UNIVERSIDAD TÉCNICA DEL NORTE

Facultad de Ingeniería en Ciencias Aplicadas

Carrera de Ingeniería en Sistemas Computacionales

Desarrollo de un módulo web para fortalecer la gestión de actividades académicas y administrativas como componente del entorno virtual de aprendizaje integrado (EVAI) para la empresa IERec

Trabajo de grado previo a la obtención del título de ingeniero

en Sistemas Computacionales

Autor:

José Breiner Pai Gonzáles

Director:

MSC. Cosme Macarthur Ortega Bustamante

Ibarra – Ecuador

2022

UNIVERSIDAD TÉCNICA DEL NORTE
BIBLIOTECA UNIVERSITARIA

AUTORIZACIÓN DE USO Y PUBLICACIÓN A FAVOR DE LA UNIVERSIDAD TÉCNICA DEL NORTE

1. IDENTIFICACIÓN DE LA OBRA

En cumplimiento del Art. 144 de la Ley de Educación Superior, hago la entrega del presente trabajo a la Universidad Técnica del Norte para que sea publicado en el Repositorio Digital Institucional, para lo cual pongo a disposición la siguiente información:

DATOS DE CONTACTO			
CÉDULA DE IDENTIDAD:	1004198006		
APELLIDOS Y NOMBRES:	PAI GONZÁLES JOSÉ BREINER		
DIRECCIÓN:	IBARRA, HUERTOS FAMILIARES		
EMAIL:	jbpaig@utn.edu.ec		
TELÉFONO FIJO:	N/A	**TELÉFONO MÓVIL:**	0998733889

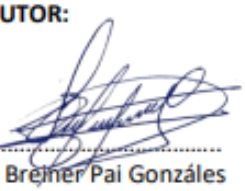
DATOS DE LA OBRA			
TÍTULO:	DESARROLLO DE UN MÓDULO WEB PARA FORTALECER LA GESTIÓN DE ACTIVIDADES ACADÉMICAS Y ADMINISTRATIVAS COMO COMPONENTE DEL ENTORNO VIRTUAL DE APRENDIZAJE INTEGRADO (EVAI) PARA LA EMPRESA IEREC.		
AUTOR (ES):	JOSE BREINER PAI GONZÁLES		
FECHA: DD/MM/AAAA	26/07/2022		
SOLO PARA TRABAJOS DE GRADO			
PROGRAMA:	[X] **PREGRADO**	[ ] **POSGRADO**	
TITULO POR EL QUE OPTA:	INGENIERÍA EN SISTEMAS COMPUTACIONALES.		
ASESOR /DIRECTOR:	MSC. COSME MACARTHUR ORTEGA BUSTAMANTE		

2. CONSTANCIAS

El autor manifiesta que la obra objeto de la presente autorización es original y se la desarrolló, sin violar derechos de autor de terceros, por lo tanto, la obra es original y que es el titular de los derechos patrimoniales, por lo que asume la responsabilidad sobre el contenido de la misma y saldrá en defensa de la Universidad en caso de reclamación por parte de terceros.

Ibarra, a los 26 días del mes de julio de 2022

EL AUTOR:



.....

José Breiner Pai Gonzáles

CERTIFICADO DEL DIRECTOR DE TRABAJO DE GRADO

UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

CERTIFICACIÓN DEL DIRECTOR

Por medio del presente yo Ing. Cosme Ortega, MSc, certifico que el Sr. José Breiner Pai Gonzáles, portador de la cédula de ciudadanía Nro. 1004198006. Ha trabajado en el desarrollo del proyecto de tesis "**Desarrollo De Un Módulo Web Para Fortalecer La Gestión De Actividades Académicas Y Administrativas Como Componente Del Entorno Virtual De Aprendizaje Integrado (EVAI) Para La Empresa IEREC.**", previo a la obtención del título de Ingeniería en Sistemas Computacionales, lo cual ha realizado en su total responsabilidad.

Es todo cuanto puedo certificar en honor a la verdad.

Atentamente:

Ing. Cosme Ortega, MSc
DIRECTOR DE TESIS

ierec
INGENIERÍA EN ENERGÍAS RENOVABLES

CERTIFICADO

Me permito informar a Ustedes que el señor **JOSÉ BREINER PAI GONZÁLES,** con cédula de ciudadanía Nro. **1004198006.** Estudiante de la Universidad Técnica del Norte, ha realizado su Trabajo de Grado con el tema: **"DESARROLLO DE UN MÓDULO WEB PARA FORTALECER LA GESTIÓN DE ACTIVIDADES ACADÉMICAS Y ADMINISTRATIVAS COMO COMPONENTE DEL ENTORNO VIRTUAL DE APRENDIZAJE INTEGRADO (EVAI) PARA LA MPRESA IEREC."** Cumpliendo con todos los requisitos reglamentarios de aprobación de la empresa, con cualidades de responsabilidad y profesionalismos.

Para efecto, se extiende el presente CERTIFICADO DE CULMINACIÓN DEL SOFTWARE, en la ciudad de Quito a los 3 días del mes de febrero de 2022.

Agradezco su atención.

Atentamente,

Ing. Guillermo Pérez Msc
CEO IER ECUADOR

0987 127 655
+593987127655
ier.energiasrenovables@gmail.com
Los Juncos 471 y Eloy Alfaro
Quito - Ecuador

Dedicatoria

Dedico el presente trabajo a todas las personas que se involucraron y creyeron en mi desde el comienzo, especialmente a mis padres y mi pareja, su ayuda ha sido fundamental, han estado conmigo incluso en los momentos más difíciles, ustedes han sido la piedra angular que me ha dado fuerzas para continuar con mi preparación académica y superar las muchas brechas que se han presentado en mi vida.

Agradecimientos

Agradezco a Dios por haberme provisto de la fuerza, paciencia y sabiduría para culminar uno de los muchos objetivos que me he planteado alcanzar. A mis padres, sus esfuerzos han sido muy invaluables en cada etapa de mi vida, han sido el mejor ejemplo de constancia y dedicación, me han brindado la oportunidad de crecer en un entorno lleno de valores y humildad fomentado e inculcado en mí el deseo de superación y como fruto de su empeño han hecho posible la culminación de mi formación profesional.

Agradezco a mi pareja por todo su apoyo, has sido una de las principales personas involucradas en ayudarme a que este proyecto fuera posible.

Agradezco a cada maestro que con su dedicación y compromiso hizo parte de este proceso integral en mi formación y sobre todo a aquellos maestros y compañeros que confiaron en mis posibilidades y con sus valiosos consejos y actitud lograron despertar el mi la curiosidad y el deseo por adquirir más conocimiento.

Tabla de contenido

Índice de Figuras

Índice de Tablas

RESUMEN

El proyecto planteado pretende solventar algunos problemas de gestión dentro de los procesos internos manejados por la empresa IERec, tales como la eficiencia en la entrega de reportes, sobrecarga de agendamientos, recurrente duplicidad, inconsistencia y perdida de datos en el proceso de cobranza los cuales causan graves problemas logísticos, desorganización y pérdida de clientes. La empresa aún no cuenta con un software que ayude a automatizar los procesos de negocio que la empresa desempeña.

Para automatizar, agilizar y optimizar los procesos de negocio, se optó por el desarrollo de un módulo web que en primera instancia pretende fortalecer la gestión de actividades académicas y administrativas como componente del entorno virtual de aprendizaje integrado (EVAI).

El proceso de construcción del software se inicia con un levantamiento de requisitos para evaluar las necesidades del cliente y las tecnologías más adecuadas para iniciar el desarrollo. Para la construcción del módulo se utilizó JavaScript como lenguaje base de programación incorporando TypeScript para proveer un tipado estricto al lenguaje.

La validación del software se realizó aplicando la ISO 25010 en la subcategoría de usabilidad mediante el cuestionario del sistema de escala de usabilidad (EUS) o por sus siglas en inglés (SUS - System Usability Scale), corresponde a un cuestionario de 10 preguntas basado en el modelo de calidad del producto que califica en un rango de 0 a 100 la aceptación de este. Se obtuvo una gran aceptación con una calificación de 84.5 que corresponde al grado B que significa que la usabilidad de software es buena y aceptable, se demuestra así que el sistema cumple con el objetivo de brindar una solución funcional y usable.

ABSTRACT

The proposed project aims to solve some management problems within the internal processes managed by the IERec company, such as the efficiency in the delivery of reports, scheduling overload, recurring duplication, inconsistency and loss of data in the collection process which cause serious logistical problems, disorganization and loss of clients. The company still does not have software that helps automate the business processes that the company performs.

To automate, streamline and optimize business processes, we opted for the development of a web module that, in the first instance, aims to strengthen the management of academic and administrative activities as a component of the virtual integrated learning environment (EVAI).

The software construction process begins with a survey of requirements to assess the client's needs and the most appropriate technologies to start development. For the construction of the module, JavaScript was used as the base programming language, incorporating TypeScript to provide a strict typing to the language.

The validation of the software was carried out by applying ISO 25010 in the usability subcategory through the usability scale system questionnaire (SUS), it corresponds to a questionnaire of 10 questions based on the product quality model that qualifies its acceptance on a range of 0 to 100. Great acceptance was obtained with a rating of 84.5, which corresponds to grade B, which means that the usability of the software is good and acceptable, thus demonstrating that the system meets the objective of providing a functional and usable solution.

INTRODUCCIÓN

Antecedentes

Los avances tecnológicos y la competitividad en el mundo globalizado de hoy, ha definido que la automatización de procesos es uno de los pilares fundamentales y de mayor importancia en diferentes áreas, en la actualidad, la administración de inventarios es primordial para las empresas, ya que en ellos se encuentra una de las mayores inversiones de la organización. Los inventarios incluyen la materia prima, productos en proceso y productos terminados, materiales y repuestos para ser consumidos en la producción de bienes fabricados para la venta o en la prestación de servicios. Se podría decir que el inventario es capital en forma de materiales, ya que éstos tienen un valor para las compañías, sobre todo para aquellas que se dedican a la venta de productos. Es por eso que el inventario es de suma importancia, ya que le permite a la empresa cumplir con la demanda y competir en el mercado (SALAZAR & MANCERA, 2017)

La empresa IERec, dedicada a buscar soluciones y exportar conocimiento sobre el uso eficiente de las energías renovables enfocado al sector térmico, en un inicio empieza brindando cursos gratuitos los cuales despertaron gran interés, es entonces que se estimula un fuerte empeño por brindar capacitaciones y asistencia técnica en esta área. Desde su fundación la empresa hace uso de software de ofimática para poder llevar un registro de calificaciones y gestión de pagos, los mismos hasta la fecha se vienen tramitando a través de redes sociales. La empresa decidió contar con varios números de la aplicación WhatsApp para tramitar los pagos, agendar citas y gestionar matriculas. Hasta el momento todos los procesos de negocio se tramitan de manera manual y no son integrados, además, no se generan reportes, los tiempos en la publicación de las notas son muy extensos, también se evidencia duplicidad y perdida de información. Todos estos inconvenientes están ocasionando un bajo desempeño en las actividades que la empresa desarrolla.

Situación Actual

Actualmente todo el proceso de cobranza, matriculas y reportes de certificados se realiza mediante la red social WhatsApp, el flujo de gestión académica es desarrollado por medio de software de ofimática y toda la promoción, publicidad y difusión de contenidos se realiza mediante redes sociales, entre ellas, Facebook como medio principal. Todos estos problemas generan un alto índice de desconfianza sobre el servicio brindado además el uso de varias tecnologías genera una sobrecarga en el modelo de gestión.

Prospectiva

La presente propuesta para el desarrollo del software pretende mejorar la eficiencia en la entrega de reportes, tratando de eliminar los tiempos muertos en la publicación de contenidos y la sobrecarga de agendamientos, además de suprimir la recurrente duplicidad, inconsistencia y perdida de datos en el proceso de cobranza que no generan confianza y credibilidad en el cliente, logrando así, mejorar notablemente los índices de desorganización en el proceso de gestión.

Planteamiento del problema

En la empresa IERec, se logra evidenciar un elevado consumo de tiempo en los procesos de gestión académico que inducen a un alto índice en la pérdida de clientes.

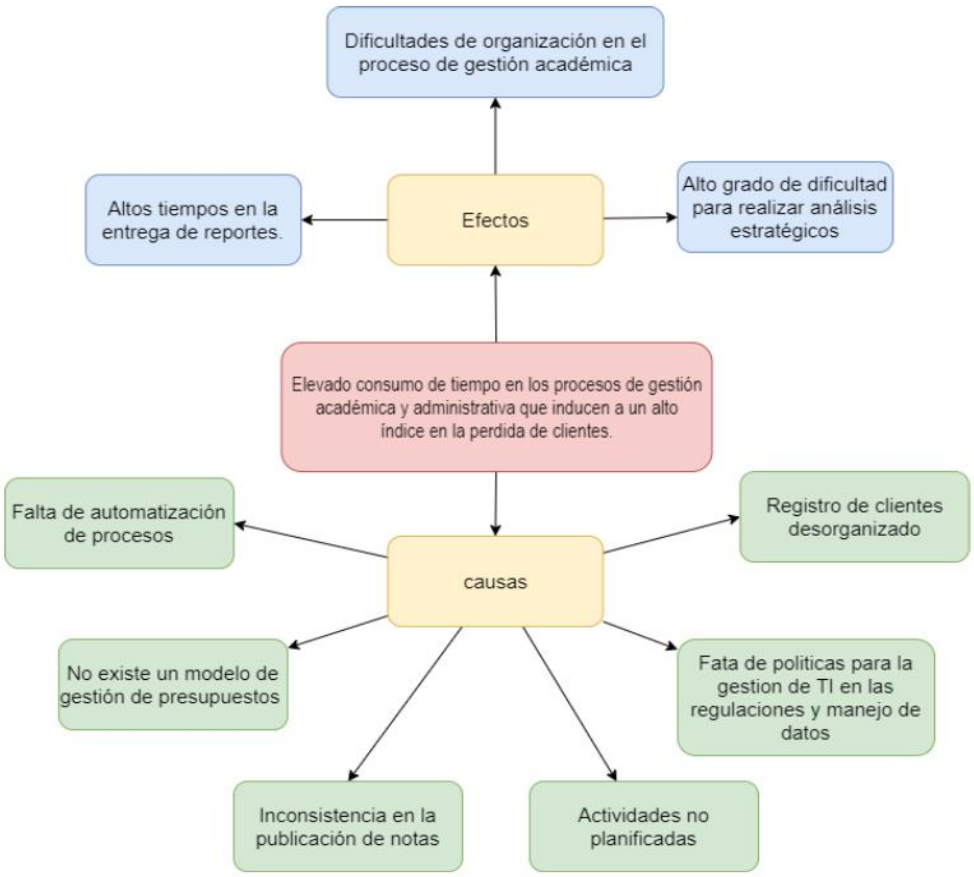


Fig. 1 Gráfico, árbol de problemas, situación actual
Fuente propia

Objetivos

Objetivo General

Desarrollar un módulo web para fortalecer la gestión de actividades académicas administrativas en la empresa IERec.

Objetivos Específicos

a) Definir un marco teórico sobre las plataformas, tecnologías y herramientas a utilizar para definir el Frontend y el Backend.

b) Diseñar el módulo para la gestión de actividades académicas y administrativas del entorno virtual de aprendizaje integrado (EVAI).

c) Validar el software utilizando la norma ISO 25010 con la subcaracterística de usabilidad.

Alcance

El software será construido bajo la arquitectura de microservicios, con el objetivo de facilitar el escalamiento del sistema. Uno de los problemas con los que se encuentran muchos sistemas en el momento de escalar es la velocidad con la que el sistema responde a un cliente (Guadalupe Cañizares-Hernández et al., 2020), tratando de evitar los tiempos muertos, se ha pensado en optar por diferentes tecnologías en el Backend, Frontend y la Base de datos más conveniente en diferentes entornos.

Se desarrollará la sección para el registro de usuarios e inicio de sesión, este contará con un bloqueo temporal de la IP del cliente en caso de tener muchas peticiones simultáneas a la API REST o tener muchos intentos fallidos en el inicio de sesión mediante el uso de tokens y variables en caché aprovechando la arquitectura de la base de datos REDIS. Además, se desea construir un registro de bitácora o logs para los accesos de peticiones a la API REST.

Para la gestión de actividades académicas se incorporará un calendario que facilite la administración del flujo del modelo de gestión manejado por la empresa. Para la gestión de actividades administrativas se desarrollará un proceso de registro de calificaciones y reporte de pagos que permitan llevar una información más organizada. El módulo de software contará con 3 tipos de roles

Backend

Se desarrollará bajo el lenguaje de programación JavaScript con el entorno en tiempo de ejecución Nodejs. Para facilitar la codificación se optó por desarrollar el código con el framework ExpressJS. Se usará la base de datos MongoDB y PostgreSQL para almacenar la información de la aplicación y Redis como base de datos de caché.

Frontend

Se decidió utilizar la librería ReactJS para el manejo de la vista pensando en la velocidad de respuesta en el sistema ya que este acelera los procesos de comunicación mutando la información únicamente en los nodos que cambian mediante la implementación de un DOM virtual (Capała & Skublewska-Paszkowska, 2018) y Redux para controlar el estado de los componentes.

Infraestructura

Para manejar la infraestructura del servidor se decide usar el sistema operativo Linux con la distribución de Ubuntu 20.04 con el servidor de aplicaciones web nginx por su versatilidad y compactibilidad con nodejs. El runtime de contenedores será controlado por Docker.

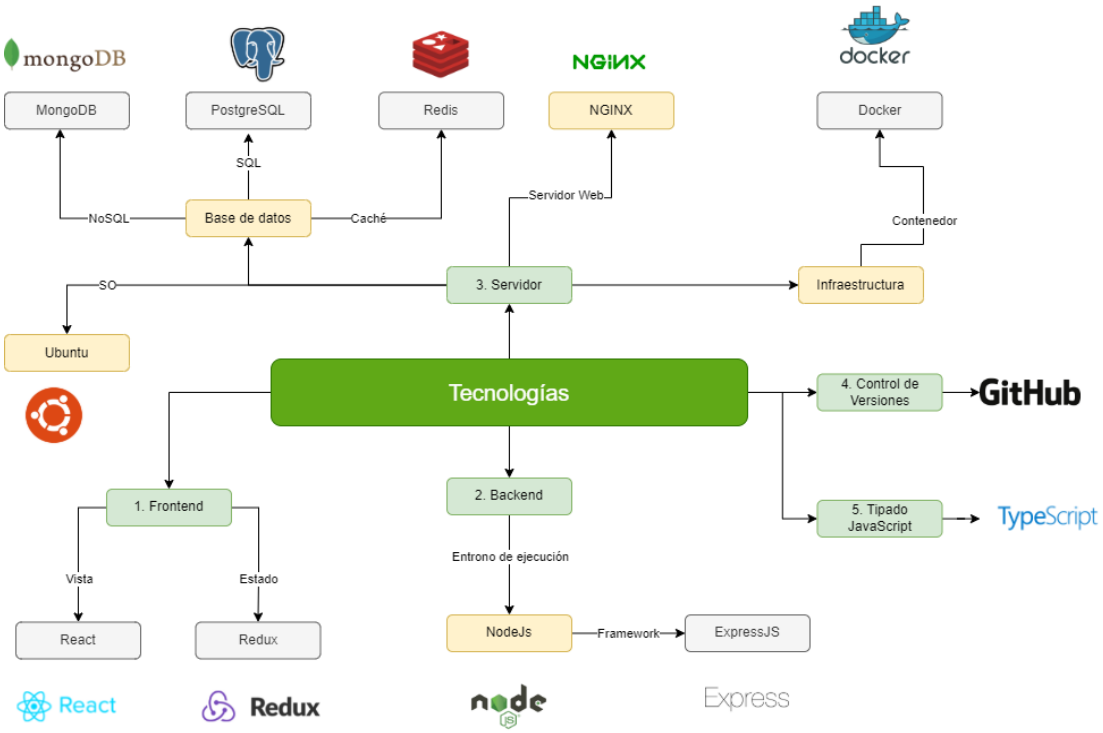


Fig. 2 Diagrama de tecnologías a usar
Fuente propia

Metodología

Para el desarrollo del presente proyecto se utilizará el tipo de investigación científica aplicada en complemento del método ingenieril.

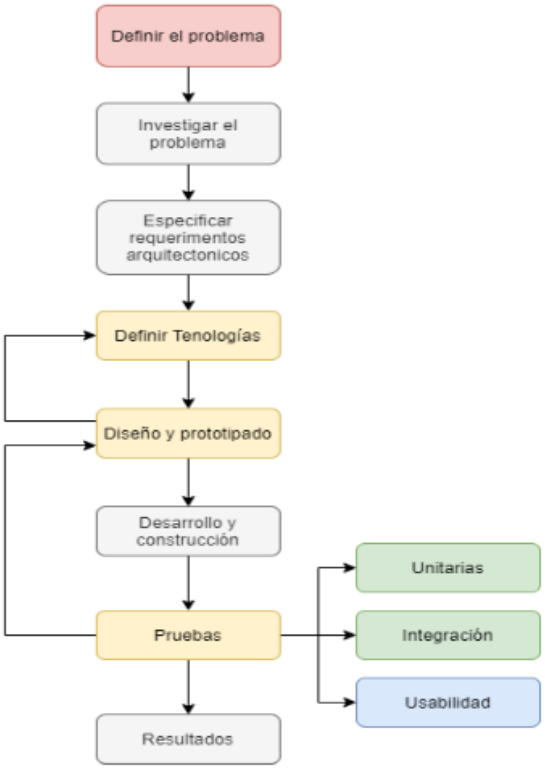


Fig. 3 Diagrama, metodología a emplear
Fuente propia

Justificación

Este proyecto se justifica por la falta de software de tipo vertical a medida que permita consultar reportes de gestión y que además integre en un mismo sistema todas las necesidades que requiere la empresa, con este enfoque se solicita el desarrollo de un módulo que permita automatizar los procesos manuales de manera adecuada, los cuales causan dificultades para realizar análisis estratégicos en tiempos apropiados. Además, el presente proyecto tiene un enfoque encaminado hacia los Objetivos de desarrollo sostenible (ODS).

Objetivo N°4: Educación de Calidad

La meta que se cumple de acuerdo con el objetivo 4 se encuentra en la sección 4.3 el cual estipula que “De aquí a 2030, asegurar el acceso igualitario de todos los hombres y las mujeres a una formación técnica, profesional y superior de calidad, incluida la enseñanza universitaria”. (Goal 4 | Department of Economic and Social Affairs, 2021)

Objetivo N°9: Industria, Innovación e Infraestructura

La meta que se cumple de acuerdo con el objetivo nueve se encuentra en la sección 9b. de los objetivos generales. “Apoyar el desarrollo de tecnologías, la investigación y la innovación nacionales en los países en desarrollo, incluso garantizando un entorno normativo propicio a la diversificación industrial y la adición de valor a los productos básicos, entre otras cosas”. (Goal 9 | Department of Economic and Social Affairs, 2021)

Justificación Tecnológica

La tecnología evoluciona constantemente y los sistemas informáticos no pueden quedarse rezagados ante el surgimiento de las nuevas tecnologías. Con el aparecimiento de la era de la web y la industria 4.0, se considera que aquellas empresas que no tienen presencia en este entorno tienen una fuerte. (Demartini & Benussi, 2017).

CAPITULO 1

Marco Teórico

1.1.1 Herramientas tecnológicas.

1.1.2 NODEJS

Ideado como un entorno de ejecución de JavaScript orientado a eventos asíncronos, Node.js está diseñado para crear aplicaciones network escalables (N. DOC, 2021).

Node.js es similar en diseño y está influenciado por sistemas como Event Machine de Ruby y Twisted de Python. Pero Node.js lleva el modelo de eventos un poco más allá. Incluye un bucle de eventos como runtime de ejecución en lugar de una biblioteca. En otros sistemas siempre existe una llamada de bloqueo para iniciar el bucle de eventos. Por lo general, el comportamiento se define mediante devoluciones callbacks de llamada al iniciarse un script y al final se inicia un servidor a través de una llamada de bloqueo como EventMachine::run(). En Node.js, no existe como tal la llamada de inicio del evento de bucle o start-the-event-loop. Node.js simplemente entra en el bucle de eventos después de ejecutar el script de entrada y sale cuando no hay más devoluciones callbacks de llamada para realizar. Se comporta de una forma similar a JavaScript en el navegador - el bucle de eventos está oculto al usuario (N. DOC, 2021).

Según el artículo (Hota y Prabhu, 2001). La plataforma Node.js consta de tres niveles. El nivel básico contiene todos los componentes principales, el nivel medio actúa como middleware estableciendo comunicación desde el nivel inferior al nivel superior.

El último nivel superior consta de todas las API de JavaScript. Los componentes principales son los siguientes:

a) V8 - Código abierto Motor de JavaScript desarrollado por Google

b) Libev – Implementos bucle de eventos y abstrae el específico subyacente uso de tecnologías select, epoll.

El motor v8 JavaScript es subyacente que Google usa con su navegador Chrome. Es un intérprete ultra-rápido escrito en C++, se puede descargar el motor e incorporarle a cualquier aplicación que se desee.

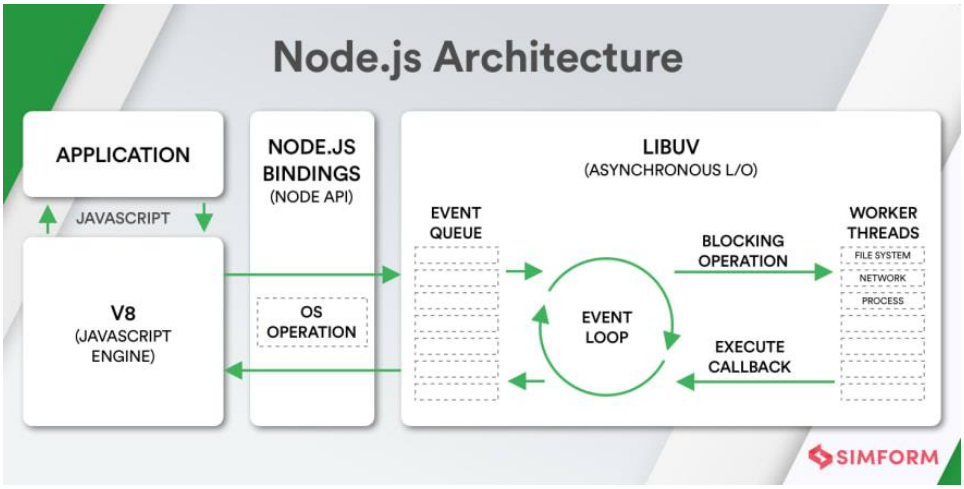


Fig. 4 arquitectura de NodeJS
Fuente adaptado de: (*5 Reasons Why You Should Consider Node.Js - DEV Community*, n.d.)

En los últimos años, Node.js ha ganado bastante popularidad por diversas razones. Ha atraído a una amplia gama de empresas. Entre las empresas que han implementado Node.js se encuentran Amazon, eBay, Reddit, Netflix, LinkedIn, Tumblr y PayPal.

Netflix informó que hubo una disminución sustancial en su tiempo de inicio después de seleccionar Node.js. Mientras que Amazon declaró que Node.js fue elegido por sus características *futuristas.*(*5 Reasons Why You Should Consider Node.Js - DEV Community*, n.d.)

1.1.1.1 NPM

Node Package Manager (npm) es una utilidad incluida en Node.js que proporciona una serie de componentes reutilizables disponibles públicamente que están disponibles a través de una instalación simple de a través de un repositorio en línea con versiones y dependencias. npm es el registro de software más grande del mundo. Los desarrolladores de código abierto de todos los continentes usan npm para compartir y tomar prestados paquetes, y muchas organizaciones también usan npm para administrar el desarrollo privado.(*About Npm | Npm Docs*, 2021)

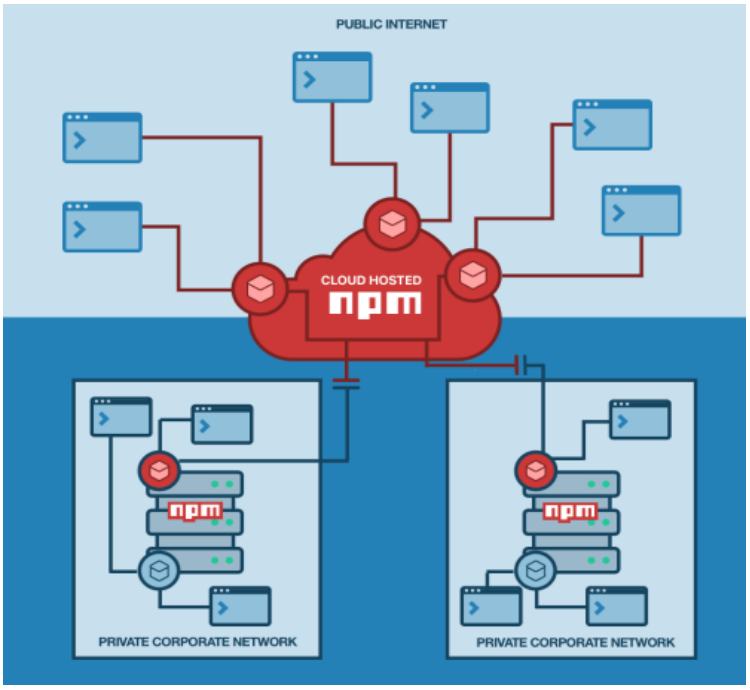


Fig. 5 Arquitectura npm en la nube
Fuente adaptado de: (Edge Node Architecture with Npm Enterprise | by Npm, Inc. | Medium, 2021)

1.1.1.2 YARN

Yarn es un administrador de paquetes que también se desempeña como administrador de proyectos. Ya sea que trabaje en proyectos únicos o grandes monorepos, como aficionado o usuario empresarial, lo tiene cubierto. Permite usar y compartir código con otros desarrolladores de todo el mundo. Yarn hace esto de manera rápida, segura y confiable para que el desarrollador no tenga que preocuparse nunca, también permite utilizar las soluciones de otros desarrolladores para diferentes problemas, lo que le facilita el desarrollo de software. Si se tiene problemas, los desarrolladores puede informar las dificultades o contribuir, y cuando el problema se solucione, puede usar Yarn para mantener todo actualizado. (*1 - Introduction | Yarn - Package Manager*, 2021)

1.1.2 MONGODB

MongoDB es una base de datos de documentos que ofrece una gran escalabilidad y flexibilidad, y un modelo de consultas e indexación avanzado (M. DOC, 2021).

MongoDB almacena datos en documentos flexibles similares a JSON, por lo que los campos pueden variar entre documentos y la estructura de datos puede cambiarse con el tiempo (M. DOC, 2021).

1.1.3 DOCKER

Las herramientas del contenedor, como Docker, ofrecen un modelo de implementación basado en imágenes. Esto permite compartir una aplicación, o un conjunto de servicios, con todas sus dependencias en varios entornos. Docker también automatiza la implementación de la aplicación (o conjuntos combinados de procesos que constituyen una aplicación) en este entorno de contenedores (R. DOC, 2021).

Docker es lo más parecido a una máquina virtual, pero sin la necesidad de incorporar la multitud de procesos que se necesitan para que un sistema operativo funcione, en lugar de eso, solo incluye las librerías y dependencias que se necesitan para el proyecto en cuestión funcione bien, de esta manera Docker ayuda a los desarrolladores a dar vida a sus ideas al conquistar la complejidad del desarrollo de aplicaciones y dando solución a la problemática de la programación y operación necesaria para que dicho programa se ejecute correctamente con todas sus dependencias en cualquier entorno. simplifica y acelera los flujos de trabajo en el desarrollo mediante la consolidación de los componentes de la aplicación. Se utiliza activamente por millones de desarrolladores en todo el mundo, Docker Desktop y Docker Hub brindan una simplicidad, agilidad y opciones incomparables (D. DOC, 2021).

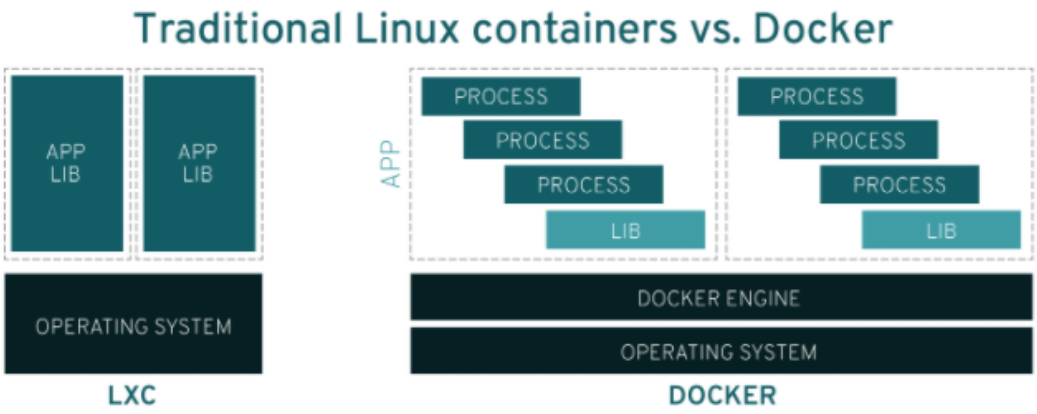


Fig. 6 Comparativa entre contenedores y sistemas operativos Linux tradicionales.
Fuente adaptado de: (R. DOC, 2021).

1.1.4 BONITA STUDIO

Bonita Studio es un software que acelera el desarrollo, producción y mantenimiento de proyectos de automatización. Si bien permite a los usuarios realizar tareas que afectan sus datos comerciales, también se administra de manera competente, se integra con los sistemas de información existentes y organiza sistemas heterogéneos, algunos de los cuales son robots blandos. Proporciona información detallada sobre la ejecución de procesos en toda la empresa a través de sus aplicaciones integradas para el usuario final o aplicaciones en vivo creadas por el equipo del proyecto para satisfacer perfectamente las necesidades comerciales. (*What Is Bonita? | Bonita Documentation*, n.d.)

1.1.5 MODELIO

Modelio es una herramienta UML de código abierto desarrollada por Modeliosoft, diseñada para la correcta creación de modelos UML. Modelio es principalmente un entorno de modelado que admite una amplia gama de modelos y diagramas y proporciona soporte de modelos y funciones de verificación de coherencia.(*Modelio*, 2020)

1.1.6 PostgreSQL

PostgreSQL, un potente sistema de base de datos relacional de objetos y de código abierto. Cuenta con más de años de desarrollo activo y una arquitectura probada que se ha ganado una sólida reputación de confiabilidad e integridad de datos. Se ejecuta en los principales sistemas operativos existentes en la actualidad, como Linux, Unix, Windows. Es totalmente compatible con ACID, tiene soporte completo para claves foráneas, uniones, vistas, disparadores y procedimientos almacenados (en varios idiomas). Incluye la mayoría de los tipos de datos SQL 2008.

PostgreSQL viene con muchas características diseñadas para ayudar a los desarrolladores a construir aplicaciones, los administradores protegen la integridad de los datos y crean entornos tolerantes a fallas y lo ayudan a administrar sus datos sin importar cuán grande o pequeño sea el conjunto de estos. Además de ser gratuito y de código abierto, PostgreSQL es altamente extensible. (*PostgreSQL: About*, 2019)

PostgreSQL intenta ajustarse al estándar SQL cuando dicha conformidad no entra en conflicto con las funciones tradicionales de o podría conducir a malas decisiones de arquitectura. Muchas de las funciones requeridas por el estándar SQL son compatibles, aunque a veces con una sintaxis o función ligeramente diferente. Espere un mayor progreso hacia el cumplimiento de a lo largo del tiempo. A partir del lanzamiento de la versión 12 en octubre de 2019, PostgreSQL-160 corresponde a las 179 funciones obligatorias para SQL: 2016 Core. Al momento de escribir este artículo, ninguna base de datos relacional cumple completamente con este estándar (*PostgreSQL: About*, 2019)

1.2 Arquitectura de Microservicios.

1.2.1 Microservicios.

En los últimos 3 o 4 años se ha acentuado a nivel mundial el desarrollo de aplicaciones Web como microservicios. Esta estrategia de desarrollo es una variante de SOA, con énfasis en la creación de pequeños sistemas desacoplados e intercomunicados, lo que conlleva múltiples beneficios en comparación a las aplicaciones Web monolíticas tradicionales. Uno de los beneficios más importantes del uso de microservicios es que el desarrollo, la prueba y la implementación de estas aplicaciones es mucho más rápido, lo que les permite responder rápidamente a las necesidades del usuario, esto se logra por que la distribución de estas aplicaciones es mucho más expedita, es decir, no se encuentran

trabas ni obstáculos. Además, permite la independencia de las tecnologías a emplear en cada uno de los componentes, tales como lenguajes de programación, frameworks de desarrollo y servicios adicionales que pudiesen ser necesarios, sin repercutir en otros componentes (Navarro et al., 2017).

Los microservicios son la realización de un estilo de arquitectura orientada a servicios como SOA, con la diferencia en que se crea software que se compone de pequeños servicios autónomos, los cuales se pueden implementar y escalar de forma independiente. Cada microservicio tiene una funcionalidad empresarial independiente, se ejecuta como un proceso independiente y se comunica a través de mecanismos ligeros (Vayghan et al., 2021).

1.2.2 Requisitos

Uno de los requisitos básicos de la arquitectura de microservicios es que todos los componentes puedan comunicarse entre sí mediante un protocolo común. La mayoría de los servicios ahora hacen esto enviando una solicitud a un conjunto particular de puntos finales mediante el protocolo HTTP. que corresponden a la Application Programming Interface (API) de los sistemas, generalmente proporcionan información en formato JSON. Estos endPoints pueden ser públicos o privados. En el segundo caso, se requiere autenticación a través de un mecanismo como OAuth o JSON Web Token. Este esquema de trabajo también permite la comunicación entre aplicaciones de escritorio y móviles sin repetir el trabajo de desarrollo (Navarro et al., 2017).

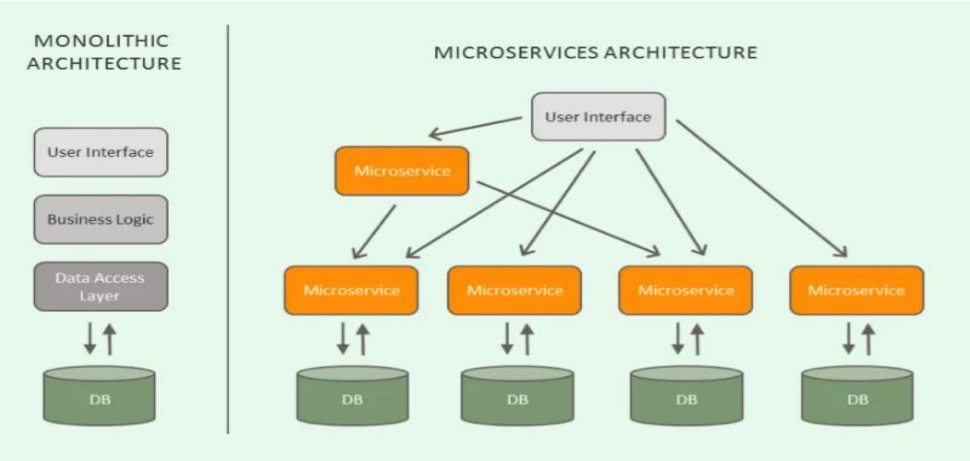


Fig. 7 Comparación complejidad en la arquitectura de microservicios y un monolito
Fuente adaptado de: (Auribox, 2017).

1.3 Metodología de desarrollo

La metodología es una palabra compuesta por tres palabras griegas. Meta que significa más allá, dos que significa camino y logos que significa estudio; considerando lo anterior como el estudio más allá del camino (Montero et al., 2018).

1.3.1 Metodología ágil.

Los métodos de desarrollo de software ágiles proporcionan un modelo de desarrollo iterativo y escalable con un enfoque en los requisitos cambiantes, la satisfacción del cliente y la colaboración en equipo. (Anwer et al., 2017).

Los modelos ágiles son en realidad una colección de mejores prácticas y principios de ingeniería de software.

Estas metodologías surgieron en 2001 en respuesta a las limitaciones de la metodología basada en planes. Debido a la alta tasa de fallas, cancelaciones y retrasos de proyectos, los profesionales del software deben coordinar los principios y prácticas de desarrollo (Anwer et al., 2017).

Si bien estos principios pueden no ser nuevos para la industria del software, el modelado ágil utiliza un enfoque diferente para aumentar la flexibilidad y la adaptabilidad durante el desarrollo y puede ajustarse a la medida de sus necesidades crecientes del desarrollo de software rápido.

Los métodos ágiles son flexibles como su función principal, y los proyectos de desarrollo se dividen en proyectos más pequeños que incluyen la comunicación continua con los usuarios, son altamente colaborativos y es mucho más adaptable a los cambios. De hecho, el cambio de requerimientos por parte del cliente es una característica especial, así como también las entregas, revisión y retroalimentación constante (Montero et al., 2018).

1.3.2 Metodología de desarrollo ágil XP.

La metodología ágil más famosa es Extreme Programming (XP). Fue desarrollada por Kent Beck para liderar un grupo de trabajo pequeño o mediano de 2 a 10 programadores en entornos donde los requisitos son imprecisos o cambiantes. La característica principal de este enfoque son las historias de

usuario que corresponde al enfoque de especificación de requisitos, esta es la forma en la que el cliente describe las características y funciones requeridas para el sistema (Montero et al., 2018).

Se le llamó "programación extrema" porque llevó al límite los métodos destinados a ayudar a desarrollar software de alta calidad. XP mejora drásticamente la satisfacción del cliente. La retroalimentación oportuna y la publicación periódica ayudan a gestionar los defectos cercanos al origen. Las bajas tasas de falla reducen los costos de desarrollo y producen productos finales más aceptables a un costo menor (Anwer et al., 2017).

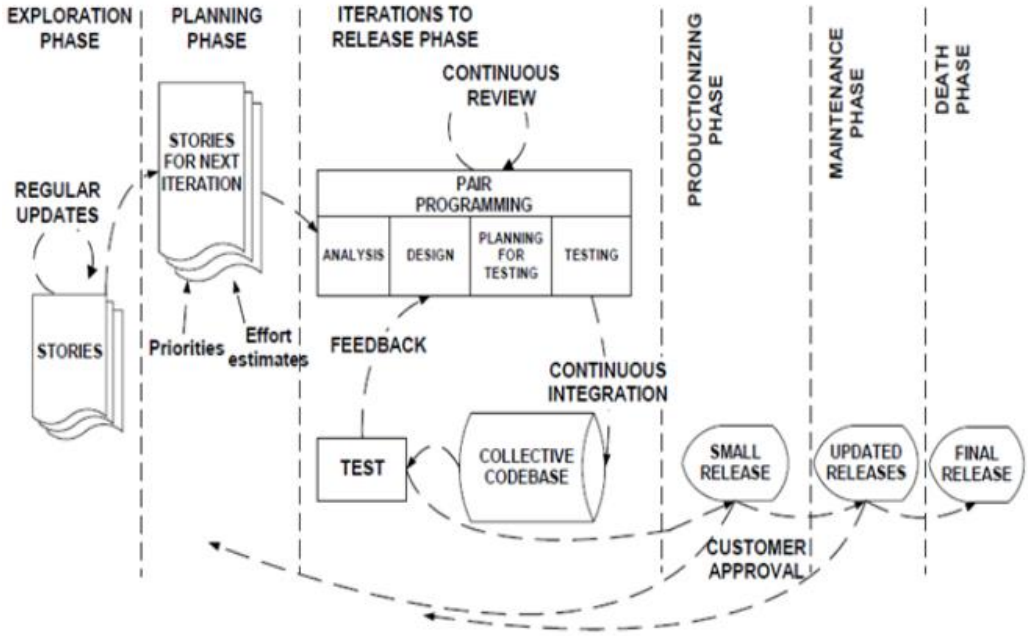


Fig. 8 Ciclo de vida de la programación extrema
Fuente adaptado de: (Anwer et al., 2017)

1.3.2.1 Planeación.

Cada iteración comienza con una planificación para dicha acción. En este punto, el desarrollador crea un plan de trabajo para implementar la funcionalidad requerida para la versión actual. Al igual que el plan de lanzamiento, el plan iterativo incluye las fases de investigación, participación y liderazgo, pero el cliente no participa en este paso. En la planificación iterativa, el planificador selecciona la actividad a realizar y estima el costo, el tiempo y el esfuerzo necesarios para

la actividad seleccionada. Puede delegar tareas a otros programadores para equilibrar su carga de trabajo (Anwer et al., 2017).

1.3.2.2 Diseño.

El diseño XP sigue rigurosamente el principio MS (mantenlo sencillo). Los diseños simples siempre tienen prioridad sobre las expresiones más complejas. Además, el diseño sirve de guía para la ejecución de la historia (Cevallos, 2016).

1.3.2.3 Codificación.

Un concepto importante en la codificación (y uno de los aspectos más comentados de XP) es la programación por parejas. En XP, se recomienda que dos personas trabajen juntas en su estación de trabajo para codificar la historia. Una vez que la pareja de desarrolladores ha completado su trabajo, el código que desarrollan se integra en el trabajo de los demás. En algunos casos, esto lo hace a diario un equipo de integración. En otros casos, la pareja de desarrolladores es responsable de la integración. Esta estrategia de "integración continua" evita problemas de compatibilidad de la interfaz y proporciona un entorno "a prueba de humo" para la detección temprana de fallas (Cevallos, 2016).

1.3.2.4 Pruebas.

La creación de pruebas unitarias antes de que comience la codificación es un elemento clave del enfoque de XP, ya que esto asegura la calidad del software (Cevallos, 2016).

1.3.2.5 Producción.

XP es un proceso incremental iterativo, por lo que proporciona pequeñas versiones del software. Las versiones son pequeños programas que satisfacen determinadas necesidades comerciales. Los Lanzamientos frecuentes en XP, permiten construir el sistema que se necesita en etapas. Un ciclo de lanzamiento puede incluir una serie de iteraciones que duran de 1 a 4 semanas. En la fase de producción, el software se implementa en versiones más pequeñas. Se realizan pruebas

de aceptación, de carga y de sistema para determinar que el software está listo. En esta fase, el programador ralentiza el desarrollo del sistema a medida que aumenta el riesgo determinando si es necesario incluir los nuevos cambios en la próxima versión (Anwer et al., 2017).

1.4 Norma ISO 25010

Hoy en día, el software se encuentra en diferentes ámbitos de la actividad humana, por lo que las empresas necesitan no solo definir estándares específicos de calidad, sino también conocer la percepción de los usuarios para poder cumplir verdaderamente con los requisitos (Cruz et al., 2020).

Hoy en día, casi todas las áreas de la actividad humana involucran computadoras que ejecutan software y sistemas de información de aplicaciones. La alta calidad es un factor importante en el uso eficiente del software y es uno de los principales requisitos de los usuarios y las partes interesadas para el software moderno.

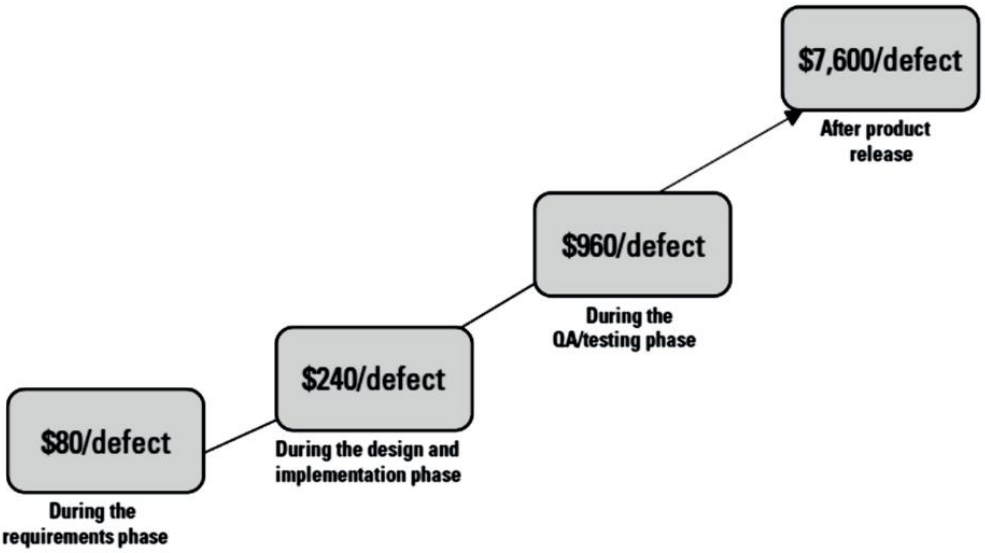


Fig. 9 El costo de corregir defectos aumenta drásticamente a lo largo del proceso de desarrollo
Fuente adaptado de: (Hovorushchenko, 2018)

La calidad del software es el grado de satisfacción de las necesidades del cliente. La investigación sobre evaluación y análisis de software muestra que la crisis en la evaluación y el aseguramiento de la calidad del software continúa hoy. Los requisitos exactos del software definen

las características esenciales de la calidad del software y su impacto en los métodos cuantitativos de evaluación de la calidad del software. Los proyectos de software con requisitos y especificaciones incompletos no podrán implementarse con éxito. Muchas fallas de software ocurren durante la fase de desarrollo de requisitos (10 – 23% de todos los defectos de software). Cuanto antes se detecte la falla, menor será el costo de reparación. Los defectos descubiertos después del lanzamiento del producto pueden costar casi 100 veces más de reparación que los defectos encontrados durante la gestión de reclamaciones (Hovorushchenko, 2018).

Tratando de prevenir los costos elevados que provoca un software mal desarrollado, nace un estándar para ISO, específicamente la ISO 25010

1.4.1 ISO 25010.

La norma ISO/IEC 25010 es un estándar internacional que define modelos de calidad para productos de software. El modelo de calidad es la base para construir un sistema de evaluación de la calidad del producto. Este modelo define las características de calidad que se tienen en cuenta al evaluar los atributos de un producto de software en particular (Cruz et al., 2020).

La norma ISO/IEC 25010 no incluye la evaluación de la funcionalidad y características específicas de una aplicación, ya que su valor solo puede medirse en relación con las necesidades de la organización adquirente. Por lo tanto, el modelo de Calidad en Uso busca cuantificar la "usabilidad" (efectividad, eficiencia y satisfacción) de la aplicación, cuando usuarios específicos intentan alcanzar sus objetivos específicos. La norma define la efectividad como la "Precisión e integridad con la que los usuarios logran objetivos específicos", y eficiencia como "recursos gastados en relación con la precisión e integridad con la que los usuarios logran objetivos" (Estdale & Georgiadou, 2018).

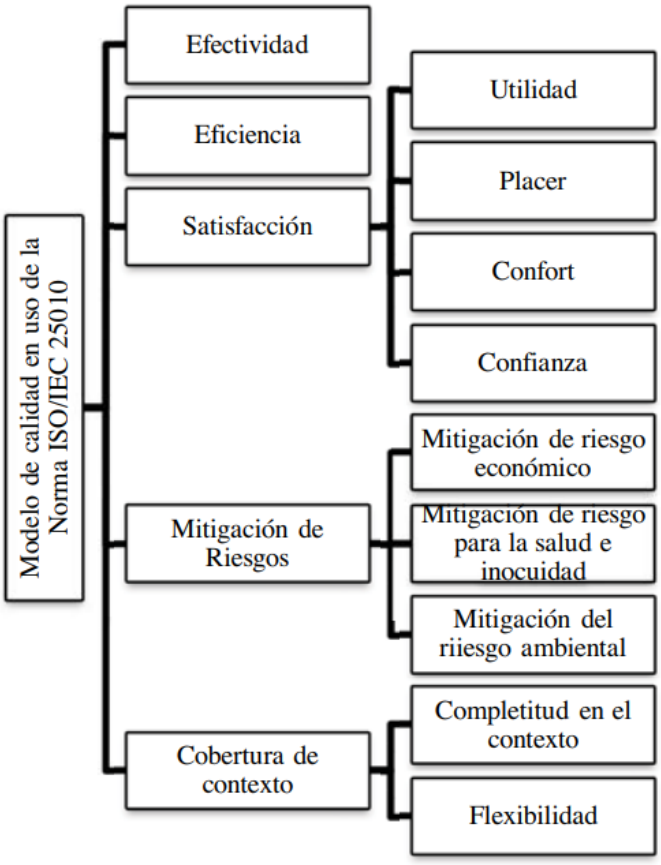


Fig. 10 Modelo de calidad de uso de la Norma ISO/IEC 25010
Fuente adaptado de: (Cruz et al., 2020).

1.4.2 Calidad del producto.

La calidad de un producto de software puede entenderse como la medida en que ese producto cumple con los requisitos del usuario y proporciona valor. Son exactamente estos requisitos (funcionalidad, rendimiento, seguridad, mantenibilidad, etc.) (ISO, 2015).

El rendimiento incluye el tiempo, el uso de recursos y la capacidad. Para los productos de software, es posible que las medidas tomadas durante el desarrollo no utilicen la plataforma o el entorno exacto del adquiriente, lo que dificulta la estimación de los servicios ofrecidos (Estdale & Georgiadou, 2018).

1.4.3 Compactibilidad.

Capacidad de dos o más sistemas o componentes para intercambiar información y/o llevar a cabo sus funciones requeridas cuando comparten el mismo entorno hardware o software (ISO, 2015).

Esto se puede dividir en interoperabilidad entre aplicaciones, el intercambio, uso de información y la coexistencia. El impacto sobre otros productos en la misma plataforma. Teniendo en cuenta que se refieren a cómo se implementan la funcionalidad, por lo que puede ser mejor tratarlas como "características" en lugar de funciones. La coexistencia no es un concepto puramente pasivo. Incluso después de la llegada de los sistemas operativos de programación múltiple debidamente protegidos en las PC, sigue siendo importante que un producto se "comporte bien" en lugar de "se comporte mal", por lo que debe utilizar los servicios suministrados para compartir recursos de manera cooperativa con las otras aplicaciones (no controladas) de la plataforma (Estdale & Georgiadou, 2018).

CAPITULO 2

DESARROLLO DEL SISTEMA WEB

Introducción:

Para iniciar con la construcción del módulo web se pensó en la utilización de una metodología ágil que permita aumentar la productividad y flexibilidad de desarrollo, como consecuencia se escoge a la metodología XP para el desarrollo de la propuesta. En la ya mencionada metodología se establece 4 etapas para agilizar la gestión del desarrollo del software, dichas etapas se representan a continuación.

2.1 Planificación:

Se define el grupo de trabajo que será parte de todo el proceso de desarrollo, en dicho proceso es importante la colaboración activa del cliente con la finalidad de realizar cambios de manera temprana y evitar errores en el software. De esta forma, se realiza un análisis preliminar del problema y se identifican historias de usuarios, las cuales están categorizadas por prioridad.

2.1.1 Participantes del proyecto

- **Programador**: Encargado del desarrollo del desarrollo para la aplicación web
- **Tester**: Encargado de realizar las pruebas unitarias y de integración.
- **Consultor**: Miembro externo al equipo, con conocimientos en temas relacionados con el proyecto.
- **Cliente**: Determina la funcionalidad que se pretende en el desarrollo.
- **Tracker**: Encargado del seguimiento en el proceso de desarrollo.

Tabla 1

Participantes del proceso de desarrollo según la metodología XP

Miembro	Roles
Est. José Pai	Programador, Tester
Msc. Cosme Ortega	Consultor, Tracker
Ing. Guillermo Pérez	Cliente

2.1.2 Historias de usuario.

Para el levantamiento de requisitos se consulta a cada uno de los participantes del proyecto solicitando los requerimientos que debe cumplir el sistema, los mismos se organizan en historias de usuario y se categorizan por el nivel de importancia en las diferentes iteraciones.

Tabla 2

Listado de historias de usuario

Nro.	Nombre	Prioridad	Riesgo	Iter.
H01	Diseño y elaboración de base de datos de autenticación.	Alto	Alto	1
H02	Diseño y elaboración del servicio backend de autenticación.	Alto	Medio	1
H03	Configuración de Docker.	Medio	Medio	1
H04	Implementación de tokens JWT	Alto	Alto	2
H05	Enrutamiento en servicio backend de autenticación.	Medio	Medio	2
H06	Desarrollo de middlewares y restricción de acceso a recursos	Medio	Alto	2
H07	Desarrollo de vistas para el bloqueo y desbloqueo de usuarios	Bajo	Bajo	2
H08	Gestión de registros de bitácora (Logs)	Medio	Bajo	3
H09	Diseño y elaboración de vistas de SignIn y SignUp	Medio	Medio	3
Nro.	Nombre	Prioridad	Riesgo	Iter.

H10	Diseño y elaboración de panel de contraseña perdida	Medio	Alto	3
H11	Diseño y elaboración del perfil de usuario	Bajo	Medio	3
H12	Gestión de pagos en el perfil de usuario	Medio	Alto	3
H13	Gestión de pagos para el perfil de administrador	Medio	Alto	4
H14	Diseño y elaboración de base de datos de cursos	Alto	Alto	4
H15	Diseño y elaboración de Backend de cursos.	Alto	Medio	4
H16	Construcción de rutas del servicio backend de cursos	Medio	Medio	4
H17	Diseño y elaboración de vistas para componentes de frontend	Alto	Alto	5
H18	Implementación de calendario de eventos.	Medio	Bajo	5
H19	Diseño y desarrollo del dashboard	Alto	Medio	5

2.1.2.1 Iteraciones

Tabla 3

Lista de historia de usuarios en la primera iteración

Nro.	Nombre	Semanas
H01	Diseño y elaboración de base de datos de autenticación.	1
H02	Diseño y elaboración del servicio backend de autenticación.	1
H03	Configuración de Docker.	0.5

Tabla 4

Lista de historia de usuarios en la segunda iteración

Nro.	Nombre	Semanas
H04	Implementación de tokens JWT	1.5
H05	Diseño y elaboración del servicio backend de autenticación.	0.5
H06	Desarrollo de middlewares y restricción de acceso a recursos	.5
H07	Desarrollo de vistas para el bloqueo y desbloqueo de usuarios	1.5

Tabla 5

Lista de historia de usuarios en la tercera iteración

Nro.	Nombre	Semanas
H08	Gestión de registros de bitácora (Logs)	1
H09	Diseño y elaboración de vistas de SignIn y SignUp	1.5
H10	Diseño y elaboración de panel de contraseña perdida	1
H11	Diseño y elaboración del perfil de usuario	1
H12	Gestión de pagos en el perfil de usuario	1

Tabla 6

Lista de historia de usuarios en la cuarta iteración

Nro.	Nombre	Semanas
H13	Gestión de pagos para el perfil de administrador	1
H14	Diseño y elaboración de base de datos de cursos	1

Nro.	Nombre	Semanas
H15	Diseño y elaboración de Backend de cursos.	1.5
H16	Construcción de rutas del servicio backend de cursos	1

Tabla 7

Lista de historia de usuarios en la quinta iteración

Nro.	Nombre	Semanas
H17	Diseño y elaboración de vistas para componentes de frontend	4
H18	Implementación de calendario de eventos.	1
H19	Diseño y desarrollo del dashborard	1.5

2.1.2.2 Historias de usuario

Tabla 8

Historia de usuario H01

Historia de Usuario	
Nro: 1	**Usuario**: Desarrollador
Nombre:	Diseño y elaboración de base de datos de autenticación.
Iteración: 1	**Prioridad:** Alto
Riesgo: Alto	**Esfuerzo:** Medio
Descripción:	Diseñar una base de datos con el fin de autenticar a los usuarios en el sistema.

Tabla 9

Historia de usuario H02

Historia de Usuario	
Nro: 2	**Usuario**: Desarrollador
Nombre:	Diseño y elaboración del servicio backend de autenticación.
Iteración: 1	**Prioridad:** Alto
Riesgo: Medio	**Esfuerzo:** Medio
Descripción:	Diseñar un servicio backend que sirva y gestione el contenido almacenado en la base de datos de autenticación.

Tabla 10

Historia de usuario H03

Historia de Usuario	
Número: 3	**Usuario**: Desarrollador
Nombre:	Configuración de Docker.
Iteración: 1	**Prioridad:** Medio
Riesgo: Medio	**Esfuerzo:** Medio
Descripción:	Configuración de los ambientes de desarrollo, pruebas y producción en Docker.

Tabla 11

Historia de usuario H04

Historia de Usuario	
Número: 4	**Usuario**: Desarrollador
Nombre:	Implementación de tokens JWT.
Iteración: 2	**Prioridad:** Alto
Riesgo: Alto	**Esfuerzo:** Bajo
Historia de Usuario	
Descripción:	Gestión de algoritmos para el intercambio de tokens, implementación de 3 tipos de token en la API de autenticación: token de acceso, token de refresco y token de cambio de contraseña.

Tabla 12

Historia de usuario H05

Historia de Usuario	
Número: 5	**Usuario**: Desarrollador
Nombre:	Enrutamiento en servicio backend de autenticación.
Iteración: 2	**Prioridad:** Medio
Riesgo: Medio	**Esfuerzo:** Medio
Descripción:	Diseñar y elaborar el sistema de rutas por cada rol asignado en el sistema.

Tabla 13

Historia de usuario H06

Historia de Usuario	
Número: 6	**Usuario**: Desarrollador
Nombre:	Desarrollo de middlewares y restricción de acceso a recursos.
Iteración: 2	**Prioridad:** Medio
Riesgo: Alto	**Esfuerzo:** Medio
Descripción:	Diseñar y desarrollar los sistemas middlewares para cada ruta implementada de acuerdo con el rol asignado en el sistema.

Tabla 14

Historia de usuario H07

Historia de Usuario	
Número: 7	**Usuario**: Cliente
Nombre:	Desarrollo de vistas de administrador – bloqueo y desbloqueo de usuarios.
Iteración: 2	**Prioridad:** Bajo
Riesgo: Bajo	**Esfuerzo:** Medio
Descripción:	Como administrador del sitio web, requiero poder tener acceso a un listado completo de los usuarios para poder evaluar y eliminar el bloqueo aplicado a diferentes cuentas.

Tabla 15

Historia de usuario H08

Historia de Usuario	
Número: 8	**Usuario**: Cliente
Nombre:	Gestión de registros de bitácora (Logs)
Iteración: 3	**Prioridad:** Medio
Riesgo: Bajo	**Esfuerzo:** Bajo
Descripción:	Como administrador del sitio web, requiero poder tener acceso a la información de bitácora o registros logs del sistema, para dar un seguimiento y control oportuno.

Tabla 16

Historia de usuario H09

Historia de Usuario	
Número: 9	**Usuario**: Cliente
Nombre:	Diseño y elaboración de vistas de SignIn (Inicio de sesión) y SignUp (Registro)
Iteración: 3	**Prioridad:** Medio
Historia de Usuario	
Riesgo: Medio	**Esfuerzo:** Medio
Descripción:	Como usuario requiero un panel que me permita iniciar sesión y registrarme en el sistema, deseo poder acceder con una cuenta local (correo y contraseña) o a través de las redes sociales de Facebook y Google.

Tabla 17

Historia de usuario H10

Historia de Usuario	
Número: 10	**Usuario**: Cliente
Nombre:	Diseño y elaboración de panel de contraseña perdida
Iteración: 3	**Prioridad:** Medio
Riesgo: Alto	**Esfuerzo:** Medio
Descripción:	Como usuario del sitio web requiero poder visualizar un panel que me permita recuperar la contraseña perdida a través del correo electrónico registrado en mi cuenta, además una vez que se genere la petición, la acción solo pueda llevarse a cabo durante 10 minutos.

Tabla 18

Historia de usuario H11

Historia de Usuario	
Número: 11	**Usuario**: Cliente
Nombre:	Diseño y elaboración del perfil de usuario.
Iteración: 3	**Prioridad:** Bajo
Riesgo: Medio	**Esfuerzo:** Medio
Descripción:	Como usuario requiero poder acceder a la información personal a la que el sitio web tiene acceso para editar y tener control sobre la información que comparto.

Tabla 19

Historia de usuario H12

Historia de Usuario	
Número: 12	**Usuario**: Cliente
Nombre:	Gestión de pagos - usuario.
Iteración: 3	**Prioridad:** Medio
Riesgo: Alto	**Esfuerzo:** Medio
Historia de Usuario	
Descripción:	Como usuario requiero poder acceder a los comprobantes de pagos de cada transacción realizada, además de poder descargar dicha información en formato PDF.

Tabla 20

Historia de usuario H13

Historia de Usuario	
Número: 13	**Usuario**: Cliente
Nombre:	Gestión de pagos - administrador.
Iteración: 4	**Prioridad:** Medio
Riesgo: Alto	**Esfuerzo:** Medio
Descripción:	Como usuario administrador requiero poder acceder a todos los comprobantes de pagos efectuados por cada usuario. además de poder descargar dicha información en formato PDF.

Tabla 21

Historia de usuario H14

Historia de Usuario	
Número: 14	**Usuario**: Consultor
Nombre:	Diseño y elaboración de base de datos de cursos
Iteración: 4	**Prioridad:** Alto
Riesgo: Alto	**Esfuerzo:** Medio
Descripción:	Diseñar y elaborar una base de datos en SQL para almacenar la información relacionada con los cursos.

Tabla 22

Historia de usuario H15

Historia de Usuario	
Número: 15	**Usuario**: Desarrollador
Nombre:	Diseño y elaboración de Backend de cursos.
Iteración: 4	**Prioridad:** Alto
Riesgo: Medio	**Esfuerzo:** Medio
Descripción:	Diseñar y elaborar un servicio backend para el posterior consumo de la API por el frontend del sistema.

Tabla 23

Historia de usuario H16

Historia de Usuario	
Número: 16	**Usuario**: Desarrollador
Nombre:	Construcción de rutas del servicio backend de cursos.
Iteración: 4	**Prioridad:** Medio
Historia de Usuario	
Riesgo: Medio	**Esfuerzo:** Alto
Descripción:	Construcción de rutas del servicio backend de cursos, se usa el token de acceso generado por el servicio de autenticación para dar acceso a los recursos de la API a través de middlewares.

Tabla 24

Historia de usuario H17

Historia de Usuario	
Número: 17	**Usuario**: Desarrollador
Nombre:	Diseño y elaboración de vistas para componentes de frontend.
Iteración: 5	**Prioridad:** Alto
Riesgo: Alto	**Esfuerzo:** Alto
Descripción:	Diseño y elaboración de componentes en el frontend para manejar la vista de la tienda, perfil, cursos, certificados y demás servicios.

Tabla 25

Historia de usuario H18

Historia de Usuario	
Número: 18	**Usuario**: Cliente
Nombre:	Implementación de calendario de eventos.
Iteración: 5	**Prioridad:** Medio
Riesgo: Bajo	**Esfuerzo:** Alto
Historia de Usuario	
Descripción:	Como cliente necesito tener acceso a un calendario que gestione de los eventos de clases y tareas asignadas. Como administrador necesito tener acceso a un calendario en el cual gestionar las videoconferencias. Como docente necesito tener acceso a un calendario en el cual se visualicen los eventos y fechas de las clases que debo dictar.

Tabla 26

Historia de usuario H19

Historia de Usuario	
Número: 19	**Usuario**: Cliente
Nombre:	Diseño y desarrollo del dashborard
Iteración: 5	**Prioridad:** Alto
Riesgo: Medio	**Esfuerzo:** Bajo
Descripción:	Como administrador necesito tener acceso a un panel de administración principal o dashborard, que indique información relacionada a los últimos usuarios registrados, cantidad de ventas por mes, cantidad de usuarios por país y cantidad de usuarios por curso.

2.2 Diseño

Se caracteriza como el proceso de emplear ciertas técnicas y principios con el propósito de definir un proceso o sistema con suficiente detalle para permitir su interpretación y realización.

2.2.1 Diagramas de procesos

Un usuario debe tener la posibilidad de registrarse en el sistema e iniciar sesión siempre y cuando su cuenta no se halle bloqueada y no incurra en superar los 5 intentos de inicio de sesión como se muestra el proceso en la Fig. 11.

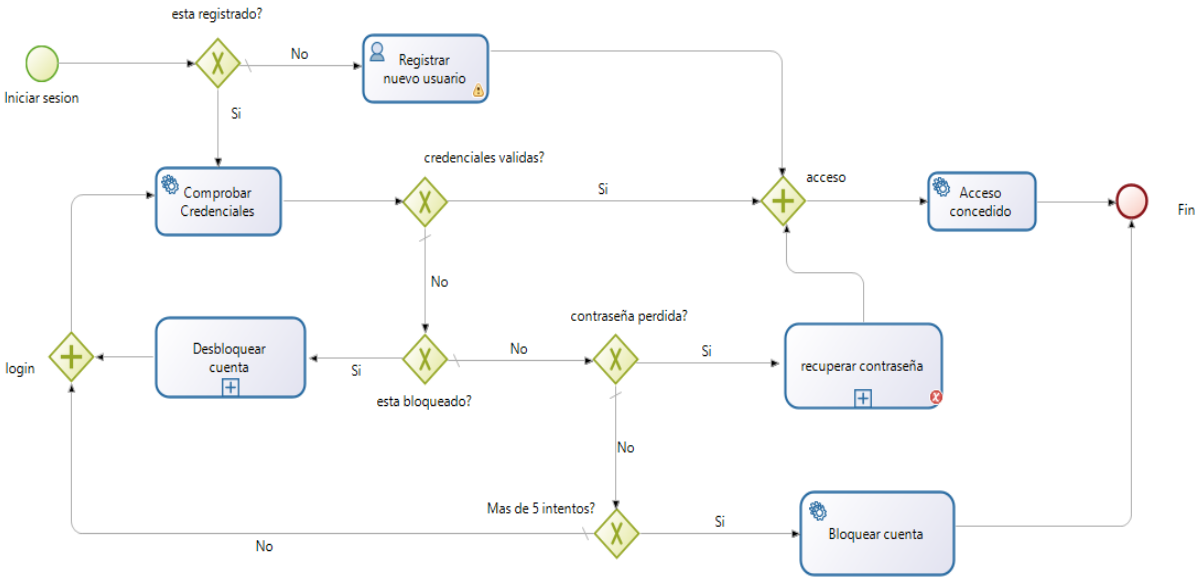


Fig. 11 Diagrama de procesos, Autenticación del sistema.
Fuente: Propia.

El usuario puede recuperar su contraseña siempre y cuando la cuenta no se encuentre bloqueada, si no es el caso, el usuario tiene 2 opciones para recuperarla, solicitar a un administrador una clave temporal de acceso al sistema o llenar un formulario, el mismo solicita el correo electrónico de la cuenta al que se envía una url con un token que valida la identidad del usuario y con un tiempo de expiración de 10 minutos como se explica en el diagrama de la Fig. 12.

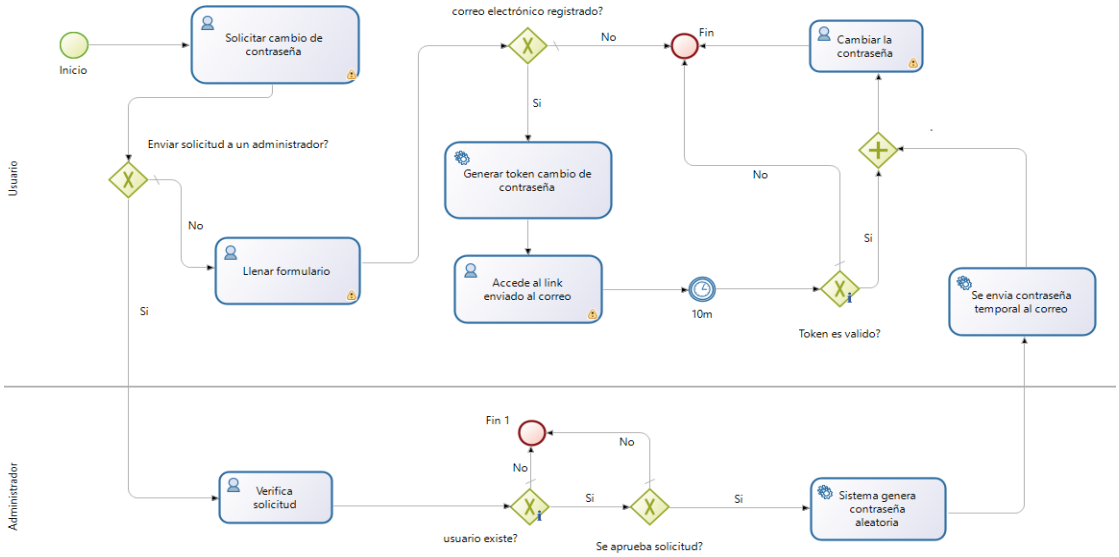


Fig. 12 Diagrama de procesos recuperación de contraseña.
Fuente: Propia.

En caso de que la cuenta ya se halle en proceso de desbloqueo como se indica en la Fig. 13, la única opción que el usuario dispone es la de solicitar un desbloqueo al administrador del sistema el cual revisa los motivos del bloqueo y toma la decisión de desbloquear la cuenta.

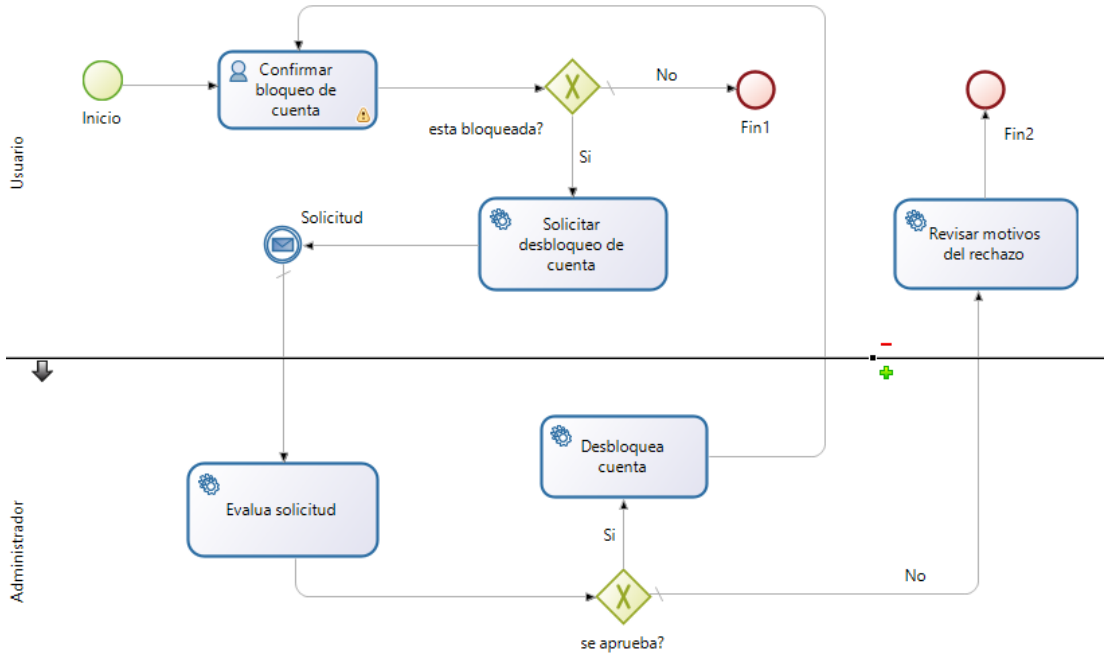


Fig. 13 Diagrama de procesos desbloqueo de cuenta.
Fuente: Propia

En el diagrama de la Fig. 14, se muestra el proceso de creación de cursos, estos solo pueden ser creados por un administrador, el mismo debe insertar la información básica (nombre, imagen de portada, etc.), posteriormente el docente asignado debe gestionar las actividades y la fecha de cada clase.

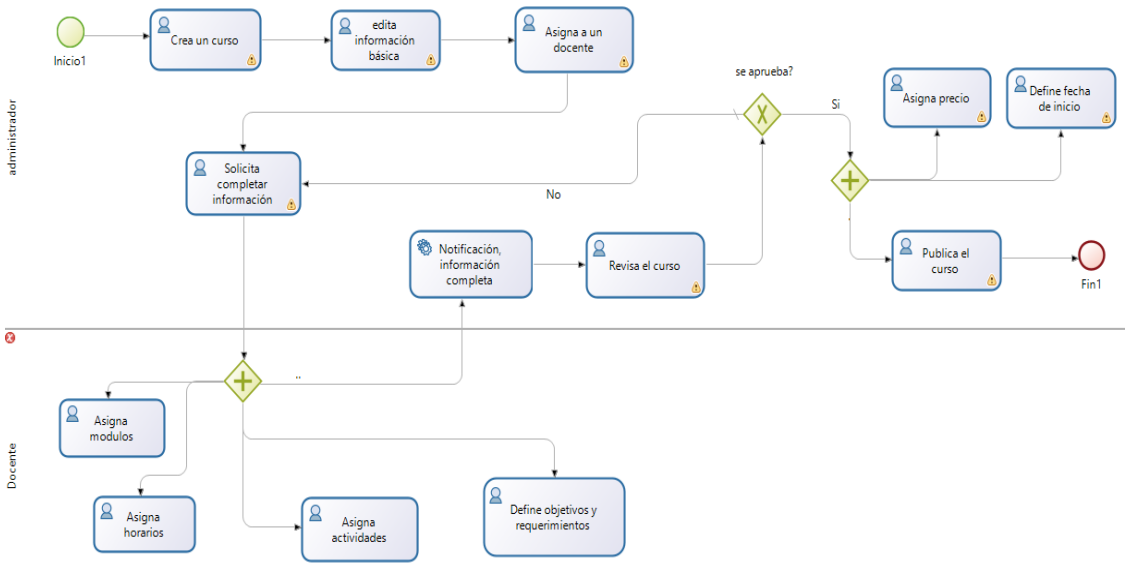


Fig. 14 Diagrama de procesos, creación de un curso.
Fuente: Propia.

El proceso de compra y venta de un curso se puede observar en el diagrama de la Fig. 15. El usuario puede acceder a la información del curso sin necesidad de haber iniciado sesión en el sistema, sin embargo, en caso de que el usuario ya tenga una sesión activa, el cliente tendrá la opción de agregar más productos al carrito o proceder con el proceso de compra el cual es efectuado mediante la plataforma de pagos PayPal. Una vez que el cliente realice el pago será redirigido a la sección de los cursos comprados.

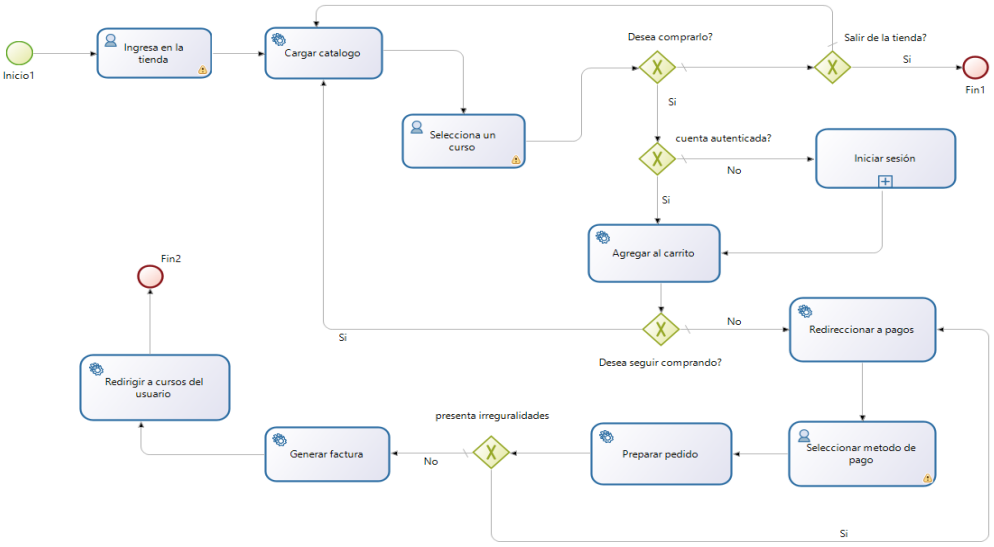


Fig. 15 Diagrama de procesos, compra de un curso
Fuente: Propia.

2.2.2 Casos de uso

El diagrama de casos de uso exhibe las acciones que ejecuta un actor determinado dentro del sistema y el comportamiento de este para establecer un proceso y definir el alcance del proyecto.

2.2.2.1 Diagrama casos de uso usuario administrador

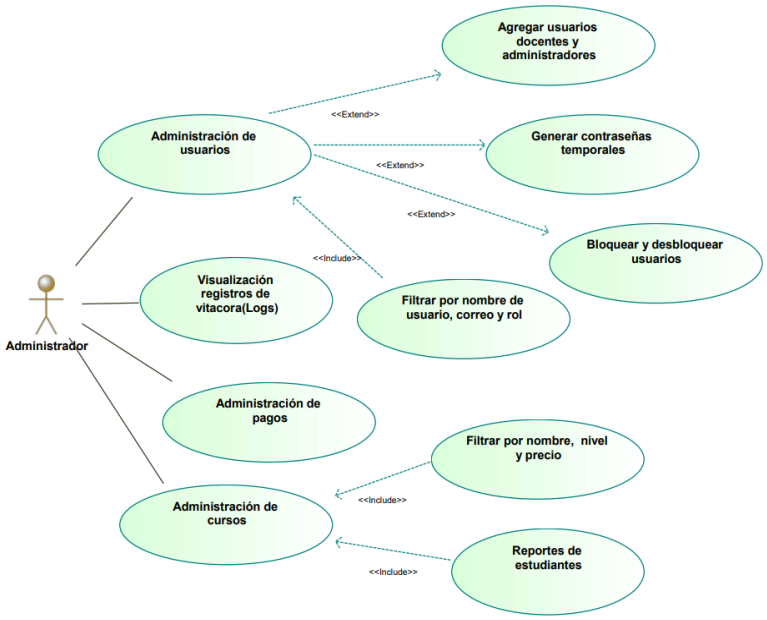


Fig. 16 Diagrama casos de uso - administrador.
Fuente: Propia

2.2.2.2 Diagrama caso de uso administrador y docente

Fig. 17 Diagrama caso de uso administrador – docente
Fuente: Propia

2.2.2.3 Diagrama caso de uso cliente

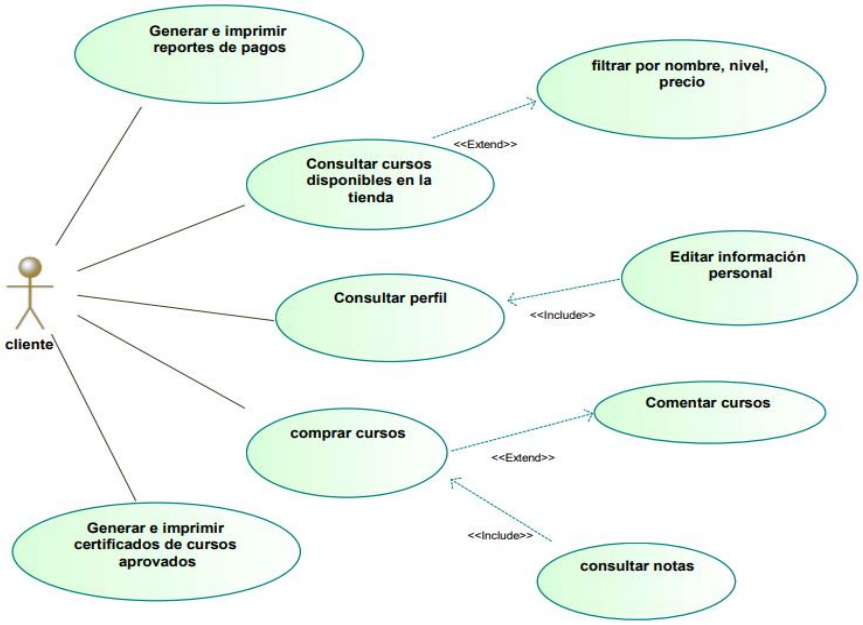


Fig. 18 Diagrama casos de uso – cliente
Fuente: Propia

2.2.2.4 Descripción de casos de uso

Tabla 27

Descripción del caso de uso para el rol de administración.

Caso de uso	**Descripción**
Nombre	Administración
Actores	Administrador
Descripción	Es el encargado de administrar y gestionar todo el sistema de información.
Precondiciones	El usuario administrador solo puede crear cuentas con rol de maestro y administrador. No puede editar los registros de bitácora, solo visualizarlos. El precio de cada curso no puede superar los $200.
Acciones del actor	**Respuestas del sistema**
Bloqueo de usuarios	Solicita confirmación de bloqueo
Genera contraseñas temporales	Genera la contraseña y la envía al correo del cliente, el administrador no visualiza la contraseña
Agrega nuevos usuarios	Muestra el formulario
Visualiza registros logs	-
Administración de pagos	Despliega todas las facturas, permite generar un reporte por factura
Crea y administra cursos	Permite generar un reporte de estudiantes

Tabla 28

Descripción del caso de uso de administración - docente.

Caso de uso	Descripción
Nombre	Gestión de cursos
Actores	Administrador, Docente
Descripción	Describe el proceso de creación y edición de cursos.
Precondiciones	El docente solo puede agendar actividades después de la fecha inicio del curso. No se puede subir recursos con un peso mayor a 30 MB.
Acciones del actor	**Respuestas del sistema**
Administrador crea el curso	Despliega el formulario
Administrador asigna un docente	-
Docente asigna recursos.	Despliega interfaz para seleccionar archivos.
Docente asigna módulos, horarios y eventos relacionados.	Ingresa eventos en el calendario, por cada actividad y enlace a clases registradas.

Tabla 29

Descripción del caso de uso de clientes.

Caso de uso	**Descripción**
Nombre	Actividades del cliente
Actores	Cliente
Descripción	Describe los procesos y actividades desarrolladas por el cliente.
Precondiciones	El cliente solo puede generar certificados cuando el administrador cierre el curso.
Acciones del actor	**Respuestas del sistema**
Generar reportes de pagos	Despliega el reporte en PDF.
Comprar cursos	Desplegar vista de cursos comprados, una vez se realice la compra.
Consultar perfil.	Despliega formulario.

2.2.3 Diagramas base de datos.

Aprovechando la arquitectura de microservicios, se decide usar 2 bases de datos con diferentes propósitos de acuerdo con el servicio. Una de las bases de datos es construida bajo la tecnología de NoSQL con MongoDB, se pretende aprovechar la velocidad de lectura en grandes volúmenes de datos para elaborar el esquema de autenticación al sistema. El servicio que provee la estructura de los cursos es desarrollado sobre la base de datos SQL con PostgreSQL, ya que en dichos servicios es fundamental evitar la redundancia de datos.

Para la base de datos NoSQL se usó 2 documentos sin relaciones entre sí. La colección de usuarios como se muestra en la Fig. 19 contiene toda la información referente al cliente, como su método de autenticación e información de perfil mientras que la colección mostrada en la Fig. 20 guarda información relacionada a las sesiones activas del usuario.

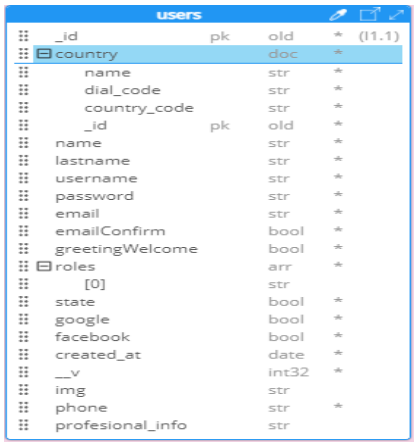


Fig. 19 Colección de usuarios para autenticación del sistema
Fuente: Propia

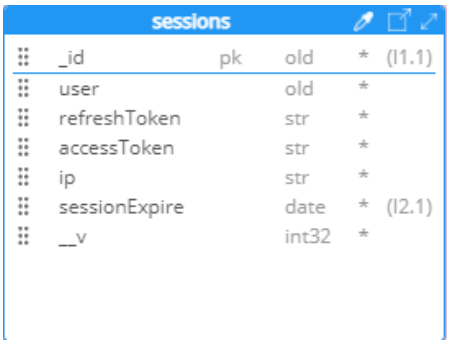


Fig. 20 Colección de sesiones en base de datos de autenticación
Fuente: Propia

En la Fig. 21 se muestra el diagrama entidad relación de la base de datos SQL utilizando la tercera forma normal con el objetivo de evitar la redundancia, ambigüedad y problemas transaccionales en los datos.

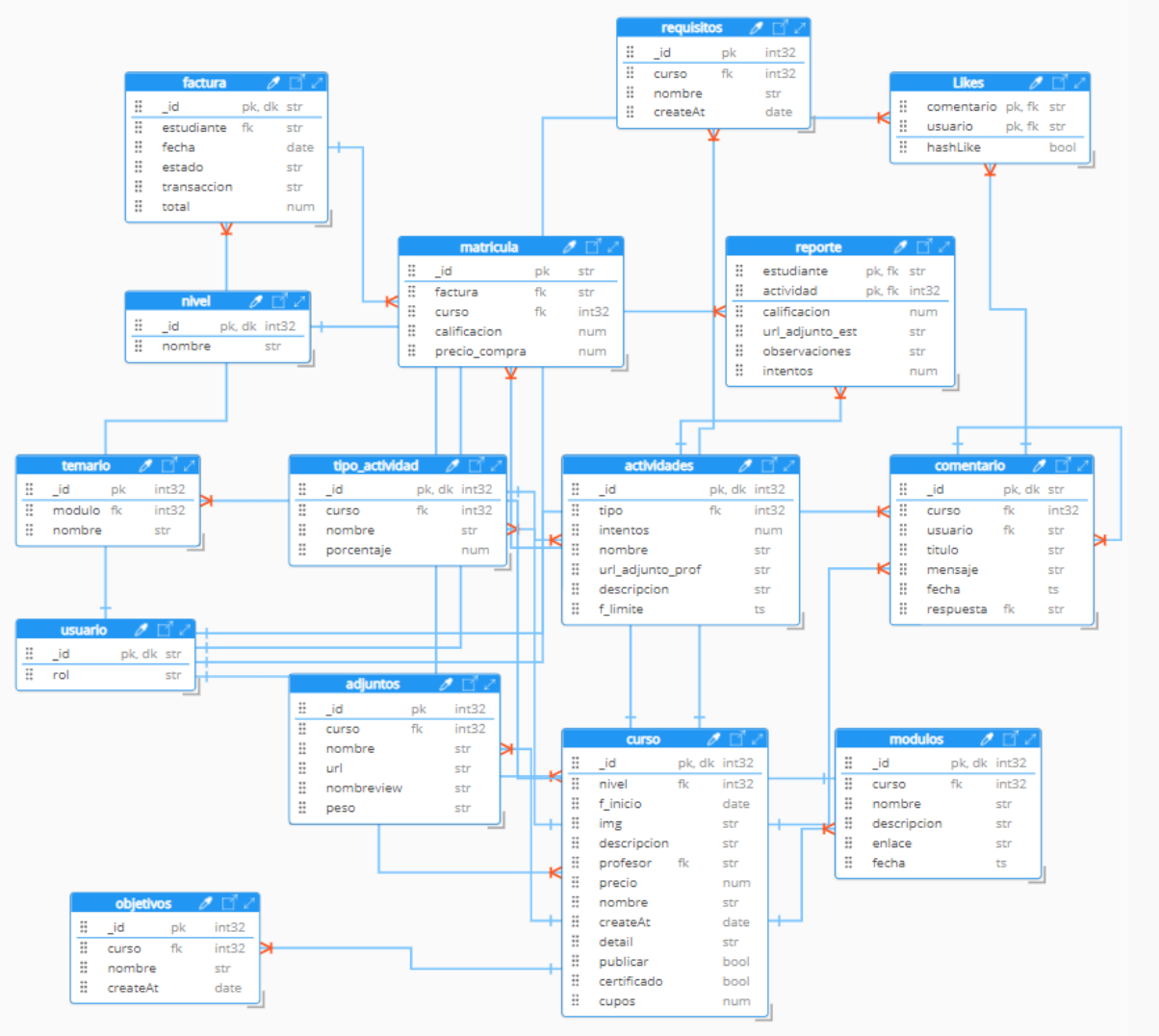


Fig. 21 Esquema E/R base de datos – cursos
Fuente: Propia

2.3 Codificación.

En esta fase se inicia la construcción del módulo web de acuerdo con la planificación de historias de usuario y los diagramas generados en la fase de diseño.

2.3.1 Definición de la estructura del proyecto

En todos los servicios a desarrollar, se incluirá la misma configuración, utilizando webpack para la construcción de módulos web de acuerdo con el entorno seleccionado y la guía de estilos que se indica en la Fig. 22.

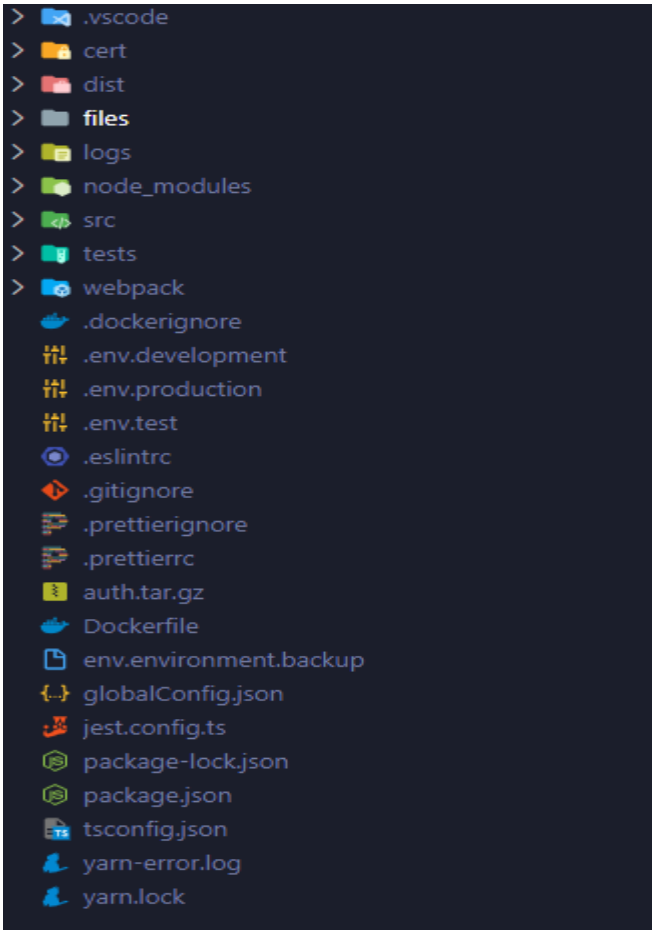


Fig. 22 Estructura del proyecto
Fuente: Propia

Cada microservicio se estructuro pensando en usar JavaScript como lenguaje de programación base, con Nodejs en el backend y ReactJS en el frontend aprovechando todos los beneficios que ofrece un compilador como TypeScript sin importar el ambiente elegido, la configuración de webpack en este punto es diferente para el Backend y el Frontend, esto debido a que el archivo de compilación resultante debe ser ejecutado en entornos diferentes, es decir, el Backend comúnmente es ejecutado como servicio en un servidor, mientras que código compilado para el Frontend es estrictamente

importante desarrollarlo para ser ejecutado de manera interpretada en un navegador web. Sin importar el ambiente elegido cada disposición de docker consta de 4 archivos de configuración de webpack, uno para cada entorno de desarrollo, el primero de ellos y más importante, es el archivo denominado "**webpack.common**", como su nombre lo indica, este archivo de configuración contiene todas las configuraciones comunes entre los ambientes disponibles, las más importantes se listan a continuación.

- **Entry:** Define cual es el archivo de entrada o comúnmente denominado índex.
- **Module:** Contiene un conjunto de reglas y librerías externas elegidas para compilar e interpretar el código fuente.
- **Resolve:** Objeto de configuraciones que indican como resolver los módulos.
- **Output:** Se especifica ruta y nombre del archivo de generado después de compilar el código fuente.
- **Stats:** Permite controlar cual es la información desplegada en pantalla cuando existen errores o advertencias.

Webpack también permite realizar una configuración de alias de ruta, este concepto es muy importante cuando se desarrolla código con módulos separados, las importaciones de ruta resultan ser muy largas y confusas para aquellos desarrolladores que se incorporan recientemente y tienen que llevar un seguimiento del funcionamiento del código. Su implementación es muy sencilla, en el archivo de webpack se define un alias (nombre alternativo), y a ese alias se le asigna la ruta de origen donde se encuentran los módulos, una vez la configuración esta lista se podrá acceder a cada módulo como se indica a continuación.

- import Foo from '../../../../../../../../../../../../../foo';
- import Foo from '@foo';

Para cada entorno es necesario definir un archivo de configuración diferente, una de las razones principales radica en que cada ambiente posee variables de entorno distintas y el archivo de compilación debe estar optimizado, minimizado y sin archivos .map del código fuente original para evitar la depuración del código resultante, etc. Para definir las variables de entorno es necesario contar con un módulo externo que ayude a simplificar la tarea de incorporar dichas variables, dado

que estas se asignan de manera diferente dependiendo del sistema operativo anfitrión en el que se desarrolló del código.

Para agregar cualquier modulo externo que no se haya construido como un módulo de webpack debe ser interpretado por una función especial "**DefinePlugin**" la cual recibe un objeto con la configuración completa de cada librería como se observa en la Fig.23 . El ambiente de producción no requiere muchas más configuraciones, sin embargo, en el ambiente de desarrollo es importante definir de qué manera se mapearán las herramientas de desarrollo o algún plugin que permita el resaltado de errores de sintaxis, formato, etc.

Fig. 23 Configuración de entornos webpack
Fuente: Propia

2.3.1.1 Configuraciones en el Frontend

En la Fig. 24 se muestra el proceso de configuración de webpack diseñado para aplicarse en el desarrollo frontend. Las configuraciones para el Frontend no distan en gran medida de aquellas configuraciones adaptadas para trabajar en un ambiente de Backend, uno de los aspectos más importantes a tener en cuenta son los módulos de resolución de compilación, ReactJS, trabaja con un marco de trabajo propio de su arquitectura denominado JSX, cualquier archivo con esta extensión podrá ser capaz de interpretar código HTML y transpilarlo a XML dentro de un archivo de JavaScript, se creó principalmente para adaptar un árbol de eventos virtual al DOM original, de esta manera puede mutar solo los nodos que hayan sido afectados sin la necesidad de redibujar el componente entero, este concepto cambia todo el paradigma de la programación Frontend, y mejora su eficiencia evidentemente.

ReactJS recomienda usar un loader para los módulos, diferente al acostumbrado a usar en el Backend "ts-loader", en el frontend es importante usar la librería de babel, sin importar si se desea desarrollar un código en TypeScript o JavaScript vanilla. Es evidente que el target es diferente, ya que este código necesita ser interpretado por un navegador y no compilarlo, además se debe agregar plugins externos que permitan desarrollar código CSS, o adaptado a cualquier preprocesador de CSS y HTML.

```
module.exports = {
    entry: path.resolve(__dirname, '..', './src/index.tsx'),
    target: 'web',
    module: {
        rules: [
            {
                test: /\.(ts|js)x?$/,
                exclude: /node_modules/,
                use: [
                    {
                        loader: 'babel-loader',
                    },
                ],
            },
            {
                test: /\.(png|jpg|jpeg|ico|gif)$/,
                type: 'asset/resource',
            },
            {
                test: /\.(woff(2)?|eot|ttf|otf|svg)$/,
                type: 'asset/inline',
            },
        ],
    },
    output: {
        path: path.resolve(__dirname, '..', './build'),
        filename: 'bundle.js',
    },
    plugins: [
        new HtmlWebpackPlugin({
            template: path.resolve(__dirname, '..', './public/index.html'),
        }),
        new NodePolyfillWebpackPlugin(),
        new CopyPlugin({
            patterns: [{ from: 'cert', to: '.' }],
        }),
    ],
```

Fig. 24 Configuración webpack Client
Fuente: Propia

2.3.1.2 configuraciones en el Backend

En la Fig. 25 se muestra el proceso de configuración de webpack diseñado para una utilización en el backend de Nodejs. El backend usa un módulo loader diferente para la configuración de TypeScript, no se usa babel y se necesita un código transpilado a JavaScript para ejecutar en el entorno de desarrollo de Nodejs, este entorno de trabajo tiene muchas ventajas, una de ellas es la posibilidad de programar códigos asíncronos, es decir, que se ejecuten concurrentemente a una acción o función mediante la utilización de callbacks o el uso de promesas en un código de JavaScript más moderno, otra de sus grandes virtudes es la posibilidad de incorporar un gran catálogo de funciones prediseñadas y librerías externas, sin embargo estas son muy pesadas, debido a esto, es necesario que el código externo al código desarrollado no sea agregado en el archivo de compilación, de lo contrario la aplicación pesaría mucho más de lo que en realidad debería, para solucionar este problema es necesario que se agregue una configuración adicional utilizando el plugin de nodos externos, esta configuración se encargará de restringir la compilación del código de librerías utilizando las bibliotecas descargadas en la carpeta node_modules ubicado en la ruta raíz del proyecto.

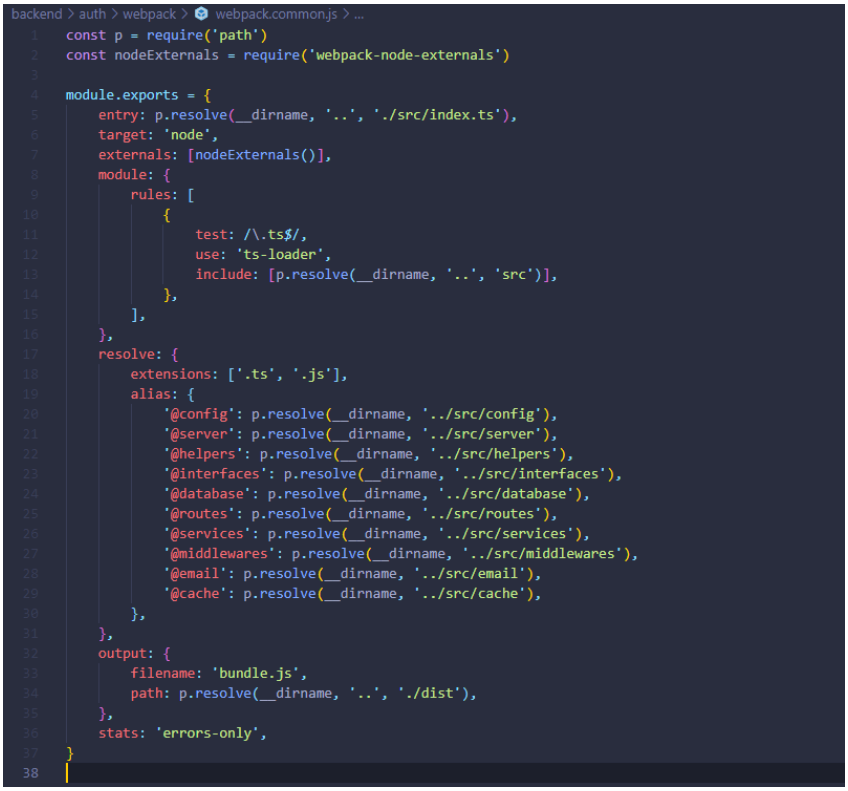


Fig. 25 Configuración de webpack, Backend
Fuente: Propia

Se definieron 3 ambientes de codificación y se establecieron archivos de variables de entorno para cada uno de ellos. Además, se incluyeron librerías para dar formato y evaluación de vulnerabilidades en el código fuente. Para cambiar de ambiente se definieron scripts en el archivo de dependencias "**package.json**" como se indica en la Fig. 26.

```
"scripts": {
    "dev": "webpack --config webpack/webpack.config --env env=dev",
    "build": "rm -rf ./dist && webpack --config webpack/webpack.config --env env=prod",
    "audit": "better-npm-audit audit audit",
    "test": "jest --watch --detectOpenHandles",
    "lint": "eslint --fix .",
    "format": "prettier --write ./src",
    "start": "node ./dist/bundle.js"
},
```

Fig. 26 Scripts configuración de entorno
Fuente: Propia

2.3.2 Configuración de entornos

Para ejecutar y desarrollar el proyecto se decidió utilizar Docker en todas las fases de vida del software, esto porque dicha tecnología permite una agrupación y acoplamiento granular desde un contenedor separado del sistema operativo anfitrión, siendo esta una de las principales ventajas, el software no tendrá problemas en ejecutarse si se cambia de entorno o de sistema operativo. En la fase de pruebas se decidió no limitar la CPU o la memoria del contenedor, pero es importante, realizar estos cambios en el entorno de producción.

2.3.2.1 Entorno de desarrollo y pruebas

Docker posee un gran repositorio de imágenes llamado DockerHub, todas las imágenes de Docker ahí alojadas se pueden descargar sin ningún tipo de costo. Para el ambiente de desarrollo es necesario descargar las imágenes oficiales de las bases de datos de mongoDB y PostgreSQL. DockerHub siempre descarga la imagen con la última versión disponible, sin embargo, si se desea descargar una versión específica se debe indicar el tag de la imagen. Los tags de cada imagen se

encuentran listadas y ordenadas en cada repositorio. A continuación, se listan los comandos necesarios para descargar las imágenes.

- docker pull postgresql
- docker pull mongo
- docker pull dpage/pgadmin4

La imagen dpage/pgadmin4, no es estrictamente necesaria. Dependiendo del sistema operativo, el software pgadmin4 (utilizado para gestionar las bases de datos de PostgreSQL gráficamente) puede no encontrarse disponible. Una vez se hallen descargado las imágenes se listan con el comando **docker images** para verificar la descarga.

Ejecutar los contenedores en docker es sumamente sencillo una vez se tenga la imagen preparada, todo se basa en comandos fáciles de utilizar, esta vez se debe ejecutar el comando indicado en la Fig. 27 para establecer una imagen como demonio o servicio en segundo plano.

Fig. 27 Ejecución de una imagen docker, entorno de desarrollo
Fuente: Propia

La forma de ejecutar imágenes en docker es sencilla, sin embargo, es evidente que se transforma en una tarea agotadora ejecutar imágenes por separado cada que se desee continuar con el proyecto, para solucionar esta problemática existe un complemento muy valorado por la comunidad denominado "`docker-compose`", esta herramienta permite la ejecución de múltiples contenedores usando un archivo de manifiestos como se indica en la Fig. 28, este archivo debe contener la configuración de cada uno de los servicios a recrear y es sumamente ideal para entornos

donde ejecutar un gran número de imágenes es necesario. Su instalación se realiza de la manera común al resto del software del sistema.

Fig. 28 Archivo de manifiestos con la configuración de docker-compose en desarrollo
Fuente: Propia

Una vez que se halle configurado el archivo de manifiesto es necesario ejecutar dicho archivo en la terminal del sistema como un demonio o servicio en segundo plano, asignado la bandera "**-d**", el comando completo se muestra a continuación.

- docker-compose -f <nombre_del_archivo_manifiestos> up -d

2.3.2.2 Entorno de producción

Para los entornos de producción la configuración dista mucho de la explicada en el ambiente de desarrollo o pruebas, esta vez es estrictamente necesario construir la imagen del producto resultante de la compilación, para lograr dicho objetivo se configura un archivo llamado Dockerfile, este contiene todas las instrucciones para la construcción de la imagen, también se agrega un archivo extra llamado ".dockerignore" en donde se adiciona todo registro o carpeta que no se desea agregar en el contenedor.

El archivo Dockerfile requiere que se implementen algunas reglas de construcción como se muestra en la Fig. 29, las mismas son listadas y explicadas a continuación.

- **FROM:** Permite especificar una imagen base de Linux desde la cual empezar a construir.
- **RUN:** Instrucciones y comandos que se ejecutaran en la terminal del contenedor una vez este se empiece a construir.
- **COPY/ADD:** Ambos realizan la misma acción, copiar los archivos que deben ser alojados en el contenedor desde la maquina host o anfitrión al sistema.
- **ENV:** Se definen las variables de entorno estrictamente necesarias y básicas en el contenedor, no precisamente se deben incluir las variables del proyecto, ya que estas son cargadas en el archivo de compilación generado por webpack.
- **WORKDIR:** Ruta donde se alojará la aplicación dentro del contenedor.
- **EXPOSE:** Expone los puertos donde el contenedor de docker escuchará conexiones externas.
- **LABEL:** Asigna etiquetas.
- **USER:** Establece un usuario administrador del contenedor, por defecto es root.
- **VOLUME:** Define todos los archivos compartidos entre el host o anfitrión y el contendedor.
- **CMD:** Ejecuta el demonio del servicio dentro del contenedor.

Todas las imágenes construidas deben partir de una imagen base construida en Linux, dockerhub, ofrece un gran catálogo de imágenes oficiales diseñadas para este propósito. La imagen predilecta para usarse en ambientes de producción es aquella construida bajo Linux alpine, esta

versión de Linux posee un peso bastante pequeño, solo 5MB, y contiene estrictamente lo necesario para ejecutar un servicio en Linux, sin embargo cada empresa desarrolladora crea sus propias versiones modificadas asegurándose que se ejecuten con alpine, de esta manera se puede descargar la imagen oficial de un servicio pero con la etiqueta alpine, asegurando que el proyecto será ejecutado en un ambiente diseñado exclusivamente para el fin establecido, sin necesidad de descargar dependencias inútiles.

Fig. 29 Archivo de configuración de Dockerfile
Fuente: Propia

En la Fig. 30 se indica la forma correcta y básica de construir una imagen. Ubicados en la raíz del proyecto, debe encontrarse el archivo Dockerfile.

Fig. 30 Construcción de imagen
Fuente: Propia

De la misma manera que en el ambiente de desarrollo, en ciertas ocasiones resulta ser una mejor opción construir las imágenes definidas en un archivo de manifiestos “docker-compose” para facilitar la interacción y evaluación de estas. Los comandos más utilizados para la iteración de docker-compose se describen en la tabla 31.

Tabla 30

Comandos básicos utilizados en la ejecución de docker-compose

Comando	Descripción
up	Ejecuta los contenedores especificados en el archivo de manifiesto.
down	Destruye los contenedores declarados
stop	Detiene la ejecución del servicio
inspect	Muestra toda la configuración relacionada con los contenedores.
logs	Muestra los archivos de bitácora de todos los contenedores
logs -f servicio	Muestra los archivos de bitácora del servicio especificado
build	Construye las imágenes de los contenedores especificados, siempre y cuando no existan.
up - d	Ejecuta los servicios en segundo plano
ps	Muestra la ejecución, proceso, puertos, etc. de todos los contenedores.
up - - build[a]	Fuerza la construcción de imágenes

[a] Todos los contenedores ejecutados con up - - build, reemplazan la imagen original, dejando imágenes huérfanas en el directorio, es conveniente eliminarlas.

Una vez que se encuentren todos los servicios en ejecución es importante resaltar que estos no deben exponerse al cliente, es decir se debe acceder a ellos a través de un proxy que redirija todo el tráfico de la red, para este propósito se decide usar "**NGINX**" utilizando la configuración mostrada en la Fig.31 para trabajar como un proxy inverso, sin embargo, es importante que este servicio cuente con certificados validos SSL. En el mercado existen varios que permiten una duración considerable, no obstante, existe otra alternativa gratis y muy eficiente, Let's Encrypt, el único inconveniente de esta herramienta consiste en la duración de los certificados, estos deben ser renovados cada 3 meses.

En caso de que se esté desplegando la aplicación en el ambiente de desarrollo, los certificados no son generados por let's encrypt, ya que estos usan una configuración de DNS válida, si es el caso, se puede generar certificados SSL para la red de localhost usando la herramienta mkcert.

```
server {
    listen 80;
    listen 443 ssl;
    ssl_certificate   /var/cert/cert.pem;
    ssl_certificate_key /var/cert/key.pem;


    server_name app1.breinertech www.app1.breinertech;
    proxy_set_header X-Forwarded-For $remote_addr;
    proxy_set_header Host $host;

    client_max_body_size 50M;

    location / {
        proxy_pass http://ierec-app:4003;
    }

    location /auth_svc/ {
        proxy_pass https://auth_svc:4000/;
    }

    location /course_svc/ {
        proxy_pass https://course-svc:4001/;
    }
```

Fig. 31 *Configuración del proxy nginx*
Fuente: Propia

Let's encrypt provee una herramienta CLI (command line interface) para facilitar la configuración y creación de los certificados llamada "`certbot`". Es importante deshabilitar el servicio de nginx antes de crear los certificados, esto se logra fácilmente de la siguiente manera.

- systemctl stop docker.service
- docker stop nginx. (En caso de usar nginx en docker).

Certbot permite la instalación de certificados de diferente manera, sin embargo, se decidió usar la configuración estándar para este propósito, la misma se indica en la Fig. 32. Para comprender de mejor manera cada instrucción, se describe la función de cada opción en la tabla 31.

```
certbot certonly \
    -d midominio.com \
    --noninteractive \
    --standalone \
    --agree-tos \
    --register-unsafely-without-email
```

Fig. 32 Configuración de certbot
Fuente: Propia

Tabla 31

Descripción de configuraciones de certbot

Comando	Descripción
standalone	crea un servidor web interno para validar el dominio.
-d [a]	Especifica el dominio.
noninteractive	Se evita que la instrucción cree diálogos de confirmación.
register-unsafely-without-email	Evita que se envíe un correo cuando el certificado ha vencido.

[a] Se puede especificar más de un dominio, separando cada dominio con comas.

2.3.3 Desarrollo del Backend

Para desarrollar una aplicación de backend en nodejs existe un comando preconfigurado, "`npm init`", este comando crea un script denominado "`package.json`", este contiene toda la configuración básica del proyecto, eso incluye a las librerías externas las cuales son separadas de la lógica del negocio en dependencias del proyecto para producción y dependencias usadas exclusivamente para el desarrollo de la aplicación, además de los scripts de configuración de ambiente y algunas licencias.

Para iniciar una compilación exitosa de TypeScript, se debe agregar el archivo de configuración en la raíz del proyecto con la estructura mostrada en la Fig. 33, para crear dicho archivo se accede a la consola y se digita el comando "`tsc --init`", este manifiesto es creado por lo general en un formato JSON, no obstante, se puede cambiar a YML o JavaScript.

Fig. 33 Configuraciones de TypeScript.
Fuente: Propia

2.3.3.1 Modelos NoSQL

Los modelos de base de datos NoSQL pueden diferir de acuerdo con el tipo de base de datos NoSQL que se esté tratando, pueden ser documentales, clave-valor, grafos, etc. En este caso el motor de la base de datos de mongoDB provee un esquema para trabajar con colección de documentos. Los esquemas se pueden definir directamente en la construcción de la base de datos o a través de cualquier librería ODM(Object Document Mapped) como se muestra en la Fig. 34, lo que significa que dicha librería permitirá definir esquemas fuertemente tipados para cada documento del modelo.

Para la construcción de los modelos NoSQL se utiliza un ODM el cual ayuda a facilitar la creación de la base de datos, cada modelo debe tener un esquema establecido que indique el modo de almacenamiento de la información en cada colección y cada esquema debe estar delimitado por una interfaz que extienda los métodos definidos por el documento del modelo como se muestra en la Fig. 34.

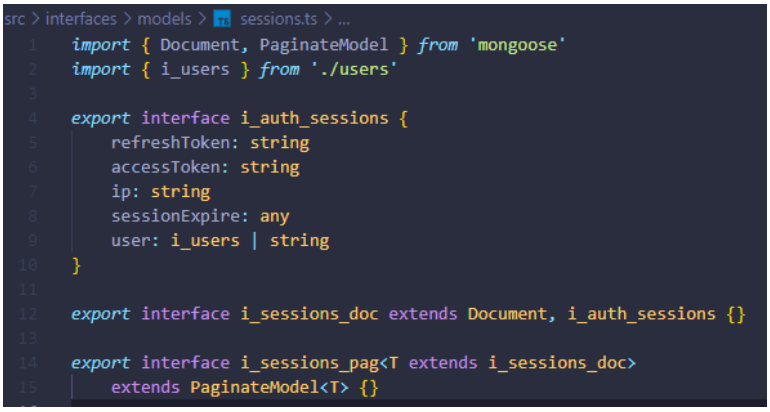


Fig. 34 Interfaz para el esquema de sesiones
Fuente: Propia

Cada modelo puede o no tener funcionalidades extra denominadas plugin o simplemente definir índices o configuraciones opcionales para cada elemento del documento como se aprecia en la Fig. 35 en donde se evalúa que para cada documento de sesión es necesario definir un índice especial de TTL (time to live), es decir, se establece un tiempo de vida para cada documento de sesión y se establece un plugin externo que permita paginar los documentos.

Fig. 35 Modelo para colección de sesiones
Fuente: Propia

2.3.3.2 Creación del servidor, rutas y servicios

El servidor se desarrolló utilizando el framework express en complementación de la librería nativa https de Nodejs. Las clases implementadas para administrar los métodos del servidor y base de datos utilizan un patrón singleton. El patrón Singleton garantiza que solo se puede tener una instancia de la clase en toda la aplicación, implementándose como muestra en la Fig. 36.

Las rutas fueron construidas usando un multiplexor que lleva como prefijo de ruta "/api/v1", haciendo referencia a que es la versión 1 de la API como se muestra en la Fig. 36.

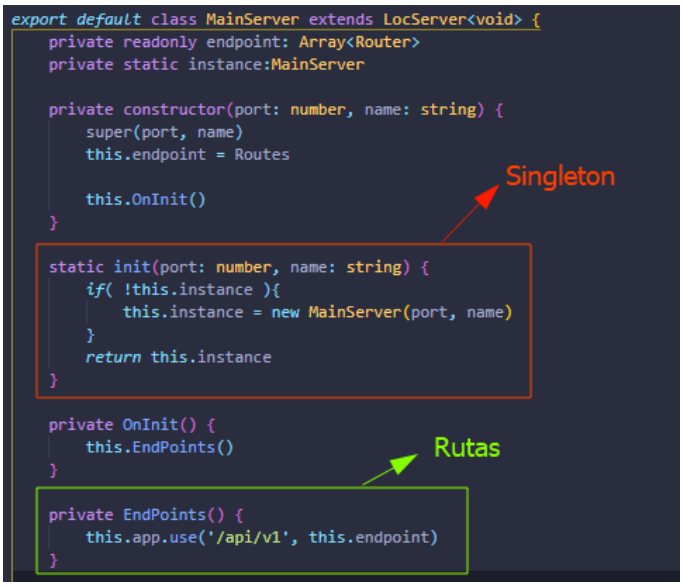


Fig. 36 Configuración del servidor
Fuente: Propia

Las rutas deben definir middlewares(funciones intermedias entre el servicio y la ruta) para la validación de información enviada a través de los diferentes verbos http disponibles, ver tabla 32.

Tabla 32

Verbos HTTP más comunes

Tipo	Descripción
GET[a]	Se utiliza para solicitar recursos y este debe ser su único propósito.
POST	Envía información al servidor que por lo general provoca cambios de estado.
PUT	Se utiliza para actualizar el estado del servidor, no se cachea información.
DELETE	Elimina el recurso especificado.

[a] Por ningún motivo se debe enviar información sensible a través de un método de petición http GET, uno de los motivos más importantes es el cacheo de información, el servidor cachea el tráfico enviado a través de este verbo para agilizar el proceso, el verbo http GET también hace uso de la url del navegador para enviar datos no sensibles como el identificador de un recurso o parámetros necesarios.

2.3.4 Desarrollo del Frontend

Para la construcción del Frontend se utiliza la metodología de desarrollo atomic design, el cual encaja perfectamente con la incorporación de Redux como librería de gestión de estado. Se decide reemplazar la estructura de clases con programación orientada a objetos por el paradigma de programación funcional con la utilización de funciones y hooks.

El paradigma de programación funcional en React hace uso de los hooks, estos son funciones especiales y nativas que permiten realizar acciones específicas, en la tabla 33 se pueden observar los hooks más usados, también se puede escribir un hook personalizado, la única regla de convención para hacerlo es que la función hook personalizada inicie con la palabra "`use`" seguido del nombre.

Tabla 33

Hooks básicos de ReactJS

Hook	Descripción
useState	Permite manejar los estados de un componente.
useEffect	Realiza efectos secundarios que pueden afectar a otros componentes.
useContext	Permite suscribirse al árbol de nodos(componentes) sin introducir el anidamiento.
useRef	Anexa una referencia a un elemento dentro del DOM real.
useMemo	Memoriza información, solo renderiza el estado de dependencias que mutan.

La construcción de código basado en componentes con Redux obliga a utilizar una guía de estilo bien definida, se organiza el código escrito en 3 secciones: Acciones, Reducers y Store. Los reducers utilizan un flujo de trabajo mostrado en la Fig. 37 y deben estar encajados dentro de 4 reglas muy importantes descritas a continuación.

- Deben ser funciones puras.
- No pueden ser funciones asíncronas.
- La función debe retornar el nuevo estado.
- Debe trabajar con 2 argumentos: estado inicial y la acción.

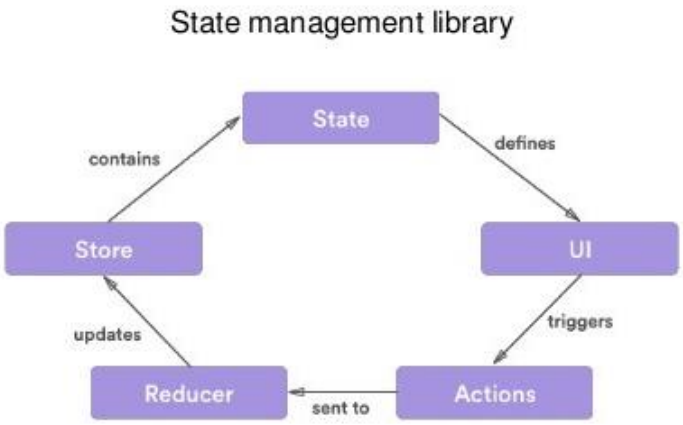


Fig. 37 Flujo de trabajo Redux
Fuente: (*Redux Refresher. I Have Not Used Redux in a While, So… | by Christopher Diep | Medium*, n.d.)

El concepto de store dentro del patrón Redux comprende una base de datos en memoria que guarda toda la información relevante para los componentes. Si se hace una analogía, se debe pensar en ella como una tienda y esta necesita de proveedores de información, estos proveedores son los reducers los cuales realizan actualizaciones de estado de acuerdo con la acción solicitada por la vista, una vez que se actualice el estado se modifica la interfaz gráfica también conocida como UI. Todo este proceso puede apreciarse mejor con la Fig. 37.

Si existen peticiones que solicitan información de una API para actualizar el estado del store se debe hacer uso de acciones asíncronas, es decir, deben esperar la respuesta del servidor para poder actualizar el estado, sin embargo, una de las reglas dicta que las funciones de reducer no pueden ser asíncronas, es por este motivo que las acciones reciben la obligación de adaptar los métodos asíncronos a su funcionamiento.

2.4 Pruebas.

Para la validar el correcto funcionamiento del software se realizaron pruebas de diferente tipo, entre ellas, la realización de pruebas unitarias y pruebas de integración utilizando la librería `Jest` para el entorno de backend y frontend.

Jest necesita un archivo de configuración base ubicado en la raíz del proyecto(ver Fig. 38), el archivo de manifiesto puede ser de tipo JSON, YML o directamente estar configurado en JavaScript, opcionalmente se puede agregar la configuración de alias de ruta declarados en el archivo de configuración de TypeScript a través de la instrucción **“**`moduleNameMapper`**”** como se muestra en la Fig. 38.

```
export default {

  collectCoverage: true,
  coverageDirectory: "coverage",
  coveragePathIgnorePatterns: [
    "/node_modules/"
  ],
  coverageProvider: "v8",
  moduleNameMapper: {
    "&config/(.*)$": "<rootDir>/src/config/$1",
    "&server/(.*)$": "<rootDir>/src/server/$1",
    "&helpers/(.*)$": "<rootDir>/src/helpers/$1",
    "&interfaces/(.*)$": "<rootDir>/src/interfaces/$1",
    "&database/(.*)$": "<rootDir>/src/database/$1",
    "&routes/(.*)$": "<rootDir>/src/routes/$1",
    "&services/(.*)$": "<rootDir>/src/services/$1",
    "&middlewares/(.*)$": "<rootDir>/src/middlewares/$1",
    "&email": "<rootDir>/src/email/index.ts",
    "&cache/(.*)$": "<rootDir>/src/cache/$1"
  },
  preset: 'ts-jest',
  setupFiles: [
    "<rootDir>/tests/setupTest.ts"
  ],
  testEnvironment: "jest-environment-node"
}
```

Fig. 38 Archivo de configuración Jest
Fuente: Propia

2.4.1 Pruebas unitarias

La realización de pruebas unitarias separa el código en fragmentos y aseguran que cada porción de código funciona correctamente. Son tan importantes como el código desarrollado. A medida que el código avanza el desarrollador modifica componentes o funciones que pueden afectar o no el comportamiento de otras funciones y métodos, si no se ha programado este tipo pruebas puede que el desarrollador no se enteré de la mutación en el comportamiento de una característica hasta que ya es demasiado tarde.

Las pruebas realizadas en Jest hacen uso de 2 secciones importantes como se muestra en la Fig. 39. El Segmento denominado **“Describe”** engloba en un conjunto todas las pruebas y el segmento **“it” o “test”**, esta función es la encargada de realizar las pruebas unitarias. También se incluyen funciones complementarias como “**beforeAll**” que permite declarar y asignar el estado de variables utilizadas en los segmentos **“test”**.

En muchas ocasiones es necesario utilizar funciones **"mock"**, son funciones falsas que simulan o sustituyen el comportamiento de las funciones reales para probar un determinado requerimiento. El propósito de esto radica en excluir la utilización de librerías reales por su peso el cual conlleva a que las pruebas se dificulten o que se tomen tiempos demasiado largos para ejecutarlas. Al usar funciones mock se rompen las dependencias y se pueden ejecutar las pruebas de manera independiente. Uno de los casos más usuales de este tipo de pruebas en el backend consiste en la simulación de las interfaces **"Request"** y **"Response"** en las funciones que especifican un servicio como se muestra en la Fig. 39, se utilizan mokcs para realizar las pruebas de validez del token.

```
describe('Test for JWT', () => {

    let token:string;
    let mockReq: Partial<Request>;
    let mockRes: Partial<Response>;
    let nextFunction: NextFunction;

    beforeAll(() => {
        token = generateJWT('access', user_payload);

        mockReq = httpMocks.createRequest();
        mockRes = httpMocks.createResponse();
        nextFunction = jest.fn();
    })

    it('Jsonwebtoken should be generate', () => {
        expect(token).toBeDefined();
    });

    it('JsonWebtoken payload is generated', () => {
        const payload = decode(token) as i_jwt_payload;
        delete payload.iat;
        delete payload.exp;

        expect(payload).toEqual(user_payload);
    })
```

Fig. 39 Funciones Mocks en pruebas del token
Fuente: Propia

Para realizar las pruebas unitarias en los componentes del Frontend de ReactJS es necesario incluir una librería externa dividida en 3 secciones.

- enzyme
- enzyme-adapter-react-<versión_react>
- enzyme-to-json

Jest proporciona casi todas las configuraciones necesarias para realizar pruebas, sin embargo, no se pueden probar hooks personalizados, ya que se viola una de las reglas descritas para los hooks. *"Los hooks solo pueden ser llamados dentro del cuerpo de un componente".* Crear un componente para probar un hook es una tarea agotadora. Para dar solución a este tipo de problemas se hace uso de la librería `@testing-library/react-hooks`.

2.4.2 Pruebas de integración

Las pruebas de integración revisan la iteración y comunicación entre componentes y diferentes partes del sistema, de esta forma el desarrollador se asegura que las funciones se complementan correctamente durante todo el ciclo de vida del software.

Las pruebas de integración realizadas en el backend corresponden a la composición de los módulos de rutas para cada solicitud de la API como se muestra en la Fig. 40. Para la realización de este tipo de pruebas HTTP se necesita incluir el módulo `supertest`.

```
it('Should login', async () => {
    const response = await app().post('/api/v1/auth/login').send({
        "email": user.email,
        "password": user.password
    });

    token = response.body.accessToken;
    refreshToken = response.body.refreshToken;

    expect(response.status).toBe(200);
    expect(response.body.accessToken).toBeDefined();
    expect(response.body.refreshToken).toBeDefined();
});

it('Should getting all refresh tokens', async () => {
    const response = await app().get('/api/v1/auth/refresh')
        .set('x-token-superadmin', token);


    expect(response.status).toBe(200);
    expect(response.body.tokens).toBeDefined();
});
```

Fig. 40 Pruebas de integración backend
Fuente: Propia

CAPITULO 3

RESULTADOS

Para el propósito de evaluación se emplea la estrategia de usabilidad, la cual hace referencia a la facilidad con la que los usuarios pueden hacer uso de un sistema de manera efectiva y sin complicaciones, siendo esencial en el rol de aceptación y efectividad de un entorno de aprendizaje virtual (EVA). (Hovorushchenko, 2018)

3.1 Metodología

Para la evaluación del producto se hace uso de la estrategia denominada sistema de escala de usabilidad (EUS) o por sus siglas en inglés(System Usability Scale), corresponde a un cuestionario de 10 preguntas basado en el modelo de calidad del producto de la norma ISO/IEC 25010 con las subcaracterísticas de accesibilidad, capacidad con la que el sistema es usado y la adecuación con las necesidades del usuario. Se realizan las preguntas listadas a continuación.

1. ¿Cree que le gustaría utilizar frecuentemente este sitio web?
2. ¿Encontró el sitio web sencillo?
3. ¿Piensa que el sitio web es fácil de usar?
4. ¿Piensa que podría utilizar este sitio web sin el apoyo de un técnico?
5. ¿Encontró que varias de las funciones en el sitio web estaba bien integradas?
6. ¿Piensa que había demasiada consistencia en el sitio web?
7. ¿Imagina que la mayoría de las personas podrían aprender a usar este sitio web muy rápido?
8. ¿Encontró el sitio web muy intuitivo?
9. ¿Se sintió muy confiado (seguro) al utilizar el sitio web?
10. ¿Puede utilizar el sitio web sin tener que aprender algo nuevo?

El sistema de escala de usabilidad utiliza un rango de puntuación de 1 a 5 con la escala de Likert donde 1 significa Total desacuerdo y 5 significa Totalmente de acuerdo.(Abuhlfaia & de Quincey, 2019)

1. Totalmente en desacuerdo
2. En desacuerdo
3. Ni de acuerdo, ni en desacuerdo
4. De acuerdo
5. Totalmente de acuerdo

3.2 Interpretación de resultados

Utilizando un método de investigación cualitativo y cuantitativo se realiza una encuesta a 60 usuarios, estudiantes de ingeniería de software de diferentes niveles para investigar si el sistema es utilizable o no. Todo el proceso se efectúa de forma remota a través de la plataforma educativa Microsoft Teams, obteniendo los resultados indexados en la tabla 34.

Tabla 34

Resultados de la encuesta clasificados por pregunta y rango de Likert

Preguntas	**Rangos**				
	1	**2**	**3**	**4**	**5**
1	1	3	3	21	32
2	1	3	2	23	31
3	1	2	5	24	28
4	0	2	3	22	33
5	0	2	3	21	34

Preguntas	Rangos				
	1	**2**	**3**	**4**	**5**
6	2	3	2	20	33
7	3	2	1	21	33
8	0	2	3	17	38
9	1	2	4	18	35
10	1	1	3	21	34

3.3 Análisis de resultados

Con los resultados obtenidos, a continuación, se detalla cada pregunta de la encuesta de escala de usabilidad (SUS) aplicada a los usuarios.

3.3.1 Pregunta 1

En el análisis realizado a la pregunta 1 indicado estadísticamente en la fig. 41 muestra que la opción más votada corresponde a la opción 5 de la escala de Likert representada como “Totalmente de acuerdo” con 32 votos y un 53% de aceptación, en contraposición de un 2% para la opción “Totalmente en desacuerdo” correspondiente a 1 voto, mientras que la segunda opción elegida por los encuestados es “De acuerdo” con 21 votos equivalente al 35% de aceptación demostrando así que los usuarios encuentran agradable la interfaz y facilidad de uso del sistema.

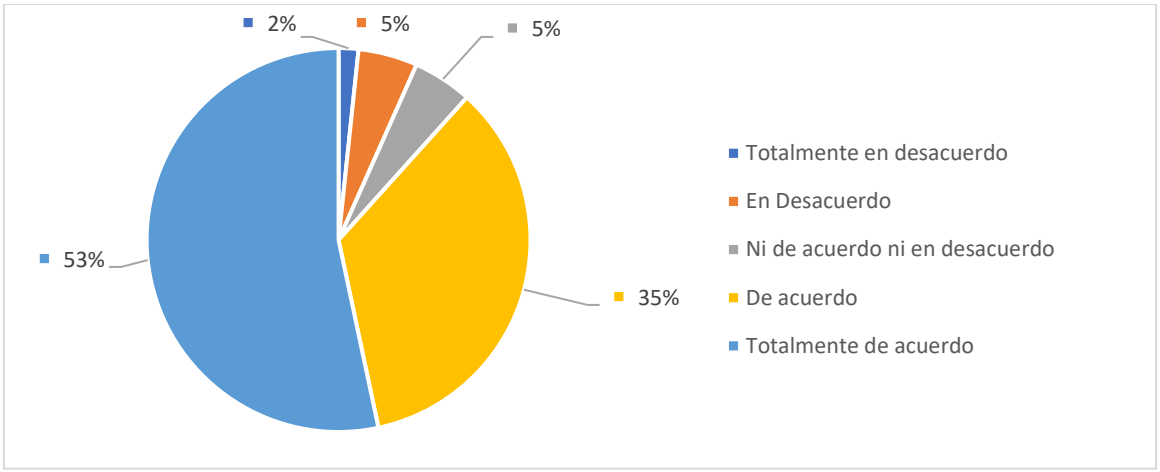


Fig. 41 Resultados de la pregunta 1
Fuente: Estudio de encuesta

3.3.2 Pregunta 2

En el análisis realizado a la pregunta 2 indicado estadísticamente en la fig. 42 muestra que la opción más votada corresponde a la opción 5 de la escala de Likert representada como "Totalmente de acuerdo" con 31 votos y un 52% de aceptación, en contraposición de un 2% para la opción "Totalmente en desacuerdo" correspondiente a 1 voto, mientras que la segunda opción elegida por los encuestados es "De acuerdo" con 23 votos equivalente al 38% de aceptación demostrando que los usuarios no encuentran la interfaz de usuario compleja.

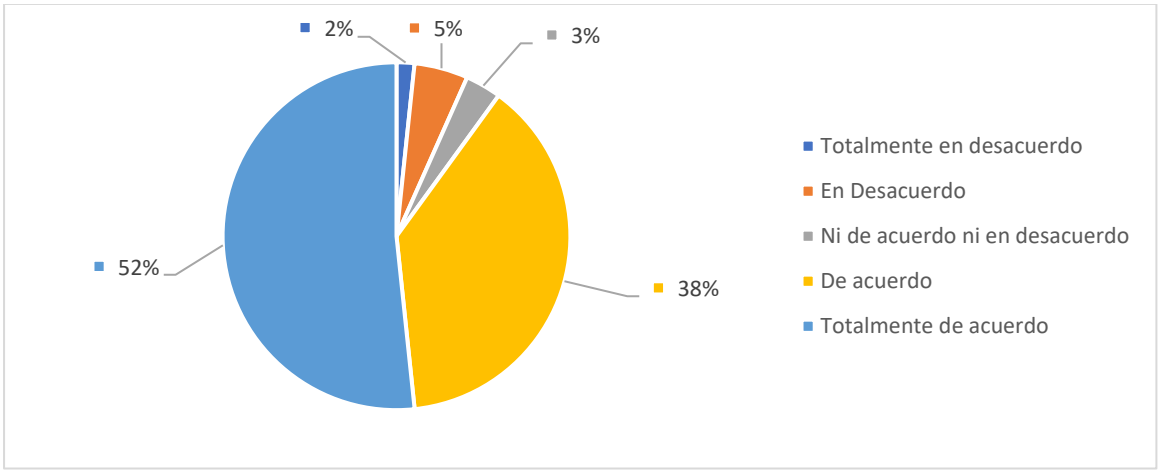


Fig. 42 Resultados de la pregunta 2
Fuente: estudio de encuesta

3.3.3 Pregunta 3

En el análisis realizado a la pregunta 3 indicado estadísticamente en la fig. 43 muestra que la opción más votada corresponde a la opción 5 de la escala de Likert representada como “Totalmente de acuerdo” con 28 votos y un 47% de aceptación, en contraposición de un 2% para la opción “Totalmente en desacuerdo” correspondiente a 1 voto, mientras que la segunda opción elegida por los encuestados es “De acuerdo” con 24 votos equivalente al 40% de aceptación, se demuestra que el sistema es fácil de usar, incluso para aquellos usuarios que no estén familiarizados con plataformas educativas (EVA).

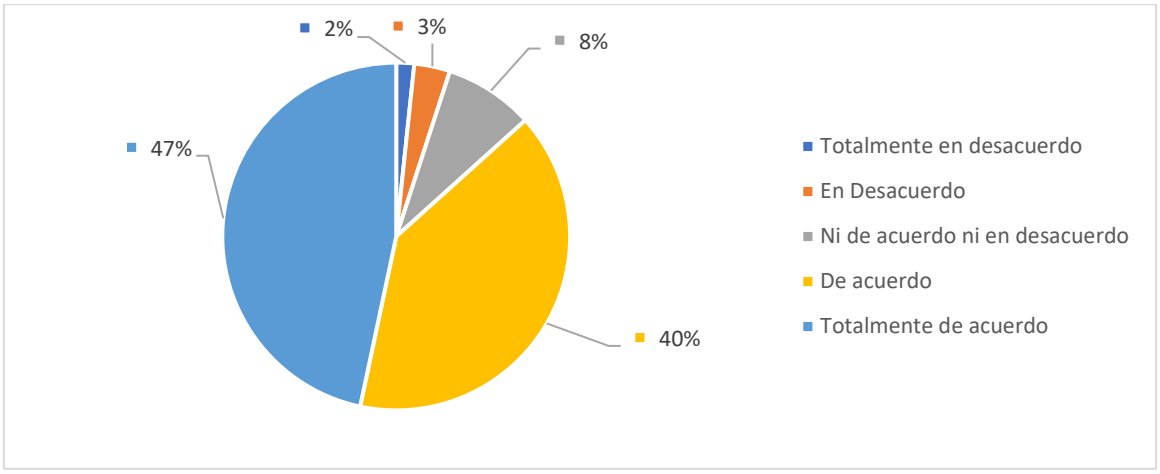


Fig. 43 Resultados de la pregunta 3
Fuente: estudio de encuesta

3.3.4 Pregunta 4

En el análisis realizado a la pregunta 4 indicado estadísticamente en la fig. 44 muestra que la opción más votada corresponde a la opción 5 de la escala de Likert representada como “Totalmente de acuerdo” con 33 votos y un 55% de aceptación, en contraposición de un 0% para la opción “Totalmente en desacuerdo” mientras que la segunda opción elegida por los encuestados es “De acuerdo” con 22 votos equivalente al 37% de aceptación. Los resultados de esta pregunta demuestran que los usuarios necesitan muy poca o ningún tipo de ayuda para usar las opciones disponibles en el sistema.

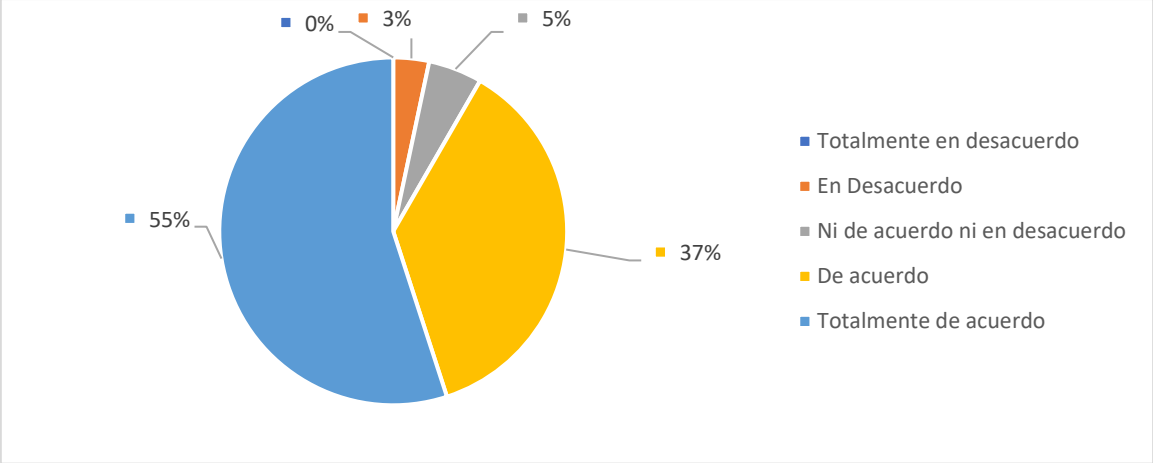


Fig. 44 Resultados de la pregunta 4
Fuente: estudio de encuesta

3.3.5 Pregunta 5

En el análisis realizado a la pregunta 5 indicado estadísticamente en la fig. 45 muestra que la opción más votada corresponde a la opción 5 de la escala de Likert representada como "Totalmente de acuerdo" con 34 votos y un 57% de aceptación, en contraposición de un 0% para la opción "Totalmente en desacuerdo" mientras que la segunda opción elegida por los encuestados es "De acuerdo" con 21 votos equivalente al 35% de aceptación, se demuestra así que las funciones expuestas en el diseño interactivo se integran de manera adecuada.

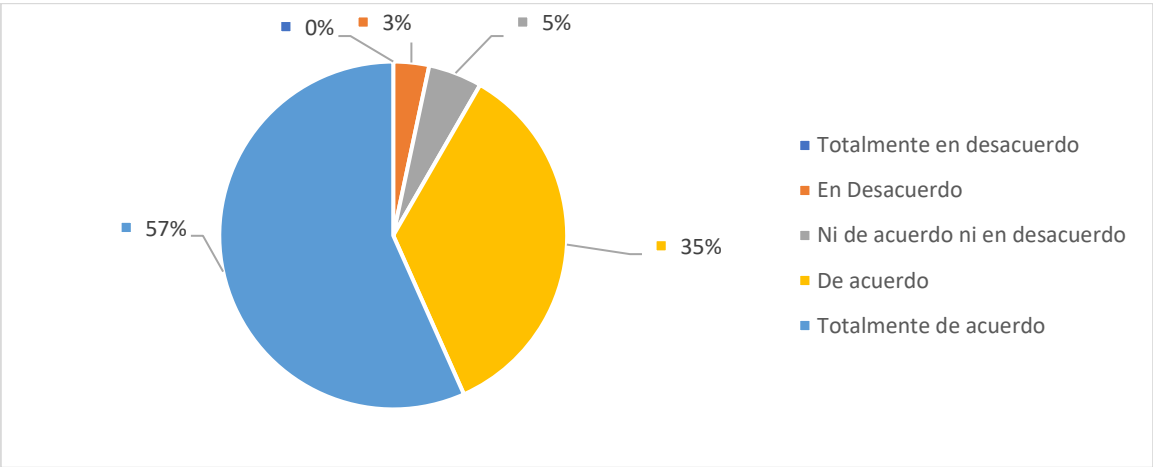


Fig. 45 Resultados de la pregunta 5
Fuente: estudio de la encuesta

3.3.6 Pregunta 6

En el análisis realizado a la pregunta 6 indicado estadísticamente en la fig. 46 muestra que la opción más votada corresponde a la opción 5 de la escala de Likert representada como “Totalmente de acuerdo” con 33 votos y un 55% de aceptación, en contraposición de un 3% equivalente a 2 votos para la opción “Totalmente en desacuerdo” mientras que la segunda opción elegida por los encuestados es “De acuerdo” con 20 votos equivalente al 34% de aceptación, se demuestra así que la consistencia y coherencia de la información del sistema es buena, sin embargo, es necesario mejorar.

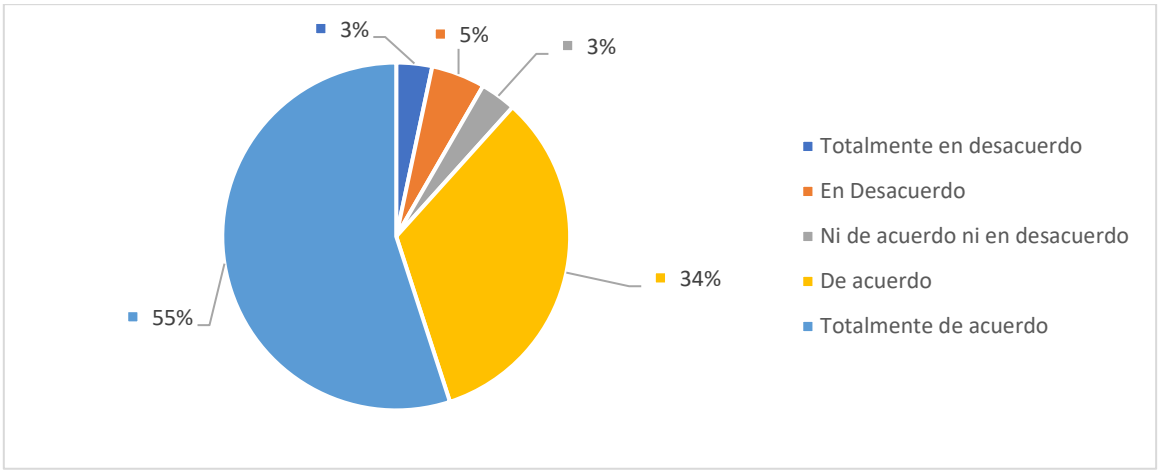


Fig. 46 Resultados de la pregunta 6
Fuente: estudio de la encuesta

3.3.7 Pregunta 7

En el análisis realizado a la pregunta 7 indicado estadísticamente en la fig. 47 muestra que la opción más votada corresponde a la opción 5 de la escala de Likert representada como “Totalmente de acuerdo” con 33 votos y un 55% de aceptación, en contraposición de un 5% equivalente a 3 votos para la opción “Totalmente en desacuerdo” mientras que la segunda opción elegida por los encuestados es “De acuerdo” con 21 votos equivalente al 35% de aceptación. Con la evaluación de los resultados, se considera que el sistema es eficiente e ideal para personas con poca experiencia en la utilización de sitios académicos de tipo EVA.

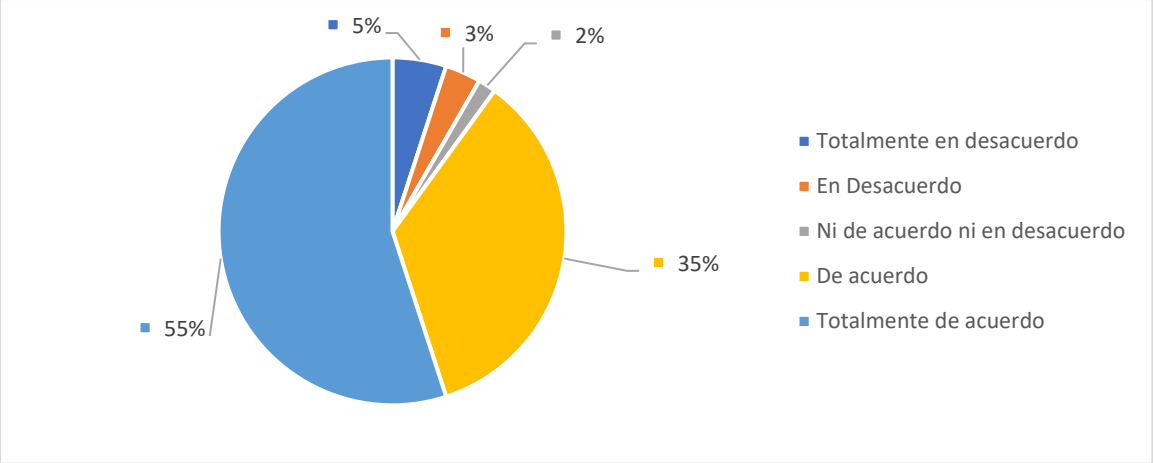


Fig. 47 Resultados de la pregunta 7
Fuente: estudio de la encuesta

3.3.8 Pregunta 8

En el análisis realizado a la pregunta 8 indicado estadísticamente en la fig. 48 muestra que la opción más votada corresponde a la opción 5 de la escala de Likert representada como "Totalmente de acuerdo" con 38 votos y un 34% de aceptación, en contraposición de un 0% para la opción "Totalmente en desacuerdo" mientras que la segunda opción elegida por los encuestados fue "De acuerdo" con 17 votos equivalente al 28% de aceptación. Con la evaluación de los resultados, se considera que un porcentaje alto de usuarios demuestran familiaridad y confianza al usar el sistema, lo que reduce el índice de error que afecte su experiencia al interactuar con el sitio web.

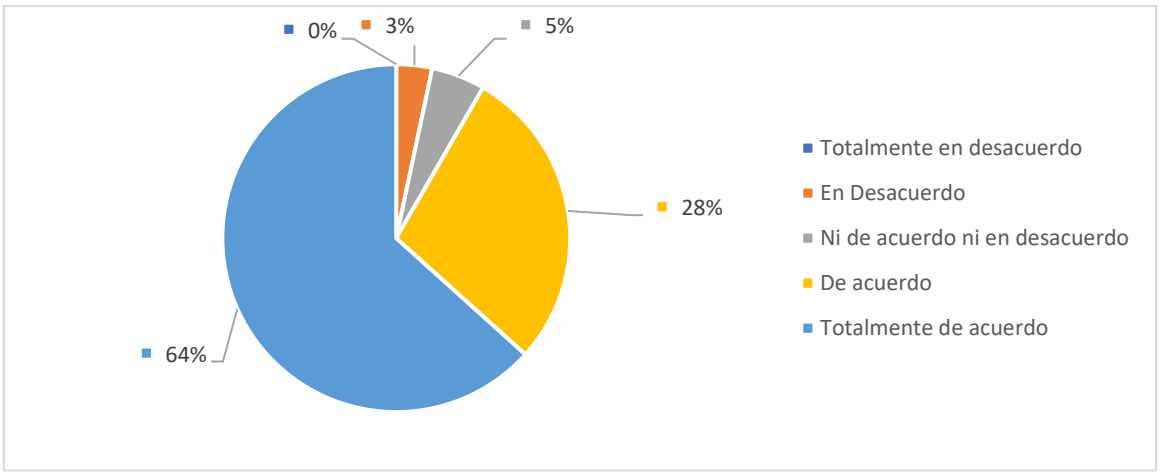


Fig. 48 Resultados de la pregunta 8
Fuente: estudio de la encuesta

3.3.9 Pregunta 9

En el análisis realizado a la pregunta 9 indicado estadísticamente en la fig. 49 muestra que la opción más votada corresponde a la opción 5 de la escala de Likert representada como “Totalmente de acuerdo” con 35 votos y un 58% de aceptación, en contraposición de un 2% para la opción “Totalmente en desacuerdo” correspondiente a 1 voto, mientras que la segunda opción elegida por los encuestados fue “De acuerdo” con 18 votos equivalente al 30% de aceptación. Con la evaluación de los resultados se considera importante realizar mejoras que refuercen la seguridad que el usuario deposita en el sitio web.

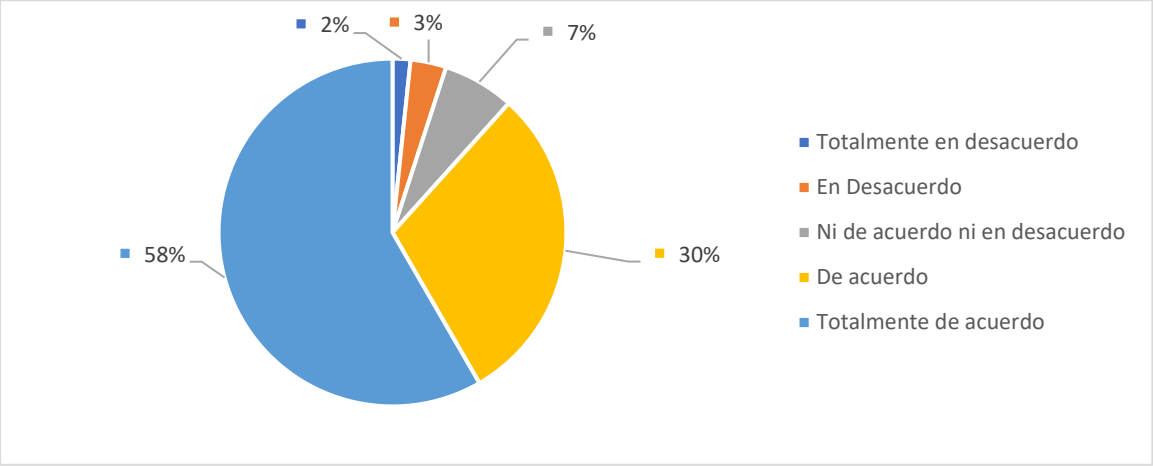


Fig. 49 Resultados de la pregunta 9
Fuente: estudio de la encuesta

3.3.10 Pregunta 10

En el análisis realizado a la pregunta 10 indicado estadísticamente en la fig. 50 muestra que la opción más votada corresponde a la opción 5 de la escala de Likert representada como “Totalmente de acuerdo” con 34 votos y un 56% de aceptación, en contraposición de un 2% para la opción “Totalmente en desacuerdo” correspondiente a 1 voto mientras que la segunda opción elegida por los encuestados es “De acuerdo” con 21 votos equivalente al 35% de aceptación. Con los resultados obtenidos, se considera que el proceso de registro, adquisición y compra de los productos ofertados no representan un nuevo reto debido a la similitud con otros sitios web que comparten el mismo propósito.

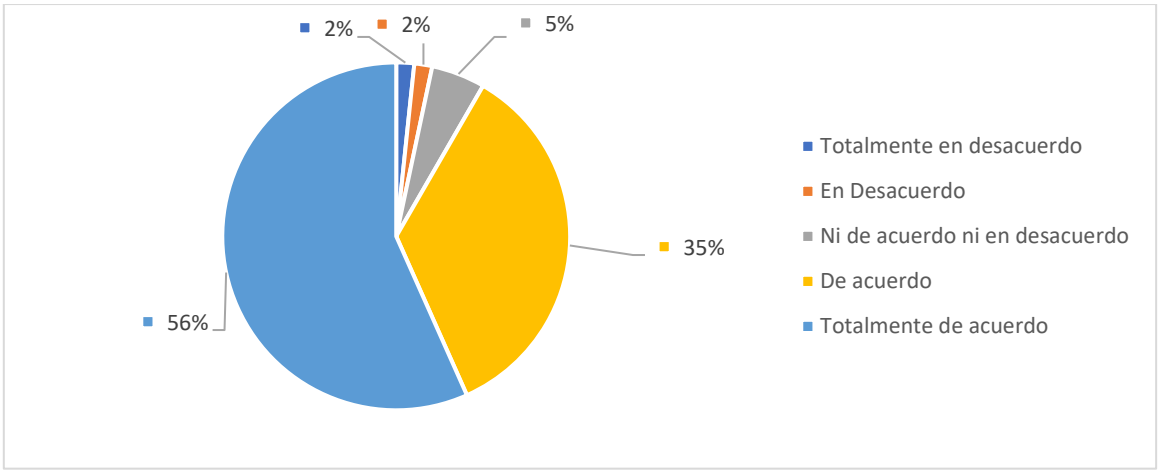


Fig. 50 Resultados de la pregunta 10
Fuente: estudio de la encuesta

En la fig. 51 se pueden apreciar los resultados globales obtenidos en la encuesta de escala de usabilidad(SUS) aplicada al sistema web, se observa de manera general que existe un alto índice de aceptación por parte del usuario.

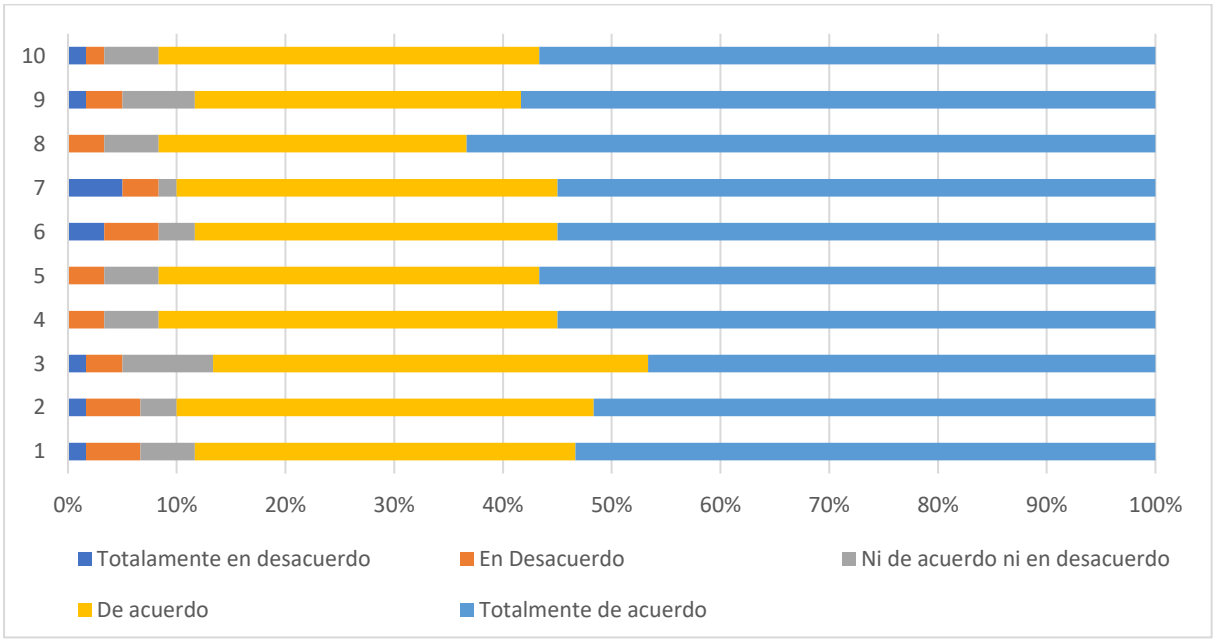


Fig. 51 Resumen estadístico de la encuesta (SUS)
Fuente: estudio de la encuesta

3.4 Cálculo de escala SUS

El siguiente paso en el estudio de la estrategia de(SUS), es calcular el grado de la escala de usabilidad del software, dicha escala permite definir de manera general el nivel de satisfacción que tiene el software con un rango de 0 a 100 como se muestra en la tabla 34.

Tabla 35

Grado de escala SUS

SUS SCORE	Grado	Rating
> 85	A	Excelente
80 - 85	B	Bueno
60 - 70	C	Está bien
40 - 60	D	Horrible
< 40	F	Muy malo

Para realizar dicho análisis se debe establecer el rango de satisfacción de cada usuario encuestado y clasificar sus respuestas en un rango de 0 a 4 donde 0 equivale a totalmente en desacuerdo y 4 a totalmente de acuerdo en la escala de Likert. En la interpretación de resultados se escogió una muestra de 60 usuarios, el resultado de cada encuesta se encuentra indexado en la tabla 36.

Tabla 36

Resultados de la encuesta, clasificación por pregunta y número de encuestado

N° Encuesta	Preguntas									
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
1	1	1	1	1	1	1	1	1	1	1
2	2	3	3	3	4	2	4	4	4	4
3	2	1	2	2	1	1	1	1	1	0
4	4	0	1	1	3	1	4	3	3	4

N° Encuesta	Preguntas									
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
5	3	4	3	4	3	3	3	4	4	3
6	0	1	2	3	3	3	4	4	4	3
7	3	3	4	3	4	4	3	4	4	3
8	3	3	3	4	3	4	4	3	4	4
9	4	4	4	4	4	4	4	4	4	4
10	4	4	4	4	4	4	4	4	4	4
11	4	4	4	4	4	4	4	4	4	4
12	3	3	4	4	4	3	4	4	4	4
13	2	3	2	4	4	0	0	3	4	4
14	4	4	4	4	3	3	0	2	2	4
15	4	4	4	4	4	4	4	4	4	4
16	4	4	3	4	2	3	2	3	4	4
17	3	3	4	4	4	3	3	4	4	3
18	3	4	3	2	3	4	4	4	4	4
19	1	2	2	3	3	4	4	4	3	3
20	4	4	4	4	4	4	4	4	4	4
21	3	3	3	4	3	4	4	4	2	4
22	4	4	4	4	4	2	3	3	3	2
23	4	4	3	4	3	4	3	4	4	3
24	1	2	3	3	4	3	4	4	4	4
25	3	3	3	3	3	3	3	4	4	3
26	3	3	3	4	4	4	3	3	3	3
27	3	3	3	3	3	3	3	3	3	3
28	4	4	3	4	2	3	3	3	3	3
29	4	4	4	4	3	4	4	4	4	4
30	3	3	3	4	4	4	4	4	4	3
31	4	4	4	4	4	4	4	4	4	4
32	4	4	4	4	3	4	4	4	4	4
33	4	4	4	4	4	4	3	3	4	4

34	4	4	4	4	4	4	4	3	3	4
35	4	3	4	4	4	4	4	4	4	4
36	4	4	4	4	4	4	4	4	4	4
37	4	4	4	4	4	4	4	4	4	4
38	4	4	4	4	4	4	4	4	4	4
39	3	4	3	3	4	4	4	4	4	4
40	4	3	3	4	4	3	3	4	4	4
41	3	4	4	3	4	4	4	4	4	4
42	3	3	3	3	3	3	3	4	0	2
43	3	3	3	3	3	0	0	2	2	3
44	3	3	4	4	4	4	4	4	3	3
45	4	3	3	3	4	4	4	4	3	4
46	3	3	4	4	4	4	4	4	4	4
47	3	4	4	3	3	3	4	4	4	3
48	4	4	3	3	3	3	3	4	4	3
49	4	4	4	3	4	4	3	4	3	3
50	4	4	2	3	3	3	3	3	3	2
51	3	3	3	3	4	4	3	3	3	3
52	4	4	4	4	4	3	3	3	3	3
53	4	4	4	4	2	3	3	2	2	3
54	4	4	4	4	4	3	3	3	3	3
55	4	4	0	2	4	4	4	4	4	4
56	3	3	3	3	3	3	3	3	3	3
57	3	3	3	3	3	3	3	3	4	4
58	4	3	3	3	4	4	4	3	3	4
59	4	3	3	3	3	4	4	4	3	4
60	4	4	4	3	4	4	4	4	3	4

Todos los valores correspondientes a las preguntas pares deben reescribirse con la escala de Likert de 4 a 0 donde 4 equivale a totalmente en desacuerdo y 0 a totalmente de acuerdo como se indica a en la tabla 37.

Nomenclatura	Rango
Totalmente en desacuerdo	4
De acuerdo	3
Ni de acuerdo ni en desacuerdo	2
De acuerdo	1
Totalmente de acuerdo	0

Para el cálculo de la escala SUS se deben aplicar la formula indexada en la Fig. 55 la cual explica que el resultado es igual a la sumatoria de todas las preguntas con índice par restándole la constante 5, más la sumatoria de todas las preguntas con índice impar restado de la constante 25, al resultado total se le multiplica la constante 2.5.

$$R = (((\sum_{i=1}^{n} 2i - 1\ ) - 5) + (\ 25 - \sum_{i=1}^{n} 2i\ )) * 2.5$$

Fig. 52 Formula, cálculo de escala SUS
Fuente: (*The System Usability Scale & How It's Used in UX | Adobe XD Ideas*, n.d.-a)

Tomando como referencia los resultados de la encuesta perteneciente al usuario 1 se procederá a calcular el rango de la escala SUS para dichos valores donde la variable X hace referencia a la sumatoria de preguntas con índice impar, mientras que la variable Y hace referencia a preguntas con índice par.

$$X = (1 + 1 + 1 + 1 + 1\ ) \approx 5$$

$$Y = (3 + 3 + 3 + 3 + 3) \approx 15$$

$$X0 = X - 5 \approx 0$$

$$Y0 = Y - 15 \approx 10$$

$$R = (X0 + Y0) * 2.5 \approx 25$$

Los resultados obtenidos para el cálculo de la escala SUS con los valores del usuario 1 corresponden a 25 en la escala, el cual equivale a un grado F como se muestra en la tabla 34. Los cálculos efectuados a cada usuario encuestado se encuentran indexados en la Tabla 38.

Tabla 37

Puntuación en la escala SUS por cada encuesta.

ID	Preguntas										Cálculos				
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**X**	**Y**	**X0**	**Y0**	**SCORE**
1	1	3	1	3	1	3	1	3	1	3	5	15	0	10	25
2	2	1	3	1	4	2	4	0	4	0	17	4	12	21	82.5
3	2	3	2	2	1	3	1	3	1	4	7	15	2	10	30
4	4	4	1	3	3	3	4	1	3	0	15	11	10	14	60
5	3	0	3	0	3	1	3	0	4	1	16	2	11	23	85
6	0	3	2	1	3	1	4	0	4	1	13	6	8	19	67.5
7	3	1	4	1	4	0	3	0	4	1	18	3	13	22	87.5
8	3	1	3	0	3	0	4	1	4	0	17	2	12	23	87.5
9	4	0	4	0	4	0	4	0	4	0	20	0	15	25	100
10	4	0	4	0	4	0	4	0	4	0	20	0	15	25	100
11	4	0	4	0	4	0	4	0	4	0	20	0	15	25	100
12	3	1	4	0	4	1	4	0	4	0	19	2	14	23	92.5
13	2	1	2	0	4	4	0	1	4	0	12	6	7	19	65
14	4	0	4	0	3	1	0	2	2	0	13	3	8	22	75
15	4	0	4	0	4	0	4	0	4	0	20	0	15	25	100
16	4	0	3	0	2	1	2	1	4	0	15	2	10	23	82.5

ID	Preguntas										Cálculos				
	1	2	3	4	5	6	7	8	9	10	X	Y	X0	Y0	SCORE
17	3	1	4	0	4	1	3	0	4	1	18	3	13	22	87.5
18	3	0	3	2	3	0	4	0	4	0	17	2	12	23	87.5
19	1	2	2	1	3	0	4	0	3	1	13	4	8	21	72.5
20	4	0	4	0	4	0	4	0	4	0	20	0	15	25	100
21	3	1	3	0	3	0	4	0	2	0	15	1	10	24	85
22	4	0	4	0	4	2	3	1	3	2	18	5	13	20	82.5
23	4	0	3	0	3	0	3	0	4	1	17	1	12	24	90
24	1	2	3	1	4	1	4	0	4	0	16	4	11	21	80
25	3	1	3	1	3	1	3	0	4	1	16	4	11	21	80
26	3	1	3	0	4	0	3	1	3	1	16	3	11	22	82.5
27	3	1	3	1	3	1	3	1	3	1	15	5	10	20	75
28	4	0	3	0	2	1	3	1	3	1	15	3	10	22	80
29	4	0	4	0	3	0	4	0	4	0	19	0	14	25	97.5
30	3	1	3	0	4	0	4	0	4	1	18	2	13	23	90
31	4	0	4	0	4	0	4	0	4	0	20	0	15	25	100
32	4	0	4	0	3	0	4	0	4	0	19	0	14	25	97.5
33	4	0	4	0	4	0	3	1	4	0	19	1	14	24	95
34	4	0	4	0	4	0	4	1	3	0	19	1	14	24	95
35	4	1	4	0	4	0	4	0	4	0	20	1	15	24	97.5
36	4	0	4	0	4	0	4	0	4	0	20	0	15	25	100
37	4	0	4	0	4	0	4	0	4	0	20	0	15	25	100
38	4	0	4	0	4	0	4	0	4	0	20	0	15	25	100
39	3	0	3	1	4	0	4	0	4	0	18	1	13	24	92.5
40	4	1	3	0	4	1	3	0	4	0	18	2	13	23	90
41	3	0	4	1	4	0	4	0	4	0	19	1	14	24	95
42	3	1	3	1	3	1	3	0	0	2	12	5	7	20	67.5
43	3	1	3	1	3	4	0	2	2	1	11	9	6	16	55
44	3	1	4	0	4	0	4	0	3	1	18	2	13	23	90

ID	Preguntas	Cálculos

	1	2	3	4	5	6	7	8	9	10	X	Y	X0	Y0	SCORE
45	4	1	3	1	4	0	4	0	3	0	18	2	13	23	90
46	3	1	4	0	4	0	4	0	4	0	19	1	14	24	95
47	3	0	4	1	3	1	4	0	4	1	18	3	13	22	87.5
48	4	0	3	1	3	1	3	0	4	1	17	3	12	22	85
49	4	0	4	1	4	0	3	0	3	1	18	2	13	23	90
50	4	0	2	1	3	1	3	1	3	2	15	5	10	20	75
51	3	1	3	1	4	0	3	1	3	1	16	4	11	21	80
52	4	0	4	0	4	1	3	1	3	1	18	3	13	22	87.5
53	4	0	4	0	2	1	3	2	2	1	15	4	10	21	77.5
54	4	0	4	0	4	1	3	1	3	1	18	3	13	22	87.5
55	4	0	0	2	4	0	4	0	4	0	16	2	11	23	85
56	3	1	3	1	3	1	3	1	3	1	15	5	10	20	75
57	3	1	3	1	3	1	3	1	4	0	16	4	11	21	80
58	4	1	3	1	4	0	4	1	3	0	18	3	13	22	87.5
59	4	1	3	1	3	0	4	0	3	0	17	2	12	23	87.5
60	4	0	4	1	4	0	4	0	3	0	19	1	14	24	95

Una vez se hallan calculado el grado en la escala SUS para cada usuario encuestado(ver la tabla 38), el siguiente paso consiste en calcular el promedio de escala para todos los valores resultantes de la muestra elegida, en este caso la muestra corresponde a 60.

$$SUS = \frac{5070}{60} \approx 84.5$$

Finalmente, después de haber calculado el promedio del total de grados de calificación se puede determinar una evaluación general del sistema. Los cálculos realizados arrojaron una calificación de 84.5 equivalente a un grado B como Bueno y aceptable, sin embargo, se aproxima a una calificación de excelente como se puede observar en la escala de la Fig. 53, esto significa que se

tiene un rendimiento bueno y superior a la media establecida, sin embargo, el modelo interactivo propuesto puede seguir mejorando.

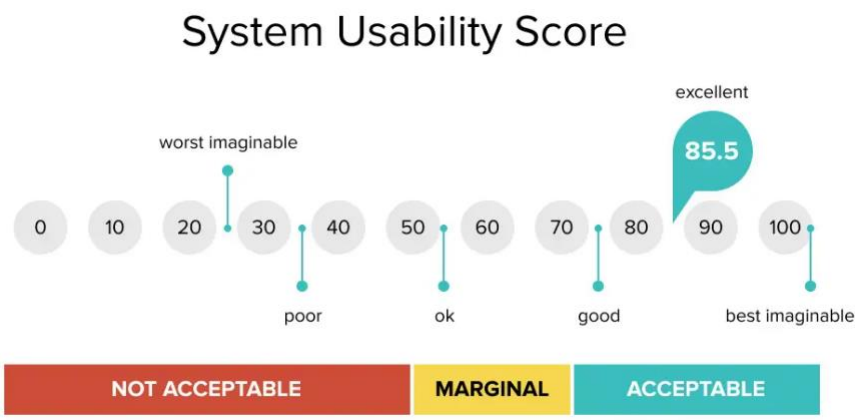


Fig. 53 Grado de calificación SUS

Fuente: (The System Usability Scale & How It's Used in UX | Adobe XD Ideas, 2021.)

Con una calificación de 84.5 en la escala de usabilidad con rangos de 0 a 100 se demuestra que el sistema cumple con el objetivo de brindar una solución funcional y usable que decremente el grado en la pérdida de clientes y mejore los tiempos en el proceso de gestión de las actividades académicas y administrativas al automatizar el flujo de trabajo.

CONCLUSIONES

- Se logró establecer un flujo de proceso adecuado, lo cual representa una disminución en los tiempos de las actividades, así como la optimización de los servicios ofertados, lo cual genera un aumento en la calidad del proceso de aprendizaje.

- Mediante la investigación y revisión sistemática de literatura se pudo establecer un marco de trabajo adecuado y determinar herramientas tecnológicas modernas que permitan desarrollar un software apropiado, seguro, confiable y que además se acople de manera eficiente al proceso de negocio y necesidades del usuario.

- La implementación de docker en el ciclo de vida del proyecto facilitó la tarea de creación de ambientes de desarrollo, producción y pruebas, mejorando en gran medida la eficiencia en la ejecución de las aplicaciones al mantener cada servicio aislado en contenedores diferentes además de permitir un escalamiento adecuado de todos los módulos complementarios a este sistema.

- La arquitectura de microservicios mejora mucho la disponibilidad, capacidad y adaptación al cambio permitiendo disminuir ampliamente el margen de error en la posibilidad de escalar el software con el transcurso del tiempo.

- Considerando la necesidad de evaluar el sistema para cumplir con un estándar internacional, se utilizó la norma ISO25010, la cual permite valorar la usabilidad del software a través de la creación de encuestas e identificar los procesos que necesitan ser reformados.

RECOMENDACIONES

- Realizar un análisis previo de toda la información recopilada para realizar el proyecto, de esto dependerá el éxito en la integración de todas las tecnologías escogidas para el desarrollo del sistema.

- Revisar la documentación de cada imagen de docker en DockerHub antes de proceder con la implementación, de esta manera el desarrollador se asegura de tener las versiones correctas de cada librería y las variables de entorno específicas para su ejecución.

- Las imágenes de docker puestas en producción deben estar optimizadas y ser los más pequeñas posibles, para lograr esto, se recomienda usar las imágenes oficiales publicadas en DockerHub utilizando Linux alpine como base y concatenar las sentencias en la instrucción RUN del archivo Dockerfile.

- La arquitectura de microservicios debe usarse en aplicaciones grandes o que tengan proyección a crecer con el transcurso del tiempo.

- En ambientes de ejecución donde la aplicación sea muy grande, manejar aplicaciones en docker con microservicios se convierte en una tarea abrumadora por la cantidad de contenedores que se deben gestionar manualmente, si este es el caso, se recomienda utilizar un motor de orquestación de contenedores como Kubernetes el cual realiza tareas fundamentales como automatizar el flujo de trabajo y optimizar los procesos de despliegue.

BIBLIOGRAFÍA

1 - Introduction | Yarn - Package Manager. (2021). https://yarnpkg.com/getting-started

5 reasons why you should consider Node.js - DEV Community. (n.d.). Retrieved November 4, 2021, from https://dev.to/tejaskaneriya/5-reasons-why-you-should-consider-node-js-5836

About npm | npm Docs. (2021). Node Package Manager. https://docs.npmjs.com/about-npm

Abuhlfaia, K., & de Quincey, E. (2019). Evaluating the usability of an e-learning platform within higher education from a student perspective. PervasiveHealth: Pervasive Computing Technologies for Healthcare, 1–7. https://doi.org/10.1145/3371647.3371661

Anwer, F., Aftab, S., Shah Muhammad Shah, S., Waheed, U., Shah, S. M., & Waheed, U. (2017). Comparative analysis of two popular agile process models: extreme programming and scrum. International Journal of Computer Science and Telecommunications, 8(2), 1–7.

Cevallos, K. (2016). Metodología de Desarrollo Ágil: XP y Scrum – Ingeniería del software. https://ingsotfwarekarlacevallos.wordpress.com/2015/05/08/metodologia-de-desarrollo-agil-xp-y-scrum/

Cruz, M. L. M. H., Segovia, M. E. C. G. M. E., Alvarez, M. D. C. M., Chan, M. J. R. C., Gonzalez, M. E. A. J. A. G., & Francisco Javier Barrera Lao, M. A. C. (2020). Analysis of the Quality in Use and Greenability with the ISO/IEC 25010 Standard. Iberian Conference on Information Systems and Technologies, CISTI, 2020-June. https://doi.org/10.23919/CISTI49556.2020.9141017

DOC, D. (2021). About Docker - Management & History | Docker. https://www.docker.com/company

DOC, M. (2021). ¿Qué es MongoDB? | MongoDB. https://www.mongodb.com/es/what-is-mongodb

DOC, N. (2021). Acerca | Node.js. https://nodejs.org/es/about/

DOC, R. (2021). ¿Qué es Docker? https://www.redhat.com/es/topics/containers/what-is-docker

Edge node architecture with npm Enterprise | by npm, Inc. | Medium. (2021). https://medium.com/@npmjs/edge-node-architecture-with-npm-enterprise-230121953910

Estdale, J., & Georgiadou, E. (2018). Applying the ISO/IEC 25010 Quality Models to Software Product. Communications in Computer and Information Science, 896, 492–503. https://doi.org/10.1007/978-3-319-97925-0_42

Hedlefs Aguilar, & Garza Villegas. (2008). An empirical evaluation of the system usability scale. International Journal of Human-Computer Interaction, 24(6), 574–594. https://doi.org/10.1080/10447310802205776

Hovorushchenko, T. (2018). Methodology of evaluating the sufficiency of information for software quality assessment according to ISO 25010. Journal of Information and Organizational Sciences, 42(1), 63–85. https://doi.org/10.31341/jios.42.1.4

ISO. (2015). Iso 25010. In Iso 25000 (p. 3).

Modelio. (2020). ModelioStudio. https://www.modelio.org/about-modelio/features.html

Montero, B. M., Cevallos, H. V., & Cuesta, J. D. (2018). Espirales revista multidisciplinaria de investigación. Espirales Revista Multidisciplinaria de Investigación, 2(17).

Navarro, G., Nelson, B., & Tataryan, B. (2017). Integración y evolución de sistemas de información del DCC. Universidad de Chile.

PostgreSQL: About. (2019). https://www.postgresql.org/about/

Redux Refresher. I have not used Redux in a while, so… | by Christopher Diep | Medium. (n.d.). Retrieved January 26, 2022, from https://medium.com/@diep.christopher/redux-refresher-b799a0b9380c

The System Usability Scale & How it's Used in UX | Adobe XD Ideas. (n.d.-a). Retrieved January 24, 2022, from https://xd.adobe.com/ideas/process/user-testing/sus-system-usability-scale-ux/

The System Usability Scale & How it's Used in UX | Adobe XD Ideas. (n.d.-b). Retrieved January 24, 2022, from https://xd.adobe.com/ideas/process/user-testing/sus-system-usability-scale-ux/

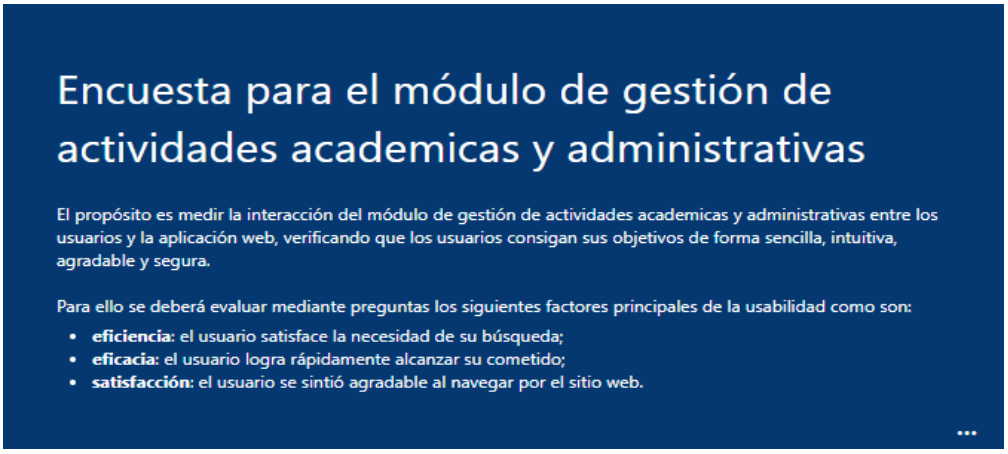
Vayghan, L. A., Saied, M. A., Toeroe, M., & Khendek, F. (2021). A Kubernetes controller for managing the availability of elastic microservice based stateful applications. Journal of Systems and Software, 110924. https://doi.org/10.1016/j.jss.2021.110924

What is Bonita? | Bonita Documentation. (n.d.). Retrieved November 4, 2021, from https://documentation.bonitasoft.com/bonita/2021.2/bonita-overview/what-is-bonita-index

Anexos

Anexo A: Encuesta de usabilidad SUS

A continuación, se presenta la encuesta de escala de usabilidad del sistema(SUS) realizada para validar los resultados de la solución propuesta.



	Totalmente en desacuerdo	En desacuerdo	Ni de acuerdo ni en desacuerdo	De acuerdo	Totalmente de acuerdo
Cree que me gustaría utilizar frecuentemente este sitio web.	○	○	○	○	○
Encontró el sitio web sencillo.	○	○	○	○	○
Piensa que el sitio web es fácil de usar.	○	○	○	○	○
Piensa que podré utilizar este sitio web sin el apoyo de personal técnico.	○	○	○	○	○
Encontró que varias de las funciones en el sitio web estaban bien integradas.	○	○	○	○	○
Piensa que había demasiada consistencia en el sitio web.	○	○	○	○	○
Imagina que la mayoría de las personas podrían aprender a usar este sitio web muy rápido	○	○	○	○	○
Encontró el sitio web muy intuitivo.	○	○	○	○	○
Se sintió muy confiado (seguro) al utilizar el sitio web.	○	○	○	○	○
Puede utilizar el sitio web sin tener que aprender algo nuevo.	○	○	○	○	○