INSTITUTO TECNOLOGICO SUPERIOR DE TANTOYUCA

Tesis

“Pruebas de software para la detección de anomalías y la verificación del correcto funcionamiento de una Plataforma Web”

Presenta

Ernesto Carlos Domínguez Alonso

Para obtener el grado de

Ingeniería en sistemas computacionales

Carrera:

Ingeniería En Sistemas Computacionales

Director de tesis:

M.C.C Manuel Hernández Hernández

No de Control:

183S0412

Tantoyuca, Ver, noviembre de 2023

Índice general

Índice de figuras

Índice de tablas

Resumen

Las pruebas de software es un proceso que consiste en evaluar el buen funcionamiento de la plataforma web en el servidor y tener una correcta instalación y configuración para los certificados digitales. Se llevó a cabo una integración de análisis de los antecedentes relacionados con las pruebas efectuadas en el enfoque de una plataforma web. Esta tesis tuvo como objetivo primordial la categorización de cada una de estas pruebas, con el propósito de profundizar en la comprensión de la materia investigada. En el transcurso de esta tesis, se procedió a describir un método detallado que permitiera la ejecución de cada prueba de manera eficiente y efectiva. Este enfoque metodológico se diseñó con la finalidad de llevar a cabo todas las pruebas requeridas de manera sistemática y organizada. Además, se persiguió la búsqueda y formulación de un método integral que abarcara la totalidad de pruebas existentes, lo que, en última instancia, posibilitaría su ejecución de manera unificada y coherente.

Este enfoque metodológico resulta esencial en el contexto de la investigación, ya que permite optimizar los recursos y asegurar una evaluación completa y precisa de la plataforma web en cuestión, lo cual contribuye al fortalecimiento de la comprensión de los procesos de prueba y a la obtención de resultados más coherentes y confiables.

La metodología utilizada para llevar a cabo las pruebas de software en una página web es un aspecto crucial que permite realizar un proceso estructurado y eficiente. A continuación, en los siguientes puntos, se menciona una metodología comúnmente utilizada para las pruebas de software en una página web.

- Planificación
- Preparación del entorno de pruebas
- Ejecución de pruebas

- Análisis de resultados
- Corrección de errores

En virtud de lo expuesto previamente, se emplearon distintos programas informáticos, tales como Visual Studio Code, Bloc de Notas y bibliotecas específicas para pruebas unitarias, así como *Gatling, Jira* y *QMetry*. En relación a las diversas evaluaciones llevadas a cabo y con el propósito de subsanar las deficiencias identificadas, se procedió a realizar las respectivas pruebas y corregir los errores encontrados. Una vez solucionados los errores y se hayan verificado las correcciones, se finalizan las pruebas y se realiza una revisión final de la página web. Se genera un informe final que resume las pruebas realizadas como (Pruebas unitarias, Pruebas de integración, Pruebas funcionales Pruebas de extremo a extremo, Pruebas de aceptación, Pruebas de rendimiento, Pruebas de humo), los resultados obtenidos y las acciones tomadas como medio para distinguir claramente entre medidas de tiempo en segundos y minutos, se utilizaron gráficas y tablas como recursos visuales y tabulares, respectivamente.

Abstract

Software testing is a process that consists of evaluating the proper functioning of the web platform on the server and having a correct installation and configuration for digital certificates. An integration of analysis of the background related to the tests carried out was carried out in the approach of a web platform. This thesis had as its primary objective the categorization of each of these tests, with the purpose of deepening the understanding of the subject investigated. In the course of this thesis, a detailed method was described that allows the execution of each test efficiently and effectively. This methodological approach was designed with the purpose of carrying out all the required tests in a systematic and organized manner. In addition, the search and formulation of a comprehensive method that covered all existing tests was pursued, which, ultimately, would enable its execution in a unified and coherent manner.

This methodological approach is essential in the context of research, since it allows optimizing resources and ensuring a complete and accurate evaluation of the web platform in question, which contributes to strengthening the understanding of testing processes and obtaining results. more coherent and reliable.

The methodology used to carry out software testing on a web page is a crucial aspect that allows for a structured and efficient process. Below, in the following points, a commonly used methodology for software testing on a web page is mentioned.

- Planning
- Preparation of the test environment
- Testing execution
- Analysis of results
- Error correction

By virtue of what was previously stated, different computer programs were used, such as Visual Studio Code, Notepad and specific libraries for unit testing, as well as Gatling, Jira and QMetry. In relation to the various evaluations carried out and with the purpose of correcting the deficiencies identified, the respective tests were carried out and the errors found were corrected. Once the errors have been resolved and the corrections have been verified, the tests are completed and a final review of the website is carried out. A final report is generated that summarizes the tests performed such as (Unit Tests, Integration Tests, Functional Tests End-to-End Tests, Acceptance Tests, Performance Tests, Smoke Tests), the results obtained and the actions taken as a means to clearly distinguish between time measurements in seconds and minutes, graphs and tables were used as visual and tabular resources, respectively.

Nomenclatura

1. ANSI: Instituto Nacional Estadounidense de Estándares
2. AUT: autorización de uso terapeutico
3. CTS: Central Standard Time Zone
4. CP: Category Partition
5. CSS: lenguaje informático que especifica cómo se presentan los documentos a los usuarios
6. DOM: Document Object Model
7. FEU: Forty Equivalent Unit
8. GAO: Grado de Apalancamiento Operativo
9. HTML: lenguaje de marcado que se utiliza para el desarrollo de páginas de Internet.
10. IDE: entorno de desarrollo integrado
11. IEEE: Instituto de Ingenieros Eléctricos y Electrónicos
12. JSE: Jefe de servicios Escolares
13. ISO: Organización Internacional de Normalización
14. JSON: notación de objetos de JavaScript
15. LIPS: Laboratorio Industrial de Pruebas de Software
16. LMP: límites máximos permisibles
17. PLUTO: Producto Lines Use case Test Optimización
18. RBC: Rehabilitación Basada en la Comunidad
19. RUP: Proceso Racional Unificado
20. SGBD: sistema gestor de base de datos
21. TDE: Cifrado de Datos transparente

22. UML: una meta clase abstracta que sirve para clasificar los elementos de un lenguaje de modelado bajo un concepto común

CAPÍTULO 1: MARCO CONCEPTUAL

1.1Antecedentes

Categoría 1: Pruebas de Calidad del Sitio Web "Allison"

En esta categoría "Pruebas de Calidad del Sitio Web 'Allison'" se centra en la evaluación exhaustiva y el aseguramiento de la calidad de un sitio web específico denominado "Allison". Este enfoque se basa en la necesidad de garantizar que dicho sitio web cumpla con los estándares y criterios de calidad establecidos en el contexto de desarrollo web. Las pruebas de calidad abordan aspectos como la funcionalidad, la usabilidad, el rendimiento y la seguridad del sitio web "Allison". Este proceso implica la aplicación de metodologías y técnicas de prueba específicas destinadas a identificar posibles defectos y áreas de mejora en el funcionamiento y la experiencia del usuario en el sitio web. La categoría abarca un conjunto de actividades orientadas a la optimización y la validación del rendimiento del sitio web "Allison", contribuyendo así a su mejora continua y su alineación con los estándares de calidad requeridos en el ámbito del desarrollo web.

Pruebas de calidad aplicadas al sitio web allison

La necesidad imperante de asegurar la calidad del software ha estimulado la concepción de diversas metodologías orientadas al desarrollo y la ejecución de pruebas de software. No obstante, numerosas empresas confrontan desafíos considerables en cuanto a la adecuada incorporación de una metodología de pruebas durante el ciclo de desarrollo del software (Camargo et al., 2013). Esto se debe, en gran medida, a que, en la realidad práctica, la definición de los pasos requeridos para optimizar y supervisar las distintas fases de un proceso de pruebas de software y la secuencia en que estas etapas deben llevarse a cabo se revela, en términos generales, como una tarea de considerable complejidad (óscar Paul, 2017).

Categoría 2: Procesos de Pruebas de Calidad de Software

En esta categoría, "Procesos de Pruebas de Calidad de Software" se enfoca en el estudio y análisis de los procedimientos y metodologías utilizados para evaluar la calidad de software en diversos contextos. Este enfoque abarca aspectos relacionados con la planificación, diseño, ejecución y seguimiento de pruebas de software con el propósito de identificar y corregir posibles defectos y asegurar el cumplimiento de los requisitos de calidad especificados. Los procesos de pruebas de calidad de software se consideran fundamentales en el ciclo de desarrollo de software, y su adecuada implementación es esencial para garantizar la fiabilidad, la eficiencia y la seguridad de las aplicaciones informáticas. En esta categoría, se exploran en detalle las diferentes etapas y enfoques involucrados en la ejecución de pruebas de calidad, así como los desafíos y mejores prácticas asociados con esta disciplina.

Análisis del proceso de pruebas de calidad de software

En este artículo, se enfoca en la línea de investigación de Ingeniería de Software, con un énfasis particular en el proceso de pruebas de calidad de software. El estado del arte abordado se extiende a nivel global, nacional y local, destacando la relevancia de la implementación adecuada de este proceso en las empresas de desarrollo de software. Se argumenta que, en la actualidad, la tecnología y la electrónica, en especial las computadoras, desempeñan un papel crítico en numerosas aplicaciones que impactan la vida cotidiana. Ejemplos concretos incluyen transacciones electrónicas, operaciones en la bolsa de valores, telemedicina y el transporte aéreo. En este contexto, se señala que los fallos en el software pueden tener consecuencias significativas en la vida de las personas, como se evidencia en casos como el fracaso del lanzamiento del cohete Ariane 5, los problemas con los misiles *Patriot* durante la Guerra del Golfo y el incidente de la máquina *Therac*-25 en terapia de radiación. A nivel internacional, se mencionan ejemplos de

instituciones y trabajos de investigación que abordan la temática de pruebas de calidad de software, incluyendo la implementación de laboratorios especializados en Argentina, investigaciones académicas en Uruguay y España, y la inclusión de la temática en programas académicos en Estados Unidos. El artículo resalta la necesidad de una mayor atención a la calidad del software y a los procesos de pruebas, particularmente en contextos críticos como la medicina. Se subraya la importancia de mitigar riesgos a través de un proceso adecuado de pruebas de calidad de software, en un momento en que el software desempeña un papel fundamental en actividades críticas para la sociedad (J. A. Mera-Paz, oct. 2016).

Proceso de planificación de pruebas de software apoyado en una herramienta basada en casos de uso

El aumento de la complejidad de los sistemas de software ha subrayado la importancia de garantizar su calidad. En este contexto, la detección de defectos durante la fase de pruebas se basa en identificar discrepancias entre la salida real y la salida esperada según las especificaciones, considerándolas como síntomas de problemas en el software. Las técnicas existentes para diseñar casos de prueba se apoyan en estas especificaciones para definir el comportamiento de la aplicación. La práctica de iniciar la planificación y diseño de pruebas en paralelo con la especificación se recomienda encarecidamente para mejorar la calidad, la productividad y la mitigación de riesgos. El enfoque propuesto se divide en dos partes: en la primera, se extienden los casos de uso de UML mediante la inclusión de contratos que comprenden pre y postcondiciones, así como casos de uso parametrizados. La segunda parte se centra en la generación automática de casos de prueba a partir de estos casos de uso extendidos. Se demuestra la aplicabilidad de este enfoque en el contexto de familias de productos, resultando en un modelo

de casos de uso extendido con contratos y un conjunto de casos de prueba que verifican la implementación del modelo. Además, se presenta la propuesta *Product Lines Use case Test Optimization* (PLUTO), que se enfoca en la generación de casos de prueba para familias de productos. PLUTO utiliza casos de uso para familias de productos como punto de partida y extiende esta notación mediante plantillas y lenguaje natural para expresar elementos comunes y puntos de variabilidad en los productos. El resultado incluye casos de prueba comunes para toda la familia de productos y casos de prueba específicos para cada uno. Estos casos de prueba se describen mediante plantillas en lenguaje natural, aunque su ejecución sobre el sistema requiere refinamiento adicional. Por otro lado, se menciona un enfoque basado en modelos para mejorar las pruebas de sistemas de aplicaciones interactivas. Este enfoque se fundamenta en la generación de casos de prueba desarrollada por Siemens y se aplica a casos de uso documentados en lenguaje natural. La conclusión de esta propuesta es la obtención de un conjunto de pruebas ejecutables a través de la interfaz gráfica del sistema. Estas pruebas son utilitarias para interactuar con la interfaz gráfica mediante instrucciones almacenadas en un script de prueba. En conjunto, estas propuestas contribuyen a mejorar la planificación y diseño de pruebas de software, permitiendo una mayor eficiencia, calidad y capacidad de adaptación a contextos específicos, incluyendo familias de productos y sistemas interactivos (Francisco Javier, julio del 2007).

Metodología para testing de software basado en componentes

El proyecto se enfoca en el desarrollo de una nueva metodología para la realización de pruebas de software basado en componentes en aplicaciones de software en desarrollo o ya desarrolladas, con un énfasis particular en la funcionalidad del producto. Esta metodología se

diferencia al centrarse en las pruebas de reutilización e integración de componentes, aspectos críticos en el desarrollo basado en componentes y, al mismo tiempo, la mayor ventaja y el mayor riesgo de este enfoque en comparación con las metodologías de pruebas tradicionales. La aplicación de esta nueva metodología se prevé que permita a las empresas desarrolladoras mejorar y optimizar sus procesos de pruebas de manera eficiente y efectiva cuando el desarrollo basado en componentes sea la opción elegida para un proyecto de desarrollo de software. El contexto histórico del desarrollo de software se remonta a sus primeros días, cuando las aplicaciones se desarrollaban de manera empírica más que metódica. La realización de pruebas de funcionalidad se llevaba a cabo durante el proceso de desarrollo, y esta actividad se consideraba parte integral de dicho proceso, sin distinción clara entre desarrollo y pruebas. A partir de la década de 1950, se comenzó a separar más claramente el proceso de desarrollo y las pruebas, y se reconoció la importancia de realizar pruebas de manera más estructurada. Durante la década de 1960, las pruebas ganaron relevancia, ya que se comprendió su impacto positivo en costos y tiempos de desarrollo. En la década de 1970, se introdujeron métodos de pruebas más rigurosos y se asignaron más recursos a esta actividad. El término "Ingeniería de Software" se acuñó durante este período, y las pruebas de software se convirtieron en una parte esencial del desarrollo de aplicaciones. En la década de 1980, la calidad se convirtió en el objetivo principal en el ámbito de las pruebas de software. Surgieron estándares de calidad, como IEEE, ANSI e ISO, y se desarrollaron metodologías de pruebas que no solo se centraron en las etapas finales de desarrollo, sino que también apoyaron todo el ciclo de vida del desarrollo de software. En resumen, a lo largo de las décadas, las metodologías de pruebas de software han evolucionado para adaptarse a los diversos métodos de desarrollo y se han convertido en una parte esencial de la búsqueda de la calidad en el producto final (Juan camilo, mayo 2010).

Proceso de automatización de pruebas de aplicaciones web desarrolladas con react, angular, ant y laravel

El proceso de pruebas en el ciclo de vida del desarrollo de software se destaca como una fase de alta prioridad debido a su papel fundamental en la garantía de la calidad del producto final. Sin embargo, a menudo esta fase se subestima o se pasa por alto, principalmente debido a desafíos económicos y de tiempo asociados con su implementación. Según Guillermo Talento, presidente ejecutivo de Software *Testing* Bureau, se ha observado que la automatización de pruebas puede generar ahorros significativos, reduciendo el tiempo dedicado a las pruebas manuales en un rango que oscila entre el 50 y el 70 por ciento. Este enfoque en la automatización de pruebas no solo ofrece beneficios en términos de eficiencia y ahorro de tiempo, sino que también mejora la calidad del proceso de prueba al reducir la posibilidad de errores humanos. Estas ventajas resaltan la importancia de considerar y priorizar adecuadamente las pruebas en el desarrollo de software, especialmente en un entorno donde la calidad y la eficiencia son factores críticos para el éxito del proyecto (Anon , 2014).

Categoría 3: Pruebas de Aplicaciones Web

En esta categoría de pruebas de aplicaciones web constituyen una categoría esencial en el campo de la ingeniería de software y la calidad de software. Se enfocan en evaluar y garantizar el rendimiento, la funcionalidad, la seguridad y la usabilidad de las aplicaciones web desarrolladas. Estas pruebas son críticas debido a la creciente dependencia de las aplicaciones web en una variedad de industrias y sectores.

Prueba de aplicaciones web y documentación

Las pruebas de software son una parte esencial del ciclo de desarrollo, ya que desempeñan un papel fundamental en la verificación de la calidad y el correcto funcionamiento de una

aplicación. Como destacó Edsger Dijkstra, estas pruebas tienen la capacidad de identificar errores existentes en el software, pero no pueden garantizar la ausencia total de fallos. Existen diversos tipos de pruebas que se aplican en el desarrollo de software, y estos principios son igualmente aplicables a las aplicaciones web. Sin embargo, debido a las particularidades del entorno web, han surgido herramientas especializadas tanto para probar la funcionalidad como para evaluar la capacidad de carga y el rendimiento de las aplicaciones web. En el contexto de las aplicaciones web, además de las pruebas convencionales, es esencial verificar el funcionamiento de componentes críticos como el servidor web, el sistema de gestión de bases de datos (SGBD) y la infraestructura de red en su conjunto. Estas pruebas pueden incluir tanto la evaluación funcional como la realización de pruebas de carga para determinar la capacidad del sistema. Para los desarrolladores de aplicaciones web, algunas herramientas fundamentales incluyen aquellas que permiten la validación del código HTML/CSS transmitido al cliente, asegurando así la correcta representación de la interfaz de la aplicación. Además, es esencial contar con herramientas que permitan simular un gran número de clientes para evaluar el rendimiento y la capacidad de respuesta del sistema frente a una carga significativa (Víctor, 2019).

Prueba de aplicaciones web

La prueba de software es un procedimiento esencial que implica poner a prueba un programa con el propósito de identificar y rectificar errores. Este proceso abarca diversas dimensiones de calidad y se orienta a la detección de errores de diversa naturaleza que puedan surgir durante las pruebas. Cuando se trata de la prueba de aplicaciones web, el enfoque se centra en evaluar y mejorar aspectos relacionados con el contenido, la usabilidad, el rendimiento, la funcionalidad, la capacidad, la navegabilidad y la seguridad de la aplicación web. Este proceso es

crucial para garantizar que los usuarios finales estén satisfechos tanto con el contenido como con la funcionalidad del producto. Por lo tanto, la realización de pruebas en aplicaciones web se convierte en un paso importante para identificar y corregir la mayor cantidad posible de errores, contribuyendo así a una experiencia de usuario más positiva y confiable (Vanessa Daboin, octubre 24, 2017).

Estrategia de pruebas para aplicaciones web

Con la evolución tecnológica, se ha evidenciado que la eficiencia y seguridad de un producto informático no pueden depender únicamente del optimismo y la profesionalidad de sus desarrolladores. La garantía de un funcionamiento completo y sin problemas a lo largo de todo el ciclo de vida de una aplicación se logra mediante la aplicación de pruebas exhaustivas en todas sus categorías. Estas pruebas se diseñan estratégicamente para evaluar todos los aspectos relevantes del producto de acuerdo a sus características particulares, asegurando así su éxito y la satisfacción de los clientes. En este contexto, se llevarán a cabo pruebas en tres productos de gran importancia: el portal de la FEU de la FRG, la Wikipedia y el Museo de la Computación. Estas pruebas abarcarán todas las categorías necesarias y se realizarán con el apoyo de herramientas como Selenium IDE y pruebas de usabilidad. El objetivo principal de estas pruebas es identificar y corregir la mayor cantidad posible de errores que puedan afectar el funcionamiento adecuado de estos productos, contribuyendo de esta manera a su calidad y confiabilidad (Yadira Caridad, Manzanillo-Granma 2013).

Pruebas e implementación de la aplicación

El departamento de *"Testing"* asume la responsabilidad de diseñar, planificar y aplicar pruebas a los sistemas bajo la responsabilidad del PROVEEDOR, con el propósito de garantizar la calidad y seguridad de los productos proporcionados a los clientes. A pesar de que el equipo de

desarrollo no estuvo involucrado en esta etapa del proyecto del sistema LMP, se aplicaron diversas técnicas de prueba de software. Aunque estas técnicas no alcanzaron el nivel de las que utiliza el departamento de *"Testing",* permitieron la validación de la operación del software y la mejora continua de la aplicación (Juan Velásquez, 2015).

1.2DEFINICIÓN DEL PROBLEMA

El presente estudio se enfoca en el problema relacionado con las pruebas de software destinadas a la detección de anomalías y la verificación del correcto funcionamiento de una Plataforma Web. Este problema se centra en el desarrollo de estrategias y metodologías de pruebas que permitan identificar y corregir posibles errores o comportamientos inesperados en la plataforma web, con el objetivo de garantizar su adecuado funcionamiento de acuerdo a las especificaciones previamente establecidas. La cuestión central radica en la necesidad de establecer un proceso de pruebas sistemático y exhaustivo que asegure la calidad y la confiabilidad de la plataforma web antes de su implementación y puesta en producción.

1.3. JUSTIFICACIÓN

El desarrollo y lanzamiento de una plataforma web exitosa requiere una planificación y ejecución rigurosa, especialmente en lo que respecta a las pruebas de software. Las pruebas adecuadas desempeñan un papel crucial en la garantía de la calidad, la funcionalidad, la seguridad y la confiabilidad de una plataforma web. Por lo tanto, la implementación de un proyecto de generación de pruebas de software se vuelve esencial para asegurar el éxito y la satisfacción de los usuarios.

Dado lo anterior se realiza diferentes pruebas de software tales como:

- Mejorar la calidad

- Asegurar la funcionalidad
- Garantizar la seguridad
- Optimización del rendimiento
- Cumplimiento de estándares y requisitos
- Ahorro de costos a largo plazo

Mejora de la calidad: Las pruebas de software permiten identificar y solucionar errores y fallos en una página web antes de que lleguen a los usuarios finales. Al someter la página a una variedad de pruebas, como pruebas de funcionalidad, compatibilidad, rendimiento y usabilidad, se puede mejorar la calidad del producto final y reducir la posibilidad de experiencias negativas para los usuarios.

Asegurar la funcionalidad: Las pruebas de software permiten verificar que todas las funcionalidades de la página web, como enlaces, formularios, interacciones y características específicas, estén trabajando de manera correcta y esperada. Esto garantiza que los usuarios puedan interactuar y utilizar la página sin dificultades, lo que aumenta la satisfacción del usuario y la confianza en la marca o servicio.

Garantizar la seguridad: Las pruebas de software incluyen pruebas de seguridad para identificar posibles vulnerabilidades y riesgos de ataques. Al realizar pruebas de penetración, pruebas de seguridad y pruebas de protección de datos, se puede fortalecer la seguridad de la página web y proteger la información sensible de los usuarios, evitando así posibles incidentes de seguridad y daños a la reputación de la empresa.

Optimización del rendimiento: Las pruebas de software ayudan a evaluar el rendimiento de la página web, como la velocidad de carga, la respuesta del servidor y la capacidad de manejar una carga de usuarios simultáneos. Mediante la identificación de cuellos de botella y la

optimización del rendimiento, se logra una página web más rápida, eficiente y que puede ofrecer una experiencia fluida incluso en situaciones de alto tráfico.

Cumplimiento de estándares y requisitos: Las pruebas de software permiten verificar si la página web cumple con los estándares, reglamentos y requisitos establecidos, ya sea a nivel legal, de accesibilidad o de la industria en la que se encuentra. Esto es especialmente relevante para garantizar la accesibilidad para usuarios con discapacidades y cumplir con las normativas de privacidad y protección de datos.

Ahorro de costos a largo plazo: Al realizar pruebas de software de manera adecuada y temprana en el ciclo de desarrollo, se pueden identificar y solucionar problemas antes de que se conviertan en costosos desafíos en las etapas posteriores. La detección temprana de errores y fallos evita retrabajos y problemas costosos en el futuro, lo que ahorra tiempo y recursos en el proceso de desarrollo y mantenimiento de la página web.

1.4. OBJETIVOS

1.4.1. Objetivo General

Realizar pruebas de software de una plataforma web que se encuentra alojada en el servidor local mediante el uso de las pruebas unitarias, integración, funcionales, de extremo a extremo, aceptación, rendimiento y humo con la finalidad de detectar anomalías y verificar su correcto funcionamiento para su operatividad.

1.4.2. Objetivos Específicos

- Llevar a cabo una investigación exhaustiva sobre las pruebas de software, analizando su composición, características y funciones individuales.

- Realizar pruebas con el objetivo de identificar cualquier anomalía que presente el software.
- Realizar pruebas de verificación de una plataforma web.
- Generar una documentación detallada que registra cada una de las pruebas de software realizadas.

1.5. ALCANCES Y LIMITACIONES

1.5.1 Alcances

El alcance del proyecto se centra en destacar la importancia de identificar y ejecutar distintas categorías de pruebas de software, así como generar documentación detallada de dichas pruebas. Las acciones mencionadas del proyecto se realizan tales cosas como:

- **Identificación del proceso de las distintas categorías de pruebas de software:** El objetivo es comprender y describir cada categoría de prueba existente, así como el enfoque y los pasos involucrados en su aplicación. Esto permitirá tener un conocimiento claro de las diferentes formas en que se pueden evaluar y validar el software.
- **Ejecución de pruebas para descubrir anomalías:** La realización de pruebas de software tiene como finalidad detectar cualquier anomalía o error presente en el software. Mediante la ejecución de estas pruebas, se busca identificar y corregir posibles fallas que podrían afectar negativamente el funcionamiento del software y la experiencia del usuario.
- **Generación de documentación detallada:** Es fundamental documentar cada una de las pruebas de software realizadas, registrando los procedimientos, resultados y hallazgos obtenidos. Esta documentación servirá como referencia en el futuro,

permitiendo un seguimiento efectivo de los problemas identificados, así como la garantía de calidad del software.

1.5.2. Limitaciones

La prueba del software de la plataforma web, si bien es una parte importante del proceso de desarrollo, también tiene ciertas limitaciones que es importante tener en cuenta. Algunas de estas restricciones incluyen:

- **Cobertura limitada:** Es difícil lograr una cobertura completa cuando se prueba el software de una página web, especialmente en proyectos grandes y complejos. La amplia gama de navegadores, dispositivos, sistemas operativos y configuraciones de red que utilizan los usuarios puede dificultar la prueba en todas las combinaciones posibles, lo que puede dejar algunos escenarios sin probar.
- **Errores difíciles de reproducir:** algunos errores pueden ser intermitentes o depender de condiciones ambientales específicas, lo que dificulta su reproducción y corrección durante las pruebas. Esto puede provocar que algunos problemas no se detecten ni solucionen correctamente antes de que se lance el sitio web.
- **Sin garantía absoluta de calidad:** aunque las pruebas de software son una herramienta importante para mejorar la calidad de un sitio web, no pueden garantizar que esté completamente libre de errores o problemas. Siempre existe la posibilidad de que algunos errores pasen desapercibidos o que surjan problemas inesperados después del lanzamiento.

CAPÍTULO 2: MARCO TEÓRICO

En este capítulo se presentan todos los fundamentos teóricos que serán usados para la realización del proyecto. Los conceptos técnicos propios del área de ingeniería en sistemas y conceptos generales.

2.1 Funcionamiento del servidor web

Los servidores web, indispensables para acceder a los datos dentro de cualquier página de internet, se encargan de varios pasos que, para un usuario que navega en internet, significa un simple clic, pero en realidad constan de procesos mucho más complejos que eso, tales como (Ávila, 24 de noviembre del 2022):

- El internauta escribe una dirección web en Chrome o cualquier otro navegador. Eso es el envío de una solicitud al servidor web.
- El servidor web busca la información solicitada, ya sea en el propio servidor físico, el llamado hardware, o en un hosting.
- La información y los archivos solicitados son enviados por el servidor web al usuario, quien los recibe y los percibe como parte del contenido de la página a la que quiere acceder. Ya en una sola presentación ensamblada.

2.1.1 Pruebas de software

Las pruebas de software son un conjunto de procesos con los que se pretende probar un sistema o aplicación en diferentes momentos para comprobar su correcto funcionamiento. Este tipo de pruebas abarca cualquier estadio del desarrollo del sistema, desde su creación hasta su puesta en producción. Lo interesante de las pruebas es que se puedan ejecutar de manera automática, para determinar en cualquier momento y en caso de que contemos con una aplicación estable o si, por el contrario, un cambio en una parte ha afectado a otras partes sin que nos demos cuenta (Jorge Turrado, 10 de marzo de 2020).

2.2 Tipos de pruebas de software

2.2.1 Pruebas unitarias

Las pruebas unitarias son una parte esencial del proceso de desarrollo de software que pone a prueba los componentes individuales de la aplicación o programa de software para detectar el error fácilmente. El objetivo principal de las pruebas unitarias es comprobar que cada parte individual funciona según los requisitos del cliente. Puede tener muchas entradas, pero una única salida (Durga Prasad Acharya, 25 de septiembre del 2023).

2.2.2 Pruebas de integración

Las pruebas de integración pruebas valoran si los componentes individuales trabajan en conjunto tal y como se espera de ellos. O lo que es lo mismo, se prueba el funcionamiento de los diferentes módulos del sistema una vez unidos o agrupados en elementos mayores, verificando el comportamiento de los mismos frente a las comunicaciones que se produzcan entre ellos. El objetivo es la localización de errores de interfaces y comprobar el correcto funcionamiento conjunto de los componentes (Gómez, 2006).

2.2.3 Pruebas funcionales

Se denominan pruebas funcionales o *Functional Testing*, a las pruebas de software que tienen por objetivo probar que los sistemas desarrollados, cumplan con las funciones específicas para los cuales han sido creados, es común que este tipo de pruebas sean desarrolladas por analistas de pruebas con apoyo de algunos usuarios finales, esta etapa suele ser la última etapa de pruebas y al dar conformidad sobre esta el paso siguiente es el pase a producción (Juan, mayo del 2010)

2.2.4 Pruebas de extremo a extremo

Las pruebas de extremo a extremo o las pruebas de interfaz de usuario son un enfoque para probar una aplicación web. Una prueba de extremo a extremo verifica que una aplicación web

cumpla con los requisitos y funcione como se espera. Se prueba el llamado flujo de usuario (David, 29 de septiembre del 2022).

2.2.5 Pruebas de aceptación

Son las pruebas finales que realiza el cliente con el fin de verificar por sí mismo el cumplimiento de los requisitos especificados, justo antes de su paso a producción. También son conocidas como pruebas de usuario o UATs. En dichas pruebas el usuario valida si el software cumple sus expectativas (Nuria Gómez, 2015).

2.2.6 Pruebas de rendimiento

Las pruebas de rendimiento son un tipo de *testing* no funcional que tiene como objetivo esencial someter al sistema en evaluación a altas cargas de trabajo transaccional, simulando la actividad real de los futuros usuarios que interactuaron con él. Estas ayudan a predecir el comportamiento del sistema y conocer el grado en que realiza las funciones para las que ha sido diseñado, sin degradación del servicio y con un tiempo de respuesta óptimo y estable (Verity, 2023).

2.2.7 Pruebas de humo

Las pruebas de humo son un conjunto de pruebas aplicadas a cada nueva versión, su objetivo es validar que las funcionalidades básicas de la versión se cumplen según lo especificado. Estas pruebas buscan grandes inestabilidades o elementos clave faltantes o defectuosos, que hacen imposible realizar la prueba como fue planificado para la versión. Si la versión no pasa las pruebas de humo, no se comienza la ejecución de las pruebas planificadas de la versión (Beatriz Pérez, 21 de Setiembre del 2006)

2.3 Aplicar las pruebas de software

2.3.1 Pruebas unitarias

Una prueba unitaria toma una pequeña unidad de la aplicación, normalmente un método, lo aísla del resto del código y comprueba que se comporta según lo previsto. Su objetivo es comprobar que cada unidad de funcionalidad funciona según lo previsto, de modo que los errores no se propaguen a lo largo de la aplicación. Detectar un error donde se produce es más eficaz que observar el efecto de un error indirectamente en un punto de error secundario. Las pruebas unitarias tienen el mayor efecto en la calidad del código cuando es una parte integral del flujo de trabajo de desarrollo de software. En cuanto se haya escrito un método, se deben escribir pruebas unitarias que comprueben el comportamiento del método en respuesta a los casos estándar, límites e incorrectos de los datos de entrada, y que comprueben las suposiciones explícitas o implícitas realizadas por el código. Como alternativa, con el desarrollo controlado por pruebas, las pruebas unitarias se escriben antes del código. En este escenario, las pruebas unitarias actúan como documentación de diseño y especificaciones funcionales (David, 13 de julio del 2023).

2.3.2 Pruebas de integración

Con las pruebas de integración se garantiza que todos los componentes de un producto funcionen conjuntamente de forma correcta. Su principal objetivo es garantizar que no haya problemas de comunicación o de transferencia de información entre componente. Las pruebas de integración de sistemas son comunes debido a diversas razones (Juan José, 24 de mayo del 2023):

- Los desarrolladores utilizan diferentes lógicas de programación al crear. Además, si un desarrollador realiza cambios en el sistema sin realizar pruebas unitarias, las pruebas de integración se vuelven esenciales para evaluar la efectividad de dichos cambios.

- Durante el paso de información entre componentes, la estructura de dicha información puede sufrir cambios ocasionando defectos en el funcionamiento.
- Los componentes interactúan con herramientas y APIs de terceros. Es crucial realizar pruebas de integración para asegurarse de que los datos enviados a estas APIs o herramientas sean correctos y que las respuestas generadas se ajusten a las expectativas establecidas.

2.3.3 Tipos de test de integración de software

Existen varios tipos o enfoques de test de integración de software, siendo el más popular el de *Big Bang*, el ascendente y el incremental. La selección de uno u otro depende de varios factores, como el coste, complejidad, criticidad de aplicación, entre otros. Hay otros tipos de test de integración, pero son menos conocidos, tal como el de servicios distribuidos, el de integración sándwich, integración de la red troncal, integración de alta frecuencia, integración de capas, entre otros *Technologies* (Gómez, 5 de octubre del 2022).

2.3.4 Incremental

El modelo incremental de gestión de proyectos tiene como objetivo un crecimiento progresivo de la funcionalidad. Es decir, el producto va evolucionando con cada una de las entregas previstas hasta que se amolda a lo requerido por el cliente o destinatario (Pérez, 2021).

2.3.5 Big Bang

Big Bang es el enfoque más fácil de aplicar en las pruebas de integración, todo el sistema se trata como un sub-sistema, es decir, se prueban todos los módulos de una sola vez (Ismael Martínez, 2017).

En la tabla 1 se muestra la composición del concepto *big bang* en la que se desglosa los siguientes elementos tales como son:

- Objetivos
- Descripción de la prueba
- Resultados esperados

Objetivo	Descripción de la prueba	Resultados esperados
Aprobar el enlace entre el *login* y la pantalla de inicio.	Credenciales de acceso y hacer clic en el botón acceder.	La aplicación lo envía al panel de control en la página de inicio.
Verificar la comunicación entre el módulo de modificación de tareas y el de la lista.	Hacer clic en el botón modificar.	La aplicación lo envía en la interfaz donde se muestra la lista de tareas abiertas.

Tabla 1Ejemplo de pruebas(Fuente: Elaboración propia, 2023)

2.3.6 Prueba funcional

Las pruebas funcionales se centran en los requisitos comerciales de una aplicación. Solo verifican la salida de una acción y no verifican los estados intermedios del sistema al realizar esa acción, en los siguientes puntos se mencionan las características para la ejecución de pruebas funcionales tales como son (Lorena ramirez,23 de noviembre del 2023):

- **Pruebas de unidad:** En las pruebas unitarias (*unit testing*), el *tester* verifica los componentes de software individuales. El objetivo es probar si los componentes se comportan de acuerdo con los requisitos. Las pruebas unitarias son generalmente bastante económicas de automatizar y pueden ejecutarse muy rápidamente en un servidor de integración continua.

- **Pruebas de integración:** Las pruebas de integración verifican que los diferentes módulos o servicios utilizados por su aplicación funcionen bien juntos. Por ejemplo, puede probar la interacción con la base de datos o asegurarse de que los micro servicios funcionen juntos como se espera.
- **Pruebas del sistema:** Aquí, se prueba la interacción de los diferentes componentes con el sistema.
- **Pruebas de cordura:** Esto prueba el razonamiento lógico relacionado con el funcionamiento del programa.
- **Prueba de humo:** La prueba de humo o *smoke testing,* prueba funcionalidades simples y básicas, como si el usuario puede iniciar o cerrar sesión.
- **Pruebas de interfaz:** El interfaz *testing* consiste en comprobar si la comunicación entre dos sistemas de software se lleva a cabo correctamente.
- **Pruebas de regresión:** Verifican que el producto siga funcionando de forma correcta después de que se realice algún cambio u actualización.
- **Pruebas beta:** Este test se realiza para que algunos usuarios puedan probar el producto e informar errores.

2.3.7 Pruebas de extremo a extremo con UTAM (unidad territorial de análisis de movilidad)

UTAM aborda las limitaciones de las pruebas E2E (extremo a extremo) convencionales haciéndolas más fáciles de escribir y mantener. UTAM no reemplaza una herramienta de prueba de UI (*User Interface*), pero la complementa al desacoplar el código de prueba del DOM (*Document Object Model*) gracias a los objetos de página (Philippe Ozil, 11 de mayo del 2022).

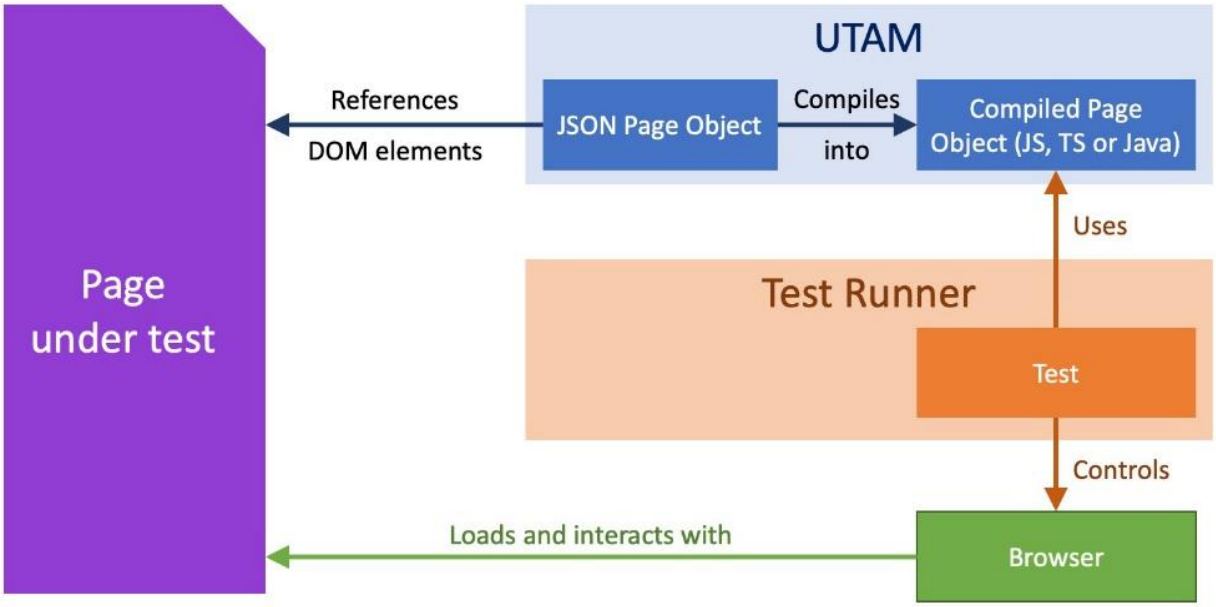

Figura 1. Diagrama UTAM (fuente: Philippe Ozil, 11 de mayo del 2022)

Los objetos de página son bloques reutilizables que le permiten hacer referencia al DOM de la página bajo prueba con selectores de CSS. Puede componer objetos de página para crear patrones avanzados.

En la figura 2 se presenta un ejemplo ilustrativo del clásico programa "Hola, mundo". Este programa es comúnmente utilizado en la enseñanza de la programación como una introducción simple a la sintaxis de un lenguaje de programación.

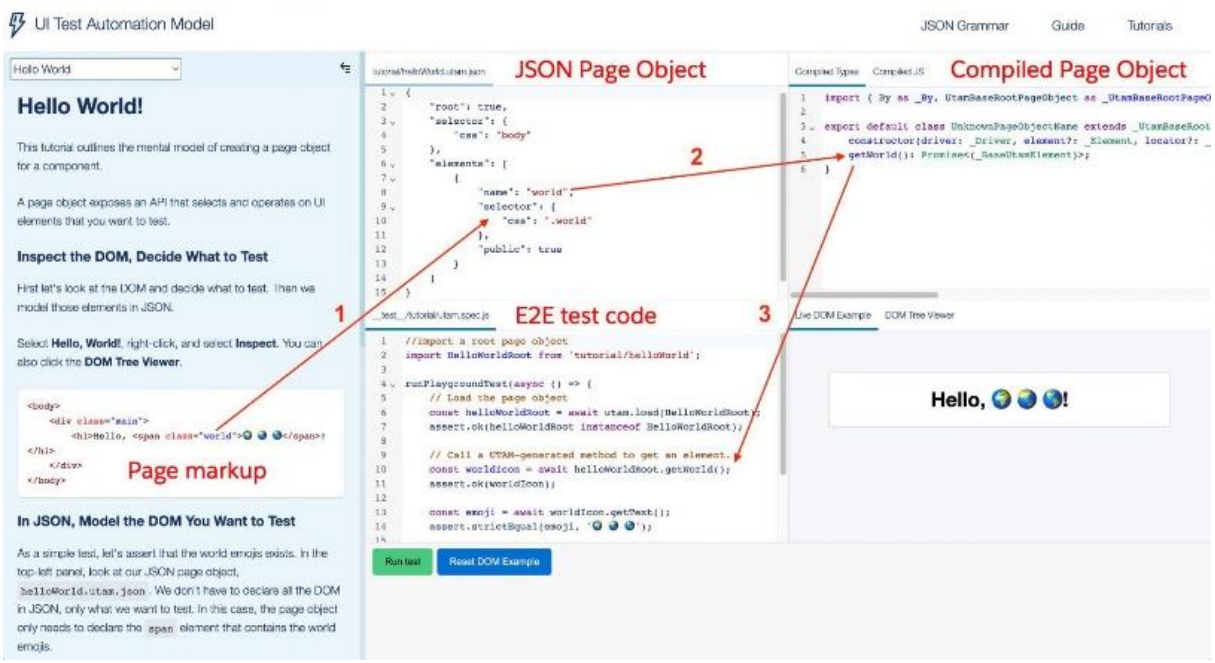

Figura 2. Ejemplo 1(fuente: Philippe Ozil, 11 de mayo del 2022)

Los pasos clave necesarios para probar esta página son:

1) Se afirma un objeto de página *JSON* que contiene un elemento *world* . Este elemento tiene un selector de clase CSS *.world* , que coincide con un elemento DOM en la página bajo prueba.
2) La compilación de la página, genera una clase con un método *getWorld* que está vinculado al elemento *world* que hemos definido en *JSON*.
3) Se proporciona el método *getWorld* del objeto de página compilado en nuestro código de prueba para acceder al contenido del DOM con la clase *.world* CSS.

2.3.8 Prueba de aceptación

Las pruebas de aceptación son pruebas formales con respecto a las necesidades del usuario. Consiste en requisitos y procesos de negocio llevados a cabo para determinar si un sistema cumple con los criterios de aceptación, y para permitir que el usuario, los clientes u otra entidad autorizada determinen si se acepta o no el sistema. El principal propósito de la prueba de aceptación es validar

de extremo a extremo el flujo de negocio en el sistema de software. Los tipos de pruebas de aceptación son las siguientes (Iván, mayo del 2023).

- **Pruebas de aceptación del usuario:** Mediante estas pruebas los usuarios verifican que el sistema de software sea funcional según sus requerimientos.
- **Pruebas de aceptación operacional:** Son las pruebas mediante las que se comprueba que los procesos y procedimientos están en su lugar, para permitir que el sistema sea usado y permita su mantenimiento. Esto incluye el control realizado a la copia de seguridad, los procedimientos para la recuperación del sistema, la capacitación para los usuarios finales, los procedimientos de mantenimiento y de seguridad.
- **Pruebas de contrato y regulación:** El sistema se prueba con los criterios de aceptación según se documenta en un contrato, antes de ser aceptado, así como para asegurar que cumple con los estándares gubernamentales, legales y de seguridad.
- **Pruebas alfa y beta:** Se llevan a cabo en los sitios de los desarrolladores, y consiste en pruebas del sistema operativo por parte del personal interno, antes de que sea entregado a los clientes externos. Las pruebas beta se llevan a cabo en los sitios de los clientes, y consiste en que un grupo de clientes que utilizan el sistema proporcionen información antes de que se libere a otros clientes. A menudo es llamada prueba de campo.
- **Método:** Esta prueba implica la técnica de caja negra que consiste en verificar que lo que se espera que realice el sistema de software, lo haga efectivamente

2.3.9 Pruebas de rendimiento

Para llevar a cabo de manera efectiva un ciclo de pruebas de rendimiento, se deben seguir siete pasos fundamentales con el objetivo de garantizar el cumplimiento de los requisitos tanto del cliente como del usuario, tales como son: (Scandaroli, 10 de febrero del 2022).

- **Identificar el entorno de prueba:** Identificar y poner a prueba el entorno de producción, lo cual también incluye el hardware, software y redes que utilizará el producto.
- **Identificar los criterios de aceptación:** Identificar las metas y limitaciones de los tiempos de respuesta, rendimiento y uso de recursos.
- **Planificar y diseñar pruebas:** Determinar la forma en que se simularán los escenarios, los datos y las métricas que se recolectarán durante la ejecución de las pruebas.
- **Configurar el entorno de prueba:** Preparar el entorno, las herramientas, el hardware, la red y cualquier otro elemento relacionado al uso del producto.
- **Implementar el diseño de las pruebas:** Desarrollar las pruebas de acuerdo con lo que se diseñó previamente.
- **Ejecutar las pruebas:** Correr y monitorear la ejecución de las pruebas.
- **Analizar resultados, reportes y repetir pruebas:** Analizar los datos que se obtuvieron y evaluar si se cumple con las métricas establecidas previamente por el equipo. La ejecución de la prueba se aprueba o se realizan cambios para correr la prueba de nuevo aplicando mejoras.

2.3.10 Pruebas de humo

Para crear un software, existen muchas acciones de programación a ejecutar. Cuando se llega a la etapa de *testing*, se deben aplicar los siguientes pasos para una prueba de humo correcta tales como (Eduardo Núñez, 31 de marzo del 2022):

1. **Establecer el número de casos de pruebas de humo**

Con el objetivo de lograr la eficiencia de una prueba de humo, es importantísimo determinar primero el número de casos a testear. Esto quiere decir que los responsables de una prueba de humo tienen que saber de antemano cuántas veces se requiere que la aplicación, sistema o plataforma pase por la prueba de humo para certificar su validez.

2. **Crear los entornos de prueba de humo**

En este paso es esencial generar la estructura, los guiones y los criterios para la prueba de humo. Tanto si se llevan a cabo pruebas de humo manuales, automatizadas o híbridas, todo el entorno debe estar preparado previamente.

3. **Ejecutar la prueba de humo**

En este momento es cuando se ejecuta realmente una prueba de humo. De ser el encargado de este *testing*, tiene que monitorizar el debido proceso conforme a lo indicado en el paso previo. Además, debe contar con un plan B por si falla la herramienta de *testing* automatizado o algún colaborador que participe como *tester* no implemente su trabajo con éxito. Todas estas funciones son parte, también, de las habilidades de un programador.

4. **Analizar los resultados de la prueba de humo**

Esta es la etapa final de la prueba de humo. Se determina cuál es el porcentaje de éxito del *testing,* si el software está listo para pasar a pruebas más avanzadas y si cumple con los estándares del producto que se ha planificado sacar al mercado.

2.3.11 Intérpretes

Un intérprete es un programa informático que procesa el código fuente de un proyecto de software durante su tiempo de ejecución, es decir, mientras el software se está ejecutando, y actúa como una interfaz entre ese proyecto y el procesador. Un intérprete siempre procesa el código línea por línea, de modo que lee, analiza y prepara cada secuencia de forma consecutiva para el procesador. Este principio también se aplica a las secuencias recurrentes, que se ejecutan de nuevo cada vez que vuelven a aparecer en el código. Para procesar el código fuente del software, el intérprete recurre a sus propias bibliotecas internas: en cuanto una línea de código fuente se ha traducido a los correspondientes comandos legibles por máquina, esta se envía directamente al procesador. El proceso de conversión no finaliza hasta que se ha interpretado todo el código. Solo se interrumpe prematuramente si se produce un fallo durante el procesamiento, lo que simplifica mucho la resolución de los errores, ya que la línea de código problemática se detecta inmediatamente después de ocurrir el fallo. BASIC, Perl, Python, Ruby y PHP son algunos de los lenguajes de programación más famosos que dependen de un intérprete para ser traducidos de código fuente a código máquina. Por ello, también suelen llamarse lenguajes interpretados tales como son: (IONOS Digital Guide, 2019)

La tabla 2, que se exhibe a continuación, presenta una comparación detallada entre las características y funcionalidades del intérprete y el compilador en el contexto del desarrollo de software.

	Intérprete	Compilador
Momentos en que se traduce el código fuente.	Durante el tiempo de ejecución del software.	Antes de ejecutar el software
Procedimientos de traducción.	Línea por línea.	Siempre todo el código
Presentación de errores de código.	Después de cada línea.	En conjunto, después de toda la compilación.
Velocidad de traducción.	Alta.	Baja
Eficiencia de traducción.	Baja.	Alto
Coste de desarrollo.	Bajo.	Alta
Lenguajes típicos.	Php , pel, Python, ruby	C, c++, pascal

Tabla 2. Diferencia de intérprete y compilador (fuente: Elaboración propia, 2023).

2.3.12 Editor de texto

Un editor de texto es cualquier programa de procesamiento de texto en la que se implementa el uso de escribir y editar un texto, como *Word Pad* y *NotePad* para Windows y *SimpleText* y *TextEdit* para Mac. Se trata de un programa informático que permite editar o crear archivos digitales compuestos únicamente por textos sin formato. Es decir, archivos que no contengan formato de texto específico y que son conocidos comúnmente como archivos de texto o texto plano (Susana, 1 de marzo del 2022).

2.4 Funciones típicas de un editor de texto

2.4.1 Registro de texto

Es el conjunto de diferencias provocadas por la relación entre el texto y el contexto situacional. Existen registros formales e informales, registros escritos y hablados, registros científicos, periodísticos, didácticos, etc. (Graciela reyes, 4 de octubre del 2013).

2.4.2 Filtros

Los Filtros son unos mecanismos que, sea en el campo de la informática o en el de las páginas web, se utilizan para determinar qué tipo de contenidos se deben visualizar o mostrar por pantalla, o cuáles no deben aparecer. Algo bastante habitual en las webs que cuidan la experiencia del usuario (García flores, 12 de febrero del 2019).

2.5 Acceso remoto

Los editores de texto están orientados a manipular y crear archivos de texto plano, los cuales tienen una enorme utilidad dentro de la Informática, sobre todo en el área de la programación, ya que el código fuente de los programas está hecho en texto plano, así como también los script interpretados de algunos lenguajes, como JavaScript, Perl, Python, shell, etc (Noguera, 15 de mayo del 2014).

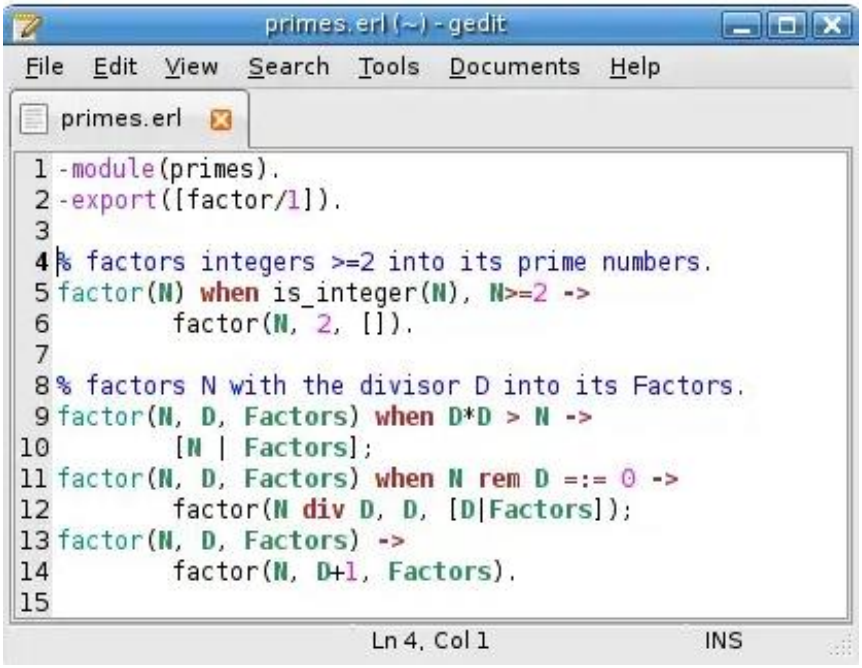

Figura 3. Ejemplo3(fuente: Noguera, 15 de mayo del 2014).

2.6 Definición de software de aplicación

El software de aplicación es el término que define cualquier programa que sea instalado en un ordenador. Esto con el fin de realizar las tareas que un usuario necesite, es decir, se adapta a las necesidades específicas. Un software de aplicación puede abarcar en distintas tareas. Por ejemplo (Alan Alvares, 19 de abril del 2022):

- Un editor de texto
- Una hoja de cálculo
- Reproductor multimedia y otros más.

2.7 Software malicioso o malintencionado

El *malware* (contracción de malicioso software), o *bardware (*software que tiene como objetivo principal introducirse en los sistemas e infectar los servidores de forma transparente a los administradores del sistema) es un código maligno o software malintencionado que infecta los

sistemas informáticos, como ya sabemos. Su objetivo es infiltrarse en un sistema o dañarlo, sin el consentimiento del propietario. Las infecciones o ataques se realizan aprovechando las vulnerabilidades o agujeros de seguridad de los sistemas informáticos (Torres, 2022).

2.8 Pruebas de caja negra

Las Pruebas de Caja Negra, es una técnica de pruebas de software en la cual la funcionalidad se verifica sin tomar en cuenta la estructura interna de código, detalles de implementación o escenarios de ejecución internos en el software. En las pruebas de caja negra, nos enfocamos solamente en las entradas y salidas del sistema, sin preocuparnos en tener conocimiento de la estructura interna del programa de software. Para obtener el detalle de cuáles deben ser esas entradas y salidas, se basa en los requerimientos de software y especificaciones funcionales (Gustavo terrera, 26 de febrero del 2017).

2.9 Enfoque de la caja negra

Las pruebas se llevan a cabo sobre la interfaz del software, y es completamente indiferente el comportamiento interno y la estructura del programa. Los casos de prueba de la caja negra pretenden demostrar tales como son (Camilo, 2010):

- Las funciones del software son operativas.
- La entrada se acepta de forma adecuada.
- Se produce una salida correcta.
- La integridad de la información externa se mantiene.

Se derivan conjunto de condiciones de entrada que ejerciten completamente todos los requerimientos funcionales del programa.

La prueba de la caja negra intenta encontrar errores de las siguientes categorías:

- Funciones incorrectas o ausentes.

- Errores de interfaz.
- Errores en estructuras de datos o en accesos a bases de datos externas.
- Errores de rendimiento.
- Errores de inicialización y de terminación.

Los casos de prueba deben satisfacer los siguientes criterios:

- Reducir, en un coeficiente que es mayor que uno, el número de casos de prueba adicionales.
- Que digan algo sobre la presencia o ausencia de clases de errores.

En la figura 4 se ilustra un ejemplo del proceso de caja negra, donde se identifican tres pasos tales como son:

- Entrada
- Funciones
- Salida

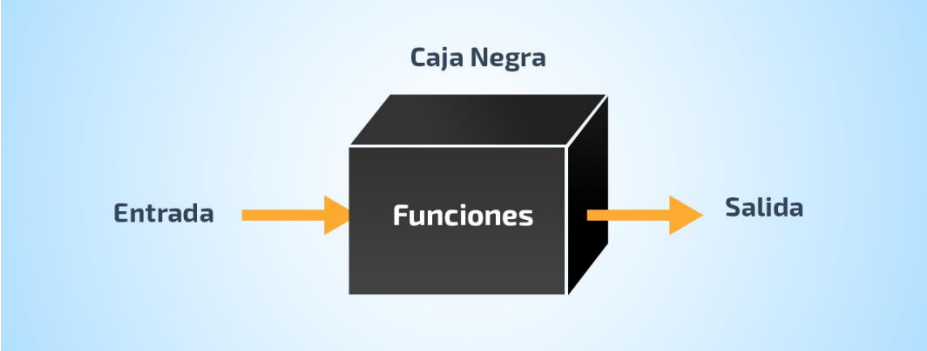

Figura 4. Función de caja negra (fuente: SQA - Software Quality, 2019).

2.10 Técnicas de caja negra

2.10.1 Tablas de decisión

Es una técnica utilizada para probar el comportamiento de un sistema utilizando diferentes combinaciones de entrada. Este es un enfoque donde las diferentes combinaciones de entrada y los resultados se muestran en forma tabular, tales como: (Gómez C, 19 de enero del 2023).

- En las filas se colocan las condiciones y los resultados, donde las condiciones se ponen en la parte superior y los resultados en la parte inferior.
- Cada columna corresponde a una regla la cual es definida por las combinaciones de entrada.
- Los valores a colocarse son generalmente booleanos como V (verdadero) y F (Falso). Si hay un valor que no importa para obtener cierto resultado se puede marcar con un guion "-".

En la Figura 5, se ilustra un ejemplo que ejemplifica el proceso de acceso a un sitio web.

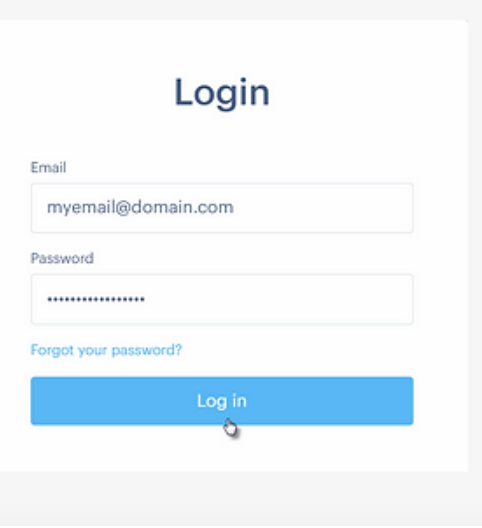

Figura 5. Login (fuente: Gómez C, 19 de enero del 2023)

Las condiciones son simples, si se proporciona un usuario y una contraseña válida, el usuario puede iniciar sesión exitosamente, de lo contrario aparecerán mensajes de error. En referencia a la Tabla 3, E=Error, S=Inicio de sesión exitoso.

Condiciones	Regla 1	Regla 2	Regla 3	Regla 4
Usuario válido (V/F)	F	V	F	V
Contraseña válida (V/F)	F	F	V	V
Resultados	**E**	**E**	**E**	**S**

Tabla 3. Tabla de condiciones (Fuente: Gómez C, 19 de enero del 2023)

2.10.2 Diagrama de transiciones

Un diagrama de transiciones es una colección finita de círculos, los cuales se pueden rotular para fines de referencia, conectados por flechas que reciben el nombre de arcos. Cada uno de estos arcos se etiqueta con un símbolo o categoría de símbolos que podría presentarse en la cadena de entrada que se analiza. Uno de los círculos se designa con un apuntador, y representa una posición inicial. Además, por lo menos uno de los círculos se representa como un círculo doble; estos círculos dobles designan posiciones del diagrama en las cuales se ha reconocido una cadena válida (Pedro Antonio, 7 de octubre del 2023).

2.10.3 Pruebas de caso de uso

El caso de uso es la descripción del conjunto de interacciones del sistema con uno, o varios actores, para alcanzar un objetivo. Éstos nos sirven como técnica para la especificación (agrupación) de requisitos funcionales. En la que se implementa los diferentes casos de uso para determinar una funcionalidad particular, o un grupo de funcionalidades relacionadas. Por otra

parte, un caso de uso debe contemplar las múltiples rutas posibles que el usuario puede seguir para llevar a cabo el proceso, llamando escenario a cada una de estas posibles rutas (Adrián Prats, 13 de febrero del 2019).

2.10.4 Partición de equivalencias

Esta técnica es una extensión de la partición de equivalencia, solo se utiliza para valores numéricos y cuando los valores se encuentran ordenados. Una partición contiene un valor inicial y un valor final (un número mínimo y uno máximo) y son denominados como valores límite (Gómez C, 19 de enero del 2023).

2.10.5 Análisis de valores límites

El análisis de valores límite es la técnica de diseño de pruebas de caja negra en la cual los casos de prueba son diseñados basándose en los valores límite. Se basa en la evidencia experimental de que los errores suelen aparecer con mayor probabilidad en los extremos de los campos de entrada. Un análisis de las condiciones límites de las clases de equivalencia aumenta la eficiencia de la prueba (Rosalba Meléndez, 26 de octubre del 2016).

2.11 Prueba de caja blanca

Las pruebas de Caja Blanca también suelen ser llamadas estructurales o de cobertura lógica. En ellas se pretende investigar sobre la estructura interna del código, exceptuando detalles referidos a datos de entrada o salida, para probar la lógica del programa desde el punto de vista algorítmico. Realizan un seguimiento del código fuente según se va ejecutando los casos de prueba, determinándose de manera concreta las instrucciones, bloques, etc. que han sido ejecutados por los casos de prueba (Eduardo Salazar, 8 de octubre del 2012).

2.12 Técnicas de prueba de caja blanca

Estas ejecuciones también son llamadas pruebas estructurales o basada en la estructura. De igual forma en las pruebas se mencionan las estructuras y procesamientos dentro del objeto de prueba. Este conjunto de pruebas toma como base el análisis de la arquitectura o el diseño detallado. También se basan en el análisis de la estructura interna o el código del objeto de prueba (Reyes Sánchez, 29 de marzo del 2021).

2.13 Tipos de pruebas de caja blanca

2.13.1 Pruebas unitarias

El programador suele realizarlo como prueba inicial de una aplicación. En este método, cada bloque de código se prueba a medida que se desarrolla. El desarrollador prueba unas pocas líneas de código, una sola función o un objeto para comprobar su correcto funcionamiento. Las pruebas unitarias son útiles porque identifican la mayoría de los errores en una fase temprana del ciclo de desarrollo, lo que hace que sean más baratos y fáciles de solucionar (Marcasco, 1 de diciembre del 2022).

2.13.2 Prueba de calidad

La calidad del software es crítica en un mundo principalmente digital. Por ejemplo, un defecto no detectado puede ocasionar indisponibilidades del sistema, y una mala configuración de plataformas en la nube puede ocasionar brechas de seguridad o pérdida de datos. Los defectos de software incrementan drásticamente el coste de desarrollo. Y, cuando el software es publicado, el coste de encontrar y solucionar es significativamente superior que durante la fase de diseño o desarrollo (Dr. Gareth Smith, 28 de enero del 2022).

2.13.3 Pruebas de penetración

El *pentesting,* también conocido como *pentest* o prueba de penetración es un tipo de prueba que utilizan las empresas para realizar un análisis de vulnerabilidades y debilidades en su seguridad informática. En otras palabras, es una prueba que consiste en atacar diferentes entornos o sistemas para detectar y prevenir posibles fallos o ataques. La palabra proviene de la abreviatura que se forma al unir las palabras "*penetration*" y "*test",* que en español significa "penetración" y "prueba" (Juan José, 22 de agosto del 2023).

2.13.4 Pruebas de mutación

Los test de mutación son pruebas para medir la calidad del código a largo plazo. Estas pruebas consisten en cambiar (mutar) ciertos puntos del código fuente y ver si se detectan estos cambios (errores) mediante los test unitarios. Las pruebas de mutación son, por tanto, un sistema doble de calidad que nos permite comprobar tanto la eficacia de nuestros test unitarios y respaldar la cobertura de código exigida en nuestros proyectos, como la robustez del código testeado mediante esos test unitarios (Ignacio rivera, 1 de junio del 2021).

2.14 Visual Studio Code

Visual Studio Code es un editor de código fuente liviano pero potente que se ejecuta en su escritorio y está disponible para Windows, *macOS* y Linux. Viene con soporte integrado para JavaScript, *TypeScript* y *Node.js* y tiene un rico ecosistema de extensiones para otros lenguajes y tiempos de ejecución (como C++, C#, Java, Python, PHP, Go, .NET) Los siguientes puntos que se mencionan a continuación constituyen las características distintivas de Visual Studio Code tales como: (*Documentation for Visual Studio code*, 2021).

- ✓ Visual Studio Code es un editor de código fuente construido sobre el *framework Electron*. Es compatible con varios lenguajes de programación y un conjunto de características que pueden o no estar disponibles para un lenguaje dado.
- ✓ Muchas de las características de Visual Studio Code no están expuestas a través de desaparecer si el usuario hace clic fuera de él o presiona una combinación de teclas en el teclado para interactuar con algo que está fuera de él. Esto también se aplica a los comandos que requieren mucho tiempo. Cuando esto sucede, el comando en progreso se cancela.
- ✓ En el rol de editor de código fuente, Visual Studio Code permite cambiar la página de códigos en la que se guarda el documento activo, el carácter que identifica el salto de línea (una opción entre LF y CRLF) y el lenguaje de programación del documento activo.

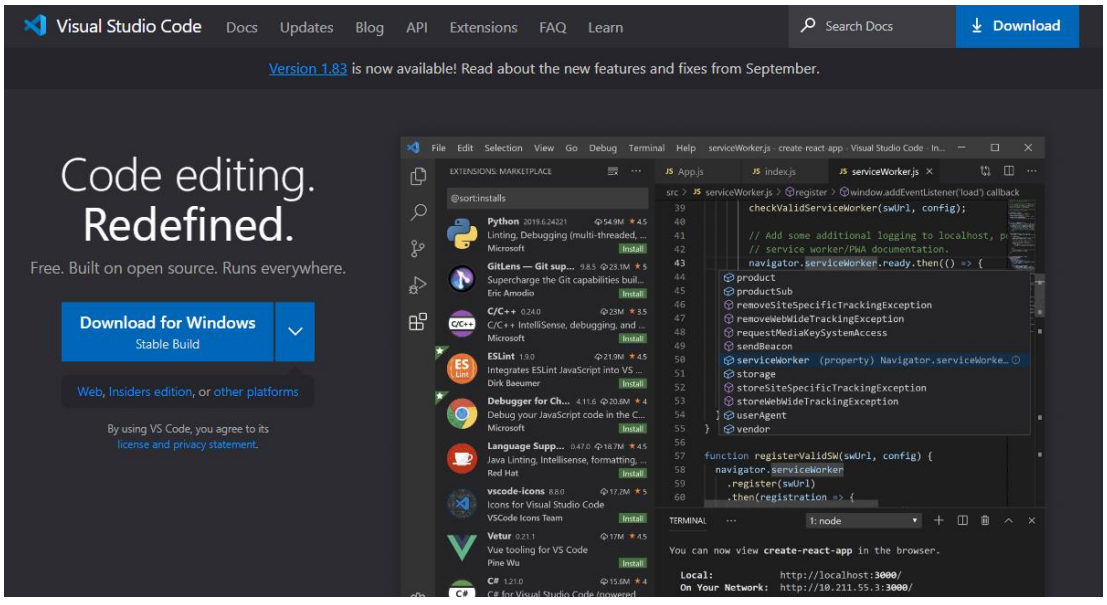

Figura 6. Visual Studio Code (fuente: Documentation for Visual Studio code*, 2021).*

2.15 Qmetry

Esta herramienta de gestión de pruebas está pensada específicamente para equipos de pruebas. Del mismo modo, su principal objetivo es que los *testers* se centren en el proceso de pruebas, sin tener que perder el tiempo en instalar y configurar una herramienta. Aunque, también permite la opción de poder alojarla en local para proyectos grandes. Es la suma de herramientas comerciales y de código abierto, basada en las tecnologías, *MySQL*, PHP, Apache y Linux , aparte de ahí prosigue, las características de *QMetry*, un conjunto de características y funcionalidades que se destacan en esta herramienta de gestión de pruebas de software tales como: (García , 2023).

- **Panel de control**

Qmetry te ofrece un panel de control desde el cual podrás tener actualizado el sistema en tiempo real y se puede consultar las métricas del proyecto. De esta forma, puedes tener un cuadro de mando listo para cada uno de los perfiles

- **Historial de contenidos**

Además, como en *JIRA*, tendrás disponible un historial de todas las modificaciones realizadas en las pruebas. Correcciones, ampliaciones, supresiones, etc. Todo queda archivado para poderlo consultar.

- **Importación y exportación de datos**

Te permite una fácil exportación e importación de los casos de prueba. Lo que te permite preparar tu plan de pruebas en una hoja de cálculo y subirlo de forma masiva. Y posteriormente, cuando tengas el plan de prueba con modificaciones, puedes sacar una copia para reutilizar.

- **Creación automatizada de casos de prueba**

Se pueden reutilizar los casos de prueba con *Exploratory testing*. Esto hace que mejore la eficiencia de los equipos, disminuyendo los esfuerzos.

- **Diversas ofertas de integración**

Dispone de diversas integraciones para sincronizar archivos, hacer pruebas continuas, automatizaciones y herramientas de BBDD. Algunos de los nombres que conocerás son *bitBucket, GitHub, GitLab, Bamboo, Jerkins, Cucumber, JUnit, TestNG, SpecFlow*, QAF y UFT.

- ***Feedback* e informes**

Puedes preparar resumen de datos en el que se tenga la trazabilidad de los distintos tipos de datos: Casos de prueba, historias de usuarios, defectos, etc. De esta forma, tendrás enlazada la información para un seguimiento más eficaz.

- **Automatización de las ejecuciones**

Esta herramienta permite la automatización de la ejecución de las pruebas. Tendrás soporte para la integración y distribución continua de software (CI/CD), RESTAPI y carga de resultado desde diversos macros.

2.16 IntelliJ IDEA

IntelliJ IDEA incorpora uno de los editores de código más potentes del sector. Comprende los detalles internos de su código gracias a la indexación inicial, lo que le permite detectar errores sobre la marcha, sugerir opciones de finalización de código con un conocimiento preciso del contexto, realizar una refactorización segura y mucho más (IntelliJ IDEA, 2023).

- **Inspecciones y acciones contextuales**

IntelliJ se destaca por su capacidad para llevar a cabo inspecciones en tiempo real que verifican la calidad y validez del código. Estas inspecciones no solo agilizan el proceso de programación, sino que también garantizan que se cumplan los más elevados de estándares de calidad y el cual se muestra en la figura 7.

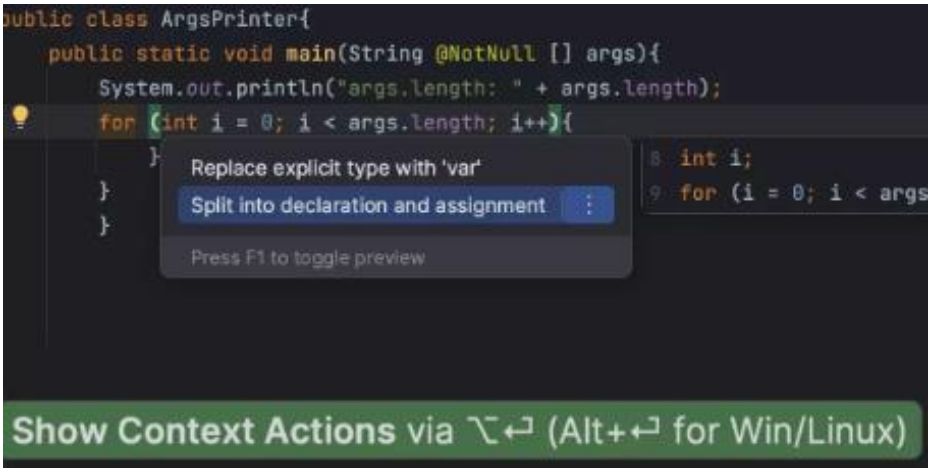

Figura 7. Ejemplo 3(fuente: Elaboración propia, 2023)

- **Finalización de código inteligente**

IntelliJ ofrece finalización de código contextual, brindando sugerencias relevantes según la ubicación del cursor en el código actual, sin necesidad de atajos o configuraciones adicionales, la función de finalización se activa de forma automática al comenzar a escribir código en el editor, en la figura 8 se ilustra un ejemplo del autocompletado de código, proporcionando un ejemplo de dicha funcionalidad.

Figura 8. Ejemplo 4 (fuente: elaboración propia, 2023)

- **Inyección de lenguaje**

En la figura 9 se habilita la manipulación de elementos de código en otros lenguajes integrados en el código principal. El usuario experimenta una uniformidad en su experiencia al

trabajar con el código Java principal y el código inyectado, ya que se brindan características como el resaltado de código, finalización y detección de errores para fragmentos de códigos insertados.

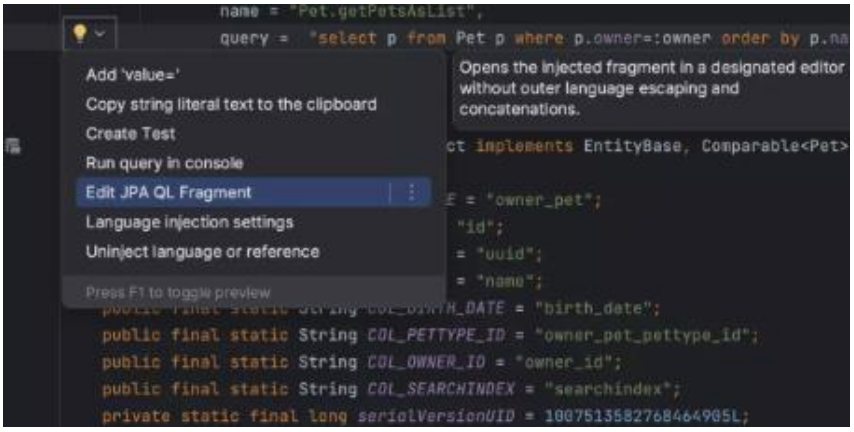

Figura 9. Ejemplo 5(fuente: IntelliJ IDEA, 2023).

- **Diagramas**

En la figura 10 se ilustra un ejemplo que ofrece una variedad de diagramas que facilitan la visualización, el análisis y la navegación del código. Estos diagramas pueden representar la estructura de clases, métodos, objetos de la base de datos y beans de entidad dentro de la aplicación. Para acceder a la lista de diagramas disponibles, se puede utilizar la opción "Diagramas / Show Diagrama" en el menú contextual.

Figura 10. Ejemplo 6(fuente: IntelliJ IDEA, 2023)

- **Live templates**

Las plantillas activas, una característica destacada, permiten incrementar la eficiencia en la programación. Al emplear abreviaturas predefinidas, se facilita la inserción veloz de estructuras de código convencionales en el desarrollo. Además, se brinda la capacidad de diseñar plantillas personalizadas para la inserción de fragmentos de código de uso frecuente, Considerando lo anterior, se ilustra un ejemplo en la Figura 11.

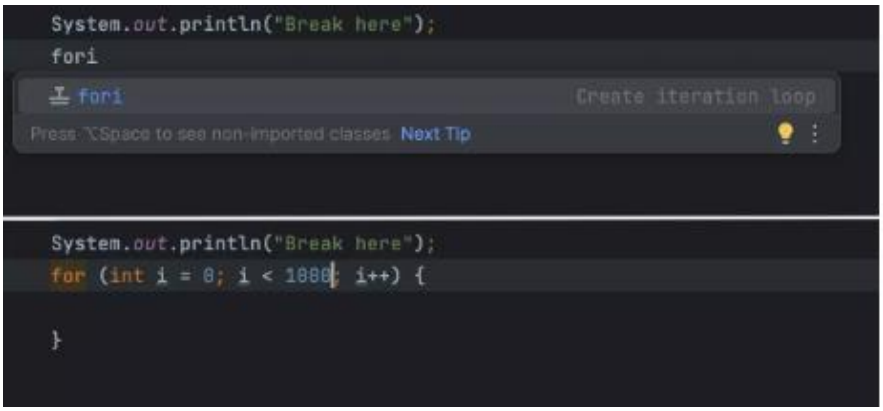

Figura 11. Ejemplo 7(fuente: IntelliJ IDEA, 2023)

- **Refactorización en todo el proyecto**

En la figura 12 se muestra la funcionalidad de refactorización automática en IntelliJ IDEA posibilita la actualización segura y eficiente del código, simplificándolo para mejorar su legibilidad y mantenibilidad. El entorno de desarrollo proporciona una amplia gama de refactorizaciones que facilitan la tarea de cambiar nombres de elementos del código, ajustar firmas de clases o métodos, extraer segmentos de código a métodos y crear variables.

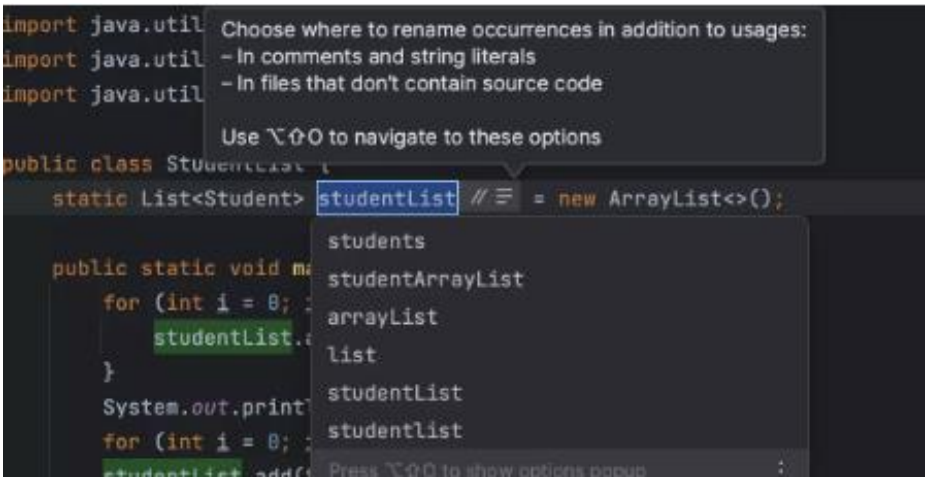

Figura 12. Ejemplo 8(fuente: IntelliJ IDEA, 2023)

Todos los aspectos de IntelliJ IDEA están diseñados para brindarle una experiencia perfecta desde el primer momento. Proporciona acceso rápido a todas las funcionalidades y herramientas integradas importantes para su trabajo, así como a una amplia variedad de opciones de personalización. Puede ajustar todo para que sea compatible con su flujo de trabajo: establezca accesos directos, instale complementos, personalice la interfaz a su gusto, etc.

CAPÍTULO 3: MARCO METODOLÓGICO

En este capítulo se presentan los métodos teóricos destinados a la realización de pruebas de software en el contexto de una plataforma web. Se detallan los procesos y enfoques fundamentales que rigen esta actividad crucial en el ciclo de desarrollo de aplicaciones web tales como son:

- Pruebas unitarias
- Pruebas de integración
- Pruebas funcionales
- Pruebas de extremo a extremo
- Pruebas de aceptación
- Pruebas de rendimiento
- Pruebas de humo

3.1 Pruebas unitarias

La elaboración de pruebas unitarias en el contexto de una plataforma web es un proceso fundamental en el desarrollo de software que sigue una metodología precisa y estructurada.

3.1.1 Definición del alcance

Antes de comenzar con la creación de pruebas unitarias, es importante definir el alcance del proyecto. Esto implica identificar las funcionalidades críticas de la plataforma web que deben ser sometidas a pruebas unitarias.

3.1.2 Selección de herramientas y tecnologías

Seguidamente, se debe seleccionar cuidadosamente las herramientas y tecnologías que se utilizarán para escribir y ejecutar las pruebas unitarias. Las opciones comunes incluyen *frameworks* de pruebas como *JUnit, NUnit o Jasmine*, según el lenguaje de programación y el entorno de desarrollo.

3.1.3 Diseño de casos de prueba

En esta etapa, se elaboran casos de prueba que evalúan la funcionalidad de unidades de código aisladas. Cada caso de prueba debe ser exhaustivo y cubrir diversos escenarios, incluyendo casos límite y situaciones de error.

3.1.4 Implementación de pruebas unitarias

Con los casos de prueba diseñados, se procede a escribir las pruebas unitarias utilizando las herramientas seleccionadas. Estas pruebas deben ser independientes y no depender de otros componentes del sistema. Se enfocan en evaluar la lógica interna de las unidades de código.

3.1.5 Ejecución de pruebas unitarias

Una vez que las pruebas unitarias están implementadas, se ejecutan de manera automatizada utilizando el *framework* de pruebas seleccionado. Esto asegura que las pruebas puedan ser repetidas fácilmente en futuras iteraciones del desarrollo.

3.1.6 Análisis de resultados

Se examinan los resultados de las pruebas unitarias para identificar fallas o errores en el código. Cada error encontrado debe ser documentado cuidadosamente, junto con información detallada sobre cómo reproducirlo.

3.1.7 Depuración y corrección

El desarrollador, basándose en los resultados de las pruebas, procede a depurar y corregir los errores identificados. Es fundamental que cada error sea abordado de manera individual, y se realicen las modificaciones necesarias en el código.

3.1.8 Validación de cobertura

Se verifica que las pruebas unitarias cubran adecuadamente todas las rutas de ejecución del código. Se busca alcanzar una cobertura completa para asegurar la integridad del software.

3.1.9 Automatización de pruebas unitarias

Para mantener la calidad del código a lo largo del tiempo, se automatizan las pruebas unitarias de modo que se ejecuten de forma regular, idealmente después de cada cambio en el código fuente.

3.1.10 Documentación y seguimiento

Finalmente, se documentan las pruebas unitarias y sus resultados de manera completa. Esto permite un seguimiento continuo de la calidad del código y facilita futuras auditorías o revisiones.

3.2 Pruebas de integración

La creación de pruebas de integración en el contexto de una plataforma web es un componente crítico del proceso de aseguramiento de la calidad del software.

3.2.1 Definición de objetivos y alcance

En primer lugar, es esencial definir claramente los objetivos y el alcance de las pruebas de integración. Esto implica identificar los módulos, componentes o servicios que serán evaluados en conjunto para garantizar su correcta interacción.

3.2.2 Identificación de dependencias

Se procede a identificar y documentar todas las dependencias entre los diferentes módulos o componentes que serán sometidos a pruebas de integración.

3.2.3 Diseño de casos de prueba

En este paso se diseñan los casos de prueba detallados que evalúan la interacción entre los diferentes elementos del sistema. Estos casos de prueba deben ser exhaustivos y cubrir diversos escenarios, incluyendo situaciones de entrada/salida, flujos de datos y posibles excepciones.

3.2.3 Configuración del entorno de prueba

Se prepara un entorno de prueba que replica el ambiente de producción tanto como sea posible. Esto puede incluir la configuración de bases de datos de prueba, la implementación de servicios simulados y la preparación de datos de prueba.

3.2.4 Implementación de pruebas de integración

Con los casos de prueba diseñados y el entorno de prueba configurado, se procede a implementar las pruebas de integración. Estas pruebas se enfocan en evaluar cómo interactúan y se comunican los diferentes componentes del sistema.

3.2.5 Ejecución de pruebas de integración

Se ejecutan las pruebas de integración de manera automatizada utilizando las herramientas adecuadas. Durante esta etapa, se registran cuidadosamente los resultados de las pruebas y cualquier anomalía detectada.

3.2.6 Análisis de resultados

Los resultados de las pruebas de integración se analizan minuciosamente para identificar posibles conflictos o errores de comunicación entre los componentes. Cada anomalía se documenta de manera detallada.

3.2.7 Depuración y corrección

Los errores identificados se abordan de manera individual, y se realizan modificaciones en el código o la configuración del sistema para corregirlos. Las correcciones se prueban nuevamente para garantizar que los problemas se hayan resuelto adecuadamente.

3.2.8 Validación de flujo de datos y procesos

Se verifica que los flujos de datos y los procesos de negocio se ejecuten de manera coherente y que los resultados sean consistentes con las expectativas.

3.2.9 Automatización y monitorización continua

Se automatizan las pruebas de integración para que se ejecuten regularmente en el proceso de desarrollo. Además, se establece un sistema de monitorización continua para detectar y abordar problemas de integración en tiempo real.

3.2.10 Documentación y reporte

Todos los aspectos de las pruebas de integración, incluyendo casos de prueba, resultados, correcciones y cambios en la configuración, se documentan de manera detallada. Esto permite un seguimiento continuo de la calidad del sistema y facilita futuras auditorías o revisiones.

3.3 Pruebas funcionales

La creación de pruebas funcionales para una plataforma web es un proceso fundamental en el desarrollo de software que permite verificar que las características y funcionalidades del sistema cumplan con los requisitos especificados.

3.3.1 Definición de requisitos funcionales

En primer lugar, se deben definir de manera precisa y detallada los requisitos funcionales de la plataforma web. Estos requisitos representan las funcionalidades que el sistema debe ofrecer y actuarán como base para el diseño de las pruebas.

3.3.2 Identificación de casos de uso

Se procede a identificar y documentar los casos de uso que representan las interacciones típicas de los usuarios con la plataforma web. Estos casos de uso servirán como base para diseñar los casos de prueba funcionales.

3.3.3 Diseño de casos de prueba

Posteriormente se diseñan casos de prueba específicos para cada uno de los casos de uso identificados. Cada caso de prueba debe incluir datos de entrada, pasos a seguir y resultados esperados, de acuerdo con los requisitos funcionales establecidos.

3.3.4 Configuración del entorno de prueba

Se prepara un entorno de prueba que refleje de manera fiel el ambiente de producción de la plataforma web. Esto incluye la configuración de bases de datos de prueba, la simulación de servicios externos y la creación de datos de prueba.

3.3.5 Implementación de pruebas funcionales

Con los casos de prueba diseñados y el entorno de prueba configurado, se procede a implementar las pruebas funcionales. Estas pruebas se enfocan en evaluar si la plataforma web cumple con los requisitos funcionales especificados.

3.3.6 Ejecución de pruebas funcionales

Las pruebas funcionales se ejecutan de manera automatizada utilizando las herramientas adecuadas. Durante esta etapa, se registran cuidadosamente los resultados de las pruebas y cualquier desviación con respecto a los resultados esperados.

3.3.7 Análisis de resultados

Los resultados de las pruebas funcionales se analizan minuciosamente para identificar cualquier discrepancia entre el comportamiento real y los requisitos funcionales. Cada desviación se documenta de manera detallada.

3.3.8 Depuración y corrección

Los problemas identificados se abordan de manera individual, y se realizan modificaciones en el código o la configuración del sistema para corregirlos. Las correcciones se prueban nuevamente para garantizar que los problemas se hayan resuelto adecuadamente.

3.3.9 Validación de la funcionalidad interdependiente

Se verifica que las funcionalidades interactúen de manera coherente y que los resultados de una función no afecten negativamente a otras. Se presta especial atención a la interoperabilidad de los componentes.

3.3.10 Automatización y repetición

Se automatizan las pruebas funcionales para que puedan ser ejecutadas de forma regular en el proceso de desarrollo, especialmente después de cada cambio en el código. La repetición de las pruebas garantiza la consistencia y la calidad del sistema en evolución.

3.3.11 Documentación y reporte

Todos los aspectos de las pruebas funcionales, incluyendo casos de prueba, resultados, correcciones y cambios en la configuración, se documenta de principio a fin. Esto permite un seguimiento continuo de la calidad del sistema y facilita futuras auditorías o revisiones.

3.4 Pruebas de extremo a extremo

La creación de pruebas de extremo a extremo (E2E) para una plataforma web es un proceso crítico en el desarrollo de software que permite verificar el funcionamiento integral del sistema desde el punto de vista del usuario

3.4.1 Definición de escenario de usuario

En primer lugar, se identifican y definen los escenarios de usuario que representan las interacciones típicas con la plataforma web. Estos escenarios deben ser lo más cercanos posible a las acciones reales que los usuarios llevarían a cabo en el sistema.

3.4.2 Identificación de flujos de trabajo

Se procede a identificar y documentar los flujos de trabajo completos que comprenden desde la acción inicial del usuario hasta el resultado final deseado. Estos flujos actuarán como base para el diseño de las pruebas de extremo a extremo.

3.4.3 Diseño de casos de prueba E2E

Después se diseñan casos de pruebas específicos para cada uno de los flujos de trabajo identificados. Cada caso de prueba debe incluir pasos detallados que describan las acciones del usuario, así como los resultados esperados, teniendo en cuenta los criterios de aceptación.

3.4.4 Configuración del entorno de prueba

Se prepara un entorno de prueba que sea una réplica lo más precisa posible del entorno de producción de la plataforma web. Esto incluye la configuración de bases de datos de prueba, la simulación de servicios externos y la creación de datos de prueba realistas.

3.4.5 Implementación de pruebas E2E

Con los casos de prueba E2E diseñados y el entorno de prueba configurado, se procede a implementar las pruebas de extremo a extremo. Estas pruebas se enfocan en simular las acciones del usuario real y evaluar todo el flujo de trabajo.

3.4.6 Ejecución de pruebas E2E

Las pruebas de extremo a extremo se ejecutan de manera automatizada utilizando herramientas de automatización de pruebas como *Selenium, Puppeteer o Cypress.* Durante esta etapa, se registran cuidadosamente los resultados de las pruebas y cualquier desviación con respecto a los resultados esperados.

3.4.7 Análisis de resultados

Los resultados de las pruebas E2E se analizan en detalle para identificar cualquier discrepancia entre el comportamiento real y los resultados esperados. Cada variante se documenta de manera detallada.

3.4.8 Depuración y corrección

Los problemas identificados se abordan de manera individual, y se realizan modificaciones en el código o la configuración del sistema para corregirlos. Las correcciones se prueban nuevamente para garantizar que los problemas se hayan resuelto adecuadamente.

3.4.9 Validación de flujos de trabajo complejos

Se verifica que los flujos de trabajo complejos que involucran múltiples acciones del usuario se ejecuten de manera coherente y que los resultados sean consistentes con las expectativas.

3.4.10 Automatización y repetición

Se automatizan las pruebas E2E para que se ejecuten regularmente en el proceso de desarrollo, especialmente después de cada cambio en el código. La repetición de las pruebas garantiza la consistencia y la calidad del sistema en evolución.

3.4.11 Documentación y reporte

Todos los aspectos de las pruebas E2E, incluyendo casos de prueba, resultados, correcciones y cambios en la configuración, se documentan de manera exhaustiva. Esto permite un seguimiento continuo de la calidad del sistema y facilita futuras auditorías o revisiones.

3.5 Pruebas de aceptación

La creación de pruebas de aceptación para una plataforma web es una fase crucial en el desarrollo de software, donde se verifica si el sistema cumple con los criterios de aceptación definidos por el cliente o el usuario final.

3.5.1 Definición de criterios de aceptación

En primer lugar, se deben definir de manera precisa y detallada los criterios de aceptación para la plataforma web. Estos criterios representan los requisitos específicos que el sistema debe cumplir para considerarse exitoso.

3.5.2 Identificación de escenarios de uso

Se procede a identificar y documentar los escenarios de uso que representan las interacciones típicas de los usuarios con la plataforma web. Estos escenarios ayudarán a diseñar casos de prueba específicos para los criterios de aceptación.

3.5.3 Diseño de casos de prueba de aceptación

Luego se diseñan casos de prueba específicos para cada uno de los criterios de aceptación identificados. Cada caso de prueba debe incluir pasos detallados que describan las acciones del usuario y los resultados esperados en función de los criterios de aceptación definidos.

3.5.4 Configuración del entorno de prueba

Se prepara un entorno de prueba que sea una réplica precisa del entorno de producción de la plataforma web. Esto incluye la configuración de bases de datos de prueba, la simulación de servicios externos y la creación de datos de prueba realistas.

3.5.5 Implementación de pruebas de aceptación

Con los casos de prueba de aceptación diseñados y el entorno de prueba configurado, se procede a implementar las pruebas de aceptación. Estas pruebas se enfocan en simular las acciones del usuario real y evaluar si el sistema cumple con los criterios de aceptación definidos.

3.5.6 Ejecución de pruebas de aceptación

Las pruebas de aceptación se ejecutan de manera automatizada utilizando herramientas apropiadas. Durante esta etapa, se registran cuidadosamente los resultados de las pruebas y cualquier desviación con respecto a los resultados esperados.

3.5.7 Análisis de resultados

Los resultados de las pruebas de aceptación se analizan minuciosamente para identificar cualquier discrepancia entre el comportamiento real y los criterios de aceptación definidos. Cada una de las variantes se documenta de manera detallada.

3.5.8 Depuración y corrección

Los problemas identificados se abordan de manera individual, y se realizan modificaciones en el código o la configuración del sistema para corregirlos. Las correcciones se prueban nuevamente para garantizar que los problemas se hayan resuelto adecuadamente.

3.5.9 Validación de casos de uso completos

Se verifica que los casos de uso completos que involucran múltiples acciones del usuario se ejecuten de manera coherente y que los resultados sean consistentes con los criterios de aceptación.

3.5.10 Automatización y repetición

Se automatizan las pruebas de aceptación para que se ejecuten regularmente en el proceso de desarrollo, especialmente después de cada cambio en el código. La repetición de las pruebas garantiza la consistencia y la calidad del sistema en evolución.

3.5.11 Documentación y reporte

Todos los aspectos de las pruebas de aceptación, incluyendo casos de prueba, resultados, correcciones y cambios en la configuración, se documentan de manera exhaustiva. Esto permite un seguimiento continuo de la calidad del sistema y facilita futuras auditorías o revisiones.

3.6 Pruebas de rendimiento

La creación de pruebas de rendimiento para una plataforma web es un proceso fundamental en el desarrollo de software que busca evaluar y garantizar la capacidad del sistema para operar eficientemente bajo carga y condiciones de uso esperadas.

3.6.1 Definición de los objetivos de rendimiento

En primer lugar, se deben definir claramente los objetivos de rendimiento que se desean lograr para la plataforma web. Estos objetivos pueden incluir tiempos de respuesta, capacidad de concurrencia, utilización de recursos, y otros indicadores clave de rendimiento.

3.6.2 Identificación de los escenarios de carga

Se procede a identificar y documentar los escenarios de carga que representan las condiciones de uso típicas y extremas del sistema. Estos escenarios ayudarán a diseñar las pruebas de rendimiento.

3.6.3 Diseño de pruebas de rendimiento

Enseguida se diseñan pruebas de rendimiento específicas para cada uno de los escenarios de carga identificados. Estas pruebas deben simular la actividad de usuarios reales y ajustarse a los objetivos de rendimiento establecidos.

3.6.4 Configuración del entorno de prueba

Se prepara un entorno de prueba que refleje el ambiente de producción de la plataforma web, incluyendo la infraestructura de hardware y software. Esto puede requerir la configuración de servidores, bases de datos y redes de prueba.

3.6.5 Implementación de pruebas de rendimiento

Con los casos de prueba de rendimiento diseñados y el entorno de prueba configurado, se procede a implementar las pruebas. Estas pruebas se enfocan en medir el rendimiento del sistema bajo carga simulada.

3.6.7 Ejecución de pruebas de rendimiento

Las pruebas de rendimiento se ejecutan de manera automatizada utilizando herramientas especializadas como Apache JMeter, Gatling, o locust.io. Durante esta etapa, se recopilan métricas (velocidad, la escalabilidad, la capacidad y la estabilidad de la aplicación) de rendimiento clave.

3.6.8 Análisis de resultados

Los resultados de las pruebas de rendimiento se analizan detalladamente para evaluar si el sistema cumple con los objetivos de rendimiento definidos. Se identifican posibles cuellos de botella y se registran las métricas relevantes (velocidad, la escalabilidad, la capacidad y la estabilidad de la aplicación).

3.6.9 Optimización y ajustes

Si se detectan problemas de rendimiento durante las pruebas, se realizan ajustes en el código, la configuración o la infraestructura para mejorar el rendimiento. Estos cambios se prueban y validan nuevamente.

3.6.10 Pruebas de carga sostenida

Se realizan pruebas de carga sostenida para evaluar la estabilidad y la resistencia del sistema bajo cargas prolongadas. Esto permite identificar problemas de memoria o fugas de recursos.

3.6.11 Automatización y monitorización continuación

Se automatizan las pruebas de rendimiento para que se ejecuten de forma regular, y se establece un sistema de monitorización continua para detectar y abordar problemas de rendimiento en tiempo real.

3.6.12 Documentación y reporte

Todos los aspectos de las pruebas de rendimiento, incluyendo casos de prueba, resultados, optimizaciones y ajustes, se documentan de manera exhaustiva. Esto permite un seguimiento continuo del rendimiento del sistema y facilita futuras auditorías o revisiones.

3.7 Pruebas de humo

La creación de pruebas de humo (*smoke tests*) para una plataforma web es un paso esencial en el proceso de aseguramiento de la calidad del software.

3.6.1 Definición de los objetivos de las pruebas de humo

En primer lugar, se deben definir claramente los objetivos de las pruebas de humo. Estos objetivos pueden incluir la verificación de la estabilidad básica del sistema, la funcionalidad esencial, la conectividad con bases de datos y servicios críticos, entre otros aspectos cruciales.

3.6.2 Identificación de las características críticas

Se procede a identificar y documentar las características críticas de la plataforma web que deben ser evaluadas durante las pruebas de humo. Estas características representan las funciones básicas y esenciales del sistema.

3.6.3 Diseño de casos de prueba de humo

Seguidamente se diseñan casos de prueba específicos para cada una de las características críticas identificadas. Cada caso de prueba debe incluir pasos detallados que describan las acciones a ejecutar y los resultados esperados.

3.6.4 Configuración del entorno de prueba

Se prepara un entorno de prueba que refleje el ambiente de producción de la plataforma web en la medida de lo posible. Esto puede incluir la configuración de servidores, bases de datos y datos de prueba.

3.6.5 Implementación de pruebas de humo

Con los casos de prueba de humo diseñados y el entorno de prueba configurado, se procede a implementar las pruebas. Estas pruebas se enfocan en verificar que las características críticas funcionen correctamente.

3.6.6 Ejecución de pruebas de humo

Las pruebas de humo se ejecutan de manera automatizada utilizando herramientas apropiadas. Durante esta etapa, se registran los resultados de las pruebas y cualquier variante con respecto a los resultados esperados.

3.6.7 Análisis de resultados

Los resultados de las pruebas de humo se analizan en detalle para identificar cualquier discrepancia entre el comportamiento real y los resultados esperados. Se priorizan los problemas críticos.

3.6.8 Reporte de problemas críticos

Si se encuentran problemas críticos durante las pruebas de humo, se documentan de manera detallada y se comunican inmediatamente al equipo de desarrollo para su corrección.

3.6.9 Validación de estabilidad básica

Se verifica que el sistema sea estable y pueda iniciar sin errores graves o bloqueos, lo que es esencial para garantizar una experiencia inicial satisfactoria para los usuarios.

3.6.10 Automatización y repetición

Se automatizan las pruebas de humo para que se ejecuten regularmente, idealmente antes de cada despliegue o actualización del sistema. La repetición de las pruebas asegura que las características críticas sigan funcionando correctamente con cada cambio.

3.6.11 Documentación y reporte

Todos los aspectos de las pruebas de humo, incluyendo casos de prueba, resultados, problemas críticos y cambios en la configuración, se documentan de manera exhaustiva. Esto permite un seguimiento continuo de la calidad del sistema y facilita futuras auditorías o revisiones.

3.8 Metodología de Pruebas

Un proceso de pruebas formales, está compuesto, cuando menos por los siguientes 5 tipos de etapas (Javier zapata, 13 de enero del 2013):

1. Planeación de pruebas.
2. Diseño de pruebas.
3. Implementación de pruebas.
4. Evaluación de criterios de salida.
5. Cierre del proceso.

3.8.1 Planeación de Pruebas.

Es la etapa en donde se ejecutan las primeras actividades correspondientes al proceso de pruebas y tiene como resultado un entregable denominado plan de pruebas el cual debe estar conformado en cuando menos por aspectos tales como (Javier zapata, 13 de enero del 2013):

- **Alcance de la prueba:** Determina que funcionalidades del producto y/o software serán probadas durante el transcurso de la prueba. Este listado de funcionalidades a probar se extrae con base a un análisis de riesgos realizado de manera previa, que tienen en cuenta variables tales como el impacto que podría ocasionar la falla de una funcionalidad y la probabilidad de falla de una funcionalidad. Producto de este análisis, se cuenta con información adicional que permite determinar además del

alcance detallado del proceso de pruebas, la prioridad con la que las funcionalidades deben probarse y la profundidad de las pruebas.

- **Tipos de Prueba:** En este punto se debe determinar qué tipos de pruebas requeriría el producto. No todos los productos de software requieren la aplicación de todos los tipos de pruebas que existen, por esta razón, es estrictamente necesario que el líder de pruebas se plantee preguntas que le permitan determinar qué tipos de prueba son aplicables al proyecto en evaluación. Los posibles tipos de prueba a aplicar son: pruebas de stress, pruebas de rendimiento, pruebas de carga, pruebas funcionales, pruebas de usabilidad, pruebas de regresión, entre otros.
- **Estrategia de Pruebas:** Teniendo en cuenta que no es viable probar con base a todas las posibles combinaciones de datos, es necesario determinar a través de un análisis de riesgos sobre que funcionalidades debemos centrar nuestra atención. Adicionalmente, una buena estrategia de pruebas debe indicar los niveles de pruebas (ciclos) que aplicaremos y la intensidad o profundidad a aplicar para cada nivel de pruebas definido. En este punto también es importante definir los criterios de entrada y salida para cada ciclo de pruebas a ejecutar.
- **Criterios de Salida:** Entre las partes involucradas en el proceso, se define de manera formal, bajo qué condiciones se puede considerar que una actividad de pruebas fue finalizada. Los criterios de salida se deben definir para cada nivel de pruebas a ejecutar. Algunos ejemplos de criterios de salida que pueden ser utilizados son: porcentaje de funcionalidades de alto riesgo probadas con éxito, número defectos críticos y/o mayores aceptados, etc.

- **Otros aspectos:** Tal y como se realiza en cualquier plan de proyecto, se debe incluir una estimación de tiempos, los roles y/o recursos que harán parte del proceso, la preparación del entorno de pruebas, cronograma base, etc.

3.8.2 Diseño de Pruebas

Posteriormente se elabora y aprueba el plan de pruebas, el equipo de trabajo debe iniciar el análisis de toda la documentación existente con respecto al sistema, con el objeto de iniciar el diseño de los casos de prueba. Los entregables claves para iniciar este diseño pueden ser: casos de uso, historias de usuario, arquitectura del sistema, diseños, manuales de usuario (si existen), manuales técnicos (si existen). El diseño de los casos, debe considerar la elaboración de casos positivos y negativos. Los casos de prueba negativos permiten validar cómo se comporta el sistema ante situaciones atípicas y permite verificar la robustez del sistema, atributo que constituye uno de los requerimientos no funcionales indispensable para cualquier software. Por último, es necesario definir cuáles son los datos de prueba necesarios para la ejecución de los casos de prueba diseñados.

3.8.3 Implementación y Ejecución de Pruebas

La ejecución de pruebas debe iniciar con la creación de los datos de prueba necesarios para ejecutar los casos de prueba diseñados. La ejecución de estos casos, puede realizarse de manera manual o automatizada; en cualquiera de los casos, cuando se detecte un fallo en el sistema, este debe ser documentado y registrado en una herramienta que permita gestionar los defectos (*Bug Tracker*). Una vez el defecto ha sido corregido por la firma desarrolladora en su respectivo proceso de depuración, es necesario realizar un re-test que permita confirmar que el defecto fue solucionado de manera exitosa. Por último, es indispensable ejecutar un ciclo de regresión que nos

permita garantizar, que los defectos corregidos en el proceso de depuración de la firma, no hayan desencadenado otros tipos de defectos en el sistema.

3.8.4 Evaluación de Criterios de Salida

Posteriormente los criterios de salida son necesarios para determinar si es posible dar por finalizado un ciclo de pruebas. Para esto, es conveniente definir una serie de métricas que permitirán al finalizar un proceso de pruebas, comparar los resultados obtenidos contra las métricas definidas, sí los resultados obtenidos no superan las métricas definidas, no es posible continuar con el siguiente ciclo de pruebas. Existen varios tipos de criterios de salida dentro de los cuales se pueden mencionar: cubrimiento de funcionalidades en general, cubrimiento de funcionalidades críticas para el sistema, Número de defectos críticos y mayores detectados, etc. También es importante aclarar que el proceso de pruebas puede ser suspendido y/o paralizado, debido entre otros, a aspectos relacionados con el presupuesto y la calidad mínima del sistema requerida para el inicio formal de pruebas.

3.8.5 Cierre del proceso

durante este periodo de cierre el cual históricamente se ha comprobado que se le destina muy poco tiempo en la planeación, se deben cerrar las incidencias reportadas, se debe verificar si los entregables planeados han sido entregados y aprobados, se deben finalizar y aprobar los documentos de soporte de prueba, analizar las lecciones aprendidas para aplicar en futuros proyectos, etc.

CAPÍTULO 4: MARCO OPERATIVO

En este capítulo se presentan todas las ejecuciones que corresponden a las pruebas de software enfocado a una plataforma web, que serán usados para la realización del proyecto. Las pruebas de software que se abordarán en este capítulo son:

1. Pruebas unitarias
2. Pruebas de integración
3. Pruebas funcionales
4. Pruebas de extremo a extremo
5. Pruebas de aceptación
6. Pruebas de rendimiento
7. Pruebas de humo

4.1 Pruebas unitarias

4.2 PHPUnit

Una vez que se ha instalado el entorno de desarrollo Visual Studio Code, se procede a la incorporación del componente denominado PHPUNIT al interior de la mencionada aplicación informática.

PHPUnit es un entorno para realizar pruebas unitarias en el lenguaje de programación PHP. PHPUnit es un framework de la familia xUnit originada con SUnit de Kent Beck. PHPUnit se puede encontrar en GitHub y ha sido creado por Sebastian Bergmann

4.3 Proceso de instalación de PHPUnit

Los siguientes pasos se describen a continuación para la instalación de PHPUnit:

1. En un primer paso, se procede a iniciar el entorno de desarrollo denominado Visual Studio Code. Una vez que se ha confirmado el acceso exitoso a este programa

informático, el usuario dirige su atención al segmento correspondiente a las extensiones, figura 13.

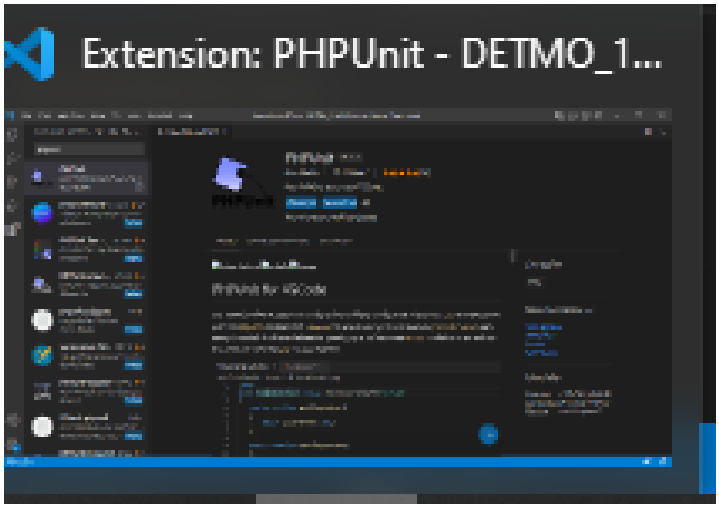

Figura 13. Phpunit (fuente: elaboración propia, 2023)

2. Una vez ubicado en la sección previamente mencionada, el usuario procederá a buscar la extensión denominada "PHPUNIT" tal como se ilustra en la figura 14 y permitirá que el proceso de carga se efectúe de acuerdo a las instrucciones proporcionadas.

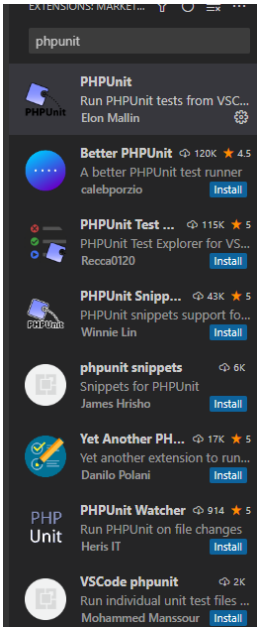

Figura 14. Descarga de phpunit (fuente: elaboración propia, 2023)

3. Cuando se identifique la extensión, el usuario procederá a su acceso de acuerdo a las indicaciones proporcionadas, y posteriormente seleccionará la opción "instalar". Una vez que la instalación haya sido completada, la extensión se mostrará en la interfaz del programa, tal como se ilustra en la figura 15.

Figura 15. Instalación de phpunit (fuente: elaboración propia, 2023)

4. Para llevar a cabo la ejecución de un test en Visual Studio de manera expedita y sencilla, el procedimiento se inicia mediante la activación de la combinación de teclas CONTROL + MAYÚSCULAS + P. Esta acción desencadena la aparición de un selector que exhibe una lista exhaustiva de comandos disponibles para su ejecución en el entorno de Visual Studio. En presencia de la instalación de la biblioteca PHPUnit, al ingresar el comando "PHPUnit", se desplegarán en pantalla todos los comandos potencialmente ejecutables, entre los cuales figura la opción "PHPUnit Test". Esta última opción permite llevar a cabo la ejecución de las pruebas correspondientes al archivo en el que el usuario se encuentre posicionado, tal y como se ilustra en la Figura 16.

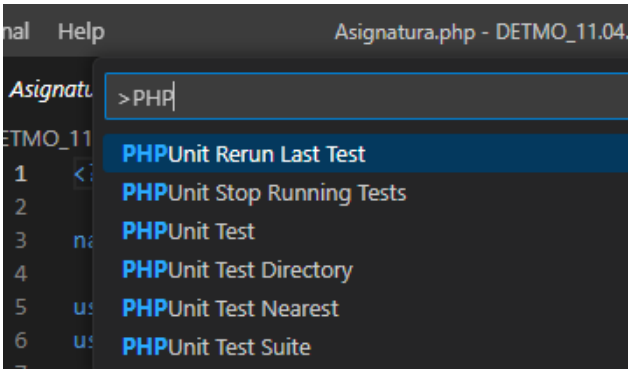

Figura 16. Extensiones de phpunit (fuente: elaboración propia, 2023).

5. Al activar el mencionado comando, se suscitará una interrogante dirigida hacia el usuario, con el propósito de discernir la preferencia por la ejecución de la totalidad de la clase denominada "Asignatura", en consideración de la eventualidad de la existencia de múltiples pruebas unitarias, o, alternativamente, la selección de una única función, tal como se ilustra en la representación gráfica consignada en la figura 17.

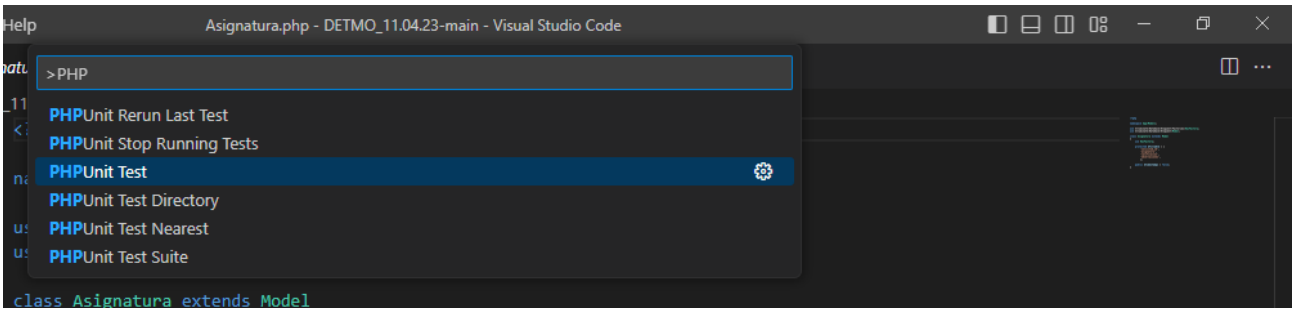

Figura 17. Extensiones de phpunit test (fuente: elaboración propia, 2023).

4.2 Pruebas de integración

Para la elaboración de las pruebas de integración, el usuario procede a emplear las herramientas Jira y QMetry. Inicialmente, se inicia sesión en la plataforma, tal y como se ilustra en la Figura 18. Subsecuentemente, una vez en el interior del sistema, se procede a la exploración minuciosa de cada ventana que conforma dicho entorno, con el propósito de facultar al usuario la comprensión de la función esencial de cada una de estas interfaces.

Figura 18 Inicio de atlassian (fuente: Elaboración propia, 2023)

Dado que todo se encuentra en óptimas condiciones al iniciar sesión, se procede a la inclusión de un producto, con el objetivo de verificar su funcionamiento exento de fallos. Este proceso se extiende a la comprobación del adecuado desempeño de las ventanas, entre otros aspectos, tal y como se ilustra en la Figura 19.

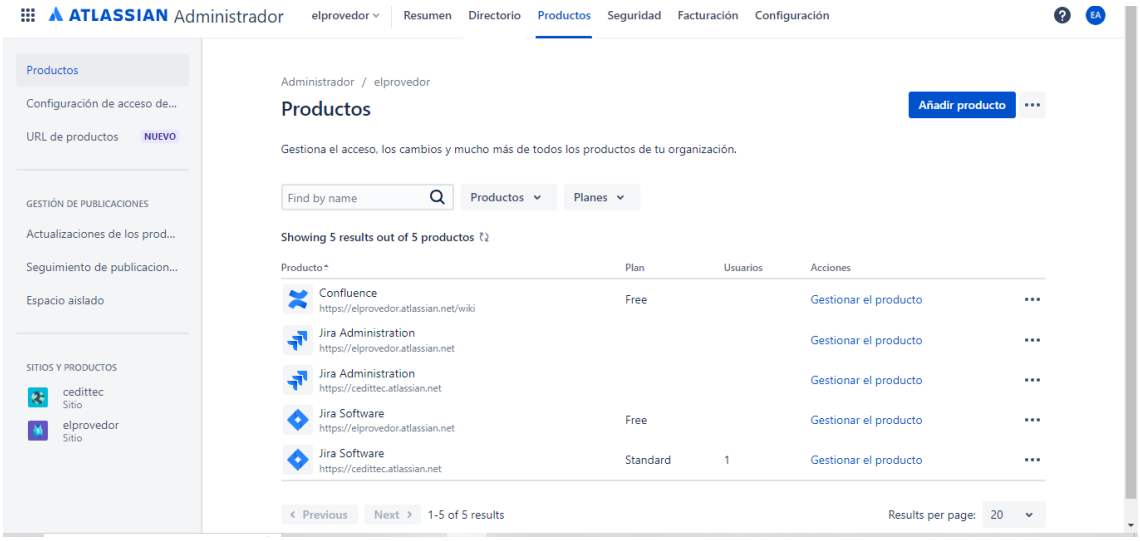

Figura 19 Añadir producto (fuente: Elaboración propia, 2023)

Concluido el procedimiento previamente mencionado, se procede a la selección del producto entre los que se encuentran representados en la Figura 20. Posteriormente, se opta por la acción de continuar.

En el siguiente apartado, se procederá a detallar minuciosamente cada una de las funciones que se encuentran a disposición del usuario, con el propósito de otorgar una comprensión integra de las opciones disponibles.

Confluence: Crea, colabora y mantén todo en un único espacio

1. Promueve la comunicación
2. Elimina los grupos aislados
3. Reduce los cambios de contexto

Jira product Discovery: Reúne a todo el mundo y consigue los resultados que buscas

1. Registra todos los datos, las ideas de productos, las solicitudes y la información
2. Prioriza fácilmente y mantén a todo el mundo involucrado
3. El descubrimiento y la entrega en la misma plataforma

Jira service management: Desata el potencial de los equipos de desarrollo, operaciones, soporte y negocios con ITSM de alta velocidad

1. Ofrece valor rápido
2. Aporta visibilidad a tu trabajo
3. Conecta el desarrollo y las operaciones

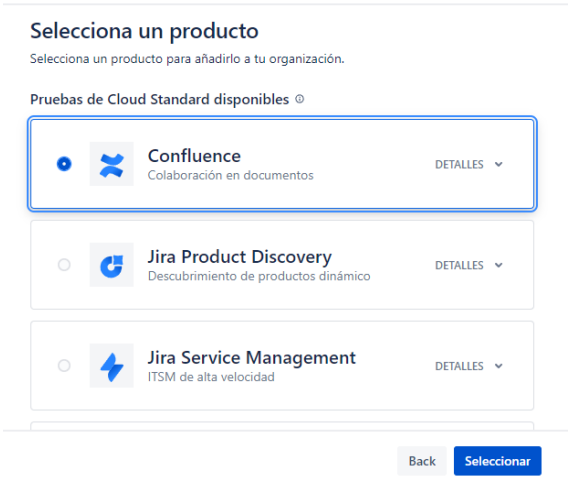

Figura 20 Selección del producto (fuente: Elaboración propia, 2023)

Al realizar la selección, se procede a la inserción de un nombre apropiado para el proyecto, conforme a lo ilustrado en la Figura 21. En esta etapa, se ha optado por el nombre indicado en la figura con el propósito de continuar con el desarrollo del proceso de prueba pertinente.

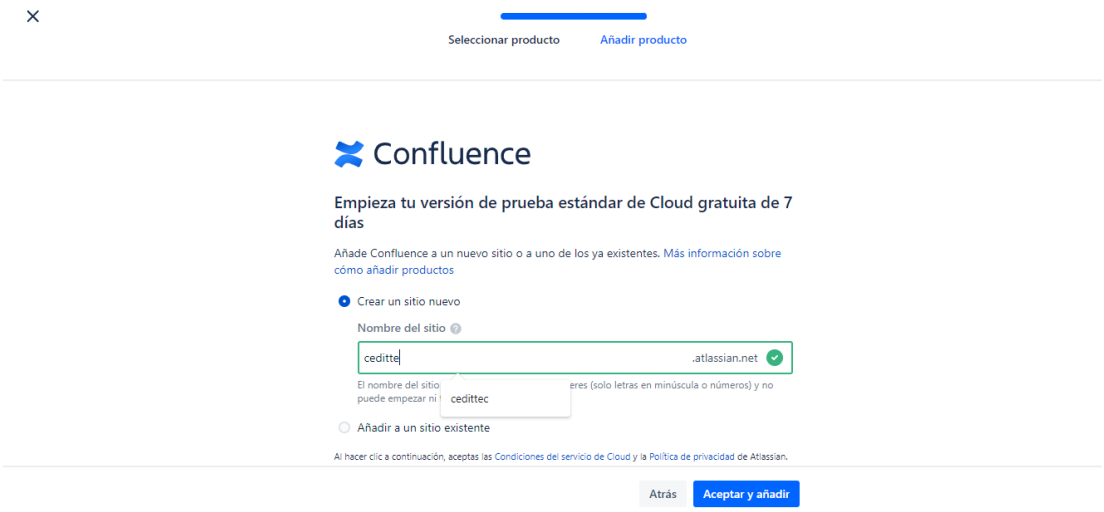

Figura 21 Nombre del producto (fuente: Elaboración propia, 2023)

Una vez que se verifique la correcta configuración, la página procederá a cargar con el propósito de procesar las instrucciones especificadas, tal como se ilustra en la figura 22.

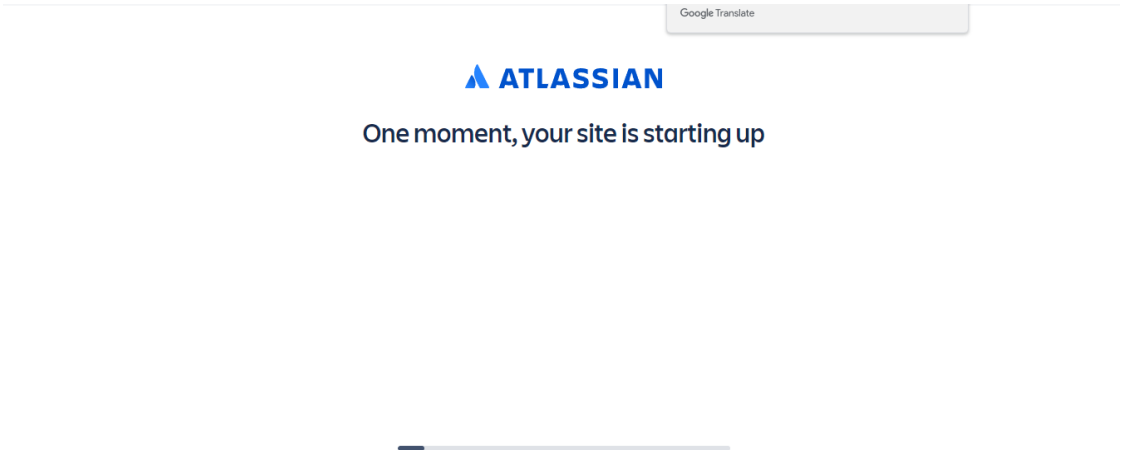

Figura 22 Proceso de verificación (Fuente: Elaboración propia, 2023)

Una vez que el proceso en la página web ha sido ejecutado con precisión y sin inconvenientes, esta indicará de manera automática su finalización, permitiendo así avanzar hacia la siguiente fase, la cual involucra la configuración del sistema, tal y como se ilustra en la figura 23.

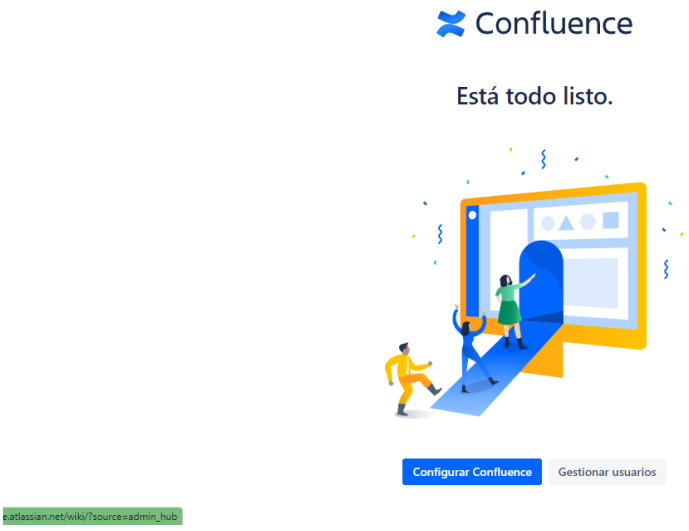

Figura 23 Finalización del registro del producto (Elaboración propia, 2023)

Una vez concluida la preparación, el sistema desplegará una notificación en la que se brindará la posibilidad de ampliar la información relacionada con las actividades del equipo y, de este modo, facilitar la colaboración con otros individuos igualmente involucrados. Esta configuración particular se encuentra representada en la figura 24.

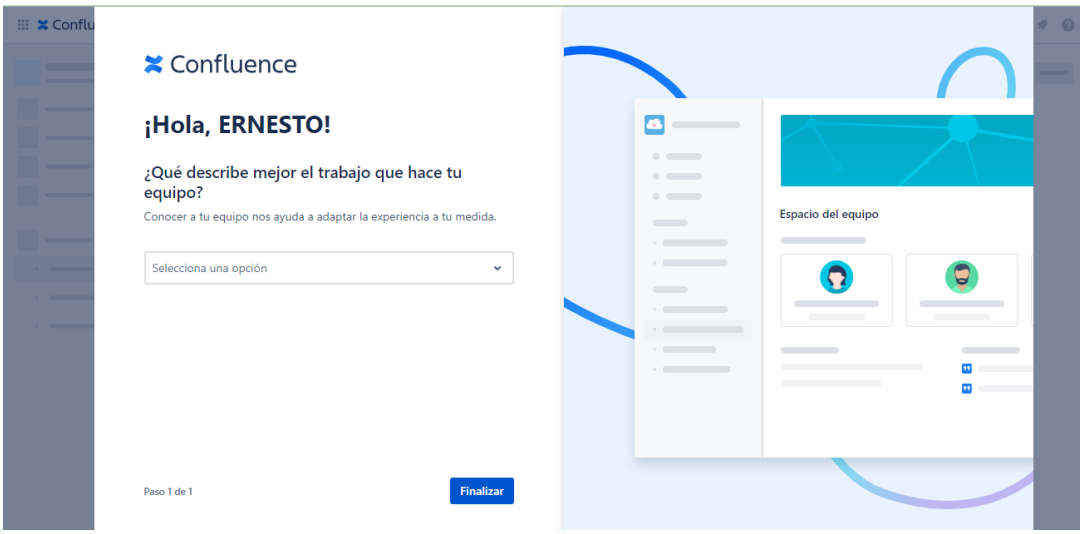

Figura 24 Bienvenida del equipo de trabajo (fuente: Elaboración propia, 2023)

Una vez que se haya completado adecuadamente el procedimiento, se desplegará una ventana que habilitará la ejecución de las pruebas necesarias. El tablero Kanban, en este contexto, se erige como una herramienta habilitadora para la creación de incidencias, facilitando de esta forma su ejecución y permitiendo la identificación de eventuales fallos que pudiesen suscitarse. La Figura 25 presenta de manera formal el tablero designado para llevar a cabo las pruebas, donde se desarrolla dicho proceso con detenimiento.

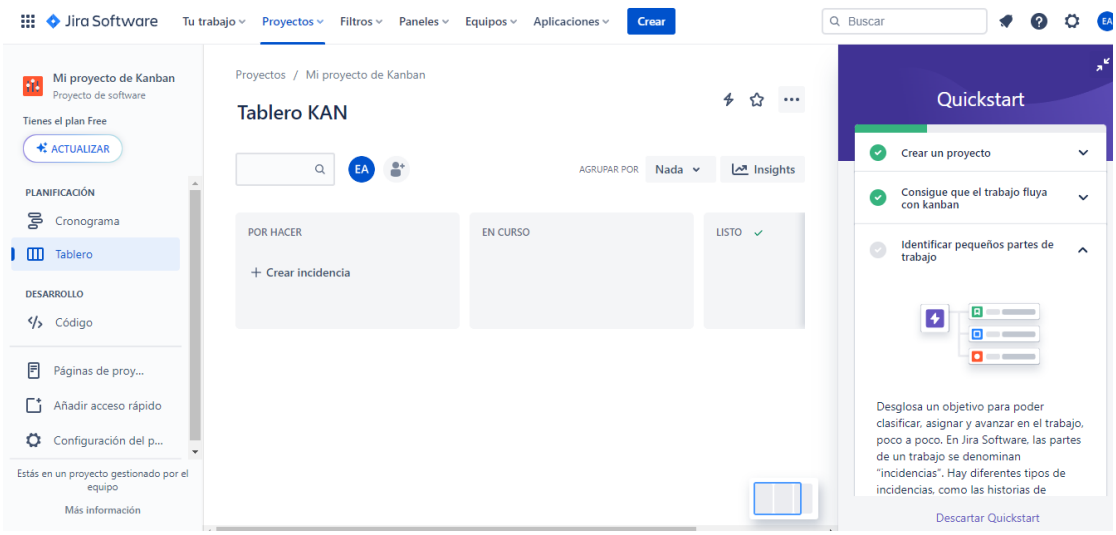

Figura 25 Tablero de pruebas (fuente: Elaboración propia, 2023)

4.3 Pruebas funcionales

Las pruebas funcionales se implementan siguiendo un enfoque de dos pasos, permitiendo así la interacción con una prueba que describe una función específica. Dentro del ámbito funcional, se encuentra un código que es responsable de habilitar la funcionalidad requerida. Esta metodología se presenta de la siguiente manera:

1. Se procede a verificar que la totalidad de los componentes que conforman el software operen de manera adecuada. Un ilustrativo ejemplo en este sentido es el proceso de inicio de sesión o la validación de ventanas interconectadas, cuya correcta funcionalidad es esencial, tal como se ilustra en la figura 26.

Figura 26. Correo de director del tecnológico (fuente: Elaboración propia, 2023).

2. Con el propósito de alcanzar los objetivos de las pruebas funcionales, se procede a acceder al sistema en calidad de administrador. Es importante destacar que el correo electrónico y la contraseña utilizados por defecto para la autenticación se encuentran bajo la administración de dicho usuario. En virtud de esta premisa, se realiza la inserción de nuevos usuarios en el sistema. Es imperativo subrayar que la habilitación de acceso por parte del administrador es un requisito esencial para garantizar el éxito

de esta funcionalidad. Es en esta etapa donde adquieren relevancia las pruebas funcionales, ya que se enfocan en la validación de posibles escenarios que puedan surgir en un contexto real.

Como se observa en la representación de la figura 27, se ejecuta un proceso de registro de usuarios, el cual constituye una parte integral de la funcionalidad dirigida a cada usuario del sistema.

Figura 27. Registro de directores institucionales (fuente: Elaboración propia, 2023).

4.4 Pruebas de extremo a extremo

Se enfatiza la esencia de la funcionalidad en el proceso de ejecución desde un punto de origen hasta un punto de destino. El objetivo fundamental radica en la verificación de la correcta transmisión y recepción de datos en cada uno de los puntos de paso intermedios. A continuación, se presenta un ejemplo ilustrativo de cómo se lleva a cabo la ejecución del sistema de extremo a extremo:

- En el procedimiento de ejecución de la mencionada prueba, se efectúa la identificación de la funcionalidad de extremo a extremo. En la figura 28, todos los usuarios se valen de sus direcciones de correo electrónico como elemento

identificativo. Cabe destacar que el administrador se erige como la entidad responsable del registro de los usuarios, otorgándoles así acceso al sistema.

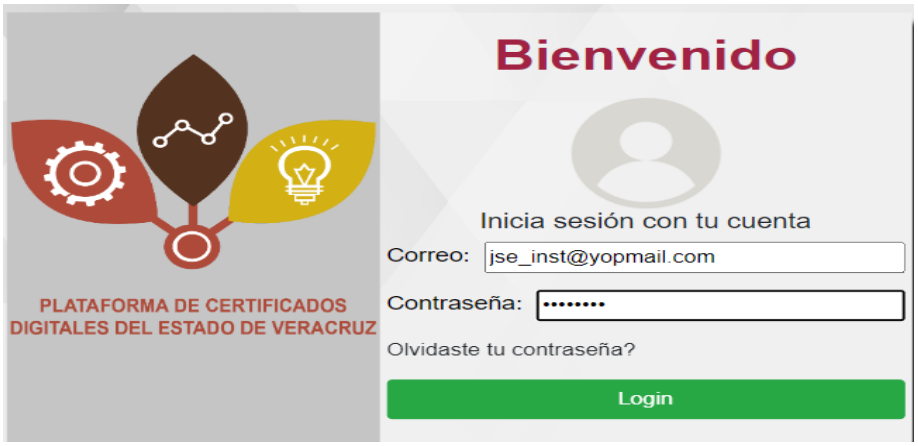

Figura 28. Correo de jefes de servicios escolares (fuente: Elaboración propia, 2023).

Una vez que el proceso de registro de estos usuarios sea ejecutado con éxito, se generará una notificación en la bandeja de mensajes, tal como se ilustra en la representación gráfica proporcionada en la figura 29.

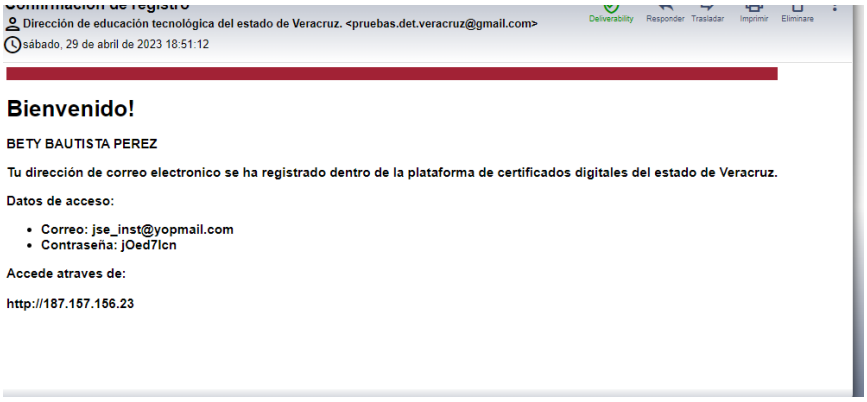

Figura 29. Correo de credenciales para acceso a la plataforma web (fuente: Elaboración propia, 2023).

Cuando el administrador procede a realizar la inserción adecuada de todos los usuarios en el sistema, cada individuo correspondiente deberá recibir una notificación por correo electrónico a modo de validación, confirmándoles la exactitud de sus datos registrados. En este punto, se da lugar a la ejecución de un proceso integral de extremo a extremo, donde se establece una

interrelación de múltiples funciones desde un punto de origen hacia otro, como se ilustra en la figura 30.

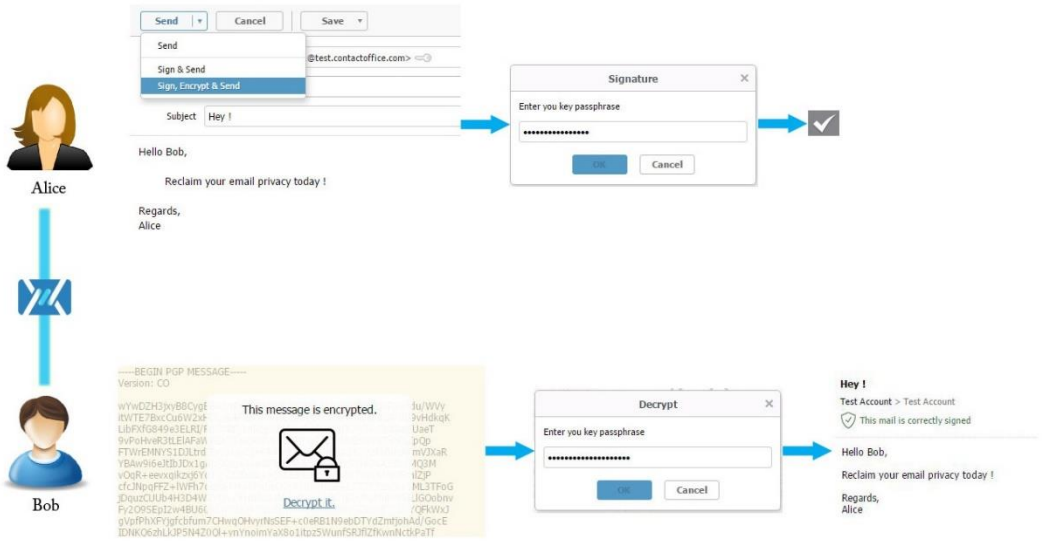

Figura 30. Correo, ejemplo de Gmail (fuente: Vladimir, 2 de marzo del 2023).

La verificación de extremo a extremo de una cuenta de Gmail incluirá los siguientes pasos tales como son: (pmoinformatica.com. (s. f.).

1. Lanzamiento de una página de inicio de sesión de Gmail a través de URL.
2. Iniciar sesión en la cuenta de Gmail con credenciales válidas.
3. Accediendo a la Bandeja de entrada. Abrir correos electrónicos leídos y no leídos.
4. Redactar un nuevo correo electrónico, responder o reenviar un correo electrónico.
5. Abrir elementos enviados y consultar correos electrónicos.
6. Comprobación de correos electrónicos en la carpeta Spam
7. Cerrar sesión en la aplicación Gmail haciendo clic en 'cerrar sesión'

4.5 Prueba de aceptación

Para llevar a cabo las pruebas de aceptación, se ejecutan una serie de procedimientos que implican el acceso al servidor, tal como se ilustra en la figura 31.

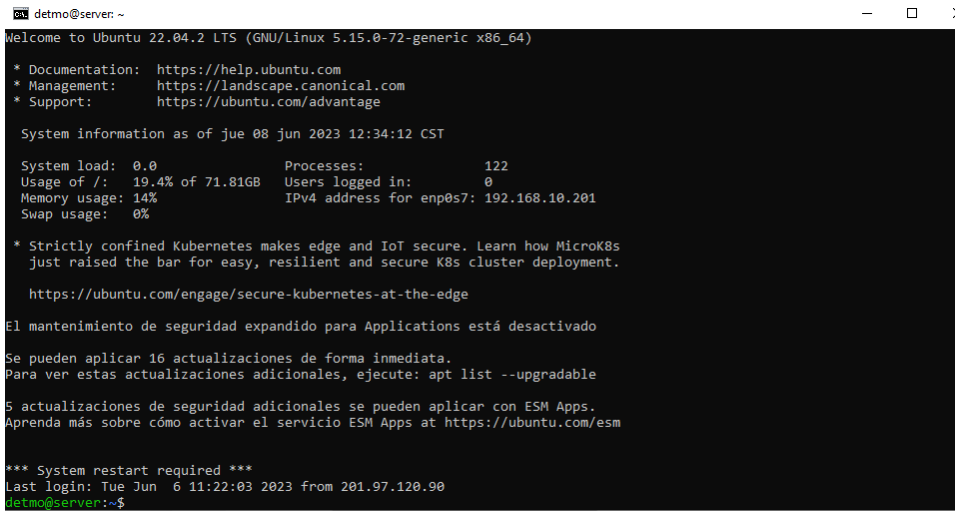

Figura 31. Acceso al servidor (fuente: Elaboración propia, 2023).

1. Para lograr acceso remoto al servidor, en primer lugar, se procede a abrir la interfaz de línea de comandos (cmd), tal y como se ilustra en la figura 32. Posteriormente, una vez que "cmd" ha sido ingresado en el cuadro de diálogo de ejecución, se procede a confirmar la acción presionando la tecla "Enter," lo que resulta en la ejecución exitosa del comando.

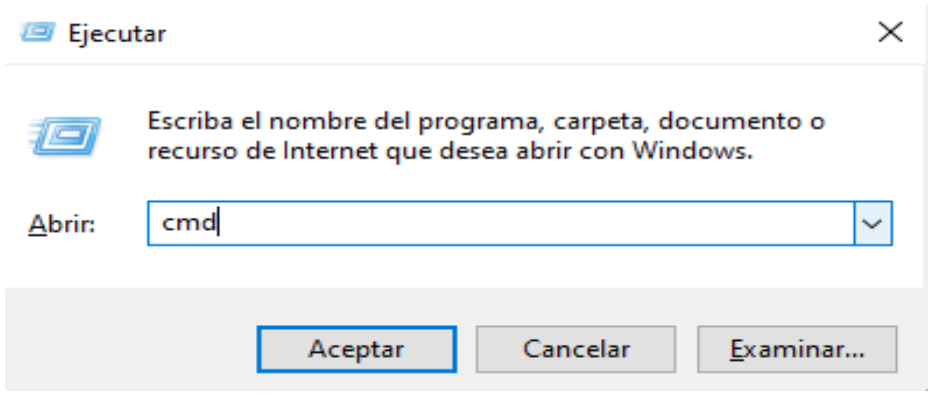

Figura 32. Ejecutar comandos (fuente: Elaboración propia, 2023).

2. Una vez ejecutado el comando, se visualizará el apartado correspondiente en la interfaz de la línea de comandos (CMD), tal como se ilustra en la figura 33. Una vez se haya ingresado en este apartado, se habilitará la posibilidad de acceder al servidor de manera remota.

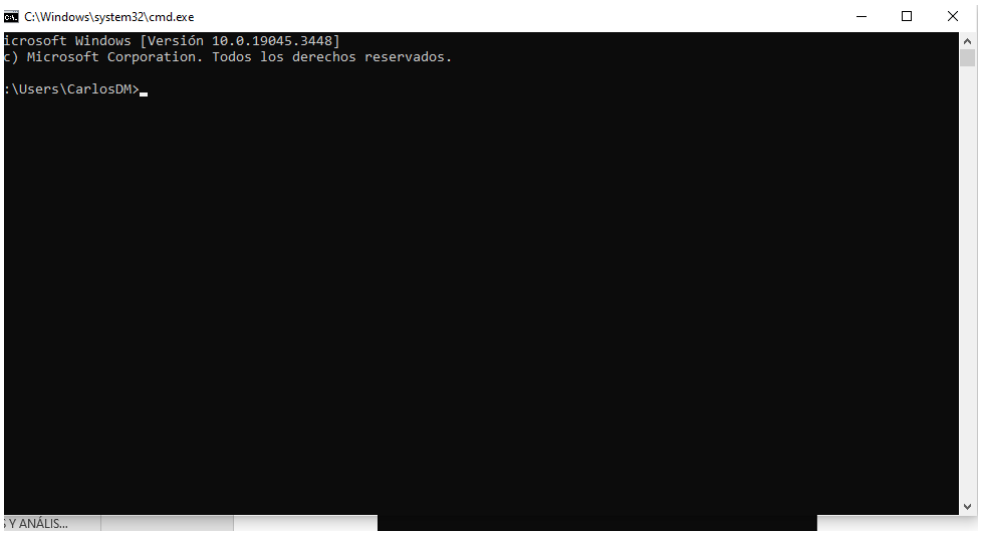

Figura 33. CMD, Comando (fuente: Elaboración propia, 2023).

3. En la Figura 34 suministrada, se procede a verificar la idónea operatividad del servidor, lo cual se manifiesta mediante un acceso exento de adversidades. Con relevancia, es imperativo señalar que se emplea la dirección IP 187.157.156.23 para llevar a cabo la conexión al servidor, como se ilustra en el siguiente ejemplo: `detmo@187.157.156.23.`

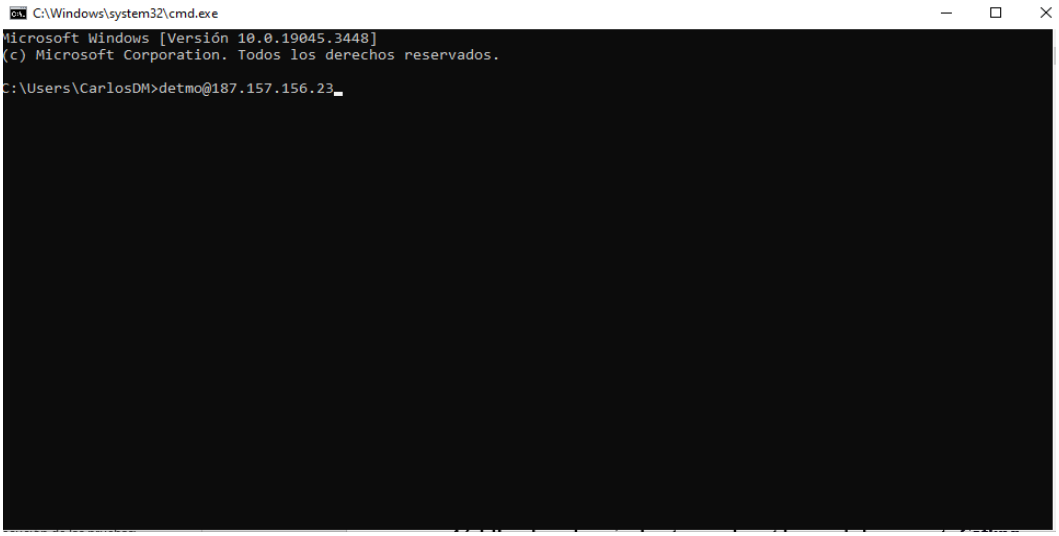

Figura 34. servidor (fuente: Elaboración propia, 2023).

4. Una vez ingresado al servidor, se proceden a llevar a cabo las pruebas de aceptación, entre las cuales se incluyen las pruebas de migración. En caso de que la migración no se realice de manera exitosa, se generará un error notificado al usuario. Por el contrario, si la migración se efectúa de forma adecuada, se presentará el resultado de esta operación tal como se ilustra en la figura 35. Es importante destacar que esta ejecución constituye una fase breve en el proceso global de pruebas de aceptación.

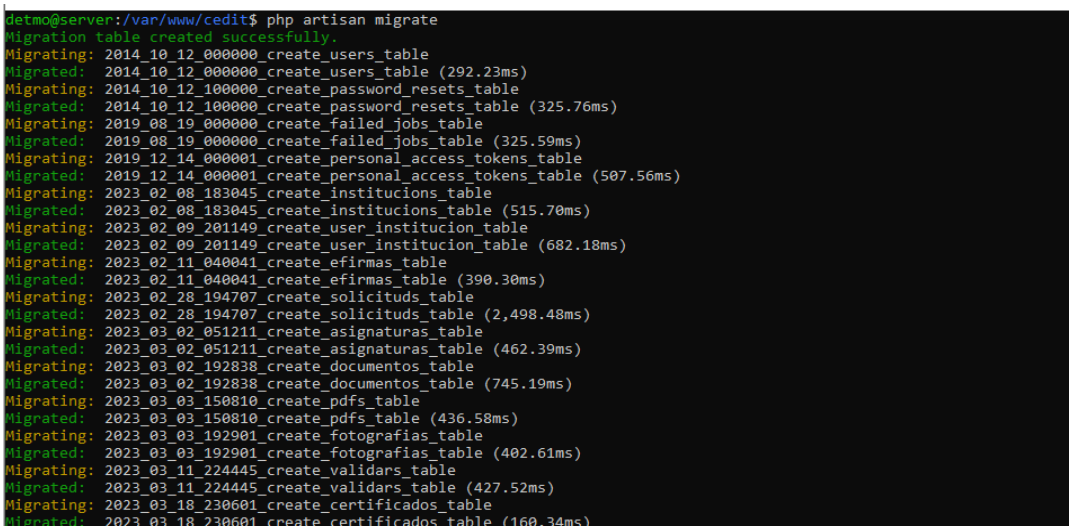

Figura 35. Migraciones (fuente: Elaboración propia, 2023).

4.6 Pruebas de rendimiento

4.6.1 Para la realización de estas pruebas utilizamos la herramienta Gatling

Posteriormente para la ejecución de estas pruebas, se emplea la herramienta Gatling, un software de pruebas de rendimiento altamente especializado en el ámbito académico y profesional. Gatling es una herramienta de código abierto diseñada específicamente para medir y evaluar el rendimiento de sistemas y aplicaciones, lo que lo convierte en una elección popular en el campo de la ingeniería de software y las pruebas de calidad.

El proceso de ejecución de pruebas utilizando Gatling implica una serie de pasos:

1. Para dar inicio al proceso, se accede inicialmente a la plataforma oficial de Gatling, donde se procederá a la autenticación con el propósito de llevar a cabo las pruebas de rendimiento, tal como ilustra la figura 36.

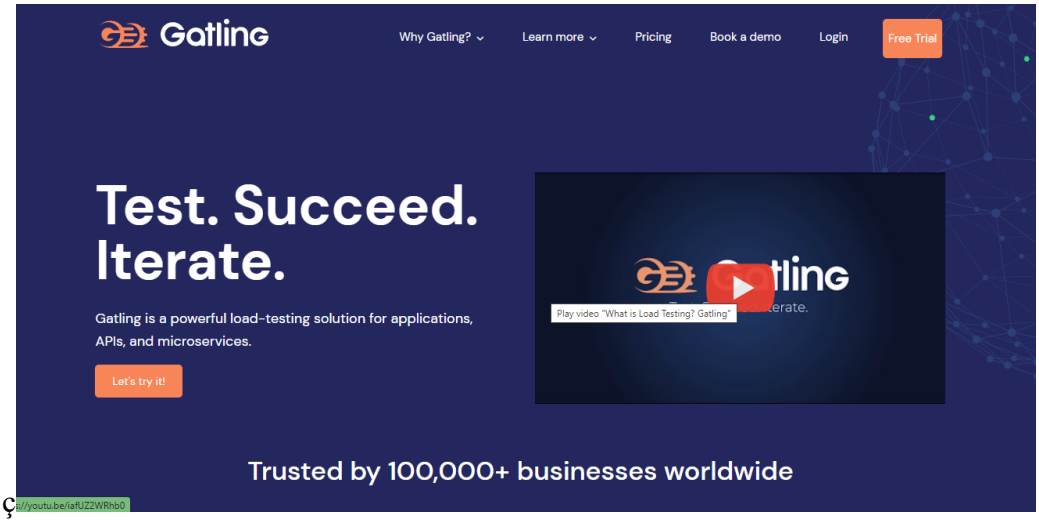

Figura 36. Gatling (fuente: Elaboración propia, 2023).

2. Una vez que el usuario se encuentra en la plataforma oficial, procede a hacer clic en la sección correspondiente al proceso de autenticación, lo que lo redirige a la ventana de inicio de sesión, tal como se ilustra en la figura 37.

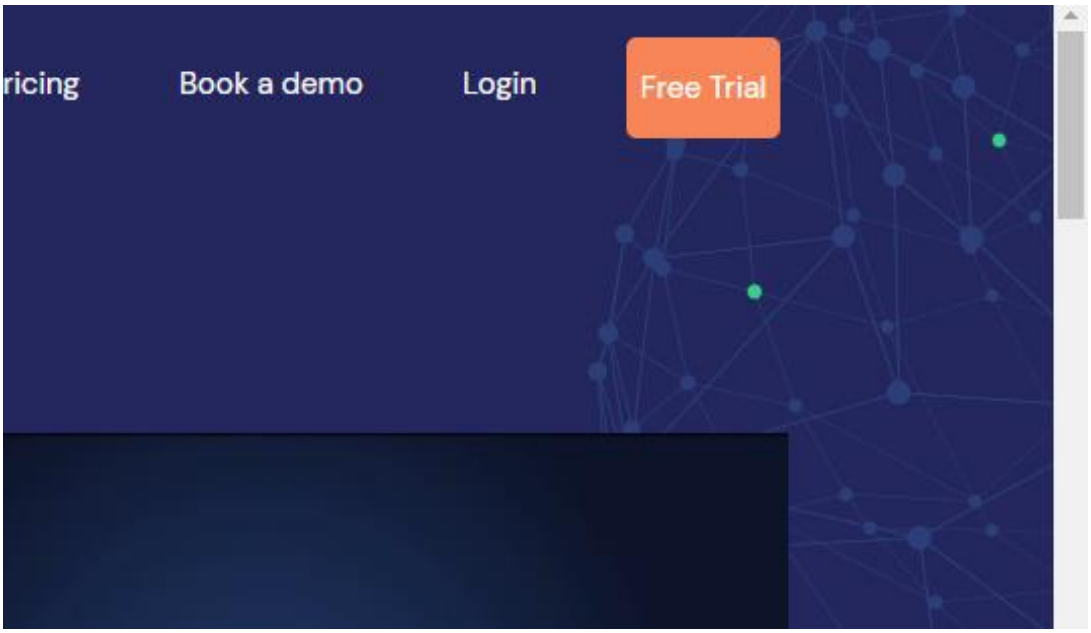

Figura 37. Acceso a login de la página (fuente: Elaboración propia, 2023).

3. Una vez que se hace clic en el enlace, se redirige al usuario a la página de inicio de sesión, donde este puede registrar sus credenciales o iniciar sesión en caso de contar con una cuenta previamente creada, tal como se ilustra en la figura 38.

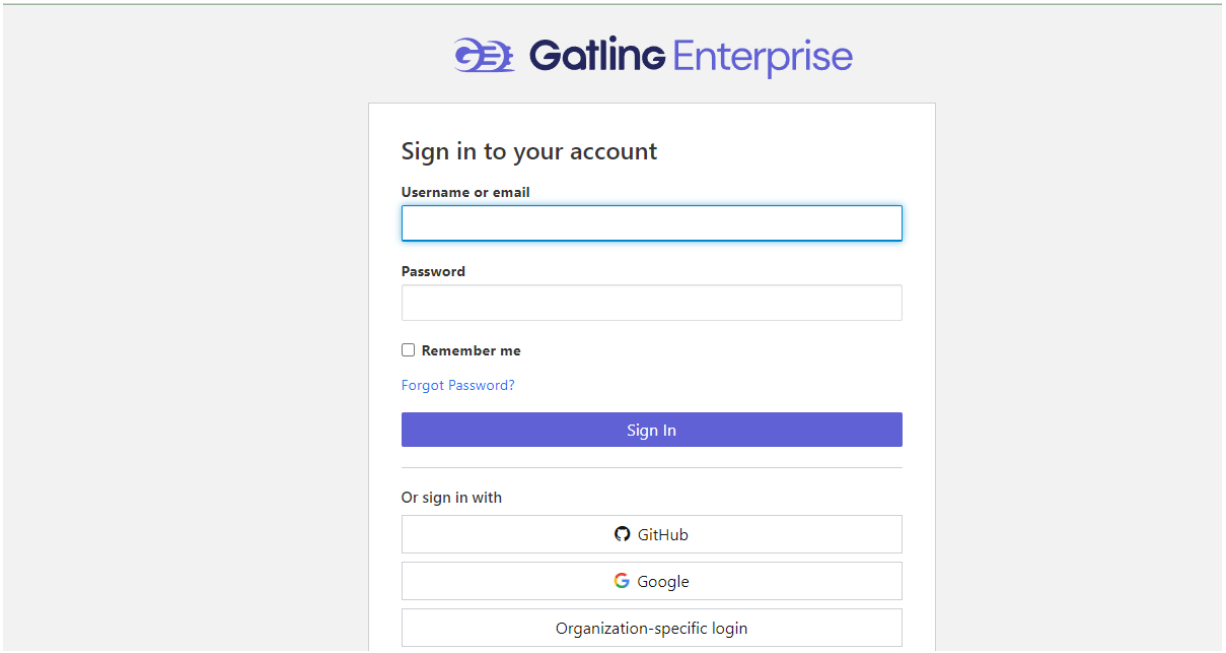

Figura 38. Registro de usuario (fuente: Elaboración propia, 2023).

4. Después que el usuario ha iniciado sesión con su cuenta, se desplegará una ventana destinada a la ejecución de pruebas de rendimiento, como se ilustra en la figura 39. En esta sección, se presentan diversas configuraciones que el usuario puede personalizar y poner en marcha de acuerdo a sus preferencias o características que considere pertinentes.

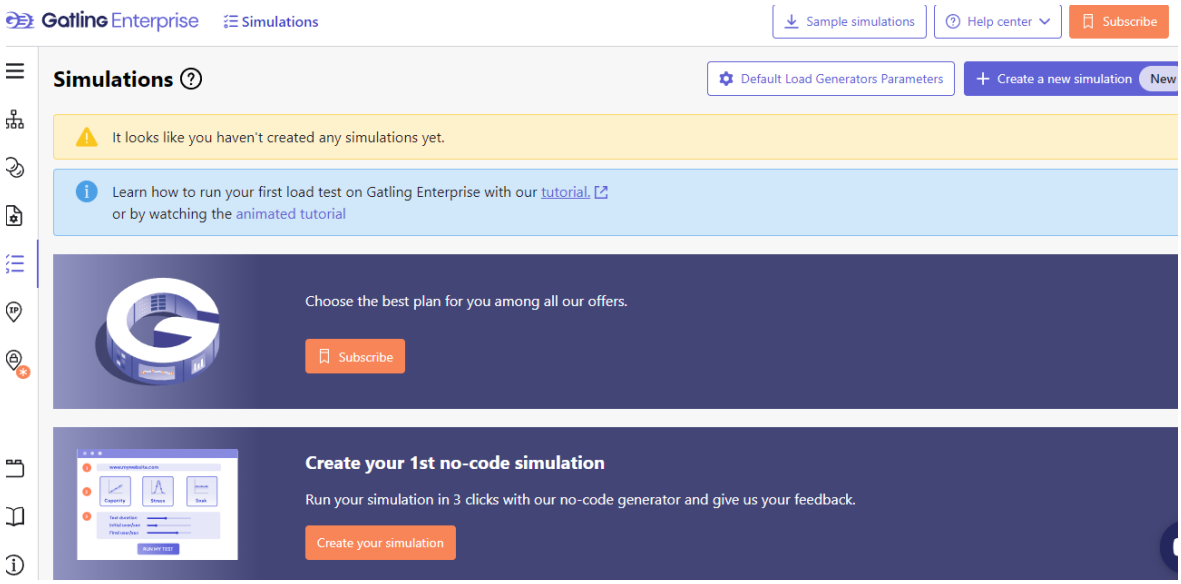

Figura 39. Ventana de ejecuciones (fuente: Elaboración propia, 2023).

4.7 Pruebas de humo

Las pruebas de humo, también conocidas como "*smoke tests*," son pruebas iniciales y superficiales que se realizan para verificar que una nueva versión de software o una nueva característica funcione de manera básica antes de realizar pruebas más exhaustivas. Estas pruebas tienen como objetivo principal detectar errores graves o problemas que podrían impedir que el software funcione en absoluto. A continuación, se describe cómo ejecutar pruebas de humo:

1. En una primera etapa, se procede a despejar un espacio idóneo para la instalación de la maquinaria, donde se minimicen potenciales riesgos asociados al contacto con agua, polvo u otros agentes adversos. Este espacio debe cumplir con requisitos de integridad estructural, así como asegurar condiciones apropiadas de temperatura y disponibilidad de infraestructura eléctrica, incluyendo interruptores y, en caso necesario, baterías, entre otros componentes relevantes.
2. En el proceso de implementación, se procede a la instalación de los componentes esenciales, tales como la pantalla, el teclado, el ratón y otros dispositivos necesarios para el funcionamiento óptimo de la página web. Posteriormente, se llevan a cabo las pruebas pertinentes con el propósito de verificar la funcionalidad y el desempeño del sitio web en cuestión.

Figura 40. Accesorios de máquina de escritorio Pc (fuente: Marizol, 6 de septiembre del 2021).

3. Se lleva a cabo una verificación del correcto funcionamiento de la unidad central de procesamiento (CPU) al intentar iniciar el sistema, dada la presencia de atributos y características inherentes al servidor que respaldan su operatividad. Este proceso es ilustrado en la figura 41, la cual exhibe el mencionado CPU.

Figura 41.Unidad central de procesamiento (fuente: Ángela, 2007)

Una vez instalado el equipo, se procede a la ejecución de la página web que integra de manera holística el sistema, con el propósito de verificar el funcionamiento apropiado de todos sus componentes, incluyendo, el ventilador y el disco duro entre otros.

CAPÍTULO 5: RESULTADOS Y ANÁLISIS DE RESULTADOS

En el presente capítulo, se expondrán los resultados obtenidos a partir de la ejecución de las pruebas de software, acompañados de una descripción detallada de las acciones llevadas a cabo.

Se llevó a cabo la instalación de una estación de trabajo con el propósito de realizar las pruebas del servidor, habilitando así su operación en un entorno local con el fin de facilitar la ejecución de los procedimientos pertinentes. Es importante destacar que el equipo empleado no requiere una asignación sustancial de recursos, y las características específicas del equipo empleadas para dichas pruebas son las siguientes:

- Características: Frecuencia del procesador: 1,8 GHz
- Memoria RAM: 4 GB
- Tipo de memoria interna: DDR2-SDRAM
- Capacidad total de almacenaje: 160 GB
- Sistema operativo instalado: Ubuntu server
- Modelo del procesador: AMD Athlon II 170u
- Unidad de DVD-Super Multi

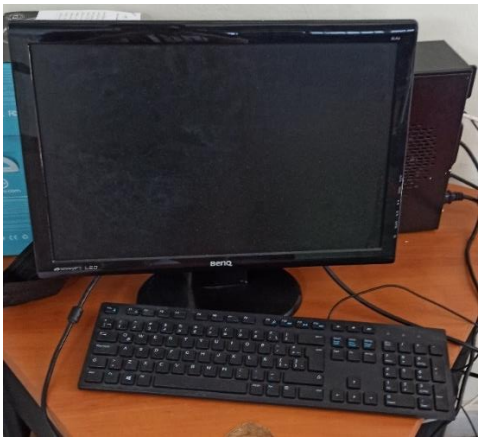

Figura 42. Computadora y sus complementos (fuente: Elaboración propia, 2023)

Después de haber instalado el servidor y los softwares correspondientes, se procedió a la realización de las siguientes pruebas.

- Pruebas Unitarias
- Pruebas Integración
- Pruebas Funcionales
- Pruebas extremo a extremo
- Pruebas de aceptación
- Pruebas de rendimiento
- Pruebas humo

5.1 Unitarias

Se procedió a realizar una indagación con el propósito de identificar las clases pertinentes que albergaban las pruebas destinadas a ser ejecutadas. Estas clases, por su parte, pueden ser localizadas en la columna situada en el margen izquierdo de la interfaz. Al seleccionar una de estas clases, se accederá a una pestaña específica que proporciona la información detallada sobre el estado del proceso de ejecución en curso, tal como se ilustra en la figura 43.

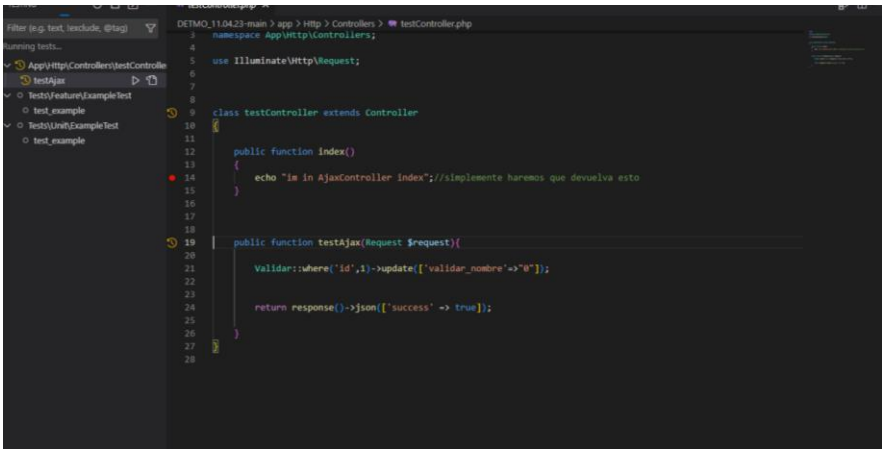

Figura 43. Ejemplo 15 (Fuente: Elaboración propia, 2023)

Tras la ejecución de las acciones previamente delineadas, se adquirirá una manifestación visual en la forma de una señal coloreada de tonalidad verde, conforme se ilustra en la figura 44. Esta manifestación visual proporciona una notificación de que el procedimiento ha sido llevado a cabo de manera exitosa. Adicionalmente, en el margen izquierdo, se despliega un seguimiento en tiempo real del progreso del proceso, tanto en la representación visual como en el código correspondiente.

Figura 44. Ejemplo 16 (fuente:Elaboración propia, 2023)

En la figura 45 se ilustra una manifestación del funcionamiento único, representada mediante un ícono de validación de tonalidad verde situado en la línea 14, abarcando simultáneamente las líneas 15, 16 y 17.

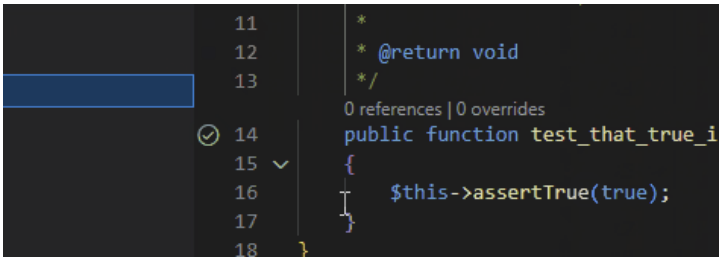

Figura 45. Ejemplo 17(fuente: Elaboración propia, 2023)

En la figura 46 se muestra un error de este estilo que se origina a raíz de una implementación defectuosa del código, manifestándose en la parte externa de la interfaz. Este error se manifiesta mediante una lista que compila la totalidad de los errores identificados, siendo distinguidos por una marca representativa de "X".

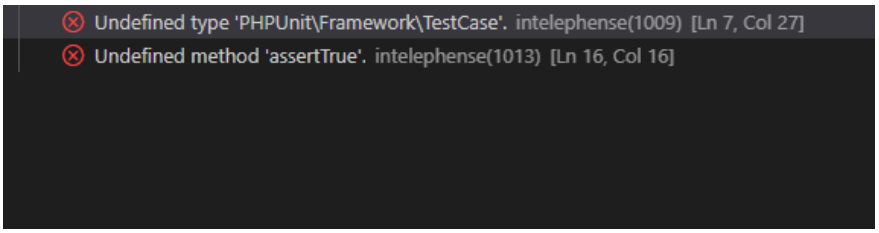

Figura 46. Ejemplo 18 (Fuente: Elaboración propia, 2023)

En el transcurso de la ejecución de las pruebas unitarias, se procedió a documentar tanto los errores identificados como aquellos que fueron exitosamente resueltos. Paralelamente, se consignaron los casos de éxito que ayudaron al proceso de validación, tal y como se ilustra en la figura 47.

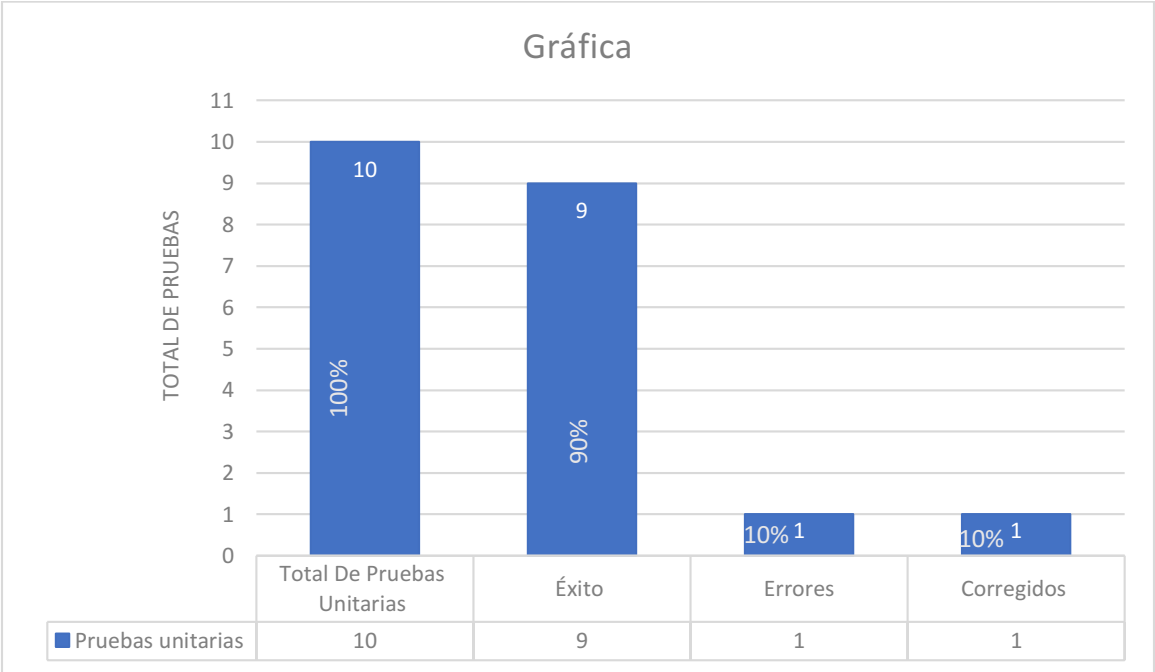

Figura 47. Gráfica de pruebas unitarias (fuente:Elaboracion propia)

En la figura 47, Se ilustra los resultados de las pruebas unitarias, las cuales se llevaron a cabo bajo un límite máximo de diez pruebas. De estas, nueve de ellas demostraron un desempeño exitoso, mientras que una de ellas presentó errores. A continuación, se presenta una redacción de manera detallada en los siguientes puntos

- En las pruebas unitarias, nueve instancias lograron ser exentadas con éxito debido a la eficiente operatividad del sistema, el cual fue evaluado a través de un minucioso análisis de su código fuente, llevado a cabo línea por línea. Esta modalidad de evaluación, conforme a los parámetros definidos por la prueba, implica una evaluación para una inspección de cada componente de código, con el propósito de detectar y corregir la mayor cantidad posible de errores en el sistema. Dicho procedimiento se ilustra de manera ejemplar en la figura 46.
- El error que se suscitó fue corregido con éxito, gracias a una correcta evaluación del código, que permitió identificar la raíz del mismo. Este descuido provino de una inadecuada implementación en el componente lógico del proyecto. Como resultado, se procedió a la corrección pertinente. En la figura 47 del presente informe gráfico se ilustra el 10% remanente de las pruebas unitarias en relación al total del 100%.

5.2 Pruebas de integración

En la figura 48 se visualiza la adecuada operatividad de la página, lo que refleja un acceso exento de inconvenientes. Es relevante subrayar que se emplea la dirección IP 187.157.156.23 con el propósito de acceder a la plataforma, en virtud de que aún no se ha otorgado un dominio particular.

Figura 48. Login de usuarios (fuente: Elaboración propia, 2023)

En la Figura 49, se procede a ilustrar el acceso a la plataforma mediante la utilización del correo electrónico previamente asignado al usuario. Con ocasión del proceso de registro, se remitieron las credenciales pertinentes al usuario con el propósito de habilitar su utilización. En situaciones en las cuales el correo electrónico no se encuentre registrado en la base de datos de usuarios, se limita el acceso a la página, procediendo así a una recarga de la misma, emulando la ausencia de ingreso de información alguna.

Figura 49. Acceso a la página (fuente: Elaboración propia, 2023)

En la figura 50 se muestra que el usuario ha logrado acceder exitosamente y se encuentra habilitado para llevar a cabo la verificación de las solicitudes pendientes de tramitación. Una vez inmerso en este entorno, dicho usuario estará en condiciones de visualizar la información siguiente:

- **El número de solicitudes:** En la presente sección, se exhibe la cuantificación de las solicitudes pendientes de firma, lo cual constituye un indicador relevante para evaluar la eficiencia del proceso de gestión documental.
- **Tipo de certificado:** En esta sección se explicita el título del certificado sobre el cual se llevará a cabo la suscripción.
- **Número de control:** La presente información se encuentra asociada al estudiante, dado que, al momento de su matriculación en la institución educativa, se le otorga un número de registro que opera como un identificador de su condición de discente.
- **Plan de estudios:** Se procede a verificar si el estudiante presenta alguna asignatura pendiente en la institución educativa.
- **Acciones:** Esta sección habilita la ejecución de la firma del certificado.
- **Seleccionar:** La mencionada funcionalidad posibilita la selección de los estudiantes que demuestran estar preparados para la recepción de rúbricas electrónicas.

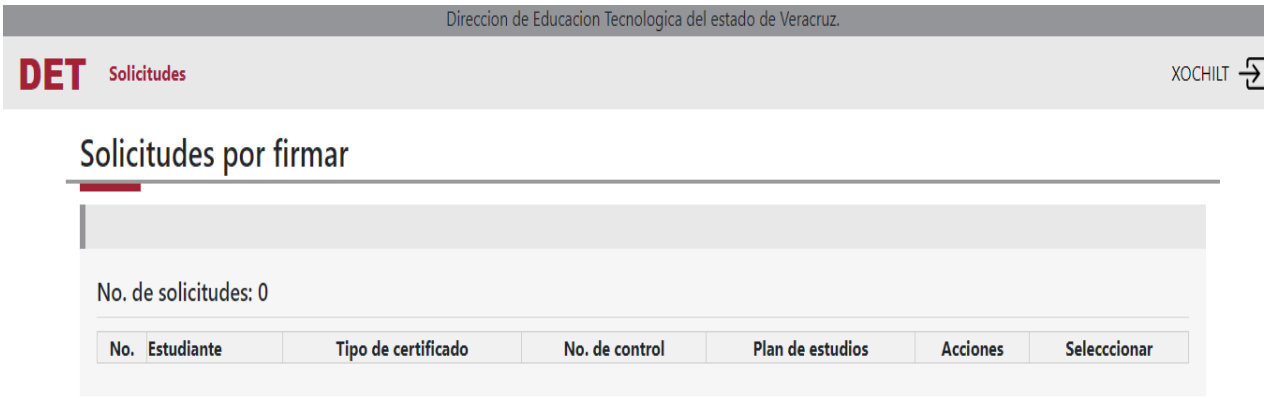

Figura 50. Solicitudes de la DET, Director de educación tecnológica (Fuente: Elaboración propia, 2023)

En la figura 51, se ilustra un registro de usuarios inscritos en el repositorio de datos. Los usuarios mencionados constituyen al exclusivo grupo que ostenta el privilegio de acceder a la página web en cuestión de uso de una base de datos gestionada por el administrador, el cual posee el privilegio de acceder a dicha información.

Figura 51. Base de datos con identificación (fuente: Elaboración propia, 2023)

El líder, desempeñando su función asignada, asume la responsabilidad de supervisar de manera integral la totalidad de los recursos tecnológicos inscritos en el sistema, lo cual queda debidamente consignado en el repositorio de datos correspondiente. Además, cuenta con una herramienta de visualización que suministra una profunda gama de datos mínimos acerca de cada director responsable de dichos recursos. En el margen derecho de la interfaz, se localiza un motor de búsqueda que facultará a los usuarios a efectuar la búsqueda más eficiente mediante la

introducción de términos clave o nombres específicos relacionados con los recursos tecnológicos en cuestión.

Además, el líder ilustra una funcionalidad de registro de instituciones, en la cual, una vez culminado el proceso de inscripción, los datos correspondientes son reflejados en el listado primordial. Esto posibilita una administración centralizada de todas las instituciones registradas, tal como se ilustra en la figura 52.

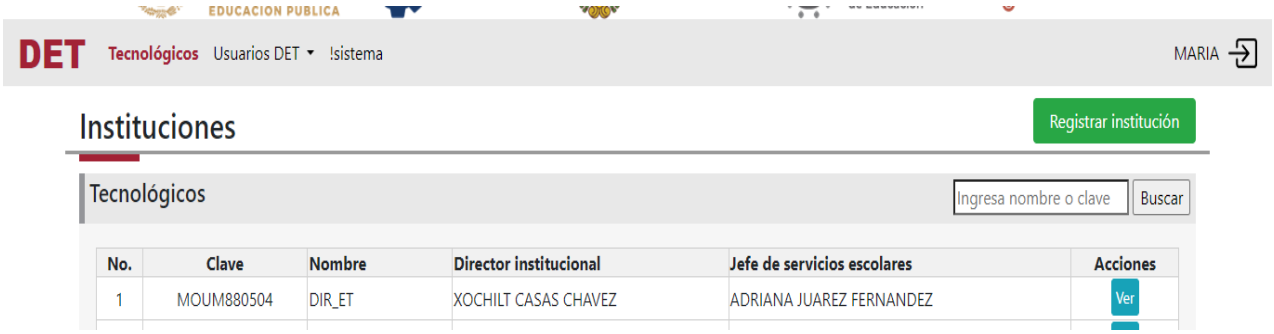

Figura 52. Instituciones registradas (fuente: Elaboración propia, 2023)

Para verificar el adecuado funcionamiento de la vista representada en la figura 53, es imperativo que el usuario proceda a seleccionar la opción "editar" ubicada en el lado derecho de la interfaz. Al ingresar a este apartado en particular, se le otorgará al usuario la capacidad de efectuar ajustes en la información relacionada con cada entidad registrada en el sitio web. Dicha información comprende aspectos tales como el nombre de la institución, dirección de correo electrónico, contraseña, entre otros datos de relevancia.

Figura 53. Editar información de tecnológicos (fuente: Elaboración propia, 2023)

La figura que se presenta a continuación muestra las estadísticas pertinentes a las correcciones llevadas a cabo en función de las pruebas de integración realizadas, tal como se ilustra en la figura 54.

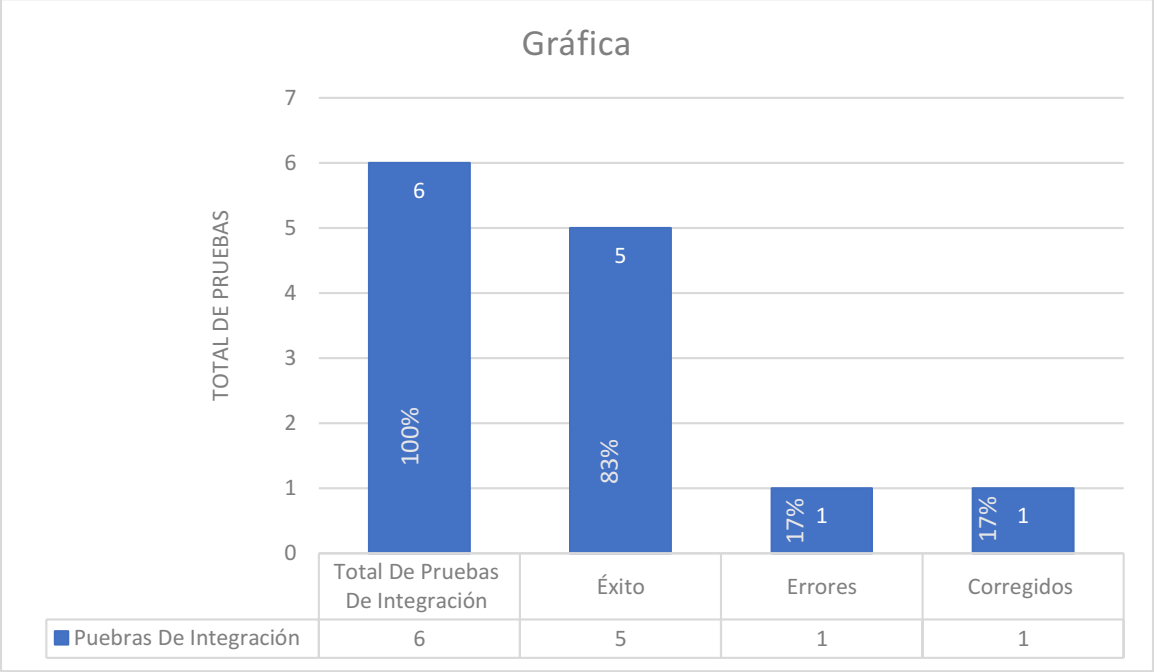

Figura 54. Gráfica de pruebas de integración (fuente:Elaboracion propia, 2023)

En la gráfica previamente mencionada, correspondiente a la figura 54, se observa una representación estadística de pruebas llevadas a cabo, con un límite máximo de 6 pruebas. De este conjunto, 5 arrojaron resultados exitosos, mientras que 1 de ellos evidenció errores. En los presentes puntos, se procederá a un análisis integro de la situación anterior descrita, teniendo en consideración la corrección de dicho error.

- En las cinco primeras pruebas, se confirmó el logro del éxito, dado que el usuario en cuestión logró acceder de manera recurrente utilizando la dirección de correo electrónico que le fue proporcionada. Este éxito se fundamenta en la correspondencia entre la información proporcionada y los datos almacenados en la

base de datos. Como consecuencia, se posibilitó el acceso a la página web. Cabe destacar que dichas pruebas se aplicaron en el contexto de la integración de la visualización de usuarios en la base de datos y su incorporación en la plataforma web. Este conjunto de eventos demuestra de manera concluyente el éxito de las integraciones en cada componente de la plataforma web. Todo lo anterior se ve respaldado por un promedio de éxito del 83 por ciento, subrayando así el nivel de éxito alcanzado.

- Cabe destacar que en la gráfica se evidencia un error. Este error, en particular, es atribuible a una incorrecta manipulación de datos. Si el correo electrónico no se encuentra registrado en la base de datos que previamente ha sido configurada por el administrador, no se permitirá el acceso al sistema. En consecuencia, se abordó esta cuestión al habilitar el acceso para el usuario que deseaba ingresar, tras haber completado la integración de su dirección de correo electrónico en la mencionada base de datos. Como resultado de este proceso, se logró reducir la tasa de errores al 1%, solucionando de esta manera la totalidad de los inconvenientes y asegurando un funcionamiento óptimo del sistema en un 100%.

5.3 Pruebas funcionales

En el marco de la presente evaluación, se persigue la validación del acceso al sistema, mediante la verificación del grado de conformidad con los preceptos preestablecidos. En situaciones en las que no se observe la plena satisfacción de dichos criterios, se procederá a denegar el acceso al sistema. Cabe resaltar que a cada usuario se le ha asignado una dirección de correo electrónico de carácter exclusivo, y con el propósito de ingresar exitosamente al sistema, será

imperativo consignar el correo electrónico registrado, en conjunto con la correspondiente información de índole personal, tal y como se ilustra en la figura 55.

Figura 55. Acceso a Directores Institucionales (Fuente: Elaboración propia, 2023)

En la Figura 56, se ilustra que el acceso ha sido exitosamente ejecutado, lo cual se evidencia claramente. Asimismo, en la Figura 55, se aprecia el ingreso a la plataforma web. Cabe destacar que cada solicitud debidamente registrada se mostrará de forma automática en un índice numérico, así como los siguientes datos:

- Número de lista
- Tipo de certificado
- Nombre del estudiante
- Número de control
- Plan de estudios
- Acciones
- Seleccionar

Figura 56. Solicitudes pendientes por confirmar (fuente: Elaboración propia, 2023)

Los individuos que se encuentren limitados en su capacidad de acceder a la plataforma exclusivamente a través de la utilización del correo electrónico previamente registrado en su perfil, deben ser conscientes de que, en caso de que los datos proporcionados sean incorrectos, la consecuencia inmediata será la negación de su acceso a la plataforma. Simultáneamente, se llevará a cabo un proceso de depuración de la página con el propósito de eliminar cualquier información que exhiba de manera eficaz o carezca de validez, tal como se representa en la figura 57.

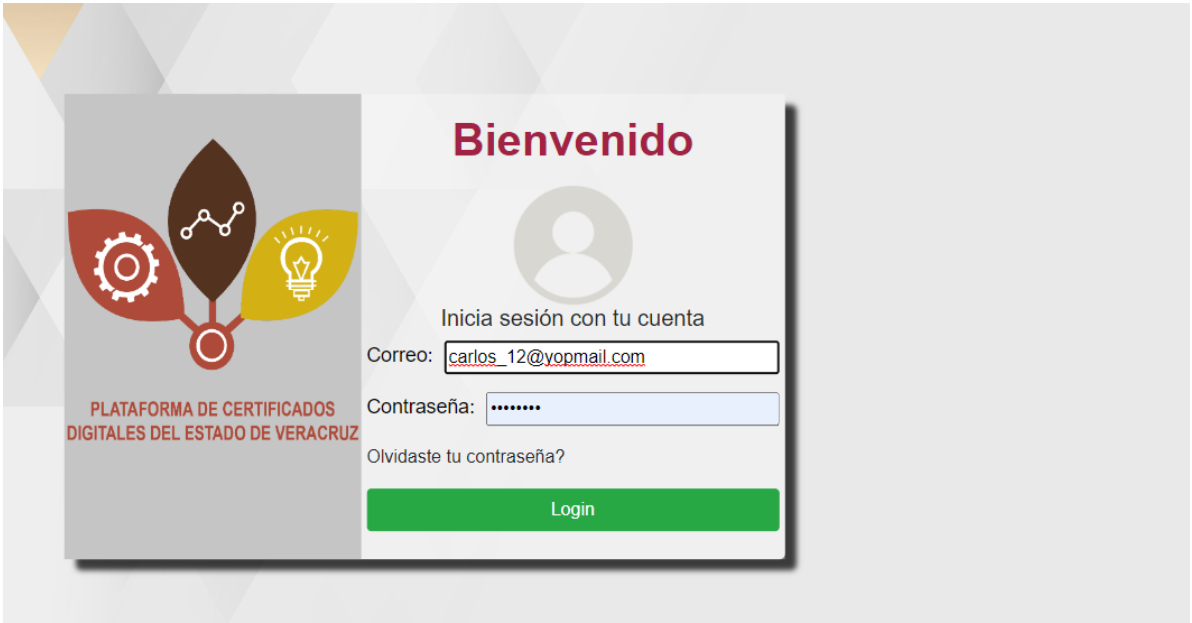

Figura 57. Datos incorrectos (fuente: Elaboración propia, 2023)

Los campos permanecerán en blanco de contenido, dado que la plataforma no habilita la entrada mediante una dirección de correo electrónico que no se encuentre previamente registrada en la base de datos. Este procedimiento se verifica como una estrategia que simplifica el proceso de detección de anomalías o la inclusión de información inexacta por parte de los usuarios, tal como se ilustra en la figura 58.

Figura 58. Corrección de datos incorrectos (Fuente: Elaboración propia, 2023)

En la siguiente grafica se exhibe los errores identificados en el transcurso de las pruebas funcionales, junto con la cantidad de fallos que fueron rectificados, tal como se evidencia en la figura 59.

ç

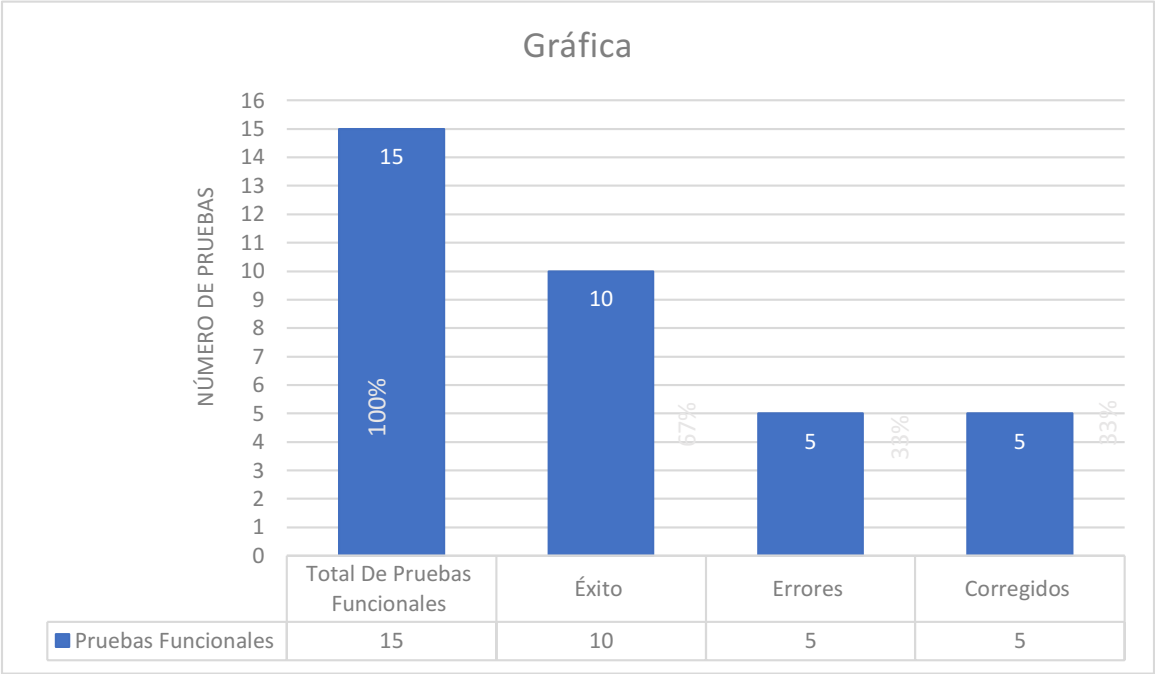

Figura 59. Gráfica de pruebas Funcionales (fuente:Elaboracion propia, 2023)

En referencia a la figura 59, la cual se encuentra dentro del contexto gráfico en consideración, se observa un porcentaje de errores que asciende al 33%. En contraposición, se constata un índice de éxito del 67%, el cual ha permitido la corrección de los 5 errores que se manifestaron inicialmente y se procederá a un análisis de las causas implícitas que dieron origen a tales errores, así como en las correcciones implementadas para su rectificación.

- El éxito de las pruebas totales se cifró en 10 de 15, dado que, de este conjunto de 10, se procedió a la ejecución de diversas verificaciones con el propósito de validar el adecuado funcionamiento de los procesos que abarcan desde la transferencia automática de datos de una ventana a otra, hasta la inserción de elementos como nombres y ediciones, entre otros.
- Los cinco errores detectados en la gráfica se atribuyen a una incorrecta inserción de datos, lo cual se debe, en parte, a la omisión de las pautas establecidas para la

utilización de caracteres alfanuméricos en determinadas ventanas de la plataforma web. La inserción de letras y números se encuentra restringida en áreas específicas, salvo en el apartado destinado al número de control, donde dicha práctica es admisible de acuerdo con las normativas establecidas.

5.4 Pruebas de extremo a extremo

En la figura 60 proporcionada se exhibe íntegramente la totalidad de la información ingresada por el administrador, siendo de importancia que cada usuario proceda a verificar con cuidado la exactitud de los datos registrados. Estos datos comprenden un conjunto de elementos que engloban, entre otros, los siguientes datos:

- Clave
- Ciudad
- Fecha de registro
- Teléfono

Figura 60. Jefe de Servicios Escolares (fuente: Elaboración propia, 2023)

Una vez que se han registrado de manera apropiada todos los datos pertinentes, se hará patente mediante la representación gráfica de un ícono en forma de símbolo de verificación en color verde que el procedimiento de inscripción ha culminado exitosamente, tal y como se puede constatar en la figura 61. En una diminuta ventana emergente, se desplegará un mensaje que

confirma con certeza el logro del proceso de inscripción, y al presionar el botón denominado "Aceptar", el usuario podrá proseguir con las acciones siguientes.

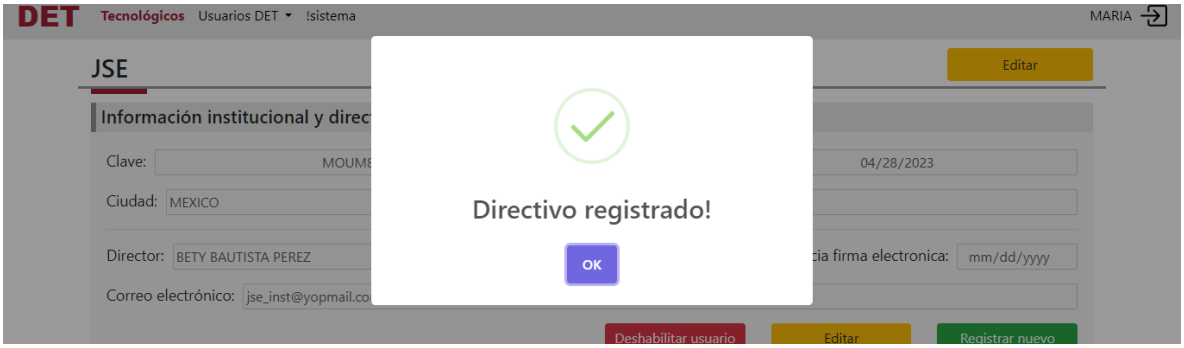

Figura 61. Éxito de registro del Jefe de Servicios Escolares (Fuente: Elaboración propia, 2023)

En la figura 62 que se presenta, se observa la actualización de la contraseña, otorgando al usuario la capacidad de adquirir acceso a la página web. Una vez que las acciones previamente mencionadas hayan sido ejecutadas, la prueba de extremo a extremo se pondrá en marcha y desempeñará su propósito, conllevando así al envío automático de las credenciales al correo electrónico pertinente.

Figura 62. Envió de credenciales (fuente: Elaboración propia, 2023)

Con el propósito de validar la adecuada ejecución de las acciones previamente efectuadas, se procederá a llevar a cabo una prueba de acceso mediante la utilización de las credenciales que han sido remitidas previamente. Esta evaluación posibilitará la confirmación de que el procedimiento ha sido exitosamente ejecutado, como se ilustra en la figura 63.

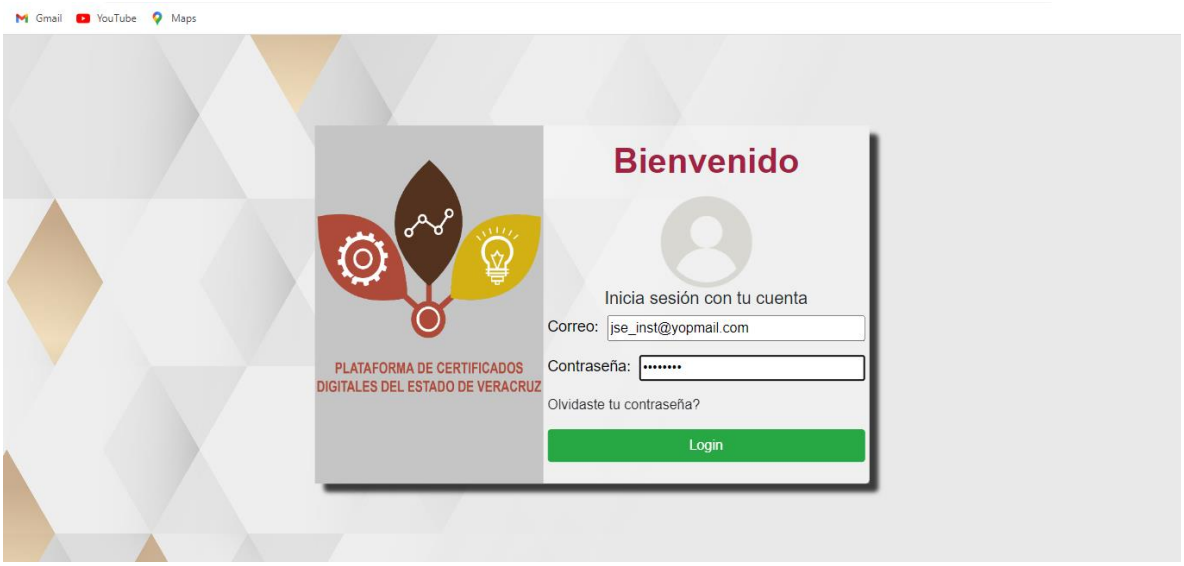

Figura 63. Inserción el correo y contraseña (fuente: Elaboración propia, 2023)

Si las credenciales son correctas, se validará el acceso al sitio web, manifestando, de esta manera, que la configuración se ha llevado a cabo de manera apropiada. No obstante, en el caso de que se tratase del primer acceso del usuario, se requerirá la presentación de un archivo denominado "key" como un elemento inevitable en el transcurso del procedimiento, tal como se ilustra en la Figura 64.

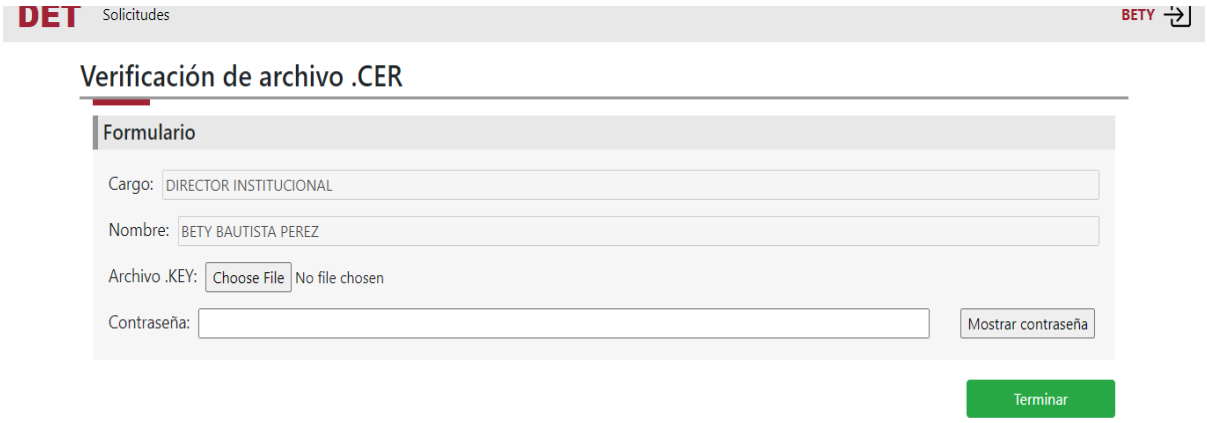

Figura 64. Verificación de archivo .cer (fuente: Elaboración propia, 2023)

La representación gráfica adjunta presenta los resultados obtenidos junto con los errores identificados durante las pruebas de extremo a extremo en la Figura 65.

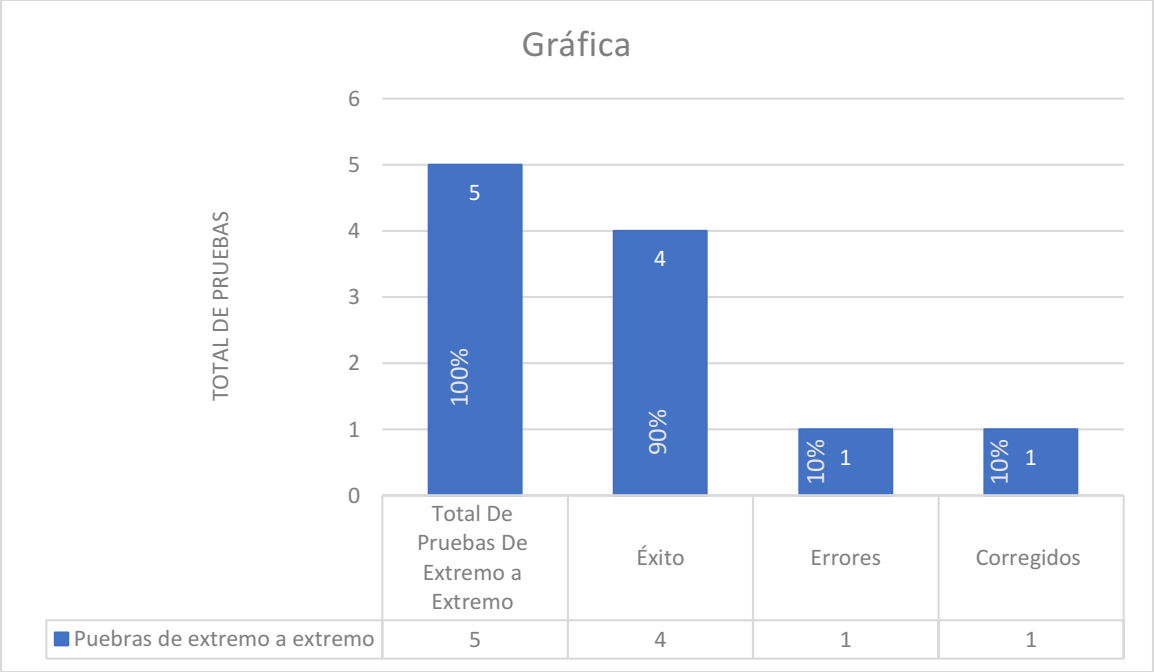

Figura 65. Gráfica de pruebas de extremo a extremo (fuente:Elaboracion propia, 2023)

En la figura 65 se muestra las pruebas de extremo a extremo donde se realizaron cinco ejecuciones, con el objetivo de centrar la atención en su totalidad sobre el proceso de pruebas. De esta manera, se constata que el 90% de las pruebas arrojaron resultados exitosos, mientras que el 10% restante evidenció deficiencias que fueron realizadas con éxito. A continuación, se presenta los siguientes puntos con una breve explicación del número de éxitos y errores.

- Las cuatro pruebas que obtuvieron resultados positivos lo hicieron debido a la inclusión de archivos que concordaban con la sección que requería documentos de tipo "key," lo que les permitió ser evaluadas exitosamente.
- En el único error identificado, se constató que se insertaban archivos que no se correspondían con el formato "key". En el acto de ingresar la contraseña de dicho archivo, esta resultaba inválida debido a la incompatibilidad del tipo documento.

5.5 Pruebas de aceptación

Cada usuario debe contar con su correspondiente dirección de correo electrónico a fin de habilitar su acceso a la plataforma, dado que, en ausencia de este dato, se verá impedido de acceder al sistema. La verificación y autorización de esta interacción se ejecuta a través de la integración con la base de datos, conforme se ilustra en la figura 66.

Figura 66. Acceso a la plataforma institucional (fuente: Elaboración propia, 2023)

Una vez otorgado el acceso al usuario, resultará exitoso al efectuar la comprobación del archivo .cer, conforme se ilustra en la figura 67. Esta ventana realiza dicha información relativa al cargo del usuario, su denominación y la localización específica en la cual se debe cargar el archivo, con el propósito de llevar a cabo la correspondiente verificación.

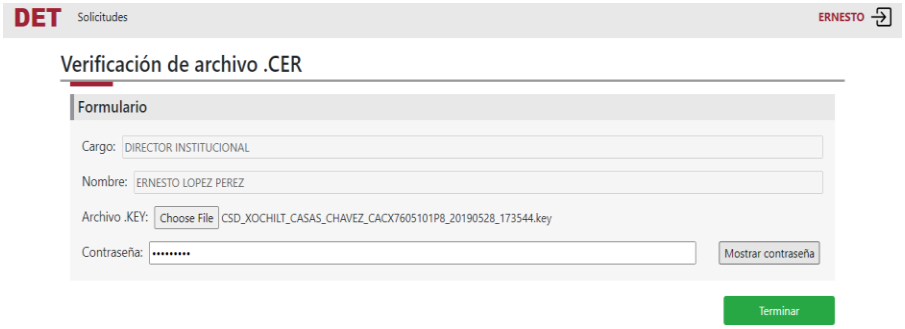

Figura 67. Verificación de archivo .cer (fuente: Elaboración propia, 2023)

Nota: En la eventualidad de que se suscite un error, el sitio web tomará la iniciativa de llevar a cabo un proceso de reinicio, con el propósito de restaurar su estado original, semejante a la circunstancia en la cual no hubieran sido ejecutado previamente las inserciones de ningún tipo.

En la figura 68 que se ilustra ejemplifica situaciones que involucran el acceso a la plataforma web.

Figura 68. Acceso a la plataforma con otro correo (fuente: Elaboración propia, 2023)

En la figura 69, se insertan datos correctos. En la suposición de que uno de estos datos no corresponda a los requisitos establecidos, se generará un error, y, como consecuencia, se le denegará el acceso a la ventana siguiente.

Figura 69. Verificación de archivo .cer (Fuente: Elaboración propia, 2023)

La representación gráfica expone el porcentaje de errores adquiridos, además del número de soluciones logradas, tal y como se observa en la figura 70.

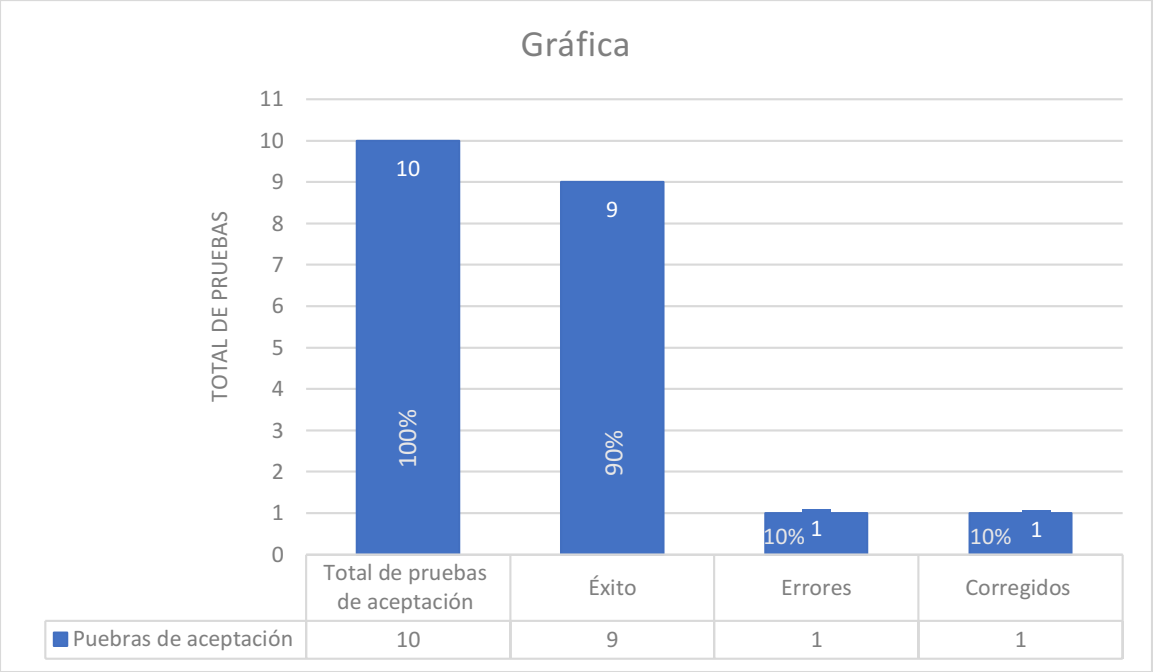

Figura 70. Gráfica de pruebas de aceptación (fuente: Elaboración propia, 2023)

En la gráfica previamente programada en la Figura 70, se observa la representación integral de las pruebas de aceptación llevadas a cabo, cuyo número total asciende a diez. De este modo, de todas las pruebas ejecutadas, nueve culminaron exitosamente, mientras que una prueba arrojó un resultado con error. Cabe destacar que este contratiempo se resolvió con prontitud, con la finalidad de asegurar el logro del 100% de pruebas con resultado satisfactorio. De igual forma, se abordará detalladamente este proceso en los siguientes puntos:

- El éxito en la ejecución de las nueve pruebas se atribuyó al óptimo manejo de la página, que involucra la verificación de archivos con extensión. key. Este procedimiento tiene como finalidad la validación de la disponibilidad para la

inserción de archivos que sean congruentes con las demandas estipuladas por la plataforma web. Se destaca que dicha validación se lleva a cabo en la parte lógica de la plataforma, la cual está específicamente diseñada para analizar y referenciar este tipo de archivos.

- El error radica en la acción de cargar archivos que no posean la extensión .key, seguido de una estadística destinada para evaluar su capacidad en relación con la integridad de la plataforma y la salvaguarda de sus usuarios. Una vez detectada esta disfunción, se procedió a corregir a nivel lógico, a fin de asegurar que tal eventualidad no sea tolerada.

5.6 Pruebas de rendimiento

En la siguiente prueba, se procede a detallar la captura y supervisión del lapso temporal necesario para la ejecución de las imprescindibles operaciones de migración en la base de datos. Cuando el conjunto de datos se encuentra por debajo del origen de 1 gigabyte, se constata una transacción de datos consumada con éxito, tal como se ilustra en la figura 71.

Para acceder al servidor de manera remota, se empleó la Interfaz de Línea de Comandos (CDM, por sus siglas en inglés, Command-line Interface) a través del protocolo SSH, siguiendo el siguiente formato: `detmo@187.157.156.23`. Esta modalidad de conexión remota se establece mediante una contraseña proporcionada por el administrador del sistema. Una vez consolidada la conexión, se procede a acceder a la carpeta denominada "cedit" haciendo uso del comando "cd /var/www/cedit/". En dicha ubicación, se ejecutan las migraciones pertinentes de la figura 71. Cabe destacar la importancia de considerar que, en caso de que surgiera alguna incidencia, resulta imperativo eliminar en su totalidad la base de datos conjuntamente con las migraciones, y proceder a su recreación. La razón implícita es la posible ocurrencia de conflictos si se llegara a generar una

migración adicional con semejantes atributos en el mismo directorio. Para la creación de dichas migraciones, se empleó la herramienta phpMyAdmin.

Para el acceso a phpMyAdmin, se hace imperativo la provisión de un usuario y contraseña específicos, los cuales habilitan al usuario para la visualización de la totalidad de las bases de datos que hayan sido previamente creadas.

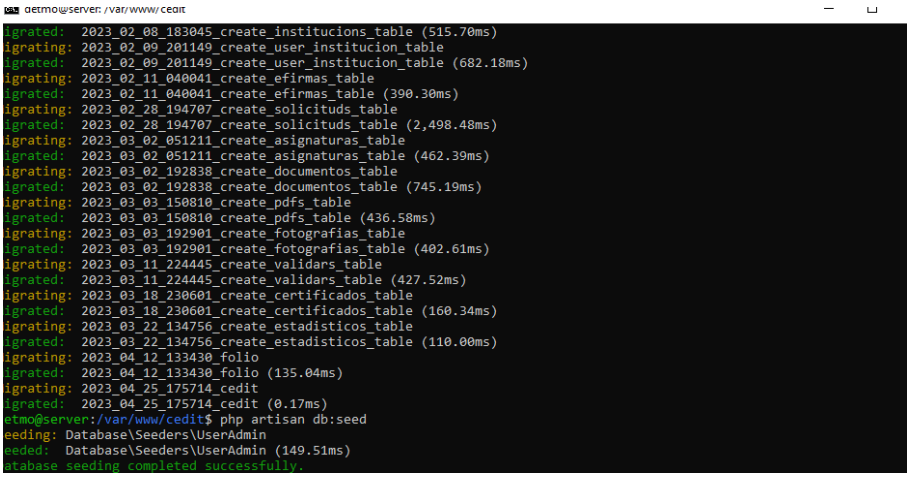

Figura 71. Migración a la base de datos (Fuente: Elaboración propia, 2023).

Se evidencia el lapso requerido para documentar los datos de cada usuario, destacando la consecución exitosa del proceso mediante la inclusión de un símbolo de aprobación, representado por una palomita. Para corroborar la ejecución de esta acción, es ansioso pulsar el botón correspondiente. Esta figura se ilustra desde la 72 a 74.

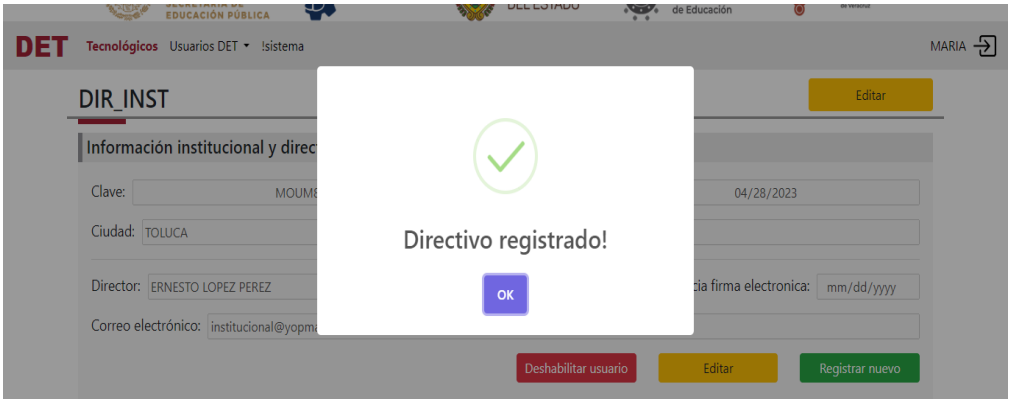

Figura 72. Registro de usuario DIR_INST, Director Institucional (Fuente: Elaboración propia, 2023)

Figura 73. Registro del usuario validador (Fuente: Elaboración propia, 2023)

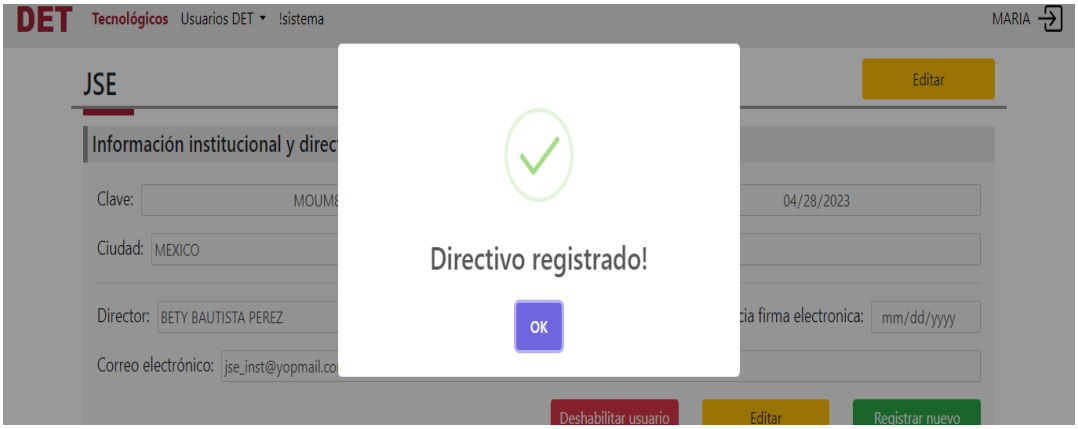

Figura 74. Registro de usuario Jefe de Servicios Escolares (Fuente: Elaboración propia, 2023)

En el gráfico expuesto, se exhiben las respuestas adquiridas a lo largo de las ejecuciones de las migraciones previas de la figura 71. El propósito primordial radica en la evaluación del rendimiento y eficacia inherentes al proceso de inicio de sesión, englobando el tiempo necesario para la culminación de dicha operación, tal como se ilustra en la figura 75.

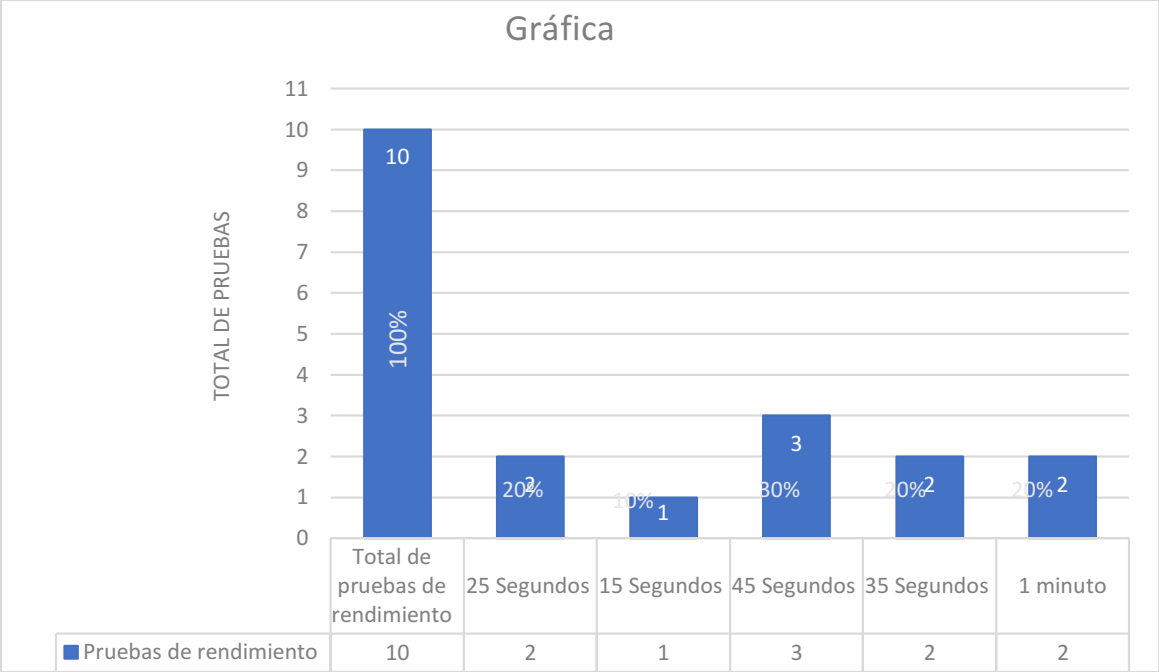

Figura 75. Gráfica de respuesta de las ejecuciones (Fuente: Elaboración propia, 2023)

En la figura 75 de la gráfica de pruebas de rendimiento previamente examinada, se procede a ilustrar el comportamiento del rendimiento en la parte de respuestas en diversas situaciones, tales como el inicio de la página web y la inserción de datos. En este análisis, se llevaron a cabo diez pruebas, las cuales fueron realizadas en distintos escenarios temporales, abarcando un rango de 15 segundos a 1 minuto. Con base en estos resultados, se procede a detallar en los siguientes puntos:

- En un lapso de quince segundos, se logró alcanzar un rendimiento del diez por ciento, lo cual se manifestó en la fase inicial del desarrollo de la página web. Esta eficiencia se atribuye a la carga del sistema operativo, el cual se caracteriza por disponer de las especificaciones adecuadas para el propósito en cuestión. Estos factores se unieron en un único punto de culminación.
- Posteriormente, se llevaron a cabo pruebas con el propósito de determinar el período en el cual la plataforma web podría retener las modificaciones. Dichas pruebas comprendieron un rango de tiempos que realizo entre 25 segundos y 1

minuto, considerando el tiempo de respuesta. Se obtuvieron dos respuestas con un tiempo de 35 segundos, otras dos con un tiempo de 25 segundos, así como tres respuestas con una duración de 45 segundos. Asimismo, se registraron dos respuestas que demandaron un minuto para su procesamiento. De este modo, se completó un conjunto de 10 pruebas que abordaron diversas acciones, tales como la carga de archivos, la modificación de información, el almacenamiento y la eliminación de datos.

Dado que el servidor dispone de diversos componentes, se han ejecutado rigurosas pruebas con el fin de determinar la estimación de la duración que el servidor mantiene su funcionamiento activo en menos tiempo.

Figura 76. Computadora (Fuente: Elaboración propia, 2023)

El sistema ilustra una variedad en su tiempo de respuesta en virtud de la disponibilidad de recursos. En la eventualidad de que los recursos sean escasos, se podría constatar un inicio de sesión más progresado. Para asegurar un acceso eficiente y exento de dificultades, resulta

imprescindible contar con una computadora de elevado rendimiento. La figura 77, que se ilustra, exhibe una representación gráfica del tiempo estimado, expresado en segundos y minutos, para llevar a cabo esta operación.

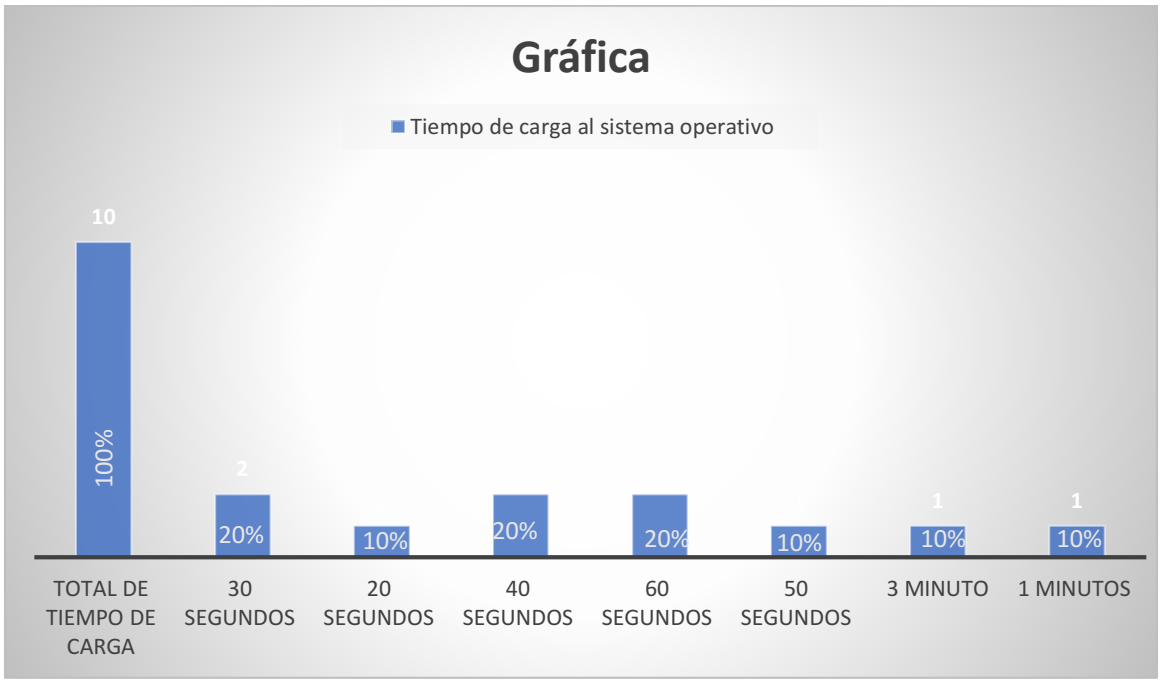

Figura 77. Carga del sistema operativo (Fuente: Elaboración propia, 2023)

La figura 77 presenta una gráfica del tiempo estimado de respuesta al iniciar el sistema, enfocándose exclusivamente en una sola variable. En este apartado, se observa que dicho tiempo se distribuye en diversos intervalos de duración, que van desde los 20 segundos hasta los 3 minutos. Este reparto se estructura de la siguiente manera: el 10% de los casos se concentra en un tiempo de respuesta de 20 segundos, el 20% en 30 segundos, el 10% en 50 segundos, el 20% en 60 segundos, y el restante 10% se extiende desde 1 minuto hasta 3 minutos.

Este análisis permite al usuario percatarse de la relativa limitación de la capacidad de respuesta del sistema en cuestión, ya que en ocasiones se experimenta dificultad al iniciar el mismo. Cabe destacar que la medición de estos tiempos se efectuó mediante un método de carácter más sencillo y convencional, a saber, el uso de un cronómetro físico.

5.7 Pruebas de humo

Es fundamental verificar la apropiada interconexión de todos los dispositivos y prevenir futuros errores durante el funcionamiento del servidor, con la finalidad de prevenir posibles perjuicios al sistema, tal como ilustra la figura 78.

Figura 78. Cableado (Fuente: Elaboración propia, 2023)

Si se ha logrado establecer una conexión adecuada, el acceso al servidor de manera remota podrá ser efectuado. No obstante, en la eventualidad de fallos, la capacidad de acceso se podría ver visto debido a una conectividad deficiente en la infraestructura de la red o a una desigualdad en la configuración de la dirección IP, lo que podría obstruir la entrada. Para efectuar la conexión exitosa, es requisito esencial introducir la dirección IP particularmente especificada seria, 187.157.156.23. En el supuesto de que los datos aportados sean exitosos y concordantes, se manifestará una representación visual semejante a la presentada en la figura 79, a modo de ilustración.

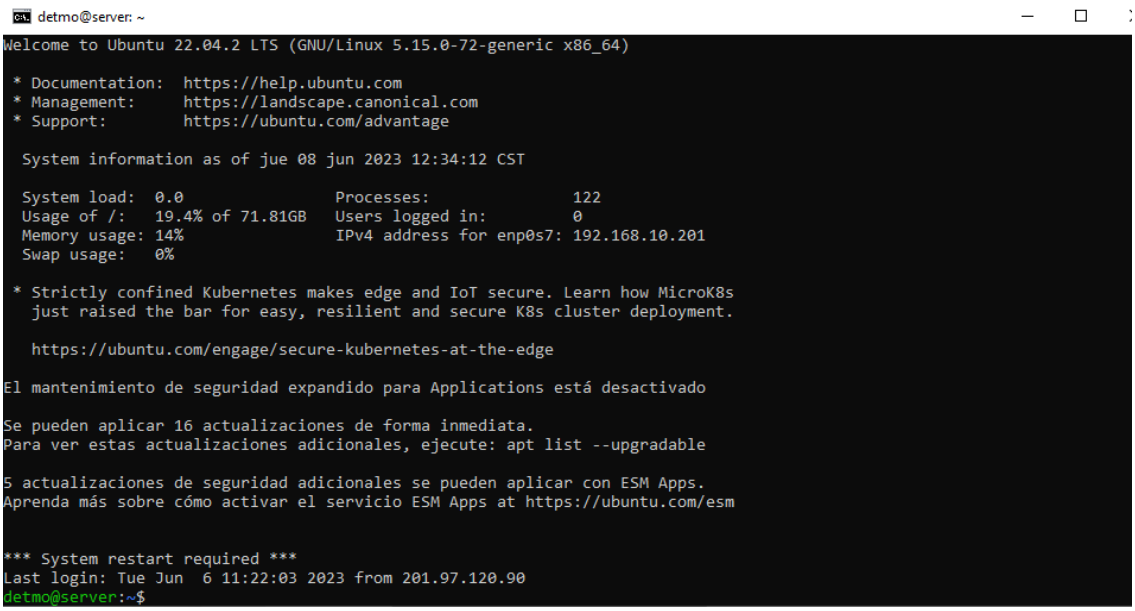

Figura 79. Acceso al servidor vía remoto (Fuente: Elaboración propia, 2023)

En la Figura 80, se visualiza los resultados derivados de las pruebas de humo, las cuales se llevaron a cabo en un número limitado de ocasiones, específicamente 3 pruebas. Cabe destacar que estas evaluaciones fueron ejecutadas bajo diversas condiciones eléctricas, incluyendo la conexión a reguladores de voltaje, la alimentación directa desde la toma de corriente y la realización de pruebas en la conexión de periféricos tales como teclados, ratones y bocinas, entre otros dispositivos.

En las dos ejecuciones que se obtuvieron resultados exitosos, estos fueron atribuibles a las conexiones óptimas del cableado con la fuente de corriente eléctrica directa. No obstante, se identificó un fallo en el 10% de los casos, el cual se vinculó a identificarlo en el voltaje proporcionado por el regulador. Es importante resaltar que dicho error fue resuelto de manera eficaz, alcanzando así un rendimiento exitoso del 100% en las tres pruebas ejecutadas. Este ajuste preciso permitió superar la limitación inicial y validar la integridad de los resultados obtenidos.

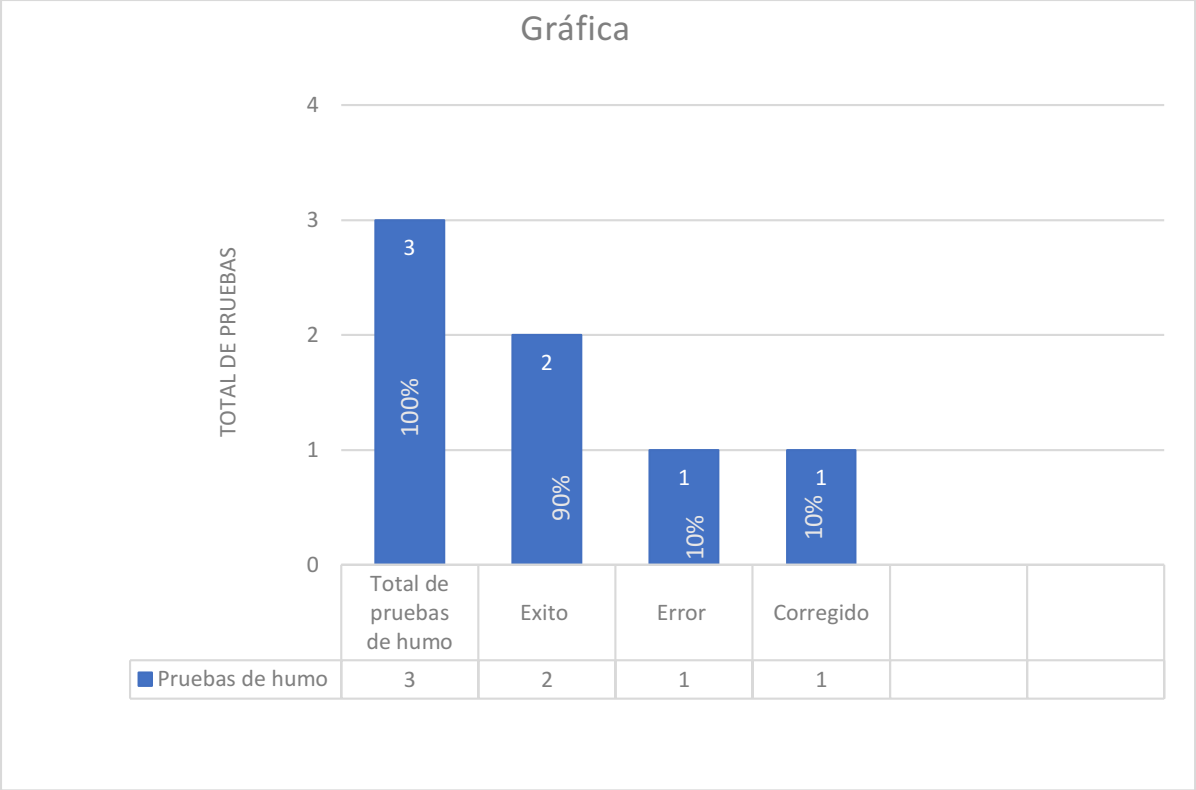

Figura 80. Gráfica de pruebas de humo (Fuente: Elaboración propia, 2023)

CAPÍTULO 6: CONCLUSIONES Y RECOMENDACIONES

6.1. Conclusiones

Después de llevar a cabo diversas pruebas de la plataforma web, se han identificado errores tanto en su funcionamiento como en su usabilidad, interfaz y otros aspectos que afectan su rendimiento óptimo. De esta manera, se ha logrado cumplir el objetivo principal de este trabajo, gracias al uso de herramientas que facilitaron la realización exitosa de estas pruebas enfocado a una plataforma web llamado (dirección de educación tecnológica del estado de Veracruz).

En el proceso de evaluación de las pruebas, se constató que varias de ellas arrojaron resultados exitosos; no obstante, se detectaron múltiples fallos, los cuales fueron atendidos con prontitud a fin de garantizar el óptimo desempeño de la plataforma web.

Cabe destacar que se hizo uso de técnicas y metodologías específicas como pruebas individuales e integral para la ejecución de pruebas de software, así como la utilización de herramientas como visual studio code, página atlassian, gatling, yopmail y bloc de notas, para llevar a cabo todas las evaluaciones requeridas. La realización de este proyecto me aportó de manera profesional conocimientos nuevos en la realización de pruebas de una plataforma web.

Finalmente, se han alcanzado resultados favorables en virtud de la ejecución de un conjunto integro de 59 pruebas, de las cuales 48 resultaron con éxito. Este logro se ha alcanzado a pesar de la presencia de 11 errores, los cuales fueron abordados de manera integral, culminando en un resultado satisfactorio para todas las evaluaciones de software orientadas a una plataforma web.

De manera acertada, se procedió a resolver de forma adecuada la totalidad de las pruebas llevadas a cabo, detallando minuciosamente cada una de las incidencias identificadas. Este proceso culminó exitosamente, abarcando de manera íntegra tanto las evaluaciones integrales como las individuales.

6.2. Recomendaciones

Se recomienda continuar realizando pruebas integras en los módulos subsiguientes que serán incorporados a la plataforma web, manteniendo un registro documentado de dichos procedimientos. En virtud de la pertinencia inherente a la adquisición de destrezas técnicas en el ámbito de la investigación, se formula la recomendación expresa al investigador o individuo interesado en esta esfera de conocimiento de llevar a cabo una práctica constante de los comandos, tanto en entornos basados en el sistema operativo Linux como en aquellos sustentados en Windows. Este imperativo se fundamenta en la premisa de que la maestría en el manejo de instrucciones y órdenes específicas en ambos entornos constituye un componente esencial para el desenvolvimiento eficaz en el ámbito investigativo, proporcionando así una base sólida para el logro de resultados óptimos en las tareas que demanda la investigación en este contexto particular. Este enfoque contribuirá a fortalecer su destreza y competencia en la administración y configuración de sistemas operativos, aspecto esencial en el desarrollo y mantenimiento eficaz de la plataforma web

Bibliografía

[1] Ávila, J. M. (2022, 24 noviembre). ¿Cómo funciona un servidor web en 2023? Blog del E-commerce. https://www.tiendanube.com/blog/mx/como-funciona-un-servidor-web/#:~:text=El%20servidor%20web%20busca%20la,a%20la%20que%20quiere%20acceder.

[2] Compilador e intérprete: definición y diferencias. (s. f.). IONOS Digital Guide. https://www.ionos.mx/digitalguide/paginas-web/desarrollo-web/compilador-e-interprete/

[3] Funcionalidades - IntelliJ IDEA. (s. f.). JetBrains. https://www.jetbrains.com/es-es/idea/features/

[4] Jasmine - Buscar con Google. (s. f.). https://www.google.com/search?kgmid=/m/0g4mp9z&hl=es-MX&q=Jasmine&kgs=e31bab1a5d10868a&shndl=0&source=sh/x/kp/1

[5] Scandaroli, A. (2022, 10 febrero). Pruebas de rendimiento: cuándo y cómo. Encora. https://www.encora.com/es/blog/pruebas-de-rendimiento-cuando-y-como

[6] Turrado, J. (s. f.). Qué son las pruebas de software - campusMVP.es. campusMVP.es. https://www.campusmvp.es/recursos/post/que-son-las-pruebas-de-software.aspx

[7] ¿Qué es un editor de texto y para qué sirve? (2022, 1 marzo). https://www.crehana.com. https://www.crehana.com/blog/transformacion-digital/que-es-un-editor-texto/

[8] Acharya, D. P. (2023). Explicación de las pruebas unitarias: qué es, por qué es importante y cómo empezar. Geekflare. https://geekflare.com/es/unit-testing-guide/

[9] Alegsa, L. (2023). Definición de importar (informática). Alegsa.com.ar. https://www.alegsa.com.ar/Dic/importar.php#gsc.tab=0

[10] Alvarez, A. (2022, 21 abril). Software de aplicación ¿Cuáles son sus tipos y características? Tiendas Virtuales en México Profesionales. https://www.creaxid.com.mx/blog/software-de-aplicacion-cuales-son-sus-tipos-y-caracteristicas/

[11] Ato, M., López, J. J. P., & Benavente, A. (2013). Un sistema de clasificación de los diseños de investigación en psicología. Anales De Psicologia, 29(3). https://doi.org/10.6018/analesps.29.3.178511

[12] Buscar y reemplazar texto en notas - soporte técnico de Microsoft. (s. f.). https://support.microsoft.com/es-es/office/buscar-y-reemplazar-texto-en-notas-34b1f7f8-d327-40c5-8b0c-8419425ed68b

[13] Chavez, J. J. S. (2023). Pentesting o prueba de penetración: qué es y tipos. www.deltaprotect.com. https://www.deltaprotect.com/blog/que-es-pentesting

[14] Davidbritch. (2023, 13 julio). Aplicaciones empresariales de pruebas unitarias - Xamarin. Microsoft Learn. https://learn.microsoft.com/es-es/xamarin/xamarin-forms/enterprise-application-patterns/unit-testing

[15] De Tecnología Educativa, B. (s. f.). Diagrama de transición y tabla de transición (Autómatas). Blog de Tecnologia, Ingenieria en Sistemas. https://www.postecnologia.com/2014/02/diagrama-de-transicion-y-tabla-de.html

[16]Defínción de formato. (s. f.). DefiniciónABC. https://www.definicionabc.com/tecnologia/formato.php

[17] Desarrollo de software: calidad con test de mutación | Mytra Control Blog. (s. f.). https://www.mytra.es/blog-post/test-de-mutacion-la-calidad-es-

prioritaria#:~:text=%C2%BFQu%C3%A9%20es%20una%20prueba%20de,errores)%20mediante%20los%20test%20unitarios.

[18] Devoteam México. (2022, 29 septiembre). Una guía práctica para las pruebas de extremo a extremo con Cypress - Devoteam Mexico. Devoteam Mexico. https://mx.devoteam.com/expert-view/una-guia-practica-para-las-pruebas-de-extremo-a-extremo-con-cypress/#:~:text=%C2%BFQu%C3%A9%20son%20las%20pruebas%20de,llamado%20%E2%80%9Cflujo%20de%20usuario%E2%80%9D.

[19] García, R. S. (2022). 8 Técnicas de prueba para obtener la certificación ISTQB FL. El mínimo viable. https://elminimoviable.es/8-tecnicas-de-prueba-para-obtener-la-certificacion-istqb-fl/

[20] Gómez, D. R. (2006a). Knowledge Creation and Management Models: A Theoretic approach. Educar, 37, 25. https://doi.org/10.5565/rev/educar.187

[21] Gómez, D. R. (2006b). Knowledge Creation and Management Models: A Theoretic approach. Educar, 37, 25. https://doi.org/10.5565/rev/educar.187

[22] Prats, A. (2019, 13 febrero). QA: caso de uso vs caso de prueba | ENCAMINA. Piensa en software, desarrolla en colores. https://blogs.encamina.com/piensa-en-software-desarrolla-en-colores/qa-caso-de-uso-vs-caso-de-prueba/

[23] Ramírez, A. (2023). La importancia de las pruebas de integración en el desarrollo de software. DataDope. https://datadope.io/la-importancia-de-las-pruebas-de-integracion-en-el-desarrollo-de-software/

[24] Ramírez, L. (2022, 24 noviembre). Software Testing: definición, tipos y beneficios. Thinking for Innovation. https://www.iebschool.com/blog/software-testing-que-es-tipos-digital-business/

[25] Stum, L. (s. f.). Registros, estilos y tipos de texto. prezi.com. https://prezi.com/g841e9r-81ki/registros-estilos-y-tipos-de-texto/

[26] Terrera, G. (s. f.). Blog. https://testingbaires.com/pruebas-caja-negra-enfoque-practico/

[27] Yadir, M. (s. f.). METODO BIG BANG. prezi.com. https://prezi.com/p/ihufv11ehwnu/metodo-big-bang/

Anexos

INSTITUTO TECNOLOGICO SUPERIOR DE TANTOYUCA

ANEXO XXXIII. FORMATO DE LIBERACION DEL PROYECTO PARA LA TITULACION INTEGRAL

Ciudad: **TANTOYUCA** Estado: **VERACRUZ** Fecha: **20/OCTUBRE/2023**

Asunto: **Liberacion De Proyecto Para La Titulacion Integral**

C. **FRANCISCO JAVIER MORALES CRUZ**
Jefe De Departamento De Estudios Profesionales Y Titulación

Por este medio le informo que ha sido liberado el siguiente proyecto para la Titulación integral:

Nombre del Alumno:	**ERNESTO CARLOS DOMINGUEZ ALONSO**
Carrera	**INGENIERÍA EN SISTEMAS COMPUTACIONALES**
No. De Control	**183S0412**
Nombre del Proyecto	**PRUEBAS DE SOFTWARE PARA LA DETECCION DE ANOMALIAS Y LA VERIFICACION DEL CORRECTO FUNCIONAMIENTO DE UNA PLATAFORMA WEB**
Producto:	TITULACION INTEGRAL POR: **TESIS**

- **Lo anterior lo hago de su conocimiento para los fines correspondientes a su Examen Profesional, por lo cual debera de entregar al Departamento de Estudios Profesionales, su trabajo profesional en 5 mini CD´s (debidamente rotulados) de su trabajo en archivo PDF y un engargolado con arillo metalico y pastas trasparentes anexando al final la autorizacion de impresion asi como Donar un libro (nuevo) de su especialidad al centro de informacion.**

Agradezco de antemano su valioso apoyo en esta importante actividad para la formacion profesional de nuestros egresados:

ATENTAMENTE

M.I. JESÚS BLADIMIR HERNÁNDEZ HERNÁNDEZ
División de Ingeniería en Sistemas Computacionales

M.C. Manuel Hernández Hernández	M.C. María Xóchitl Altamirano Herrera	Lic. José Roberto Campos Medellín
Nombre y firma del Asesor	Nombre y firma del Revisor	Nombre y firma del Revisor

C.c.p Depto. De Servicios Escolares.- Para carta de no inconveniencia
Depto. De Recursos Financieros.- Para el cobro de arancel correspondiente
Depto. Estudios Profesionales
Expediente

R01/0322 F-EP-33