ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

FACULTAD DE INFORMÁTICA Y ELECTRÓNICA

CARRERA SOFTWARE

DESARROLLO DE UN SITIO WEB PARA LA GENERACIÓN AUTOMATIZADA DE CASOS DE PRUEBAS UNITARIAS UTILIZANDO LA API DE OPEN AI

Trabajo de Integración Curricular

Tipo: Proyecto Técnico

Presentado para optar al grado académico de:

INGENIERO DE SOFTWARE

AUTOR:

JOHN VLADIMIR CUVI GUAMAN

Riobamba – Ecuador

2024

ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

FACULTAD DE INFORMÁTICA Y ELECTRÓNICA

CARRERA SOFTWARE

DESARROLLO DE UN SITIO WEB PARA LA GENERACIÓN AUTOMATIZADA DE CASOS DE PRUEBAS UNITARIAS UTILIZANDO LA API DE OPEN AI

Trabajo de Integración Curricular

Tipo: Proyecto Técnico

Presentado para optar al grado académico de:

INGENIERO DE SOFTWARE

AUTOR: JOHN VLADIMIR CUVI GUAMAN

DIRECTOR: ING. MIGUEL ÁNGEL DUQUE VACA

Riobamba – Ecuador

2024

Yo, John Vladimir Cuvi Guaman, declaro que el presente Trabajo de Integración Curricular es de mi autoría y los resultados del mismo son auténticos. Los textos en el documento que provienen de otras fuentes están debidamente citados y referenciados.

Como autor asumo la responsabilidad legal y académica de los contenidos de este Trabajo de Integración Curricular; el patrimonio intelectual pertenece a la Escuela Superior Politécnica de Chimborazo.

Riobamba, 30 de abril de 2024

John Vladimir Cuvi Guaman
0606253169

ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

FACULTAD DE INFORMÁTICA Y ELECTRÓNICA

CARRERA SOFTWARE

El Tribunal del Trabajo de Integración Curricular certifica que: El Trabajo de Integración Curricular; Tipo: Proyecto Técnico**, DESARROLLO DE UN SITIO WEB PARA LA GENERACIÓN AUTOMATIZADA DE CASOS DE PRUEBAS UNITARIAS UTILIZANDO LA API DE OPEN AI,** realizado por el señor: **JOHN VLADIMIR CUVI GUAMAN**, ha sido minuciosamente revisado por los Miembros del Tribunal del Trabajo de Integración Curricular, el mismo que cumple con los requisitos científicos, técnicos, legales, en tal virtud el Tribunal autoriza su presentación.

	FIRMA	FECHA
Ing. Gisel Katerine Bastidas Guacho **PRESIDENTA DEL TRIBUNAL**		2024-04-30
Ing. Miguel Ángel Duque Vaca **DIRECTOR DEL TRABAJO DE INTEGRACIÓN CURRICULAR**		2024-04-30
Dra. Gloria de Lourdes Arcos Medina **ASESORA DEL TRABAJO DE INTEGRACIÓN CURRICULAR**		2024-04-30

DEDICATORIA

A mis padres por el apoyo y motivación durante todo el proceso para finalizar este proyecto, a mis maestros quienes han contribuido en mi formación académica, pero sobre todo a Dios quien me ha guiado, cuidado y bendecido en cada paso durante todo este proceso.

John.

AGRADECIMIENTO

A Dios, mis padres, hermanos, Nataly y a la Escuela Superior Politécnica de Chimborazo, por brindarme una educación de calidad junto a diversas experiencias únicas. A mi director Ing. Miguel Duque y asesora Dra. Gloria Arcos por la orientación y apoyo brindado para el presente proyecto.

John.

ÍNDICE DE CONTENIDO

CAPÍTULO III

ÍNDICE DE TABLAS

ÍNDICE DE ILUSTRACIONES

ÍNDICE DE ANEXOS

RESUMEN

Los desarrolladores y testers enfrentan diversos desafíos al realizar pruebas unitarias de forma manual. Esto incluye la considerable inversión de tiempo necesaria para escribir y ejecutar pruebas, lo que puede comprometer los plazos de entrega del proyecto afectando la cobertura de las mismas sumado a la falta de experiencia en la creación de numerosas pruebas unitarias es otro obstáculo, la necesidad de ajustes manuales en las pruebas debido al cambio de requerimientos también contribuyen a la complejidad y el consumo de tiempo excesivo en este proceso. Por tanto, el presente trabajo de integración curricular tiene como objetivo desarrollar un sitio web para la generación automatizada de casos de pruebas unitarias utilizando la API de Open AI. El tipo de estudio que se empleó para el desarrollo fue la investigación aplicada y cuantitativa empleando métodos inductivos, analíticos, observación junto a la metodología Scrumban. Mediante la aplicación de la metodología Scrumban permitió gestionar el desarrollo del proyecto siguiendo un enfoque ágil utilizando tecnologías como JavaScript con su framework React, Node.js con Express y una base de datos en FireStore, siguiendo una arquitectura basada en componentes y cliente – servidor, estas soluciones desarrolladas se desplegaron en Netlify para el frontend y Render para el backend. Se realizó un proceso para la evaluación de la solución desarrollada a los estudiantes de octavo semestre de la Escuela Superior Politécnica de Chimborazo del periodo octubre 2023 - marzo 2024. Para evaluar la calidad de del software se empleó el estándar ISO/IEC 25010:2011 enfocado a medir el comportamiento temporal. Los resultados muestran que los tiempos de desarrollo de pruebas unitarias usando de la solución desarrollada se reducen significativamente en un 783.13% en comparación con el desarrollo de pruebas manuales.

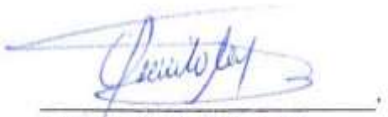
Palabras clave: <INGENIERIA DE SOFTWARE> <INTELIGENCIA ARTIFICIAL> <CHAT GPT> <SITIO WEB> <OPEN AI> <AUTOMATIZACIÓN>.

0503-DBRA-UPT-2024

SUMMARY

Developers and testers face significant challenges when performing unit tests manually. This includes the considerable time investment required to write and execute tests, which can complicate project delivery dates affecting coverage. Lack of experience in creating numerous unit tests is another obstacle; the need for manual adjustments to tests due to changing requirements also contributes to complexity and excessive time consumption in this process; therefore, the purpose of this integration curricular work was to develop a web site for the automated generation of unit test cases using the Open AI API. The type of study used for development was applied, and quantitative research was conducted using inductive, analytical, and observation methods along with the Scrumban methodology. Applying the Scrumban methodology allowed managing the development of the project following an agile approach using technologies such as JavaScript with its React framework, Node.js with Express, and a Firestore database, following a component-based and client-server architecture; these developed solutions were deployed to Netlify for the frontend and Render for the backend. Finally, a process was carried out to evaluate the solution designed for students in the eighth semester of the Escuela Superior Politécnica de Chimborazo from October 2023 to March 2024. To assess the quality of the software, the ISO/IEC 25010 standard was used, focusing on measuring temporal behavior. The results show that unit testing development times using the developed solution are significantly reduced by 783.13% compared to manual tests.

Keywords: <SOFTWARE ENGINEERING>, <ARTIFICIAL INTELLIGENCE>, <CHAT GPT>, <UNIT TESTING>, <TESTING>, <OPEN AI>, <AUTOMATION>, <TESTERS>.



Prof. Nelly Padilla. Mgs.
0603818717
DOCENTE FIE

INTRODUCCIÓN

Actualmente existe un gran auge en el desarrollo e implementación de sistemas basados en inteligencia artificial. Gracias a los avances en aprendizaje automático y procesamiento de lenguaje natural, la IA se integra en industrias y actividades cotidianas con el objetivo de automatizar y agilizar procesos. El desarrollo de software ha sido uno de los campos beneficiados.

Las actividades de pruebas, en particular las pruebas unitarias, son una parte fundamental del ciclo de desarrollo de software porque permiten la detección temprana de errores. Sin embargo, escribir y ejecutar manualmente estas pruebas requiere de mucho tiempo y esfuerzo por parte de los desarrolladores.

Es aquí donde la IA podría hacer una importante contribución al aplicar técnicas de procesamiento de lenguaje natural, que automatice este proceso importante permitiendo además generar una mayor cobertura de pruebas reduciendo tiempo y esfuerzo aprovechando el potencial de hoy en día ofrecen la Inteligencia Artificial.

Con base en lo expuesto, el presente trabajo busca desarrollar un sitio web que utilice la API de Open AI para automatizar la generación de pruebas unitarias al proporcionar la funcionalidad a probar.

La organización de este trabajo se desglosa en cinco capítulos mostradas a continuación:

El CAPÍTULO I aborda los antecedentes del estudio, se presenta la problemática junto con su justificación, y se describen de manera detallada los objetivos generales y específicos del trabajo actual.

En el CAPÍTULO II se lleva a cabo una revisión de la literatura relacionada con el tema, se explora la conceptualización teórica de los conceptos de cada una de las herramientas utilizadas, la metodología de desarrollo, estándares, normas y trabajos relacionados.

En el CAPÍTULO III se expone el tipo de estudio a emplear mismas que son conformados por la investigación aplicada y cuantitativa, así como también los métodos, técnicas e instrumentos empleadas para el desarrollo del sitio web aplicando la metodología Scrumban con el fin de dar cumplimiento a los objetivos.

En el CAPÍTULO IV se describen, evalúan y analizan los resultados obtenidos mismas que consisten en evaluar el comportamiento temporal con base en la norma ISO/IEC 25010.

En el CAPITULO V se detallan las conclusiones y recomendaciones del trabajo de integración curricular.

CAPÍTULO I

1. DIAGNÓSTICO DEL PROBLEMA

Este capítulo presenta los antecedentes teóricos y prácticos que justifican el presente trabajo de integración curricular. Además, menciona el objetivo general y los objetivos específicos. De este modo, el capítulo enmarca el problema, fundamenta su relevancia y delinea el propósito general y los particulares del estudio.

1.1. Antecedentes

Open AI es una organización dedicada a la investigación y el desarrollo en el campo de la inteligencia artificial y el aprendizaje automático. Una de sus mayores contribuciones ha sido la creación de una interfaz de programación de aplicaciones que permite a los desarrolladores acceder a las capacidades y funciones de Open AI dentro de sus propias aplicaciones y proyectos. Utilizando esta API, los programadores pueden aprovechar el poder y la complejidad de los modelos de Open AI en sus soluciones e incorporar la inteligencia artificial de manera innovadora.

Las pruebas unitarias son un conjunto de pruebas automatizadas que verifican el correcto funcionamiento de unidades individuales de código, generalmente métodos o funciones, aisladas del resto del sistema. El desarrollo de casos de pruebas muchas veces causa retrasos en la entrega de software. Esto es debido a varias razones siendo las más comunes el excesivo tiempo dedicado al desarrollo de casos de pruebas, pruebas con coberturas limitadas, la falta de experiencia y la propensión a errores humanos.

Otros factores que afectan la planificación del desarrollo de software son:

- La dificultad para prever todos los posibles escenarios de uso y errores, lo que puede dejar casos de prueba por cubrir.
- La necesidad de modificar constantemente las pruebas a medida que evoluciona el código, consumiendo recursos adicionales.
- La manualidad del proceso hace que sea propenso a errores y omisiones que solo se detectan tardíamente durante la ejecución.

En consecuencia, la planificación se ve afectada por un aumento de costos en el desarrollo del proyecto, tanto en términos de tiempo como de recursos económicos. Por ello, es importante buscar soluciones que permitan agilizar y automatizar en la medida de lo posible el proceso de pruebas.

Existen varios estudios realizados que muestran como la generación de pruebas automatizadas aportan positivamente en el desarrollo de software. En el estudio realizado por (Chinarro Morales, Ruiz Rivera y Ruiz Lizama, 2017) donde desarrollaron un modelo de pruebas funcionales de software basado en la herramienta SELENIUM consiguen obtener un modelo de pruebas automatizadas con el cual logran un mejor desempeño durante el proceso de las pruebas funcionales en la etapa de desarrollo del aplicativo.

Otro estudio realizado por (Gómez del Mónaco, 2021) en la empresa Despegar.com muestra cómo los tests automatizados aportan calidad al proceso de integración continua del producto de software y del monitoreo de las aplicaciones. El estudio se centró en el desarrollo de una aplicación web para reservar vuelos y hoteles. Se determinó que los tests automatizados ayudaron a mejorar la calidad del producto de software al identificar y corregir errores en una etapa temprana del proceso de desarrollo. Además, los tests automatizados ayudaron a reducir el tiempo necesario para lanzar nuevas versiones del producto de software.

En el informe desarrollado por (Lopezosa, 2023) identifican patrones de respuestas en muy poco tiempo mediante el uso de ATLAS.ti misma que está impulsada con el modelo GPT de Open AI, con la cual analizaron 28 documentos que representaban las entrevistas realizadas a aspirantes.

En el estudio realizado por (Gordon, González y Bultrón, 2023) en la Universidad de Panamá muestra cómo los chatbots contribuyen a nuevas oportunidades de negocio en el país. El estudio analizó la implementación de Open AI API en Panamá permite a los profesionales de la ingeniería del software desarrollar aplicaciones sin requerir una especialización avanzada.

Con base en lo expuesto, es necesario desarrollar un sitio web para mejorar los tiempos de desarrollo de casos de pruebas unitarias automatizadas haciendo uso de la API de OPEN AI. El sitio web será utilizado por los desarrolladores de software y testers en etapas de desarrollo de software o pruebas.

1.2. Formulación del problema

¿Cuál es el porcentaje de reducción de tiempos de desarrollo de pruebas unitarias al utilizar el sitio web que ofrece generación automatizada de pruebas unitarias mediante la API de Open AI?

1.3. Sistematización del problema

- ¿Cuáles son los problemas más comunes que tienen los desarrolladores y testers al momento de realizar pruebas unitarias?

- ¿Cuáles son las características fundamentales, el comportamiento y el funcionamiento de la API de Open AI?
- ¿Cuál es la arquitectura de la aplicación?
- ¿Cuál es el tiempo de respuesta del sitio web respecto a la generación de un caso de prueba?

1.4. Justificación

1.4.1. Justificación Teórica

Open AI y su modelo Chat GPT han ganado mucha importancia por su capacidad para generar texto coherente y relevante con base en grandes cantidades de datos que se genera diariamente. Esta tecnología ha revolucionado la forma en que interactuamos con la inteligencia artificial y ha abierto nuevas oportunidades en diversos campos. La capacidad de analizar, comprender y producir texto de alta calidad ha impulsado avances significativos en el procesamiento del lenguaje natural y ha abierto un nuevo camino hacia aplicaciones más sofisticadas lo que ha impulsado ventajas significativas.

En el estudio realizado por (Surameery & Shakor, 2023) mencionan que la integración con otros sistemas de inteligencia artificial, la personalización y la mejora continua del rendimiento del modelo de lenguaje son algunas de las muchas oportunidades emocionantes para que esta tecnología tenga un impacto positivo en nuestras vidas. Algunas de las ventajas mencionadas son:

- Capacidad para proporcionar asistencia de depuración, predicción de errores y explicación de errores en la programación.
- La capacidad para aceptar información adicional, como salidas esperadas o mensajes de error, lo que puede aumentar su tasa de éxito.
- Puede generar respuestas para exámenes universitarios y obtener una calificación promedio equivalente a un estudiante con una calificación C+.

El artículo concluye destacando el potencial de Chat GPT como parte de un completo kit de herramientas de depuración, y los beneficios de combinar sus puntos fuertes con los de otras herramientas de depuración para identificar y corregir errores de forma más eficaz.

El presente trabajo de integración curricular busca aprovechar todas las ventajas que brinda OPEN AI API, desarrollando un sitio web que permita la automatización de la generación de casos de pruebas. Esta solución tiene como objetivo principal reducir tiempo y esfuerzo en comparación con el desarrollo manual, al generar casos de prueba en poco tiempo con una mayor cobertura. Esto se traduce en una mayor cobertura de pruebas, ya que se evalúan diferentes escenarios y se detectan posibles errores de manera exhaustiva en el software. Además, al agilizar el proceso de

desarrollo de casos de pruebas unitarias, se acelera el ciclo de desarrollo, lo que resulta en una mejora en la calidad del producto final.

El sitio web generará casos de pruebas unitarias para los diferentes lenguajes de programación más usados en la actualidad, misma que se configurará previamente seleccionando el lenguaje de programación, la herramienta de testing e ingresando un contexto sobre la funcionalidad.

1.4.2. Justificación Aplicativa

Con el desarrollo del sitio web que permita la automatización de la generación de casos de pruebas unitarias, ofrecen varias ventajas significativas para los desarrolladores de software y los testers, tales como:

- Ahorro considerable de tiempo, ya que la generación de casos de prueba se realiza de manera automatizada. De esta manera contribuirá a que se realicen entregas de software a tiempo, ayudando a cumplir con el cronograma de actividades planificadas en el proyecto.

Otros beneficios que se obtendrían con el aplicativo serán:

- Disminuir la probabilidad de cometer errores en la redacción de los casos de prueba, lo que resulta en una menor tasa de errores en el software final.
- Generar variedad de casos de prueba, abarcando diferentes escenarios y posibles combinaciones de datos de entrada. Contribuyendo a una mayor cobertura de pruebas, asegurando que se evalúen múltiples condiciones.

Para del sitio web se contempla la aplicación de la arquitectura basada en componentes que permitirá dividir la aplicación en secciones independientes y reutilizables, facilitando su desarrollo y mantenimiento. Mismos componentes que serán desarrolladas con React JS Sus componentes permitirán mostrar de forma dinámica los diferentes casos de prueba generados. También se contempla la aplicación de la arquitectura cliente – servidor usando a Express JS que ayudarán a construir el sitio web muy robusto y con buen rendimiento separando las responsabilidades.

Se utilizará OPEN AI API para aprovechar las capacidades de sus modelos de generación de lenguaje natural. Esto permitirá que el sitio web pueda generar de manera automatizada los casos de prueba unitarias contextualizados a partir del código ingresado por el usuario.

Para el desarrollo del sitio web se contempla desarrollar los siguientes componentes principales:

- Un componente de configuración tanto del lenguaje de programación en el que está desarrollado la función a examinar como el ingreso del contexto sobre lo que hace la función.

- Un componente de entrada para código a examinar. Donde el usuario podrá ingresar su código.
- Un componente que abarcara todos los casos de pruebas unitarias.
- Un componente donde se mostrará el historial de los casos de pruebas unitarias generadas que hayan sido guardados por parte del usuario.

Para realizar este proyecto se ha seleccionado una aplicación web denominado Renta Car, misma que servirá como el ambiente de pruebas.

El presente trabajo de integración corresponde a la línea de investigación de Tecnologías de la Información y Comunicación bajo el programa de Ingeniería de Software de la Escuela Superior Politécnica de Chimborazo. Dentro del Plan Nacional de Desarrollo, pertenece al Eje Institucional, objetivo 14 que señala "Fortalecer las capacidades del Estado con énfasis en la administración de justicia y eficiencia en los procesos de regulación y control, con independencia y autonomía"; con su política 14.3 que menciona "Generar y fortalecer sistemas de producción local que permiten robustecer la producción de estadística oficial, para mejorar la toma de decisiones tanto a nivel territorial como aquella a nivel país".

1.5. Objetivos

1.5.1. Objetivo General

Desarrollar un sitio web para la generación automatizada de casos de prueba unitarias utilizando la API de OPEN AI.

1.5.2. Objetivos Específicos

- Identificar los problemas más comunes al momento de desarrollar casos de pruebas unitarias.
- Estudiar las características, comportamiento y el funcionamiento de la API de OPEN AI.
- Desarrollar los componentes de ingreso de funciones, configuración y los componentes de salida de resultados.
- Evaluar el comportamiento temporal del sistema web basado en las normativas ISO/IEC 25010.

CAPÍTULO II

2. MARCO TEÓRICO

Este capítulo brinda una descripción de los conceptos básicos sobre las herramientas y metodología de desarrollo; estándares, normas y trabajos relacionados. También se describen los problemas más comunes que tienen los desarrolladores para realizar pruebas unitarias, así como las características de OPEN AI API.

2.1. Trabajos relacionados

Existen numerosas investigaciones que se centran en la implementación de herramientas para automatizar diversas tareas en el ámbito tecnológico obteniendo resultados positivos entre las cuales se destaca las siguientes investigaciones:

En el trabajo realizado por (Lopezosa & Codina, 2023) desarrollaron un sistema automatizado para el diagnóstico de COVID-19 y la generación de informes médicos utilizando aprendizaje profundo mediante OPEN AI API. El sistema utiliza escaneos de tomografía computarizada para identificar lesiones y cuantificar la efectividad pulmonar, luego generan informes médicos utilizando modelos propios de la OPEN AI API. También se incluye un chatbot para que los usuarios hagan preguntas sobre la identificación de COVID-19. El enfoque propuesto logra un rendimiento de vanguardia en la generación de informes médicos y tiene el potencial de reducir la carga de trabajo de los profesionales de la salud. El modelo GPT3 utilizado genera informes médicos basados en los resultados de la segmentación y proporciona razonamiento para los hallazgos. Los informes médicos generados por los diferentes modelos de lenguaje demuestran si el paciente tiene COVID-19 y porcentaje de sus pulmones han sido afectados.

En el trabajo realizado por (Chinarro Morales et al, 2017) presenta un modelo para automatizar las pruebas funcionales durante la fase de evaluación de calidad del desarrollo de software. El modelo utiliza la herramienta Selenium para evaluar herramientas de gestión de pruebas automatizadas y automatizar las pruebas funcionales. Proporciona una visión general de los conceptos de pruebas de software, incluyendo la importancia de las pruebas en todo el proceso de desarrollo de software para garantizar la satisfacción del cliente y evitar defectos en el producto final. El modelo incluye actividades como la definición de las pruebas automatizadas, la definición de los procedimientos de prueba, la generación de los scripts, la ejecución de las pruebas automatizadas del ciclo funcional, la investigación y modificación de herramientas, la configuración del entorno, la validación de las pruebas automatizadas y la organización de las pruebas automatizadas.

En el artículo de (Serna López et al., 2020) se presenta un proyecto piloto de automatización de pruebas de software en la Unidad de Servicios Tecnológicos del Centro de Servicios y Gestión Empresarial (CESGE) del SENA. El objetivo del proyecto consistió en la mejora de la calidad del software desarrollado por el CESGE y reducir el tiempo y el costo de las pruebas. Se implementó en dos fases que consistía en identificar procesos de pruebas de software y desarrollar scripts de automatización. El autor menciona que se logró automatizar el 80% de los procesos de prueba de software, lo que permitió reducir el tiempo y el costo de las pruebas mejorando la calidad del software desarrollado por la entidad. El estudio concluye afirmando que la automatización de pruebas de software es una herramienta valiosa para mejorar la calidad del software y reducir el tiempo y el costo de las pruebas.

Según los estudios expuestos, se evidencia que la implementación de herramientas de optimización, como la inteligencia artificial, es crucial para impulsar la automatización en diversos sectores industriales. En particular, la API de OPEN AI se destaca como una herramienta que permite aprovechar modelos avanzados de procesamiento del lenguaje, lo que facilita de manera efectiva la automatización de tareas manuales complejas. Entre estas aplicaciones, la automatización de pruebas de software y la generación de informes/reportes emergen como casos prometedores con múltiples beneficios.

2.2. Interfaz de Programación de Aplicaciones

Una Interfaz de Programación de Aplicaciones también conocida como API, consiste en un conjunto de protocolos, rutinas y herramientas que permiten el desarrollo de aplicaciones de software. Su propósito es establecer la forma en que los distintos componentes de un software interactúan entre sí, ofreciendo a los desarrolladores bloques de construcción predefinidos. Con su utilización, los desarrolladores pueden construir software de manera más eficiente al aprovechar funcionalidades preexistentes y estandarizadas (Gadia et al., 2022).

La **Ilustración 2-1** muestra la interacción entre la API cliente y la API de datos en un entorno real donde el cliente envía solicitudes y comandos a la API de datos, que responde proporcionando información o realizando acciones específicas. Esta representación visual destaca la comunicación esencial entre ambas interfaces en el proceso de automatización y transferencia de datos.

Ilustración 2-1: Funcionamiento de la API

Realizado por: Cuvi J., 2023

A partir de ahora, la Interfaz de Programación de Aplicaciones se llamará API, un acrónimo utilizado en el ámbito tecnológico.

2.3. Pruebas de software

Para (Gamido & Gamido, 2019) una prueba de software es el proceso de examinar el software para identificar posibles fallos. Hay dos formas de realizar pruebas de software: pruebas manuales y pruebas automatizadas. Se considera que las pruebas automatizadas son más efectivas en cuanto a tiempo, costos y usabilidad, y existen una amplia variedad de herramientas de prueba automatizadas disponibles, ya sea de código abierto o comerciales. El propósito de las pruebas de software es verificar si el sistema cumple con los requisitos y especificaciones del usuario. Las pruebas implican examinar el comportamiento de un sistema para descubrir posibles fallos.

Para (Sánchez Peño 2015) en su estudio define una prueba de software como el proceso de evaluación de un producto de software para verificar que cumpla con determinados requisitos funcionales y de calidad. Las pruebas de software se basan en fundamentos clave, como verificar el cumplimiento de las especificaciones y detectar posibles errores antes de la entrega del producto. Su objetivo es validar el funcionamiento del software e identificar posibles fallos para mejorar la confiabilidad y calidad del producto final.

2.3.1. Tipos de pruebas

En el artículo publicado por (Unir, 2022) señala que existen 8 tipos de pruebas de software fundamentales descritas a continuación:

- **Pruebas unitarias:** Comprueban el correcto funcionamiento de unidades/componentes aislados del software, como funciones o clases. Son las primeras en realizarse.
- **Pruebas de integración:** Verifican que los diferentes componentes funcionan correctamente conjuntamente después de integrarlos.

- **Pruebas funcionales:** Validan que el software cumple con los requerimientos funcionales especificados, simulando los flujos de usuario. Incluyen pruebas unitarias y de regresión.
- **Pruebas de aceptación:** Corroboran que el sistema funciona según lo esperado, verificando los criterios de aceptación definidos.
- **Pruebas de rendimiento:** Evalúan la estabilidad, velocidad y respuesta ante diferentes cargas de trabajo.
- **Pruebas de estrés:** Simulan cargas de trabajo muy altas para determinar el límite de saturación del sistema.
- **Pruebas de humo:** Garantizan la funcionalidad principal para determinar la preparación para pruebas más profundas.

2.3.2. Pruebas unitarias

(Rojas-Robert et al., 2019) mencionan que las pruebas unitarias son actividades de vital importancia en el testing de software, ya que permiten diseñar y crear casos de prueba efectivos asociados con una generación adecuada de valores de prueba. Las pruebas empíricas no pueden garantizar la detección de todos los errores, y las pruebas exhaustivas son muy costosas en tiempo y esfuerzo. Por lo tanto, la estrategia es tratar de hacer que las pruebas sean lo más inclusivas posible para considerar un subconjunto de casos posibles y valores de prueba más propensos a detectar errores.

Para (Viteri Arias et al., 2018) las pruebas unitarias son parte importante del proceso de desarrollo de software, ya que permiten detectar errores en una etapa temprana del proceso de desarrollo, lo que reduce el costo y el tiempo necesario para corregir los errores. Además, las pruebas unitarias mejoran la calidad del software y permiten a los desarrolladores realizar cambios en el código con confianza, ya que pueden verificar que las pruebas unitarias siguen pasando después de cada cambio, esto según. Las pruebas unitarias se realizan con el acceso al código que se está probando y con el soporte de herramientas de depuración, involucrando principalmente a los programadores que escribieron el código fuente.

A continuación, se listan las herramientas más conocidas para la automatización de pruebas en la siguiente **Tabla 2-1:**

Tabla 2-1: Herramientas más conocidas para automatización de pruebas unitarias

Herramienta	Tecnología	Licencia
C++ test	C/C++	Comercial
DDCput	C/C++	Código abierto
JMock	Java	Gratuito
Junit	Java	Código abierto

Herramienta	Tecnología	Licencia
Mocha	JavaScript	Código abierto
QUnit	JavaScript	Gratuito
Jasmine	JavaScript	Código abierto
PHPUnit	PHP	Código abierto
Enhance PHP	PHP	Comercial
Go2xunit	Go	Código abierto
Pytest	Python	MIT
PyUnit	Python	Código abierto
Xunit.net	.NET	Código abierto

Fuente: Rodríguez, Piattini y Ebert (2019)

Realizado por: Cuvi J., 2023

Para ejemplificar, se presenta una prueba unitaria desarrollada en el lenguaje JavaScript usando Mocha como herramienta de testing:

```
const chai = require('chai');
const compararFechas = require('./fechaComparacion');
const { expect } = chai;
describe('Comparación de fechas', () => {
  it('La diferencia entre las fechas debe ser como máximo 5 días y la fecha de alquiler no
puede ser menor que la fecha actual', () => {
    const fechaActual = new Date();
    const fechaAlquiler = new Date(fechaActual.getTime() - 2 * 24 * 60 * 60 * 1000);
    const fechaMaxima = new Date(fechaActual.getTime() + 4 * 24 * 60 * 60 * 1000
    expect(compararFechas(fechaAlquiler, fechaMaxima)).to.be.true;
```

2.4. Proceso para la realizar pruebas unitarias

(Yeeply, 2022) señala que realizar pruebas unitarias mediante un proceso estructurado es crucial para asegurar la calidad del software. Este proceso conformado por 6 pasos permite garantizar una cobertura exhaustiva y una identificación eficiente de errores, lo que mejora la confiabilidad y mantenibilidad del código.

1. Identificación de la unidad de código a probar: Por lo general, se trata de una función, método o clase específica, y el objetivo es evaluar cada unidad de manera individual y aislada, manteniendo una cantidad constante de líneas de código en cada prueba.

2. Creación de casos de prueba: Se generan escenarios de prueba que llaman a la unidad en diversas situaciones, incluyendo casos válidos e inválidos. Estos escenarios deben abarcar todas las rutas lógicas posibles dentro de la unidad. Para elaborar los casos de pruebas se debe también definir:

- El entorno de pruebas
- Establecimiento de las respuestas que se esperan obtener

3. Ejecución y Verificación de las pruebas: Se realizan las pruebas diseñadas y se analizan los resultados obtenidos durante la ejecución.
4. Documentación de los resultados: Registra los resultados de cada prueba, detallando los casos de prueba empleados, los resultados previstos, los resultados obtenidos y cualquier error detectado.

En la **Ilustración 2-2** se muestra el proceso recomendado por el autor (Yeeply, 2022) siendo el primer paso la identificación de unidades a probar:

Ilustración 2-2: Proceso para realizar pruebas unitarias

Realizado por: Cuvi J., 2023

2.5. Problemas frecuentes en el desarrollo de pruebas unitarias

Las pruebas unitarias desempeñan un papel crucial en el proceso de desarrollo de software al posibilitar a los desarrolladores identificar errores y fallos en el código antes de que estos lleguen a los usuarios finales. Aunque la escritura de pruebas unitarias puede presentar desafíos, como la complejidad del código y la falta de cobertura de código, es importante que los desarrolladores dediquen tiempo y recursos a la creación y mantenimiento de pruebas unitarias efectivas.

(Romero, 2021) en su estudio indica que los desarrolladores a menudo enfrentan desafíos al implementar pruebas unitarias en su trabajo diario. Estos desafíos pueden surgir debido a diversas razones, como la falta de comprensión del proceso de pruebas unitarias, la complejidad del código a probar, la falta de tiempo o recursos y la resistencia al cambio en la organización.

A continuación, se presentan los problemas más comunes al momento de desarrollar pruebas unitarias:

- **Flaky tests**

También llamados pruebas no determinísticas para (Mascheroni y Irrazábal, 2018) en su artículo señala que las pruebas unitarias deben ser deterministas para garantizar confiabilidad en el despliegue continuo. Sin embargo, algunas pruebas denominadas "flaky tests" pueden arrojar resultados distintos al probar la misma versión del software, lo que se debe principalmente a que incluyen factores externos de difícil control o tiempos de ejecución impredecibles. Estos tests "no deterministas" generan falsos positivos o negativos de forma aleatoria, poniendo en duda la validez de sus resultados. Al ser inestables, dificultan la detección del origen real de los fallos y retrasan innecesariamente los despliegues al no poder confiar plenamente en ellos. Algunas de las causas más comunes de flaky tests son la dependencia de servicios externos como bases de datos o APIs que podrían no responder de manera consistente.

Para (Luo et al, 2021) una prueba no determinista es una prueba que puede producir diferentes resultados cuando se ejecuta en el mismo código. Esto puede deberse a una serie de factores, como:

- Llamadas asíncronas: Una llamada asíncrona es una llamada que no espera a que se complete antes de continuar. Esto puede dar lugar a una prueba no determinista si la llamada asíncrona no se completa antes de que la prueba intente utilizar su resultado.
- Concurrencia: La concurrencia es la ejecución simultánea de múltiples hebras. Esto puede dar lugar a una prueba no determinista si las hebras interactúan de forma no deseada, por ejemplo, debido a carreras de datos, violaciones de la atomicidad o interbloqueos.
- Dependencia de la entrada del usuario: Una prueba puede depender de la entrada del usuario. Esto puede dar lugar a una prueba no determinista si el usuario introduce diferentes valores cada vez que se ejecuta la prueba.

Las pruebas no deterministas pueden ser difíciles de depurar y pueden hacer que sea difícil confiar en los resultados de las pruebas. La investigación concluye que las pruebas no determinísticas son un problema común y que pueden tener un impacto significativo en la calidad del software.

- **No saber cuándo finalizar el desarrollo de las pruebas**

En la investigación realizada por (Velázquez, 2021) menciona que es difícil saber cuándo terminar y es fácil quedarse con la sensación de que no todo está probado correctamente, esto también debido en gran parte a la poca o nula experiencia que se tiene. Si probamos los componentes que vamos creando desde el principio, es mucho más fácil saber cómo atacar el problema y hacer pruebas más completas. El autor propone que si se aplica la metodología TDD (Desarrollo guiado por

pruebas) y se prueban los componentes, funciones y clases a medida que se van creando, es mucho más fácil saber cómo abordar el problema de testing y hacer pruebas más completas.

- **Dejar las pruebas para el final**

De acuerdo con (Velázquez, 2021) dejar las pruebas para el final del desarrollo es un problema muy común y esto implica varias desventajas como mayor esfuerzo mental para probar un programa grande al final sobre un programa pequeño. Esto dificulta saber por dónde empezar y cuándo terminar las pruebas siendo más complicado hacer estimaciones precisas sobre los recursos necesarios, disminuyendo la flexibilidad para ajustar fechas o alcance. Existe una tendencia a restar importancia a los errores cuando sólo queda entregar, en especial cercano a la fecha límite. El autor menciona además que al no probar los programas durante el desarrollo implica que se pierde la ventaja de escribir código más limpio y fácil de probar, asegurando su calidad y estabilidad. En general, probar desde el inicio permite reducir el esfuerzo, aumentar la calidad y minimizar riesgos mediante pruebas más completas y automatizadas, evitando posponerlas hasta el final del desarrollo.

Con base en la investigación realizado por (Meenakshi, Swami Naik y Raghavendra Reddy, 2014) señalan que las pruebas de software realizadas durante el proceso de desarrollo pueden reducir los costos de corrección de errores en un 10 a 100 veces en comparación con las pruebas realizadas después de la implementación del software.

- **Falta de cobertura**

Con base en el estudio de (Andrade, 2021) afirma que las estrategias de cobertura de pruebas ofrecen un medio efectivo para monitorear continuamente la calidad de las pruebas y abordar aquellas áreas que aún no han sido validadas. Los principales factores que inciden en la falta de cobertura de las pruebas unitarias son la presión por entregar rápido sin asignar tiempo para pruebas, la falta de capacitación en diseño de casos de prueba, el código no diseñado para ser testeado, la presencia de dependencias y los constantes cambios que vuelven obsoletas las pruebas existentes.

Para (Palamarchuk, 2020) la cobertura de prueba marca qué porcentaje del código se ha probado, pero tampoco significa que se haya probado en todas las situaciones, sin embargo, no siempre se logra abarcar una buena cobertura en las debidas a una serie de factores que afectan la falta de cobertura en las pruebas unitarias como la falta de tiempo, falta de motivación, falta de compresión y la falta de gestión.

- **Complejidad del código a probar**

(Alores, 021) indica que en proyectos de software complejos o sumamente grandes puede ser difícil identificar y aislar unidades de código para probar individualmente. Esto puede llevar a pruebas

unitarias ineficaces o a la omisión de pruebas importantes. El 33% de los encuestados de los desarrolladores en Stack OverFlow consideren que construir productos con requisitos poco específicos es su mayor reto dado que las principales razones de la complejidad de los proyectos de software es el cambio constante de los requisitos.

Para (Jayatilleke y Lai, 2018) los cambios en los requisitos iniciales pueden propagarse y afectar negativamente otras fases del desarrollo si no son gestionados adecuadamente desde un inicio, dado que podrían invalidar diseños preexistentes y llevar a la introducción de errores en la codificación al no estar alineada con las nuevas necesidades, dando como resultado un sistema final que no satisfaga realmente las expectativas del usuario debido a que no se siguió con la planificación reduciendo el tiempo en los ciclos posteriores.

Para (Auditor, 2019) un código de alta complejidad o no testeable se refiere a aquel cuya estructura hace difícil o imposible probarlo mediante pruebas unitarias. Suele estar fuertemente acoplado y depender de factores externos que dificultan su aislamiento, como APIs, archivos o bases de datos. El autor de la investigación considera como código no testeable aquella que sobrepasa un punto de complejidad por encima de 50.

A continuación, se presenta los niveles de puntuación ciclomática junto a su categoría en la **Tabla 2-2:**

Tabla 2-2: Niveles de puntuación en complejidad ciclomática

Puntuación	Categoría del código
De 0 a 10	Código testeable, complejidad aceptable
De 11 a 15	Riesgo medio, más complejidad
De 16 a 20	Código de alto riesgo, demasiadas decisiones para una unidad de código.
Por encima de 50	Código no testeable

Fuente: (Owasp, 2017)
Realizado por: Cuvi J., 2023

- **Falta de tiempo o de recursos**

Para (Ahmed, 2019) los desarrolladores a menudo se enfrentan a plazos ajustados y a la presión para entregar nuevas funcionalidades. Esto puede hacer que dediquen menos tiempo a desarrollar pruebas unitarias adecuadas o que no cuenten con los recursos necesarios, como herramientas de prueba automatizadas, para realizar pruebas eficientes. Además, puede hacer que los desarrolladores se centren en la implementación de nuevas funcionalidades en detrimento de la calidad del código y la robusteza del software. También ocasiona que los desarrolladores tengan

que tomar decisiones difíciles sobre qué pruebas realizar y cuáles omitir, lo que puede afectar la calidad del software.

Conforme con el estudio desarrollado por (Cristina López, 2022) señala que en la mayoría de los casos analizados durante la planificación de un proyecto de desarrollo de software el equipo de pruebas cuenta con unos días determinados para realizar su labor, los cuales suelen coincidir con fechas cercanas a un despliegue o entregas. Sin embargo, cuando se presentan retrasos durante la fase de desarrollo, es común que el tiempo asignado para pruebas sea el más afectado, llegando incluso a eliminarse por falta de días convirtiéndose en un problema muy recurrente.

- **Depender de entornos complejos**

Otro problema común identificado en el estudio realizado por (Díaz, 2019) es la dependencia de componentes del entorno como bases de datos, servicios externos, sistemas de archivos, etc. Esto dificulta la portabilidad y mantenimiento de los tests. El autor también recomienda utilizar Mocks y Stubs para aislar el código bajo prueba y sus dependencias. De esta forma las pruebas solo validan el comportamiento de la unidad en sí, sin verse afectadas por fluctuaciones en otros componentes.

(Velázquez, 2021) señala que para pruebas unitarias es contraproducente replicar el entorno de producción real, ya que incrementa la fragilidad de los tests. Al depender de servicios externos como bases de datos, servicios web, colas de tareas, aumentando las posibilidades de que las pruebas fallen por problemas de configuración en lugar de errores de código. El objetivo de las pruebas unitarias es comprobar el correcto funcionamiento de un componente de forma aislada, no su integración completa.

- **Falta de patrones de diseño**

Para (Capel, Grimán y Garví, 2016) no existe una guía unificada de patrones y mejores prácticas aceptada mundialmente para el diseño de pruebas unitarias lo cual dificulta su desarrollo. Esto genera que cada desarrollador las implemente de forma diferente, afectando su mantenibilidad y reusabilidad. Proponen la necesidad de estandarizar patrones de testing para cada lenguaje/framework, de forma que se facilite el desarrollo de pruebas consistentes y de calidad. Puede resultar difícil escribir pruebas unitarias efectivas y completas, especialmente en proyectos grandes y complejos. Es difícil mantener las pruebas unitarias actualizadas a medida que el código evoluciona, lo que puede llevar a que las pruebas se vuelvan obsoletas y pierdan su valor. La falta de patrones de diseño puede hacer que las pruebas no sean lo suficientemente flexibles, lo que puede afectar la capacidad del software para adaptarse a cambios en el requisito.

- **Falta de métricas de evaluación**

Para (Orozco y Righi, 2018) en su investigación señala que la mayoría del código no está cubierto por pruebas unitarias, pruebas de integración, ni siquiera para los procesos más significativos que cubre el proyecto en cuestión. Al carecer de métricas claras sobre cobertura de código y automatización de pruebas, éstas pierden trazabilidad y dejan de ser útiles para garantizar la no regresión de funcionalidades. Esto genera la percepción de que representan una carga de trabajo sin beneficios reales, afectando negativamente el incentivo para su creación y mantenimiento. También se menciona que los desarrolladores no ven la inversión de tiempo en generar pruebas unitarias de sus frutos, y con razón, ya que, en el mejor de los casos, las pruebas una vez construidas solo las ejecuta el desarrollador. La investigación concluye afirmando que se requiere la implementación urgente de métricas, herramientas para medir claramente el avance de pruebas y procesos de integración continua que las ejecuten automáticamente y provean retroalimentación continua. Los autores de esta investigación recomiendan el flujo de pruebas presentado en la **Ilustración 2-3** donde el primer proceso es elegir medidas a realizar para continuar con los demás procesos recomendados:

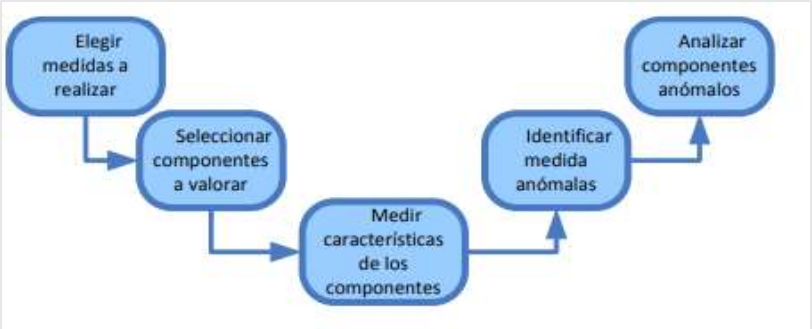


Ilustración 2-3: Proceso de medición del software

Realizado por: Cuvi J., 2023

2.6. Aplicación web

Para (Oliveros, Danyans y Mastropietro, 2014) una aplicación web es un sistema software al que se accede a través de Internet o Intranet y conforman una clase especial de aplicaciones de software que se construye de acuerdo con ciertas tecnologías y estándares. La diferencia fundamental entre una aplicación web y un simple sitio web radica en la capacidad que brinda al usuario para interactuar con la lógica de negocio del servidor.

2.7. Arquitectura Cliente – Servidor

Con base en el estudio de (GuidoM. Mantilla, JoffreL. Leon y LuisA. Jurado, 2018) la arquitectura cliente-servidor es un modelo ampliamente utilizado en la computación distribuida, donde dos componentes principales, el cliente y el servidor, desempeñan roles distintos. El cliente solicita

servicios o recursos, como datos o funciones, mientras que el servidor proporciona estos servicios en respuesta a las solicitudes del cliente. En la **Ilustración 2-4** se representa el funcionamiento de la arquitectura cliente – servidor:

Ilustración 2-4: Arquitectura cliente - servidor

Fuente: (Espinoza et al, 2017)

2.8. Arquitectura basada en componentes

En el trabajo realizado por (Sánchez Ledesma, 2018) menciona que una aplicación basada en componentes es un sistema de software que se construye ensamblando componentes de software preexistentes y reutilizables. Estos componentes están diseñados para trabajar en conjunto y brindar una funcionalidad específica o un conjunto de funcionalidades. Además, la reutilización de componentes permite una mayor modularidad y flexibilidad en el diseño de aplicaciones.

Para (Peredo Valderrama, 2020) la arquitectura basada en componentes de software es una técnica utilizada en el desarrollo de software que consiste en la selección y ensamblaje de componentes de software predefinidos para crear un sistema.

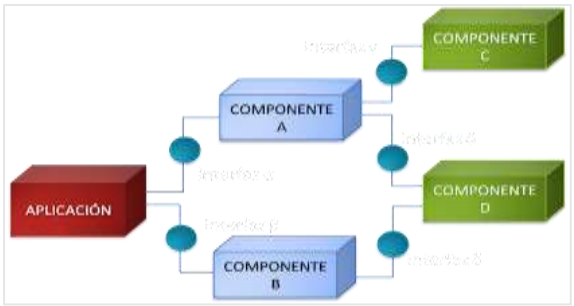


Ilustración 2-5: Arquitectura basada en componentes

Fuente: (Poblete, 2021)

2.9. Herramientas para el desarrollo de la aplicación web

Para el desarrollo del presente trabajo de titulación, se ha contemplado el uso una pila de tecnologías que facilitan el desarrollo las cuales se mencionan a continuación:

- Open AI Api
- Node JS
- JavaScript

- React JS
- FireBase
- Material UI
- Express JS

2.9.1. OPEN AI API

Esta herramienta pertenece a la empresa OPEN AI la cual es una organización de investigación en inteligencia artificial sin fines de lucro fundada en 2015 por expertos como Elon Musk y Sam Altman. Su objetivo es desarrollar IA de forma segura y beneficiosa para la humanidad. Realizan investigación de punta en áreas como redes neuronales profundas, procesamiento de lenguaje natural, visión por computadora y robótica (García, 2023).

La API de Open AI permite a los desarrolladores acceder a sus modelos de lenguaje generados por redes neuronales como CLIP, DALL-E, GPT-3 y más modelos que se van sumando a medida que avanza el desarrollo de nuevas versiones. Esto implica que los desarrolladores pueden aprovechar la potencia de la inteligencia artificial de Open AI para agregar características de procesamiento de lenguaje natural, generación de texto y otras capacidades avanzadas en sus propias soluciones. La autenticación en la API se realiza mediante un token de acceso personal que debe mantenerse confidencial. Ofrece endpoints REST para interactuar con los modelos mediante solicitudes HTTP en formato JSON (OpenAI, 2023).

2.9.2. Características de OPEN AI API

La API de Open AI permite acceder de manera sencilla a sus avanzados modelos de inteligencia artificial desde cualquier lugar. Implementa una interfaz sencilla que sigue estándares REST, haciéndola fácil de usar para desarrolladores de diversos lenguajes de programación. Ofrece importante documentación y soporte técnico que guían en la integración correcta.

A continuación, se detallan las principales características que la convierten en una herramienta para trabajar con conceptos de IA de forma escalable y segura.

- **Accesibilidad:** Para (Miranda, 2023) la accesibilidad la API permite a desarrolladores acceder fácilmente a avanzados modelos de IA mediante una interfaz web. Menciona que, si bien la moda es montar Chat GPT en donde se pueda, OPEN AI muestra que su API puede usarse para algo más que una interfaz de chat común. La empresa tiene algunos ejemplos de lo que hacen algunas compañías que ya acceden al modelo de IA.

- **Documentación extensa:** (OpenAI, 2023) menciona que se cuenta con amplia documentación en su propio portal web, así como guías, tutoriales y SDKs para facilitar la integración. A continuación, se detalla toda la documentación de distintos tópicos disponibles:

1. Documentación técnica: Contiene detalles completos sobre la arquitectura de la API, autenticación, endpoints disponibles, parámetros de entrada/salida, límites, etc. Ayuda a familiarizarse con su funcionamiento.
2. Guías de inicio: Incluyen tutoriales paso a paso para realizar las primeras solicitudes a los modelos, probar ejemplos simples y entender el flujo general de trabajo con la API.
3. Referencia de modelos: Cada modelo como GPT-3, CLIP o DALL-E cuenta con su propia documentación explícita sobre capacidades, límites y buenas prácticas de uso.
4. SDK de diferentes lenguajes: Facilitan la integración mediante librerías para Python, JavaScript, Java, Go, C#, Ruby y más.
5. Foro de soporte técnico: Para resolver dudas mediante interacción directa con el equipo de Open AI y otros usuarios.

- **Seguridad:** Open AI emplea técnicas como encriptación, autenticación, control de acceso y monitoreo constante para mantener la privacidad y seguridad de los datos en su API.

  A continuación, se presenta las distintas estrategias de seguridad que emplea:

1. Open AI cifra todos los datos en reposo (AES-256) y en tránsito (TLS 1.2+), y utiliza estrictos controles de acceso para limitar quién puede acceder a los datos. El equipo de seguridad cuenta con personal disponible las 24 horas del día, los 7 días de la semana y los 365 días del año para responder a cualquier incidente potencial de seguridad (Enterprise privacy, 2023).
2. Los datos de la API se almacenan en los sistemas de Open AI y de sus subprocesadores, y se transmiten mediante Transport Layer Security (TLS), garantizando su encriptación (OpenAI, 2023).
3. La interfaz sigue estándares REST con endpoints que se consumen mediante peticiones HTTP/JSON, lo que la hace familiar para quienes trabajen con APIs. Provee seguridad mediante HTTPS y firmas en las solicitudes (OpenAI, 2023).

- **Privacidad:** En el portal de (Enterprise privacy, 2023) se menciona que no comparte modelos entrenados ni información de usuarios con terceros. Desde el 1 de marzo de 2023, Open AI no utiliza los datos enviados por los clientes a través de su API para entrenar o mejorar sus modelos, a menos que el cliente decida compartir sus datos explícitamente. Los datos enviados a través de la API se retienen por un máximo de 30 días únicamente con fines de monitoreo de abuso y uso indebido eliminándose posteriormente.

- **Modelos Pre–Entrenados:** (Avila, 2022) en su investigación señala que modelos de IA de lenguaje generativo conocidos como GPT, que significa "Generative Pre-trained Transformer", son una familia de modelos desarrollados por Open AI. Estos modelos han demostrado un rendimiento sobresaliente en tareas de procesamiento de lenguaje natural y generación de texto. La base de la arquitectura de estos modelos es el Transformer, un tipo de arquitectura de red neuronal que ha mostrado ser altamente efectiva para procesar secuencias de datos, como texto.

Entre los modelos Bases que se encuentra disponible se detalla a continuación:

1. GPT-3: Este es uno de los modelos de lenguaje natural más avanzados y sofisticados disponibles. Fue entrenado en una gran cantidad de datos de texto utilizando una arquitectura de red neuronal conocida como "transformer".
2. InstructGPT: Esta es una versión más reciente del modelo GPT-3 que es más rápida y menos costosa. Incluye varios submodelos, incluyendo text-curie, text-davinci-001 y text-davinci-002, que tienen capacidades y limitaciones variadas.
3. ChatGPT: Este es un modelo diseñado específicamente para generar respuestas en conversaciones. Forma parte de la serie de modelos GPT-3.5, que también incluye gpt-3.5-turbo, text-davinci-003 y otros.
4. DALL-E: Modelo diseñado para generar imágenes realistas que coinciden con descripciones detalladas en lenguaje natural.

- **Precisión**: Conforma (Biever, 2023) los modelos de inteligencia artificial pre-entrenados que ofrece la OPEN AI API demuestran en general altos niveles de precisión, especialmente para tareas de visión por computadora, procesamiento del lenguaje natural y generación de texto. Sin embargo, dada la gran cantidad de información a la que estos modelos deben responder en ocasiones pueden cometer errores u obtener resultados imprecisos, sobre todo para consultas específicas o contextuales. Los niveles de precisión varían entre los diferentes modelos y tareas, aunque suelen estar por encima del 80% en tareas de clasificación semántica y extracción de información de internet o documentos de texto. Sin embargo, es fundamental considerar los márgenes de error inherentes al usar estos sistemas de inteligencia artificial, que aún están en mejora continua.

A continuación, se presenta en la **Tabla 2-3** los niveles de precisión según el modelo:

Tabla 2-3: Niveles de precisión

Modelo	Nivel de precisión	Descripción
CLIP	90%+	Para visión por computadora suelen ser muy precisos reconociendo y clasificando imágenes.
GPT-3	80-90%	Demuestra altos niveles de comprensión lectora y generación de textos consistentes. Su precisión depende del contexto y puede dar respuestas vagas o inexactas.
Chat GPT	85-95%	Suelen ser bastante precisos en diálogos simples, aunque aún se les dificulta seguir conversaciones largas y complejas.
DALL-E	65%+	Para generación de imágenes son menos precisos que otros basados en texto o visión.

Fuente: (Biever, 2023)

Realizado por: Cuvi J., 2023

2.9.3. Comportamiento de OPEN AI API

La interacción con la API varía mucho en su comportamiento de acuerdo con el modelo que se esté usando. Los modelos entrenados buscan responder de manera relevante, coherente y contextualizada utilizando un filtrado de contenido por cada modelo que se emplee.

- **Filtrado de contenido y restricciones:** (Mrbullwinkle, 2023) señala que OPEN AI API incorpora un mecanismo de filtrado, este mecanismo evalúa tanto la solicitud como la respuesta utilizando varios modelos de clasificación que están diseñados para identificar y prevenir la generación de contenido dañino. Este sistema de filtrado se encarga de detectar y abordar distintas categorías de contenido potencialmente perjudicial en las solicitudes y en las respuestas generadas. Es relevante tener en cuenta que las configuraciones de la API y el diseño de las aplicaciones pueden influir en el comportamiento que tendrán las respuestas generadas en consecuencia, en el comportamiento del filtro. Cuando se identifica contenido en la categoría de "seguro", se le asignan etiquetas de anotación, pero no se somete al proceso de filtrado y no puede modificarse.

A continuación, en la **Tabla 2-4** se presenta la clasificación de filtrado de contenido por categoría y descripción que emplea cada modelo.

Tabla 2-4: Categorías de filtrado de contenido

Categoría	Descripción
Odio	La categoría de odio describe los ataques o usos del lenguaje que incluyen un lenguaje peyorativo o discriminatorio con referencia a una persona o grupo de identidad sobre la base de ciertos atributos diferenciadores de estos grupos.
Sexual	La categoría sexual describe el lenguaje relacionado con los órganos anatómicos y los genitales, las relaciones románticas, los actos representados en términos eróticos y los actos sexuales físicos.
Violencia	La categoría de violencia describe el lenguaje relacionado con acciones físicas destinadas a herir o dañar alguien o algo; describe armas, etc.
Autolesiones	La categoría de autolesiones describe el lenguaje relacionado con acciones físicas destinadas a herir, lesionar o dañar intencionadamente el propio cuerpo.

Fuente: (Mrbullwinkle, 2023)

Realizado por: Cuvi J., 2023

- **Serie de modelos entrenados**: OPEN AI API puede producir comportamientos y respuestas muy diferentes dependiendo del modelo empleado y los parámetros con los que fue entrenado. Para seleccionar un modelo en específico es importante sopesar los requerimientos específicos entre creatividad y precisión. Mientras modelos como GPT-3 pueden generar texto novedoso, suelen cometer más errores factuales que alternativas como GTP4 optimizadas para contextos técnicos y creatividad (OpenAI, 2023).

Los modelos se entrenan ajustando millones de parámetros internos mediante un proceso de aprendizaje automático determinando el comportamiento que se obtendrá como resultado. Estos parámetros simbolizan las conexiones entre los nodos de la red neuronal del modelo (Peiró, 2020).

A continuación, en la **Tabla 2-5** se detalla todos los modelos disponibles junto a su número parámetros de entrenamiento:

Tabla 2-5: Modelos entrenados

Modelos	Descripción	Parámetros
GPT 4	Un conjunto de modelos que mejoran GPT-3.5 y pueden comprender, así como generar lenguaje natural o código.	100 billones
GPT 3.5	Un conjunto de modelos que mejoran GPT-3 y pueden comprender, así como generar lenguaje natural o código.	175 mil millones
GPT base	Un conjunto de modelos que pueden comprender y generar lenguaje natural o código base.	17.000 millones

Modelos	Descripción	Parámetros
DALL-E	Un modelo que puede generar y editar imágenes a partir de una descripción en lenguaje natural.	12 mil millones
Whisper	Un modelo que puede convertir audio a texto.	39 millones

Fuente: (OpenAI, 2023)

Realizado por: Cuvi J., 2023

2.9.4. Funcionamiento de OPEN AI API

OPEN AI API permite acceder a poderosos modelos de inteligencia artificial pre-entrenados de manera sencilla a través de una interfaz basada en REST. El funcionamiento básico de la API consiste en que los desarrolladores pueden realizar solicitudes HTTP a diferentes endpoints, pasando parámetros como datos de entrada en formato JSON. Luego, la API procesa las solicitudes a través de sus servicios y devuelve la respuesta en formato JSON. Algunos servicios como el generador de textos son asíncronos, por lo que las respuestas tardan unos segundos. De esta forma, la API abstracte la complejidad del machine learning y habilita su uso para distintas aplicaciones de manera flexible a través de un simple protocolo basado en web (OpenAI, 2023).

- **Interpretación de entradas**: Para (Grinberg, 2020) los modelos basados en redes neuronales profundas emplean la técnica de atención para anticipar la palabra que sigue en una oración. Son entrenado con un conjunto de datos que supera los mil millones de palabras y puede generar texto con una gran exactitud a nivel de caracteres. Su estructura incluye dos partes esenciales: un codificador y un decodificador. El codificador toma la palabra anterior de la oración y genera una representación vectorial de la misma, que luego se procesa mediante un mecanismo de atención para predecir la siguiente palabra. Por otro lado, el decodificador utiliza la palabra previa y su representación vectorial como entrada, generando una distribución de probabilidad para todas las posibles palabras basada en esas entradas.

- **Lenguajes soportados:** La API ofrece soporte y funcionamiento para diversos lenguajes de programación que facilitan su integración en proyectos tanto web como móviles. Entre los lenguajes soportados destacan: JavaScript, Java, C/C++, GO, RUBY, PHP, Python y más. La API apoya la integración multiplataforma con las principales tecnologías de desarrollo actuales, cubriendo tantas aplicaciones web, móviles, Backend, análisis de datos y más (OpenAI, 2023).
- **Autenticación:** Utiliza un sistema de claves API para autenticar solicitudes a sus servicios. Cada usuario recibe una clave API única que debe incluirse en las cabeceras HTTP de las solicitudes.

- **Autenticación con claves de API:** para este tipo de autenticación, todas las solicitudes de API deben incluir la clave de API en el encabezado HTTP api-key. El inicio rápido proporciona una guía sobre cómo realizar llamadas con este tipo de autenticación.

- **Interfaz REST:** Implementa endpoints REST para interactuar mediante peticiones HTTP/JSON para lo cual es necesario cumplir obligatoriamente con los siguientes parámetros al momento de realizar una petición, a continuación, se presenta todos los parámetros:

Tabla 2-6: Cuerpo de la solicitud de la API

Parámetro	Tipo	¿Obligatorio?	Descripción
ApiKey	String	Obligatorio	La clave de autentificación y autorización.
Messages	Array	Obligatorio	Colección de mensajes de contexto asociados a esta solicitud de finalización de chat.
Model	String	Obligatorio	El modelo utilizado para la finalización
Promt	Matriz, String	Obligatorio	Una cadena de orden usado para la finalización para el modelo.
Content-Type	String	Obligatorio	El tipo de respuesta que se envia y la forma que debe responder.
Authorization	Object	Obligatorio	Cadena de autorización donde va el encabezado de la API KEY
Created	Int	Opcional	La unidad de tiempo en el que fue creado en segundos
Role	ChatRole	Opcional	Rol asociado a esta carga de mensajes

Fuente: (OpenAI, 2023).

Realizado por: Cuvi J., 2023

- **Pago por uso:** En base en la información presentada por el propio portal de (Stripe, 2023) señala que la empresa Open AI se asoció con Stripe para comercializar su innovadora tecnología de inteligencia artificial. Aprovechando el potente conjunto de herramientas de Stripe, logrando poner en funcionamiento un mecanismo de cobros internacional para diversas soluciones en apenas unos pocos períodos. Gracias a la colaboración con Stripe, ahora se tiene la capacidad

de procesar pagos de manera certera y a escala global, impulsando así el despliegue y aplicación práctica de su revolucionaria tecnología de IA

Cada cuenta de usuario tiene asociado un límite de crédito en dólares que se reduce a medida que se realizan solicitudes a la API, al superar el límite, la API bloquea el acceso hasta recargar la cuenta con más fondos. Los modelos de la serie GPT cobran por fragmentos de caracteres denominado tokens donde 4 caracteres equivalen a 1 token.

A continuación, se presentan los detalles del pago por cada modelo según las características en las siguientes tablas:

Tabla 2-7: Detalles de pagos del modelo GPT4

Modelos de lenguaje	Modelo	Costo por entrada	Costo por salida
GPT4	8K context	$0.03 / 1K tokens	$0.06 / 1K tokens
	32K context	$0.06 / 1K tokens	$0.12 / 1K tokens

Fuente: (OpenAI,2023)

Realizado por: Cuvi J., 2023

Tabla 2-8: Detalles de pagos del modelo 3.5 Turbo

Modelos de lenguaje	Modelo	Costo por entrada	Costo por salida
GPT-3.5 Turbo	4K context	$0.0015 / 1K tokens	$0.002 / 1K tokens
	16K context	$0.003 / 1K tokens	$0.004 / 1K tokens

Fuente: (OpenAI,2023).

Realizado por: Cuvi J., 2023

Tabla 2-9: Detalles de pagos del modelo Fine-tuning models

Modelos de lenguaje	Modelo	Training	Costo por entrada	Costo por salida
Fine-tuning models	babbage-002	$0.0004 / 1K tokens	$0.0016 / 1K tokens	$0.0016 / 1K tokens
	davinci-002	$0.0060 / 1K tokens	$0.0120 / 1K tokens	$0.0120 / 1K tokens
	GPT-3.5 Turbo	$0.0080 / 1K tokens	$0.0120 / 1K tokens	$0.0160 / 1K tokens

Fuente: (OpenAI,2023).

Realizado por: Cuvi J., 2023

Tabla 2-10: Detalles de pagos del modelo Embedding models

Modelos de lenguaje	Modelo	Uso
Embedding models	Ada v2	$0.0001 / 1K tokens

Fuente: (OpenAI, 2023).

Realizado por: Cuvi J., 2023

Tabla 2-11: Detalles de pagos del modelo Base models

Modelos de lenguaje	Modelo	Costo por entrada
Base models	babbage-002	$0.0004 / 1K tokens
	davinci-002	$0.0020 / 1K tokens

Fuente: (OpenAI, 2023)

Realizado por: Cuvi J., 2023

2.9.5. JavaScript

Para (Carion, 2020) JavaScript es un lenguaje de programación utilizado para crear sitios web interactivos y dinámicos. Es un lenguaje de alto nivel e interpretado que se ejecuta en el lado del cliente o en el lado del servidor. JavaScript Object Notation es un formato de intercambio de datos liviano que es fácil de leer y escribir para los humanos, y fácil de analizar y generar para las máquinas. JSON Data Definition Format (JDDF) y JSON Type Definition (JTD) son formatos propuestos para describir la estructura de los mensajes JSON.

2.9.6. React JS

Conforme a (Aggarwal, 2018) React JS es una biblioteca de JavaScript utilizada para desarrollar interfaces de usuario reutilizables. Fue desarrollada por Facebook y se utiliza para crear aplicaciones web de una sola página, aplicaciones móviles y aplicaciones de escritorio. React JS utiliza un enfoque basado en componentes para la construcción de interfaces de usuario, lo que significa que las interfaces de usuario se dividen en componentes más pequeños y reutilizables.

2.9.7. Express JS

Para (Aguayo Cáceres y Freire Orozco, 2021) Express JS es un framework de servidor web flexible para Node.js, ideal para construir APIs y aplicaciones web full-stack. Proporciona un conjunto robusto de características para enrutamiento, middleware, gestión de solicitudes y respuestas. Express simplifica tareas como definir endpoints y middlewares antes de enviar respuestas, haciendo más simple y placentero el desarrollo de aplicaciones web y APIs en Node.js

2.9.8. Firebase

Para (Páez y Vinueza, 2019) firebase es un conjunto integral de recursos diseñado para respaldar el desarrollo de aplicaciones móviles y web. Proporciona una amplia gama de herramientas y servicios destinados a facilitar a los desarrolladores la creación y mejora de sus aplicaciones.

Entre las funciones que ofrece se encuentran la verificación de usuarios, una base de datos que se actualiza en tiempo real, la posibilidad de almacenar datos en la nube entre otras funciones.

2.9.8.1. Firestore

Para (Bahtiar Semma et al, 2023) firestore se destaca como una base de datos NoSQL a nivel mundial que proporciona funcionalidades clave, como el almacenamiento en tiempo real, la escalabilidad y una integración perfecta con Firebase. Estas características hacen que sea una elección idónea para una amplia variedad de aplicaciones y casos de uso.

2.9.9. Metodología Scrumban

Para (Petricioli y Fertalj, 2022) Scrumban es una metodología híbrida que combina elementos de Scrum y Kanban, con un enfoque en la mejora continua, la flexibilidad, la adaptabilidad, la limitación del trabajo en progreso y la visualización del proceso. Esta metodología se compone de 6 etapas, las cuales se describen a continuación:

- **Análisis preliminar:** El análisis preliminar en Scrumban se refiere a la etapa inicial de evaluación donde se identifican los objetivos del proyecto y se determinan los elementos necesarios para su ejecución.
- **Planificación:** La fase de planificación implica la definición de tareas, prioridades y asignación de trabajo, así como la estimación de tiempos para las actividades de desarrollo.
- **Desarrollo**: Es una parte fundamental del ciclo de vida de un proyecto en Scrumban y se caracteriza por ser flexible y adaptable a los cambios además se refiere al proceso de creación y mejora constante del producto o software componiéndose de las siguientes subetapas:
  - Tipos y roles de usuario
  - Diseño de la base de datos
  - Diseño de la interfaz de usuario
  - Codificación o Diagramas UML
- **Pruebas:** En esta etapa el equipo realiza validaciones de cada tarea asignada y cumplida en el tiempo de desarrollo.
- **Despliegue:** Implica la implementación y puesta en marcha de las funcionalidades desarrolladas durante las iteraciones del proyecto.
- **Cierre:** Marca el fin del proyecto y la transición hacia la operación y el mantenimiento continuo del producto.

2.10. Normas ISO/IEC 25010

El modelo de calidad del producto, en consonancia con las directrices de las normas ISO/IEC 25010, presenta una estructura sólida compuesta por ocho características principales que a su vez

se desglosan en subcaracterísticas. La versatilidad de este enfoque de calidad lo hace aplicable tanto a sistemas informáticos como a productos de software, conforme a los estándares establecidos por la (ISO/IEC 25010, 2011). La **Ilustración 2-6** se proporciona una representación visual de cómo estas características se relacionan con las propiedades del software y del sistema en un contexto amplio. El presente trabajo se centrará en evaluar el comportamiento temporal, una de las subcaracterísticas de la Eficiencia de desempeño

Ilustración 2-6: Características de calidad de las normas ISO/IEC 25010

Fuente: ISO/IEC 25010, 2011

2.10.1. Eficiencia de desempeño

Según (ISO/IEC 25010, 2011) la eficiencia en el software se refiere a la capacidad del software para realizar sus funciones previstas de manera oportuna y eficiente en términos de recursos. La característica de Eficiencia de desempeño se subdivide en 3 subcaracterísticas: comportamiento temporal, utilización de recursos y capacidad.

A continuación, se detalla individualmente las 3 subcaracterísticas que conforman la Eficiencia de desempeño:

- **El comportamiento temporal:** Pertenece a la subcaracterística de eficiencia denominada "Comportamiento en el tiempo". Esta subcaracterística se refiere a la capacidad del software para procesar y responder en tiempos adecuados, así como a la capacidad de mantener un rendimiento constante a lo largo del tiempo.
- **Utilización de recursos:** Las cantidades y tipos de recursos utilizados cuando el software ejecuta sus funciones en condiciones particulares.
- **Capacidad:** Grado en que los límites máximos de un parámetro de un producto o sistema software se ajustan a los requisitos establecidos.

CAPÍTULO III

3. MARCO METODOLÓGICO

El capítulo expone la metodología Scrumban aplicada al desarrollo del sitio web, abarcando el análisis inicial, planificación, desarrollo del frontend y backend mediante codificación, pruebas y de integración, despliegue y cierre. Además, describe aspectos técnicos como el modelado de datos, la arquitectura, la selección de tecnologías de inteligencia artificial proporcionada por OPEN AI API, el diseño visual y la gestión de riesgos durante todas las fases del proyecto.

3.1. Tipo de estudio

El presente trabajo se relaciona con dos tipos de estudio:

- **Investigación Aplicada:** Siendo el tipo de estudio más relevante en el presente trabajo debido a que requiere demostrar los conocimientos existentes para lograr desarrollar la solución tecnológica más adecuada haciendo uso de una combinación de varias tecnologías para desarrollar el sitio web para la generación automatizada de casos de pruebas unitarias utilizando la API de Open AI.
- **Investigación cuantitativa:** Esta investigación permitirá realizar pruebas comparativas con las personas involucradas en el desarrollo de pruebas unitarias misma que permitirá evaluar la eficiencia de los casos de prueba generados automáticamente en términos del comportamiento temporal.

3.2. Métodos y Técnicas

A continuación, se detallan los métodos y técnicas aplicadas de acuerdo con los objetivos planteamos del presente trabajo en la **Tabla 3-1.**

Tabla 3-1: Métodos y Técnicas

Objetivos	Métodos	Técnicas	Fuentes
Identificar los problemas más comunes al momento de desarrollar casos de pruebas unitarias	Inductivo Analítico	Revisión de documentos	Artículos académicos y científicos Internet Blogs y canales de desarrolladores
Estudiar las características, comportamiento y funcionamiento de la API de OPEN AI	Inductivo	Revisión de documentos	Documentación oficial de la API Blogs y canales de desarrolladores
Desarrollar los componentes de ingreso de funciones,	Scrumban	Tablero Kanban Flujos de trabajo	Trabajos relacionados

Objetivos	Métodos	Técnicas	Fuentes
configuración y los componentes de salida de resultados		Listado de pendientes visible	Videotutoriales Artículos académicos y científicos Internet
Evaluar el comportamiento temporal del sitio web mediante la norma ISO/IEC 25010	Cuantitativo Observación Experimental	Pruebas	Norma ISO/IEC 25010 Desarrolladores Casos de pruebas Requisitos

Realizado por: Cuvi J., 2024

3.2.1. Método Inductivo

Es un enfoque de investigación que se centra en la recolección de datos y la observación de patrones o regularidades para llegar a conclusiones generales o teorías. Por ello, con la información que se recolecte de las diferentes fuentes de información se concluirá de manera general los problemas que enfrentan los desarrolladores y testers para desarrollar casos de pruebas unitarias.

3.2.2. Método Analítico

Es un enfoque de investigación que se basa en el análisis detallado y la descomposición de un problema en sus componentes más pequeños. Este método implica examinar y estudiar las partes individuales para comprender cómo se relacionan y contribuyen al todo. El método analítico busca identificar las causas y los efectos, mismas que serán usadas para identificar los problemas al desarrollar casos de pruebas unitarias por parte de los desarrolladores y testers.

3.2.3. Metodología Scrumban

Para (Petricioli y Fertalj, 2022) Scrumban es un enfoque de desarrollo de proyectos que se basa en principios iterativos e incrementales, centrados en la adaptación continua y la colaboración haciendo uso de tareas en tableros. Las 6 fases por implementar en este proyecto son las siguientes:

- Fase de análisis
- Fase de planificación
- Fase de desarrollo
- Fase de pruebas
- Fase despliegue
- Fase de cierre

3.2.4. Método de Observación

Es un enfoque de investigación que implica la observación sistemática y el registro de fenómenos o comportamientos en su contexto natural. El método de observación se utiliza para recolectar datos sobre eventos, interacciones o situaciones sin intervenir activamente. Se observará el tiempo que tarda el desarrollar pruebas unitarias con y sin la aplicación por parte de los desarrolladores y testers.

3.2.5. *Método cuantitativo*

Es un enfoque de investigación que se basa en la recopilación y el análisis de datos numéricos para obtener conclusiones y respuestas precisas. En el presente trabajo se planea evaluar el comportamiento temporal del sitio web basado en la norma ISO/IEC 25010.

3.2.6. *Método experimental*

Es un enfoque de investigación que implica la manipulación controlada de variables y la observación de los efectos resultantes. En un experimento, se establecen condiciones controladas y se realizan medidas sistemáticas para determinar la relación causal entre variables.

3.3. Determinación del comportamiento temporal

En este punto se abarca la determinación del comportamiento temporal del sitio web según la norma ISO/IEC 25010. Describe la unidad de análisis, los indicadores para evaluar el comportamiento temporal en cuanto a tiempo de respuesta, espera y rendimiento. También presenta la operacionalización conceptual de variables y las fuentes e instrumentos para la recolección de datos.

3.3.1. Unidad de análisis

- Requisitos funcionales del aplicativo web denominado Renta Car
- Estudiantes de octavo semestre cursando la materia de DevOps

3.3.2. Indicadores del comportamiento temporal

Para la evaluación del comportamiento temporal se describen los dos indicadores seleccionados por el autor del presente trabajo detallados a la **Tabla 3-2.**

Tabla 3-2: Operacionalización conceptual del comportamiento temporal

Variable	Indicador	Descripción
Comportamiento temporal	Tiempo de respuesta	Es el tiempo que le lleva al sistema reaccionar a una entrada.
	Tiempo de espera	El intervalo total transcurrido desde que se envía una instrucción al software, dando inicio a una tarea o proceso interno, hasta el instante en que dicha operación concluye completamente

Realizado por: Cuvi J., 2024

Tabla 3-3: Indicador de Tiempo de respuesta

Característica	Eficiencia de desempeño
Subcaracterística	Comportamiento en el tiempo
Indicador	Tiempo de respuesta
Propósito	¿Cuál es el tiempo estimado en reaccionar a una interacción?
Fórmula	X = B - A A= Tiempo de envío de solicitud B = Tiempo en recibir la primera respuesta
Medida	Tiempo
Valor deseado	X<=0 El más cercano a 0 es el mejor

Fuente: (Salazar Fierro et al, 2020)

Realizado por: Cuvi J., 2023

Tabla 3-4: Indicador de Tiempo de espera

Característica	Eficiencia de desempeño
Subcaracterística	Comportamiento en el tiempo
Indicador	Tiempo de espera
Propósito	¿Cuál es el tiempo desde que se envía una instrucción, para que inicie un trabajo, hasta que lo completa?
Fórmula	X = B - A A= Tiempo cuando se inicia un trabajo B = Tiempo en completar un trabajo
Medida	Tiempo
Valor deseado	X<=0 El más cercano a 0 es el mejor

Fuente: (Salazar Fierro et al, 2020)

Realizado por: Cuvi J., 2024

3.3.3. Operacionalización conceptual y metodológica de variables

Tabla 3-5: Operacionalización conceptual de variables

Formulación del problema	Variable	Indicador	Técnica	Fuente
¿Cuál es el porcentaje de reducción de tiempos de desarrollo de pruebas unitarias al utilizar el sitio web que ofrece generación automatizada de pruebas unitarias	Comportamiento temporal	Tiempo que tarde en reaccionar a una interacción	Observación Cuantitativo	Sitio web Software PageSpeed Insights
		Tiempo que tarde en generar casos de pruebas	Observación Experimental Cuantitativo	Repuestas de la API Sitio web

Formulación del problema	Variable	Indicador	Técnica	Fuente
mediante la API de Open AI?				

Realizado por: Cuvi J., 2024

3.3.4. Secuencias de pasos para medir el comportamiento temporal

En este apartado se detallan la secuencia de pasos para medir el comportamiento temporal del sitio web mediante donde como primer paso se realiza la medición del tiempo respuesta usando el software PageSpeed Insights, como segundo paso se realiza la medición del tiempo de espera usando el ambiente de pruebas propuestas en este trabajo documentando los resultados en la ficha propuesta y como último paso ponderar la valoración de ambos indicadores

3.3.4.1. PageSpeed Insights

PageSpeed Insights es una herramienta gratuita de Google que analiza la velocidad de carga de una página web en función de una serie de factores, incluidos el tiempo de carga del contenido principal, el tamaño de los archivos, la complejidad del código y la configuración del servidor web. La herramienta realiza pruebas simultáneas en móviles y ordenadores, así como en redes 3G y 4G. En total, PageSpeed Insights realiza 24 pruebas para determinar la velocidad de carga de una página web. Los resultados de estas pruebas se combinan para generar una puntuación de rendimiento para la página web, que se basa en una escala de 100, siendo 100 la puntuación más alta. Una puntuación alta indica que la página web se carga rápidamente y que es eficiente en cuanto al uso de los recursos. PageSpeed Insights también proporciona sugerencias sobre cómo mejorar el rendimiento de una página web (Tekla, 2022).

3.3.4.2. Ficha técnica para el comportamiento temporal

La ficha técnica desempeña un papel crucial al facilitar la recopilación de datos destinados a evaluar el comportamiento temporal. En particular, para obtener información detallada sobre la métrica de tiempo de espera asociada al comportamiento temporal, se propone la utilización de la ficha detallada que se describe en la **Tabla 3-6.**

Tabla 3-6: Ficha del tiempo de espera

Titulo			
Autor		**Fecha:**	

Id caso		**Nombre de la Prueba**	
Descripción del caso			
Escenario		**Resultado esperado**	
Lenguaje		**Herramienta**	
Resultado Obtenido		**Tiempo de desarrollo**	

Realizado por: Cuvi J., 2024

3.3.4.3. Ponderación del comportamiento temporal

Para evaluar el comportamiento temporal se ha planteado una ponderación a cada uno de los indicadores contemplados por el autor del presente trabajo mismos que se detallan en la **Tabla 3-7.**

Tabla 3-7: Ponderación de los indicadores del comportamiento temporal

Variable	Indicador	Ponderación	Total
Comportamiento temporal	Tiempo de respuesta	50%	100%
	Tiempo de espera	50%	

Realizado por: Cuvi J., 2024

3.3.5. Fuentes e instrumentos

3.3.5.1. Fuentes

Como fuentes de recolección de información se plantea utilizar:

- **Documentación de la API de Open AI:** Servirá para entender las capacidades, funcionalidades, parámetros de entrada/salida y demás características necesarias para interactuar con la API dentro del sistema.
- **Videotutoriales:** Servirán como guías prácticas para aprender el uso de las tecnologías involucradas en el desarrollo del proyecto.
- **Blogs y canales de desarrolladores:** aportan conocimiento acumulado por la comunidad de desarrolladores.
- **Artículos académicos y científicos:** brindan una perspectiva más amplia y actualizada del tema, así como soporte para la definición del enfoque y medios de validación del proyecto

- **Fuentes de Internet**: Sitios web y repositorios con información técnica, documentación, tutoriales, artículos, entre otros recursos relacionados con las tecnologías y temas abordados en el proyecto.
- **Trabajos relacionados:** Estos aportarán información sobre experiencias previas y buenas prácticas abordadas en proyectos similares.
- **Internet**: Las búsquedas en internet permitirán encontrar documentación, foros, blogs y otros recursos que aporten más datos para el desarrollo del proyecto.
- **OPEN AI API**: Al estar en interacción constante con esta herramienta durante el desarrollo, proporcionará información de primera mano que permitirá recolectar datos lo cual será invaluable para cumplir con los objetivos del presente trabajo.
- **Sitio web**: El propio sitio web desarrollado será una fuente primaria de información, ya que a través de él se recolectarán datos sobre el comportamiento y performance mediante técnicas como registros de ejecución, tiempos de respuesta entre otros.
- **PageSpeed Insights:** una herramienta de Google que analiza el rendimiento y la optimización de un sitio web.
- **Norma ISO/IEC 25010:** Esta norma internacional brindará los lineamientos para realizar la medición y evaluación de la calidad del software desarrollado.

3.3.5.2. Instrumentos

- **Ficha estructurada**: Para registrar los resultados obtenidos usando el sitio web y la realización de pruebas manualmente.
- **Logs y registros de ejecuciones**: Para analizar performance del sitio y casos generados.
- **Cronómetro Digital**: Permitirá saber el tiempo que toma en desarrollar los casos de pruebas de manera manual.

3.3.5.3. Ambiente de pruebas

- RentaCar

El proyecto RentaCar es un aplicativo web de alquiler de autos mismos que contienen reglas de negocios, lógica de negocio que serán utilizados como ambiente de pruebas para evaluar el comportamiento temporal del sitio web en la generación automatizada de casos de pruebas unitarias. Esto permitirá probar la solución en un contexto realista y relevante, utilizando las funciones del proyecto RentaCar con complejidad suficiente para poner a prueba las capacidades del sitio web.

Para evaluar el comportamiento temporal utilizando nuestro entorno de pruebas, se llevará a cabo un proceso de realización de pruebas con dos escenarios distintos mismos que se detallan a continuación:

- **Escenario A:** Generación de pruebas con enfoques tradicionales

En este escenario, se llevará a cabo un proceso convencional de generación de pruebas, donde los participantes realizarán pruebas manuales utilizando métodos tradicionales. Se proporcionarán a los participantes 3 funciones de las reglas de negocios y la lógica del proyecto RentaCar donde deberán desarrollar pruebas unitarias manualmente para cada función proporcionada registrando los tiempos dedicados al desarrollo de las pruebas.

- **Escenario B:** Generación de pruebas con el sitio web

En este escenario, se utilizará el sitio web diseñado para la generación automatizada de casos de pruebas unitarias. Ejecutarán las mismas 3 pruebas correspondientes a las tres funciones de las reglas de negocios y la lógica del proyecto RentaCar, utilizando el sitio web. Durante este proceso, registrarán el tiempo que el sitio web requiere para generar las pruebas unitarias de manera automatizada.

En el **ANEXO G** se muestra un ejemplo de los procesos que deberán realizar los participantes para desarrollar pruebas unitarias en ambos escenarios propuestos.

3.3.5.4. Alcance y requisitos del Proyecto RentaCar

El alcance y los requisitos del proyecto RentaCar definen el conjunto de funcionalidades, lógica de negocio y reglas empresariales que serán objeto para la realización de pruebas unitarias. A continuación, se detalla el alcance y los requisitos de proyecto RentaCar:

- *Alcance del Proyecto*

El aplicativo web RentaCar cuenta con las siguientes características de alcance:

1. Permite la gestión de información de clientes, vehículos, rentas y pagos.
2. Cuenta con un módulo de búsqueda y reservación de vehículos disponibles por parte de los clientes.
3. Posee funcionalidades de generación y seguimiento del alquiler.
4. Genera reportes estadísticos de los ingresos obtenidos, clientes, autos y pasarela de pagos.
5. El aplicativo está optimizado para ejecutarse en navegadores web modernos.
6. El aplicativo esta optimizado para ser usado en dispositivos móviles y ordenadores.

- *Requisitos del Proyecto*

Los requisitos del sistema especifican las condiciones, reglas, lógica de negocio y características necesarias para el correcto funcionamiento del aplicativo RentaCar detalladas a continuación:

1. Registro y autenticación de usuarios (clientes y administradores).
2. Leer, eliminar, insertar y editar información de vehículos, clientes y pagos.

3. Búsqueda y filtrado de vehículos por características.
4. Reservación de vehículos por fechas determinadas.
5. Pasarela de pagos.
6. Devolución de pagos.
7. Devolución de vehículos y cierre de contratos.
8. Generación de reportes y dashboard de información.

3.4. Población y muestra

Para el desarrollo del presente trabajo de integración curricular, se determinó utilizar un muestreo no probabilístico por conveniencia, seleccionando como población objetivo a estudiantes de octavo semestre de la carrera de Ingeniería de Software de la Escuela Superior Politécnica de Chimborazo (ESPOCH). Esta elección muestral responde a la accesibilidad y cercanía de los participantes para la investigación, así como al interés de evaluar la propuesta curricular en un grupo de futuros profesionales del desarrollo de software, validando así su relevancia y aplicabilidad.

La muestra conformada por 31 estudiantes pertenecientes al octavo semestre de la carrera en el período académico vigente durante el desarrollo del presente trabajo. Esta cantidad de participantes permite realizar un análisis del impacto de la variable independiente sobre las mediciones repetidas del grupo, tanto antes como después de la implementación.

3.5. Planteamiento de la hipótesis para la evaluar el comportamiento temporal

Para el presente trabajo se manejan dos hipótesis relacionadas a la eficiencia específicamente en el comportamiento temporal detalladas a continuación:

- **Hipótesis alterna:** Existe una diferencia significativa entre el tiempo que toma el desarrollo de pruebas unitarias haciendo uso del sitio web respecto al tiempo que toma el desarrollo de pruebas unitarias de manera manual.
- **Hipótesis nula:** No existe una diferencia significativa entre el tiempo que toma desarrollar pruebas unitarias con el sitio web respecto al tiempo que toma desarrollar pruebas unitarias manera manual.

3.6. Desarrollo del proyecto mediante la aplicación de Scrumban

En este apartado se presenta el desarrollo del sitio web con la aplicación de la metodología Scrumban misma que cuenta con las siguientes fases: Análisis preliminar, planificación, desarrollo, pruebas, despliegue y cierre.

3.6.1. Análisis preliminar

Para desarrollar este trabajo, es esencial realizar un estudio de factibilidad, identificación de riesgos, requisitos funcionales y no funcionales detallados en los siguientes puntos.

3.6.1.1. Personas involucradas

Al tratarse de un trabajo de propósito general únicamente se cuenta con una persona involucrada con el desarrollo de este siendo:

- **Desarrollador**

  John Cuvi

3.6.1.2. Requisitos de software

A continuación, se describen los requisitos del sistema los cuales representan sus funcionalidades.

- Requisitos funcionales

En la **Tabla 3-8** se presenta los requisitos funcionales del sitio web:

Tabla 3-8: Requisitos funcionales

Identificador	Requisito
RF01	Iniciar sesión
RF02	Crear cuenta
RF03	Configurar código.
RF04	Generar Pruebas
RF05	Guardar las respuestas.
RF06	Visualizar el historial
RF07	Visualizar perfil
RF08	Eliminar cuenta
RF09	Recuperar contraseña
RF10	Cambiar contraseña
RF11	Seleccionar un historial
RF12	Eliminar un historial

Realizado por: Cuvi J., 2024

- Requisitos no funcionales

En la **Tabla 3-9** se presenta el requisito no funcional del sitio web:

Tabla 3-9: Requisito no funcional

Identificador	Requisito
RNF01	Comportamiento temporal

Realizado por: Cuvi J., 2024

3.6.1.3. Diagramas de casos de uso

En esta sección se presenta los diagramas de uso de acuerdo con su rol representando las acciones que esté puede realizar en el sitio web.

- Usuario no registrado

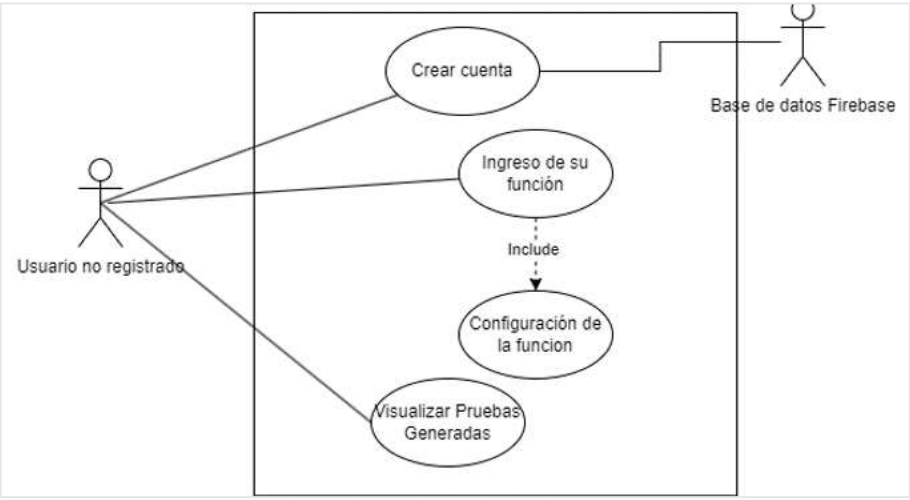


Ilustración 3-1: Caso de uso de usuario no registrado

Realizado por: Cuvi J., 2023

- Usuario registrado

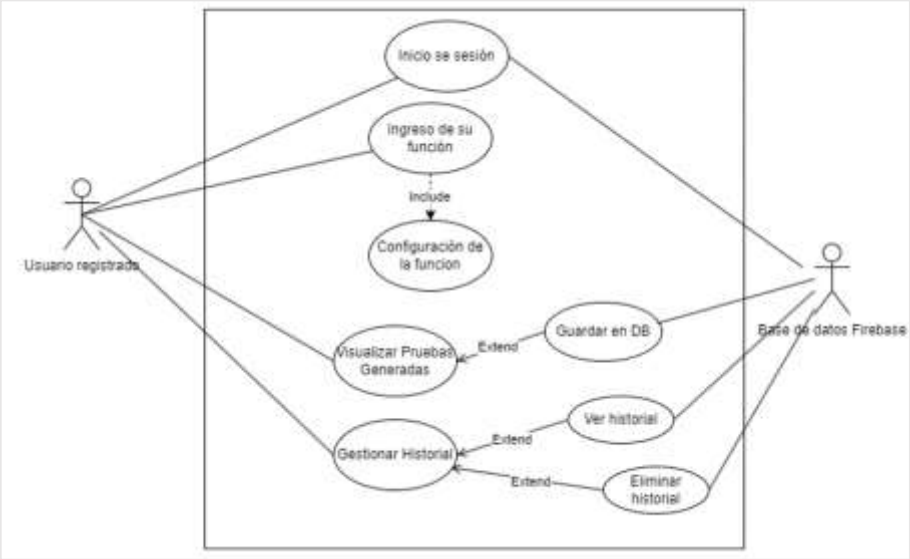


Ilustración 3-2: Caso de uso de usuario registrado

Realizado por: Cuvi J., 2023

3.6.1.4. Estudio de factibilidad

Para el desarrollo del presente trabajo de integración curricular primeramente se llevó a cabo un análisis de viabilidad para evaluar la idoneidad del presente trabajo. Adicionalmente, se

definieron estrategias y se planificaron los recursos humanos, materiales y financieros requeridos para su implementación descritas en los siguientes apartados.

- **Factibilidad técnica**

Como primer punto de partida se realizó un análisis de viabilidad técnica para evaluar si la tecnología disponible permite la implementación del trabajo. Esta evaluación permite identificar los recursos de hardware, software, infraestructura y conocimientos técnicos necesarios para desarrollar el presente trabajo de integración curricular.

Se realizó un estudio de factibilidad técnica, cuyos detalles se encuentran en el **ANEXO A**, con base en los requerimientos del proyecto de integración curricular. Los resultados del estudio indican que es viable llevar a cabo dicho proyecto, dado que se cuenta con todos los elementos técnicos necesarios

- **Factibilidad económica**

Para conocer si el presente del trabajo cuenta con factibilidad económica necesaria se llevó a cabo un análisis económico para evaluar si se cuenta con los recursos económicos necesarios que permitan llevar a cabo la ejecución de cada proceso que se tiene para la ejecución del presente proyecto. Durante esta evaluación, se recopiló información detallada las cuales se descritas en el **ANEXO B.**

Como resultado de dicha evaluación se concluye que, si es viable la realización de trabajo de integración curricular, además el presente trabajo tiene un costo de 1259 dólares americanos mismos que serán financiados por el autor de este presente trabajo de integración curricular.

- **Factibilidad Operativa**

La factibilidad operativa evalúa si un proyecto o plan de acción puede ser implementado de manera efectiva dentro de la organización, considerando los recursos humanos disponibles. Conforme a lo expuesto, se presenta en la **Tabla 3-10** un análisis que determina los recursos humanos necesarios para llevar a cabo la realización del presente trabajo teniendo en cuenta que el sitio web está dirigido a aquellas personas que desarrollan y prueban productos software.

Tabla 3-10: Recursos humanos para la operación del sitio web

Rol	Conocimientos Requeridos	Función
Programadores	Pruebas de software Pruebas unitarias Rest API Programación Intermedia	Usar el sitio web generando pruebas unitarias automatizadas para validar el correcto funcionamiento de las diferentes
Testers		

Rol	Conocimientos Requeridos	Función
	Herramientas de Pruebas	funcionalidades y flujos de sus productos software

Realizado por: Cuvi J., 2023

3.6.1.5. Estimación del proyecto

En esta sección se realizaron los siguientes cálculos utilizando el modelo COCOMO obtenido los siguientes resultados:

- Hombres-mes: aproximadamente 6,9858.
- Tiempo de desarrollo estimado: 5,23 meses.
- Número de programadores necesarios: 1,326.

Para obtener información más detallada sobre estos cálculos y su metodología, se puede consultar el **ANEXO C** adjunto, donde se proporciona una explicación más detallada de la sección de estimación del trabajo de integración curricular.

3.6.1.6. Análisis de riesgos

Se evaluaron los riesgos potenciales que podrían surgir durante su desarrollo, previo al desarrollo de este trabajo de integración curricular. Durante este análisis, se identificaron un total de siete riesgos específicos, los cuales se describen a continuación:

- **Identificación de riesgos**

En la **Tabla 3-11** se describen los riesgos potenciales que pueden afectar al desarrollo del presente trabajo de integración curricular.

Tabla 3-11: Identificación de riesgos

Requisito	Descripción	Categoría	Consecuencias
R01	Robo del equipo de cómputo	Técnico	Pérdida de tiempo y costo adicional del proyecto
R02	Daño del equipo de cómputo	Técnico	Pérdida de tiempo y costo adicional del proyecto.
R03	Diseño erróneo de base de datos.	Técnico	Pérdida de tiempo, mal manejo de datos.
R04	Cambio de políticas de la OPEN AI API	Proyecto	Posible abandono del proyecto.
R05	Cambio de autoridades	Negocio	Se suspendería la realización del proyecto.

Requisito	Descripción	Categoría	Consecuencias
R06	Mal rendimiento del sitio web	Proyecto	Dificultad de obtención de pruebas unitarias generadas.

Realizado por: Cuvi J., 2023

- **Determinación de la probabilidad**

Es importante definir la gama de posibilidades de que ocurran los riesgos, así como establecer una escala de evaluación que va del uno al tres. En la **Tabla 3-12**, se detallan las descripciones de esta escala conforme a sus probabilidades junto a su valoración.

Tabla 3-12: Determinación de la probabilidad

Rango de Probabilidad %	Descripción	Valor
1 - 33	Baja	1
34 - 66	Media	2
67 - 99	Alta	3

Fuente: https://www.ealde.es/como-elaborar-matriz-de-riesgos/

Realizado por: Cuvi J., 2023

- **Determinación del impacto**

Para determinar el impacto de cada riesgo identificado se clasifica según su nivel de impacto en el desarrollo del proyecto. Se asigna un valor que refleja el grado de impacto que podría tener cada riesgo y la afección del proyecto en reflejado en días, siguiendo las pautas detalladas en la **Tabla 3-13**.

Tabla 3-13: Nivel de impacto

Impacto	Impacto técnico	Valor	Retraso en días
Bajo	El retraso es mínimo	1	3-5 días
Moderado	El retraso es considerable	2	6 – 14 días
Alto	El retraso es grande	3	15 - 30 días
Critico	No se puede continuar con el proyecto.	4	31 – 50 días

Realizado por: Cuvi J., 2023

- **Exposición de riesgos**

La exposición al riesgo se ha clasificado en tres niveles: bajo, medio y alto, detallado en la **Tabla 3-14**. Esta clasificación permite distinguir el grado de riesgo al que está expuesto el trabajo y facilita la toma de decisiones para su gestión adecuada.

Tabla 3-14: Valoración de exposición de riesgos

Exposición al riesgo	Valor
Baja	1 o 2
Media	3 o 4
Alta	Mayor igual a 6

Fuente: https://www.ealde.es/como-elaborar-matriz-de-riesgos/

Realizado por: Cuvi J., 2023

- **Resultados del análisis de riesgos**

La **Tabla 3-15** resume el análisis de riesgos realizado a el presente trabajo. Se muestra los apartados detallados anteriormente junto a sus respectivas valoraciones.

Tabla 3-15: Resultados del análisis de riesgos

Id del Riesgo	Probabilidad		Impacto		Exposición al riesgo	
	%	Valoración	Valoración	Impacto	Valoración	Exposición
R01	40%	2	3	Alto	6	Alta
R02	30%	1	1	Bajo	1	Baja
R03	40%	2	3	Alto	6	Alta
R04	60%	3	4	Crítico	12	Alta
R05	20%	1	1	Bajo	1	Baja
R06	40%	2	2	Moderado	4	Media

Realizado por: Cuvi J., 2023

Con base en los resultados obtenidos en la **Tabla 3-16** sobre el análisis de riesgos realizado, se presenta la valoración respectiva por cada riesgo ordenada ascendentemente junto a su prioridad.

Tabla 3-16: Determinación de exposición de riesgos – Determinación de la prioridad

Identificación del Riesgo	Exposición de riesgo	Valor	Prioridad
R04	Bajo	12	1
R01	Alto	6	2
R03	Alto	6	2
R06	Medio	4	3
R02	Baja	1	4
R05	Bajo	1	4

Realizado por: Cuvi J., 2023

Se ha elaborado un plan destinado a reducir, vigilar y administrar cada riesgo detallado en el **ANEXO D**. En este documento se especifican las razones, implicaciones y medidas de mitigación

asociadas a cada riesgo. Cada riesgo cuenta con una hoja individual en el **ANEXO D**, muestra un plan de acción adaptado para gestionar de manera efectiva todos los riesgos identificados.

3.6.2. Planificación

La planificación es la etapa inicial que precede al desarrollo del sitio web en el presente trabajo. En esta sección, se detallan todas las actividades y funcionalidades que aún no han sido programadas para una iteración específica.

3.6.2.1. Tareas por hacer

Esta sección está dedicada a albergar todas las actividades y funcionalidades que aún no han sido planificadas para una iteración específica representadas en la **Tabla 3-17:**

Tabla 3-17: Tareas por hacer

ID	TAREAS POR HACER
T1	En mi competencia como desarrollador, necesito establecer la arquitectura de la aplicación de Frontend y Backend.
T2	En mi competencia como desarrollador, necesito diseñar la base de datos.
T3	En mi competencia como desarrollador, necesito desarrollar el servidor Backend para que responda a las peticiones.
T4	En mi competencia como desarrollador, necesito configurar el servidor Backend para que haga una conexión a la API de OPEN AI enviando el cuerpo de la solicitud receptada.
T5	En mi competencia como desarrollador, necesito desarrollar el esquema el cliente Frontend con base en la arquitectura seleccionada.
T6	En mi competencia como desarrollador, necesito establecer la comunicación entre los dos servidores.
T7	En mi competencia como desarrollador, necesito desarrollar los componentes que formaran parte de la interfaz principal de usuario.
T8	En mi competencia como desarrollador, necesito desarrollar todas las páginas que formaran parte del sitio web.
T9	Se debe permitir al usuario externo visualizar un menú de las páginas propias del sitio web.
T10	Se debe permitir al usuario externo la posibilidad de registrarse y eliminar cuenta.
T11	Se debe permitir al usuario externo la posibilidad de iniciar sesión y cerrar sesión.
T12	Se debe permitir al usuario configurar el código a evaluar.
T13	Se debe permitir al usuario externo visualizar las pruebas unitarias generadas.
T14	Se debe permitir al usuario externo gestionar las pruebas unitarias generadas.

ID	TAREAS POR HACER
T15	Se debe permitir al usuario externo visualizar el historial de pruebas almacenadas.
T16	Se debe permitir al usuario externo poder eliminar un historial en específico.
T17	Se debe permitir al usuario externo tener un manual de uso del sitio web.
T18	En mi competencia como desarrollador, necesito alojar los dos servidores en internet.

Realizado por: Cuvi J., 2023

3.6.2.2. Sprint Backlog

Se ha realizado una planificación inicial identificando las tareas necesarias para desarrollar el sitio web en un mes. En la **Tabla 3-18** se presenta la planificación detallada con las fechas de inicio y fin estimadas para cada tarea. Durante el desarrollo del trabajo, se podrán hacer ajustes a esta planificación inicial para adaptarse a la velocidad u otros factores.

Tabla 3-18: Sprint Backlog

Sprint	ID tarea	Descripción	Fecha inicio	Fecha fin
Sprint 1	T1	Definir la arquitectura frontend y backend	01/11/2023	02/11/2023
	T2	Diseño inicial de interfaz de usuario	03/11/2023	03/11/2023
	T3	Integrar API de OpenAI en backend	04/11/2023	07/11/2023
Sprint 2	T4	Establecer parámetros de integración con API de OpenAI	08/11/2023	10/11/2023
	T5	Desarrollar componentes principales de frontend	11/11/2023	14/11/2023
	T6	Implementar comunicación en tiempo real entre clientes	15/11/2023	17/11/2023
Sprint 3	T7	Construir interfaz principal de la aplicación	18/11/2023	20/11/2023
	T8	Desarrollar secciones secundarias (acerca de, contacto, etc.)	21/11/2023	23/11/2023
	T9	Desplegar versión beta de la aplicación	24/11/2023	27/11/2023
Sprint 4	T10	Desarrollar inicio y cierre de sesión	28/11/2023	29/11/2023
	T11	Desarrollar la gestión del perfil del usuario	30/11/2023	01/12/2023
	T12	Desarrollar el componente de configuración de código.	02/12/2023	03/12/2023

Sprint	ID tarea	Descripción	Fecha inicio	Fecha fin
	T13	Desarrollar componente de visualización de respuestas	03/12/2023	03/12/2023
Sprint 5	T14	Desarrollar módulo para almacenar pruebas generadas	04/12/2023	06/12/2023
	T15	Implementar visualización de historial de pruebas	07/12/2023	08/12/2023
Sprint 6	T16	Completar módulo de gestión de historial	09/12/2023	11/12/2023
	T17	Crear manual de usuario	12/12/2023	13/12/2023
	T18	Desplegar versión final de la aplicación	14/12/2023	15/12/2023

Realizado por: Cuvi J., 2023

3.6.3. Desarrollo

3.6.3.1. Análisis

- **Tablero Kanban**

El tablero Kanban ofrece una visualización efectiva del progreso y estado actual de las tareas durante de los Sprint establecidos en el presente trabajo. Muestra las tareas y elementos de trabajo en tres columnas principales: "Por hacer" que incluye las tareas pendientes que aún no se han iniciado, "En proceso" que muestra las tareas en las que se está trabajando en el momento, y "Tareas Finalizadas" que señala las tareas completadas, con un límite WIP de 1 para “Tareas en proceso” indicando el número máximo de tareas. En la **Ilustración 3 3**, se presenta el tablero Scrumban del Sprint 1, donde cada tarea tiene una fecha límite asignada y el responsable indicado. De esta forma, el tablero Kanban permite gestionar el flujo de trabajo de manera ágil y visual para optimizar la productividad.

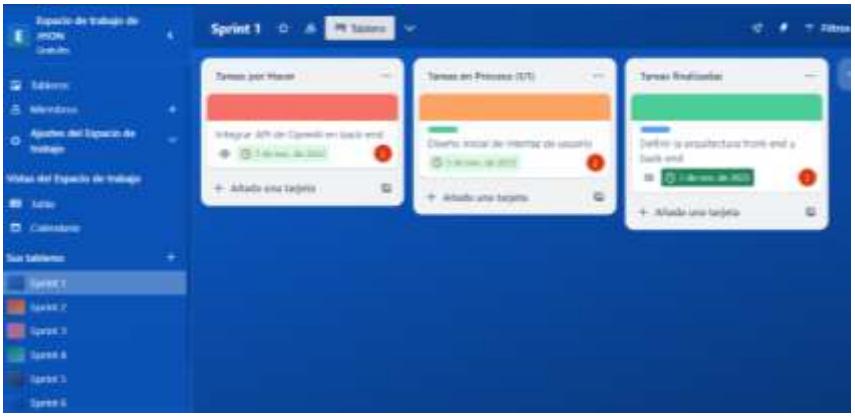


Ilustración 3-3: Tablero Kanban

Realizado por: Cuvi J., 2023

- **Tipos de usuarios**

1. **Usuario registrado:** es aquel que se ha registrado en el sitio web, proporcionando sus datos personales y creando una cuenta de usuario. Este usuario registrado tiene acceso completo a todas las funcionalidades del sitio web, incluyendo aquellas que requieren autenticación.
2. **Usuario no registrado**: es el usuario que ingresa al sitio web sin haberse registrado ni creado una cuenta de usuario. Este tipo de usuario tiene un acceso limitado, pudiendo navegar por las secciones públicas del sitio y utilizar algunas funciones básicas como búsqueda y vista de contenidos.

3.6.3.2. Diseño de base de datos

- Convención de codificación

Para el nombrado de colecciones, documentos y campos en la base de datos NoSQL se define el estándar snake_case, el cual facilita la lectura al separar palabras y mantiene compatibilidad con lenguajes de programación; esto establece una convención consistente que permite una rápida comprensión de los elementos sin documentación adicional.

Ejemplo:

- codigo_ingresado
- codigo_respuesta

- Diseño

Con el fin de proporcionar al usuario una autenticación rápida y segura, se plantea la incorporación de Google Auth de Firebase utilizando su base de datos Firestore para almacenar y sincronizar los datos de los usuarios generados durante las interacciones con el sitio web.

Para modelar los datos del sitio se siguió un proceso de diseño conceptual de las entidades y sus relaciones. Luego se trasladó esto a un modelo de datos NoSQL orientado a documentos, aprovechando las capacidades de Firestore utilizando diagramas UML. Este modelado NoSQL evita la necesidad de normalización, permitiendo representar las entidades requeridas de manera simple y rápida tal y como se muestra en la **Ilustración 3-4**.

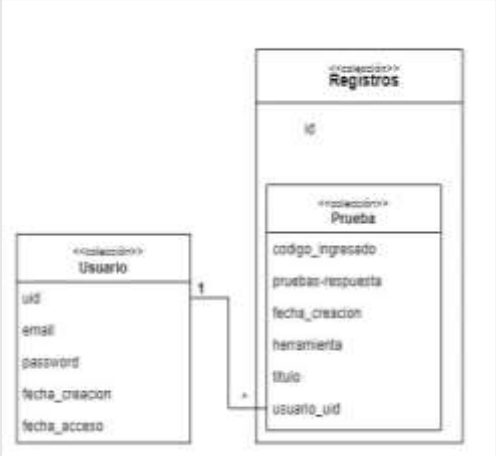


Ilustración 3-4: Modelo de colecciones y documentos de Firestore

Realizado por: Cuvi J., 2023

- Diccionario de datos

El compendio de datos proporciona explicaciones y detalladas descripciones sobre las colecciones, documentos y campos presentes en el modelo NoSQL, facilitando una comprensión precisa de la significancia y aplicación de cada componente. Además, establece pautas, como la convención de nomenclatura snake_case, para uniformizar la identificación de los elementos.

Tabla 3-19: Diccionario de datos

Colección	Nombre del campo	Descripción	Tipo	Null permitidos
Usuario	email	Correo electrónico del usuario utilizado.	Text (25)	No
	password	Clave de acceso a su cuenta.	Text (25)	No
	fecha_creacion	Fecha y hora en la que se creó el registro del usuario en la base de datos.	timestamp	No
	fecha_acceso	Fecha y hora en la que el usuario hizo su último ingreso al sistema.	timestamp	No
Pruebas	uid	Identificador único asignado automáticamente por FireBase Auth	Text (20)	No
	codigo_ingresado	Código de pruebas ingresada por los usuarios.	Text (10000)	No
	codigo_respuesta	Código de pruebas generas por parte del sitio web	Text (10000)	No
	fecha_creacion	Fecha y hora de creación de las pruebas.	timestamp	No

	herramienta	Herramienta de testing para la cual se generó el código de pruebas.	Text (100)	No
	lenguaje	Lenguaje de programación al que pertenece el código ingresado.	Text (50)	No
	titulo	El título de la prueba ingresada por parte del usuario	Text (1000)	No
	usuario	Identificador del usuario que se relaciona con la entidad del usuario	Text (20)	No
Registro	id	Identificador de la colección generado automáticamente.	Text (20)	No

Realizado por: Cuvi J., 2023

3.6.3.3. Arquitectura del sitio web

El sitio web emplea una arquitectura Cliente-Servidor para distribuir las responsabilidades entre el frontend y el backend. El frontend o parte cliente es la interfaz de usuario que interactúa directamente con el consumidor final de la aplicación. El backend o parte servidor recibe las solicitudes HTTP del cliente, las procesa, devuelve las respuestas correspondientes.

Tal como se muestra en la **Ilustración 3-5**, el dispositivo del usuario realiza peticiones al servidor web a través de Internet utilizando el protocolo HTTP o HTTPS. El servidor evalúa la lógica conforme a la solicitud y devuelve la información solicitada al cliente.

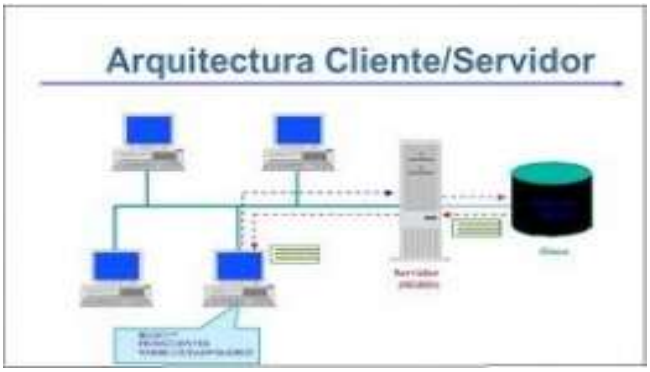


Ilustración 3-5: Arquitectura Cliente – Servidor

Fuente: https://www.ecured.cu/Cliente-Servidor

También emplea la arquitectura basada en componentes para el diseño del sitio web utilizando React JS, permitiendo crear interfaces de usuario interactivas y dinámicas mediante la composición y el intercambio de componentes independientes y reutilizables tal y como se muestra en la **Ilustración 3-6**.

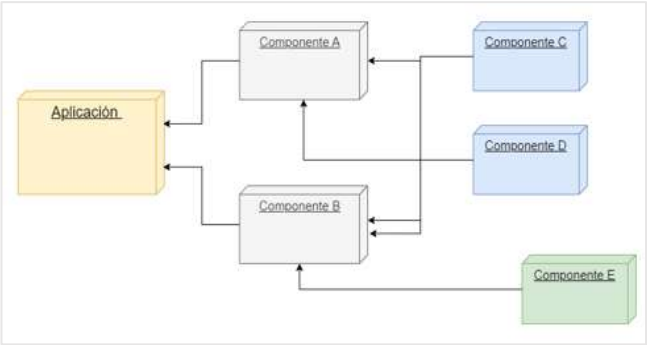


Ilustración 3-6: Arquitectura basa en componentes

Realizado por: Cuvi J., 2024

3.6.3.4. Modelo de gpt-3.5 turbo

Para el desarrollo de este trabajo de integración curricular, se seleccionó el modelo de lenguaje GPT-3.5-turbo de la serie GPT-3 por ser la versión más avanzada actualmente disponible. GPT-3.5-turbo fue entrenado con más datos y parámetros que sus predecesores, lo que resulta en mejoras sustanciales en la calidad y coherencia de texto generado, según lo reportado por (OpenAI, 2023). Además, GPT-3.5-turbo demuestra una comprensión contextual superior y una mayor eficiencia en tareas como la generación de código, en comparación con versiones anteriores.

La **Tabla 3-20** resume las características principales del modelo de lenguaje GPT-3.5-turbo, incluyendo detalles como la serie a la que pertenece, número de parámetros, tokens, costos asociados y fecha de última actualización.

Tabla 3-20: Detalles del modelo gpt-3.5-turbo

Serie	GPT-3
Modelo	GPT-3.5-turbo
Parámetros	170 mil millones
Nivel de precisión	93%
Costo de entrada	0.012/1k token $
Costo de salida	0.012/1k token $
Ultima actualización	22 de agosto de 2023
Datos de entrenamiento	Hasta septiembre de 2021
Arquitectura	Transformer

Realizado por: Cuvi J., 2023

3.6.3.5. Diseño de la interfaz de usuario

La interfaz de usuario fue diseñada con un enfoque UX centrado en la brindar una experiencia optimizada, utilizando React JS y una arquitectura basada en componentes. Previo a su implementación se desarrollaron mockups iniciales definiendo la estructura y jerarquía visual.

- Mockups

Se generaron bosquejos con Balsamiq Mockups para esbozar la estructura del contenido y la disposición visual de los componentes principales del sitio web. Esto posibilitó visualizar rápidamente el flujo de las interfaces y la jerarquía antes de su implementación.

En la **Ilustración 3-7** se muestra el mockup de la interfaz principal del sitio web mostrando principalmente los componentes de entrada y salida de pruebas:

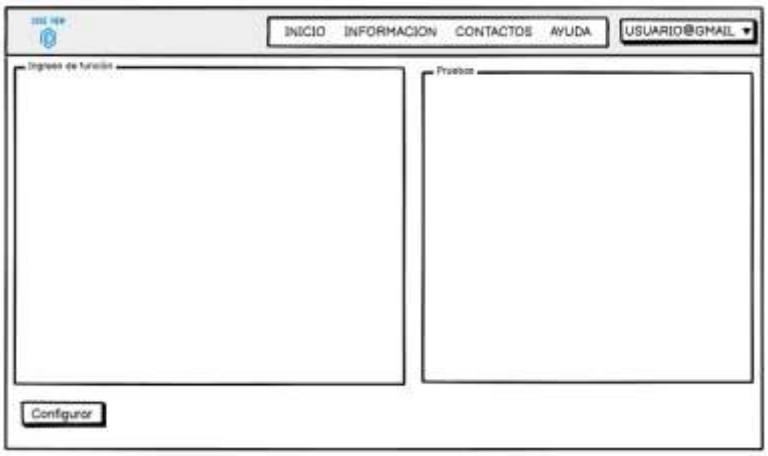


Ilustración 3-7: Mockup de la interfaz principal

Realizado por: Cuvi J., 2023

En la **Ilustración 3-8** se muestra el mockup del componente de configuración de la función:

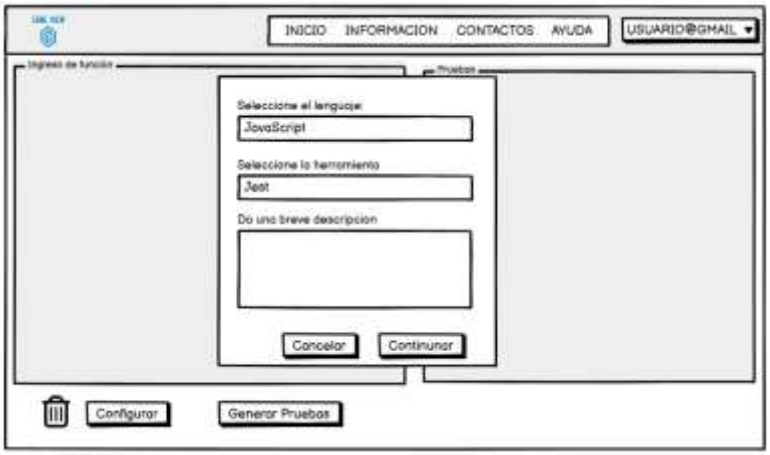


Ilustración 3-8: Mockup del componente de configuración

Realizado por: Cuvi J., 2023

En la **Ilustración 3-9** se muestra el resultado de la vista del historial de los usuarios sobre las pruebas guardadas en la base de datos.

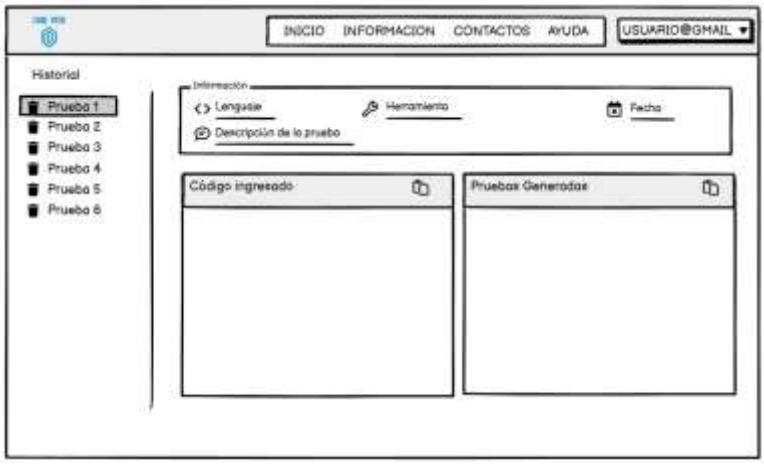


Ilustración 3-9: Mockup de la interfaz del historial

Realizado por: Cuvi J., 2023

- Implementación

Una vez finalizados los mockups en Balsamiq, se procedió con el desarrollo de la interfaz del sitio web utilizando React JS.

A continuación, en la **Ilustración 3-10** se muestra implementación final de la interfaz de inicio con React, siguiendo las especificaciones definidas en los mockups:

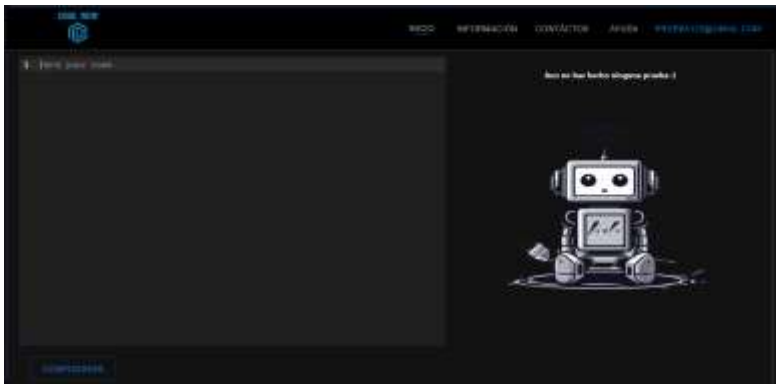


Ilustración 3-10: Página principal

Realizado por: Cuvi J., 2023

Las diferentes interfaces que conforman el sitio web se pueden observar detalladamente en el **ANEXO E.**

3.6.3.6. Codificación

- Estándar de codificación

Para la codificación Frontend con React, se optó por el estándar de codificación StandardJS, eliminando la necesidad de configuraciones adicionales. React proporciona una configuración predeterminada de ESLint para garantizar la coherencia en el estilo del código JavaScript. Esta implementación establece pautas de formato, que incluyen el uso de comillas simples, sangrías

de 2 espacios y líneas finales en blanco, siguiendo las prácticas recomendadas en la industria enfocados en mejorar la legibilidad. A continuación, se proporciona un fragmento de código aplicando el estándar de codificación mencionado:

```
import React, { useState, useEffect, useContext } from 'react'

export const ChatApp = () => {
  const [chats, setChats] = useState([]); // Títulos de los chats
  useEffect(() => {
    obtenerChats();
  }, []);

  return (
      <div className="chat-content">
          <ChatResponse chat={currentChat} />
      </div>
    </div>
```

- Organización del código

Dado que el sitio web sigue una arquitectura cliente-servidor, la codificación se dividió claramente entre el frontend y el backend, ubicándolos en carpetas separadas para distinguir sus responsabilidades como se lo muestra en la **Ilustración 3-11**

Ilustración 3-11: Organización de Carpetas

Realizado por: Cuvi J., 2023

El frontend del sitio web se implementa todas las interfaces gráficas con las que el usuario final navegará e interactuará. Se utilizó React para desarrollar componentes reutilizables, gestionar el enrutamiento entre vistas y manejar los estados tal y como lo muestra **Ilustración 3-12**.

Ilustración 3-12: Organización Frontend

Realizado por: Cuvi J., 2023

El backend, se construyó la API REST que expone los datos y servicios necesarios para alimentar al cliente. Node.js con Express sirvieron como marco de trabajo logrando separar en módulos y funciones específicas las operaciones con la API, el procesamiento de datos, validaciones y otras tareas propias del servidor tal y como lo muestra en la **Ilustración 3-13**.

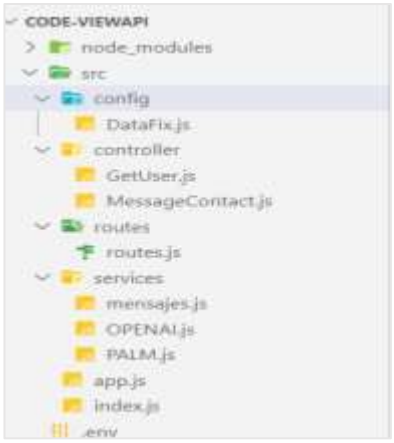


Ilustración 3-13: Organización Backend

Realizado por Cuvi J., 2023

A continuación, en la **Ilustración 3-14** se presenta un diagrama de componentes que describe la arquitectura del sitio web, con un frontend en React, backend en Node y Express, y la integración de proveedores externos como Firebase y OPEN AI.

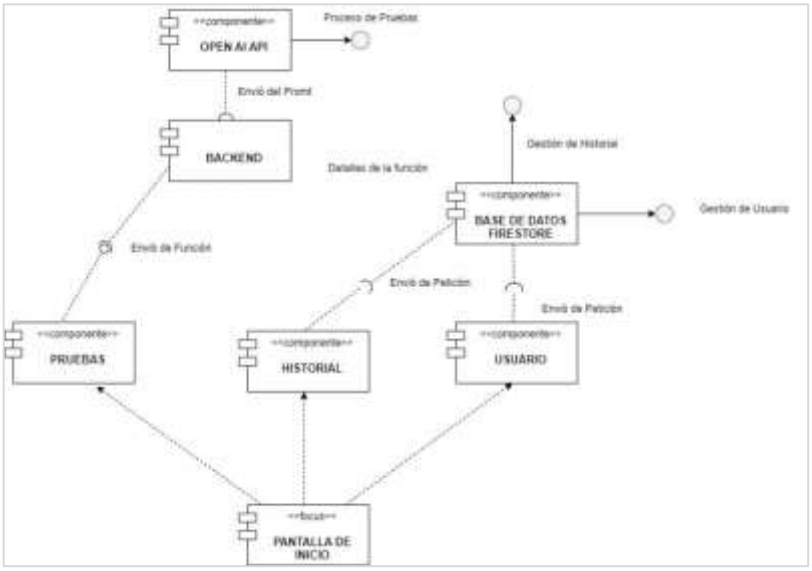


Ilustración 3-14: Diagrama de componentes

Realizado por: Cuvi J., 2024

3.6.4. Pruebas

Se realizaron pruebas de validación en el sitio encargado de la generación automatizada de pruebas unitarias, para asegurar el cumplimiento adecuado de los requerimientos funcionales definidos y su funcionamiento en distintos escenarios. Estas pruebas comprendieron la ejecución de diversos casos empleando diferentes conjuntos de datos de entrada.

A continuación, en la **Tabla 3-21** se muestra un ejemplo a detalle de la realización de una de las pruebas desarrolladas:

Tabla 3-21: Caso de prueba CP0-01

Caso de prueba	CP-01	**RF asociado**	RF01	**Estado**	Aprobado
Descripción	Ingreso de usuario no registrado				
Datos de entrada	Correo no existente Contraseña válida				
Resultado esperado	Mensaje de alerta de error al iniciar sesión				
Resultado obtenido					
Responsable	John Cuvi		**Fecha**		14/11/2023

Realizado por: Cuvi J., 2023

Se desarrollan 22 casos de pruebas y los resultados indican que se satisfacen el correcto funcionamiento de cada requerimiento establecido. Se proporciona un detallado informe de los casos ejecutados en el **ANEXO F**.

3.6.4.1. Proceso para realizar pruebas unitarias

Para validar el funcionamiento del sitio web, se adopta el proceso de pruebas unitarias recomendado por (Yeeply, 2022) enfocado en probar cada función del código de forma aislada, mismos que son detallados a continuación:

- **Identificación de unidades a probar:** Se seleccionan funciones de las reglas de negocio del escenario de pruebas planteado denominado RentaCar. El objetivo es verificar el

comportamiento de aquellas unidades clave relacionadas con lógica de negocio compleja o sensible.

- **Creación de casos de prueba**: Para cada unidad identificada en la fase previa, se diseñan diversos casos de prueba que consideren tanto escenarios válidos como inválidos, según las especificaciones y reglas de negocio establecidas haciendo uso del sitio web desarrollado previamente que permite generar pruebas unitarias automatizadas para cada función. El sitio web permite configurar la función con base en el lenguaje y la herramienta de testing de preferencia lo que permite generar pruebas automatizadas proporcionando los siguientes datos para cada prueba generada:
  - La descripción de la prueba
  - Datos de entrada
  - Resultado esperado
  - Código de la prueba
- **Ejecución y verificación:** Las pruebas unitarias generadas por el sitio web se ejecutan sobre el escenario de pruebas RentaCar. Los resultados obtenidos en cada prueba ejecutada se comparan directamente con los resultados esperados previamente en el sitio web. De esta forma se valida que el comportamiento real de cada unidad coincida con el comportamiento definido en las especificaciones y reglas de negocio del escenario RentaCar con una variedad de pruebas generadas por el sitio web.
- **Documentación de resultados**: Los resultados obtenidos en cada prueba unitaria ejecutada se documentan en una ficha técnica planteada. En esta se detalla el caso de prueba, los datos de entrada, el resultado esperado, resultado obtenido y el tiempo que le tomo desarrollar la prueba unitaria. De esta manera se deja constancia del proceso completo, facilitando la detección de posibles inconsistencias y sirviendo como evidencia del correcto funcionamiento de las unidades antes de la mediante las pruebas generadas por el sitio web.

3.6.5. Despliegue

El despliegue del sitio web se realizó por separado para el frontend y el backend, en servicios especializados en alojamiento web. Esta estrategia ofrece mayor flexibilidad y escalabilidad.

- Despliegue del frontend

El frontend implementado en React se desplegó sobre Netlify, plataforma optimizada para aplicaciones web modernas. Netlify gestiona de forma automatizada tareas como construcción, publicación de actualizaciones y provisionamiento.

- Despliegue del backend

Por otro lado, el backend en Node/Express se alojó en Render, proveedor cloud que simplifica el hosting de APIs y backends. Render administra funciones, colas de trabajos, caché, monitoreo y escalado, permitiendo enfocar los esfuerzos en la lógica de negocio.

- Diagrama de despliegue

El frontend en React desplegado en Netlify interactúa tanto con Firebase como con el backend en Node/Express hospedado en Render. Por su parte, el backend se integra adicionalmente con la API de Open AI para incorporar las capacidades de procesamiento de lenguaje. Esta estructura desacoplada donde cada componente se hospeda de forma independiente permitiendo escalarlos y administrarlos por separado. Como se aprecia en la **Ilustración 3-15** el frontend gestiona la interfaz de usuario y el backend gestiona la lógica y datos de negocio, mientras proveedores SaaS externos se encargan de funcionalidades específicas de IA.

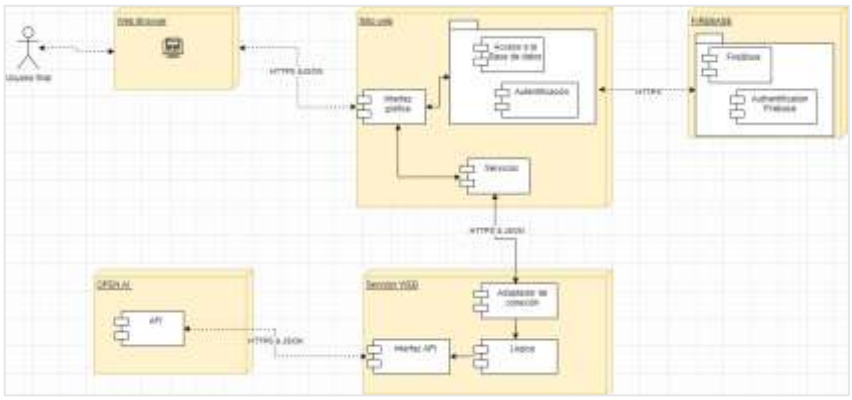


Ilustración 3-15: Diagrama de despliegue

Realizado por: Cuvi J., 2023

3.6.6. Cierre

Tras finalizar la fase de pruebas y el exitoso despliegue de la solución desarrollada, se procede a realizar el cierre formal del del desarrollo siguiendo los lineamientos establecidos en Scrumban. En la **Ilustración 3-16** se observa un tablero marcando que el 100% de las tareas definidas inicialmente se han completado, confirmando que toda la funcionalidad requerida se ha implementado según lo planificado. Habiendo completado todas las etapas previo al cierre, se determinar que se ha desarrollado exitosamente la solución.

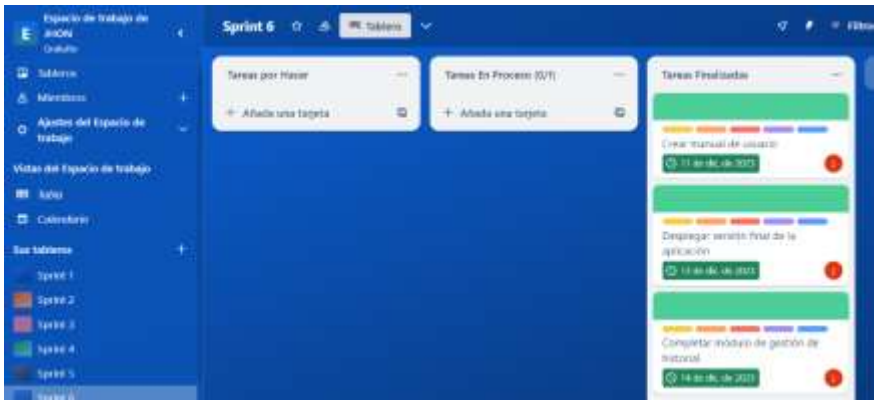


Ilustración 3-16: Tablero Trello completada

Realizado por: Cuvi J., 2023

3.6.6.1. Gestión de riesgos

Durante el desarrollo del sitio web el riesgo identificado como R02 se convirtió en un problema cuando la batería de la laptop no fue reconocida en algunas instancias. Esto resultó en interrupciones en las tareas planificadas.

A pesar de este contratiempo, la gestión del riesgo R02 demostró su eficacia al abordar de manera exitosa el problema durante la fase de reducción implementando acciones correctivas resultando en una resolución efectiva del problema, asegurando así el desarrollo continuo del proyecto según lo planificado.

CAPÍTULO IV

4. ANÁLISIS E INTERPRETACIÓN DE RESULTADOS

En este capítulo se presenta el análisis de los resultados obtenidos al evaluar el comportamiento temporal del sitio web desarrollado en este presente trabajo. Se describen los métodos utilizados para su evaluación detallando los resultados de la evaluación realizada.

4.1. Medición del comportamiento temporal

En este estudio, se adoptó un enfoque de muestreo por conveniencia para lo cual se trabajó con 31 estudiantes de la carrera de software pertenecientes al 8vo semestre del periodo pertinente para analizar el comportamiento temporal en la generación automatizada de pruebas unitarias a través de la solución desarrollada Se llevaron a cabo pruebas manuales a tres funciones específicas del escenario de pruebas denominado "Renta Car", documentando los resultados obtenidos posteriormente, se empleó el sitio web desarrollado para realizar las mismas pruebas de forma automatizada.

Además, se empleó Google PageSpeed Insight para evaluar los tiempos de carga y el rendimiento del sitio web, ofreciendo así una evaluación más detallada de los tiempos de respuesta y el performance del sitio web.

4.1.1. Niveles de evaluación para el tiempo de respuesta

La **Tabla 4-1** presenta los niveles de evaluación para el tiempo de respuesta, expresados como porcentajes en relación con los tiempos obtenidos. Este enfoque permite cuantificar los requisitos de la aplicación, donde "t" representa el valor obtenido en una escala de tiempo y "x" representa la escala de medición correspondiente para "t".

Tabla 4-1: Niveles de evaluación del indicador de tiempo de respuesta

Tiempo de respuesta (segundos)	Porcentaje de medición	Nivel de satisfacción
0 – 2.99	75% < x <= 100%	Muy satisfactorio
3 – 8.99	55% < x <= 75%	Satisfactorio
9 – 11.99	40% < x <= 55%	Más o menos satisfactorio
t > 12	x <= 40%	Nada satisfactorio

Fuente: (Jurnal, 2023)

Realizado por: Cuvi J., 2024

4.1.2. Niveles de evaluación para el tiempo de espera

La **Tabla 4-2** presenta los indicadores utilizados para evaluar el tiempo de espera, expresados en porcentajes con respecto a los tiempos medidos. Este enfoque permite cuantificar los requisitos de la aplicación, donde "t" representa el valor de tiempo obtenido en una escala temporal, mientras que "x" representa la escala de medición asociada con dicho valor "t".

Tabla 4-2: Niveles de evaluación del indicador de tiempo de espera

Tiempo de espera (segundos)	Porcentaje de medición	Nivel de satisfacción
0 - 29	75% < x <= 100%	Muy satisfactorio
30 - 59	55% < x <= 75%	Satisfactorio
60 - 119	40% < x <= 55%	Más o menos satisfactorio
t > 120	x <= 40%	Nada satisfactorio

Fuente: (Branaghan y Sanchez, 2009)

Realizado por: Cuvi J., 2024

4.1.3. Niveles de puntuación para medir el comportamiento temporal

La **Tabla 4-3** se muestran los niveles para medir el comportamiento general de la solución desarrollada donde se muestra mediante una escala de medición porcentual.

Tabla 4-3: Escala de medición del comportamiento temporal

Escala porcentual	Valor Cuantitativo
91% - 100%	Excelente
76% - 90%	Muy buena
51% - 75%	Bueno
21% - 50%	Aceptable
11% - 20%	Regular
0% - 10%	Malo

Fuente: Gómez et al., 2020

Realizado por: Cuvi J., 2024

4.2. Comportamiento temporal

Una subcaracterística de eficiencia de desempeño, según la norma ISO/IEC 25010:2011, implica evaluar la capacidad del sistema para llevar a cabo sus funciones de manera eficiente en términos de tiempo contando con 2 indicadores de evaluación contemplados por el autor del presente trabajo de integración siendo el tiempo de respuesta y tiempo de espera.

4.2.1. Indicadores de evaluación

Los indicadores son fundamentales para comprender cómo el sistema responde y se comporta en relación con el tiempo. La **Tabla 4-4** detalla estos indicadores, incluyendo su propósito, análisis y el proceso de implementación.

Tabla 4-4: Indicadores de evaluación del comportamiento temporal

Indicador	Propósito	Análisis Estadístico	Proceso
Tiempo de respuesta	Medir el tiempo que le lleva al sistema reaccionar a una interacción.	Descriptivo	Ejecutar una serie de pruebas de diagnóstico y análisis técnicos en el sitio, tanto para la versión de escritorio como móvil.
Tiempo de espera	Medir el intervalo total transcurrido desde que se envía la función hasta recibe las pruebas automatizadas.	Descriptivo e inferencial	Comparar el tiempo de desarrollo de pruebas unitarias de manera manual frente al tiempo de desarrollo usando el sitio web a 3 funciones del escenario de pruebas.

Realizado por: Cuvi J., 2024

4.3. Resultados

En este apartado se presentan los resultados de las mediciones realizadas para cada uno de los indicadores seleccionados con el fin de evaluar el comportamiento temporal del sistema.

4.3.1 Tiempo de respuesta

Para la medición de esta métrica se llevó a cabo mediante la aplicación de PageSpeed Insights misma que nos permitió evaluar los tiempos respuesta del sitio web tanto en dispositivos móviles y ordenadores.

En la **Ilustración 4-1** se muestra el reporte general obtenido en dispositivos móviles mediante PageSpeed Insights donde se obtiene un 81% de rendimiento respecto al tiempo de respuesta.

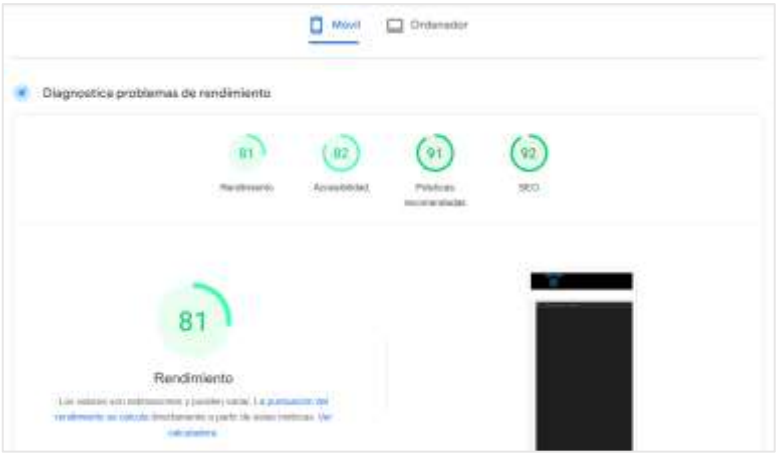


Ilustración 4-1: Reporte general de los tiempos de respuesta en dispositivos móviles

Realizado por: Cuvi J., 2024

En la **Ilustración 4-2** se muestra el reporte a detalle obtenido en dispositivos móviles.

Ilustración 4-2: Reporte detallado de los tiempos de respuesta en dispositivos móviles

Realizado por: Cuvi J., 2024

En la **Ilustración 4-3** se muestra el reporte general obtenido en ordenadores donde se obtiene un 91% de rendimiento respecto al tiempo de respuesta.

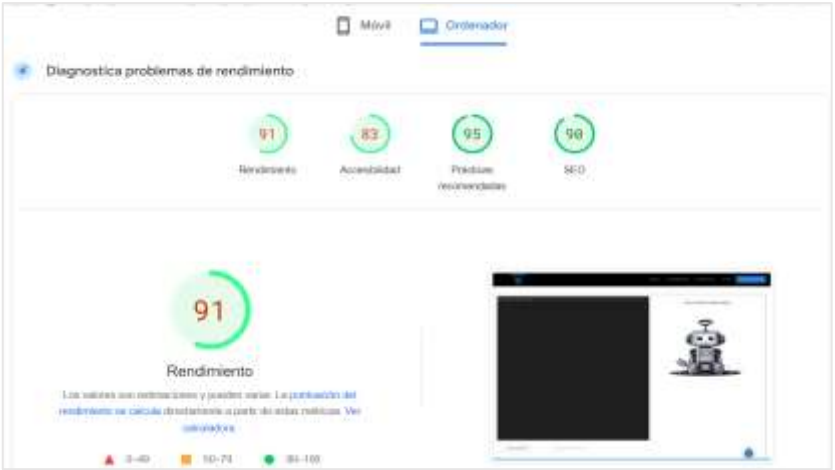


Ilustración 4-3: Reporte general de los tiempos de respuesta - Ordenadores

Realizado por: Cuvi J., 2024

En la **Ilustración 4-4** se muestra el reporte a detalle obtenido en ordenadores mediante PageSpeed Insights.

Ilustración 4-4: Reporte detallado de los tiempos de respuesta en ordenadores

Realizado por: Cuvi J., 2024

En la **Tabla 4-5** se presentan los detalles de los resultados obtenidos del tiempo de respuesta del sitio web mediante la herramienta PageSpeed Insights realizados en ordenadores y dispositivos móviles.

Tabla 4-5: Resultados de las mediciones del tiempo de respuesta.

Métrica	Descripción	Resultado en dispositivos móviles en segundos	Resultado en ordenadores en segundos
Tiempo de carga de la página inicial	Tiempo hasta mostrar el primer contenido	2,6	1,2
Tiempo carga de imágenes	Tiempo hasta mostrar las imágenes de la página inicial.	3,4	1,4
Tiempo Total de Bloqueo	Tiempo total de tareas que bloquean la carga	64	1.31
Cambio Acumulado de Diseño	Movimiento de elementos durante la carga	0,0098	0.000063
Índice de Velocidad	Rapidez percibida de carga del contenido.	4,7	2,7
Resultado porcentual		**81%**	**91%**

Realizado por: Cuvi J., 2024

Para obtener un porcentaje promedio general de los resultados de evaluación del tiempo de respuesta, se realizó la siguiente operación:

$$\%tiempo\ de\ respuesta = \frac{81\% + 91\%}{2}$$

$$\%tiempo\ de\ respuesta = 86\%$$

Con la operación anterior realizada se obtiene un porcentaje de tiempo de respuesta del 86%. Dado que el tiempo de respuesta contribuye con un 50% a la evaluación general del comportamiento temporal del sitio web, se realiza la siguiente operación cuyos valores se muestran a continuación:

$$\%tiempo\ de\ respuesta\ ponderado = 0.5\ x\ 86\%$$
$$\%tiempo\ de\ respuesta\ ponderado = 43\%$$

4.3.2 Tiempo de espera

La medición de esta métrica se llevó a cabo mediante la aplicación de la técnica de observación, utilizando un cronómetro digital en un proceso que involucraba la ejecución de pruebas unitarias a tres funciones dentro del entorno de pruebas diseñado para este proyecto denominado RentaCar. El proceso consistió en obtener el promedio de 31 estudiantes sobre los tiempos de desarrollo de las pruebas unitarias de manera manual y el desarrollo de pruebas unitarias usando el sitio web para posteriormente realizar la respectiva comparación.

La **Tabla 4-6** muestra los resultados obtenidos del proceso de la realización de pruebas unitarias de manera manual.

Tabla 4-6: Resultados del tiempo de espera del desarrollo manual

Proceso	Tiempo promedio en segundos	Tiempo promedio en minutos
Realización de pruebas unitarias a la primera función	428,4	7,14
Realización de pruebas unitarias a la segunda función	279	4,65
Realización de pruebas unitarias a la tercera función	295	4,92

Realizado por: Cuvi J., 2024

- **Tiempo de espera del desarrollo usando el sitio web**

En la **Tabla 4-7** se muestra los resultados obtenidos del proceso de la realización de pruebas unitarias usando el sitio web.

Tabla 4-7: Resultados del tiempo de espera del desarrollo automatizado

Proceso	Tiempo promedio en segundos	Tiempo promedio en minutos
Realización de pruebas unitarias a la primera función	38,56	0,642
Realización de pruebas unitarias a la segunda función	37,29	0,621
Realización de pruebas unitarias a la tercera función	37,48	0,624

Realizado por: Cuvi J., 2024

4.3.3. Comparación de tiempos

En la **Tabla 4-8** se resume los resultados de los tiempos obtenidos en minutos, mismos que muestran los tiempos de desarrollo manual, desarrollo automatizado, incluyendo el porcentaje de reducción del tiempo de espera.

Tabla 4-8: Comparación de tiempos de espera

Proceso	Tiempos de espera de desarrollo manual en minutos	Tiempos de espera de desarrollo automatizado en minutos	Porcentaje de reducción
Realización de pruebas unitarias a la primera función	7,14	0,642	1012.15%
Realización de pruebas unitarias a la segunda función	4,65	0,621	648.8%
Realización de pruebas unitarias a la tercera función	4,92	0,624	688.46%
Promedio porcentual			783.13%

Realizado por: Cuvi J., 2024

En la **Ilustración 4-5** se presenta una comparación detallada de los tiempos de espera en minutos de cada proceso de pruebas realizada. La representación gráfica refleja claramente la marcada diferencia entre el desarrollo manual y el automatizado a través del sitio web.

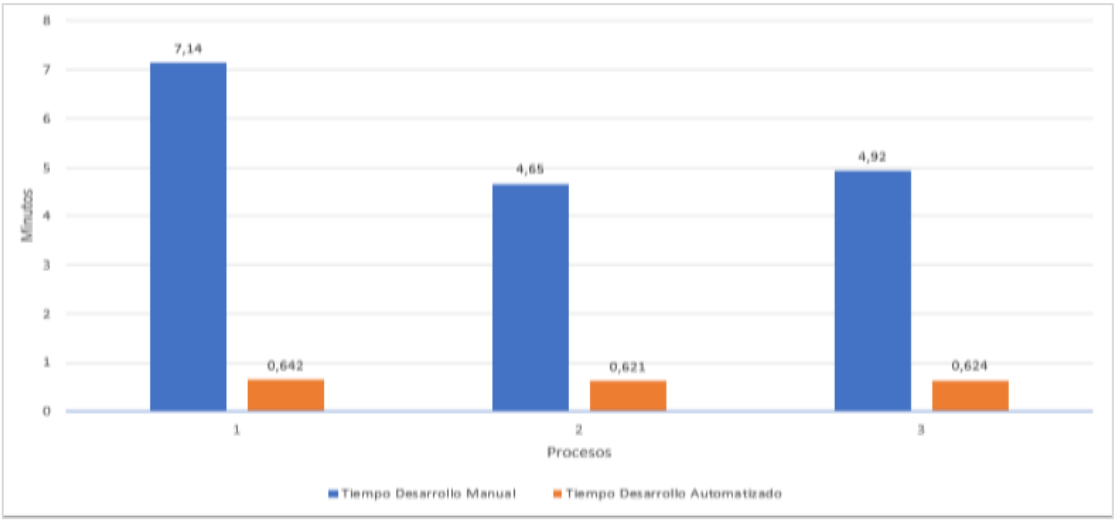


Ilustración 4-5: Comparación de Tiempos de desarrollo

Realizado por: Cuvi J., 2024

La representación gráfica de los resultados detallada en la **Ilustración 4-6** muestra que la sumatoria del tiempo total de desarrollo automatizado es mucho menor respecto a la sumatoria del tiempo total de desarrollo manual.

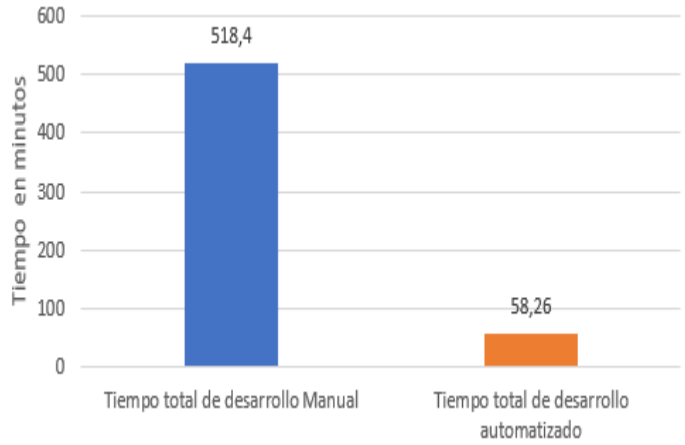


Ilustración 4-6: Tiempo total de desarrollo manual y automatizado

Realizado por: Cuvi J., 2024

Sobre los procesos realizados en la **Tabla 4-8** se aplica la ponderación para el tiempo de espera cuyos valores se muestran a continuación:

$$\%tiempo\ de\ espera\ \ ponderado = \ 0.5\ x\ 783.13\%$$

$$\%tiempo\ de\ espera\ \ ponderado = \ 391.56\%$$

Considerando que el peso del tiempo de espera para la evaluación del comportamiento temporal está valorado en un 50% y el valor del tiempo de espera obtenido es del 391.56% se asigna la

valoración completa para el tiempo de espera siendo el 50% debido a que este valor excede el peso asignado.

$$\%tiempo\ de\ espera\ \ ponderado = 50\%$$

4.4 Análisis, interpretación y discusión de resultados

4.4.1 Análisis descriptivo

En la **Tabla 4-9** se presentan los porcentajes derivados de las mediciones de cada indicador del comportamiento temporal.

Tabla 4-9: Resultados obtenidos para el comportamiento temporal

Subcaracterística	Indicadores	Peso	Porcentaje ponderado%
Comportamiento temporal	Tiempo de respuesta	50%	43%
	Tiempo de espera	50%	50%
	Resultados	100%	93%

Realizado por: Cuvi J., 2024

Los valores obtenidos en la evaluación del comportamiento temporal dan como resultado un 93%, por lo que se interpreta que la solución desarrollada se encuentra en la escala de 91 – 1000 % siendo valorada en **Excelente.**

4.4.2 Análisis inferencial

Para validar los datos recopilados, se realizaron pruebas estadísticas para evaluar su confiabilidad. Al examinar la normalidad de los datos, se identificó que no siguen una distribución normal, optando por el test de Mann-Whitney U debido a su idoneidad para datos no paramétricos, garantizando una evaluación precisa de las diferencias entre las muestras del desarrollo de pruebas manual y el desarrollo de pruebas automatizados.

- *Hipótesis de estudio*

1. **Hipótesis alterna:** Existe una diferencia significativa entre el tiempo que toma el desarrollo de pruebas unitarias haciendo uso del sitio web respecto al tiempo que toma el desarrollo de pruebas unitarias de manera manual.
2. **Hipótesis nula:** No existe una diferencia significativa entre el tiempo que toma desarrollar pruebas unitarias con el sitio web respecto al tiempo que toma desarrollar pruebas unitarias manera manual.

- *Distribución de datos*

Utilizando el lenguaje de programación R, se aplicó la prueba de Shapiro-Wilk para evaluar la normalidad de los tiempos de desarrollo manual y los tiempos con el sistema automatizado, expresados en segundos.

La prueba de Shapiro-Wilk arrojó un p-valor de 0.02 para los tiempos manuales y 0.01 para los tiempos del sistema automatizado considerando un nivel de significancia del 5%. Dado que ambos p-valores son menores a 0.05, se concluye que los tiempos de desarrollo manuales y los tiempos de desarrollo automatizado no siguen una distribución normal tal y como se muestra en la **Tabla 4-10.**

Tabla 4-10: Tabla de los resultados de la prueba de Shapiro-Wilk

Estadístico	Tiempos Manuales	Tiempos Automatizados
P-valor	0.02	0.01
Estadístico W	0.85	0.80
Región de aceptación (95%)	[0.97, 1]	[0.97, 1]
Efecto KS-D	0.15	0.15

Realizado por: Cuvi J., 2024

- *Prueba Wilcoxon*

En el desarrollo de nuestro estudio, se utilizó R Studio como la herramienta para llevar a cabo el análisis estadístico. Se optó por la prueba de Wilcoxon para comparar los tiempos de desarrollo manual y automatizado. La elección de la prueba de Wilcoxon se fundamenta en su capacidad para manejar datos no paramétricos e ideal para realizar procesos con distribuciones no normales.

La **Tabla 4-11** proporcionan detalles de la distribución de los tiempos en ambas muestras, permitiendo una comprensión más completa de la variabilidad y tendencias en los datos.

Tabla 4-11: Resumen estadístico de los tiempos

Medida	Tiempos Manual (s)	Tiempos Automatizado (s)
Mínimo	60.0	0.22
Máximo	1500.0	120.00
Mediana	300.0	35.00
Media	334.5	37.78
N°	93	93
Desviación estándar	254.1037	23.02554
Media de error estándar	26.34933	2.387638

Realizado por: Cuvi J., 2024

Al realizar la prueba de Wilcoxon, se obtuvieron los resultados detallados en la **Ilustración 4-9**. Se observa un valor de p significativamente bajo, menor a 2.2e-16, lo que conduce al rechazo de la hipótesis nula en favor de la hipótesis alternativa. Estos resultados respaldan la hipótesis alterna que indica que hay una diferencia significativa en los tiempos de desarrollo manual y automatizado.

```
        Wilcoxon rank sum test with continuity correction

data:  tiempo_manual and tiempo_automatizado
W = 8593, p-value < 2.2e-16
alternative hypothesis: true location shift is not equal to 0
```

Ilustración 4-7: Resultados de la prueba de Wilcoxon

Realizado por: Cuvi J., 2024

CAPÍTULO V

5. CONCLUSIONES Y RECOMENDACIONES

5.1. Conclusiones

- Se identificaron los principales problemas al desarrollar pruebas unitarias de forma manual de los cuales destacan el excesivo tiempo requeridos para el desarrollo de pruebas unitarias, la falta de experiencia, y los plazos ajustados mismos que afectan el desarrollo de pruebas unitarias provocando que no se realicen pruebas exhaustivas y retrasos generando presión sobre los desarrolladores y testers por el limitado tiempo con los que cuentan para el desarrollo del mismo.

- El estudio realizado a la API de Open AI permitió comprender a profundidad sus características, funcionamiento y comportamiento. Se identificó un destacado rendimiento en tareas de procesamiento de lenguaje natural, variedad de modelos entrenados y opciones de personalización. Los hallazgos permitieron aprovechar eficientemente las capacidades de la API para automatizar pruebas unitarias, resaltando su potencial estratégico y aplicaciones prácticas en este trabajo.

- El desarrollo del sitio web basado en la metodología Scrumban permitió crear una solución web funcional basada en la arquitectura Cliente-Servidor que integra tecnologías como Express JS, Open AI API, React JS y Material UI para conseguir un sitio web eficiente. El enfoque ágil de Scrumban fue fundamental para la entrega exitosa de los componentes requeridos, garantizando una mayor eficiencia y adaptabilidad durante el desarrollo del proyecto lo cual facilitó la adaptación a los cambios.

- La evaluación del comportamiento temporal del sitio web se midió a través de 2 indicadores las cuales fueron ponderadas en un 50% cada una. Los resultados obtenidos son: 43% para el tiempo de respuesta y 50% para el tiempo de espera que sumados dieron un total de 93%. Además, el tiempo de desarrollo de pruebas unitarias se redujo en un 783.13% usando el sitio web respecto al desarrollo unitarias de manera manual por lo que, se puede afirmar que el sitio web desarrollado permite automatizar el desarrollo de pruebas unitarias.

5.2. Recomendaciones

- Se recomienda utilizar la inteligencia artificial como una herramienta complementaria y no como un sustituto de la labor humana. La inteligencia artificial, aunque tiene mucho potencial, puede presentar limitaciones, errores que demandan supervisión y validación humanas. Por lo tanto, se recomienda una estrategia de combinación, donde la colaboración entre la inteligencia artificial y las habilidades cognitivas humanas se traduzca en resultados más sólidos y fiables.

- El desarrollo de pruebas unitarias es muy importante para garantizar un desarrollo de software de calidad por ello se recomienda que se destine el tiempo, los recursos hardware y software, así como el personal especializado que elaboren planes de pruebas bajo un cronograma que permita garantizar que se lleven a cabo suficientes pruebas unitarias permitiendo obtener una mejor cobertura reduciendo la tasa de errores.

- Para trabajos futuros se recomienda explorar la aplicación de esta tecnología para la generación automatizada de otros tipos de pruebas como integración, funcionamiento, seguridad, entre otras.

GLOSARIO

IA (Inteligencia Artificial): Disciplina informática dedicada al desarrollo de sistemas capaces de realizar tareas que requieren la intervención de la inteligencia humana, como el aprendizaje, la percepción visual, el reconocimiento de voz y la toma de decisiones.

API (Interfaz de Programación de Aplicaciones): Conjunto de reglas y herramientas diseñadas para posibilitar la interacción fluida entre distintas aplicaciones de software. Su función principal es facilitar la comunicación y el intercambio de datos entre sistemas diversos, promoviendo la interoperabilidad y la eficiencia en el desarrollo de software.

Frontend: La parte de un sistema informático o aplicación que interactúa directamente con el usuario. Incluye la interfaz gráfica y la experiencia de usuario.

Backend: La parte de un sistema informático o aplicación que gestiona la lógica de negocio, la base de datos y otras funciones que no están directamente relacionadas con la interfaz de usuario.

Framework: Estructura conceptual y tecnológica que sirve de base para el desarrollo de software. Proporciona herramientas, bibliotecas y estándares que facilitan el proceso de creación de aplicaciones.

Cloud: Modelo de prestación de servicios de computación a través de internet. Permite el acceso a recursos informáticos, como almacenamiento y procesamiento, de manera remota y según la demanda.

Testing: Proceso de evaluación de un sistema o aplicación para identificar posibles errores, verificar su funcionamiento correcto y asegurar la calidad del software.

NoSQL: Tipo de sistema de gestión de bases de datos que no utiliza el modelo relacional. Se caracteriza por ser flexible y escalable, adecuado para manejar grandes volúmenes de datos no estructurados.

ESLint: Herramienta de análisis estático para identificar patrones problemáticos en el código JavaScript. Ayuda a mantener un código consistente y libre de errores mediante la aplicación de reglas predefinidas o personalizadas.

Token: Un token es una ocurrencia de una secuencia de caracteres en un texto que se considera como una unidad con significado.

BIBLIOGRAFÍA

1. **CRISTINA LÓPEZ, G.J.** La importancia del testing de software y de la automatización de pruebas. *Knowmadmood.com* [en línea], 2019. [consulta: 4 octubre 2023]. Disponible en: https://www.knowmadmood.com/blog/la-importancia-del-testing-de-software-y-de-la-automatizacin-de-pruebas.

2. **AGGARWAL, S.** Modern Web-Development using ReactJS. *Ijrra.net* [en línea], 2018. [consulta: 6 septiembre 2023]. Disponible en: http://ijrra.net/Vol5issue1/IJRRA-05-01-27.pdf.

3. **AHMED, A.** *Software project management: A process-driven approach* [en línea], 2019. London, England: CRC Press. ISBN 9781439846568. Disponible en: https://www.researchgate.net/publication/327396260_Software_Project_Management_A_Process-Driven_Approach.

4. **ALORES**. 10 retos a los que se enfrenta todo desarrollador de productos de software. *Velneo.com* [en línea], 2021. [consulta: 28 septiembre 2023]. Disponible en: https://www.velneo.com/blog/10-retos-a-los-que-se-enfrenta-todo-desarrollador-de-productos-de-software.

5. **ANDRADE, A.** ¿Por qué la cobertura de pruebas unitarias es una parte importante de QA? *Alex Andrade* [en línea], 2021. [consulta: 4 octubre 2023]. Disponible en: https://alexandrade.net/blog-de-ingenieria-de-software/calidad-de-software/por-que-la-cobertura-de-pruebas-unitarias-es-una-parte-importante-de-qa/.

6. **AUDITOR.** Complejidad ciclomática y calidad del código. *Auditoría de código* [en línea], 2019. [consulta: 28 septiembre 2023]. Disponible en: https://auditoriadecodigo.com/complejidad-ciclomatica-mide-la-calidad-y-seguridad-de-tu-codigo/.

7. **AVILA, D.** Modelos de GPT-3 y Codex. *Medium* [en línea], 2022. [consulta: 12 octubre 2023]. Disponible en: https://medium.com/@dan.avila7/modelos-de-gpt-3-y-codex-11a64948d87.

8. **BAHTIAR SEMMA, A., ALI, M., SAEROZI, M., MANSUR, M. y KUSRINI, K.** Cloud computing: google firebase firestore optimization analysis. *Indonesian journal of electrical engineering and computer science* [en línea], 2023, vol. 29, no. 3, [consulta: 25 noviembre 2023]. ISSN 2502-4752. DOI 10.11591/ijeecs.v29.i3.pp1719-1728. Disponible en: https://www.semanticscholar.org/paper/b8f31acbcc7f7f093154bfea55523d08a409729e.

9. **BAIG, M.S., DR y GOND, S.** On Open Sources Node Js Iot Frame Works. [en línea], 2021, [consulta: 18 septiembre 2023]. Disponible en: https://www.semanticscholar.org/paper/On-Open-Sources-Node-Js-Iot-Frame-Works-Baig-Dr/93c67835d261e20aa04b4ff3e30b846572c17349.

10. **BERTOLINI, M., MEZZOGORI, D., NERONI, M. y ZAMMORI, F.** A scrumban board-based approach to improve material flow in engineering to order (ETO) companies: an industrial application based on action research. *Production planning & control* [en línea], 2023, ISSN 0953-7287. DOI 10.1080/09537287.2023.2248940. Disponible en: http://dx.doi.org/10.1080/09537287.2023.2248940.

11. **BIEVER, C.** ChatGPT broke the Turing test - the race is on for new ways to assess AI. *Nature* [en línea], 2023, vol. 619, no. 7971, ISSN 0028-0836. DOI 10.1038/d41586-023-02361-7. Disponible en: https://www.nature.com/articles/d41586-023-02361-7.

12. **CALDERÓN, A.C.B., HOYOS, E.R.H. y DÍAZ, L.Y.Z.** Diseño de una guía metodológica de gerencia ágil para proyectos de investigación y desarrollo en áreas biológicas. [en línea], 2017, [consulta: 6 septiembre 2023]. Disponible en: https://www.semanticscholar.org/paper/ebb2b6bfe6ccbf8b57eab968526781713db3aad1

13. **CAPEL, M.I., GRIMÁN, A. y GARVÍ, E.** Calidad Ágil: Patrones de Diseño en un contexto de Desarrollo Dirigido por Pruebas. *XXI Jornadas de Ingeniería del Software y Bases de Datos*. [en línea], 2017, [consulta: 6 septiembre 2023]. Disponible en: https://www.semanticscholar.org/paper/ebb2b6bfe6ccbf8b57eab968526781713db3aad1

14. **BIEVER, C.** Características del Software. *CAVSI* [en línea], 2015. [consulta: 14 octubre 2023]. Disponible en: https://www.cavsi.com/espanol/blog/caracteristicas-del-software/.

15. **CARION, U.** JSON Data Definition Format (JDDF). [en línea], 2020, [consulta: 6 septiembre 2023]. Disponible en: https://www.semanticscholar.org/paper/6a03828c5e23f83f4956303771a90db60c9bb53.

16. **CHINARRO MORALES, E., RUIZ RIVERA, M.E. y RUIZ LIZAMA, E.** Desarrollo de un Modelo de Pruebas Funcionales de Software Basado en la Herramienta SELENIUM. *Industrial data* [en línea], 2017, vol. 20, no. 1, ISSN 1560-9146. DOI 10.15381/idata.v20i1.13498. Disponible en: https://www.redalyc.org/pdf/816/81652135017.pdf.

17. **AGUAYO CÁCERES, S.I. y FREIRE OROZCO, C.V.** *Desarrollo de una plataforma web de comercio electrónico B2C para las Pymes de la ciudad de Macas utilizando el framework Express.js*. S.l.: Escuela Superior Politécnica de Chimborazo.

18. **ENTERPRISE PRIVACY.** *Openai.com* [en línea], 2023. [consulta: 21 septiembre 2023]. Disponible en: https://openai.com/enterprise-privacy.

19. **ESPINOZA, J.J.V., IPANAQUE, W.** Arquitectura cliente-servidor del laboratorio remoto. *ResearchGate* [en línea], 2021. [consulta: 1 noviembre 2023]. Disponible en: https://www.researchgate.net/figure/Arquitectura-cliente-servidor-del-laboratorio-remoto_fig5_327564949.

20. **GADIA, R., SHAH, R., VARSHNEY, S. y SAWANT, V.** A system on automated database and API (application programming interface) management. *International journal for research in applied science and engineering technology* [en línea], 2022, vol. 10, no. 4, ISSN 2321-9653. DOI 10.22214/ijraset.2022.41827. Disponible en: https://www.ijraset.com/best-journal/automated-database-and-api-management.

21. **GAMIDO, H.V. y GAMIDO, M.V.** Comparative review of the features of automated software testing tools. *International Journal of Electrical and Computer Engineering (IJECE)* [en línea] 2019, vol. 9, no. 5, ISSN 2088-8708. DOI 10.11591/ijece.v9i5.pp4473-4478. Disponible en: https://ijece.iaescore.com/index.php/IJECE/article/view/17570.

22. **GARCÍA, F.** ¿Qué es OpenAI? Objetivos, limitaciones y versión paga. *Cliengo Blog* [en línea], 2023. [consulta: 21 septiembre 2023]. Disponible en: https://blog.cliengo.com/que-es-openai/.

23. **GÓMEZ DEL MÓNACO, A.P.** *Tests automatizados en aplicaciones web*. S.l.: Universidad Nacional de La Plata.

24. **GORDON GRAELL, R.D., GONZÁLEZ ÁGUILA, M.C. y BULTRÓN BATISTA, A.V.** Desarrollo de chatbots con la aplicación chatgpt: una revisión bibliográfica. *Centros: Revista Científica Universitaria* [en línea], 2023, vol. 12, no. 2, ISSN 2953-3007. DOI 10.48204/j.centros.v12n2.a4048. Disponible en: https://uptv.up.ac.pa/index.php/centros/article/view/4048.

25. **GRINBERG, M.** La guía máxima para el modelo de lenguaje GPT-3 de OpenAI. *Twilio Blog* [en línea] .2020. [consulta: 12 octubre 2023]. Disponible en: https://www.twilio.com/es-mx/blog/la-guia-maxima-para-el-modelo-de-lenguaje-gpt-3-de-openai.

26. **GUIDOM. MANTILLA, B., JOFFREL. LEON, V. y LUISA. JURADO, P.** Análisis & diseño orientado a objetos de una aplicación de gestión comercial usando arquitectura cliente/servidor. [en línea], 2018, [consulta: 2 noviembre 2023]. Disponible en: https://www.semanticscholar.org/paper/2eb7fba1f208c0c151b6e256f538362e0e7b55a6.

27. **HNP.** Arquitectura basada en servicios - El sistema distribuido más simple. *Blog Hector Poblete* [en línea], 2021. [consulta: 8 septiembre 2023]. Disponible en: https://hectorpoblete.cl/2021/03/23/arquitectura-basada-en-servicios-el-sistema-distribuido-mas-simple/.

28. **IJRASET PUBLICATION.** A System on Automated Database and API (Application Programming Interface) Management. *International Journal for Research in Applied Science & Engineering Technology (IJRASET)* [en línea], 2022, [consulta: 5 septiembre 2023]. Disponible en: https://www.academia.edu/78512002/A_System_on_Automated_Database_and_API_Application_Programming_Interface_Management.

29. **ISO 25010.** *Iso25000.com* [en línea], 2011. [consulta: 1 noviembre 2023]. Disponible en: https://iso25000.com/index.php/normas-iso-25000/iso-25010.

30. **JAYATILLEKE, S. y LAI, R.** A systematic review of requirements change management. *Information and software technology* [en línea], vol. 93, ISSN 0950-5849. DOI 10.1016/j.infsof.2017.09.004. Disponible en: https://www.sciencedirect.com/science/article/pii/S0950584917304664.

31. **LOPEZOSA, C. y CODINA, L.** ChatGPT y software CAQDAS para el análisis cualitativo de entrevistas: pasos para combinar la inteligencia artificial de OpenAI con ATLAS.ti, Nvivo y MAXQDA. [en línea] 2023, [consulta: 4 septiembre 2023]. Disponible en: https://repositori.upf.edu/handle/10230/55477.

32. **LUO, Q., HARIRI, F., ELOUSSI, L. y MARINOV, D.** An empirical analysis of flaky tests. *Illinois.edu* [en línea], 2021. [consulta: 28 septiembre 2023]. Disponible en: https://mir.cs.illinois.edu/lamyaa/publications/fse14.pdf.

33. **MASCHERONI, M.A. y IRRAZÁBAL, E.** Problemas que afectan a la Calidad de Software en Entrega Continua y Pruebas Continuas. *Edu.ar* [en línea], 2018. [consulta: 28 septiembre 2023]. Disponible en: http://sedici.unlp.edu.ar/bitstream/handle/10915/73270/Documento_completo.pdf-PDFA.pdf?sequence=1&isAllowed=y.

34. **MEENAKSHI, D., SWAMI NAIK, J. y RAGHAVENDRA REDDY, M.** Software testing techniques in software development life cycle. [en línea], 2014. [consulta: 28 septiembre

2023]. Disponible en: https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=402de360180f292bfbd1eb0341c9ef0891ec4bc0.

35. **MIRANDA, L.** OpenAI lanza una API para que todos puedan usar ChatGPT en sus aplicaciones. *Hipertextual* [en línea], 2023. [consulta: 21 septiembre 2023]. Disponible en: http://hipertextual.com/2023/03/openai-lanza-api-chatgpt-ia-apps.

36. **MRBULLWINKLE.** Referencia de la API REST de los servicios de Azure OpenAI. *Microsoft.com* [en línea], 2023. [consulta: 21 septiembre 2023]. Disponible en: https://learn.microsoft.com/es-es/azure/ai-services/openai/reference.

37. **MRBULLWINKLE.** Filtrado de contenido. *Microsoft.com* [en línea], 2023. [consulta: 12 octubre 2023]. Disponible en: https://learn.microsoft.com/es-es/azure/ai-services/openai/concepts/content-filter.

38. **OFOEDA, J., BOATENG, R. y EFFAH, J.** Application programming interface (API) research: A review of the past to inform the future. *International journal of enterprise information systems* [en línea], vol. 15, no. 3, ISSN 1548-1115. DOI 10.4018/ijeis.2019070105. Disponible en: https://www.igi-global.com/article/application-programming-interface-api-research/232166.

39. **OLIVEROS, A., DANYANS, F.J. y MASTROPIETRO, M.L.** Prácticas de Ingeniería de Requerimientos en el desarrollo de aplicaciones Web. *17th Workshop on Requirements Engineering*. S.l. [en línea], 2023 [consulta: 12 octubre 2023]. Disponible en: https://learn.microsoft.com/es-es/azure/ai-services/openai/concepts/content-filter.,

40. **OROZCO, G. y RIGHI, L.G.** Integración continua: solución a los problemas de productividad y calidad del código en un entorno ágil testigo. [en línea], 2018, [consulta: 28 septiembre 2023]. Disponible en: https://rdu.iua.edu.ar/handle/123456789/1781.

41. **OWASP.** Category:OWASP Code Review Project. *Owasp.org* [en línea], 2017. [consulta: 28 septiembre 2023]. Disponible en: https://wiki.owasp.org/index.php/Category:OWASP_Code_Review_Project.

42. **PÁEZ, K.P.N. y VINUEZA, O.E.P.** Desarrollo de una aplicación móvil que facilite la repartición de alimentos para el proyecto Pancomido. [en línea], 2019, [consulta: 25 noviembre 2023]. Disponible en: https://www.semanticscholar.org/paper/bd191faf16d5691a53556a1488ffbf7dd3f1bec9

43. **PALAMARCHUK, S.** Cobertura de Pruebas de Software: ¿cómo aumenta con la Automatización? *Blog de Testing y Calidad de Software | Abstracta Chile* [en línea], 2020. [consulta: 4 octubre 2023]. Disponible en: https://cl.abstracta.us/blog/cobertura-pruebas-software-automatizacion/.

44. **PAZ, J.A.M., GÓMEZ, M.Y.M. y ROSAS, S.C.** Análisis sistemático de información de la Norma ISO 25010 como base para la implementación en un laboratorio de Testing de software en la Universidad Cooperativa de Colombia Sede Popayán. *Memorias de Congresos UTP* [en línea], 2017, [consulta: 6 septiembre 2023]. Disponible en: https://revistas.utp.ac.pa/index.php/memoutp/article/view/1483.

45. **PEIRÓ, K.** GPT-3: ¿Creatividad literaria de la inteligencia artificial? *Think by SHIFTA* [en línea], 2020. [consulta: 28 octubre 2023]. Disponible en: https://medium.com/think-by-shifta/gpt-3-la-esperada-creatividad-literaria-de-la-inteligencia-artificial-d200aa974596.

46. **PEREDO VALDERRAMA, R. y PEREDO VALDERRAMA, I.** Propuesta de Arquitectura basada en Componentes de Software para el desarrollo de Materiales Educativos Didácticos Web bajo el modelo de Educación Basada en Web: Architecture Proposal Based on Software Components for the Development of Web Educational Materials under the Web-Based Education model. *Tecnología Educativa Revista CONAIC* [en línea], 2020, vol. 7, no. 1, ISSN 2395-9061. DOI 10.32671/terc.v7i1.9. Disponible en: https://terc.mx/index.php/terc/article/view/9.

47. **PETRICIOLI, L. y FERTALJ, K.** Agile software development methods and hybridization possibilities beyond scrumban. *2022 45th Jubilee International Convention on Information, Communication and Electronic Technology (MIPRO)*. S.l.: IEEE.

48. **ROMERO, G.** ¿Qué son las pruebas unitarias? Proceso, retos, herramientas y más. [en línea], 2021, [consulta: 28 septiembre 2023]. Disponible en: https://www.zaptest.com/es/que-son-las-pruebas-unitarias-profundice-en-el-proceso-los-beneficios-los-retos-las-herramientas-y-mucho-mas/.

49. **SALAZAR FIERRO, F.A., PINEDA MANOSALVAS, C.A., CERVANTES RODRÍGUEZ, N.N. y LANDETA, P.** Home page. *Doi.org* [en línea], 2020. [consulta: 7 octubre 2023]. Disponible en: https://doi.org/.

50. **SÁNCHEZ LEDESMA, F.A.** *Development, analysis and deployment of component-based applications with real-time requirements. A model-driven approach*. S.l.: Universidad Politecnica de Cartagena.

51. **SERNA L´ÓPEZ, G.A., JARAMILLO TERÁN, L.M., UPEGUI GIRALDO, G. de J., GÓMEZ GIRALDO, R. y BUENO, Y.** Piloto de automatización de pruebas de software en la Unidad de Servicios Tecnológicos del Centro de Servicios y Gestión Empresarial (CESGE) del SENA. *Cintex* [en línea], 2020, vol. 25, no. 2, ISSN 0122-350X. DOI 10.33131/24222208.363. Disponible en: https://revistas.pascualbravo.edu.co/index.php/cintex/article/view/363.

52. **STRIPE.** Stripe y OpenAI colaboran para monetizar los productos estrella de OpenAI y mejorar Stripe con GPT-4. *Stripe.com* [en línea], 2023. [consulta: 22 septiembre 2023]. Disponible en: https://stripe.com/es-us/newsroom/news/stripe-and-openai.

53. **SURAMEERY, N.M.S. y SHAKOR, M.Y**. Use Chat GPT to solve programming bugs. *International Journal of Information Technology & Computer Engineering (IJITC) ISSN : 2455-5290* [en línea], vol. 3, no. 31, ISSN 2455-5290. DOI 10.55529/ijitc.31.17.22. Disponible en: http://journal.hmjournals.com/index.php/IJITC/article/view/1679.

54. **UNIR, V.** La importancia de las pruebas de software. *UNIR* [en línea], 2022. [consulta: 20 septiembre 2023]. Disponible en: https://www.unir.net/ingenieria/revista/pruebas-software/.

55. **VELÁZQUEZ, E.M.** Siete problemas al probar programas. *Github.io* [en línea], 2021. [consulta: 28 septiembre 2023]. Disponible en: https://emanchado.github.io/camino-mejor-programador/html/ch06.html.

56. **BRANAGHAN, R.J. y SANCHEZ, C.A.** Feedback preferences and impressions of waiting. *Human factors* [en línea], 2009, vol. 51, no. 4, [consulta: 12 enero 2024]. ISSN 0018-7208. DOI 10.1177/0018720809345684. Disponible en: https://www.semanticscholar.org/paper/5780608b4e4d031d5dac193ba545446f82af2127.

ANEXOS

ANEXO A: ESTUDIO DE FACTIBILIDAD TÉCNICA

A continuación, se detalla el estudio realizado para determinar la viabilidad de realizar el proyecto.

En la **Tabla 1** se encuentran descritas las características mínimas junto al que se cuenta para el desarrollo del proyecto.

Tabla 1: Hardware requerido

Cantidad	Nombre	Características mínimas	Características disponibles
1	Laptop	Procesador: AMD RYZEN 3 3500U	Procesador: AMD RYZEN 7 3700U
		RAM: 4.0 GB	RAM: 12 GB
		Almacenamiento: 128 GB	Almacenamiento: 1 Tera
1	Router	Conexiones: LAN, WLAN	Conexiones: POWER, PON, LOS, LAN1, LAN2, LAN4, TEL, WLAN y WPS
		Velocidad: 20 megabits por segundo	Velocidad: 100 megabits por segundo

Realizado por: Cuvi J., 2023

Las herramientas de software necesarias para llevar a cabo el desarrollo de este proyecto se han detallado a la **Tabla 2.**

Tabla 2: Software requerido

Software	Descripción	Disponibilidad
OPEN AI API	Plataforma de acceso a modelos de lenguaje avanzados y generación de texto.	Si
Firebase	Plataforma de desarrollo de aplicaciones web y móviles con servicios en la nube de Google	Si
Node JS	Entorno de ejecución de JavaScript	Si
React JS	Biblioteca de JavaScript	Si
Visual Studio Code	IDE	Si
Git	Control de versiones	Si
GitHub	Plataforma de versiones	Si
Material UI	Biblioteca de componentes	Si

Realizado por: Cuvi J., 2023

ANEXO B: ESTUDIO DE FACTIBILIDAD ECONÓMICA

En esta sección, se hace el análisis económico para evaluar la viabilidad financiera del proyecto y calcular los posibles gastos que podrían estar involucrados en su realización.

Dentro de la **Tabla 1** se encuentran los recursos técnicos que serán necesarios para el desarrollo del proyecto.

Tabla 1: Recursos técnicos

Descripción	Cantidad	Valor unitario ($)	Total ($)
OPEN AI API	1 token	0,008	87,00
Laptop Lenovo Ryzen 7 3500u	1	850,00	850,00
Servidor web	1 año	34,00	34,00
Disco Solido 1 Tera	1	65,00	65,00
		Total	**1036,00**

Realizado por: Cuvi J., 2023

Otros recursos necesarios para el desarrollo del presente trabajo se describen en la **Tabla 2.**

Tabla 2: Otros recursos

Descripción	Cantidad	Valor unitario ($)	Total ($)
Materiales de oficina	varios	$19,00	$19,00
Servicios de internet	6 meses	25,00	150,00
Servicios básicos	6 meses	15,00	90,00
		Total	**259,00.**

Realizado por: Cuvi J., 2023

Finalmente, realizando la suma de ambos recursos suman la cantidad de 1295,00 dólares americanos.

ANEXO C: ESTIMACIONES

Para la determinación de las estimaciones respectivas se sigue el siguiente proceso:

1. Identificación de funciones

En el proceso de estimación de puntos de función, se deben identificar las funciones o funcionalidades del sistema y organizarlas en cinco categorías: Entradas, Salidas, Consultas, Archivos e Interfaces.

1. Entradas (EI, External Inputs): Son las funciones que permiten al usuario ingresar datos al sistema.
2. Salidas (EO, External Outputs): Son las funciones que generan resultados o información para los usuarios.
3. Consultas (EQ, External Inquiry): Son las funciones que permiten a los usuarios realizar consultas o solicitar información específica al sistema.
4. Archivos (ILF, Internal Logical Files): Archivos pueden ser creados, leídos, actualizados o eliminados por el sistema.
5. Interfaces (EIF, External Interface Files): Son las funciones que representan la interacción del sistema con otras aplicaciones o sistemas externos.

En la **Tabla 1** se clasifica y pondera cada función por su nivel de complejidad (baja, intermedia, alta), en este paso se tiene PFs sin ajustar.

Tabla 1: Factor de ponderación

	Factor de ponderación		
Parámetros de medición	**Simple**	**Medio**	**Complejo**
Número de entradas de usuario	3	4	6
Número de salidas de usuario	4	5	7
Número de peticiones de usuario	3	4	6
Numero de archivos	7	10	15
Número de interfaces externas	5	7	10

Fuente: Busquelle, 2010

Realizado por: Cuvi J., 2023

A continuación, en la **Tabla 2** se presenta la tabla general de la ponderación de cada una las categorías:

Tabla 2: Estimación de puntos de función

Categoría	**Descripción**	**Complejidad**	**PF**
	El sistema permitirá registrar cuentas	Simple	3
	El sistema permitirá iniciar sesión con el usuario y contraseña	Simple	3

Entradas	El sistema permitirá que el usuario reestablezca su contraseña	Simple	3
	El sistema debe permitir que se ingrese el código de la función	Simple	3
Salidas	El sistema debe permitir visualizar el resultado de la generación de pruebas	Medio	5
	El sistema deber permitir visualizar el historial de pruebas guardadas en la base de datos.	Medio	5
Consultas	El sistema permitirá eliminar la cuenta de un usuario	Simple	3
	El sistema debe permitir eliminar un historial	Simple	3
Archivos	El sistema debe permitir guardar en un historial las pruebas generadas.	Medio	7
	El sistema debe permitir guardar la información ingresa del código de la función.	Simple	7
Interfaces	Panel principal de visualización de ingreso y figuración del código.	Medio	7
	Panel secundario de visualización del historial.	Simple	54
		Total PF sin ajuste	60

Realizado por: Cuvi J., 2023

2. Determinación del factor de ajuste

Para la determinación del VAF se presenta como primer punto en la **Tabla 3** la ponderación de las características generales.

Tabla 3: Ponderación de las características generales.

Nombre	**Ponderación**
Sin influencia	0
Incidental	1
Moderado	2
Medio	3
Significativo	4
Absolutamente esencial	5

Realizado por: Cuvi J., 2023

En la **Tabla 4** se presentan las características generales junto a su estimación de acuerdo con el sitio web.

Tabla 4: Características generales del software

Características generales del software	**Categoría**	**Estimación**
¿El software que estamos haciendo debe satisfacer todas las especificaciones establecidas por el cliente?	Corrección	2
Facilidad de aprendizaje: Debe ser sencillo de aprender.	Usabilidad	5
Un software de calidad no debe tener efectos secundarios.	Integridad	3
El producto de software no debería tener ningún defecto. No sólo esto, no debe fallar mientras la ejecución.	Fiabilidad	4
El software debe hacer un uso eficaz del espacio de almacenamiento y el comando ejecutar según los requisitos de tiempo deseados.	Eficiencia	5
Los cambios en el software deben ser fácil de hacer.	Flexibilidad	4
Se deben tomar medidas apropiadas para mantener los datos a salvo de las amenazas externas	Seguridad	4
Debe ser muy fácil de actualizar para más trabajo	Escalabilidad	3
Prueba del software debe ser fácil.	Capacidad de prueba	2
Debe estar compuesto por unidades y módulos independientes entre sí.	Modularidad	4
Debe ser fácil de aumentar nuevas funciones.	Extensibilidad	2
Capacidad para llevar a cabo las mismas funciones en todos los entornos y plataformas.	Portabilidad	3
Es poder utilizar el código de software con algunas modificaciones para diferentes propósitos.	Reutilización	5
Total (TDI)		**44**

Fuente: (Bertolini et al. 2023)

Realizado por: Cuvi J., 2023

Se procede a realizar la suma los valores de las 10 características y se calcula el VAF de acuerdo con la ecuación:

Tabla 5: Cálculo del VAF

VAF = (TDI x 0,01) + 0,65

VAF =	**TDI**	**0,01**	**0,65**
VAF =	44	0,01	0,65
VAF =	1,09		

Realizado por: Cuvi J., 2023

Por último, para calcular los PFs ajustados se emplea la ecuación:

Tabla 6: Cálculo de PFs

PFs ajustados = PFs sin ajustar x VAF

PFs ajustados =	PFs sin ajustar	VAF
PFs ajustados =	54	1,09
PFs ajustados =	58.86	

Realizado por: Cuvi J., 2023

Utilizando los puntos de función ajustados, es posible convertirlos en líneas de código (LOC) y aplicar el modelo COCOMO para estimar el esfuerzo. El lenguaje JavaScript, en promedio, un punto de función equivale a aproximadamente 47 líneas de código. Para calcular los KLOC (kilolíneas de código, que se utilizan en COCOMO simple), se emplea la siguiente ecuación:

Tabla 7: Cálculo de los KLOC

kLOC = (PFs ajustados x LOC_leng)/1000

kLOC =	PF ajustados	LOC_leng (JavaScript)	1000
kLOC =	58.86	47	1000
kLOC =	2.7664		

Realizado por: Cuvi J., 2023

Con el KLOC estimado puede entonces usarse el modelo COCOMO simple para estimar los hombres-mes, duración y recursos. Los coeficientes c1, c2, c3 se determinan de acuerdo con el tipo de proyecto:

- C1 = 2.4
- C2 = 1.05
- C3 = 0.38

Esfuerzo medido en Hombres-Mes (H*M):

Tabla 8: Cálculo del esfuerzo medido en hombres-mes

H*M = C1 * kLOC^C2

H*M =	**C1**	**kLOCs**	**C2**
H*M =	2,4	2.7664	1,05
H*M =	6.9858		

Realizado por: Cuvi J., 2023

Tiempo de desarrollo en meses (T Desarrollo):

Tabla 9: Cálculo del tiempo de desarrollo en meses

T Desarrollo = 2,5 * (H*M)^C3

T Desarrollo =	2,5	H*M	C3
T Desarrollo =	2,5	6.9858	0,38

T Desarrollo =	5.23

Realizado por: John Cuvi 2023

Número de programadores (N° **Prog**ramadores**):**

Tabla 10: Cálculo del número de programadores

N° Programadores = (H*M)/(T Desarrollo)

N° Programadores =	**H*M**	T Desarrollo
N° Programadores =	6.9858	5.23
N° Programadores =	1.326	

Realizado por: Cuvi J., 2023

ANEXO D: PLAN DE REDUCCIÓN, SUPERVISIÓN Y GESTIÓN DE RIESGOS

<table>
<tr><th colspan="3">FICHA DE GESTIÓN DE RIESGO DE PRIORIDAD - R1</th></tr>
<tr><td>Probabilidad: Medio</td><td>Impacto: Moderado</td><td>Exposición: Alta</td></tr>
<tr><td>Porcentaje: 40%</td><td rowspan="2">Valor: 3</td><td rowspan="2">Valor: 6</td></tr>
<tr><td>Valoración: 2</td></tr>
<tr><td colspan="2">Descripción: Robo del equipo de cómputo</td><td>Fecha: 9/11/2023</td></tr>
<tr><td colspan="3">Refinamiento:
• Causas
El estado actual del país con los robos frecuentes a jóvenes estudiantes
• Consecuencias
Pérdida de tiempo en el desarrollo del trabajo.
Aumento de costos al proyecto.</td></tr>
<tr><td colspan="3">Reducción:
• Evitar lugares considerados peligrosos.
• No llevar el equipo informático a lugares con altos índices de robos.</td></tr>
<tr><td colspan="3">Supervisión:
• Planificar rutas seguras por donde transitar.</td></tr>
<tr><td colspan="3">Gestión:
• Disponer de mochila antirrobo.</td></tr>
<tr><td colspan="3">Estado Actual:
Fase de reducción iniciado ■
Fase de supervisión iniciada ☐
Gestionando el riesgo ☐</td></tr>
<tr><td colspan="3">Encargados:
• John Cuvi</td></tr>
</table>

HOJA DE GESTIÓN DE RIESGO DE PRIORIDAD - R2		
Probabilidad: Bajo	**Impacto:** Bajo	**Exposición:** Bajo
Porcentaje: 30%	**Valor:** 1	**Valor:** 1
Valoración: 1		
Descripción: Daño del equipo de cómputo		**Fecha:** 9/11/2023

Refinamiento:

- **Causas**

Mal funcionamiento de algún componente.

Vida útil de cada equipo.

- **Consecuencias**

Pérdida de tiempo en el desarrollo del trabajo.

Aumento de costos al proyecto.

Reducción:

- Mantenimiento preventivo a todos los equipos.

Supervisión:

- Revisión periódica del estado del equipo.

Gestión:

- Disponer de herramientas necesarias para el mantenimiento.
- Disponer de alimentador eléctrico calificado.

Estado Actual:

Fase de reducción iniciado	☐
Fase de supervisión iniciada	■
Gestionando el riesgo	☐

Encargado:

- John Cuvi

HOJA DE GESTIÓN DE RIESGO DE PRIORIDAD – R3		
Probabilidad: Medio	**Impacto:** Alto	**Exposición:** Alto
Porcentaje: 40%	**Valor:** 3	**Valor:** 6
Valoración: 2		
Descripción: Diseño erróneo de base de datos.		**Fecha:** 9/11/2023

Refinamiento:

- **Causas**

Mala comprensión de requisitos

Consecuencias

Fallas en el sitio web.

Retraso en el proyecto

Aumento de costos

Reducción:

- Analizar de manera correcta e interpretar de manera adecuada los requisitos.

Supervisión:

- Verificar el diseño gráfico de la base de datos
- Analizar si las entidades y relaciones sean las correctas en el modelo entidad relación.

Gestión:

- Verificar periódicamente los Datos almacenados, corregir a tiempo posibles cambios.

Estado Actual:

Fase de reducción iniciado ☐

Fase de supervisión iniciada ☐

Gestionando el riesgo ☐

Encargado:

- John Cuvi

HOJA DE GESTIÓN DE RIESGO DE PRIORIDAD - R4		
Probabilidad: Medio	**Impacto:** Critico	**Exposición:** Medio
Porcentaje: 60%	**Valor:** 4	**Valor:** 12
Valoración: 3		
Descripción: Cambio de políticas de la OPEN AI API		**Fecha:** 9/11/2023

Refinamiento:

- **Causas**

Perdidas constantes a la empresa

Ataques realizados usando la API

Uso inadecuado de la API

Consecuencias

Suspensión del proyecto

Reducción:

- Realizar un uso adecuado de la API.
- Respetar las reglas impuestas.

Supervisión:

- Verificar la implementación de nuevas políticas de su uso.

Gestión:

- Utilizar otros modelos propios de OPEN AI API comercializadas por terceros.

Estado Actual:

Fase de reducción iniciado ☐

Fase de supervisión iniciada ☐

Gestionando el riesgo ■

Encargado:

- John Cuvi

HOJA DE GESTIÓN DE RIESGO DE PRIORIDAD – R5		
Probabilidad: Bajo	**Impacto:** Bajo	**Exposición:** Bajo
Porcentaje: 20%	**Valor:** 1	**Valor:** 1
Valoración: 1		
Descripción: Cambio de autoridades		**Fecha:** 9/11/2023

Refinamiento:

- **Causas**

Recorte de presupuesto de la universidad

Nuevas resoluciones administrativas

Consecuencias

Retraso del trabajo

Reducción:

- Comunicación constante con las autoridades de la universidad y facultad.

Supervisión:

- Verificar la implementación de nuevas resoluciones administrativas.

Gestión:

- Solicitar a la FEPOCH que este atenta ante cualquier circunstancia.

Estado Actual:

Fase de reducción iniciado ☐

Fase de supervisión iniciada ☐

Gestionando el riesgo ☐

Encargado:

- John Cuvi

<table>
<tr><th colspan="3">HOJA DE GESTIÓN DE RIESGO DE PRIORIDAD – R6</th></tr>
<tr><td>Probabilidad: Alto</td><td>Impacto: Moderado</td><td>Exposición: Media</td></tr>
<tr><td>Porcentaje: 40%</td><td rowspan="2">Valor: 2</td><td rowspan="2">Valor: 4</td></tr>
<tr><td>Valoración: 2</td></tr>
<tr><td colspan="2">Descripción: Mal rendimiento del sitio web</td><td>Fecha: 9/11/2023</td></tr>
<tr><td colspan="3">Refinamiento:
• Causas
Mala implementación de la arquitectura
Frameworks con obsoletos
Consecuencias
Retraso del trabajo
Abandono del proyecto
Aumento en costos</td></tr>
<tr><td colspan="3">Reducción:
• Uso adecuado de patrones de diseño.
• Implementación adecuada de componentes</td></tr>
<tr><td colspan="3">Supervisión:
• Verificar las versiones y actualizaciones de las herramientas que se están usando.</td></tr>
<tr><td colspan="3">Gestión:
• Poner a prueba nuestro sitio web en medidores de rendimiento disponibles en Internet.</td></tr>
<tr><td colspan="3">Estado Actual:
Fase de reducción iniciado ☐
Fase de supervisión iniciada ■
Gestionando el riesgo ☐</td></tr>
<tr><td colspan="3">Encargado:
• John Cuvi</td></tr>
</table>

ANEXO E: INTERFACES GRÁFICAS DEL SITIO WEB

- **Inicio de sesión**

En la **Figura 1** se muestra el diseño del inicio de sesión el cual permitirá a los usuarios acceder a las funcionalidades del sistema mediante sus credenciales.

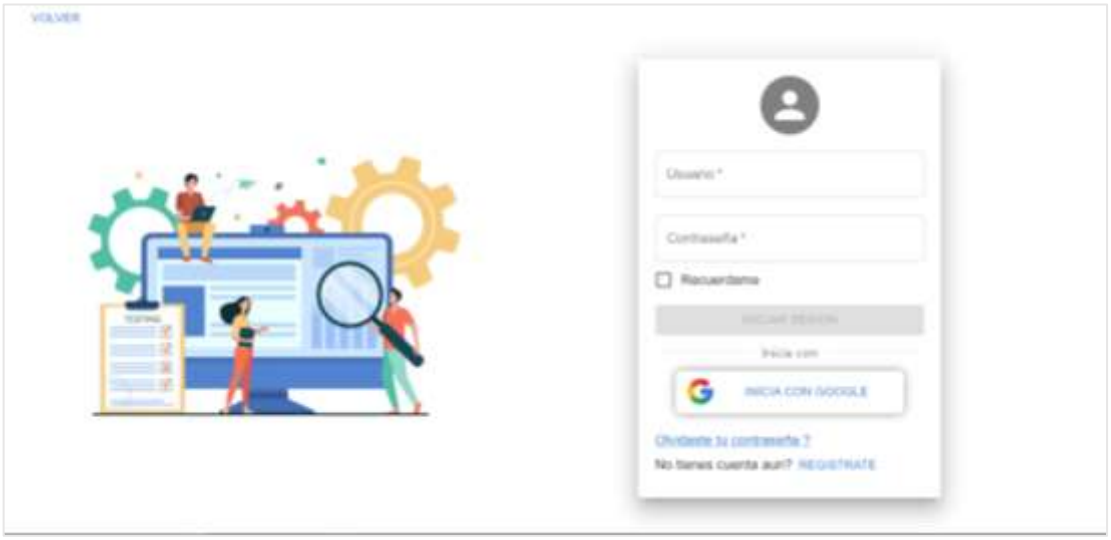


Figura 1: Interfaz de inicio de sesión

Realizado por: Cuvi J., 2023

- **Creación de cuenta**

En la **Figura 4** se muestra el diseño de creación de cuenta de un usuario, misma que puede registrarse por medio de su cuenta de Google, así como ingresando un correo y contraseña.

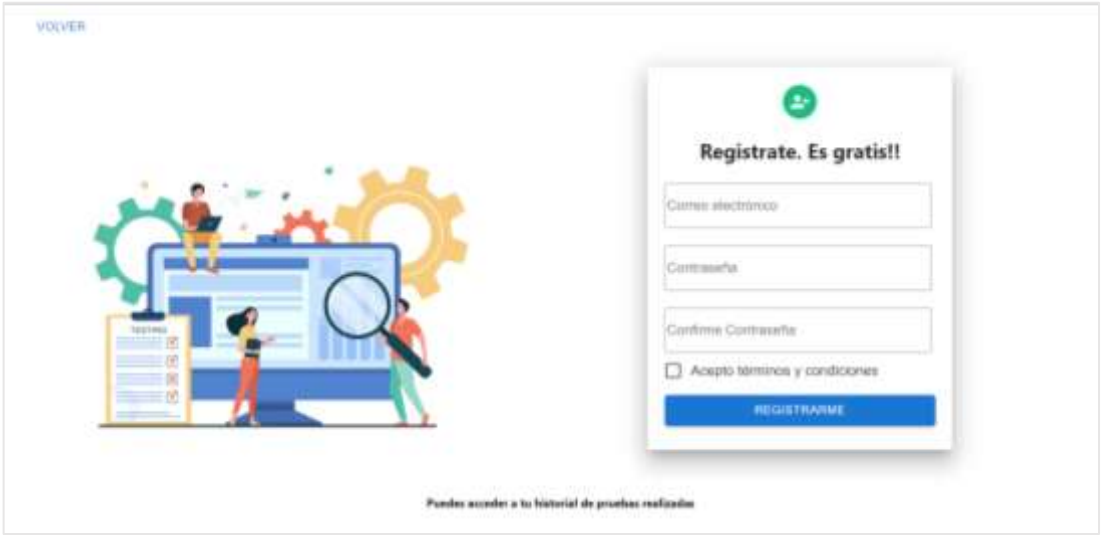


Figura 2: Interfaz de creación de cuenta

Realizado por: Cuvi J., 2023

- **Pantalla principal usuario registrado**

En la **Figura 5** se muestra la pantalla principal de un usuario registrado en el sitio web.

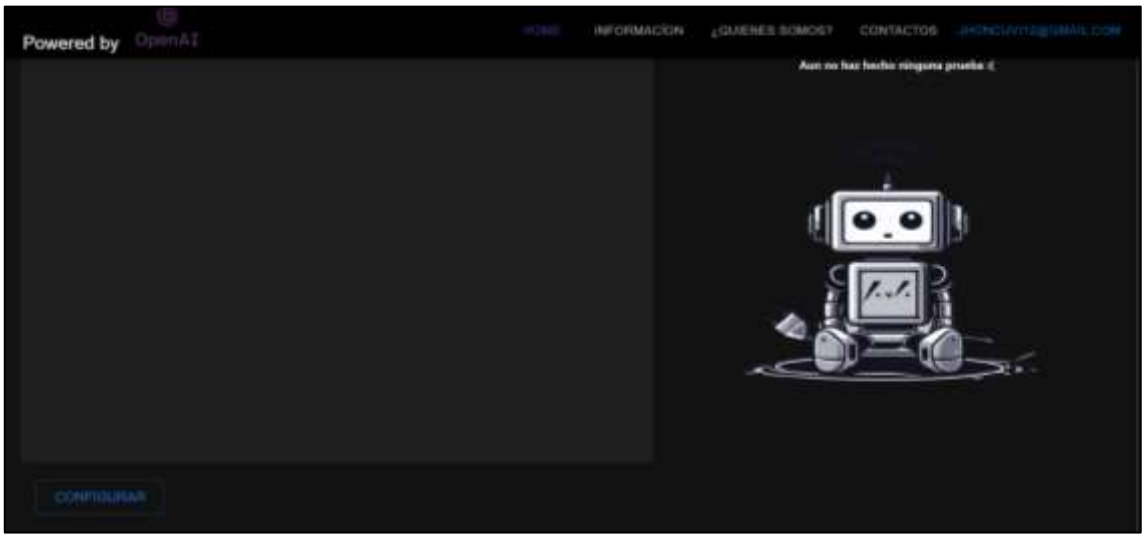


Figura 3: Interfaz principal de usuario registrado

Realizado por: Cuvi J., 2023

- **Pantalla principal de usuario No registrado**

En la **Figura 6 s**e muestra la interfaz de un usuario no registrado en el sitio web contando con ciertas limitaciones.

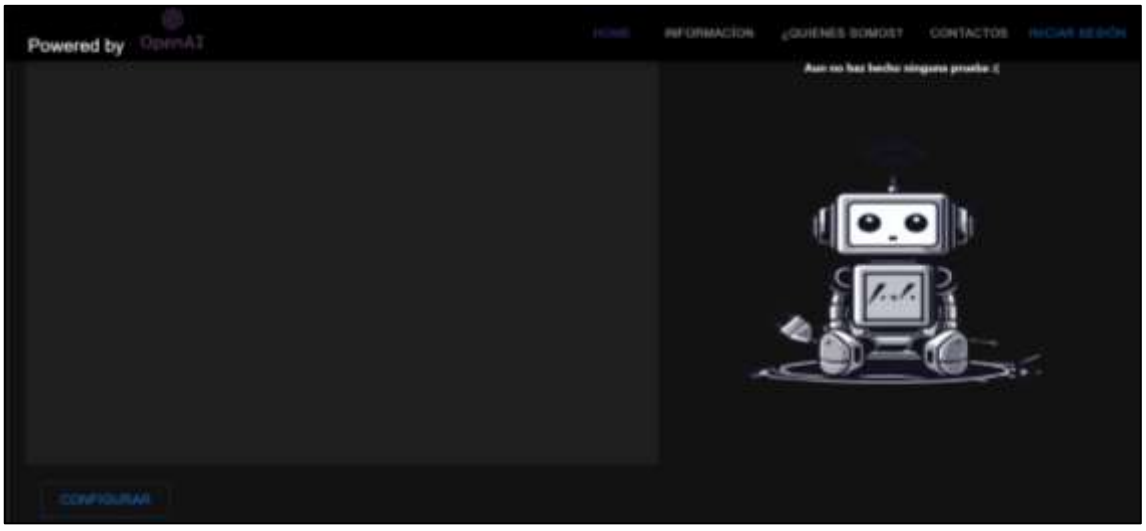


Figura 4: Interfaz principal de usuario no registrado

Realizado por: Cuvi J., 2023

- **Pantalla principal de ingreso de código**

En la **Figura 7** se muestra la vista del sitio web cuando configura el ingreso de código para la generación de pruebas unitarias posteriormente.

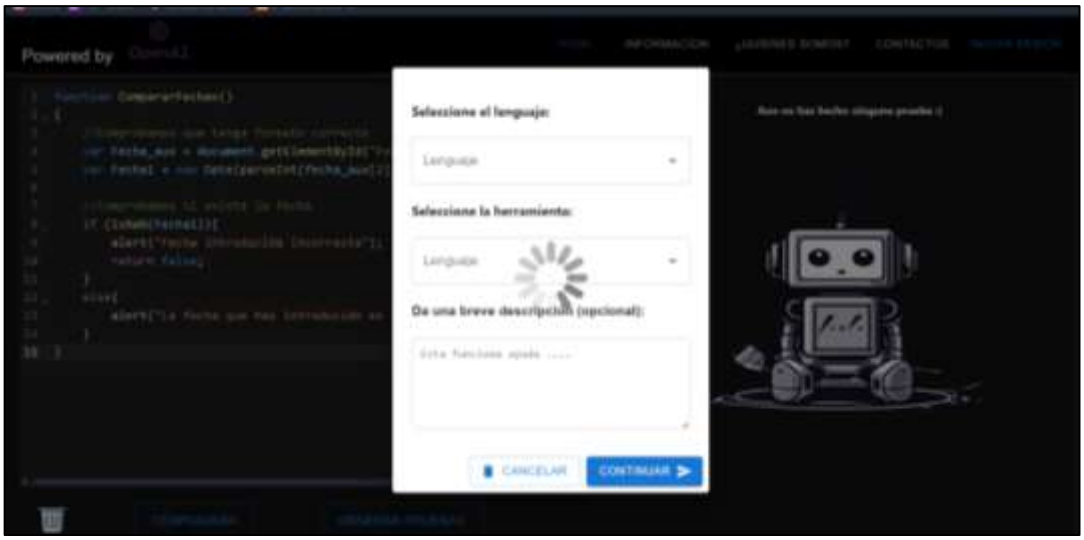


Figura 5: Interfaz de configuración de código

Realizado por: Cuvi J., 2023

- **Pantalla de presentación de pruebas unitarias generadas**

En la **Figura 8** se muestra las pruebas unitarias generadas por parte del sitio web contando con acciones.

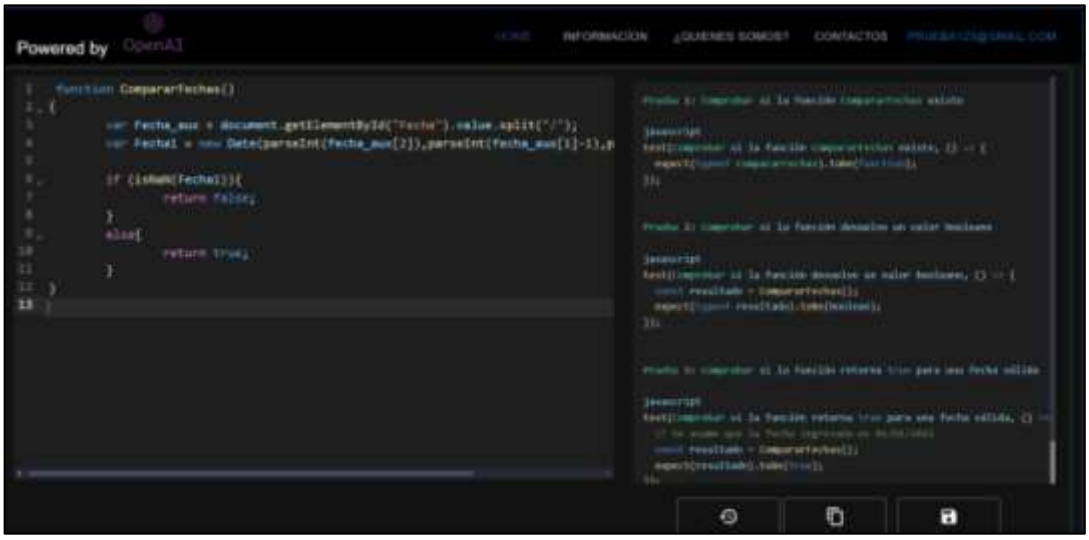


Figura 6: Interfaz de pruebas generadas

Realizado por: Cuvi J., 2023

- Pantalla de historial de pruebas

En la **Figura 9** se muestra el apartado del historial de pruebas guardadas por parte de un usuario registrado en el sitio web.

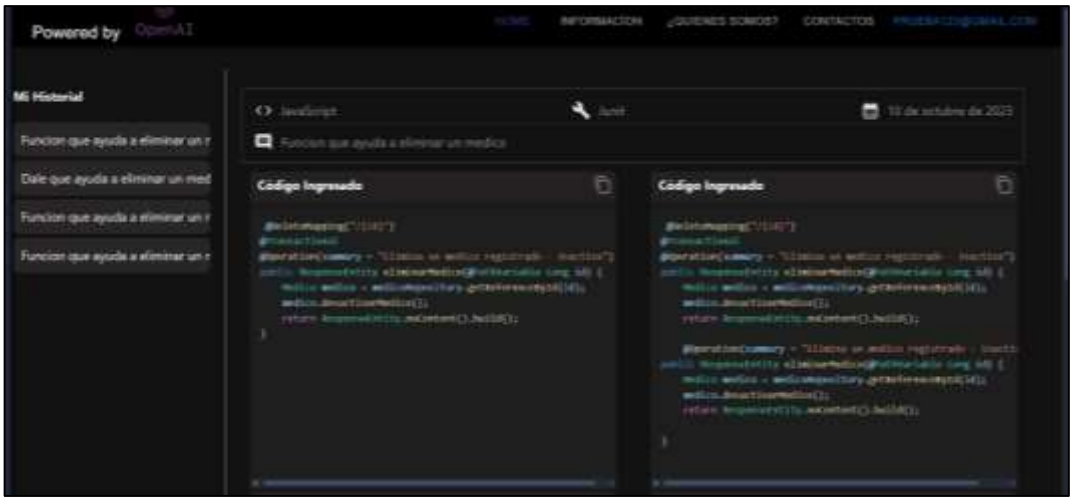


Figura 7: Interfaz de historial de pruebas guardadas

Realizado por: Cuvi J., 2023

ANEXO F: DESARROLLO DE PRUEBAS

Tabla 1: Caso de prueba CP-01

Caso de prueba	CP-01	**RF asociado**	RF01	**Estado**	Aprobado
Descripción	Ingreso de usuario no registrado				
Datos de entrada	Correo no existente Contraseña válida				
Resultado esperado	Mensaje de alerta de error al iniciar sesión				
Resultado obtenido					
Responsable	John Cuvi		**Fecha**	14/11/2023	

Realizado por: Cuvi J., 2023

Tabla 2: Caso de prueba CP-02

Caso de prueba	CP-02	**RF asociado**	RF01	**Estado**	Aprobado
Descripción	Ingreso de usuario con una contraseña no válida				
Datos de entrada	Correo existente Contraseña inválida				
Resultado esperado	Mensaje de alerta de error al iniciar sesión				
Resultado obtenido					
Responsable	John Cuvi		**Fecha**	14/11/2023	

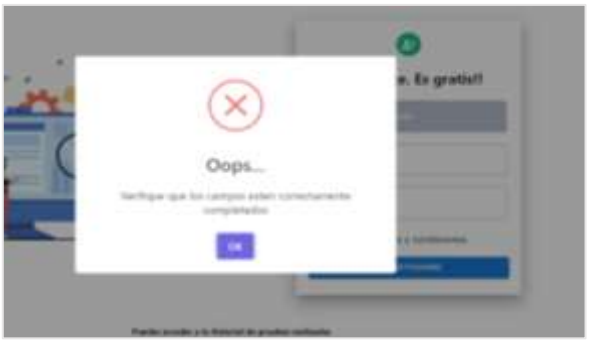
Realizado por: Cuvi J., 2023

Tabla 3: Caso de prueba CP-03

Caso de prueba	CP-03	**RF asociado**	RF01	**Estado**	Aprobado
Descripción	Ingreso de usuario con credenciales válidas				
Datos de entrada	Correo válido Contraseña válida				
Resultado esperado	Mensaje de bienvenida y cambio de pantalla.				
Resultado obtenido					
Responsable	John Cuvi		**Fecha**		14/11/2023

Realizado por: Cuvi J., 2023

Tabla 4: Caso de prueba CP-04

Caso de prueba	CP-04	**RF asociado**	RF04	**Estado**	Aprobado
Descripción	Crear una cuenta con un correo ya registrado				
Datos de entrada	Correo existente Contraseña válida				
Resultado esperado	Mensaje de alerta de error al crear la cuenta				
Resultado obtenido					
Responsable	John Cuvi		**Fecha**		15/11/2023

Realizado por: Cuvi J., 2023

Tabla 5: Caso de prueba CP-05

Caso de prueba	CP-05	**RF asociado**	RF02	**Estado**	Aprobado
Descripción	Creación de cuenta de un usuario nuevo con credenciales inválidas.				
Datos de entrada	Correo válido Contraseña inválida Contraseña de confirmación no válida				
Resultado esperado	Alerta de aviso que la contraseña es inválida por la cantidad de caracteres mínimas.				
Resultado obtenido					
Responsable	John Cuvi		**Fecha**	15/11/2023	

Realizado por: Cuvi J., 2023

Tabla 6: Caso de prueba CP-06

Caso de prueba	CP-06	**RF asociado**	RF02	**Estado**	Aprobado
Descripción	Creación de cuenta de un usuario con credenciales válidas				
Datos de entrada	Correo válido Contraseña válida				
Resultado esperado	Mensaje de bienvenida Redirección a la página principal				
Resultado obtenido					
Responsable	John Cuvi		**Fecha**	15/11/2023	

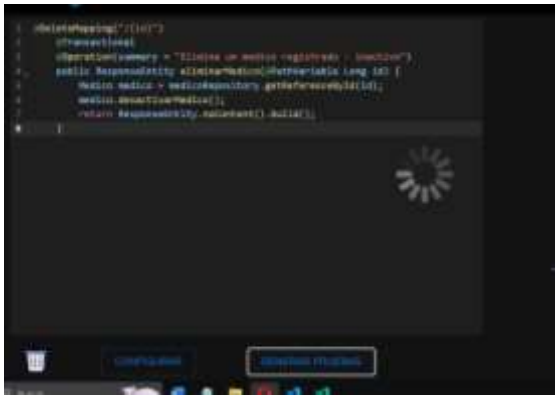
Realizado por: Cuvi J., 2023

Tabla 7: Caso de prueba CP-07

Caso de prueba	CP-07	**RF asociado**	RF03	**Estado**	Aprobado
Descripción	Mala configuración de la función, no se ingresa el lenguaje de programación a de la función.				
Datos de entrada	Lenguaje de programación inválido.				
Resultado esperado	No se debe habilitar el botón de “Continuar”				
Resultado obtenido					
Responsable	John Cuvi		**Fecha**	15/11/2023	

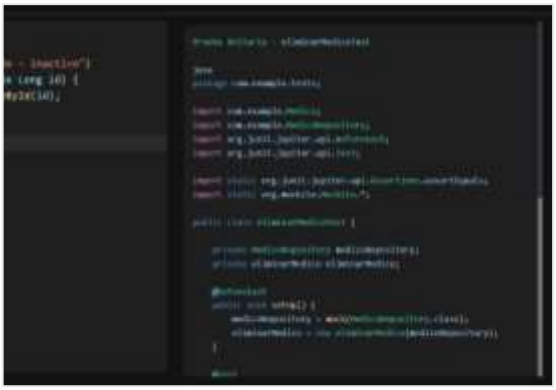
Realizado por: Cuvi J., 2023

Tabla 8: Caso de prueba CP-08

Caso de prueba	CP-08	**RF asociado**	RF03	**Estado**	Aprobado
Descripción	Ingreso de campos válidos				
Datos de entrada	Lenguaje válido, herramienta válida, descripción.				
Resultado esperado	Habilitación del botón de pruebas				
Resultado obtenido					
Responsable	John Cuvi		**Fecha**	15/11/2023	

Realizado por: Cuvi J., 2023

Tabla 9: Caso de prueba CP-09

Caso de prueba	CP-09	**RF asociado**	RF04	**Estado**	Aprobado
Descripción	Desarrollo de la generación de pruebas haciendo click en el botón de "Generar"				
Datos de entrada	N/A				
Resultado esperado	Visualización de leader y posteriormente de un tiempo la visualización de pruebas generadas.				
Resultado obtenido					
Responsable	John Cuvi		**Fecha**		15/11/2023

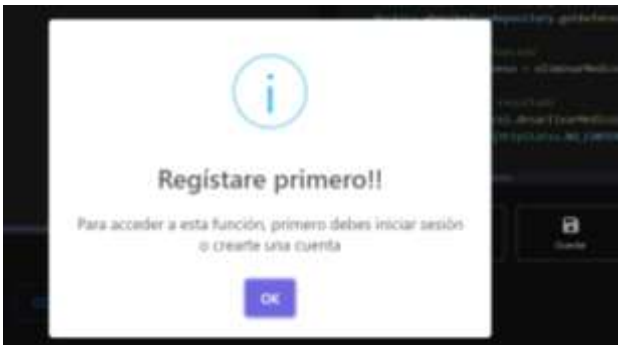
Realizado por: Cuvi J., 2023

Tabla 10: Caso de prueba CP-10

Caso de prueba	CP-10	**RF asociado**	RF05	**Estado**	Aprobado
Descripción	Guardar en el historial las pruebas generadas haciendo click en el botón de "Guardar"				
Datos de entrada	Pruebas unitarias generadas.				
Resultado esperado	Mostrar una notificación indicando que las pruebas han sido guardadas en el historial.				
Resultado obtenido					
Responsable	John Cuvi		**Fecha**		15/11/2023

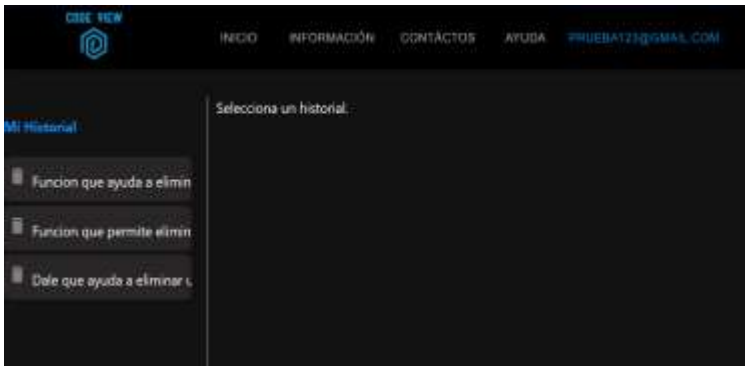
Realizado por: Cuvi J., 2023

Tabla 11: Caso de prueba CP-11

Caso de prueba	CP-11	**RF asociado**	RF05	**Estado**	Aprobado
Descripción	Guardar en el historial las pruebas generas por parte de un usuario no registrado				
Datos de entrada	Pruebas unitarias generadas				
Resultado esperado	Mostrar una alerta indicando que primero debe iniciar sesión o crearse una cuenta				
Resultado obtenido					
Responsable	John Cuvi		**Fecha**		16/11/2023

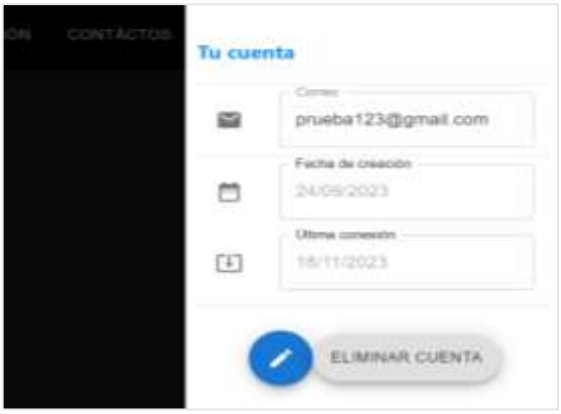
Realizado por: Cuvi J., 2023

Tabla 12: Caso de prueba CP-12

Caso de prueba	CP-12	**RF asociado**	RF06	**Estado**	Aprobado
Descripción	Ver el historial de pruebas guardadas haciendo click en el perfil seleccionando la opción de" Historial".				
Datos de entrada	N/A				
Resultado esperado	Redirección a la página de historial de pruebas visualizando el menú de pruebas generadas.				
Resultado obtenido					
Responsable	John Cuvi		**Fecha**		16/11/2023

Realizado por: Cuvi J., 2023

Tabla 13: Caso de prueba CP-13

Caso de prueba	CP-13	**RF asociado**	RF07	**Estado**	Aprobado
Descripción	Visualización de la información de la cuenta seleccionando la opción del "Perfil" por parte de un usuario con correo y contraseña.				
Datos de entrada	N/A				
Resultado esperado	Visualización de componente con la información básica del usuario.				
Resultado obtenido					
Responsable	John Cuvi		**Fecha**		16/11/2023

Realizado por: Cuvi J., 2023

Tabla 14: Caso de prueba CP-14

Caso de prueba	CP-14	**RF asociado**	RF08	**Estado**	Aprobado
Descripción	Eliminar una cuenta un usuario registrado seleccionando la opción del "Eliminar Cuenta".				
Datos de entrada	N/A				
Resultado esperado	Mostrar un mensaje de alerta para confirmar la acción.				
Resultado obtenido					
Responsable	John Cuvi		**Fecha**		16/11/2023

Realizado por: Cuvi J., 2023

Tabla 15: Caso de prueba CP-15

Caso de prueba	CP-15	**RF asociado**	RF09	**Estado**	Aprobado
Descripción	Recuperación de contraseña por parte de un usuario con correo inexistente.				
Datos de entrada	Correo inexistente.				
Resultado esperado	Mostrar una alerta indicando que la acción ha fallado.				
Resultado obtenido					
Responsable	John Cuvi		**Fecha**		16/11/2023

Realizado por: Cuvi J., 2023

Tabla 16: Caso de prueba CP-16

Caso de prueba	CP-16	**RF asociado**	RF09	**Estado**	Aprobado
Descripción	Recuperación de contraseña por parte de un usuario registrado en Firestore.				
Datos de entrada	Correo válido.				
Resultado esperado	Mostrar una alerta indicando que se ha enviado a su correo los siguientes pasos para recuperar la contraseña.				
Resultado obtenido					
Responsable	John Cuvi		**Fecha**		17/11/2023

Realizado por: Cuvi J., 2023

Tabla 17: Caso de prueba CP-17

Caso de prueba	CP-17	**RF asociado**	RF10	**Estado**	Aprobado
Descripción	Cambio de contraseña ingresando una contraseña corta.				
Datos de entrada	Contraseña con longitud menor a 8 caracteres,				
Resultado esperado	No se debe habilitar el botón de “Actualizar datos”				
Resultado obtenido					
Responsable	John Cuvi		**Fecha**	17/11/2023	

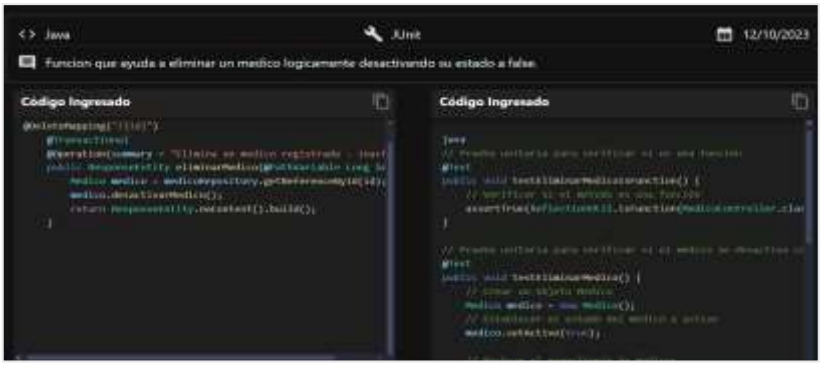
Realizado por: Cuvi J., 2023

Tabla 18: Caso de prueba CP-18

Caso de prueba	CP-18	**RF asociado**	RF10	**Estado**	Aprobado
Descripción	Cambio de contraseña				
Datos de entrada	Contraseña válida				
Resultado esperado	Se debe habilitar el botón de “Actualizar datos” y posteriormente una notificación del estado de la acción siendo redirigido a la página de inicio de inicio se sesión posteriormente.				
Resultado obtenido					
Responsable	John Cuvi		**Fecha**	17/11/2023	

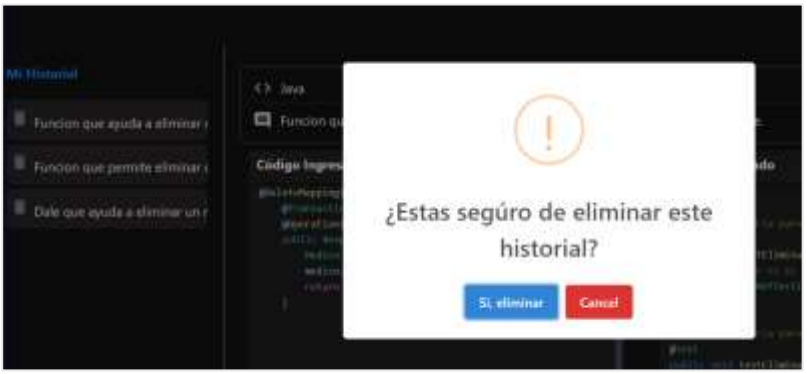
Realizado por: Cuvi J., 2023

Tabla 19: Caso de prueba CP-19

Caso de prueba	CP-19	**RF asociado**	RF11	**Estado**	Aprobado
Descripción	Seleccionar un historial haciendo click en una de las opciones del menú de historial disponibles.				
Datos de entrada	N/A				
Resultado esperado	Mostrar los datos de ese historial siendo: Lenguaje, Herramienta, fecha, código ingresado y pruebas generadas.				
Resultado obtenido					
Responsable	John Cuvi		**Fecha**		18/11/2023

Realizado por: Cuvi J., 2023

Tabla 20: Caso de prueba CP-20

Caso de prueba	CP-20	**RF asociado**	RF12	**Estado**	Aprobado
Descripción	Eliminar un historial haciendo click en el icono de borrar en uno de los títulos de menú de historial.				
Datos de entrada	N/A				
Resultado esperado	Mostrar una alerta de confirmación y si esta es confirmada se debe mostrar una notificación indicando que la acción ha sido finalizada con éxito.				
Resultado obtenido					
Responsable	John Cuvi		**Fecha**		18/11/2023

Realizado por: Cuvi J., 2023

ANEXO G: PROCESO PARA EL DESARROLLO DE PRUEBAS UNITARIAS CON ENFOQUE MANUAL Y AUTOMATIZADO.

Preparación de entorno: Como primer paso para la realización de pruebas unitarias, se debe considerar dos aspectos fundamentales detalladas a continuación:

- *Seleccionar la herramienta de pruebas:* Para el desarrollo de pruebas en este ejemplo se ha optado instalar la herramienta JEST misma que es una herramienta especializada en testing.
- *Carpeta de pruebas:* Creamos una carpeta denominada Test donde se deberá crear los archivos de pruebas que deben tener la extensión “test.js” para que puedan ser ejecutadas.

Proceso para la realización pruebas unitarias con enfoque tradicional

1. Selección de la función a evaluar

En este apartado se recibe una función específica, en nuestro caso, tomemos como ejemplo la función " verificarAnio ()" del proyecto RentaCar. Esta función se encarga de verificar el año de fabricación del vehiculo sea mayor a 2000 y sea menor al año actual retornando un valor booleano. Función:

```
export const verificarAnio = (anio) => {

    const anioActual = new Date().getFullYear();
    const anioIngresado = parseInt(anio);

    if (!Number.isInteger(anioIngresado) || anioIngresado < 2000 ||
anioIngresado > anioActual) {
        return false;
    }

    return true;
}
```

2. Desarrollo manual de pruebas unitarias usando JEST

Descripción del Escenario: El escenario consiste en verificar el correcto funcionamiento de la función 'verificarAnio()' cuando se le proporciona un año dentro del rango válido, es decir, mayor a 2000 y menor o igual al año actual.

Datos de Entrada:

- Se selecciona un año de prueba dentro del rango válido. Por ejemplo, si el año actual es 2024, se elige el año 2023.

Paso a Paso del Desarrollo Manual con JEST:

- Se escribe un script de prueba utilizando JEST.
- Se considera nuestro escenario pensado para esta prueba
- Se ingresan los datos de entrada planteados

```
import { verificarAnio } from './verificarAnio';

describe('verificarAnio()', () => {
    test('Debería retornar true para un año dentro del rango válido', ()
=> {
        // Año de prueba dentro del rango válido
        const anioPrueba = 2023;

        // Llamada a la función verificarAnio()
        const resultado = verificarAnio(anioPrueba);

        // Verificación del resultado
        expect(resultado).toBe(true);
    });
});
```

3. **Ejecución de la Prueba con JEST:**

- Se ejecuta manualmente el script de prueba utilizando JEST desde la línea de comandos: ***npm run test***

4. **Verificación de Resultados:**

- Se verifica la salida en la consola para determinar si la prueba ha pasado o ha fallado.

```
 PASS  src/test/pruebas.test.js

Test Suites: 1 failed, 1 passed, 2 total
Tests:       1 passed, 1 total
Snapshots:   0 total
Time:        1.73 s, estimated 2 s
Ran all test suites.
PS D:\Aplicaciones\RENTACAR\Api-Sequelize>
```

Figura 1: Ejecución de la prueba manual

Realizado por: Cuvi J., 2024

En la **Figura 1** se muestra el resultado de la ejecución de la prueba manual desarrollada.

5. **Documentación de Resultados:**

- Se documenta el todo el proceso realizado en la ficha propuesta mostrada a continuación:

Prueba Manual a la primera función			
Autor	John Cuvi	**Fecha:**	27/12/2023

Id caso	1	**Nombre de la Prueba**	Verificar Año Correcto
Descripción del caso	El escenario consiste en verificar el correcto funcionamiento de la función 'verificarAnio()' cuando se le proporciona un año dentro del rango válido, es decir, mayor a 2000 y menor o igual al año actual.		
Escenario	Año valido = 2017	**Resultado esperado**	True
Lenguaje	JavaScript	**Herramienta**	JEST
Resultado Obtenido	true	**Tiempo de desarrollo**	4 minutos y 28 segundos

Realizado por: Cuvi J., 2024

Este proceso se repetiría para cada una de las 3 funciones proporcionadas.

Proceso para la realización pruebas unitarias con el sitio web

1. Selección de la función a evaluar

En este apartado se recibe una función específica, en este caso es la misma función proporcionada para el desarrollo con enfoque manual.

Función:

```
export const verificarAnio = (anio) => {

    const anioActual = new Date().getFullYear();
    const anioIngresado = parseInt(anio);

    if (!Number.isInteger(anioIngresado) || anioIngresado < 2000 ||
anioIngresado > anioActual) {
        return false;
    }

    return true;
}
```

2. Desarrollo de la prueba usando el sitio web

Se debe ingresar al sitio web alojado en la siguiente dirección: https://code-view.netlify.app

Se debe pegar el código a evaluar en casillero de prueba tal y como se muestra en **Figura 2**:

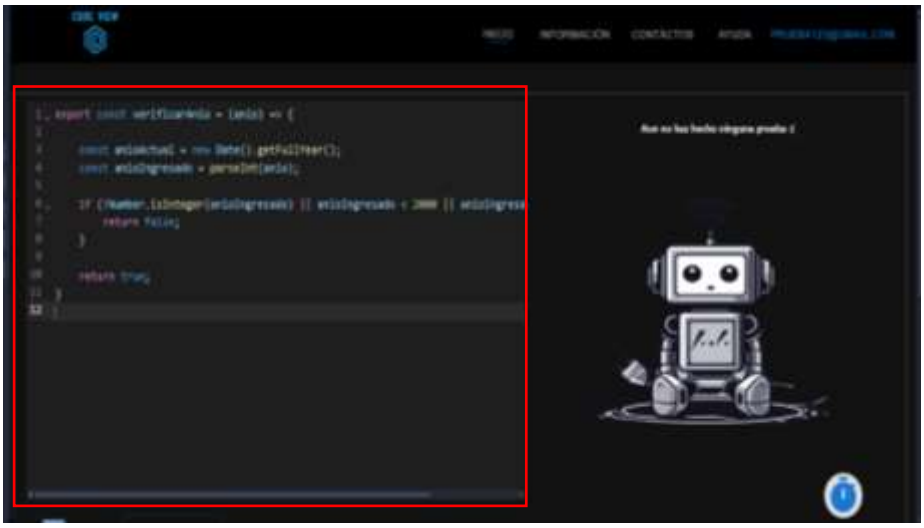


Figura 2: Ejecución de la prueba manual

Realizado por: Cuvi J., 2024

Configurar la prueba

Para generar pruebas unitarias con mayor precisión se debe configurar la prueba indicando el lenguaje en el que esta desarrollado, la herramienta para la cual se debe generar la prueba e ingresar un contexto sobre que es lo que realiza la función tal y como se muestra en **Figura 2**:

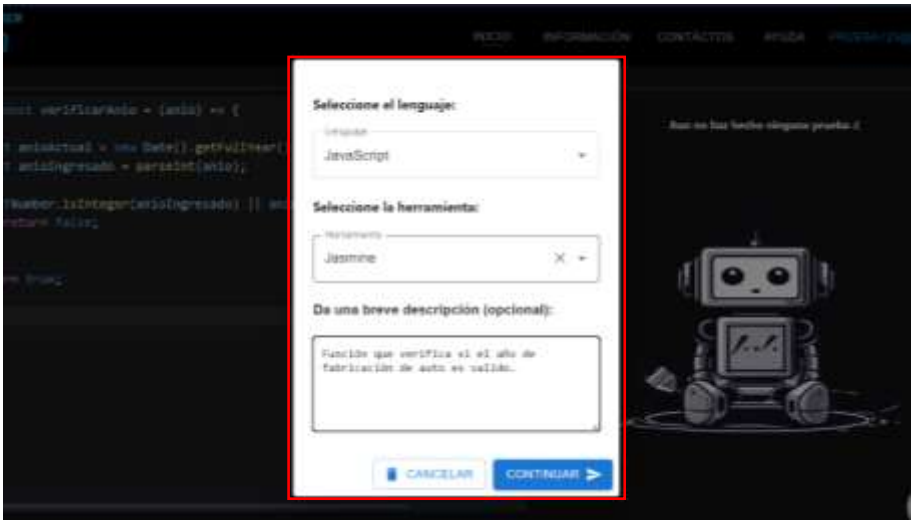


Figura 3: Configuración de la prueba

Realizado por: Cuvi J., 2024

Generar la prueba

Una vez configurada, se habilita el botón de Generar prueba misma que permite empezar la generación de pruebas unitarias tal y como se muestra en la **Figura 4**:

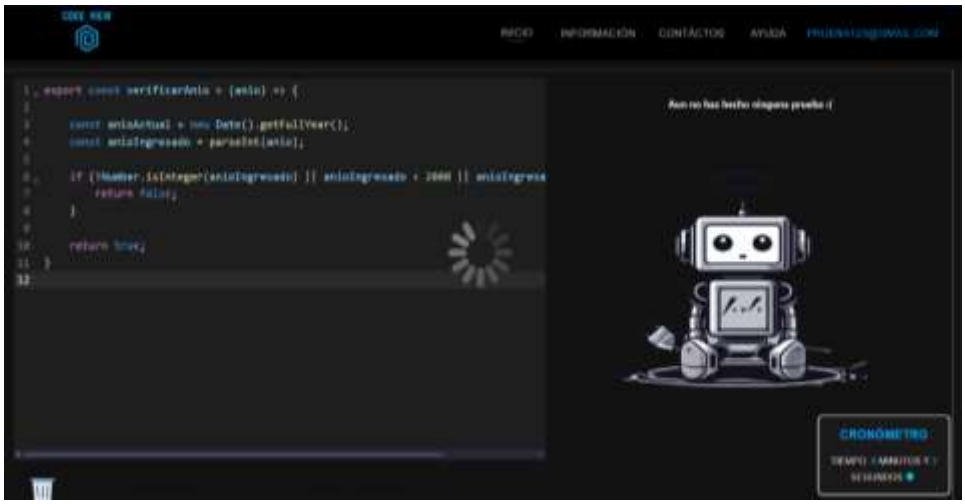


Figura 4: Generación de pruebas mediante el sitio web

Realizado por: Cuvi J., 2024

Salida de pruebas

El sitio web proporciona en promedio 3 pruebas misma que cada uno contiene la descripción de la prueba, el escenario propuesto, el resultado esperado y el fragmento de código de la prueba tla y como se muestra en la **Figura 5:**

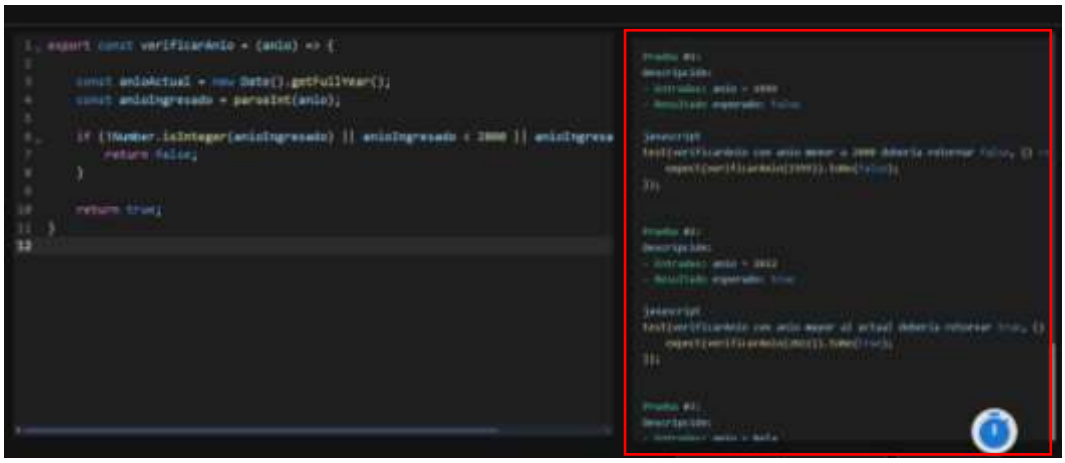


Figura 5: Salida de pruebas

Realizado por: Cuvi J., 2024

3. Ejecución de prueba

Una vez obtenidos las pruebas se debe copiar el fragmento de código proporcionado y vincularlo a un archivo de prueba para poder ejecutarlo mediante los comandos: ***npm run test.***

La **Figura 6** muestra el resultado de la ejecución de una prueba generado a través del sitio web indicándonos que la prueba paso el test de JEST,

Figura 6: Ejecución de la prueba manual

Realizado por: Cuvi J., 2024

4. **Documentación de Resultados:**

- Se documenta el todo el proceso realizado en la ficha propuesta mostrada a continuación:

Prueba Manual a la primera función				
Autor	John Cuvi		**Fecha:**	27/12/2023
Id caso	1	**Nombre de la Prueba**	Verificar Año Correcto	
Descripción del caso	Verificar que el la función devuelva un valor true con un año valido			
Escenario	2022		**Resultado esperado**	True
Lenguaje	JavaScript		**Herramienta**	JEST
Resultado Obtenido	true		**Tiempo de desarrollo**	43 segundos

Realizado por: Cuvi J., 2024

Este proceso se repetiría para cada una de las 3 funciones proporcionadas.

ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO

CERTIFICADO DE CUMPLIMIENTO DE LA GUÍA PARA NORMALIZACIÓN DE TRABAJOS DE FIN DE GRADO

Fecha de entrega: 16/ 04 / 2024

INFORMACIÓN DEL AUTOR
Nombres – Apellidos: JOHN VLADIMIR CUVI GUAMAN
INFORMACIÓN INSTITUCIONAL
Facultad: INFORMÁTICA Y ELECTRÓNICA
Carrera: SOFTWARE
Título a optar: INGENIERO DE SOFTWARE
Ing. Miguel Duque Vaca **Director del Trabajo de Integración Curricular** **Ing. Gloria Arcos Medina** **Asesor del Trabajo de Integración Curricular**