba:** Autentificación de Usuario | |
| **Descripción:**  Este permitirá que el usuario se pueda autentificar en el sistema para hacer uso de las funcionalidades del sistema. | |
| **Precondiciones:**  Este se llevara a cabo si el usuario se encuentra registrado en la base de datos. | **Ejecución:**   1. Ingresar al Sistema 2. Ingresar el nombre de usuario correcto y contraseña correcto 3. Presionar Iniciar Sesión. |
| **Post condiciones:**  El sistema permite el acceso al usuario, identificándolo correctamente y mostrándole la pantalla de bienvenida. | **Evaluación de Prueba:**  Exitosa |

**Fuente:** María José Samaniego

De un total de 37 pruebas de aceptación de cada una de las historias técnicas y de usuario que conforman el Producto Backlog, se logró identificar que el sistema presenta un 5% errores correspondientes a los mensajes de confirmación de las funcionalidades de la aplicación, los cuales se verificarán nuevamente para obtener los resultados esperados.

Para de esta manera llegar al cumplimiento de las historias de usuario y técnicas, por lo cual el Sistema de Seguimiento de Incidentes para la Infraestructura de Red fue aceptado y recibido por la Dirección de Tecnologías de la Información y Comunicación de la Escuela Superior Politécnica de Chimborazo.

Las demás tareas de ingeniería, historias de usuario y pruebas de aceptación, junto con los demás Sprints que se utilizaron en el desarrollo del sistema se encuentran en el Anexo B**.**

* 1. Sistema de seguimiento de incidentes con Burn up Chart

El Burn Up Chart de la Figura 5-4 muestra las 4 iteraciones definidas para el desarrollo del SDSI-DTIC, cada iteración tiene una fecha de inicio y de fin, permitiendo observar como aumentan los puntos de cada historia de usuario a lo largo del proyecto.

La planificación inicial tenía estimado 130 puntos de historias en total, pero como se puede observar en la figura, se tuvo que añadir más historias de usuario y técnicas en el último Sprint, lo que significa más trabajo añadido por el cliente. Todas las tareas requeridas fueron completadas.

Figura. 10-4. Burn-Up Chart SDSI-DTIC

**Fuente:** María José Samaniego

CONCLUSIONES

Al finalizar el trabajo de investigación “Estudio comparativo de productividad de frameworks PHP orientados a objetos para desarrollar el Sistema de Seguimiento de Incidentes de la Infraestructura de Red en la ESPOCH se han obtenido las siguientes conclusiones:

1. El estudio comparativo entre los frameworks CakePHP y Laravel, determinó que CakePHP permite aumentar productividad del programador a un mayor nivel en la construcción de aplicaciones web.
2. En lo que se refiere a manejo de los modelos, CakePHP utiliza su propio lenguaje para enlazarse a las tablas de la base de datos todo en base a arrays, mientras que Laravel utiliza Eloquent como mapeo de Objeto-Relacional. Las vistas en CakePHP tiene la opción de utilizar el lenguaje de vistas, mientras que Laravel si la utiliza. Los controladores funcionan de manera similar.
3. Los parámetros establecidos y adaptados en esta investigación para la medición de productividad según el estándar IEEE 1045-1992 son tamaño, esfuerzo e índice de productividad física.
4. Por medio de la creación de un prototipo de prueba con CakePHP y Laravel se pudo recolectar la información para ser analizada y posteriormente interpretada.
5. El framework CakePHP presenta mejores condiciones para el aumento de productividad del programador con un promedio de 92.903 LDC/TE con respecto a Laravel con 39.102 LDC/TE, por lo tanto se procedió a crear el Sistema de Seguimiento de Incidentes de la Infraestructura de Red (SDSI-DTIC) para la Escuela Superior Politécnica de Chimborazo con el framework CakePHP.
6. El desarrollo del sistema SDSI-DTIC permitió automatizar la administración y seguimiento de incidencias en la Escuela Superior Politécnica de Chimborazo para una óptima atención al cliente con el fin de menorar los tiempos de trabajo.

RECOMENDACIONES

1. Se recomienda utilizar las convenciones sobre configuración de CakePHP, en base al idioma español para un mejor entendimiento del sistema que se esté desarrollando.
2. Debido a las constantes actualizaciones liberadas de CakePHP, se recomienda analizar la posibilidad de migrar el sistema a las últimas versiones, para obtener así una aplicación más estable.
3. El estudio realizado en esta investigación se centra en la productividad, por lo tanto es recomendable un estudio sobre la eficiencia, mantenibilidad y rendimiento que presenta CakePHP para de esta manera desarrollar aplicaciones web óptimas.
4. El presente estudio comparativo de productividad se lo realizo utilizando líneas de código, se recomienda utilizar otra medida directa de software para la métrica de tamaño como número de clases.
5. Debido a que se utilizó métricas orientadas al tamaño, se recomienda optar en utilizar las métricas orientadas a la función para la medición de productividad con aplicaciones web con diferentes funcionalidades.
6. El SDSI-DTIC permite la administración y seguimiento de incidentes a nivel de la aplicación web dentro de la ESPOCH, por lo tanto se recomienda crear la funcionalidad de enviar notificaciones por correo electrónico.

**GLOSARIO DE TÉRMINOS**

|  |  |
| --- | --- |
| **Aplicación web:** | Es cualquier aplicación que usa un navegador web como cliente, y se encuentra almacenada en un servidor remoto. |
| **Convenciones:** | Las convenciones de código se utilizan para mejoran la lectura del software, permitiendo entender código nuevo mucho más rápidamente y más a fondo. |
| **Desarrollador:** | Es un programador que se dedica a uno o más aspectos del proceso de desarrollo de software. |
| **Esfuerzo:** | Empleo de medios superiores a los normales para conseguir un fin determinado. |
| **Framework web:** | Es una estructura de software compuesta de componentes personalizables e intercambiables para el desarrollo de una aplicación. |
| **Historia técnica:** | Son requerimientos no funcionales que deben hacerse pero que no son un entregable ni están directamente relacionadas con ninguna historia específica, y no son de valor inmediato para el Dueño de Producto. |
| **Historias de usuario:** | Una historia de usuario describe una funcionalidad que, por sí misma, aporta valor al usuario. |
| **Incidente:** | Es aquello que se interpone en el transcurso normal de una situación o de un evento. |
| **Internacionalización de software:** | A menudo abreviada como i18n, es el proceso de generalización de un producto durante su desarrollo para que pueda admitir varios idiomas y convenciones culturales. |
| **Metodología ágil:** | Son una serie de técnicas para la gestión de proyectos que permiten incorporar cambios con rapidez en el desarrollo de software y que han surgido como contraposición a los métodos tradicionales de gestión como el de cascada. |
| **Métricas de software:** | Es cualquier medida o conjunto de medidas destinadas a conocer o estimar el tamaño u otra propiedad de un software o un sistema de información, generalmente para realizar comparativas o para la planificación de proyectos de desarrollo. |
| **Patrón de diseño:** | Es una solución a un problema de diseño en el desarrollo de software y otros ámbitos referentes al diseño de interacción o interfaces. |
| **Productividad económica:** | Es la relación del valor del producto producido con los costos de los recursos utilizados. |
| **Productividad física:** | Es la cantidad de producto que se puede obtener de una unidad de ese factor de producción en una unidad de tiempo. |
| **Productividad funcional:** | Es la relación de la cantidad de funcionalidad entregada con el esfuerzo invertido. |
| **Refactorización código**: | Consiste en tomar código ya existente y mejorarlo. De esta manera el código será más legible. |
| **Requerimiento de software:** | Son las características que debe tener el software instalado en una computadora para poder soportar y/o ejecutar una aplicación o un dispositivo específicos. |
| **Reutilización de código:** | La reutilización de código se refiere al comportamiento y a las técnicas que garantizan que una parte o la totalidad de un programa informático existente se puedan emplear en la construcción de otro programa. |
| **Tickets:** | Es el resguardo que contiene datos que acreditan ciertos derechos, en la mayoría de los casos obtenidos mediante un pago o servicio. |

**BIBLIOGRAFÍA**

[1] *¿Cuál es la definición de PHP?,* [en línea], s.f, [Consulta 07 de Julio 2015]

Disponible en: [http://www.alegsa.com.ar/Dic/php.php](%20http:/www.alegsa.com.ar/Dic/php.php)

[2] *¿Deberías usar un framework para tu proyecto en Internet?* , [en línea], s.f, [Consulta 01 de Junio 2014] Disponible en: <http://www.emprenderalia.com/deberias-usar-un-framework-para-tu-proyecto-web/>

[3] *¿Qué es JSP?,* [en línea], s.f, Disponible en:

http://www.oocities.org/es/rojasalazar02/ads/T2.html

[Consulta 04 de Agosto 2014]

[4] *A Guide to Using Eloquent ORM in Laravel*, [en línea], s.f, [Consulta 15 de Agosto 2014] Disponible en:

<http://scotch.io/tutorials/php/a-guide-to-using-eloquent-orm-in-laravel>

[5] **ALTEN, C**., *Mejora de la productividad en el desarrollo de aplicaciones con Frameworks,* [en línea], 2009, p.1 [Consulta 05 de Mayo 2014].

Disponible en: <http://www.capito.es/capitoweb/contenidos/Numero1/Reingenieria/alten.pdf>

[6] *Architecture of Laravel Applications - Laravel Book*, [en línea], s.f, [Consulta 15 de Agosto 2014] Disponible en:

<http://laravelbook.com/laravel-architecture/>

[7] *Beneficios de Scrum | proyectos Ágiles*, [en línea], s.f,

[Consulta 15 de Mayo 2015] Disponible en:

[http://www.proyectosagiles.org/beneficios-de-scrum#productividad-calidad](http://www.proyectosagiles.org/beneficios-de-scrum%23productividad-calidad)

[8] *Best PHP MVC frameworks of 2013*, [en línea], s.f, [Consulta 11 de Agosto 2014] Disponible en:

<http://jonathanmh.com/best-php-mvc-frameworks-of-2013/>

[9] **CÁCERES, P; & MARCOS, E**. *Procesos Agiles para el Desarrollo de Aplicación Web,* [en línea].España-Madrid: Universidad Rey Juan Carlos, s.f, p.2, [Consulta 06 de Marzo 2014].

Disponible en:

<http://www.dlsi.ua.es/~jaime/webe/articulos/s112.pdf>

[10] **CALERO, C**., *Métricas de Software* [en línea], 2006, p.2, [Consulta 20 de Agosto 2014]. Universidad de Castilla-La Mancha Disponible en: <http://eisc.univalle.edu.co/materias/Material_Desarrollo_Software/Metricas4.pdf>

[11] *CakePHP Conventions — CakePHP Cookbook 2.x documentation*, [en línea], s.f, [Consulta 13 de Agosto 2014] Disponible en:

<http://book.cakephp.org/2.0/en/getting-started/cakephp-conventions.html>

[12] **CARD, D**. *The Challenge of Productivity Measurement,* [en línea]. Pacific Northwest Software Quality Conference, 2006, pp. 2-7, [Consulta 25 de Agosto 2014]

Disponible en: <http://www.netvalence.net/sites/default/files/resources/recom_read/TheChallengeofProductivityMeasurement.pdf>

[13] **CÓDIGOLÍNEA**, *Comparación y Rendimiento de Frameworks PHP*, [en línea], s.f, [Consulta 02 de Junio 2014] Disponible en: <http://codigolinea.com/2008/06/04/compararcion-y-rendimiento-de-frameworks-php/>

[14] **COGNEAU, A**. *Laravel: Is It Really Clean and Classy?* ,[en línea] , 2012, [Consulta 18 de Agosto 2014] Disponible en: <http://www.sitepoint.com/laravel-really-clean-and-classy/>

[15] *Convenciones en CakePHP — documentación de CakePHP Cookbook - 2.x,* [en línea], s.f, [Consulta 12 de Agosto 2014] Disponible en: <http://book.cakephp.org/2.0/es/getting-started/cakephp-conventions.html>

[16] *Count Lines of Code*, [en línea], s.f, [Consulta 07 de Julio 2015] Disponible en:

[http://cloc.sourceforge.net/](%20http:/cloc.sourceforge.net/)

[17] *Curso de PHP*, [en línea], s.f, [Consulta 05 de Agosto 2014] Disponible en:

<http://docencia.udea.edu.co/cci/php/>

[18] *Desarrollo de aplicaciones web,* [en línea], s.f, [Consulta 05 de Mayo 2014] Disponible en:

<http://www.enubes.com/desarrollo-de-aplicaciones-web.html>

[19] *Documentación de Laravel en Español versión 4.1*, [en línea], s.f, [Consulta 14 de Agosto 2014] Disponible en: <http://laravel.montogeek.co/page/introduction>

[20] **EBERT, D. C**. *Measuring and Improving Productivity,* [en línea]. Zurich-Suiza: Alcatel SET, 2006, pp. 2-6, [Consulta 25 de Agosto 2014]. Disponible en: <http://www.compaid.com/caiinternet/ezine/ebert-productivity-metrics.pdf>

[21] *Entendiendo el Modelo - Vista - Controlador — documentación de CakePHP Cookbook - 2.x,* [en línea], s.f, [Consulta 12 de Agosto 2014] Disponible en:

<http://book.cakephp.org/2.0/es/cakephp-overview/understanding-model-view-controller.html>

[22] *Escuela Superior Politécnica de Chimborazo*, [en línea], s.f, [Consulta 10 de Julio 2015] Disponible en:

<http://espoch.edu.ec/>

[23] *Frameworks (CSS avanzado)*, [en línea], s.f, [Consulta 06 de Agosto 2014] Disponible en:

<http://librosweb.es/css_avanzado/capitulo_5.html>

[24] **GONZÁLEZ, E.** *Tipos de variables en PHP. Declaración y asignación. La función echo: mostrar texto en pantalla (CU00816B),* [en línea], s.f, p.1, [Consulta 06 de Agosto 2014]. Disponible en: <http://www.aprenderaprogramar.com/index.php?option=com_content&view=article&id=544:tipos-de-variables-en-php-declaracion-y-asignacion-la-funcion-echo-mostrar-texto-en-pantalla-cu00816b&catid=70:tutorial-basico-programador-web-php-desde-cero&Itemid=193>

[25] *Google Trends - Web Search interest: laravel, CodeIgniter, CakePHP, Symfony - Worldwide, Jun 2011 - Feb 2014*, [en línea], s.f, [Consulta 02 de Junio 2014] Disponible en: [http://www.google.com/trends/explore#q=laravel%2C%20%2Fm%2F02qgdkj%2C%20cakephp%2C%20symfony&date=6%2F2011%2033m&cmpt=q](http://www.google.com/trends/explore%23q=laravel%2C%20%2Fm%2F02qgdkj%2C%20cakephp%2C%20symfony&date=6%2F2011%2033m&cmpt=q)

[26] **GUTIÉRREZ, J. J**. *¿Qué es un framework web?,* [en línea]. Sevilla-España: Universidad de Sevilla,s.f, p.1, [Consulta 06 de Agosto 2014] Disponible en: <http://www.lsi.us.es/~javierj/investigacion_ficheros/Framework.pdf>

[27] **IEEE 1045-1992**, *IEEE Standard for Software Productivity Metrics,* [en línea]1992, pp. 5-11. [Consulta 25 de Agosto 2014] Disponible en:

[http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=2858](%20http:/ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=2858)

[28] **ISO/IEC FDIS 9126-1**, *International Standard ISO/IEC FDIS 9126-1* [en línea], s.f., p.3. [Consulta 02 de Junio 2014] Disponible en:

<http://www.cse.unsw.edu.au/~cs3710/PMmaterials/Resources/9126-1%20Standard.pdf>

[29] **INCE, D.; SHARP, H.; & WOODMAN M**. *Introduction to software project management and quality assurance,* [en línea]. Londres-Inglaterra: McGraw-Hill Book, 1993, pp. 80-81, [Consulta 20 de Agosto 2014] Disponible en: <http://www.petronet.ir/documents/10180/2323250/introduction_to_software_project_management_and_quality_>

[30] **LANCOR, L.; & SAMYUKTA, K**. *Analyzing PHP Frameworks for Use in a Project-Based Software Engineering Course,* [en línea]. New Haven-USA: Southern Connecticut State University, s.f., p.4. [Consulta 02 de Marzo 2015] Disponible en: <http://db.grinnell.edu/sigcse/sigcse2013/Program/viewAcceptedProposal.pdf?sessionType=paper&sessionNumber=116>

*Laravel*, [en línea], s.f, [Consulta 14 de Agosto 2014] Disponible en: <http://laraveldominicana.blogspot.com/p/que-es-laravel.html>

[31] *Laravel - The PHP framework for web artisans*, [en línea], s.f, [Consulta 14 de Agosto 2014] Disponible en:

<http://laravel.com/docs>

[32] **LETELIER, P.; & PENADÉS, M. C**., *Metodologías ágiles para el desarrollo de software: eXtreme Programming (XP),* [en línea]*,* s.f*,*  [Consulta 09 de Marzo 2014] Disponible en:

<http://www.cyta.com.ar/ta0502/v5n2a1.htm>

[33] **MATEU, C**. *Desarrollo de aplicaciones web* [en línea]. Barcelona-España: Universidad Abierta de Cataluña, 2004, p.20. [Consulta 31 de Julio 2014] Disponible en: <http://www.uoc.edu/masters/softwarelibre/esp/materials/Desarrollo_web.pdf>

[34] **MENÉNDEZ, R.**, Historia del desarrollo de aplicaciones Web, [en línea] , 2012, [Consulta 31 de Julio 2014] Disponible en:

<http://www.um.es/docencia/barzana/DIVULGACION/INFORMATICA/Historia-desarrollo-aplicaciones-web.html>

[35] *Metodologías Agiles de Desarrollo*, [en línea], 2009, [Consulta 09 de Marzo 2015] Disponible en:

<http://ronald-sistemas.blogspot.com/2009/10/metodologias-agiles-de-desarrollo.html>

[36] **MICROSOFT**, *ASP.NET Overview*, [en línea], s.f, [Consulta 04 de Agosto 2014] Disponible en: <http://msdn.microsoft.com/en-us/library/vstudio/4w3ex9c2(v=vs.100).aspx>

[37] **NAVARRO, A**. *Proceso de software y métricas de proyectos,* [en línea]. Corrientes-Argentina: Universidad Nacional del Nordeste,s.f, pp. 2-12 [Consulta 19 de Agosto 2014]. Disponible en: <http://exa.unne.edu.ar/informatica/evalua/Sitio%20Oficial%20ESPD-Talleres/TRABAJO%20DE%20PAG%20DE%20LEON/web/Otros%20metricas/4._Proceso_de_software_y_metricas_de_proyectos.pdf>

[38] **NODA, T.; & JIA, C**. *Empirical Case Study of Measuring Productivity of Programming Language Ruby and Ruby on Rails*, [en línea]. The Sixth International Conference on Software Engineering Advances,2011, p.2. [Consulta 01 de Marzo 2015] Disponible en:

<http://www.thinkmind.org/download.php?articleid=icsea_2011_15_30_10268>

[39] **NWELIH, E.; & AMADIN, I**. *Modeling Software Reuse in Traditional Productivity Model* [en línea]. Benin-Nigeria: Medwell Journals, 2008, p.1, [Consulta 07 de Julio 2015] Disponible en:

<http://docsdrive.com/pdfs/medwelljournals/ajit/2008/484-488.pdf>

[40] **PALACIO, J.** *El Modelo Scrum,* [en línea]*,* 2006, p.1, [Consulta 09 de Marzo 2015] Disponible en:

http://www.navegapolis.net/files/s/NST-010\_01.pdf

[41] **PERALTA, A**. *Metodología SCRUM,* [en línea]. Montevideo-Uruguay: Universidad ORT Uruguay, 2003, pp. 2-9, [Consulta 10 de Marzo 2015] Disponible en: <http://fi.ort.edu.uy/innovaportal/file/2021/1/scrum.pdf>

[42] *PHP: Definición,* [en línea], s.f, [Consulta 04 de Agosto 2014] Disponible en: [http://www.ecured.cu/index.php/PHP#Caracter.C3.ADsticas\_de\_PHP](http://www.ecured.cu/index.php/PHP%23Caracter.C3.ADsticas_de_PHP)

[43] *PHP: ¿Qué es PHP? - Manual,*[en línea], s.f, [Consulta 01 de Agosto 2014] Disponible en:

<http://php.net/manual/es/intro-whatis.php>

[44] *PHP: Comentarios - Manual,* [en línea], s.f, [Consulta 06 de Agosto 2014] Disponible en:

<http://php.net/manual/es/language.basic-syntax.comments.php>

[45] *PHP: Historia de PHP - Manual*, [en línea], s.f, [Consulta 04 de Agosto 2014] Disponible en:

[http://php.net/manual/es/history.php.php](%20http:/php.net/manual/es/history.php.php)

[46] *Programming Language Definition*, [en línea], 2011, [Consulta 01 de Agosto 2014] Disponible en: <http://www.techterms.com/definition/programming_language>

[47] **RAMÍREZ, J**., *Programación rápida con CakePHP,* [en línea]*,* s.f, pp. 2-3, [Consulta 12 de Agosto 2014]. Disponible en:

<https://www.linux-magazine.es/issue/23/CakePHP.pdf>

[48] **SAAVEDRA, E,** *Catalyst: Framework para el desarrollo de aplicaciones Web*, [en línea]. Chile: Encuentro Linux, 2009, p.4, [Consulta 06 de Agosto 2014]. Disponible en: <http://2009.encuentrolinux.cl/wp-content/uploads/2009/09/catalyst.pdf>

[49] **SALCEDO, E., GIL, C.; & GARCÍA, A. P**, *Calidad y métricas de software: una propuesta desde la lógica difusa* [en línea].Bogotá-Colombia: Fundación Universitaria Los Libertadores*,* 2011, p.3, [Consulta 18 de Agosto 2014]. Disponible en: <http://www.libertadores.edu.co:8089/?idcategoria=5776&download=Y>

[50] **SCITOOLS**, *UnderstandTM Static Code Analysis Tool | SciTools.com*, [en línea], s.f, Disponible en: https://scitools.com/

2015-07-07

[51] **SIDLER, ROBER**. *Software Productivity,* [en línea],2002*,* [Consulta 07 de Julio 2015] Disponible en: <http://www.umsl.edu/~sauterv/analysis/488_f02_papers/SoftwareProductivity.html>

[52] **SMITH, G**., *13 PHP Frameworks to Help Build Agile Applications,*

[en línea] 2014, [Consulta 06 de Mayo 2014] Disponible en:

<http://mashable.com/2014/04/04/php-frameworks-build-applications/>

[53] **SURGUY, M**., *History of Laravel PHP framework, Eloquence emerging*

*- Maks Surguy’s blog on PHP and Laravel,* [en línea], 2013 *,* [Consulta 07 de Julio 2015] Disponible en:

<http://maxoffsky.com/code-blog/history-of-laravel-php-framework-eloquence-emerging/>

[54] *Top PHP Frameworks 2014*, [en línea], 2014, [Consulta 02 de Junio 2014] Disponible en:

<http://www.php-developers.org/blog/php/top-php-frameworks-2014.html>

[55] *Usar o no un Framework.*, [en línea], 2012, [Consulta 04 de Mayo 2014] Disponible en: <http://www.kabytes.com/programacion/usar-o-no-un-framework/>

[56] *Variables Constantes Tipos De Datos*, [en línea], s.f, p.1. [Consulta 20 de Agosto 2014] Disponible en: <http://amecsa.org/Diplomado/Modulo13/M2_VariablesConstantesTiposDeDatos.pdf>

[57] **VISCONTI, M**., Medición *de Productividad de Software*, [en línea]. Valparaiso-Chile, s.f, p.2, [Consulta el 25 de Agosto 2014] Disponible en: <http://www.inf.utfsm.cl/~visconti/iswav/documentos/ProdyEstEsfuerzo.pdf>

[58] *What is CakePHP? Why use it? — CakePHP Cookbook 2.x documentation*, [en línea], s.f, [Consulta 01 de Agosto 2014] Disponible en: <http://book.cakephp.org/2.0/en/cakephp-overview/what-is-cakephp-why-use-it.html>

[59] *What is Scrum? An Agile Framework for Completing Complex Projects - Scrum Alliance*, [en línea], s.f, [Consulta 10 de Marzo 2015] Disponible en:

<https://www.scrumalliance.org/why-scrum>

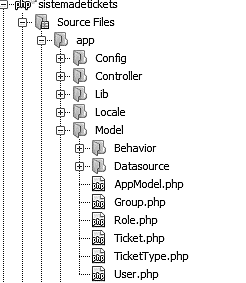
**ANEXO A:**

**Análisis Comparativo de los Frameworks CakePHP y Laravel**

**Anexo A1: Directorio de archivos dentro de MVC**

**Archivos Prototipo CakePHP**

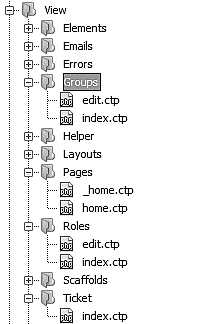
* **Modelo**



**Figura. 1-A.** Archivos Modelo Prototipo CakePHP

**Fuente**: María José Samaniego

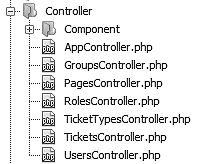
* **Vista**



**Figura. 2-A.** Archivos Vista Prototipo CakePHP

**Fuente**: María José Samaniego

* **Controlador**

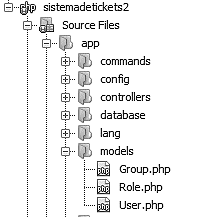


**Figura. 3-A.** Archivos Controlador Prototipo CakePHP

**Fuente:** María José Samaniego

**Archivos Prototipo Laravel**

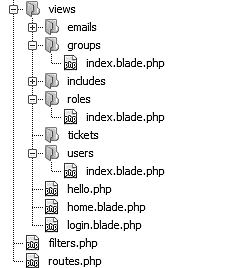
* **Modelo**



**Figura. 4-A.** Archivos Modelo Prototipo Laravel

**Fuente**: María José Samaniego

* **Vista**



**Figura. 5-A.** Archivos Vista Prototipo Laravel

**Fuente**: María José Samaniego

* **Controlador**



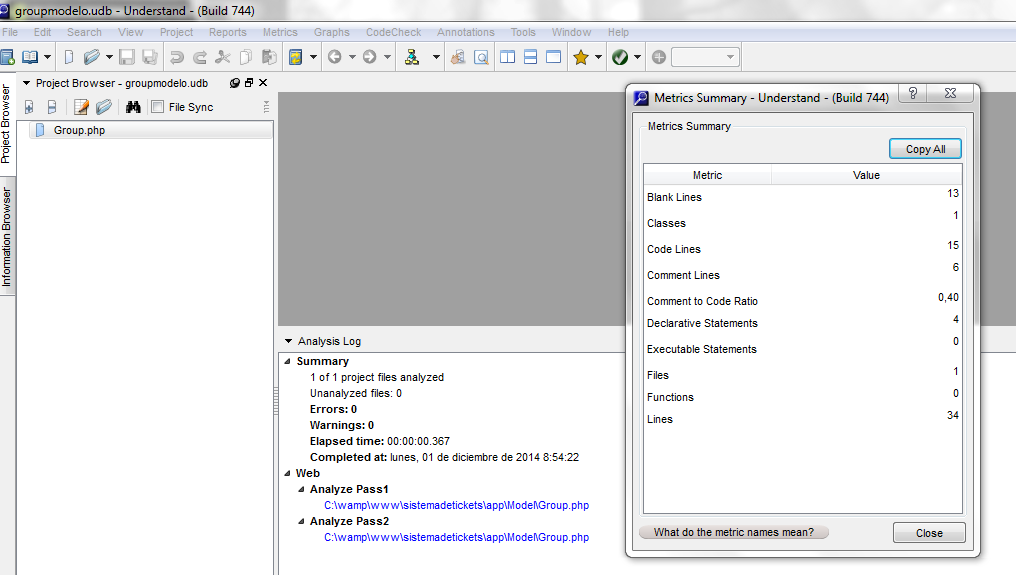
**Figura. 6-A.** Archivos Controlador Prototipo CakePHP

**Fuente**: María José Samaniego

**Anexo A2: Tamaño**

**Resultado de Líneas de código por el programa Understand para CakePHP**

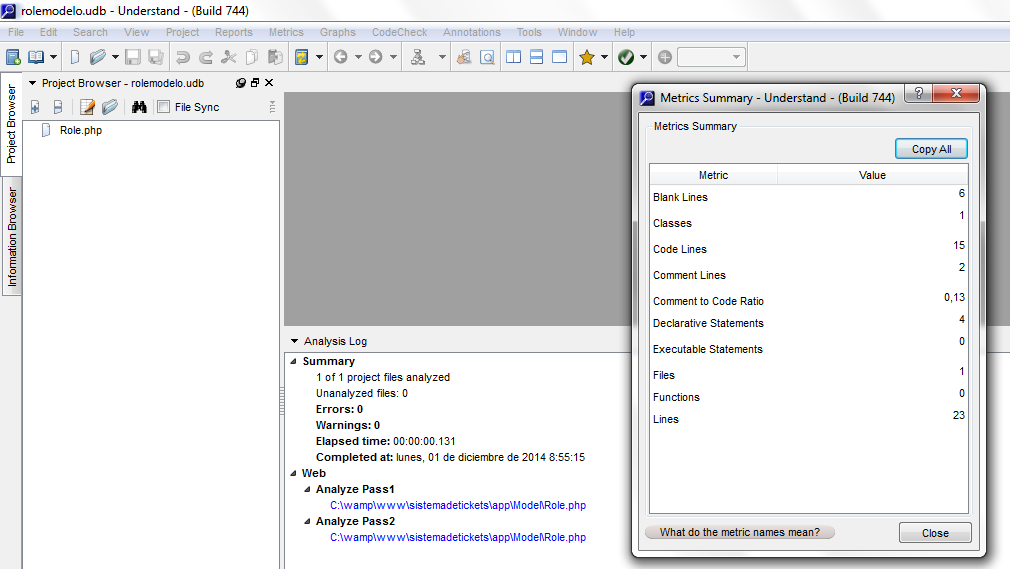
* /app/Model/Group.php



**Figura. 7-A**. Resultados LDC Modelo Grupo CakePHP

**Fuente**: María José Samaniego

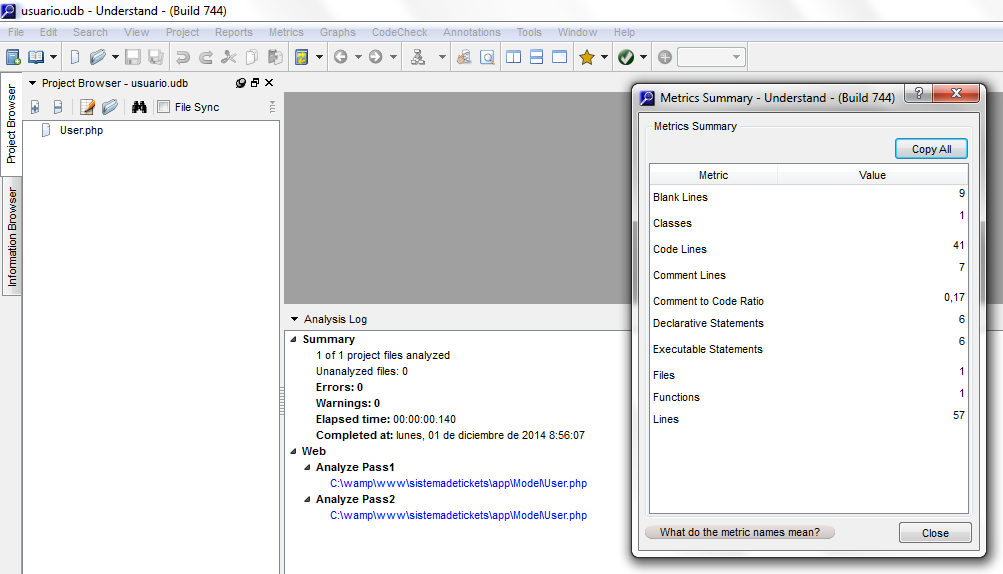
* /app/Model/Role.php



**Figura. 8-A.** Resultados LDC Modelo Rol CakePHP

**Fuente:** María José Samaniego

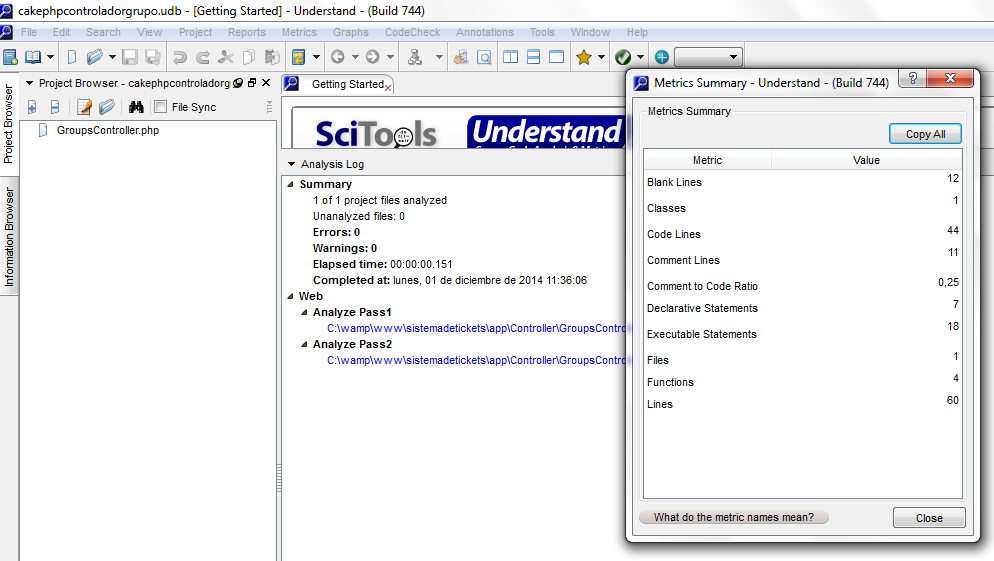
* /app/Model/User.php



**Figura. 9-A.** Resultados LDC Modelo Usuario CakePHP

**Fuente:** María José Samaniego

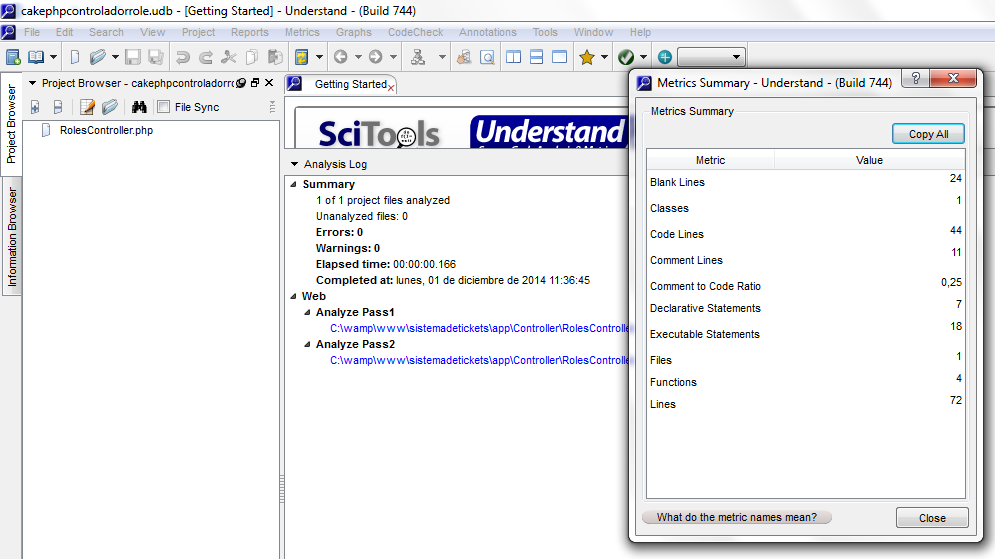
* /app/Controller/GroupsController.php



**Figura. 10-A.** Resultados LDC Controlador Grupos CakePHP

**Fuente:** María José Samaniego

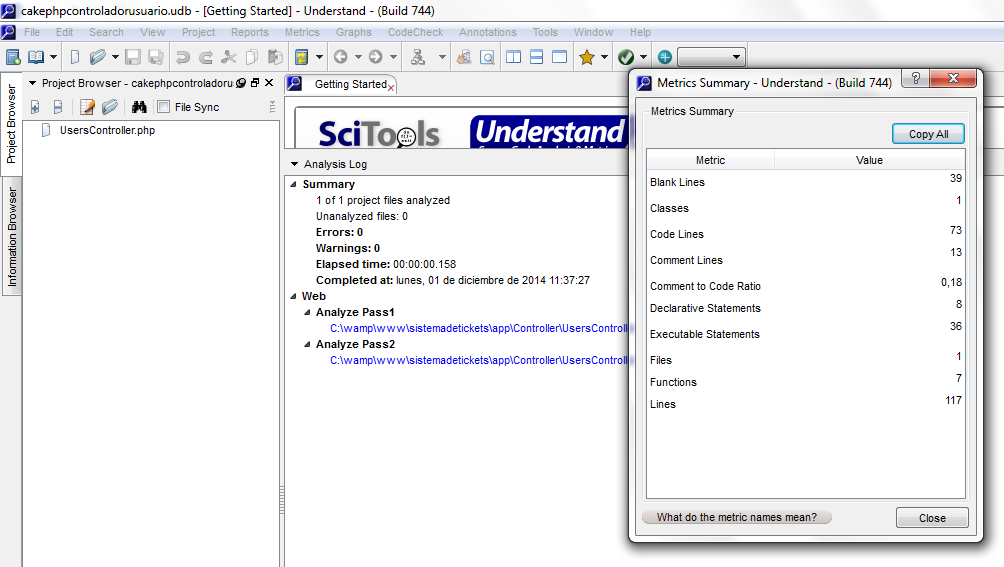
* /app/Controller/RolesController.php



**Figura. 11-A.** Resultados LDC Controlador Roles CakePHP

**Fuente:** María José Samaniego

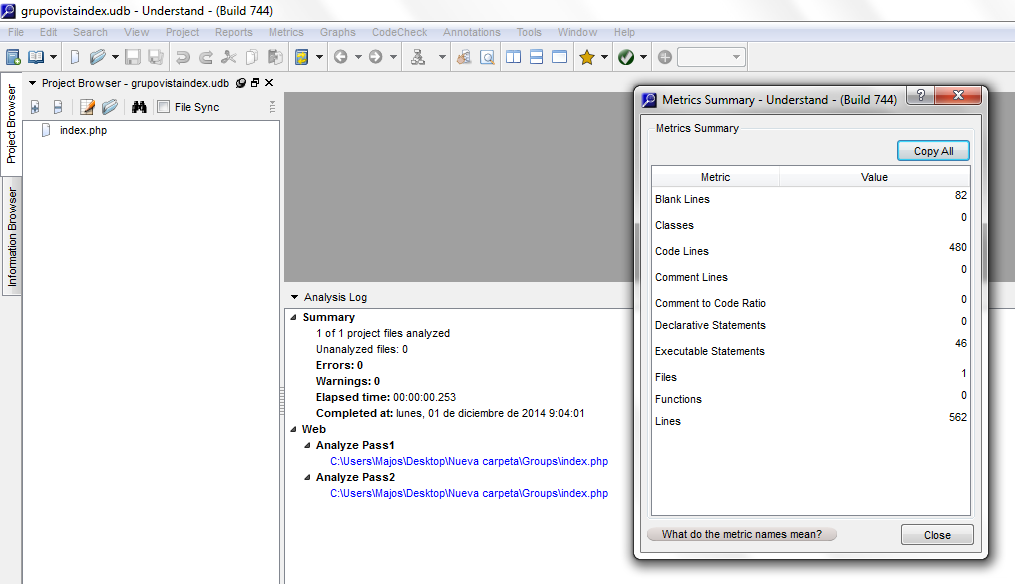
* /app/Controller/UsersController.php



**Figura. 12-A.** Resultados LDC Controlador Usuarios CakePHP

**Fuente:** María José Samaniego

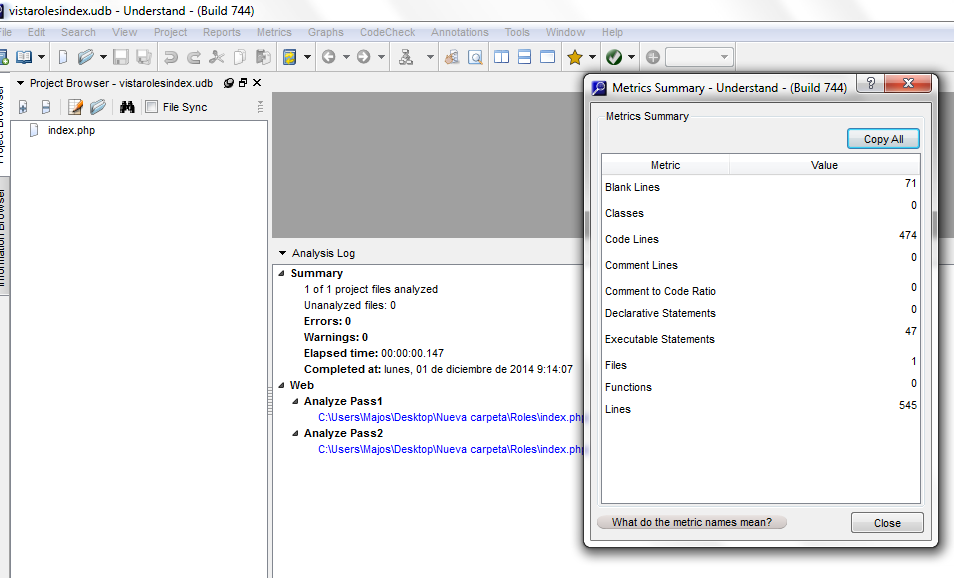
* /app/View/Groups/index.ctp



**Figura. 13-A.** Resultados LDC Vista Grupos CakePHP

**Fuente**: María José Samaniego

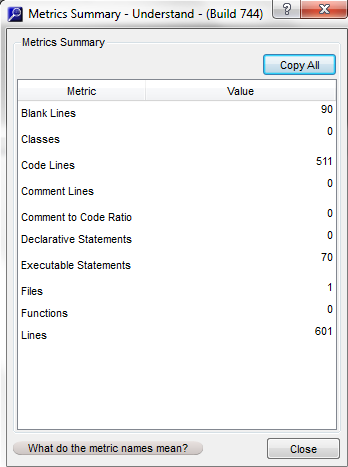
* /app/View/Roles/index.ctp



**Figura. 14-A.** Resultados LDC Vista Roles CakePHP

**Fuente**: María José Samaniego

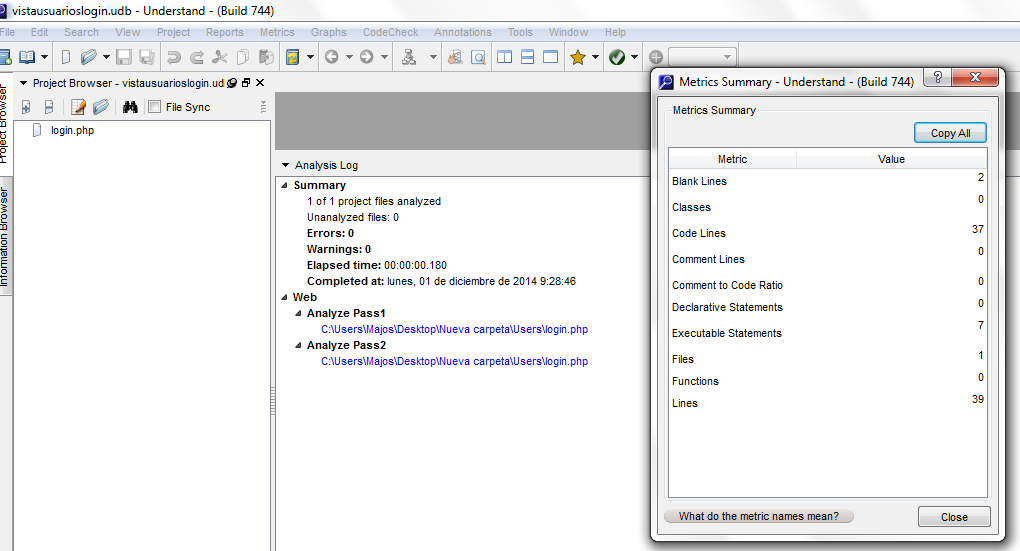
* /app/View/Users/index.ctp



**Figura. 15-A.** Resultados LDC Vista Usuarios CakePHP

**Fuente:** María José Samaniego

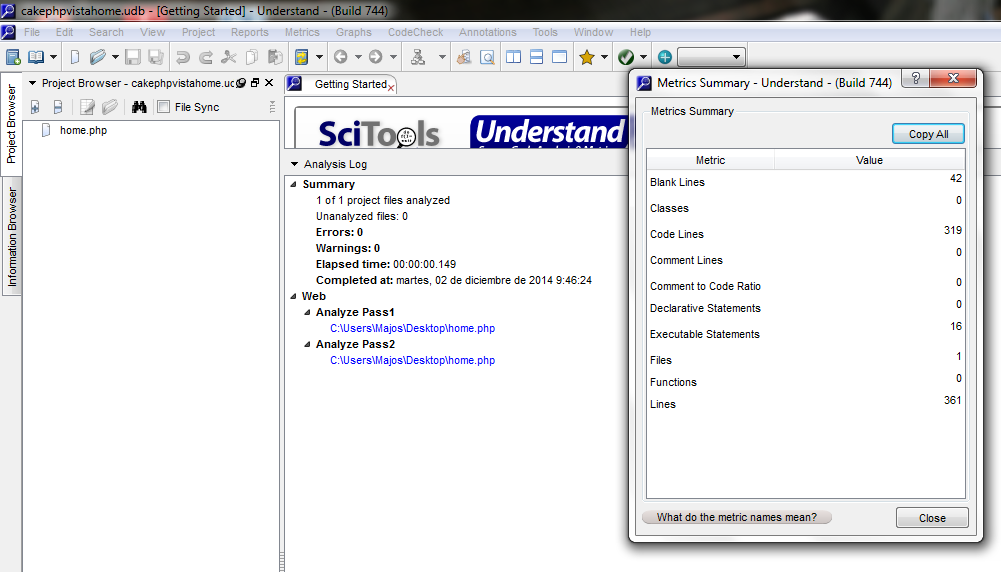
* /app/View/Users/login.ctp



**Figura. 16-A.** Resultados LDC Vista Usuarios 2 CakePHP

**Fuente:** María José Samaniego

* /app/View/Pages/home.ctp

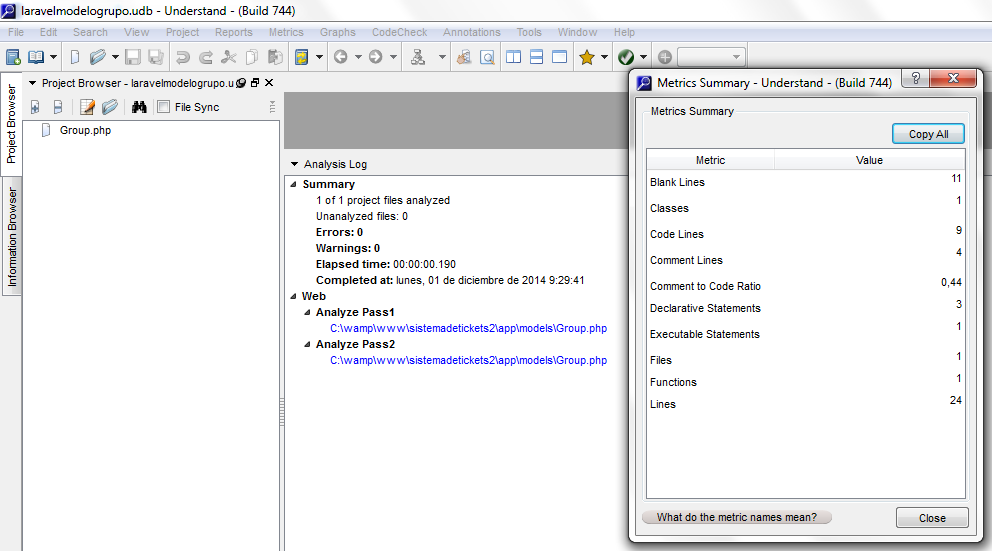


**Figura. 17-A.** Resultados LDC Vista Home CakePHP

**Fuente**: María José Samaniego

**Resultado de Líneas de código por el programa Understand para Laravel**

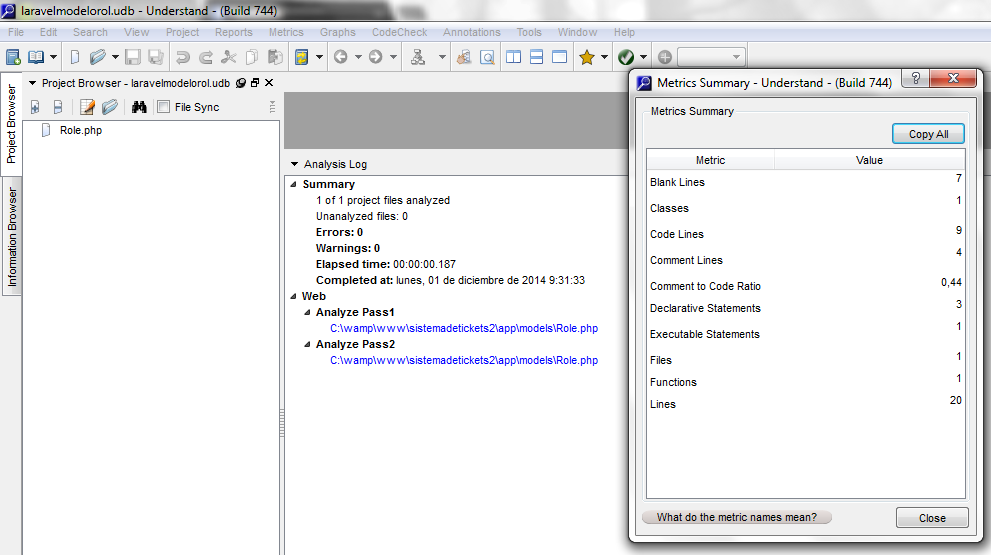
* /app/models/Group.php



**Figura. 18-A.** Resultados LDC Vista Grupos Laravel

**Fuente:** María José Samaniego

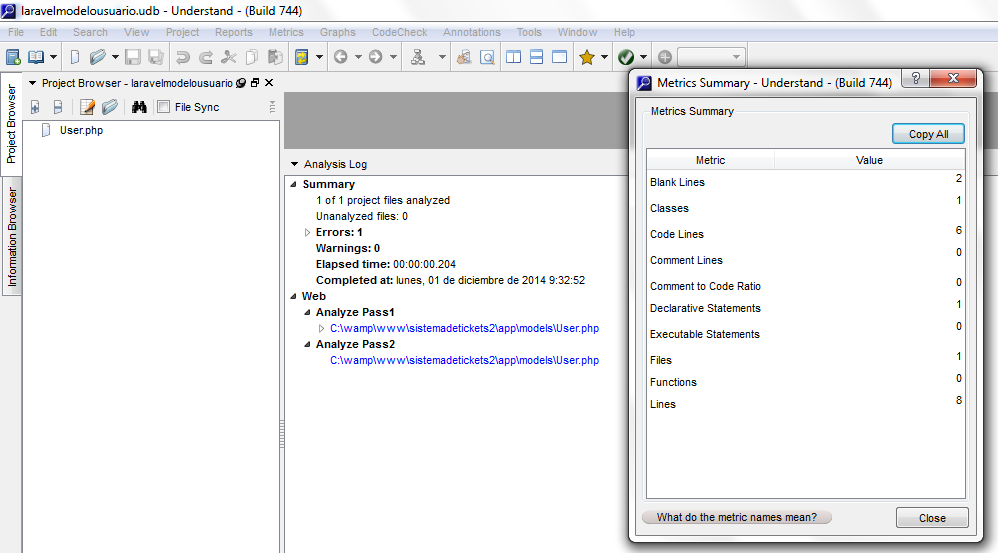
* /app/models/Role.php



**Figura. 19-A.** Resultados LDC Vista Roles Laravel

**Fuente**: María José Samaniego

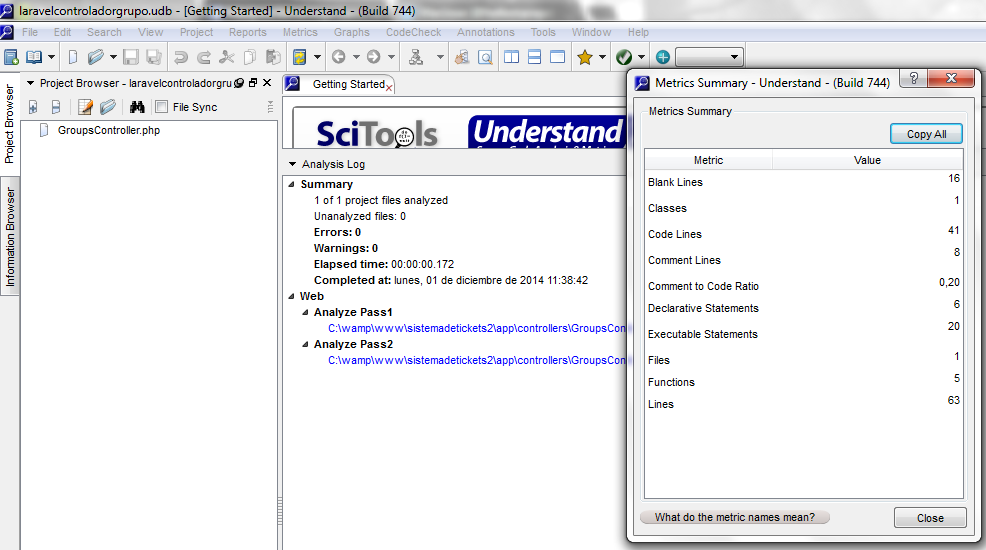
* /app/models/User.php



**Figura. 20-A.** Resultados LDC Vista Usuarios Laravel

**Fuente:** María José Samaniego

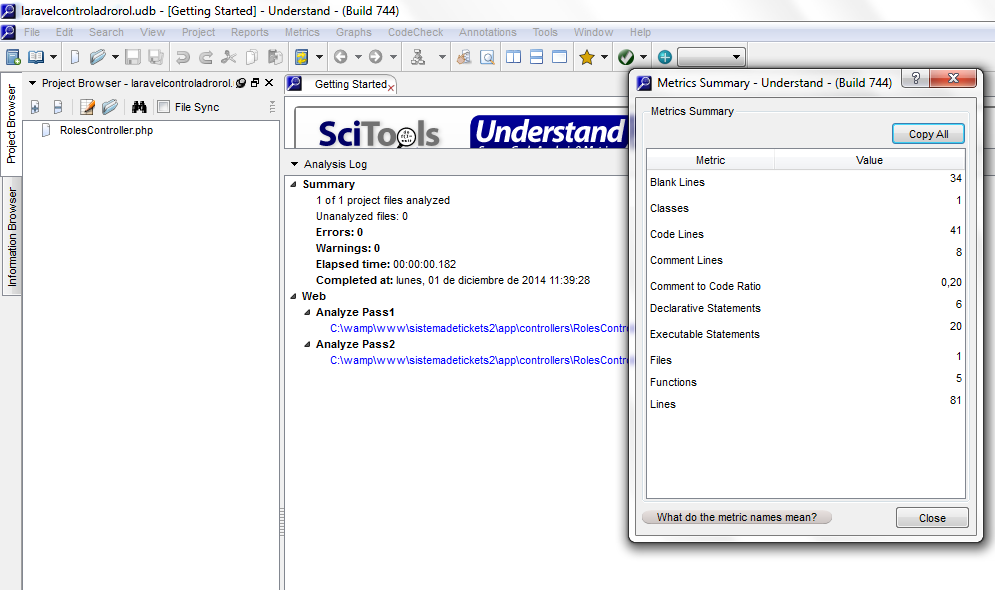
* /app/controllers/GroupsController.php



**Figura. 21-A.** Resultados LDC Controlador Grupos Laravel

**Fuente:** María José Samaniego

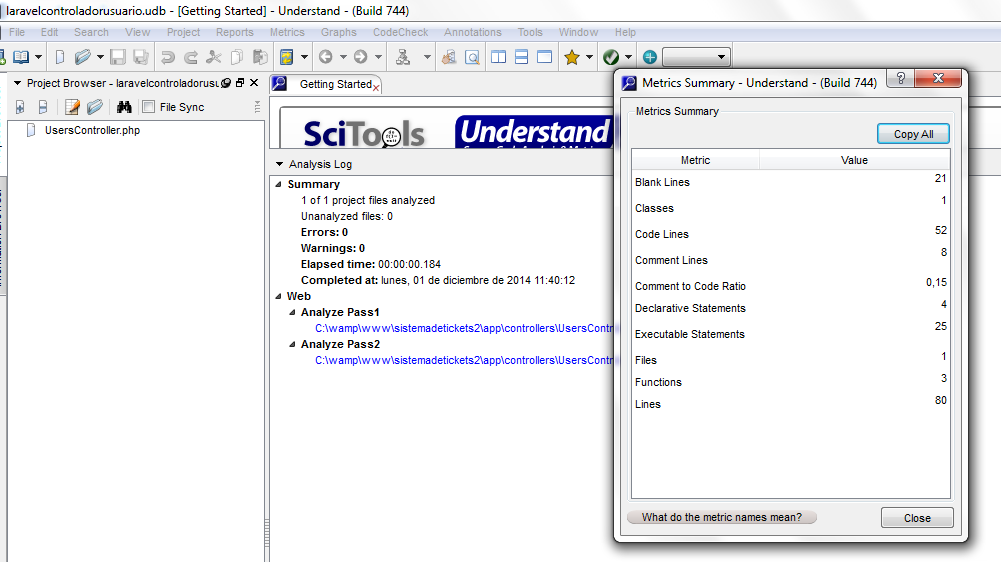
* /app/controllers/RolesController.php



**Figura. 22-A.** Resultados LDC Controlador Roles Laravel

**Fuente:** María José Samaniego

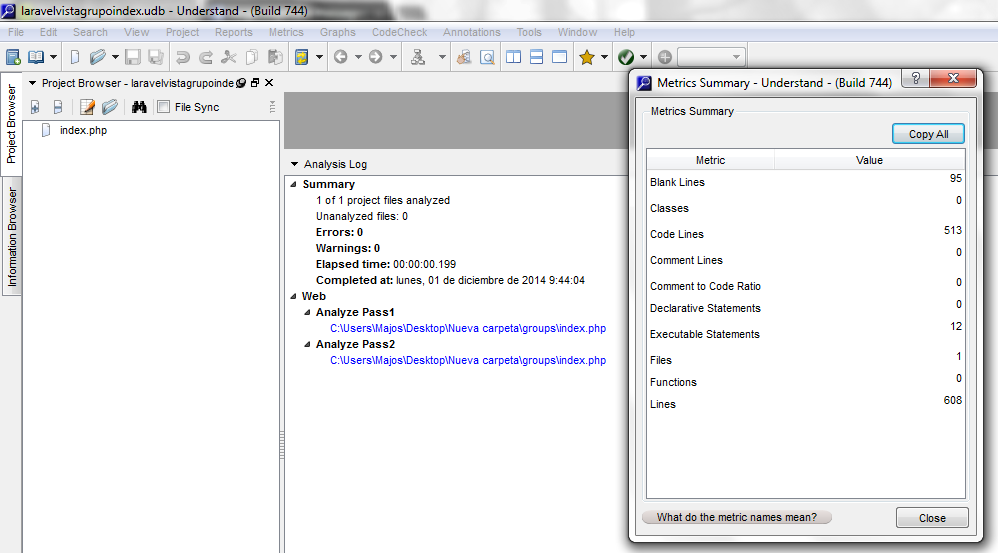
* /app/controllers/UsersController.php



**Figura. 23-A.** Resultados LDC Controlador Usuarios Laravel

**Fuente:** María José Samaniego

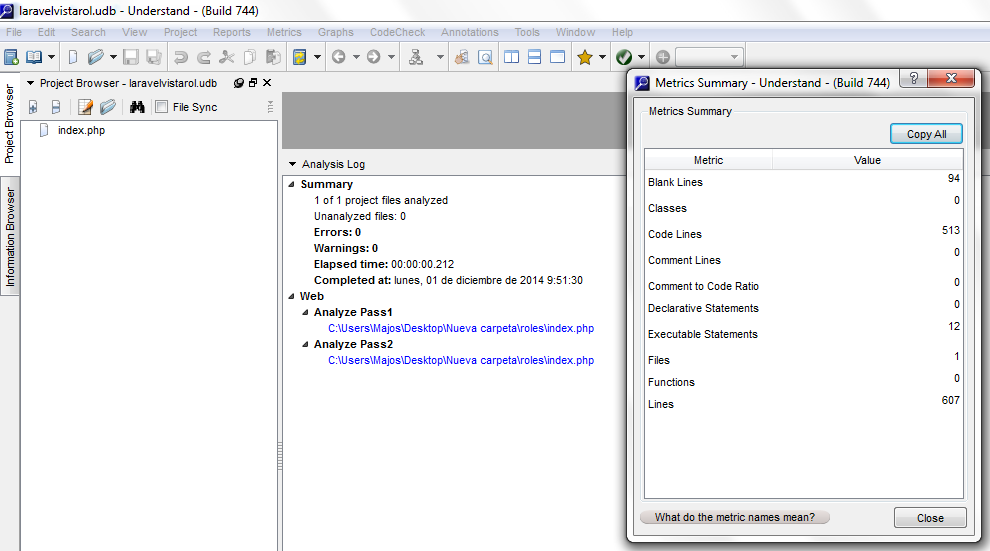
* /app/views/groups/index.blade.php



**Figura. 24-A.** Resultados LDC Vista Grupos Laravel

**Fuente:** María José Samaniego

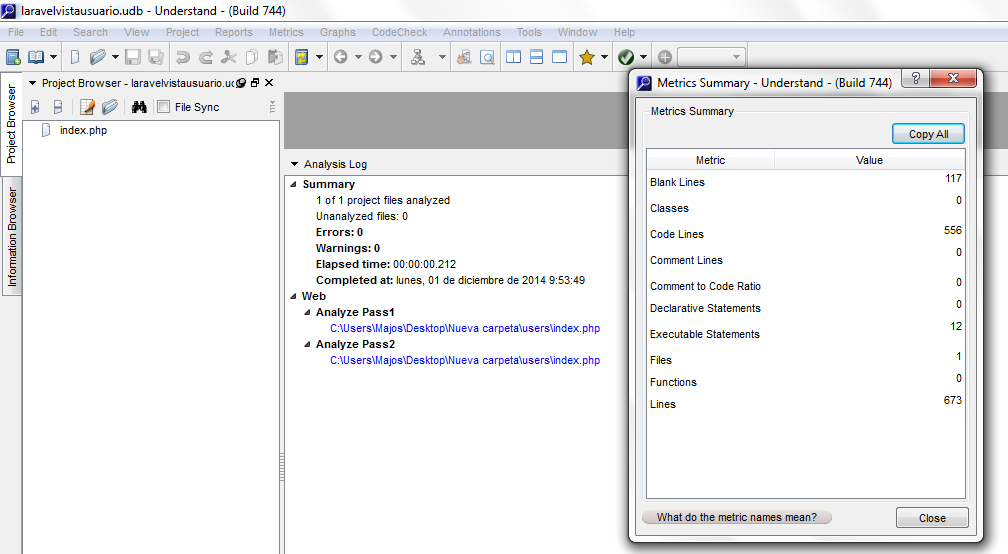
* /app/views/roles/index.blade.php



**Figura. 25-A.** Resultados LDC Vista Roles Laravel

**Fuente:** María José Samaniego

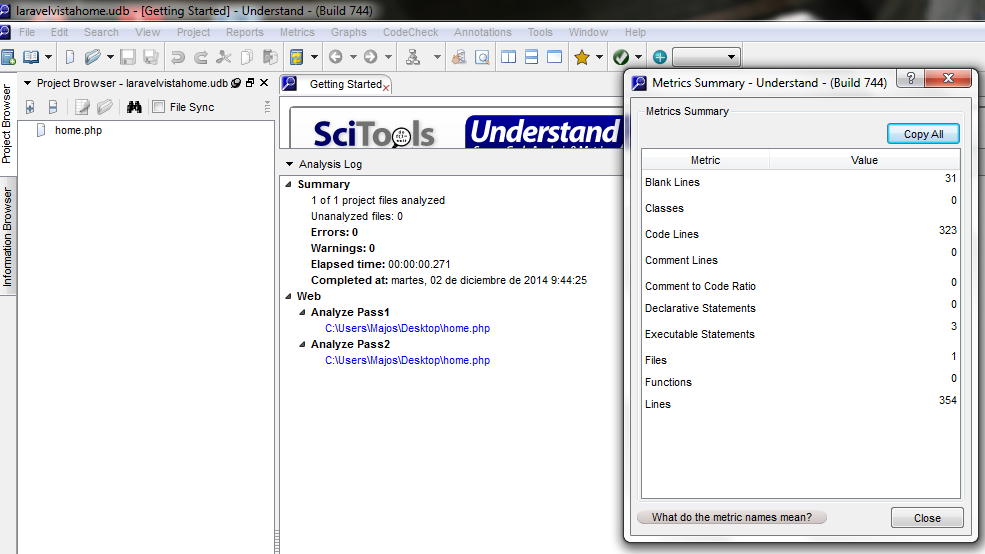
* /app/views/users/index.blade.php



**Figura. 26-A.** Resultados LDC Vista Usuarios Laravel

**Fuente:** María José Samaniego

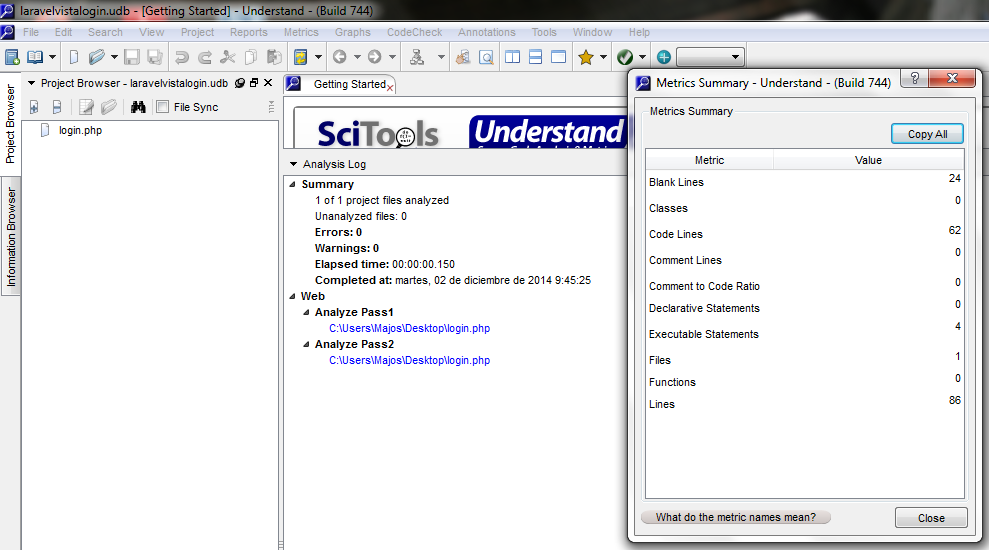
* /app/views/home.blade.php



**Figura. 27-A.** Resultados LDC Vista Home Laravel

Fuente: María José Samaniego

* /app/views/users/login.blade.php



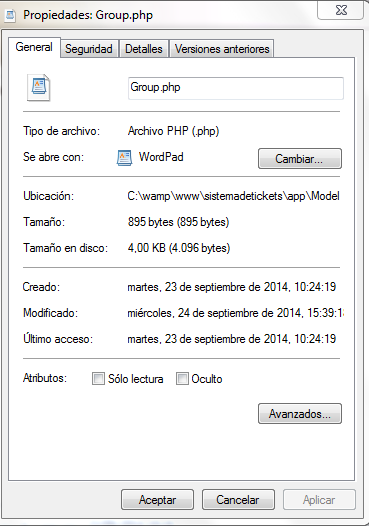
**Figura. 28-A.** Resultados LDC Vista Login Laravel

**Fuente:** María José Samaniego

**Anexo A3: Tiempo**

**CakePHP**

* /app/Model/Group.php



**Figura. 29-A.** Propiedad archivo Grupo

**Fuente:** María José Samaniego

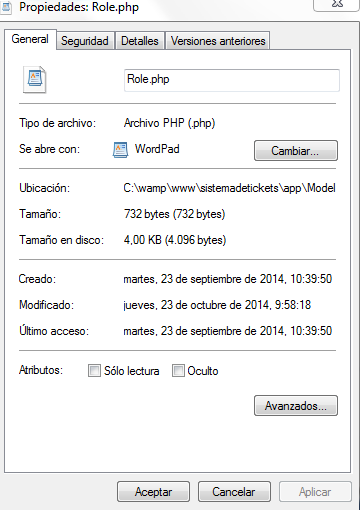
TIEMPO DE DESARROLLO + TIEMPO DE ULTIMA MODIFICACIÓN

TD = 52 MINUTOS = 0,87 HORAS

TUM= 1 HORA 30 MINUTOS = 1, 5 HORAS

0,82 HORAS + 1,5 HORAS + 0 = 2,32 HORAS

* /app/Model/Role.php



**Figura. 30-A.** Propiedad archivo Rol

**Fuente:** María José Samaniego

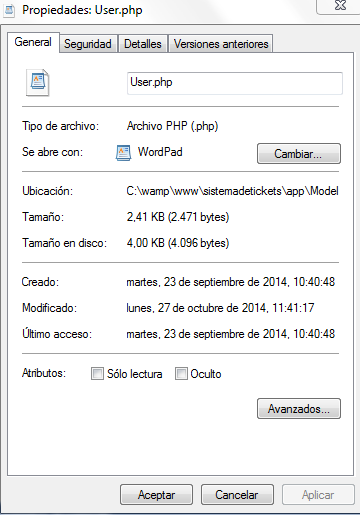
TIEMPO DE DESARROLLO + TIEMPO DE ULTIMA MODIFICACIÓN

TD = 37 MINUTOS = 0,62 HORAS

TUM= 22 MINUTOS = 0,37 HORAS

0.62 HORAS + 0,37 HORAS = 0,99 HORAS

* /app/Model/User.php



**Figura. 31-A.** Propiedad archivo Usuario

**Fuente:** María José Samaniego

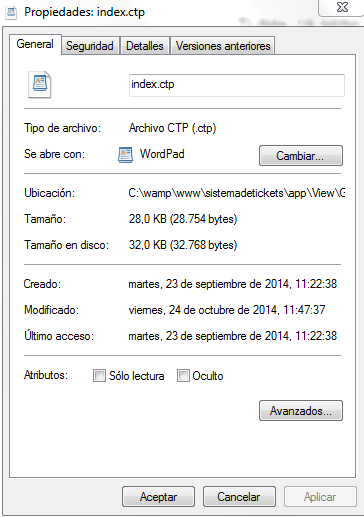
TIEMPO DE DESARROLLO + TIEMPO DE ULTIMA MODIFICACIÓN

TD = 36 MINUTOS = 0,6 HORAS

TUM= 19 MINUTOS = 0.32 HORAS

0.6 HORAS + 0.32 HORAS = 0,92 HORAS

* /app/View/Groups/index.ctp



**Figura. 32-A.** Propiedad archivo Vista Grupo

**Fuente:** María José Samaniego

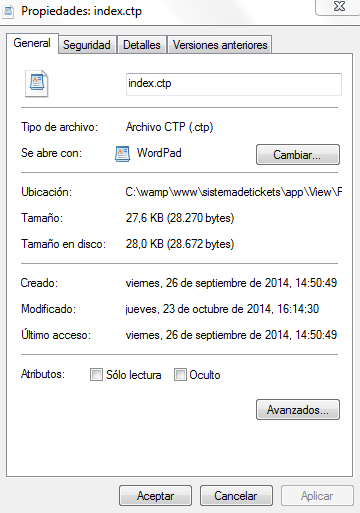
TIEMPO DE DESARROLLO + TIEMPO DE ULTIMA MODIFICACIÓN

TD = 1 HORA 44 MINUTOS = 1,73 HORAS

TUM= 13 MINUTOS = 0.22 HORAS

1,73 HORAS + 0.22 HORAS = 1, 95 HORAS

* /app/View/Roles/index.ctp



**Figura. 33-A.** Propiedad archivo Vista Rol

**Fuente:** María José Samaniego

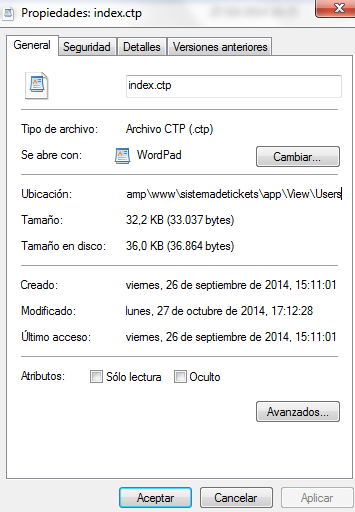
TIEMPO DE DESARROLLO + TIEMPO DE ULTIMA MODIFICACIÓN

TD = 2 HORAS 30 MINUTOS = 2,05 HORAS

TUM= 35 MIN = 0,58 HORAS

2,05 HORAS + 0,58 HORAS = 2,63 HORAS

* /app/View/Users/index.ctp



**Figura. 34-A.** Propiedad archivo Vista Usuario

**Fuente:** María José Samaniego

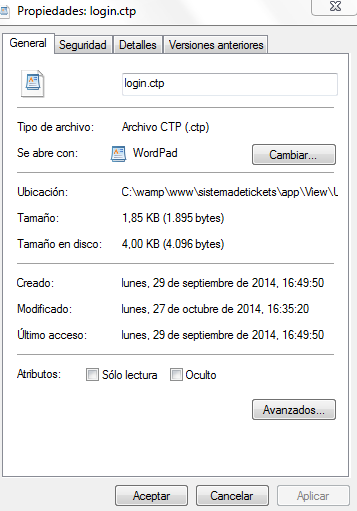
TIEMPO DE DESARROLLO + TIEMPO DE ULTIMA MODIFICACIÓN

TD = 5 HORAS 27 MINUTOS = 7,45 HORAS

TUM= 8 MINUTOS = 0.13 HORAS

5.45 HORAS + 0.13 HORAS = 5,58 HORAS

* /app/View/Users/login.ctp



**Figura. 35-A.** Propiedad archivo Login

**Fuente:** María José Samaniego

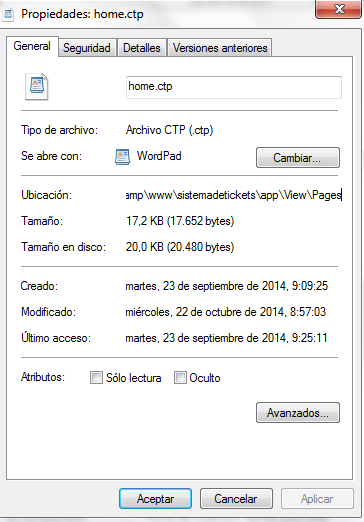
TIEMPO DE DESARROLLO + TIEMPO DE ULTIMA MODIFICACIÓN

TD = 31 MINUTOS = 0,52 HORAS

TUM= 45 MINUTOS = 0.75 HORAS

0,52 HORAS + 0.75 HORAS = 1,27 HORAS

* /app/View/Pages/home.ctp



**Figura. 36-A.** Propiedad archivo vista home

**Fuente**: María José Samaniego

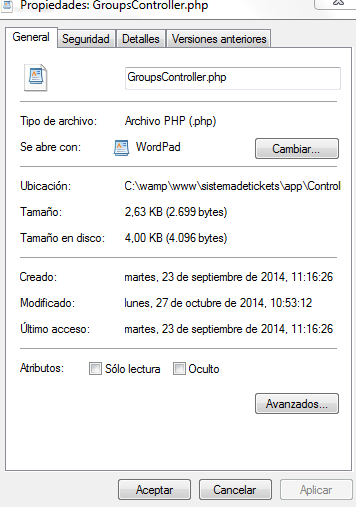
TIEMPO DE DESARROLLO + TIEMPO DE ULTIMA MODIFICACIÓN

TD = 1 HORAS 12 MINUTOS = 1,02 HORAS

TUM= 1 HORAS 57 MINUTOS = 2.95 HORAS

1,02 HORAS + 1.95 HORAS = 1,97 HORAS

* /app/Controller/GroupsController.php



**Figura. 37-A.** Propiedad archivo Controlador Grupo

**Fuente:** María José Samaniego

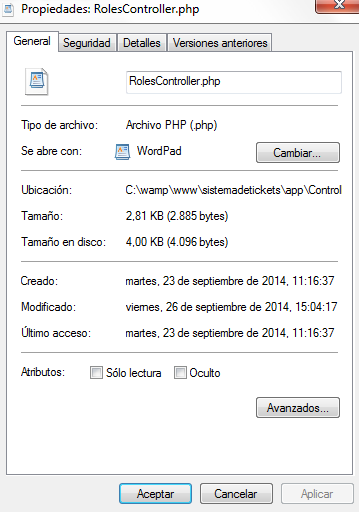
TIEMPO DE DESARROLLO + TIEMPO DE ULTIMA MODIFICACIÓN

TD = 6 MINUTOS = 0.1 HORAS

TUM= 1 HORAS 57 MINUTOS = 1.95 HORAS

0,1 HORAS + 1.95 HORAS = 1,96 HORA

* /app/Controller/RolesController.php



**Figura. 38-A.** Propiedad archivo Controlador Rol

**Fuente:** María José Samaniego

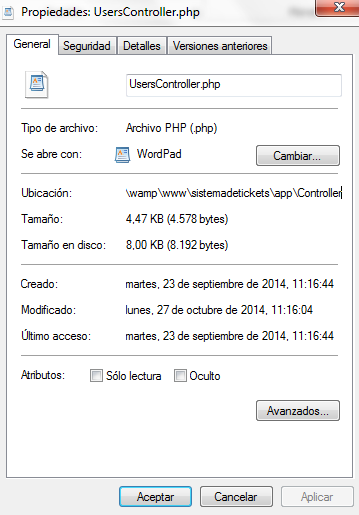
TIEMPO DE DESARROLLO + TIEMPO DE ULTIMA MODIFICACIÓN

TD = 6 MINUTOS = 0.1 HORAS

TUM= 16 MINUTOS = 0.27 HORAS

0,1 HORAS + 0.27 HORAS = 0.28 HORAS

* /app/Controller/UsersController.php



**Figura. 39-A.** Propiedad archivo Controlador Usuario

**Fuente:** María José Samaniego

TIEMPO DE DESARROLLO + TIEMPO DE ULTIMA MODIFICACIÓN

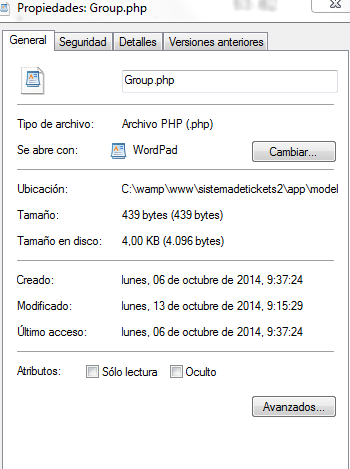
TD = 6 MINUTOS = 0.1 HORAS

TUM= 30 MINUTOS = 0.5 HORAS

0,1 HORAS + 0.5 HORAS = 0.6 HORAS

**Laravel**

* /app/models/Group.php



**Figura. 40-A.** Propiedad archivo Grupo

**Fuente:** María José Samaniego

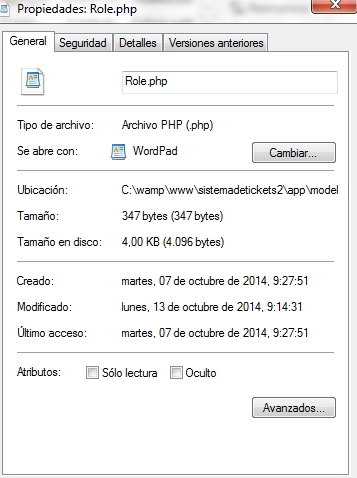
TIEMPO DE DESARROLLO + TIEMPO DE ULTIMA MODIFICACIÓN

TD = 2 HORAS 23 MINUTOS = 2,38 HORAS

TUM= 70 MINUTOS = 1.17 HORAS

2,38 HORAS + 1,17 HORAS = 3.55 HORAS

* /app/models/Role.php



**Figura. 41-A.** Propiedad archivo Rol

**Fuente**: María José Samaniego

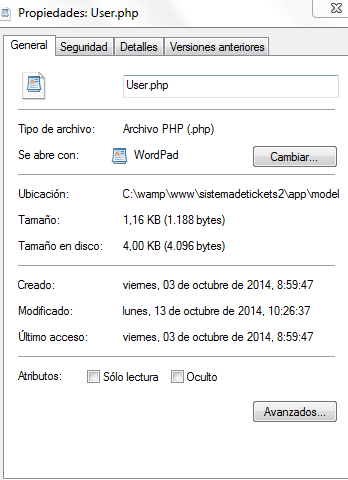
TIEMPO DE DESARROLLO + TIEMPO DE ULTIMA MODIFICACIÓN

TD = 2 HORAS 9 MINUTOS = 2,15 HORAS

TUM= 1 MINUTO = 0,02 HORAS

2,15 HORAS + 0,02 HORAS = 2,17 HORAS

* /app/models/User.php



**Figura. 42-A.** Propiedad archivo Usuario

**Fuente**: María José Samaniego

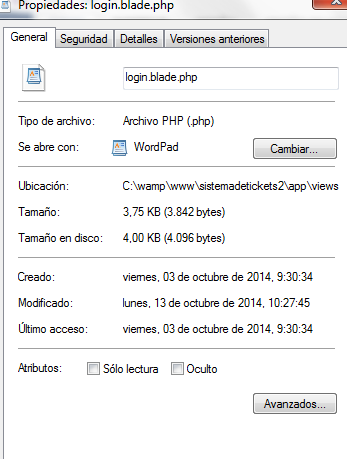
TIEMPO DE DESARROLLO + TIEMPO DE ULTIMA MODIFICACIÓN

TD = 49 MINUTOS = 0,82 HORAS

TUM= 1 MINUTO = 0,02 HORAS

0,82 HORAS + 0,02 HORAS = 0,84 HORAS

* /app/views/users/login.blade.php



**Figura. 42-A.** Propiedad archivo Vista Login

**Fuente:** María José Samaniego

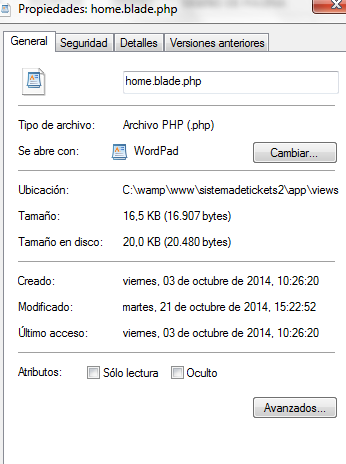
TIEMPO DE DESARROLLO + TIEMPO DE ULTIMA MODIFICACIÓN

TD = 2 HORAS 56 MINUTOS = 2,93 HORAS

TUM= 1 HORA 33 MINUTOS= 1,55 HORAS

2,93 HORAS + 1,55 HORAS = 4,48 HORAS

* /app/views/home.blade.php



**Figura. 43-A.** Propiedad archivo Vista Home

**Fuente:** María José Samaniego

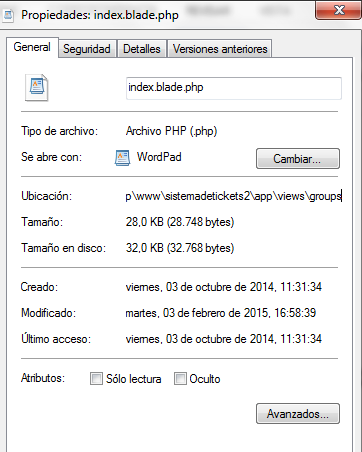
TIEMPO DE DESARROLLO + TIEMPO DE ULTIMA MODIFICACIÓN

TD = 1 HORA 34 MINUTOS = 1,57 HORAS

TUM= 2 HORAS 1 MINUTO = 2,02 HORAS

1,57 HORAS + 2,02 HORAS = 3,59 HORAS

* /app//views/groups/index.blade.php



**Figura. 44-A.** Propiedad archivo Vista Grupos

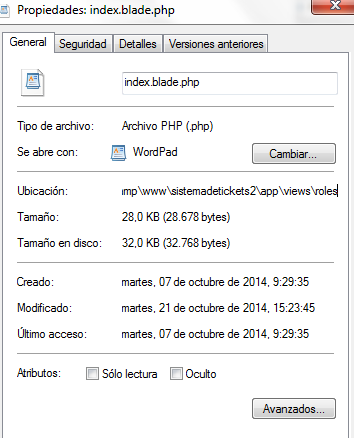
**Fuente:** María José Samaniego

TIEMPO DE DESARROLLO + TIEMPO DE ULTIMA MODIFICACIÓN

TD = 8 HORAS 48 MINUTOS = 8,08 HORAS

TUM= 32 MINUTOS = 0,53 HORAS

8,08 HORAS + 0,53 HORAS = 8,61 HORAS

* /app/views/roles/index.blade.php
* 

**Figura. 45-A.** Propiedad archivo Vista Rol

**Fuente:** María José Samaniego

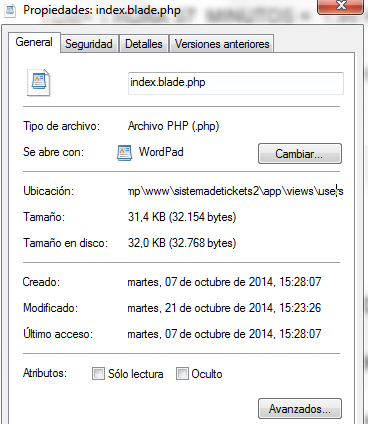
TIEMPO DE DESARROLLO + TIEMPO DE ULTIMA MODIFICACIÓN

TD = 2 HORA 7 MINUTOS = 2,12 HORAS

TUM= 1 HORA 57 MINUTOS = 1,95 HORAS

2,12 HORAS + 1,95 HORAS = 4,07 HORAS

* /app/views/users/index.blade.php



**Figura. 46-A.** Propiedad archivo Vista Usuarios

**Fuente:** María José Samaniego

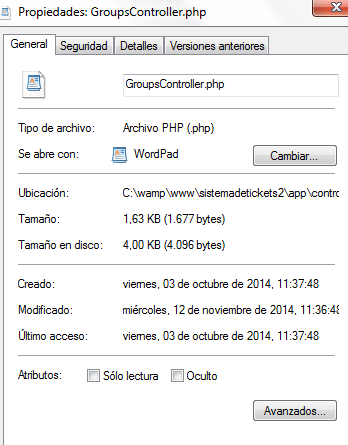
TIEMPO DE DESARROLLO + TIEMPO DE ULTIMA MODIFICACIÓN

TD = 5 HORAS 52 MINUTOS = 3,87 HORAS

TUM= 1 HORA 57 MINUTOS = 1,95 HORAS

5,87 HORAS + 1,95 HORAS = 7,82 HORAS

* /app/controllers/GroupsController.php



**Figura. 47-A.** Propiedad archivo Controlador Grupo

**Fuente:** María José Samaniego

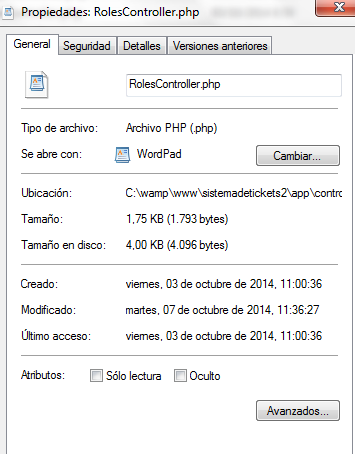
TIEMPO DE DESARROLLO + TIEMPO DE ULTIMA MODIFICACIÓN

TD = 1 HORA 23 MINUTOS = 1,38 HORAS

TUM= 1 HORA 24 MINUTOS = 1,4 HORAS

1,38 HORAS + 1,4 HORAS = 2,78 HORAS

* /app/controllers/RolesController.php



**Figura. 48-A.** Propiedad archivo Controlador Rol

**Fuente**: María José Samaniego

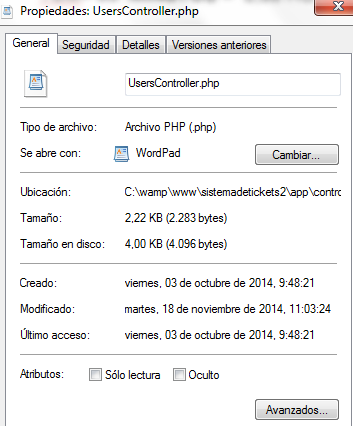
TIEMPO DE DESARROLLO + TIEMPO DE ULTIMA MODIFICACIÓN

TD = 1 HORAS 24 MINUTOS = 1,4 HORAS

TUM= 1 HORA 23 MINUTOS = 1,38 HORAS

1,4 HORAS + 1,38 HORAS = 2,78 HORAS

* /app/controllers/UsersController.php



**Figura. 49-A.** Propiedad archivo Controlador Usuario

**Fuente:** María José Samaniego

TIEMPO DE DESARROLLO + TIEMPO DE ULTIMA MODIFICACIÓN

TD = 1 HORA 2 MINUTOS = 1,03 HORAS

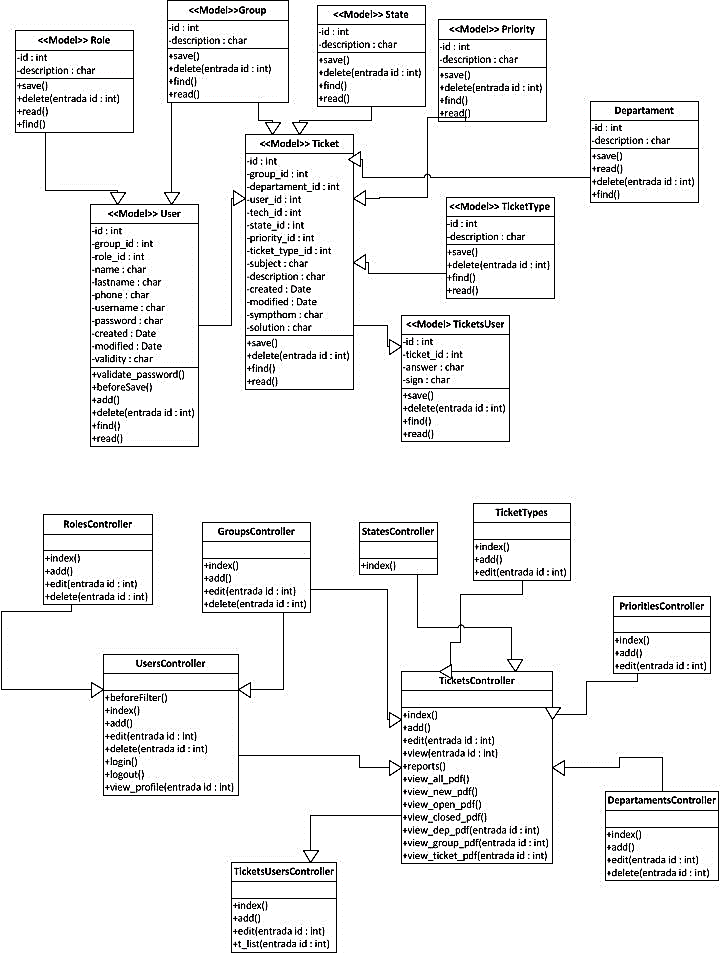
TUM= 57 MINUTOS = 0,95 HORAS

1,03 HORAS + 0,95 HORAS = 1,98 HORAS

**ANEXO B**

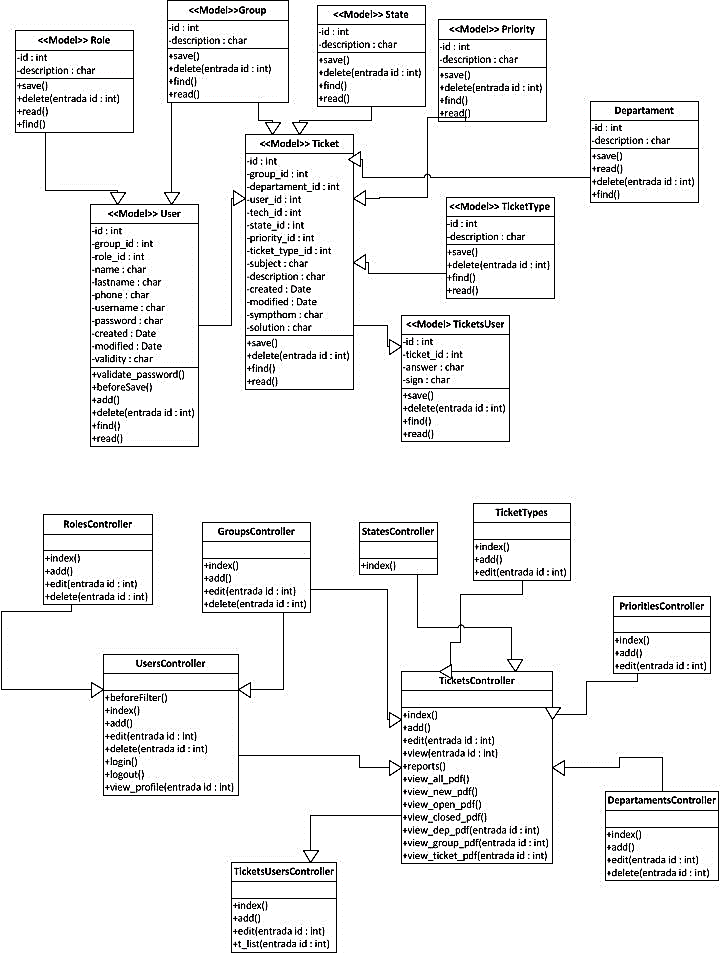
**Sistema de Seguimiento de Incidentes de la Infraestructura de Red de la ESPOCH**

**Anexo B1: Diagramas UML**

****

**Figura. 1-B.** Diagrama de Clases Modelo SDSI-DTIC

**Fuente:** María José Samaniego



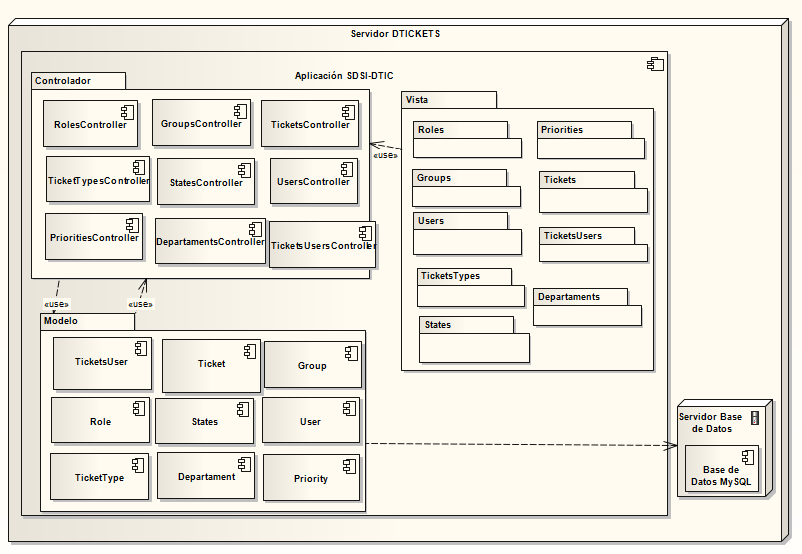
**Figura. 2-B.** Diagrama de Clases Controlador SDSI-DTIC

**Fuente:** María José Samaniego



**Figura. 3-B.** Diagrama de Despliegue SDSI-DTIC

**Fuente:** María José Samaniego



**Figura. 3-B.** Diagrama de Componentes SDSI-DTIC

**Fuente:** María José Samaniego

**Anexo B2: Diccionario de Datos**

**Tabla 1-B:** Tabla departaments

|  |  |  |  |
| --- | --- | --- | --- |
| **Columna** | **Tipo** | **Llave** | **Nulo** |
| id | int(5) | Primary Key | No |
| description | varchar(100) |  | No |

**Fuente:** María José Samaniego

**Tabla 2-B:** Tabla groups

|  |  |  |  |
| --- | --- | --- | --- |
| **Columna** | **Tipo** | **Llave** | **Nulo** |
| id | int(5) | Primary Key | No |
| description | varchar(50) |  | No |
| validity | varchar(25) |  | No |

**Fuente:** María José Samaniego

**Tabla 3-B:** Tabla priorities

|  |  |  |  |
| --- | --- | --- | --- |
| **Columna** | **Tipo** | **Llave** | **Nulo** |
| id | int(2) | Primary Key | No |
| description | varchar(25) |  | No |

**Fuente:** María José Samaniego

**Tabla 4-B:** Tabla roles

|  |  |  |  |
| --- | --- | --- | --- |
| **Columna** | **Tipo** | **Llave** | **Nulo** |
| id | int(5) | Primary Key | No |
| description | varchar(25) |  | No |
| validity | varchar(25) |  | No |

**Fuente:** María José Samaniego

**Tabla 5-B:** Tabla states

|  |  |  |  |
| --- | --- | --- | --- |
| **Columna** | **Tipo** | **Llave** | **Nulo** |
| id | int(5) | Primary Key | No |
| description | varchar(25) |  | No |

**Fuente:** María José Samaniego

**Tabla 6-B:** Tabla ticket\_types

|  |  |  |  |
| --- | --- | --- | --- |
| **Columna** | **Tipo** | **Llave** | **Nulo** |
| id | int(11) | Primary Key | No |
| description | varchar(50) |  | No |

**Fuente:** María José Samaniego

**Tabla 7-B:** Tabla tickets

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Columna** | **Tipo** | **Llave** | **Nulo** | **Predeterminado** |
| id | int(11) | Primary Key | No |  |
| ticket\_type\_id | int(11) |  | No |  |
| state\_id | int(11) |  | No |  |
| departament\_id | int(5) |  | No |  |
| group\_id | int(5) |  | No |  |
| priority\_id | varchar(25) |  | No |  |
| user\_id | int(5) |  | Sí | NULL |
| tech\_id | int(5) |  | Sí | NULL |
| subject | varchar(100) |  | No |  |
| description | text |  | No |  |
| created | datetime |  | No |  |
| modified | datetime |  | No |  |
| symptom | text |  | Sí | NULL |
| solution | text |  | Sí | NULL |

**Fuente:** María José Samaniego

**Tabla 8-B:** Tabla tickets\_users

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Columna** | **Tipo** | **Nulo** | **Llave** | **Predeterminado** |
| id | int(5) | No | Primary Key |  |
| ticket\_id | int(5) | No |  |  |
| answer | text | Sí |  | *NULL* |
| created | datetime | No |  |  |
| sign | varchar(56) | Sí |  | *NULL* |

**Fuente:** María José Samaniego

**Tabla 8-B:** Tabla users

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Columna** | **Tipo** | **Llave** | **Nulo** | **Predeterminado** |
| id | int(5) | Primary Key | No |  |
| name | varchar(50) |  | No |  |
| lastname | varchar(50) |  | No |  |
| email | varchar(50) |  | No |  |
| phone | int(11) |  | No |  |
| username | varchar(16) |  | No |  |
| password | varchar(255) |  | No |  |
| validity | varchar(25) |  | No |  |
| created | datetime |  | No |  |
| modified | datetime |  | No |  |
| last\_session | timestamps |  | No |  |
| role\_id | int(5) |  | No |  |
| group\_id | int(5) |  | No |  |
| departament\_id | int(5) |  | Sí | *NULL* |

**Fuente:** María José Samaniego

**Anexo B3: Sprint 0**

**Tabla 9-B:** Sprint Backlog – Iteration 0

|  |  |  |  |
| --- | --- | --- | --- |
| **ITERACIÓN 0 PROYECTO: SDSI-DTIC** | | | |
| **ID** | **Tipo** | **Descripción** | **Estado** |
| **HT1** | **Historia**  **Técnica** | **Diseño del esquema de la base de datos** | **Completo** |
|  | Tarea | Creación del diagrama entidad relación UML | Completo |
|  | Tarea | Creación de la base de datos | Completo |
| **HT2** | **Historia**  **Técnica** | **Diseño de la Arquitectura general del Sistema** | **Completo** |
| **HT3** | **Historia**  **Técnica** | **Codificación del Sistema** | **Completo** |
| **HT4** | **Historia**  **Técnica** | **Diseño de las Interfaces de Usuario** | **Completo** |
| **HT5** | **Historia**  **Técnica** | **Creación del Product Backlog** | **Completo** |

**Fuente:** María José Samaniego

**Tabla 10-B:** Historias de Técnica 1

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HT1 |
| **Identificador:** TI2 |
| **Nombre de la Tarea:** Creación de la Base de datos |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**  Se creará las distintas tablas con sus respectivos atributos, tipos de datos y relaciones |

**Fuente:** María José Samaniego

**Tabla 11-B:** Historias de Técnica 2

|  |  |
| --- | --- |
| **Historia Técnica** | |
| **Identificador:** HT2 | |
| **Nombre de la Historia:** Diseño de la Arquitectura general del Sistema | |
| **Prioridad en Negocio:** Muy Alta  (Muy Alta/Alta / Media / Baja) | **Riesgo en Desarrollo:** Normal  (Alta / Normal/Medio/ Bajo) |
| **Puntos Estimados:** 5 | **Iteración Asignada:** 0 |
| **Programador Responsable:** María Jose Samaniego Larrea | |
| **Descripción:**  Como equipo de desarrollo se necesita crear el diseño de la arquitectura general del sistema. | |
| **Observaciones:** | |

**Fuente:** María José Samaniego

**Tabla 12-B:** Historias de Técnica 3

|  |  |
| --- | --- |
| **Historia Técnica** | |
| **Identificador:** HT3 | |
| **Nombre de la Historia:** Codificación del Sistema | |
| **Prioridad en Negocio:** Muy Alta  (Muy Alta/Alta / Media / Baja) | **Riesgo en Desarrollo:** Normal  (Alta / Normal/Medio/ Bajo) |
| **Puntos Estimados:** 5 | **Iteración Asignada:** 0 |
| **Programador Responsable:** María Jose Samaniego Larrea | |
| **Descripción:**  Como equipo de desarrollo se necesita establecer el estándar de codificación para el desarrollo del SDSI-DTIC: | |
| **Observaciones:** | |

**Fuente:** María José Samaniego

**Tabla 13-B:** Historias de Técnica 4

|  |  |
| --- | --- |
| **Historia Técnica** | |
| **Identificador:** HT4 | |
| **Nombre de la Historia:** Codificación del Sistema | |
| **Prioridad en Negocio:** Muy Alta  (Muy Alta/Alta / Media / Baja) | **Riesgo en Desarrollo:** Normal  (Alta / Normal/Medio/ Bajo) |
| **Puntos Estimados:** 5 | **Iteración Asignada:** 0 |
| **Programador Responsable:** María Jose Samaniego Larrea | |
| **Descripción:**  Como equipo de desarrollo se desea establecer un estándar para la codificación del sistema siguiendo las convenciones de CakePHP | |
| **Observaciones:** | |

**Fuente:** María José Samaniego

**Tabla 14-B:** Historias de Técnica 5

|  |  |
| --- | --- |
| **Historia Técnica** | |
| **Identificador:** HT5 | |
| **Nombre de la Historia:** Creación del Product Backlog | |
| **Prioridad en Negocio:** Muy Alta  (Muy Alta/Alta / Media / Baja) | **Riesgo en Desarrollo:** Normal  (Alta / Normal/Medio/ Bajo) |
| **Puntos Estimados:** 5 | **Iteración Asignada:** 0 |
| **Programador Responsable:** María Jose Samaniego Larrea | |
| **Descripción:**  Como equipo de desarrollo se necesita crear la pila del producto con la lista de requerimientos del cliente. | |
| **Observaciones:** | |

**Fuente:** María José Samaniego

**Anexo B4: Sprint 1**

**Tabla 14-B:** Sprint Backlog Iteración 2

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Sprint Backlog Iteration 1 del SDSI-DTIC** | | | | | | | | | |
| **ID** | **Tipo** | | | **Descripción** | | | | **Estado** | |
| **HU1** | **Historia de Usuario** | | | **El sistema debe permitir el ingreso mediante autentificación de usuarios.** | | | | **Completo** | |
|  | Tarea | | | Conexión con la base de datos desde la aplicación. | | | | Completo | |
|  | Tarea | | | Creación del modelo usuario. | | | | Completo | |
|  | Tarea | | | Creación del controlador para el modelo usuario. | | | | Completo | |
|  | Tarea | | | Creación de la vista login. | | | | Completo | |
|  | Tarea | | | Validación de formularios login. | | | | Completo | |
|  | Tarea | | | Prueba Unitaria | | | | Completo | |
| **HU2** | **Historia de Usuario** | | | **Como administrador deseo ingresar y visualizar los datos de los usuarios que tendrán acceso al sistema.** | | | | **Completo** | |
|  | Tarea | | | Configuración del controlador del modelo usuario | | | | Completo | |
|  | Tarea | | | Creación de la vista para insertar usuario. | | | | Completo | |
|  | Tarea | | | Validación de formulario. | | | | Completo | |
|  | Tarea | | | Prueba Unitaria | | | | Completo | |
| **HU3** | **Historia de Usuario** | | | **Como administrador deseo modificar los datos de los usuarios** | | | | **Completo** | |
|  | Tarea | | | Configuración del controlador del modelo usuario | | | | Completo | |
|  | Tarea | | | Creación de la vista para insertar usuario. | | | | Completo | |
|  | Tarea | | | Validación de formulario. | | | | Completo | |
|  | Tarea | | | Prueba Unitaria | | | | Completo | |
| **HU4** | **Historia de Usuario** | | | **Como administrador deseo dar validez o no a un usuario** | | | | **Completo** | |
|  | Tarea | | | Configuración del controlador del modelo usuario | | | | Completo | |
|  | Tarea | | | Creación de la vista para insertar usuario. | | | | Completo | |
|  | Tarea | | | Validación de formulario. | | | | Completo | |
|  | Tarea | | | Prueba Unitaria | | | | Completo | |
| **HU5** | | **Historia de Usuario** | | | **Como administrador deseo modificar la información de los roles de trabajo de los usuarios** | | **Completo** | | |
|  | | Tarea | | | Creación del modelo roles. | | Completo | | |
|  | | Tarea | | | Configuración del controlador del modelo rol. | | Completo | | |
|  | | Tarea | | | Creación de la vista para actualizar los roles. | | Completo | | |
|  | | Tarea | | | Validación de formularios del modelo rol. | | Completo | | |
|  | | Tarea | | | Prueba Unitaria | | Completo | | |
| **HU6** | | **Historia de Usuario** | | | **Como administrador deseo dar validez o no a los roles de trabajo de los usuarios.** | | **Completo** | | |
|  | | Tarea | | | Configuración del controlador del modelo rol. | | Completo | | |
|  | | Tarea | | | Creación de la vista para actualizar los roles. | | Completo | | |
|  | | Tarea | | | Validación de formularios del modelo rol. | | Completo | | |
|  | | Tarea | | | Prueba Unitaria | | Completo | | |
| **HU7** | | **Historia de Usuario** | | | **Como administrador deseo modificar la información de los grupos o áreas de trabajo.** | | **Completo** | | |
|  | | Tarea | | | Creación del modelo grupos | | Completo | | |
|  | | Tarea | | | Configuración del controlador del modelo grupo | | Completo | | |
|  | | Tarea | | | Creación de formulario de actualización de grupo. | | Completo | | |
|  | | Tarea | | | Validación de formularios del modelo. | | Completo | | |
|  | | Tarea | | | Prueba Unitaria | | Completo | | |
| **HU8** | | **Historia de Usuario** | | | **Como administrador deseo dar validez o no a los grupos o áreas de trabajo.** | | **Completo** | | |
|  | | Tarea | | | Configuración del controlador del modelo grupo | | Completo | | |
|  | | Tarea | | | Modificación de la vista para actualizar grupo. | | Completo | | |
|  | | Tarea | | | Validación de formularios del modelo. | | Completo | | |
| **HU8** | | **Historia de Usuario** | | | **Como administrador deseo ingresar la información de las facultades o dependencias existentes en la ESPOCH** | | **Completo** | | |
|  | | Tarea | | | Creación del modelo | | Completo | | |
|  | | Tarea | | | Configuración del controlador | | Completo | | |
|  | | Tarea | | | Creación de formulario. | | Completo | | |
|  | | Tarea | | | Validación de formularios del modelo. | | Completo | | |
|  | | Tarea | | | Prueba Unitaria | | Completo | | |
| **U10** | | **Historia de Usuario** | | | **Como administrador deseo modificar la información de las facultades o dependencias existentes en la ESPOCH** | | **Completo** | | |
|  | | Tarea | | | Configuración del controlador | | Completo | | |
|  | | Tarea | | | Creación de formulario. | | Completo | | |
|  | | Tarea | | | Validación de formularios del modelo. | | Completo | | |
|  | | Tarea | | | Prueba Unitaria | | Completo | | |
| **HU11** | | **Historia de Usuario** | | | **Como administrador deseo eliminar a una facultad o dependencia de la ESPOCH de la base de datos del sistema SDSI-DTIC.** | | **Completo** | | |
|  | | Tarea | | | Configuración del controlador | | Completo | | |
|  | | Tarea | | | Creación de formulario. | | Completo | | |
|  | | Tarea | | | Validación de formularios del modelo. | | Completo | | |
|  | | Tarea | | | Prueba Unitaria | | Completo | | |
| **HU12** | | **Historia de Usuario** | | | **Como administrador deseo crear un nuevo tipo de ticket la cual será contenida en la generación de uno, por parte de los usuarios.** | | **Completo** | | |
|  | | Tarea | | | Configuración del controlador | | Completo | | |
|  | | Tarea | | | Creación de formulario. | | Completo | | |
|  | | Tarea | | | Validación de formularios del modelo. | | Completo | | |
|  | | Tarea | | | Prueba Unitaria | | Completo | | |
| **HU13** | | **Historia de Usuario** | | | **Como administrador deseo modificar la información de un tipo de ticket la cual será contenida en la generación de uno, por parte de los usuarios** | | **Completo** | | |
|  | | Tarea | | | Configuración del controlador | | Completo | | |
|  | | Tarea | | | Creación de formulario. | | Completo | | |
|  | | Tarea | | | Validación de formularios del modelo. | | Completo | | |
|  | | Tarea | | | Prueba Unitaria | | Completo | | |
| **HU14** | | **Historia de Usuario** | | | **Como administrador deseo visualizar la información correspondiente a los estados posibles que tendrá un ticket.** | | **Completo** | | |
|  | | Tarea | | | Configuración del controlador | | Completo | | |
|  | | Tarea | | | Creación de la vista. | | Completo | | |
|  | | Tarea | | | Validación de formularios del modelo. | | Completo | | |
|  | | | Tarea | | | Prueba Unitaria | | | Completo |
| **HU15** | | | **Historia de Usuario** | | | **Como administrador deseo crear una nueva prioridad de ticket.** | | | **Completo** |
|  | | | Tarea | | | Configuración del controlador | | | Completo |
|  | | | Tarea | | | Creación de formulario. | | | Completo |
|  | | | Tarea | | | Validación de formularios del modelo. | | | Completo |
|  | | | Tarea | | | Prueba Unitaria | | | Completo |
| **HU16** | | | **Historia de Usuario** | | | **Como administrador deseo modificar la información de una prioridad de ticket, la cual será contenida en la generación de uno, por parte de los usuarios.** | | | **Completo** |
|  | | | Tarea | | | Configuración del controlador | | | Completo |
|  | | | Tarea | | | Creación de formulario. | | | Completo |
|  | | | Tarea | | | Validación de formularios del modelo. | | | Completo |
|  | | | Tarea | | | Prueba Unitaria | | | Completo |

**Fuente:** María José Samaniego

**Tabla 15-B:** Historias de Usuario 1

|  |  |
| --- | --- |
| **Historia de Usuario** | |
| **Identificador:** HU1 | **Usuario:** Técnico DTIC |
| **Nombre de la Historia:** Autentificar Usuario | |
| **Prioridad en Negocio:** Muy Alta  (Muy Alta/Alta / Media / Baja) | **Riesgo en Desarrollo:** Alta  (Alta / Normal/Medio/ Bajo) |
| **Puntos Estimados:** 3 | **Iteración Asignada:** 1 |
| **Programador Responsable:** María Jose Samaniego Larrea | |
| **Descripción:**  Como Técnico DTIC deseo autentificarme al sistema, para poder acceder a las funcionalidades asignadas. | |
| **Observaciones:** | |

**Fuente:** María José Samaniego

**Tabla 16-B:** Tarea de Ingeniería 3

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU1 |
| **Identificador:** TI3 |
| **Nombre de la Tarea:** Conexión con la base de datos desde la aplicación. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**  Se configurara el archivo database.php, y se procederá a ingresar los datos respectivos a la base de datos creada para el sistema. |

**Fuente:** María José Samaniego

**Tabla 17-B:** Tarea de Ingeniería 4

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU1 |
| **Identificador:** TI4 |
| **Nombre de la Tarea:** Creación del modelo usuario. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**  Se creará el modelo, con las respectivas relaciones y validaciones del mismo. |

**Fuente:** María José Samaniego

**Tabla 18-B:** Tarea de Ingeniería 5

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU1 |
| **Identificador:** TI5 |
| **Nombre de la Tarea:** Creación del controlador para el modelo usuario. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**  Se creará el controlador del modelo, con las respectivas funciones y métodos para la autentificación de usuarios al sistema. |

**Fuente:** María José Samaniego

**Tabla 19-B:** Tarea de Ingeniería 6

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU1 |
| **Identificador:** TI6 |
| **Nombre de la Tarea:** Creación de la vista login. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**  Se creará la plantilla .ctp para la autentificación de usuarios. |

**Fuente:** María José Samaniego

**Tabla 20-B:** Prueba de Aceptación 5

|  |  |
| --- | --- |
| **Prueba de Aceptación** | |
| **Identificador:** PA5 | |
| **Nombre de la Prueba:** Autentificación de Usuario | |
| **N° de Historia que pertenece:** HU1 | |
| **Descripción:**  Este permitirá que el usuario se puede autentificar en el sistema para hacer uso de las funcionalidades del sistema. | |
| **Precondiciones:**  Este se llevara a cabo si el usuario se encuentra registrado en la base de datos. | **Ejecución:**   1. Ingresar al Sistema 2. Ingresar el nombre de usuario correcto y contraseña correcto 3. Presionar Iniciar Sesión. |
| **Post condiciones:**  El sistema permite el acceso al usuario, identificándolo correctamente y mostrándole la pantalla de bienvenida. | **Evaluación de Prueba:**  Exitosa |

**Fuente:** María José Samaniego

**Tabla 21-B:** Historia de Usuario 2

|  |  |
| --- | --- |
| **Historia de Usuario** | |
| **Identificador:** HU2 | **Usuario:** Administrador |
| **Nombre de la Historia:** Creación y Visualización de Usuarios | |
| **Prioridad en Negocio:** Muy Alta  (Muy Alta/Alta / Media / Baja) | **Riesgo en Desarrollo:** Normal  (Alta / Normal/Medio/ Bajo) |
| **Puntos Estimados:** 3 | **Iteración Asignada:** 1 |
| **Programador Responsable:** María Jose Samaniego Larrea | |
| **Descripción:**  Como administrador deseo ingresar y visualizar usuarios que tendrán acceso al sistema. | |
| **Observaciones:** | |

**Fuente:** María José Samaniego

**Tabla 22-B:** Tarea de Ingeniería 7

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU2 |
| **Identificador:** TI7 |
| **Nombre de la Tarea:** Configuración del controlador del modelo usuario. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**  Se creará los métodos para el ingreso de usuarios al sistema. |

**Tabla 23-B:** Tarea de Ingeniería 8

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU2 |
| **Identificador:** TI8 |
| **Nombre de la Tarea:** Creación de la vista para el registro de un usuario. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**   * Se creará la plantilla .ctp para listar los usuarios que son ingresados. * Se creara la plantilla .ctp que mostrara la información correspondiente para el ingreso de un usuario. * Se mostrara un mensaje de confirmación de ingreso. |

**Fuente:** María José Samaniego

**Tabla 23-B:** Prueba de Aceptación 6

|  |  |
| --- | --- |
| **Prueba de Aceptación** | |
| **Identificador:** PA6 | |
| **Nombre de la Prueba:** Registro de Usuarios | |
| **N° de Historia que pertenece:** HU2 | |
| **Descripción:**  Este permitirá ingresar los datos de los usuarios y almacenarlos en la base de datos. | |
| **Precondiciones:**  Este se llevara a cabo si se encuentra creada el formulario de ingreso. | **Ejecución:**   1. Ingresar al Sistema 2. Seleccionar Administración de personas, Usuarios. 3. Presionar el botón Nuevo. 4. Llenar los datos del formulario. 5. Clic en Guardar. |
| **Post condiciones:**  El sistema guarda el nuevo Usuario y retorna al index de los Usuarios registrados, con un mensaje de información “Usuario Guardado Exitosamente”. | **Evaluación de Prueba:**  Exitosa |

**Fuente:** María José Samaniego

**Tabla 24-B:** Historia de Usuario 3

|  |  |
| --- | --- |
| **Historia de Usuario** | |
| **Identificador:** HU3 | **Usuario:** Administrador |
| **Nombre de la Historia:** Modificación de Información de Usuarios | |
| **Prioridad en Negocio:** Alta  (Muy Alta/Alta / Media / Baja) | **Riesgo en Desarrollo:** Normal  (Alta / Normal/Medio/ Bajo) |
| **Puntos Estimados:** 3 | **Iteración Asignada:** 1 |
| **Programador Responsable:** María Jose Samaniego Larrea | |
| **Descripción:**  Como administrador deseo modificar los datos de los usuarios. | |
| **Observaciones:** | |

**Fuente:** María José Samaniego

**Tabla 25-B:** Tarea de Ingeniería 9

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU3 |
| **Identificador:** TI9 |
| **Nombre de la Tarea:** Configuración del controlador del modelo usuario. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**  Se creará el método para la actualización de datos del usuario. |

**Fuente:** María José Samaniego

**Tabla 26-B:** Tarea de Ingeniería 10

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU3 |
| **Identificador:** TI10 |
| **Nombre de la Tarea:** Creación de la vista para la actualización de un usuario. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**   * Se creara la plantilla .ctp para la actualización de información un usuario. * Se mostrara un mensaje de confirmación. |

**Fuente:** María José Samaniego

**Tabla 27-B:** Prueba de Aceptación 7

|  |  |
| --- | --- |
| **Prueba de Aceptación** | |
| **Identificador:** PA7 | |
| **Nombre de la Prueba:** Modificación de información de Usuarios | |
| **N° de Historia que pertenece:** HU3 | |
| **Descripción:**  Este permitirá modificar los datos de los usuarios y almacenarlos en la base de datos. | |
| **Precondiciones:**  Este se llevara a cabo si el usuario se encuentra registrado en la base de datos. | **Ejecución:**   1. Ingresar al Sistema 2. Seleccionar Administración de personas, Usuarios. 3. Presionar el botón Editar. 4. Modificar los datos del formulario. 5. Clic en Guardar. |
| **Post condiciones:**  El sistema guarda la información del usuario y retorna al index de los Usuarios registrados, con un mensaje de información “Usuario Guardado Exitosamente”. | **Evaluación de Prueba:**  Exitosa |

**Fuente:** María José Samaniego

**Tabla 28-B:** Historia de Usuario 4

|  |  |
| --- | --- |
| **Historia de Usuario** | |
| **Identificador:** HU4 | **Usuario:** Administrador |
| **Nombre de la Historia:** Cambiar validez a usuarios | |
| **Prioridad en Negocio:** Alta  (Muy Alta/Alta / Media / Baja) | **Riesgo en Desarrollo:** Medio  (Alta / Normal/Medio/ Bajo) |
| **Puntos Estimados:** 3 | **Iteración Asignada:** 1 |
| **Programador Responsable:** María Jose Samaniego Larrea | |
| **Descripción:**  Como administrador deseo dar validez a un usuario para que este pueda atender un ticket. | |
| **Observaciones:** | |

**Fuente:** María José Samaniego

**Tabla 29-B:** Tarea de Ingeniería 11

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU4 |
| **Identificador:** TI11 |
| **Nombre de la Tarea:** Configuración del controlador del modelo usuario. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**  Se creará el método para la actualización de datos del usuario. |

**Fuente:** María José Samaniego

**Tabla 30-B:** Tarea de Ingeniería 12

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU4 |
| **Identificador:** TI12 |
| **Nombre de la Tarea:** Creación de la vista para la actualización de un usuario. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**   * Se creara la plantilla .ctp para la actualización de información un usuario. * Se mostrara un mensaje de confirmación. |

**Fuente:** María José Samaniego

**Tabla 31-B:** Prueba de Aceptación 8

|  |  |
| --- | --- |
| **Prueba de Aceptación** | |
| **Identificador:** PA8 | |
| **Nombre de la Prueba:** Validez de Usuarios | |
| **N° de Historia que pertenece:** HU4 | |
| **Descripción:**  Este permitirá modificar la validez de usuarios y almacenarlos en la base de datos. | |
| **Precondiciones:**  Este se llevara a cabo si el usuario se encuentra registrado en la base de datos. | **Ejecución:**   1. Ingresar al Sistema 2. Seleccionar Administración de personas, Usuarios. 3. Presionar el botón Editar. 4. Modificar los datos del formulario. 5. Clic en Guardar. |
| **Post condiciones:**  El sistema guarda la información del usuario y retorna al index de los Usuarios registrados, con un mensaje de información “Usuario Guardado Exitosamente”. | **Evaluación de Prueba:**  Exitosa |

**Fuente:** María José Samaniego

**Tabla 32-B:** Historia de Usuario 5

|  |  |
| --- | --- |
| **Historia de Usuario** | |
| **Identificador:** HU5 | **Usuario:** Administrador |
| **Nombre de la Historia:** Modificar Roles de Usuario | |
| **Prioridad en Negocio:** Alta  (Muy Alta/Alta / Media / Baja) | **Riesgo en Desarrollo:** Normal  (Alta / Normal/Medio/ Bajo) |
| **Puntos Estimados:** 3 | **Iteración Asignada:** 1 |
| **Programador Responsable:** María Jose Samaniego Larrea | |
| **Descripción:**  Como administrador deseo modificar la información de los roles de trabajo de los usuarios | |
| **Observaciones:** | |

**Fuente:** María José Samaniego

**Tabla 33-B:** Tarea de Ingeniería 13

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU5 |
| **Identificador:** TI13 |
| **Nombre de la Tarea:** Creación del modelo rol. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**  Se creará el modelo, con las respectivas relaciones y validaciones del mismo |

**Fuente:** María José Samaniego

**Tabla 34-B:** Tarea de Ingeniería 14

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU5 |
| **Identificador:** TI14 |
| **Nombre de la Tarea:** Configuración del controlador del modelo rol. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**  Se creará el método para la actualización de información del rol. |

**Fuente:** María José Samaniego

**Tabla 35-B:** Tarea de Ingeniería 15

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU5 |
| **Identificador:** TI15 |
| **Nombre de la Tarea:** Creación de la vista para la actualización del rol. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**   * Se creara la plantilla .ctp para la actualización de información un rol. * Se mostrara un mensaje de confirmación. |

**Fuente:** María José Samaniego

**Tabla 36-B:** Prueba de Aceptación 9

|  |  |
| --- | --- |
| **Prueba de Aceptación** | |
| **Identificador:** PA9 | |
| **Nombre de la Prueba:** Modificación de información de roles | |
| **N° de Historia que pertenece:** HU5 | |
| **Descripción:**  Este permitirá modificar los datos de los roles y almacenarlos en la base de datos. | |
| **Precondiciones:**  Este se llevara a cabo si el rol se encuentra registrado en la base de datos. | **Ejecución:**   1. Ingresar al Sistema 2. Seleccionar Administración de personas, Roles. 3. Presionar el botón Editar. 4. Modificar los datos del formulario. 5. Clic en Guardar. |
| **Post condiciones:**  El sistema guarda la información del usuario y retorna al index de los Usuarios registrados, con un mensaje de información “Rol Guardado Exitosamente”. | **Evaluación de Prueba:**  Exitosa |

**Fuente:** María José Samaniego

**Tabla 37-B:** Historia de Usuario 6

|  |  |
| --- | --- |
| **Historia de Usuario** | |
| **Identificador:** HU6 | **Usuario:** Administrador |
| **Nombre de la Historia:** Cambiar validez del rol | |
| **Prioridad en Negocio:** Alta  (Muy Alta/Alta / Media / Baja) | **Riesgo en Desarrollo:** Medio  (Alta / Normal/Medio/ Bajo) |
| **Puntos Estimados:** 3 | **Iteración Asignada:** 1 |
| **Programador Responsable:** María Jose Samaniego Larrea | |
| **Descripción:**  Como administrador deseo dar validez o no a los roles de trabajo de los usuarios. | |
| **Observaciones:** | |

**Fuente:** María José Samaniego

**Tabla 38-B:** Tarea de Ingeniería 16

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU6 |
| **Identificador:** TI16 |
| **Nombre de la Tarea:** Configuración del controlador del modelo rol. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**  Se creará el método para la actualización de información del rol. |

**Fuente:** María José Samaniego

**Tabla 39-B:** Tarea de Ingeniería 17

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU6 |
| **Identificador:** TI17 |
| **Nombre de la Tarea:** Creación de la vista para la actualización del rol. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**   * Se creara la plantilla .ctp para la actualización de información un rol. * Se mostrara un mensaje de confirmación. |

**Fuente:** María José Samaniego

**Tabla 40-B:** Prueba de Aceptación 10

|  |  |
| --- | --- |
| **Prueba de Aceptación** | |
| **Identificador:** PA10 | |
| **Nombre de la Prueba:** Validez de información de roles | |
| **N° de Historia que pertenece:** HU6 | |
| **Descripción:**  Este permitirá modificar los datos de los roles y almacenarlos en la base de datos. | |
| **Precondiciones:**  Este se llevara a cabo si el rol se encuentra registrado en la base de datos. | **Ejecución:**   1. Ingresar al Sistema 2. Seleccionar Administración de personas, Roles. 3. Presionar el botón Editar. 4. Modificar los datos del formulario. 5. Clic en Guardar. |
| **Post condiciones:**  El sistema guarda la información del usuario y retorna al index de los Usuarios registrados, con un mensaje de información “Rol Guardado Exitosamente”. | **Evaluación de Prueba:**  Exitosa |

**Fuente:** María José Samaniego

**Tabla 41-B:** Historia de Usuario 7

|  |  |
| --- | --- |
| **Historia de Usuario** | |
| **Identificador:** HU7 | **Usuario:** Administrador |
| **Nombre de la Historia:** Modificar Grupos de Trabajo | |
| **Prioridad en Negocio:** Alta  (Muy Alta/Alta / Media / Baja) | **Riesgo en Desarrollo:** Normal  (Alta / Normal/Medio/ Bajo) |
| **Puntos Estimados:** 3 | **Iteración Asignada:** 1 |
| **Programador Responsable:** María Jose Samaniego Larrea | |
| **Descripción:**  Como administrador deseo modificar la información de los grupos o áreas de trabajo. | |
| **Observaciones:** | |

**Fuente:** María José Samaniego

**Tabla 42-B:** Tarea de Ingeniería 18

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU7 |
| **Identificador:** TI18 |
| **Nombre de la Tarea:** Creación del modelo grupo. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**  Se creará el modelo, con las respectivas relaciones y validaciones del mismo |

**Fuente:** María José Samaniego

**Tabla 43-B:** Tarea de Ingeniería 19

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU7 |
| **Identificador:** TI19 |
| **Nombre de la Tarea:** Configuración del controlador del modelo grupo. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**  Se creará el método para la actualización de información del grupo. |

**Fuente:** María José Samaniego

**Tabla 44-B:** Tarea de Ingeniería 20

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU7 |
| **Identificador:** TI20 |
| **Nombre de la Tarea:** Creación de la vista para la actualización del grupo. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**   * Se creara la plantilla .ctp para la actualización de información un rol. * Se mostrara un mensaje de confirmación. |

**Fuente:** María José Samaniego

**Tabla 45-B:** Prueba de Aceptación 11

|  |  |
| --- | --- |
| **Prueba de Aceptación** | |
| **Identificador:** PA11 | |
| **Nombre de la Prueba :** Modificación de información de grupos | |
| **N° de Historia que pertenece:** HU7 | |
| **Descripción:**  Este permitirá modificar los datos de los roles y almacenarlos en la base de datos. | |
| **Precondiciones:**  Este se llevara a cabo si el grupo se encuentra registrado en la base de datos. | **Ejecución:**   1. Ingresar al Sistema 2. Seleccionar Administración de personas, Grupos. 3. Presionar el botón Editar. 4. Modificar los datos del formulario. 5. Clic en Guardar. |
| **Post condiciones:**  El sistema guarda la información del usuario y retorna al index de los Usuarios registrados, con un mensaje de información “Grupo de Trabajo Guardado Exitosamente”. | **Evaluación de Prueba:**  Exitosa |

**Fuente:** María José Samaniego

**Tabla 46-B:** Historia de Usuario 8

|  |  |
| --- | --- |
| **Historia de Usuario** | |
| **Identificador:** HU8 | **Usuario:** Administrador |
| **Nombre de la Historia:** Cambiar validez de Grupos de Trabajo | |
| **Prioridad en Negocio:** Alta  (Muy Alta/Alta / Media / Baja) | **Riesgo en Desarrollo:** Medio  (Alta / Normal/Medio/ Bajo) |
| **Puntos Estimados:** 3 | **Iteración Asignada:** 1 |
| **Programador Responsable:** María Jose Samaniego Larrea | |
| **Descripción:**  Como administrador deseo dar validez o no a los grupos o áreas de trabajo. | |
| **Observaciones:** | |

**Fuente:** María José Samaniego

**Tabla 47-B:** Tarea de Ingeniería 21

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU8 |
| **Identificador:** TI21 |
| **Nombre de la Tarea:** Configuración del modelo grupo. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**  Se creará el modelo, con las respectivas relaciones y validaciones del mismo |

**Fuente:** María José Samaniego

**Tabla 48-B:** Tarea de Ingeniería 22

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU8 |
| **Identificador:** TI22 |
| **Nombre de la Tarea:** Configuración del controlador del modelo grupo. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**  Se creará el método para la actualización de información del grupo. |

**Fuente:** María José Samaniego

**Tabla 49-B:** Prueba de Aceptación 12

|  |  |
| --- | --- |
| **Prueba de Aceptación** | |
| **Identificador:** PA12 | |
| **Nombre de la Prueba:** Validación de grupos. | |
| **N° de Historia que pertenece:** HU8 | |
| **Descripción:**  Este permitirá modificar los datos de los grupos y almacenarlos en la base de datos. | |
| **Precondiciones:**  Este se llevara a cabo si el grupo se encuentra registrado en la base de datos. | **Ejecución:**   1. Ingresar al Sistema 2. Seleccionar Administración de personas, Grupos. 3. Presionar el botón Editar. 4. Modificar los datos del formulario. 5. Clic en Guardar. |
| **Post condiciones:**  El sistema guarda la información del usuario y retorna al index de los Usuarios registrados, con un mensaje de información “Grupo Guardado Exitosamente”. | **Evaluación de Prueba:**  Exitosa |

**Fuente:** María José Samaniego

**Tabla 50-B:** Historia de usuario 9

|  |  |
| --- | --- |
| **Historia de Usuario** | |
| **Identificador:** HU9 | **Usuario:** Administrador |
| **Nombre de la Historia:** Ingresar Dependencias Facultades | |
| **Prioridad en Negocio:** Alta  (Muy Alta/Alta / Media / Baja) | **Riesgo en Desarrollo:** Normal  (Alta / Normal/Medio/ Bajo) |
| **Puntos Estimados:** 3 | **Iteración Asignada:** 1 |
| **Programador Responsable:** María Jose Samaniego Larrea | |
| **Descripción:**  Como administrador deseo ingresar la información de las facultades o dependencias existentes en la ESPOCH. | |
| **Observaciones:** | |

**Fuente:** María José Samaniego

**Tabla 51-B:** Tarea de Ingeniería 23

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU9 |
| **Identificador:** TI23 |
| **Nombre de la Tarea:** Creación del modelo departamento. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**  Se creará el modelo, con las respectivas relaciones y validaciones del mismo |

**Fuente:** María José Samaniego

**Tabla 52-B:** Tarea de Ingeniería 24

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU9 |
| **Identificador:** TI24 |
| **Nombre de la Tarea:** Configuración del controlador del modelo departamento. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**  Se creará el método para el ingreso de información del departamento. |

**Fuente:** María José Samaniego

**Tabla 53-B:** Tarea de Ingeniería 25

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU9 |
| **Identificador:** TI25 |
| **Nombre de la Tarea:** Creación de la vista para la actualización del departamento. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**   * Se creará la plantilla .ctp para listar los departamentos que son ingresados. * Se creara la plantilla .ctp que mostrara la información correspondiente para el ingreso de un departamento. * Se mostrara un mensaje de confirmación de ingreso. |

**Fuente:** María José Samaniego

**Tabla 54-B:** Prueba de Aceptación 13

|  |  |
| --- | --- |
| **Prueba de Aceptación** | |
| **Identificador:** PA13 | |
| **Nombre de la Prueba:** Registro de un departamento | |
| **N° de Historia que pertenece:** HU9 | |
| **Descripción:**  Este permitirá registrar un nuevo departamento y almacenarlo en la base de datos. | |
| **Precondiciones:**  Este se llevara a cabo si el departamento quiere utilizar el sistema. | **Ejecución:**   1. Ingresar al Sistema 2. Seleccionar Departamentos. 3. Presionar el botón Nuevo. 4. Ingresar los datos del formulario. 5. Clic en Guardar. |
| **Post condiciones:**  El sistema guarda la información del usuario y retorna al index de los Usuarios registrados, con un mensaje de información “Departamento Guardado Exitosamente”. | **Evaluación de Prueba:**  Exitosa |

**Fuente:** María José Samaniego

**Tabla 55-B:** Historia de Usuario 10

|  |  |
| --- | --- |
| **Historia de Usuario** | |
| **Identificador:** HU10 | **Usuario:** Administrador |
| **Nombre de la Historia:** Modificar Dependencias Facultades | |
| **Prioridad en Negocio:** Alta  (Muy Alta/Alta / Media / Baja) | **Riesgo en Desarrollo:** Normal  (Alta / Normal/Medio/ Bajo) |
| **Puntos Estimados:** 3 | **Iteración Asignada:** 1 |
| **Programador Responsable:** María Jose Samaniego Larrea | |
| **Descripción:**  Como administrador deseo modificar la información de las facultades o dependencias existentes en la ESPOCH. | |
| **Observaciones:** | |

**Fuente:** María José Samaniego

**Tabla 56-B:** Tarea de Ingeniería 26

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU10 |
| **Identificador:** TI26 |
| **Nombre de la Tarea:** Configuración del controlador del modelo departamento. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**  Se creará el método para la actualización de información del departamento. |

**Fuente:** María José Samaniego

**Tabla 57-B:** Tarea de Ingeniería 27

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU10 |
| **Identificador:** TI27 |
| **Nombre de la Tarea:** Creación de la vista para la actualización del departamento. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**   * Se creara la plantilla .ctp para la actualización de información un departamento. * Se mostrara un mensaje de confirmación. |

**Fuente:** María José Samaniego

**Tabla 58-B:** Prueba de Aceptación 14

|  |  |
| --- | --- |
| **Prueba de Aceptación** | |
| **Identificador:** PA14 | |
| **Nombre de la Prueba:** Modificación de información de un departamento | |
| **N° de Historia que pertenece:** HU10 | |
| **Descripción:**  Este permitirá modificar información departamento y almacenarlo en la base de datos. | |
| **Precondiciones:**  Este se llevara a cabo si el departamento se encuentra registrado en la base de datos. | **Ejecución:**   1. Ingresar al Sistema 2. Seleccionar Departamentos. 3. Presionar el botón Editar. 4. Modificar los datos del formulario. 5. Clic en Guardar. |
| **Post condiciones:**  El sistema guarda la información del usuario y retorna al index de los Usuarios registrados, con un mensaje de información “Departamento Guardado Exitosamente”. | **Evaluación de Prueba:**  Exitosa |

**Fuente:** María José Samaniego

**Tabla 59-B:** Historia de Usuario 11

|  |  |
| --- | --- |
| **Historia de Usuario** | |
| **Identificador:** HU11 | **Usuario:** Administrador |
| **Nombre de la Historia:** Eliminar Dependencias Facultades | |
| **Prioridad en Negocio:** Alta  (Muy Alta/Alta / Media / Baja) | **Riesgo en Desarrollo:** Normal  (Alta / Normal/Medio/ Bajo) |
| **Puntos Estimados:** 3 | **Iteración Asignada:** 1 |
| **Programador Responsable:** María Jose Samaniego Larrea | |
| **Descripción:**  Como administrador deseo eliminar a una facultad o dependencia de la ESPOCH de la base de datos del sistema SDSI-DTIC. | |
| **Observaciones:** | |

**Fuente:** María José Samaniego

**Tabla 60-B:** Tarea de Ingeniería 28

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU11 |
| **Identificador:** TI28 |
| **Nombre de la Tarea:** Configuración del controlador del modelo departamento. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**  Se creará el método para la eliminación lógica del departamento de la base de datos. |

**Fuente:** María José Samaniego

**Tabla 61-B:** Prueba de Aceptación 15

|  |  |
| --- | --- |
| **Prueba de Aceptación** | |
| **Identificador:** PA15 | |
| **Nombre de la Prueba:** Eliminación lógica de un departamento | |
| **N° de Historia que pertenece:** HU10 | |
| **Descripción:**  Este permitirá cambiar la validez de un departamento. | |
| **Precondiciones:**  Este se llevara a cabo si el departamento se encuentra registrado en la base de datos. | **Ejecución:**   1. Ingresar al Sistema 2. Seleccionar Departamentos. 3. Presionar el botón Editar. 4. Modificar el campo de validez del formulario. 5. Clic en Guardar. |
| **Post condiciones:**  El sistema guarda la información del usuario y retorna al index de los Usuarios registrados, con un mensaje de información “Departamento Guardado Exitosamente”. | **Evaluación de Prueba:**  Exitosa |

**Fuente:** María José Samaniego

**Tabla 62-B:** Historia de Usuario 12

|  |  |
| --- | --- |
| **Historia de Usuario** | |
| **Identificador:** HU12 | **Usuario:** Administrador |
| **Nombre de la Historia:** Creación de Tipos de Ticket | |
| **Prioridad en Negocio:** Alta  (Muy Alta/Alta / Media / Baja) | **Riesgo en Desarrollo:** Normal  (Alta / Normal/Medio/ Bajo) |
| **Puntos Estimados:** 3 | **Iteración Asignada:** 1 |
| **Programador Responsable:** María Jose Samaniego Larrea | |
| **Descripción:**  Como administrador deseo crear un nuevo tipo de ticket la cual será contenida en la generación de uno, por parte de los usuarios. | |
| **Observaciones:** | |

**Fuente:** María José Samaniego

**Tabla 63-B:** Tarea de Ingeniería 29

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU12 |
| **Identificador:** TI29 |
| **Nombre de la Tarea:** Creación del modelo tipo de ticket. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**  Se creará el modelo, con las respectivas relaciones y validaciones del mismo |

**Fuente:** María José Samaniego

**Tabla 64-B:** Tarea de Ingeniería 30

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU12 |
| **Identificador:** TI30 |
| **Nombre de la Tarea:** Configuración del controlador del modelo tipo de ticket. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**  Se creará el método para el ingreso de información del tipo de ticket. |

**Fuente:** María José Samaniego

**Tabla 65-B:** Tarea de Ingeniería 31

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU12 |
| **Identificador:** TI31 |
| **Nombre de la Tarea:** Creación de la vista para el ingreso del tipo de ticket. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**   * Se creara la plantilla .ctp para el ingreso de información un tipo de ticket. * Se mostrara un mensaje de confirmación. |

**Fuente:** María José Samaniego

**Tabla 66-B:** Prueba de Aceptación 16

|  |  |
| --- | --- |
| **Prueba de Aceptación** | |
| **Identificador:** PA16 | |
| **Nombre de la Prueba:** Registro de un tipo de ticket | |
| **N° de Historia que pertenece:** HU12 | |
| **Descripción:**  Este permitirá registrar un nuevo tipo de ticket y almacenarlo en la base de datos. | |
| **Precondiciones:**  Este se llevara a cabo si el tipo de ticket si el administrador desee agregar un nuevo tipo de ticket como requerimiento. | **Ejecución:**   1. Ingresar al Sistema 2. Seleccionar Administración de Tickets. 3. Seleccionar Tipo de Ticket. 4. Presionar el botón Nuevo. 5. Ingresar los datos del formulario. 6. Clic en Guardar. |
| **Post condiciones:**  El sistema guarda la información del usuario y retorna al index de los Usuarios registrados, con un mensaje de información “Tipo de ticket Guardado Exitosamente”. | **Evaluación de Prueba:**  Exitosa |

**Fuente:** María José Samaniego

**Tabla 67-B:** Historia de Usuario 13

|  |  |
| --- | --- |
| **Historia de Usuario** | |
| **Identificador:** HU13 | **Usuario:** Administrador |
| **Nombre de la Historia:** Modificar Tipo de Ticket | |
| **Prioridad en Negocio:** Alta  (Muy Alta/Alta / Media / Baja) | **Riesgo en Desarrollo:** Normal  (Alta / Normal/Medio/ Bajo) |
| **Puntos Estimados:** 3 | **Iteración Asignada:** 1 |
| **Programador Responsable:** María Jose Samaniego Larrea | |
| **Descripción:**  Como administrador deseo modificar la información de un tipo de ticket la cual será contenida en la generación de uno, por parte de los usuarios | |
| **Observaciones:** | |

**Fuente:** María José Samaniego

**Tabla 68-B:** Tarea de Ingeniería 32

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU13 |
| **Identificador:** TI32 |
| **Nombre de la Tarea:** Configuración del controlador del modelo tipo de ticket. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**  Se creará el método para la actualización de información del tipo de ticket. |

**Fuente:** María José Samaniego

**Tabla 69-B:** Tarea de Ingeniería 33

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU13 |
| **Identificador:** TI33 |
| **Nombre de la Tarea:** Creación de la vista para la actualización del tipo de ticket. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**   * Se creara la plantilla .ctp para la actualización de información un tipo de ticket. * Se mostrara un mensaje de confirmación. |

**Fuente:** María José Samaniego

**Tabla 70-B:** Prueba de Aceptación 17

|  |  |
| --- | --- |
| **Prueba de Aceptación** | |
| **Identificador:** PA17 | |
| **Nombre de la Prueba:** Modificación de un tipo de ticket | |
| **N° de Historia que pertenece:** HU13 | |
| **Descripción:**  Este permitirá registrar un nuevo tipo de ticket y almacenarlo en la base de datos. | |
| **Precondiciones:**  Este se llevara a cabo si el tipo de ticket se encuentra registrado en la base de datos. | **Ejecución:**   1. Ingresar al Sistema 2. Seleccionar Administración de Tickets. 3. Seleccionar Tipo de ticket. 4. Presionar el botón Editar. 5. Ingresar los datos del formulario. 6. Clic en Guardar. |
| **Post condiciones:**  El sistema guarda la información del usuario y retorna al index de los Usuarios registrados, con un mensaje de información “Tipo de ticket Guardado Exitosamente”. | **Evaluación de Prueba:**  Exitosa |

**Fuente:** María José Samaniego

**Tabla 71-B:** Historia de Usuario 14

|  |  |
| --- | --- |
| **Historia de Usuario** | |
| **Identificador:** HU14 | **Usuario:** Administrador |
| **Nombre de la Historia:** Visualizar los estados disponibles para un Ticket | |
| **Prioridad en Negocio:** Alta  (Muy Alta/Alta / Media / Baja) | **Riesgo en Desarrollo:** Normal  (Alta / Normal/Medio/ Bajo) |
| **Puntos Estimados:** 3 | **Iteración Asignada:** 1 |
| **Programador Responsable:** María Jose Samaniego Larrea | |
| **Descripción:**  Como administrador deseo visualizar la información correspondiente a los estados posibles que tendrá un ticket. | |
| **Observaciones:** | |

**Fuente:** María José Samaniego

**Tabla 72-B:** Tarea de Ingeniería 34

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU14 |
| **Identificador:** TI34 |
| **Nombre de la Tarea:** Configuración del controlador del modelo estados. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**  Se creará el método para visualización de información de los estados ingresados. |

**Fuente:** María José Samaniego

**Tabla 73-B:** Tarea de Ingeniería 35

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU14 |
| **Identificador:** TI35 |
| **Nombre de la Tarea:** Creación de la vista para la actualización de estados. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**   * Se creara la plantilla .ctp para la actualización de información un estados. * Se mostrara un mensaje de confirmación. |

**Fuente:** María José Samaniego

**Tabla 74-B:** Prueba de Aceptación 18

|  |  |
| --- | --- |
| **Prueba de Aceptación** | |
| **Identificador:** PA18 | |
| **Nombre de la Prueba:** Visualización de un estados | |
| **N° de Historia que pertenece:** HU14 | |
| **Descripción:**  Este permitirá visualizar la lista de todos los estados e un ticket ingresados. | |
| **Precondiciones:**  Este se llevara a cabo si los estados se encuentran registrados en la base de datos. | **Ejecución:**   1. Ingresar al Sistema 2. Seleccionar Administración de Tickets. 3. Seleccionar Estados. |
| **Post condiciones:**  El sistema muestra la lista de los estados registrados en la base de datos con su correspondiente información. | **Evaluación de Prueba:**  Exitosa |

**Fuente:** María José Samaniego

**Tabla 75-B:** Historia de Usuario 15

|  |  |
| --- | --- |
| **Historia de Usuario** | |
| **Identificador:** HU15 | **Usuario:** Administrador |
| **Nombre de la Historia:** Creación de Prioridades para un Ticket | |
| **Prioridad en Negocio:** Alta  (Muy Alta/Alta / Media / Baja) | **Riesgo en Desarrollo:** Normal  (Alta / Normal/Medio/ Bajo) |
| **Puntos Estimados:** 3 | **Iteración Asignada:** 1 |
| **Programador Responsable:** María Jose Samaniego Larrea | |
| **Descripción:**  Como administrador deseo crear una nueva prioridad de ticket para poder asignarla a un incidente. | |
| **Observaciones:** | |

**Fuente:** María José Samaniego

**Tabla 76-B:** Tarea de Ingeniería 36

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU15 |
| **Identificador:** TI36 |
| **Nombre de la Tarea:** Creación del modelo prioridades. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**  Se creará el modelo, con las respectivas relaciones y validaciones del mismo |

**Fuente:** María José Samaniego

**Tabla 77-B:** Tarea de Ingeniería 37

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU15 |
| **Identificador:** TI37 |
| **Nombre de la Tarea:** Configuración del controlador del modelo prioridades. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**  Se creará el método para el ingreso de información de prioridades. |

**Fuente:** María José Samaniego

**Tabla 78-B:** Tarea de Ingeniería 38

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU15 |
| **Identificador:** TI38 |
| **Nombre de la Tarea:** Creación de la vista para el ingreso de prioridades. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**   * Se creara la plantilla .ctp para el ingreso de prioridades. * Se mostrara un mensaje de confirmación. |

**Fuente:** María José Samaniego

**Tabla 79-B:** Prueba de Aceptación 19

|  |  |
| --- | --- |
| **Prueba de Aceptación** | |
| **Identificador:** PA19 | |
| **Nombre de la Prueba:** Registro de una prioridad | |
| **N° de Historia que pertenece:** HU15 | |
| **Descripción:**  Este permitirá registrar una nueva prioridad del ticket y almacenarlo en la base de datos. | |
| **Precondiciones:**  Este se llevara a cabo si el administrador desea añadir una nueva prioridad para asignarla al ticket. | **Ejecución:**   1. Ingresar al Sistema 2. Seleccionar Administración de Tickets. 3. Seleccionar Prioridades del Ticket. 4. Presionar el botón Nuevo. 5. Ingresar los datos del formulario. 6. Clic en Guardar. |
| **Post condiciones:**  El sistema guarda la información del usuario y retorna al index de los Usuarios registrados, con un mensaje de información “Prioridad Guardada Exitosamente”. | **Evaluación de Prueba:**  Exitosa |

**Fuente:** María José Samaniego

**Tabla 80-B:** Historia de Usuario 16

|  |  |
| --- | --- |
| **Historia de Usuario** | |
| **Identificador:** HU16 | **Usuario:** Administrador |
| **Nombre de la Historia:** Modificar Información de Prioridad | |
| **Prioridad en Negocio:** Alta  (Muy Alta/Alta / Media / Baja) | **Riesgo en Desarrollo:** Normal  (Alta / Normal/Medio/ Bajo) |
| **Puntos Estimados:** 3 | **Iteración Asignada:** 1 |
| **Programador Responsable:** María Jose Samaniego Larrea | |
| **Descripción:**  Como administrador deseo modificar la información de una prioridad de ticket. | |
| **Observaciones:** | |

**Fuente:** María José Samaniego

**Tabla 81-B:** Tarea de Ingeniería 39

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU16 |
| **Identificador:** TI39 |
| **Nombre de la Tarea:** Configuración del controlador del modelo prioridades. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**  Se creará el método para la modificación de información de prioridades. |

**Fuente:** María José Samaniego

**Tabla 82-B:** Tarea de Ingeniería 39

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU15 |
| **Identificador:** TI39 |
| **Nombre de la Tarea:** Creación de la vista para la modificación de prioridades. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**   * Se creara la plantilla .ctp para la modificación de prioridades. * Se mostrara un mensaje de confirmación. |

**Fuente:** María José Samaniego

**Tabla 83-B:** Prueba de Aceptación 20

|  |  |
| --- | --- |
| **Prueba de Aceptación** | |
| **Identificador:** PA20 | |
| **Nombre de la Prueba:** Modificación de una prioridad | |
| **N° de Historia que pertenece:** HU16 | |
| **Descripción:**  Este permitirá modificar una prioridad del ticket y almacenarlo en la base de datos. | |
| **Precondiciones:**  Este se llevara a cabo si el la prioridad se encuentra registrada en la base de datos. | **Ejecución:**   1. Ingresar al Sistema 2. Seleccionar Administración de Tickets. 3. Seleccionar Prioridades del Ticket. 4. Presionar el botón Editar. 5. Ingresar los datos del formulario. 6. Clic en Guardar. |
| **Post condiciones:**  El sistema guarda la información del usuario y retorna al index de los Usuarios registrados, con un mensaje de información “Prioridad Guardada Exitosamente”. | **Evaluación de Prueba:**  Exitosa |

**Fuente:** María José Samaniego

**Anexo B5: Sprint 2**

**Tabla 84-B:** Sprint Backlog Iteración 2

|  |  |  |  |
| --- | --- | --- | --- |
| **SPRINT BACKLOG ITERACIÓN 2 PROYECTO: SDSI-DTIC** | | | |
| **ID** | **Tipo** | **Descripción** | **Estado** |
|  | **Tarea** | **Depuración del producto entregado en el Sprint 2.** | **Completo** |
| **HU17** | **Historia de Usuario** | **Como técnico deseo visualizar la lista de todos los tickets asignados.** | **Completo** |
|  | Tarea | Creación del modelo Ticket. | Completo |
|  | Tarea | Configuración del controlador de Tickets | Completo |
|  | Tarea | Creación de la vista para la lista de Tickets. | Completo |
|  | Tarea | Pruebas unitarias. | Completo |
| **HU18** | **Historia de Usuario** | **Como técnico deseo generar un ticket o caso.** | **Completo** |
|  | Tarea | Configuración del controlador del Ticket. | Completo |
|  | Tarea | Creación de la vista para generar un Ticket. | Completo |
|  | Tarea | Prueba Unitaria | Completo |
| **HU19** | **Historia de Usuario** | **Como técnico deseo visualizar el ticket generado con su correspondiente información.** | **Completo** |
|  | Tarea | Configuración del controlador del Ticket. | Completo |
|  | Tarea | Creación de la vista para atender un Ticket. | Completo |
|  | Tarea | Prueba Unitaria | Completo |
| **HU20** | **Historia de Usuario** | **Como técnico de facultad deseo atender un ticket a las correspondientes áreas DTIC.** | **Completo** |
|  | Tarea | Configuración del controlador del Ticket. | Completo |
|  | Tarea | Creación de la vista para atender un Ticket. | Completo |
|  | Tarea | Prueba Unitaria | Completo |
| **HU21** | **Historia de Usuario** | **Como director DTIC deseo transferir un ticket o caso a un área.** | **Completo** |
|  | Tarea | Configuración del controlador del Ticket. | Completo |
|  | Tarea | Modificación de la vista para atender un Ticket. | Completo |
|  | Tarea | Prueba Unitaria | Completo |
| **HU22** | **Historia de Usuario** | **Como jefe de área deseo asignarme el ticket o asignarlo a un técnico.** | **Completo** |
|  | Tarea | Configuración del controlador del Ticket. | Completo |
|  | Tarea | Modificación de la vista para atender un Ticket. | Completo |
|  | Tarea | Prueba Unitaria | Completo |
| **HU23** | **Historia de Usuario** | **Como técnico deseo asignarme un ticket.** | **Completo** |
|  | Tarea | Configuración del controlador del Ticket. | Completo |
|  | Tarea | Modificación de la vista para atender un Ticket. | Completo |
|  | Tarea | Prueba Unitaria | Completo |
| **HU24** | **Historia de Usuario** | **Como jefe de área deseo transferir un ticket a otra área de DTIC.** | **Completo** |
|  | Tarea | Configuración del controlador del Ticket. | Completo |
|  | Tarea | Modificación de la vista para atender un Ticket. | Completo |
|  | Tarea | Prueba Unitaria | Completo |
| **HU25** | **Historia de Usuario** | **Como técnico deseo llevar una bitácora de un Ticket o Caso.** | **Completo** |
|  | Tarea | Configuración del controlador del Ticket. | Completo |
|  | Tarea | Modificación de la vista para atender un Ticket. | Completo |
|  | Tarea | Prueba Unitaria | Completo |
| **HU26** | **Historia de Usuario** | **Como técnico deseo cambiar de estado del ticket.** | **Completo** |
|  | Tarea | Configuración del controlador del Ticket. | Completo |
|  | Tarea | Modificación de la vista para atender un Ticket. | Completo |
|  | Tarea | Prueba Unitaria | Completo |
| **HU27** | **Historia de Usuario** | **Como técnico deseo realizar búsquedas de tickets.** | **Completo** |
|  | Tarea | Configuración del controlador del Ticket. | Completo |
|  | Tarea | Modificación de la vista para atender un Ticket. | Completo |
|  | Tarea | Prueba Unitaria | Completo |

**Fuente:** María José Samaniego

**Tabla 85-B:** Historia de Usuario 17

|  |  |
| --- | --- |
| **Historia de Usuario** | |
| **Identificador:** HU17 | **Usuario:** Técnico DTIC |
| **Nombre de la Historia:** Listar Tickets | |
| **Prioridad en Negocio:** Alta  (Muy Alta/Alta / Media / Baja) | **Riesgo en Desarrollo:** Normal  (Alta / Normal/Medio/ Bajo) |
| **Puntos Estimados:** 5 | **Iteración Asignada:** 2 |
| **Programador Responsable:** María Jose Samaniego Larrea | |
| **Descripción:**  Como técnico DTIC deseo visualizar la lista de todos los tickets asignados | |
| **Observaciones:** | |

**Fuente:** María José Samaniego

**Tabla 86-B:** Tarea de Ingeniería 40

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU17 |
| **Identificador:** TI40 |
| **Nombre de la Tarea:** Creación del modelo Ticket. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**  Se creará el modelo, con las respectivas relaciones y validaciones del mismo |

**Fuente:** María José Samaniego

**Tabla 87-B:** Tarea de Ingeniería 41

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU17 |
| **Identificador:** TI41 |
| **Nombre de la Tarea:** Configuración del controlador del modelo Ticket. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**  Se creará el método para listar los tickets ingresados. |

**Fuente:** María José Samaniego

**Tabla 88-B:** Tarea de Ingeniería 42

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU17 |
| **Identificador:** TI42 |
| **Nombre de la Tarea:** Creación de la vista para listar tickets. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**   * Se creara la plantilla .ctp para listar los tickets con su información y acciones. |

**Fuente:** María José Samaniego

**Tabla 89-B:** Prueba de Aceptación 21

|  |  |
| --- | --- |
| **Prueba de Aceptación** | |
| **Identificador:** PA21 | |
| **Nombre de la Prueba:** Listar Tickets | |
| **N° de Historia que pertenece:** HU17 | |
| **Descripción:**  Este permitirá listar los tickets que han sido generados. | |
| **Precondiciones:**  Este se llevara a cabo si existen tickets generados y almacenados en la base de datos. | **Ejecución:**   1. Ingresar al Sistema 2. Menú principal, Mis Tickets. |
| **Post condiciones:**  El sistema muestra la lista de los tickets con su información y acciones. | **Evaluación de Prueba:**  Exitosa |

**Fuente:** María José Samaniego

**Tabla 90-B:** Historia de Usuario 18

|  |  |
| --- | --- |
| **Historia de Usuario** | |
| **Identificador:** HU18 | **Usuario:** Técnicos Facultad. |
| **Nombre de la Historia:** Generar Tickets | |
| **Prioridad en Negocio:** Alta  (Muy Alta/Alta / Media / Baja) | **Riesgo en Desarrollo:** Alta  (Alta / Normal/Medio/ Bajo) |
| **Puntos Estimados:** 2 | **Iteración Asignada:** 2 |
| **Programador Responsable:** María Jose Samaniego Larrea | |
| **Descripción:**  Como técnico de facultad deseo generar un ticket o caso, para obtener una solución. | |
| **Observaciones:** | |

**Fuente:** María José Samaniego

**Tabla 91-B:** Tarea de Ingeniería 43

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU18 |
| **Identificador:** TI43 |
| **Nombre de la Tarea:** Configuración del controlador del modelo Ticket. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**  Se creará el método para generar los tickets. |

**Fuente:** María José Samaniego

**Tabla 92-B:** Tarea de Ingeniería 44

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU18 |
| **Identificador:** TI44 |
| **Nombre de la Tarea:** Creación de la vista para generar un ticket. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**   * Se creara el formulario para generar tickets. * Se creara mensaje de confirmación. |

**Fuente:** María José Samaniego

**Tabla 93-B:** Prueba de Aceptación 22

|  |  |
| --- | --- |
| **Prueba de Aceptación** | |
| **Identificador:** PA22 | |
| **Nombre de la Prueba:** Generar Tickets | |
| **N° de Historia que pertenece:** HU18 | |
| **Descripción:**  Este permitirá generar tickets y almacenarlos en la base de datos. | |
| **Precondiciones:**  Este se llevara a cabo si existe un incidente que se desee reportar, y el usuario tenga acceso al sistema. | **Ejecución:**   1. Ingresar al Sistema 2. Menú principal, Mis Tickets. 3. Seleccionar pestana Nuevo ticket. 4. Llenar los campos 5. Generar Ticket. |
| **Post condiciones:**  El sistema mostrara un mensaje de confirmación “Ticket Generado”, y regresara al index. | **Evaluación de Prueba:**  Exitosa |

**Fuente:** María José Samaniego

**Tabla 94-B:** Historia de Usuario 19

|  |  |
| --- | --- |
| **Historia de Usuario** | |
| **Identificador:** HU19 | **Usuario:** Técnico Facultad |
| **Nombre de la Historia:** Visualizar información el Ticket | |
| **Prioridad en Negocio:** Alta  (Muy Alta/Alta / Media / Baja) | **Riesgo en Desarrollo:** Normal  (Alta / Normal/Medio/ Bajo) |
| **Puntos Estimados:** 2 | **Iteración Asignada:** 2 |
| **Programador Responsable:** María Jose Samaniego Larrea | |
| **Descripción:**  Como técnico de facultad deseo visualizar el ticket generado con su correspondiente información. | |
| **Observaciones:** | |

**Fuente:** María José Samaniego

**Tabla 95-B:** Tarea de Ingeniería 45

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU19 |
| **Identificador:** TI45 |
| **Nombre de la Tarea:** Configuración del controlador del modelo Ticket. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**  Se creará el método view para ver información del ticket seleccionado. |

**Fuente:** María José Samaniego

**Tabla 96-B:** Tarea de Ingeniería 46

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU19 |
| **Identificador:** TI46 |
| **Nombre de la Tarea:** Creación de la vista para visualizar información del ticket. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**   * Se creará la vista que contendrá la información del ticket |

**Fuente:** María José Samaniego

**Tabla 97-B:** Prueba de Aceptación 23

|  |  |
| --- | --- |
| **Prueba de Aceptación** | |
| **Identificador:** PA23 | |
| **Nombre de la Prueba:** Visualizar Ticket | |
| **N° de Historia que pertenece:** HU19 | |
| **Descripción:**  Este permitirá visualizar la información ingresada del ticket generado. | |
| **Precondiciones:**  Este se llevara a cabo si el ticket se encuentra almacenado en la base de datos y se muestra en la lista de tickets generados. | **Ejecución:**   1. Ingresar al Sistema 2. Menú principal, Mis Tickets. 3. Clic en el Ticket. |
| **Post condiciones:**  El sistema mostrara la vista de información del ticket. | **Evaluación de Prueba:**  Exitosa |

**Fuente:** María José Samaniego

**Tabla 98-B:** Historia de Usuario 20

|  |  |
| --- | --- |
| **Historia de Usuario** | |
| **Identificador:** HU20 | **Usuario:** Técnico Facultad |
| **Nombre de la Historia:** Atender un ticket | |
| **Prioridad en Negocio:** Alta  (Muy Alta/Alta / Media / Baja) | **Riesgo en Desarrollo:** Normal  (Alta / Normal/Medio/ Bajo) |
| **Puntos Estimados:** 5 | **Iteración Asignada:** 2 |
| **Programador Responsable:** María Jose Samaniego Larrea | |
| **Descripción:**  Como técnico de facultad deseo dar atención a un ticket que se me ha sido asignado para dar solución. | |
| **Observaciones:** | |

**Fuente:** María José Samaniego

**Tabla 99-B:** Tarea de Ingeniería 46

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU20 |
| **Identificador:** TI46 |
| **Nombre de la Tarea:** Configuración del controlador del modelo Ticket. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**  Se modificará el método de actualizar información del ticket para la atención del mismo. |

**Fuente:** María José Samaniego

**Tabla 100-B:** Tarea de Ingeniería 47

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU20 |
| **Identificador:** TI47 |
| **Nombre de la Tarea:** Modificación de la vista de visualización del Ticket. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**   * Se creará pestañas de navegación con las diferentes acciones para dar atención a un ticket. * Se crearan listas de información de acuerdo a los roles de usuario que se autentifican al sistema. * Se crearan mensajes de confirmación. |

**Fuente:** María José Samaniego

**Tabla 101-B:** Prueba de Aceptación 24

|  |  |
| --- | --- |
| **Prueba de Aceptación** | |
| **Identificador:** PA24 | |
| **Nombre de la Prueba:** Atención de un Ticket | |
| **N° de Historia que pertenece:** HU20 | |
| **Descripción:**  Este permitirá que se pueda dar atención a un ticket. | |
| **Precondiciones:**  Este se llevara a cabo si el ticket se encuentra almacenado en la base de datos y se muestra en la lista de tickets generados. | **Ejecución:**   1. Ingresar al Sistema 2. Menú principal, Mis Tickets. 3. Clic en el Ticket. |
| **Post condiciones:**  El sistema abrirá el ticket y mostrará su información, además de las pestañas de navegación. | **Evaluación de Prueba:**  Exitosa |

**Fuente:** María José Samaniego

**Tabla 102-B:** Historia de Usuario 21

|  |  |
| --- | --- |
| **Historia de Usuario** | |
| **Identificador:** HU21 | **Usuario:** Director DTIC |
| **Nombre de la Historia:** Transferir un ticket | |
| **Prioridad en Negocio:** Alta  (Muy Alta/Alta / Media / Baja) | **Riesgo en Desarrollo:** Normal  (Alta / Normal/Medio/ Bajo) |
| **Puntos Estimados:** 3 | **Iteración Asignada:** 2 |
| **Programador Responsable:** María Jose Samaniego Larrea | |
| **Descripción:**  Como Director DTIC deseo transferir un ticket para su inmediata atención. | |
| **Observaciones:** | |

**Fuente:** María José Samaniego

**Tabla 103-B:** Tarea de Ingeniería 48

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU21 |
| **Identificador:** TI48 |
| **Nombre de la Tarea:** Configuración del controlador del modelo Ticket. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**  Se modificará el método de actualizar información del ticket para la transferencia del mismo. |

**Fuente:** María José Samaniego

**Tabla 104-B:** Tarea de Ingeniería 49

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU21 |
| **Identificador:** TI49 |
| **Nombre de la Tarea:** Modificación de la vista de visualización del Ticket. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**   * Se creará una lista desplegable con las áreas DTIC. * Se controlará la información que se liste por medio del rol del usuario que se encuentre autentificado. * Se creará mensaje de confirmación de la transferencia del ticket. |

**Fuente:** María José Samaniego

**Tabla 105-B:** Prueba de Aceptación 25

|  |  |
| --- | --- |
| **Prueba de Aceptación** | |
| **Identificador:** PA25 | |
| **Nombre de la Prueba:** Transferencia de un Ticket | |
| **N° de Historia que pertenece:** HU21 | |
| **Descripción:**  Este permitirá que se pueda transferir un ticket a un área DTIC. | |
| **Precondiciones:**  Este se llevara a cabo si el ticket se encuentra almacenado en la base de datos y se muestra en la lista de tickets generados. | **Ejecución:**   1. Ingresar al Sistema 2. Menú principal, Mis Tickets. 3. Clic en el Ticket. 4. Seleccionar pestaña, Transferir a: 5. Seleccionar área. 6. Clic en Transferir. |
| **Post condiciones:**  El sistema mostrara un mensaje de confirmación “Información del ticket modificada” | **Evaluación de Prueba:**  Exitosa |

**Fuente:** María José Samaniego

**Tabla 106-B:** Historia de Usuario 22

|  |  |
| --- | --- |
| **Historia de Usuario** | |
| **Identificador:** HU22 | **Usuario:** Jefe de Área |
| **Nombre de la Historia:** Asignar ticket | |
| **Prioridad en Negocio:** Alta  (Muy Alta/Alta / Media / Baja) | **Riesgo en Desarrollo:** Normal  (Alta / Normal/Medio/ Bajo) |
| **Puntos Estimados:** 3 | **Iteración Asignada:** 2 |
| **Programador Responsable:** María Jose Samaniego Larrea | |
| **Descripción:**  Como jefe de área deseo asignarme el ticket o asignarlo a un técnico para su tratamiento. | |
| **Observaciones:** | |

**Fuente:** María José Samaniego

**Tabla 107-B:** Tarea de Ingeniería 50

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU22 |
| **Identificador:** TI50 |
| **Nombre de la Tarea:** Configuración del controlador del modelo Ticket. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**  Se modificará el método de actualizar información del ticket para la asignación del ticket a un técnico. |

**Fuente:** María José Samaniego

**Tabla 108-B:** Tarea de Ingeniería 51

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU22 |
| **Identificador:** TI51 |
| **Nombre de la Tarea:** Modificación de la vista de visualización del Ticket. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**   * Se creará una lista desplegable con los nombres de los técnicos que se encuentren disponibles dentro de cada área DTIC. * Se creará mensaje de confirmación de la transferencia del ticket. |

**Fuente:** María José Samaniego

**Tabla 109-B:** Prueba de Aceptación 26

|  |  |
| --- | --- |
| **Prueba de Aceptación** | |
| **Identificador:** PA26 | |
| **Nombre de la Prueba:** Asignación de un Ticket | |
| **N° de Historia que pertenece:** HU22 | |
| **Descripción:**  Este permitirá que se pueda asignar un ticket a un técnico para que le de atención. | |
| **Precondiciones:**  Este se llevara a cabo si el ticket se encuentra almacenado en la base de datos y se muestra en la lista de tickets generados. | **Ejecución:**   1. Ingresar al Sistema 2. Menú principal, Mis Tickets. 3. Clic en el Ticket. 4. Selección de la pestaña, Asignar a: 5. Seleccionar técnico. 6. Clic en Asignar. |
| **Post condiciones:**  El sistema mostrara un mensaje de confirmación “Información del ticket modificada” | **Evaluación de Prueba:**  Exitosa |

**Fuente:** María José Samaniego

**Tabla 110-B:** Historia de Usuario 25

|  |  |
| --- | --- |
| **Historia de Usuario** | |
| **Identificador:** HU25 | **Usuario:** Técnicos de Facultad. |
| **Nombre de la Historia:** Crear bitacora | |
| **Prioridad en Negocio:** Alta  (Muy Alta/Alta / Media / Baja) | **Riesgo en Desarrollo:** Normal  (Alta / Normal/Medio/ Bajo) |
| **Puntos Estimados:** 2 | **Iteración Asignada:** 2 |
| **Programador Responsable:** María Jose Samaniego Larrea | |
| **Descripción:**  Como técnico de facultad deseo llevar una bitácora de un Ticket, con su respectivo problema y solución. | |
| **Observaciones:** | |

**Fuente:** María José Samaniego

**Tabla 111-B:** Tarea de Ingeniería 52

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU25 |
| **Identificador:** TI52 |
| **Nombre de la Tarea:** Configuración del controlador del modelo Ticket. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**  Se modificará el método de actualizar información del ticket para la creación de la bitácora del ticket. |

**Fuente:** María José Samaniego

**Tabla 112-B:** Tarea de Ingeniería 53

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU25 |
| **Identificador:** TI53 |
| **Nombre de la Tarea:** Modificación de la vista de visualización del Ticket. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**   * Se creará el formulario para la creación de la bitacora del ticket, por medio del registro del problema y solución de ese caso. * Se creará mensaje de confirmación de la transferencia del ticket. |

**Fuente:** María José Samaniego

**Tabla 113-B:** Prueba de Aceptación 27

|  |  |
| --- | --- |
| **Prueba de Aceptación** | |
| **Identificador:** PA27 | |
| **Nombre de la Prueba:** Crear bitacora del Ticket. | |
| **N° de Historia que pertenece:** HU25 | |
| **Descripción:**  Este permitirá que se pueda llevar una bitacora el ticket. | |
| **Precondiciones:**  Este se llevara a cabo si el ticket se encuentra almacenado en la base de datos y se muestra en la lista de tickets generados. | **Ejecución:**   1. Ingresar al Sistema 2. Menú principal, Mis Tickets. 3. Clic en el Ticket. 4. Selección de la pestaña, Nota Interna: 5. Llenar los campos 6. Clic en Guardar Nota. |
| **Post condiciones:**  El sistema mostrara un mensaje de confirmación “Información del ticket modificada” | **Evaluación de Prueba:**  No muestra mensaje de confirmación. |

**Fuente:** María José Samaniego

**Tabla 114-B:** Historia de Usuario 26

|  |  |
| --- | --- |
| **Historia de Usuario** | |
| **Identificador:** HU26 | **Usuario:** Técnicos de Facultad. |
| **Nombre de la Historia:** Cambiar estado del ticket | |
| **Prioridad en Negocio:** Alta  (Muy Alta/Alta / Media / Baja) | **Riesgo en Desarrollo:** Normal  (Alta / Normal/Medio/ Bajo) |
| **Puntos Estimados:** 5 | **Iteración Asignada:** 2 |
| **Programador Responsable:** María Jose Samaniego Larrea | |
| **Descripción:**  Como técnico de facultad deseo cambiar de estado del ticket. | |
| **Observaciones:** | |

**Fuente:** María José Samaniego

**Tabla 115-B:** Tarea de Ingeniería 54

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU26 |
| **Identificador:** TI54 |
| **Nombre de la Tarea:** Configuración del controlador del modelo Ticket. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**  Se modificará el método de actualizar información del ticket para el cambio del estado de un ticket. |

**Fuente:** María José Samaniego

**Tabla 116-B:** Tarea de Ingeniería 55

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU26 |
| **Identificador:** TI55 |
| **Nombre de la Tarea:** Modificación de la vista de visualización del Ticket. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**   * Se creará el formulario para la creación de la bitacora del ticket, por medio del registro del problema y solución de ese caso. * Se creará mensaje de confirmación de la transferencia del ticket. |

**Fuente:** María José Samaniego

**Tabla 117-B:** Prueba de Aceptación 28

|  |  |
| --- | --- |
| **Prueba de Aceptación** | |
| **Identificador:** PA28 | |
| **Nombre de la Prueba:** Cambio de estado del ticket | |
| **N° de Historia que pertenece:** HU26 | |
| **Descripción:**  Este permitirá que se pueda cambiar el estado del ticket después de ser abierto. | |
| **Precondiciones:**  Este se llevara a cabo si el ticket se encuentra almacenado en la base de datos y se muestra en la lista de tickets generados. | **Ejecución:**   1. Ingresar al Sistema 2. Menú principal, Mis Tickets. 3. Clic en el Ticket. 4. Selección de la pestaña, Siguiente Estado: 5. Seleccionar estado. 6. Clic en Cambiar. |
| **Post condiciones:**  El sistema mostrara un mensaje de confirmación “Información del ticket modificada” | **Evaluación de Prueba:**  No muestra mensaje de confirmación. |

**Fuente:** María José Samaniego

**Tabla 118-B:** Historia de Usuario 27

|  |  |
| --- | --- |
| **Historia de Usuario** | |
| **Identificador:** HU27 | **Usuario:** Técnicos de Facultad. |
| **Nombre de la Historia:** Búsqueda de tickets | |
| **Prioridad en Negocio:** Alta  (Muy Alta/Alta / Media / Baja) | **Riesgo en Desarrollo:** Normal  (Alta / Normal/Medio/ Bajo) |
| **Puntos Estimados:** 2 | **Iteración Asignada:** 2 |
| **Programador Responsable:** María Jose Samaniego Larrea | |
| **Descripción:**  Como técnico de facultad deseo realizar búsquedas de tickets. | |
| **Observaciones:** | |

**Fuente:** María José Samaniego

**Tabla 119-B:** Tarea de Ingeniería 56

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU27 |
| **Identificador:** TI56 |
| **Nombre de la Tarea:** Configuración del controlador del modelo Ticket. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**  Se creara el método de buscar un ticket en el controlador del ticket. |

**Fuente:** María José Samaniego

**Tabla 119-B:** Tarea de Ingeniería 57

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU27 |
| **Identificador:** TI57 |
| **Nombre de la Tarea:** Modificación de la vista de visualización del Ticket. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**   * Se creará el campo para el ingreso de la palabra clave para la búsqueda del ticket. |

**Fuente:** María José Samaniego

**Tabla 120-B:** Prueba de Aceptación 29

|  |  |
| --- | --- |
| **Prueba de Aceptación** | |
| **Identificador:** PA29 | |
| **Nombre de la Prueba:** Búsqueda del ticket. | |
| **N° de Historia que pertenece:** HU27 | |
| **Descripción:**  Este permitirá que se pueda cambiar el estado del ticket después de ser abierto. | |
| **Precondiciones:**  Este se llevará a cabo, si el usuario se encuentra ubicado dentro la opción de la lista de tickets generados. | **Ejecución:**   1. Ingresar al Sistema 2. Menú principal, Mis Tickets. 3. Clic en el Ticket. 4. Ingreso de la palabra clave |
| **Post condiciones:**  El sistema mostrara la lista actualizada con los tickets que contengan la palabra clave ingresada. | **Evaluación de Prueba:**  Exitosa |

**Fuente:** María José Samaniego

**Anexo B6: Sprint 3**

**Tabla 121-B:** Sprint Backlog Iteración 3

|  |  |  |  |
| --- | --- | --- | --- |
| **SPRINT BACKLOG ITERACIÓN 3 PROYECTO: SDSI-DTIC** | | | |
| **ID** | **Tipo** | **Descripción** | **Estado** |
|  | **Tarea** | **Depuración del producto entregado en el Sprint 3.** | **Completo** |
| **HU28** | **Historia de Usuario** | **Como técnico deseo generar un reporte con la lista de todos tickets creados.** | **Completo** |
|  | Tarea | Configuración del controlador | Completo |
|  | Tarea | Creación de la vista para los reportes. | Completo |
|  | Tarea | Pruebas unitarias. | Completo |
| **HU29** | **Historia de Usuario** | **Como técnico deseo generar un reporte con la lista de todos tickets nuevos.** | **Completo** |
|  | Tarea | Configuración del controlador del ticket. | Completo |
|  | Tarea | Creación de la vista para el reporte. | Completo |
|  | Tarea | Validación de formularios del modelo. | Completo |
|  | Tarea | Prueba Unitaria | Completo |
| **HU30** | **Historia de Usuario** | **Como técnico deseo generar un reporte con la lista de todos tickets abiertos.** | **Completo** |
|  | Tarea | Configuración del controlador del ticket. | Completo |
|  | Tarea | Creación de la vista para el reporte. | Completo |
|  | Tarea | Validación de formularios del modelo. | Completo |
|  | Tarea | Prueba Unitaria | Completo |
| **HU31** | **Historia de Usuario** | **Como técnico deseo generar un reporte con la lista de todos tickets cerrados.** | **Completo** |
|  | Tarea | Configuración del controlador del ticket. | Completo |
|  | Tarea | Creación de la vista para el reporte. | Completo |
|  | Tarea | Validación de formularios del modelo. | Completo |
|  | Tarea | Prueba Unitaria | Completo |
| **HU32** | **Historia de Usuario** | **Como director DTIC deseo generar un reporte con la lista de tickets por área DTIC.** | **Completo** |
|  | Tarea | Configuración del controlador del ticket. | Completo |
|  | Tarea | Creación de la vista para el reporte. | Completo |
|  | Tarea | Validación de formularios del modelo. | Completo |
|  | Tarea | Prueba Unitaria | Completo |
| **HU33** | **Historia de Usuario** | **Como jefe de área deseo generar un reporte con la lista de tickets del área.** | **Completo** |
|  | Tarea | Configuración del controlador del ticket. | Completo |
|  | Tarea | Creación de la vista para el reporte. | Completo |
|  | Tarea | Prueba Unitaria | Completo |
| **HU34** | **Historia de Usuario** | **Como director DTIC deseo generar un reporte con la lista de tickets por Dependencia o Facultad ESPOCH** | **Completo** |
|  | Tarea | Configuración del controlador del ticket. | Completo |
|  | Tarea | Creación de la vista para el reporte. | Completo |
|  | Tarea | Prueba Unitaria | Completo |
| **HU35** | **Historia de Usuario** | **Como técnico, deseo generar un reporte del ticket.** | **Completo** |
|  | Tarea | Configuración del controlador del ticket. | Completo |
|  | Tarea | Creación de la vista para el reporte. | Completo |
|  | Tarea | Prueba Unitaria | Completo |
| **HU36** | **Historia de Usuario** | **Como técnico, deseo generar un reporte del ticket.** | **Completo** |
|  | Tarea | Configuración del controlador del ticket. | Completo |
|  | Tarea | Creación de la vista para el reporte. | Completo |
|  | Tarea | Prueba Unitaria | Completo |
| **HT6** | **Historia Técnica** | **Elaboración del manual de usuario** | **Completo** |
| **HT7** | **Historia Técnica** | **Elaboración del manual de instalación** | **Completo** |
| **HT8** | **Historia Técnica** | **Implantación del Sistema** | **Completo** |

**Fuente:** María José Samaniego

**Tabla 123-B:** Historia de Usuario 28

|  |  |
| --- | --- |
| **Historia de Usuario** | |
| **Identificador:** HU28 | **Usuario:** Técnicos de Facultad. |
| **Nombre de la Historia:** Reporte de Todos los Tickets generados. | |
| **Prioridad en Negocio:** Media  (Muy Alta/Alta / Media / Baja) | **Riesgo en Desarrollo:** Normal  (Alta / Normal/Medio/ Bajo) |
| **Puntos Estimados:** 3 | **Iteración Asignada:** 3 |
| **Programador Responsable:** María Jose Samaniego Larrea | |
| **Descripción:**  Como técnico de facultad deseo generar un reporte con la lista de todos tickets creados. | |
| **Observaciones:** | |

**Fuente:** María José Samaniego

**Tabla 124-B:** Tarea de Ingeniería 58

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU28 |
| **Identificador:** TI58 |
| **Nombre de la Tarea:** Configuración del controlador del modelo Ticket. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**  Se creara el método para generar el reporte con la lista de todos los tickets. |

**Fuente:** María José Samaniego

**Tabla 125-B:** Tarea de Ingeniería 59

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU28 |
| **Identificador:** TI59 |
| **Nombre de la Tarea:** Modificación de la vista de reportes. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**   * Se creará la vista para la generación de los distintos reportes sobre un ticket. * Se creara la vista para el reporte con la lista de todos los tickets. |

**Fuente:** María José Samaniego

**Tabla 126-B:** Prueba de Aceptación 30

|  |  |
| --- | --- |
| **Prueba de Aceptación** | |
| **Identificador:** PA30 | |
| **Nombre de la Prueba:** Reporte de todos los tickets. | |
| **N° de Historia que pertenece:** HU28 | |
| **Descripción:**  Este permitirá generar reporte con la lista de todos los tickets. | |
| **Precondiciones:**  Este se llevará a cabo si en la base de datos se encuentran registrados los tickets. | **Ejecución:**   1. Ingresar al Sistema 2. Menú principal, Reportes. 3. Todos los Tickets. 4. Generar PDF |
| **Post condiciones:**  El sistema mostrara una nueva ventana con el reporte en formato PDF. | **Evaluación de Prueba:**  Exitosa |

**Fuente:** María José Samaniego

**Tabla 127-B:** Historia de usuario 29

|  |  |
| --- | --- |
| **Historia de Usuario** | |
| **Identificador:** HU29 | **Usuario:** Técnicos de Facultad |
| **Nombre de la Historia:** Reporte de Todos los tickets nuevos. | |
| **Prioridad en Negocio:** Media  (Muy Alta/Alta / Media / Baja) | **Riesgo en Desarrollo:** Normal  (Alta / Normal/Medio/ Bajo) |
| **Puntos Estimados:** 3 | **Iteración Asignada:** 3 |
| **Programador Responsable:** María Jose Samaniego Larrea | |
| **Descripción:**  Como técnico de facultad deseo generar un reporte con la lista de todos tickets nuevos. | |
| **Observaciones:** | |

**Fuente:** María José Samaniego

**Tabla 128-B:** Tarea de Ingeniería 60

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU29 |
| **Identificador:** TI60 |
| **Nombre de la Tarea:** Configuración del controlador del modelo Ticket. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**  Se creará el método para generar el reporte con la lista de todos los tickets nuevos. |

**Fuente:** María José Samaniego

**Tabla 129-B:** Tarea de Ingeniería 61

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU29 |
| **Identificador:** TI61 |
| **Nombre de la Tarea:** Modificación de la vista de reportes. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**   * Se creara la vista para el reporte con la lista de todos los tickets nuevos. |

**Fuente:** María José Samaniego

**Tabla 130-B:** Prueba de Aceptación 31

|  |  |
| --- | --- |
| **Prueba de Aceptación** | |
| **Identificador:** PA31 | |
| **Nombre de la Prueba:** Reporte de todos los tickets nuevos. | |
| **N° de Historia que pertenece:** HU29 | |
| **Descripción:**  Este permitirá generar reporte con la lista de todos los tickets nuevos. | |
| **Precondiciones:**  Este se llevará a cabo si en la base de datos se encuentran registrados los tickets. | **Ejecución:**   1. Ingresar al Sistema 2. Menú principal, Reportes. 3. Tickets Nuevos. 4. Generar PDF |
| **Post condiciones:**  El sistema mostrara una nueva ventana con el reporte en formato PDF. | **Evaluación de Prueba:**  Exitosa |

**Fuente:** María José Samaniego

**Tabla 131-B:** Historia de Usuario 30

|  |  |
| --- | --- |
| **Historia de Usuario** | |
| **Identificador:** HU30 | **Usuario:** Técnicos de Facultad. |
| **Nombre de la Historia:** Reporte de los Tickets Abiertos. | |
| **Prioridad en Negocio:** Media  (Muy Alta/Alta / Media / Baja) | **Riesgo en Desarrollo:** Normal  (Alta / Normal/Medio/ Bajo) |
| **Puntos Estimados:** 3 | **Iteración Asignada:** 3 |
| **Programador Responsable:** María Jose Samaniego Larrea | |
| **Descripción:**  Como técnico de facultad deseo generar un reporte con la lista de todos tickets abiertos. | |
| **Observaciones:** | |

**Fuente:** María José Samaniego

**Tabla 132-B:** Tarea de Ingeniería 62

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU30 |
| **Identificador:** TI62 |
| **Nombre de la Tarea:** Configuración del controlador del modelo Ticket. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**  Se creará el método para generar el reporte con la lista de todos los tickets abiertos. |

**Fuente:** María José Samaniego

**Tabla 133-B:** Tarea de Ingeniería 63

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU30 |
| **Identificador:** TI63 |
| **Nombre de la Tarea:** Modificación de la vista de reportes. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**   * Se creara la vista para el reporte con la lista de todos los tickets abiertos. |

**Fuente:** María José Samaniego

**Tabla 134-B:** Prueba de Aceptación 32

|  |  |
| --- | --- |
| **Prueba de Aceptación** | |
| **Identificador:** PA32 | |
| **Nombre de la Prueba:** Reporte de todos los tickets nuevos. | |
| **N° de Historia que pertenece:** HU30 | |
| **Descripción:**  Este permitirá generar reporte con la lista de todos los tickets abiertos. | |
| **Precondiciones:**  Este se llevará a cabo si en la base de datos se encuentran registrados los tickets. | **Ejecución:**   1. Ingresar al Sistema 2. Menú principal, Reportes. 3. Tickets Abiertos. 4. Generar PDF |
| **Post condiciones:**  El sistema mostrara una nueva ventana con el reporte en formato PDF. | **Evaluación de Prueba:**  Exitosa |

**Fuente:** María José Samaniego

**Tabla 135-B:** Historia de Usuario 31

|  |  |
| --- | --- |
| **Historia de Usuario** | |
| **Identificador:** HU31 | **Usuario:** Técnicos de Facultad. |
| **Nombre de la Historia:** Reporte de los Tickets Cerrados | |
| **Prioridad en Negocio:** Media  (Muy Alta/Alta / Media / Baja) | **Riesgo en Desarrollo:** Normal  (Alta / Normal/Medio/ Bajo) |
| **Puntos Estimados:** 3 | **Iteración Asignada:** 3 |
| **Programador Responsable:** María Jose Samaniego Larrea | |
| **Descripción:**  Como técnico de facultad deseo generar un reporte con la lista de todos tickets cerrados. | |
| **Observaciones:** | |

**Fuente:** María José Samaniego

**Tabla 136-B:** Tarea de Ingeniería 64

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU31 |
| **Identificador:** TI64 |
| **Nombre de la Tarea:** Configuración del controlador del modelo Ticket. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**  Se creará el método para generar el reporte con la lista de todos los tickets cerrados. |

**Fuente:** María José Samaniego

**Tabla 137-B:** Tarea de Ingeniería 65

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU31 |
| **Identificador:** TI65 |
| **Nombre de la Tarea:** Modificación de la vista de reportes. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**   * Se creara la vista para el reporte con la lista de todos los tickets cerrados. |

**Fuente:** María José Samaniego

**Tabla 138-B:** Prueba de Aceptación 33

|  |  |
| --- | --- |
| **Prueba de Aceptación** | |
| **Identificador:** PA33 | |
| **Nombre de la Prueba:** Reporte de todos los tickets nuevos. | |
| **N° de Historia que pertenece:** HU31 | |
| **Descripción:**  Este permitirá generar reporte con la lista de todos los tickets abiertos. | |
| **Precondiciones:**  Este se llevará a cabo si en la base de datos se encuentran registrados los tickets. | **Ejecución:**   1. Ingresar al Sistema 2. Menú principal, Reportes. 3. Tickets Cerrados. 4. Generar PDF |
| **Post condiciones:**  El sistema mostrara una nueva ventana con el reporte en formato PDF. | **Evaluación de Prueba:**  Exitosa |

**Fuente:** María José Samaniego

**Tabla 139-B:** Historia de Usuario 32

|  |  |
| --- | --- |
| **Historia de Usuario** | |
| **Identificador:** HU32 | **Usuario:** Director DTIC. |
| **Nombre de la Historia:** Reporte de los Tickets por Área | |
| **Prioridad en Negocio:** Media  (Muy Alta/Alta / Media / Baja) | **Riesgo en Desarrollo:** Normal  (Alta / Normal/Medio/ Bajo) |
| **Puntos Estimados:** 3 | **Iteración Asignada:** 3 |
| **Programador Responsable:** María Jose Samaniego Larrea | |
| **Descripción:**  Como director DTIC deseo generar un reporte con la lista de tickets por área DTIC. | |
| **Observaciones:** | |

**Fuente:** María José Samaniego

**Tabla 140-B:** Tarea de Ingeniería 66

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU32 |
| **Identificador:** TI66 |
| **Nombre de la Tarea:** Configuración del controlador del modelo Ticket. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**  Se creará el método para generar el reporte con la lista de todos los tickets por área DTIC. |

**Fuente:** María José Samaniego

**Tabla 141-B:** Tarea de Ingeniería 67

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU32 |
| **Identificador:** TI67 |
| **Nombre de la Tarea:** Modificación de la vista de reportes. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**   * Se creará la vista modal para el reporte con la lista de todos los tickets por área DTIC. * Se creará la lista de todas las áreas DTIC por medio de una tabla. |

**Fuente:** María José Samaniego

**Tabla 142-B:** Prueba de Administración 34

|  |  |
| --- | --- |
| **Prueba de Aceptación** | |
| **Identificador:** PA34 | |
| **Nombre de la Prueba:** Reporte de todos los tickets por área DTIC. | |
| **N° de Historia que pertenece:** HU32 | |
| **Descripción:**  Este permitirá generar reporte con la lista de todos los tickets abiertos. | |
| **Precondiciones:**  Este se llevará a cabo si en la base de datos se encuentran registrados los tickets. | **Ejecución:**   1. Ingresar al Sistema 2. Menú principal, Reportes. 3. Tickets por área DTIC. 4. Clic en seleccionar 5. Seleccionar área. 6. Generar PDF |
| **Post condiciones:**  El sistema mostrara una nueva ventana con el reporte en formato PDF. | **Evaluación de Prueba:**  Exitosa |

**Fuente:** María José Samaniego

**Tabla 143-B:** Historia de Usuario 34

|  |  |
| --- | --- |
| **Historia de Usuario** | |
| **Identificador:** HU34 | **Usuario:** Director DTIC |
| **Nombre de la Historia:** Reporte de los Tickets por facultad o dependencia. | |
| **Prioridad en Negocio:** Media  (Muy Alta/Alta / Media / Baja) | **Riesgo en Desarrollo:** Normal  (Alta / Normal/Medio/ Bajo) |
| **Puntos Estimados:** 3 | **Iteración Asignada:** 3 |
| **Programador Responsable:** María Jose Samaniego Larrea | |
| **Descripción:**  Como director DTIC deseo generar un reporte con la lista de tickets por Dependencia o Facultad ESPOCH. | |
| **Observaciones:** | |

**Fuente:** María José Samaniego

**Tabla 144-B:** Tarea de Ingeniería 68

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU34 |
| **Identificador:** TI68 |
| **Nombre de la Tarea:** Configuración del controlador del modelo Ticket. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**  Se creará el método para generar el reporte con la lista de todos los tickets por facultad o dependencia. |

**Fuente:** María José Samaniego

**Tabla 145-B:** Tarea de Ingeniería 69

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU34 |
| **Identificador:** TI69 |
| **Nombre de la Tarea:** Modificación de la vista de reportes. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**   * Se creará la vista modal para el reporte con la lista de todos los tickets por facultad o dependencia * Se creará la lista de todas las facultades o dependencias por medio de una tabla. |

**Fuente:** María José Samaniego

**Tabla 146-B:** Prueba de Aceptación 35

|  |  |
| --- | --- |
| **Prueba de Aceptación** | |
| **Identificador:** PA35 | |
| **Nombre de la Prueba:** Reporte de todos los tickets por área DTIC. | |
| **N° de Historia que pertenece:** HU34 | |
| **Descripción:**  Este permitirá generar reporte con la lista de todos los tickets abiertos. | |
| **Precondiciones:**  Este se llevará a cabo si en la base de datos se encuentran registrados los tickets. | **Ejecución:**   1. Ingresar al Sistema 2. Menú principal, Reportes. 3. Tickets por área DTIC. 4. Clic en seleccionar 5. Seleccionar área. 6. Generar PDF |
| **Post condiciones:**  El sistema mostrara una nueva ventana con el reporte en formato PDF. | **Evaluación de Prueba:**  Exitosa |

**Fuente:** María José Samaniego

**Tabla 147-B:** Historia de Usuario 35

|  |  |
| --- | --- |
| **Historia de Usuario** | |
|  | |
| **Identificador:** HU35 | **Usuario:** Técnicos de Facultad. |
| **Nombre de la Historia:** Generar el reporte de un ticket. | |
| **Prioridad en Negocio:** Media  (Muy Alta/Alta / Media / Baja) | **Riesgo en Desarrollo:** Normal  (Alta / Normal/Medio/ Bajo) |
| **Puntos Estimados:** 3 | **Iteración Asignada:** 3 |
| **Programador Responsable:** María Jose Samaniego Larrea | |
| **Descripción:**  Como director, jefe de área, o técnico, deseo generar un reporte del ticket. | |
| **Observaciones:** | |

**Fuente:** María José Samaniego

**Tabla 148-B:** Tarea de Ingeniería 70

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU35 |
| **Identificador:** TI70 |
| **Nombre de la Tarea:** Configuración del controlador del modelo Ticket. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**  Se creará el método para generar el reporte del ticket. |

**Fuente:** María José Samaniego

**Tabla 149-B:** Tarea de Ingeniería 71

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU35 |
| **Identificador:** TI71 |
| **Nombre de la Tarea:** Modificación de la vista de reportes del ticket. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**   * Se creará la vista para el reporte del ticket. |

**Fuente:** María José Samaniego

**Tabla 150-B:** Prueba de Aceptación 37

|  |  |
| --- | --- |
| **Prueba de Aceptación** | |
| **Identificador:** PA36 | |
| **Nombre de la Prueba:** Reporte de todos los tickets por área DTIC. | |
| **N° de Historia que pertenece:** HU35 | |
| **Descripción:**  Este permitirá generar reporte con la información del ticket. | |
| **Precondiciones:**  Este se llevará a cabo si en la base de datos se encuentran registrados los tickets. | **Ejecución:**   1. Ingresar al Sistema 2. Menú principal, Mis Tickets. 3. Seleccionar ticket 4. Generar PDF |
| **Post condiciones:**  El sistema mostrara una nueva ventana con el reporte en formato PDF. | **Evaluación de Prueba:**  Exitosa |

**Fuente:** María José Samaniego

**Tabla 151-B:** Historia de Usuario 36

|  |  |
| --- | --- |
| **Historia de Usuario** | |
| **Identificador:** HU36 | **Usuario:** Jefe de Área. |
| **Nombre de la Historia:** Generar el reporte de un ticket por técnico. | |
| **Prioridad en Negocio:** Media  (Muy Alta/Alta / Media / Baja) | **Riesgo en Desarrollo:** Normal  (Alta / Normal/Medio/ Bajo) |
| **Puntos Estimados:** 3 | **Iteración Asignada:** 3 |
| **Programador Responsable:** María Jose Samaniego Larrea | |
| **Descripción:**  Como jefe de área de DTIC, deseo generar un reporte del ticket por técnico que ha sido asignado o reportado. | |
| **Observaciones:** | |

**Fuente:** María José Samaniego

**Tabla 152-B:** Tarea de Ingeniería 72

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU36 |
| **Identificador:** TI72 |
| **Nombre de la Tarea:** Configuración del controlador del modelo Ticket. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**  Se creará el método para generar el reporte del ticket por técnico. |

**Fuente:** María José Samaniego

**Tabla 153-B:** Tarea de Ingeniería 73

|  |
| --- |
| **Tarea de Ingeniería** |
| **Historia de Usuario que pertenece:** HU36 |
| **Identificador:** TI73 |
| **Nombre de la Tarea:** Modificación de la vista de reportes del ticket. |
| **Tipo de Tarea:** Desarrollo |
| **Programador Responsable:** María José Samaniego Larrea |
| **Descripción:**   * Se creará la vista para el reporte del ticket por técnico. |

**Fuente:** María José Samaniego

**Tabla 154-B:** Prueba de Aceptación 37

|  |  |
| --- | --- |
| **Prueba de Aceptación** | |
| **Identificador:** PA37 | |
| **Nombre de la Prueba:** Reporte de todos los tickets por técnico. | |
| **N° de Historia que pertenece:** HU35 | |
| **Descripción:**  Este permitirá generar reporte con la lista de todos los tickets por técnico. | |
| **Precondiciones:**  Este se llevará a cabo si en la base de datos se encuentran registrados los tickets. | **Ejecución:**   1. Ingresar al Sistema 2. Menú principal, Reportes. 3. Tickets por área DTIC. 4. Clic en seleccionar 5. Seleccionar área. 6. Generar PDF |
| **Post condiciones:**  El sistema mostrara una nueva ventana con el reporte en formato PDF. | **Evaluación de Prueba:**  Exitosa |

**Fuente:** María José Samaniego

**Tabla 155-B:** Historia Técnica 5

|  |  |
| --- | --- |
| **Historia Técnica** | |
| **Identificador:** HT5 | |
| **Nombre de la Historia:** Elaboración del manual de usuario del SDSI-DTIC | |
| **Prioridad en Negocio:** Media  (Muy Alta/Alta / Media / Baja) | **Riesgo en Desarrollo:** Normal  (Alta / Normal/Medio/ Bajo) |
| **Puntos Estimados:** 3 | **Iteración Asignada:**3 |
| **Programador Responsable:** María Jose Samaniego Larrea | |
| **Descripción:**  Como equipo de desarrollo se necesita elaborar el documento de manual de usuario del sistema. | |
| **Observaciones:** | |

**Fuente:** María José Samaniego

**Tabla 156-B:** Historia Técnica 6

|  |  |
| --- | --- |
| **Historia Técnica** | |
| **Identificador:** HT6 | |
| **Nombre de la Historia:** Elaboración del Manual de Instalación del SDSI-DTIC | |
| **Prioridad en Negocio:** Media  (Muy Alta/Alta / Media / Baja) | **Riesgo en Desarrollo:** Normal  (Alta / Normal/Medio/ Bajo) |
| **Puntos Estimados:** 3 | **Iteración Asignada:**3 |
| **Programador Responsable:** María Jose Samaniego Larrea | |
| **Descripción:**  Como equipo de desarrollo se necesita elaborar el documento de manual de instalación del sistema. | |
| **Observaciones:** | |

**Fuente:** María José Samaniego

**Tabla 156-B:** Historia Técnica 7

|  |  |
| --- | --- |
| **Historia Técnica** | |
| **Identificador:** HT7 | |
| **Nombre de la Historia:** Implantación del SDSI-DTIC | |
| **Prioridad en Negocio:** Media  (Muy Alta/Alta / Media / Baja) | **Riesgo en Desarrollo:** Normal  (Alta / Normal/Medio/ Bajo) |
| **Puntos Estimados:** 3 | **Iteración Asignada:**3 |
| **Programador Responsable:** María Jose Samaniego Larrea | |
| **Descripción:**  Como equipo de desarrollo se necesita realizar el despliegue e implantación del Sistema de Seguimiento de Incidentes en DTIC. | |
| **Observaciones:** | |

**Fuente:** María José Samaniego

**Tabla 157-B:** Historia Técnica 8

|  |  |
| --- | --- |
| **Historia Técnica** | |
| **Identificador:** HT8 | |
| **Nombre de la Historia:** Elaboración del manual de usuario | |
| **Prioridad en Negocio:** Normal  (Muy Alta/Alta / Media / Baja) | **Riesgo en Desarrollo:** Normal  (Alta / Normal/Medio/ Bajo) |
| **Puntos Estimados:** 3 | **Iteración Asignada:** 0 |
| **Programador Responsable:** María Jose Samaniego Larrea | |
| **Descripción:**  Como equipo de desarrollo se necesita elaborar el manual de usuario del sistema. | |
| **Observaciones:** | |

**Fuente:** María José Samaniego

**Tabla 157-B:** Historia Técnica 9

|  |  |
| --- | --- |
| **Historia Técnica** | |
| **Identificador:** HT7 | |
| **Nombre de la Historia:** Elaboración del manual de instalación | |
| **Prioridad en Negocio:** Muy Alta  (Muy Alta/Alta / Media / Baja) | **Riesgo en Desarrollo:** Normal  (Alta / Normal/Medio/ Bajo) |
| **Puntos Estimados:**3 | **Iteración Asignada:** 0 |
| **Programador Responsable:** María Jose Samaniego Larrea | |
| **Descripción:**  Como equipo de desarrollo se necesita elaborar el manual de instalación del sistema. | |
| **Observaciones:** | |

**Fuente:** María José Samaniego