

UNIVERSIDAD MAYOR DE SAN ANDRÉS
FACULTAD DE CIENCIAS PURAS Y NATURALES
CARRERA DE INFORMÁTICA



PROYECTO DE GRADO
SOFTWARE DE LOGÍSTICA Y GESTIÓN DE BUSES PARA TRANSPORTE DE
PASAJEROS Y ENVÍO DE ENCOMIENDAS.

CASO: EMPRESA DE TRANSPORTES CALI INTERNACIONAL

Proyecto de Grado para obtener el Título de Licenciatura en Informática

Mención: Ingeniería de Sistemas Informáticos

POSTULANTE: BLADIMIR WILSON RAMOS ESCOBAR

TUTOR: Ph. D. FRANZ CUEVAS QUIROZ

NUESTRA SEÑORA DE LA PAZ – BOLIVIA

2025

DEDICATORIA

A mis padres, por ser el pilar fundamental de mi vida y por enseñarme con su ejemplo el valor del esfuerzo y la perseverancia. En especial, a mi querida madre Celia (+), que desde el cielo me acompaña e ilumina mi camino; por ser la única que confiaba en mí, su amor infinito sigue guiando cada uno de mis pasos.

A mi amada esposa Rosmery, por su paciencia, apoyo incondicional y por creer en mí incluso en los momentos más difíciles.

Y a mis cuatro patitas Duke, mi leal compañero, cuyo cariño y alegría hicieron más llevaderos los días de estudio.

A todos ustedes, gracias por ser mi inspiración y mi fuerza.

AGRADECIMIENTOS

A la Universidad Mayor de San Andrés, y en especial a la Carrera de Informática de la Facultad de Ciencias Puras y Naturales, por proporcionar la formación académica que hizo posible la ejecución del presente proyecto.

Al Ph.D. Franz Cuevas Quiroz, tutor de este trabajo, por su guía constante, orientación metodológica y compromiso durante todas las etapas del desarrollo del proyecto.

A la Empresa de Transportes Cali Internacional, por permitir el acceso a información clave de sus procesos operativos, y por la disposición y colaboración del personal durante el levantamiento de requerimientos y la validación del sistema.

Asimismo, se agradece a todas las personas que brindaron apoyo técnico y académico en la implementación y evaluación del Software de logística y gestión de buses para transporte de pasajeros y envío de encomiendas.

A todos, mi más sincero reconocimiento y gratitud.

RESUMEN

El proyecto titulado “Software de logística y gestión de buses para transporte de pasajeros y envío de encomiendas” tiene como objetivo implementar un sistema computarizado que optimice los servicios de venta y reserva de pasajes, así como la gestión de la recepción y entrega de encomiendas en la empresa. Durante el análisis de procesos, se identificó que los registros de pasajes y encomiendas se realizaban de forma manual, lo que resultaba en un uso ineficiente del tiempo y los recursos materiales.

El desarrollo del sistema se fundamentó en un análisis detallado de las actividades de la empresa Cali Internacional, empleando el modelo en cascada como metodología principal. Los requisitos funcionales y no funcionales fueron obtenidos a través de observaciones directas y entrevistas con el personal. Para la implementación técnica, se seleccionó el framework Django de Python, aprovechando sus capacidades de programación orientada a objetos, junto con PostgreSQL como gestor de base de datos, debido a su excelente integración con Django.

La finalización del proyecto permitió una notable optimización de las operaciones, reflejada en una reducción del 64.62 por ciento en los tiempos de procesamiento. Las actividades relacionadas con el registro y la venta de boletos se agilizaron considerablemente, mejorando la atención al cliente y haciendo el proceso más eficiente. Este avance tuvo un impacto positivo en la productividad diaria y la calidad del servicio brindado.

Palabras clave: pasajes, encomiendas, Python, Django.

ABSTRACT

The project entitled “Logistics and bus management software for passenger transport and parcel delivery” aims to implement a computerized system that optimizes ticket sales and reservation services, as well as the reception and delivery of packages. During the process analysis, it was identified that ticket and package records were handled manually, which resulted in inefficient use of time and material resources.

The system development was based on a detailed analysis of the activities of the company Cali Internacional, using the waterfall model as the main methodology. Functional and non-functional requirements were obtained through direct observations and interviews with the staff. For the technical implementation, the Django framework from Python was selected, leveraging its object-oriented programming capabilities, along with PostgreSQL as the database manager, due to its excellent integration with Django.

The completion of the project enabled a significant optimization of operations, reflected in a 64.62 percent reduction in processing times. Activities related to ticket registration and sales were significantly streamlined, improving customer service and making the process more efficient. This advance had a positive impact on daily productivity and service quality.

Keywords: tickets, parcels, Python, Django.

CONTENIDO

| | |
|--|-----------|
| RESUMEN | |
| ABSTRACT | |
| ÍNDICE DE FIGURAS | |
| ÍNDICE DE TABLAS | |
| 1. INTRODUCCIÓN | 1 |
| 1.1. ANTECEDENTES | 3 |
| 1.1.1. Antecedentes institucionales..... | 4 |
| 1.1.2. Proyectos similares..... | 6 |
| 1.2. OBJETO DE ESTUDIO..... | 8 |
| 1.3. PLANTEAMIENTO DEL PROBLEMA..... | 9 |
| 1.4. JUSTIFICACIÓN | 9 |
| 1.5. OBJETIVOS | 11 |
| 1.5.1. Objetivo general | 11 |
| 1.5.2. Objetivos específicos..... | 11 |
| 1.6. ALCANCES Y LÍMITES | 11 |
| 1.6.1. Alcances | 11 |
| 1.6.2. Límites | 12 |
| 1.7. IMPORTANCIA DEL ESTUDIO..... | 12 |
| 2. MARCO TEÓRICO | 14 |
| 2.1. LOGÍSTICA DEL TRANSPORTE DE PASAJEROS | 14 |
| 2.1.1. Logística | 14 |
| 2.1.2. Transporte | 14 |

| | | |
|--------|--|----|
| 2.1.3. | Pasajero | 15 |
| 2.1.4. | Sistema de transporte..... | 16 |
| 2.2. | RESERVA Y VENTA DE PASAJES | 16 |
| 2.2.1. | Proceso de reserva y venta de pasajes | 16 |
| 2.2.2. | Emisión y gestión de boletos | 16 |
| 2.2.3. | Cambios y cancelaciones | 17 |
| 2.3. | LOGÍSTICA DE ENVÍO DE ENCOMIENDAS | 17 |
| 2.3.1. | Encomienda | 17 |
| 2.3.2. | Proceso de recepción | 18 |
| 2.3.3. | Clasificación de encomiendas | 18 |
| 2.3.4. | Embalaje y etiquetado | 19 |
| 2.3.5. | Entrega al destinatario..... | 19 |
| 2.4. | GESTIÓN DE BUSES | 20 |
| 2.4.1. | Asignación de rutas | 20 |
| 2.4.2. | Programación y asignación de conductores..... | 20 |
| 2.5. | MARCO LEGAL Y NORMATIVO..... | 21 |
| 2.5.1. | Autoridad de regulación y fiscalización de telecomunicaciones y transportes .. | 21 |
| 2.5.2. | Ley general del transporte | 22 |
| 2.5.3. | Reglamento regulatorio de transporte terrestre de pasajeros y carga | 23 |
| 2.6. | INGENIERÍA DE SOFTWARE | 26 |
| 2.7. | MODELO EN CASCADA | 27 |
| 2.7.1. | Análisis y definición de requerimientos | 28 |
| 2.7.2. | Diseño del sistema y del software | 28 |

| | | |
|---------|---|----|
| 2.7.3. | Implementación y prueba de unidad | 29 |
| 2.7.4. | Integración y prueba de sistema | 29 |
| 2.7.5. | Operación y mantenimiento..... | 30 |
| 2.8. | INGENIERÍA DE REQUERIMIENTOS..... | 30 |
| 2.8.1. | Requerimientos funcionales..... | 30 |
| 2.8.2. | Requerimientos no funcionales | 31 |
| 2.8.3. | Actividades de la ingeniería de requerimientos..... | 31 |
| 2.9. | MODELO ENTIDAD - RELACIÓN | 39 |
| 2.10. | BASE DE DATOS | 41 |
| 2.10.1. | Base de datos relacional | 42 |
| 2.10.2. | Modelo relacional..... | 42 |
| 2.10.3. | Normalización | 42 |
| 2.11. | PATRÓN DE DISEÑO WEB | 44 |
| 2.12. | DISEÑO EN FORMA DE F..... | 45 |
| 2.13. | PATRÓN DE ARQUITECTURA | 46 |
| 2.14. | PATRÓN MODEL-TEMPLATE-VIEW (MTV)..... | 48 |
| 2.15. | HERRAMIENTAS DE DESARROLLO | 48 |
| 2.16. | PRUEBAS | 50 |
| 2.17. | MODELO DE CALIDAD BOEHM | 51 |
| 2.17.1. | Estructura jerárquica del modelo de calidad Boehm | 51 |
| 2.17.2. | Usabilidad..... | 52 |
| 2.17.3. | Portabilidad | 53 |

| | |
|---|-----------|
| 3. MARCO APLICATIVO | 55 |
| 3.1. ANÁLISIS Y DEFINICIÓN DE REQUERIMIENTOS..... | 55 |
| 3.1.1. Levantamiento de requerimientos | 55 |
| 3.1.2. Análisis de requerimientos..... | 59 |
| 3.1.3. Especificación de requerimientos | 69 |
| 3.1.4. Gestión de requerimientos | 70 |
| 3.2. DISEÑO DEL SISTEMA Y DEL SOFTWARE | 73 |
| 3.2.1. Diagrama entidad - relación..... | 73 |
| 3.2.2. Diagrama relacional..... | 75 |
| 3.2.3. Diseño de la interfaz | 77 |
| 3.3. IMPLEMENTACIÓN Y PRUEBA DE UNIDAD..... | 80 |
| 3.3.1. Gestión de usuarios | 80 |
| 3.3.2. Gestión de rutas..... | 82 |
| 3.3.3. Gestión de buses | 83 |
| 3.3.4. Gestión de viajes | 85 |
| 3.3.5. Venta y reserva de pasajes | 87 |
| 3.3.6. Registro de encomiendas..... | 89 |
| 3.3.7. Gestión de clientes | 90 |
| 3.4. PRUEBA DE SISTEMA | 92 |
| 3.4.1. Pruebas unitarias | 92 |
| 3.4.2. Resultados de las pruebas | 96 |
| 3.5. OPERACIÓN Y MANTENIMIENTO | 97 |
| 3.6. RESULTADOS | 102 |

| | |
|---|------------|
| 4. CALIDAD Y SEGURIDAD..... | 107 |
| 4.1. CALIDAD DE SOFTWARE..... | 107 |
| 4.1.1. Usabilidad..... | 107 |
| 4.1.2. Portabilidad | 108 |
| 4.1.3. Mantenibilidad..... | 110 |
| 4.2. SEGURIDAD | 111 |
| 4.2.1. Autenticación y control de acceso | 112 |
| 4.2.2. Almacenamiento seguro de contraseñas | 112 |
| 4.2.3. Prevención de ataques comunes | 113 |
| 4.2.4. Seguridad en la transmisión de datos | 114 |
| 4.2.5. Gestión de sesiones | 115 |
| 4.2.6. Auditoría y registro de actividades | 116 |
| 5. CONCLUSIONES Y RECOMENDACIONES | 117 |
| 5.1. CONCLUSIONES | 117 |
| 5.2. RECOMENDACIONES | 118 |
| Apéndice A | 124 |
| Apéndice B | 125 |

ÍNDICE DE FIGURAS

| Fig. | Descripción | Pag. |
|-------------|--|-------------|
| 1.1 | Organigrama empresa | 5 |
| 2.1 | Etapas del Modelo en Cascada | 28 |
| 2.2 | Fases de la entrevista | 33 |
| 2.3 | Ejemplo Diagrama de casos de uso | 36 |
| 2.4 | Ejemplo MER | 41 |
| 2.5 | Ejemplo Primera Forma Normal | 43 |
| 2.6 | Ejemplo Segunda Forma Normal | 43 |
| 2.7 | Ejemplo Tercera Forma Normal | 44 |
| 2.8 | Estudios de seguimiento ocular | 45 |
| 2.9 | Diseño en forma de F | 47 |
| 2.10 | Estructura jerárquica del Modelo de Calidad Boehm | 52 |
| 3.1 | Diagrama de Casos de Uso | 60 |
| 3.2 | Diagrama de actividades - Ingresar al sistema | 70 |
| 3.3 | Diagrama de actividades - Venta y reserva de pasajes | 71 |
| 3.4 | Diagrama de actividades - Registro de encomiendas | 72 |
| 3.5 | Diagrama de actividades - Registro de personal | 73 |
| 3.6 | Diagrama Entidad-Relación | 74 |
| 3.7 | Diagrama Entidad-Relación | 75 |
| 3.8 | Diagrama Relacional | 76 |
| 3.9 | Diseño de la página web principal del sistema | 77 |

| | |
|---|-----|
| 3.10 Diseño del inicio de sesión | 78 |
| 3.11 Diseño de la pantalla principal | 78 |
| 3.12 Diseño interfaz Venta de pasajes | 79 |
| 3.13 Diseño interfaz Envío de encomiendas | 79 |
| 3.14 Pruebas Unitarias | 96 |
| 3.15 Página web - Cali Internacional | 97 |
| 3.16 Inicio de sesión | 98 |
| 3.17 Panel principal del sistema | 98 |
| 3.18 Creación de clientes | 99 |
| 3.19 Listado de clientes | 99 |
| 3.20 Creación de un viaje | 100 |
| 3.21 Venta o reserva de pasajes | 100 |
| 3.22 Gestión de encomiendas | 101 |
| 3.23 Informe de ventas | 101 |
| 3.24 Comparación de mejora de tiempos | 105 |
| 4.1 Navegador Brave | 109 |
| 4.2 Navegador Microsoft Edge | 109 |
| 4.3 Navegador Chrome - Móvil | 110 |
| 4.4 Lista de accesos | 116 |

ÍNDICE DE TABLAS

| Tabla | Descripción | Pag. |
|--------------|---|-------------|
| 2.1 | Elementos de un diagrama de casos de uso | 35 |
| 2.2 | Herramientas front-end | 49 |
| 2.3 | Herramientas back-end | 49 |
| 2.4 | Herramientas de programación | 50 |
| 3.1 | Cronograma de reuniones y entrevistas | 55 |
| 3.2 | Especificación de casos de uso: Autenticación del usuario | 61 |
| 3.3 | Especificación de casos de uso: Gestionar bus | 62 |
| 3.4 | Especificación de casos de uso: Gestionar rutas | 63 |
| 3.5 | Especificación de casos de uso: Gestionar viajes | 64 |
| 3.6 | Especificación de casos de uso: Gestionar venta y reserva de pasaje | 65 |
| 3.7 | Especificación de casos de uso: Gestionar rol de usuario | 66 |
| 3.8 | Especificación de casos de uso: Gestionar encomienda | 67 |
| 3.9 | Especificación de casos de uso: Generar reporte | 68 |
| 3.10 | Lista de Requerimientos Funcionales | 69 |
| 3.11 | Lista de Requerimientos No Funcionales | 70 |
| 3.12 | Tiempos para Registro de Usuarios | 102 |
| 3.13 | Tiempos para Venta de Pasajes | 103 |
| 3.14 | Tiempos para Reserva de Pasajes | 103 |
| 3.15 | Tiempos para Registro de Encomiendas | 104 |
| 3.16 | Tiempos para Generar Reportes | 104 |
| 3.17 | Resumen tiempos de actividades | 105 |

| | |
|---|-----|
| 4.1 Número de Clics - Escala Likert | 108 |
|---|-----|

1 INTRODUCCIÓN

Los avances actuales de la informática (2025) y la difusión global de la Internet han cambiado la manera en que se desarrollan las actividades de la sociedad en los ámbitos de la comunicación, la calidad de vida y el comercio. Internet ofrece nuevas alternativas de negocio ya que esta nos permite llegar a una audiencia masiva y a un gran número de posibles clientes; se puede ofrecer nuestros servicios a un mercado mucho mayor porque el tiempo y la distancia dejan de ser obstáculos (Abarca et al., 2009).

Este desarrollo de la tecnología y su notable avance han hecho posible que los sistemas de información se integren en empresas, ya sean pequeñas, medianas o grandes. La competitividad del mercado ha sido el principal impulsor de este fenómeno, ya que obliga a las organizaciones a actualizar y mejorar sus mecanismos operativos para seguir siendo eficientes. Es fundamental en este escenario incorporar un sistema de información que no solo facilite la gestión y control de las operaciones, sino también brinde una solución completa para mejorar los procedimientos internos de la empresa. La adopción de estos sistemas tecnológicos brinda beneficios importantes al facilitar un seguimiento más preciso, la automatización de tareas repetitivas y una toma de decisiones mejorada mediante el uso de datos confiables en tiempo real. Además de mejorar la eficiencia operativa, esta acción también fortalece la capacidad de adaptación de la empresa a las demandas cambiantes del mercado.

Según Casanueva y García (2000) una empresa es como una entidad que mediante la organización de elementos humanos, materiales, técnicos y financieros proporciona bienes o servicios

a cambio de un precio que le permite la reposición de los recursos empleados y la consecución de unos objetivos determinados. Manejar grandes cantidades de información dentro de cualquier empresa demanda un nivel elevado de responsabilidad, usualmente, las compañías ponen más énfasis en la promoción de sus productos o servicios, sin embargo, es crucial no descuidar el aspecto administrativo.

Para las empresas de transporte y logística la digitalización de sus servicios se ha convertido en un factor crucial para la competitividad y eficiencia, la integración de soluciones tecnológicas ha permitido a muchas organizaciones optimizar sus operaciones y mejorar la experiencia del cliente. Las empresas de transporte y logística enfrentan la necesidad de modernizar sus sistemas para satisfacer las expectativas de sus clientes.

En este contexto, el desarrollo de un software de logística y gestión de buses para transporte de pasajeros y envío de encomiendas representa una oportunidad significativa para modernizar las operaciones y mejorar la experiencia del cliente.

Para una mejor percepción del presente proyecto a continuación se describe la estructura del documento.

En el capítulo I se establece las bases al plantear la problemática central, definir los objetivos clave y determinar tanto los alcances como las limitaciones que enmarcan la investigación.

El capítulo II provee el soporte teórico necesario, presentando una compilación de referencias bibliográficas y conceptos fundamentales que sustentan la investigación y el desarrollo del proyecto.

En el capítulo III se realiza un análisis exhaustivo y se desarrolla el proyecto, apoyándose en la base teórica establecida en el capítulo anterior. Esta sección examina detalladamente cada aspecto de la implementación.

En el capítulo IV se abordan los criterios de calidad y las medidas de seguridad implementadas en el sistema, garantizando su correcto funcionamiento, integridad de datos y protección frente

a posibles riesgos técnicos o humanos.

La investigación culmina con el capítulo V, donde se exponen las conclusiones derivadas del trabajo realizado y se presentan las recomendaciones específicas para optimizar y dar continuidad a los resultados alcanzados.

1.1. ANTECEDENTES

La transformación digital ha impactado significativamente a diversas industrias, incluida la del transporte y la logística. La creciente demanda por servicios rápidos, eficientes y accesibles ha impulsado a las empresas a adoptar tecnologías avanzadas para mejorar sus operaciones.

La implementación de software especializado en la venta de pasajes y gestión de encomiendas no es un concepto nuevo, pero su evolución ha sido notable. Con el tiempo, la incorporación de tecnologías más avanzadas, como bases de datos relacionales, interfaces de usuario mejoradas y capacidades de integración con otros sistemas, ha permitido el desarrollo de soluciones más robustas y eficientes. Estos avances han sido impulsados por la necesidad de mejorar la experiencia del cliente, reducir costos operativos y aumentar la competitividad en un mercado cada vez más exigente.

La adopción de tecnologías como la computación en la nube, el Internet de las Cosas (IoT) y el análisis de big data ha abierto nuevas posibilidades para la gestión integral de operaciones en el transporte terrestre. Estas tecnologías permiten la creación de ecosistemas digitales donde la venta de boletos, la gestión de flotas y el manejo de encomiendas pueden integrarse de manera fluida y eficiente. Sin embargo, el desarrollo e implementación de tales sistemas integrales presenta desafíos significativos, desde la complejidad técnica hasta la necesidad de adaptarse a diversas regulaciones y prácticas operativas existentes.

A nivel global, muchas empresas de transporte y logística han adoptado con éxito plataformas digitales para la venta de pasajes y gestión de encomiendas, logrando mejoras significativas en sus

operaciones. Por ejemplo, compañías de renombre han implementado sistemas que permiten a los clientes reservar boletos y rastrear envíos en tiempo real, lo que ha aumentado la satisfacción del cliente y optimizado el flujo de trabajo interno.

1.1.1. Antecedentes institucionales

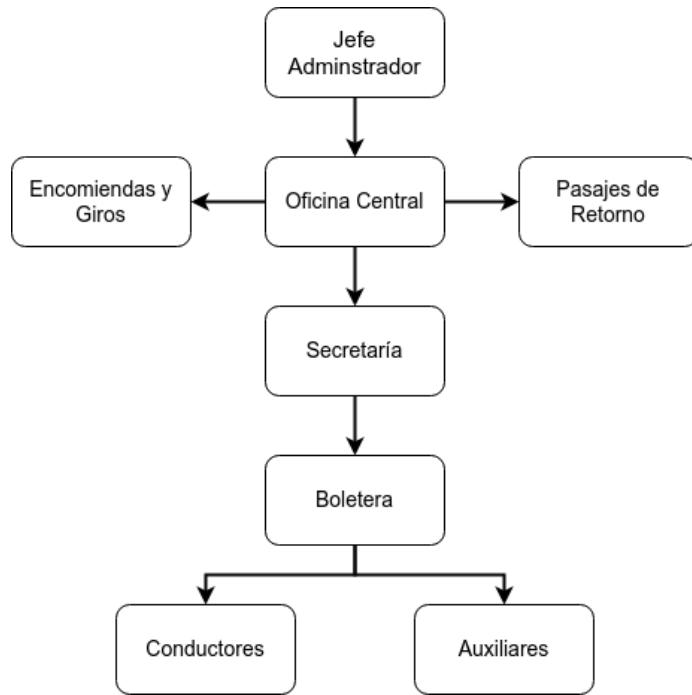
La empresa de transportes Cali Internacional, con sede en la Terminal de Buses de La Paz y número de NIT 491462023, es una compañía destacada en el sector del transporte y la logística en Bolivia, desde su fundación, Cali Internacional ha brindado servicios de venta de pasajes y gestión de encomiendas, ganándose una sólida reputación por su compromiso con la calidad y la satisfacción del cliente. La ubicación estratégica en la Terminal de Buses de La Paz permite a la empresa atender a un amplio espectro de clientes, facilitando tanto los viajes como el envío de paquetes de manera eficiente y segura.

A lo largo de los años, Cali Internacional ha experimentado un crecimiento constante, adaptándose a los cambios del mercado y las necesidades de sus clientes. La empresa ha reconocido la importancia de incorporar tecnologías avanzadas para mejorar sus operaciones y mantenerse competitiva. Actualmente, enfrenta el desafío de modernizar sus procesos tradicionales de venta de pasajes y gestión de encomiendas, buscando una solución tecnológica que optimice sus operaciones y reduzca las ineficiencias. A continuación, en la (figura 1.1) se muestra el organigrama de la empresa de transportes Cali Internacional.

En la figura 1.1, se observa la estructura organizativa de la empresa, destacando los diferentes cargos que desempeñan los empleados, que van desde el administrador hasta los auxiliares de apoyo. Dentro del negocio, se encuentran los boleteros y conductores, quienes son responsables de la atención directa a los pasajeros y la operación de los vehículos. Paralelamente, en la oficina central se gestionan las encomiendas, que son recibidas, clasificadas y preparadas para su envío. Cada uno de estos roles desempeña una función esencial para el funcionamiento eficiente y efectivo

Figura 1.1

Organigrama de la empresa de transportes “Cali Internacional”.



Nota. Organigrama obtenido en entrevista con el administrador.

de la empresa, asegurando que tanto el transporte de pasajeros como la gestión de encomiendas se realicen con éxito y dentro de los estándares de calidad establecidos.

Misión de la empresa

Proporcionar servicios de transporte y logística de alta calidad, enfocándose en la venta de pasajes y el envío de encomiendas, con el objetivo de satisfacer plenamente las necesidades de nuestros clientes. Nos comprometemos a ofrecer un servicio eficiente, seguro y confiable, contribuyendo al bienestar y comodidad de nuestros usuarios.

Visión de la empresa

Ser la empresa líder en el sector del transporte y la logística en Bolivia, reconocida por nuestra innovación, eficiencia operativa y excelencia en el servicio al cliente. Aspiramos a expandir nuestra presencia y mejorar continuamente nuestros servicios para mantenernos a la vanguardia

de la industria.

Objetivo general de la empresa

El objetivo general de Cali Internacional es consolidar y expandir nuestra posición en el mercado del transporte y la logística, mejorando continuamente la calidad de nuestros servicios y adoptando tecnologías avanzadas para optimizar nuestras operaciones y satisfacer las necesidades cambiantes de nuestros clientes.

Objetivos específicos de la empresa

- Mejorar la experiencia del cliente mediante la oferta de servicios más rápidos, seguros y fiables.
- Capacitar continuamente a nuestro personal para asegurar que estén equipados con las habilidades necesarias para manejar las nuevas tecnologías y brindar un servicio de alta calidad.
- Implementar prácticas sostenibles en nuestras operaciones, minimizando el impacto ambiental y promoviendo la responsabilidad social corporativa.

1.1.2. Proyectos similares

Para la presente investigación se han considerado los siguientes antecedentes:

Hurtado Samaniego (2019). “Aplicación web administrativa para reserva de servicios de transporte y envío de encomiendas para la empresa Romero y Asociados (AMBASEUR) de la ciudad de Ambato”. En este proyecto, se implementó una aplicación web para automatizar los procesos manuales de una empresa, mejorando la gestión de reservas de transporte y envíos de encomiendas. La plataforma permite publicitar las actividades de la empresa y recopilar información precisa sobre los clientes. Desarrollada utilizando la metodología XP, la aplicación facilita la adaptación rápida a cambios y la incorporación de funciones adicionales, como un chat en línea, optimizando así la eficiencia y aumentando la base de clientes.

Mora (2022). “Sistema gestión de servicio de viajes para la empresa Nuestra Señora de la Asunción C.I.S.A.”, esta investigación se centra en automatizar los procesos manuales de la empresa Nuestra Señora de la Asunción CISA mediante un sistema informático. En la primera etapa, se diagnosticaron los módulos de viajes, tráfico y ventas, entrevistando a responsables clave y recopilando los requerimientos necesarios. En la segunda etapa, se desarrolló un sistema informático web responsive que procesa automáticamente la información de estos módulos, integrando análisis, diseño y programación orientada a objetos, culminando en un sistema integrado con soporte audiovisual.

Arévalo Pineda y Vargas Gallardo (2021). “Desarrollo de una aplicación web para agilizar los procesos de la compra y venta de boletos de buses interprovinciales en el terminal de Milagro.”, este proyecto desarrolló un sistema web para la compra y venta de boletos en el terminal terrestre del Cantón Milagro, con el objetivo de agilizar el proceso de boletería sin necesidad de contacto físico en ventanilla. Tras entrevistar a los socios del terminal para identificar los requisitos funcionales y no funcionales, se eligió la metodología ágil SCRUM para la organización y monitoreo constante del proyecto. El sistema se implementó utilizando Python, con Pycharm como IDE, Bootstrap 4 y Adminlte3 como plantillas, y PostgreSQL como base de datos. El resultado fue un sistema que satisface las necesidades del cliente, mejorando significativamente la experiencia de compra de boletos..

Sosa Pajuelo (2019). “Sistema informático web para la gestión de pasajes de la empresa de transporte Turismo Transol Barranca S.A.C.”, en la tesis se propone como objetivo principal desarrollar un sistema informático web para la gestión de pasajes en la empresa de transportes Turismo Transol Barranca S.A.C., abarcando tanto la venta como la reserva de boletos. Este sistema busca optimizar el tiempo de procesamiento mediante el uso de tecnología web. La investigación se llevó a cabo con un enfoque descriptivo, un diseño no experimental y un corte transversal,

utilizando una población de 42 personas y una muestra de 6 usuarios. Se aplicó la metodología Proceso Unificado de Rational (RUP), empleando el Lenguaje de Modelamiento Unificado (UML) para la construcción de diagramas de casos de uso, facilitando el análisis del software. El sistema fue desarrollado en Java, con MySQL como gestor de datos y MySQL Workbench 6.3 CE para el modelado de la base de datos, entre otras herramientas que ayudaron a cumplir los requisitos de diseño. Los resultados permitieron agilizar los procesos de venta y reserva de pasajes, mejorando el manejo de la información y extendiendo el alcance a los clientes, lo que fortaleció el posicionamiento competitivo de la empresa a nivel regional.

Vivas Mena (2019). “Propuesta de implementación del sistema web de venta de boletos de viaje y gestión de encomiendas para la empresa Transportes Montero S.A.C. Piura; 2018.”, en esta investigación que fue desarrollada por la Escuela Profesional de Ingeniería de Sistemas de la Universidad Católica Los Ángeles de Chimbote, se centró en la mejora de procesos en organizaciones peruanas mediante la implementación de un sistema web para la venta de boletos y gestión de encomiendas en la empresa TRANSPORTES MONTERO S.A.C. en 2018. La investigación, de tipo cuantitativo y descriptivo con diseño no experimental y corte transversal, incluyó una muestra de 14 trabajadores. Los resultados mostraron que el 63 por ciento de los empleados consideraba que la empresa brindaba calidad en procesos y servicios, el 84 por ciento creía que los sistemas web agilizan los procesos, y el 81 por ciento opinaba que dichos sistemas eran eficientes, confirmando así la hipótesis planteada.

1.2. OBJETO DE ESTUDIO

Software de logística y gestión de buses para transporte de pasajeros y envío de encomiendas, el cual va automatizar y mejorar la venta de pasajes, así como en la recepción, procesamiento y envío de encomiendas.

1.3. PLANTEAMIENTO DEL PROBLEMA

La empresa de transportes Cali Internacional, se encuentra ante diversos retos importantes en cuanto a administrar sus procesos tanto de venta de pasajes como de envío de paquetería, estas operaciones son llevadas a cabo de forma manual, lo que genera ineficiencias en el funcionamiento, largos tiempos de espera para los clientes y una alta posibilidad de cometer errores. Además de afectar negativamente la experiencia del cliente, estos problemas también restringen las posibilidades de crecimiento y competencia efectiva en un mercado cada vez más digital, la implementación de soluciones tecnológicas integrales se ha convertido en una estrategia clave para optimizar procesos.

Algunos de los problemas mas frecuentes son:

- Largos tiempos de espera en la compra de pasajes y envío de encomiendas debido a la falta de automatización.
- Errores en la gestión de reservas y envíos, lo que puede resultar en pérdidas financieras y descontento entre los clientes.
- Falta de visibilidad y control sobre la demanda de servicios, lo que limita la capacidad de la empresa para ajustar su oferta y optimizar recursos.
- Dificultad para generar reportes y análisis que ayuden en la toma de decisiones estratégicas para la empresa.

Por lo tanto, se plantea la siguiente interrogante:

¿Cómo mejorar la venta de pasajes y la gestión de envío de encomiendas en la empresa Cali Internacional?

1.4. JUSTIFICACIÓN

La implementación de un software de logística y gestión de buses para transporte de pasajeros y envío de encomiendas representa una respuesta estratégica ante la creciente demanda de

soluciones tecnológicas en el sector de transporte y logística. La automatización de estos procesos no solo optimiza las operaciones internas, sino que también reduce significativamente los errores humanos y mejora la eficiencia. En vista de ello la mayoría de organizaciones se ha visto obligada a desarrollar un sistema web de calidad que brinde un mejor servicio a la comunidad, mejorando su imagen corporativa, demostrando que están al día con las nuevas tecnologías (Nuñez & Tituaña, 2005).

Además, este proyecto aborda la necesidad de ofrecer un servicio más accesible y conveniente para los clientes. En un entorno donde la digitalización se ha vuelto imprescindible, la adopción de un sistema informático para estos servicios es una ventaja competitiva que no se puede ignorar.

La digitalización de estos procesos en una plataforma única no solo agilizará las operaciones al automatizar tareas repetitivas y reducir la necesidad de intervención manual, sino que también mejorará significativamente la precisión y la transparencia de la información. Esta mejora permitirá a la empresa ofrecer un servicio más coherente y eficiente, ya que todos los datos estarán centralizados y accesibles en tiempo real, lo que facilitará una gestión más efectiva de los recursos. Además, la integración de estos procesos en una sola plataforma reducirá costos operativos al eliminar redundancias y optimizar el uso de la infraestructura tecnológica. En última instancia, esto resultará en una mejor experiencia para el cliente, aumentando su satisfacción al recibir un servicio más rápido y confiable, y posicionando a la empresa como líder en innovación y eficiencia dentro de su sector.

Este proyecto se adapta a la necesidad de mantenerse al día con las tendencias tecnológicas actuales. Las empresas están siendo revolucionadas por la transformación digital, y aquellas que no se adapten corren el riesgo de quedarse atrás. Cuando la empresa implementa un software especializado, no solo se adapta a estas tendencias, sino que también está preparada para hacer frente a los desafíos futuros como la necesidad de incorporar nuevas tecnologías y responder a las

demandas del mercado en constante cambio.

1.5. OBJETIVOS

1.5.1. Objetivo general

Desarrollar un software de logística y gestión de buses para transporte de pasajeros y envío de encomiendas para la empresa de transportes Cali Internacional de la ciudad de La Paz.

1.5.2. Objetivos específicos

- Analizar los procesos actuales de venta de pasajes y envío de encomiendas en la empresa de transportes Cali Internacional para identificar las áreas de mejora y las necesidades tecnológicas específicas.
- Formular un diseño de interfaz centrado en la experiencia del usuario, que facilite la interacción con el sistema.
- Elaborar el diseño de la base de datos relacional a partir del análisis de los requerimientos del sistema, para llevar a la Tercera Forma Normal (3FN) y almacenar los datos.
- Diseñar el back-end para gestionar la venta de pasajes y el envío de encomiendas, asegurando la integración eficiente con la base de datos y la correcta ejecución de las operaciones solicitadas por los usuarios.
- Generar reportes y análisis de datos que facilite la toma de decisiones informadas por parte de la administración de la empresa.

1.6. ALCANCES Y LÍMITES

1.6.1. Alcances

El desarrollo de la presente investigación se encuentra dentro de los siguientes alcances:

- El proyecto abarcará la creación de una plataforma digital que permita a los usuarios realizar la compra de pasajes y la gestión de envíos de encomiendas de manera eficiente y segura.

- Se desarrollará un sistema de gestión de usuarios que permitirá a los empleados: iniciar sesión y gestionar las ventas de pasajes y envíos de encomiendas, mientras que los administradores podrán supervisar y manejar las operaciones.
- Se implementarán módulos que automatizarán tareas repetitivas como la generación de recibos, el seguimiento de envíos y la asignación de asientos en los buses de transportes.
- El sistema incluirá un módulo de reportes que permitirá a los administradores generar informes detallados sobre las ventas, la ocupación de los transportes, y la gestión de encomiendas.
- La plataforma será accesible desde diferentes tipos de dispositivos, incluyendo computadoras, tabletas, y smartphones, garantizando una experiencia de usuario consistente y accesible.

1.6.2. Límites

Los límites de la investigación son los siguientes:

- El sistema estará diseñado inicialmente para cubrir las operaciones de la Empresa de transportes Cali Internacional en su sede de la Terminal de Buses en La Paz.
- La integración se centrará en los sistemas internos existentes de la empresa.
- El software será compatible con las plataformas y dispositivos especificados.
- El soporte se limitará a las funcionalidades implementadas, las actualizaciones o desarrollos adicionales quedarán para fases futuras.

1.7. IMPORTANCIA DEL ESTUDIO

La importancia del estudio del proyecto radica en la necesidad de modernizar los procesos operativos de empresas de transporte y logística, especialmente en un entorno donde la eficiencia y la rapidez son factores clave para la competitividad. En la actualidad, muchas empresas en este sector aún dependen de sistemas manuales o desactualizados que ralentizan las operaciones, sino que también incrementan el riesgo de errores humanos, afectando directamente la calidad del

servicio ofrecido al cliente. Este proyecto, por lo tanto, no solo aborda una necesidad tecnológica, sino que también busca mejorar la experiencia del cliente al ofrecerle un servicio más ágil y fiable.

Desde una perspectiva social, este estudio tiene una importancia significativa al contribuir al avance tecnológico en un sector que afecta directamente a un gran número de personas. Al mejorar la eficiencia y la precisión en la venta de pasajes y el envío de encomiendas, se generan beneficios directos no solo para la empresa, sino también para los usuarios finales, quienes experimentarán un servicio más confiable y accesible. Esto, a su vez, puede fomentar una mayor confianza en los servicios digitales en general, impulsando el uso de la tecnología en otras áreas de la vida diaria.

Finalmente, la importancia de este estudio también reside en su capacidad para servir como modelo para futuras implementaciones tecnológicas en empresas similares. La metodología empleada, así como los desafíos superados durante el desarrollo del software, pueden ofrecer valiosas lecciones para otros proyectos dentro del sector, promoviendo un enfoque más sistemático y eficiente en la adopción de tecnologías de la información en la industria del transporte y la logística.

2 MARCO TEÓRICO

2.1. LOGÍSTICA DEL TRANSPORTE DE PASAJEROS

2.1.1. Logística

“Logística es planificar, operar, controlar y detectar oportunidades de mejora del proceso de flujo de materiales (insumos, productos), servicios, información y dinero. Es la función que normalmente opera como nexo entre las fuentes de aprovisionamiento y suministro y el cliente final o la distribución. Su objetivo es satisfacer permanentemente la demanda en cuanto a cantidad, oportunidad y calidad al menor costo posible para la empresa.”(Carro & González Gómez, 2013)

2.1.2. Transporte

Según Koch (2001): “El concepto de “transporte” hace referencia al traslado de personas y mercancías de un lugar a otro por diversas razones en el menor tiempo posible. En el caso de las personas, destacan los motivos laborales, de estudio o de satisfacción de otras necesidades como el ocio, el acceso a servicios de salud, entre otros; en el caso de las mercancías, la necesidad de producción de bienes industriales y de consumo y la posterior comercialización de estos hacen del proceso de transporte un elemento central.” Por su parte en la Ley General de Transporte (2011): “Se denomina transporte al traslado de un lugar a otro de personas y carga.”

El transporte es un componente de la logística, que se refiere al conjunto de recursos y estrategias utilizados para organizar un servicio o administrar una empresa. En el ámbito comercial, la logística se relaciona con el envío de productos al lugar adecuado, en el momento correcto y bajo las condiciones necesarias. Por lo tanto, el transporte de mercancías es una parte integral de la

logística. El propósito de una empresa es asegurar que la distribución y venta de sus productos se realice de manera eficiente y al menor costo posible. En este contexto, el transporte abarca tanto los vehículos como las infraestructuras asociadas, como camiones, barcos, trenes de carga, carreteras y puertos.

También existen dos tipos de transporte, el público y el privado.

Se habla de transporte público, para hacer referencia a los autobuses, trenes y otras unidades móviles que sirven para la movilización de los ciudadanos de una comunidad y que está solventado y manejado por el Estado vigente. Cabe señalar que en algunos casos, dichos coches pertenecen a empresas privadas que tienen algún tipo de acuerdo con el gobierno y han asumido la responsabilidad de brindar un servicio determinado a la comunidad. Resulta importante señalar que esta clase de transporte no tiene como propósito la generación de ganancias, sino que debe cumplir con un fin social y ser útil para la comunidad. Por ejemplo: “Los transportes públicos están colapsados y requieren de mayores inversiones para poder satisfacer las necesidades de la población”.

El transporte privado, en cambio, es el que pertenece a individuos o empresas particulares. En este caso los responsables de la manutención de dichos vehículos son sus dueños, al igual que serán quienes respondan por ellos en caso de accidente.

2.1.3. Pasajero

En la Ley Municipal de Transporte y Tránsito Urbano' (2012) en su artículo 59 se define a los usuarios o pasajeros como “Las personas naturales o jurídicas que utilizan un vehículo del servicio público o privado de transporte, para trasladarse de un origen a un destino a cambio de una tarifa establecida o remuneración convenida, son considerados usuarios o pasajeros en el marco de la presente Ley Municipal.”

2.1.4. Sistema de transporte

De acuerdo con García (2016): “Un sistema de transporte desde la perspectiva informática es una red interconectada de componentes físicos y digitales que utiliza tecnologías avanzadas de información y comunicación para optimizar el flujo de personas y mercancías. Incluye sistemas de gestión de tráfico, planificación de rutas en tiempo real, control de flotas y plataformas de información al usuario, todos ellos integrados mediante software especializado y bases de datos.”

2.2. RESERVA Y VENTA DE PASAJES

2.2.1. Proceso de reserva y venta de pasajes

El proceso de reserva y venta de pasajes es un componente fundamental en la operación de cualquier empresa de transporte de pasajeros. Según Aparicio (2013), este proceso implica una serie de pasos secuenciales que permiten al cliente asegurar su lugar en un viaje específico. Tradicionalmente, las empresas de transporte han utilizado diversos canales para la reserva y venta, incluyendo puntos de venta físicos, call centers, y más recientemente, plataformas en línea.

2.2.2. Emisión y gestión de boletos

La emisión y gestión de boletos es un proceso crítico que ha evolucionado significativamente con la tecnología. Agenjo y Mateu (2008) describen dos tipos principales de boletos en el transporte moderno: los electrónicos (e-tickets) y los impresos tradicionales. Independientemente del formato, los boletos deben contener información esencial como datos del pasajero, detalles del viaje, asiento asignado y un método de validación.

El proceso de emisión, según García (2016), debe ser ágil y estar vinculado directamente con la confirmación del pago. La gestión eficiente de boletos implica un sistema robusto de validación, ya sea en terminales o a bordo de los vehículos, así como la capacidad de reimpresión en caso de pérdida. Además, como señala Robusté Antón (2005), el seguimiento y registro de los boletos emitidos es importante para el control operativo y financiero de la empresa de transporte.

2.2.3. Cambios y cancelaciones

La gestión de cambios y cancelaciones es un aspecto delicado que requiere un equilibrio entre la flexibilidad para los clientes y la protección de los intereses de la empresa. Según Tejero (2015), las políticas de cambios y cancelaciones deben ser claras, especificando plazos permitidos y cargos aplicables.

El proceso de solicitud de cambios, como describe Castellanos Ramírez (2015), implica la verificación de disponibilidad para nuevas fechas y el cálculo de diferencias tarifarias. En cuanto a las cancelaciones, el sistema debe determinar el monto del reembolso según la política establecida. La gestión de reembolsos, de acuerdo con García (2016), debe ser eficiente y transparente, ofreciendo múltiples métodos según las preferencias del cliente.

Un aspecto importante señalado por Rivera et al. (2002) es la reasignación de asientos liberados, lo que permite optimizar la ocupación de los vehículos. Además, el registro detallado de cambios y cancelaciones proporciona datos valiosos para el análisis de patrones de comportamiento de los clientes y la mejora continua de los servicios.

En conjunto, estos procesos de reserva, emisión de boletos y gestión de cambios y cancelaciones forman la columna vertebral de la operación de venta de pasajes en una empresa de transporte. Su eficiente implementación y gestión son cruciales para la satisfacción del cliente y el éxito operativo de la empresa.

2.3. LOGÍSTICA DE ENVÍO DE ENCOMIENDAS

2.3.1. Encomienda

La encomienda es el objeto o paquete que se transporta de un punto a otro a través de un servicio de mensajería o transporte. Según Lambert y Stock (2001), “una encomienda representa una unidad logística que debe ser gestionada y tratada como tal, garantizando su integridad desde el origen hasta el destino final”. En el contexto del transporte de encomiendas, es fundamental contar

con un sistema que permita la correcta identificación, seguimiento y gestión de cada paquete.

García (2016) destaca que el concepto de encomienda ha evolucionado con el tiempo, especialmente con el auge del comercio electrónico. Actualmente, las empresas de transporte deben estar preparadas para manejar una amplia gama de artículos, desde documentos hasta productos perecederos, cada uno con sus propios requisitos de manipulación y transporte. Esta diversidad exige sistemas flexibles y adaptables que puedan responder a las necesidades cambiantes de los clientes y del mercado.

2.3.2. Proceso de recepción

El proceso de recepción es la primera etapa en la gestión de encomiendas, donde se verifica la información proporcionada por el remitente, se inspecciona el paquete y se registran los detalles necesarios para su envío. De acuerdo con García (2016), este proceso implica la verificación inicial del paquete, el registro de información relevante y la asignación de un identificador único. Escudero Serrano (2019) enfatiza la importancia de este paso para garantizar la trazabilidad y el manejo adecuado de la encomienda durante todo su trayecto.

Con el avance de las tecnologías, muchas empresas han implementado sistemas que automatizan el proceso de recepción, permitiendo la digitalización de la información desde el inicio del proceso logístico. Esto facilita un flujo continuo de datos entre las distintas etapas del envío, reduciendo errores humanos y optimizando los tiempos de procesamiento.

2.3.3. Clasificación de encomiendas

La clasificación de encomiendas es un paso fundamental para optimizar el proceso de envío. Según i Cos, De Navascués et al. (2001), las encomiendas se pueden clasificar según diversos criterios, como tamaño, peso, destino, urgencia o tipo de contenido. Tejero (2015) señala que una clasificación eficiente permite una mejor planificación de rutas y utilización de los espacios de carga.

Escudero Serrano (2019) agrega que la clasificación también juega un papel crucial en la

priorización de los envíos y la asignación de recursos. Por ejemplo, las encomiendas urgentes o perecederas pueden requerir un tratamiento especial y rutas más directas. Además, una clasificación adecuada facilita el cumplimiento de regulaciones específicas, como las relacionadas con el transporte de mercancías peligrosas o artículos restringidos.

2.3.4. Embalaje y etiquetado

El embalaje y etiquetado son procesos críticos para garantizar la integridad y correcta identificación de las encomiendas. Castellanos Ramírez (2015) destaca que el embalaje debe proporcionar protección adecuada según la naturaleza del contenido y las condiciones del transporte. Esto puede incluir el uso de materiales de amortiguación, envoltorios impermeables o contenedores especializados para artículos frágiles o sensibles a la temperatura.

El etiquetado, por su parte, es igualmente importante, ya que proporciona la información necesaria para la correcta identificación del paquete. Esta información incluye los datos del remitente y del destinatario, instrucciones especiales de manejo y, en muchos casos, códigos de seguimiento que permiten rastrear el paquete en tiempo real. Ballou (2004) menciona que un etiquetado claro y preciso es esencial para evitar errores en la clasificación y garantizar que el paquete llegue a su destino de manera eficiente. Las tecnologías modernas, como los códigos QR, también han facilitado este proceso, permitiendo una gestión más ágil de los envíos.

2.3.5. Entrega al destinatario

La entrega al destinatario es la fase final en la logística de encomiendas, y su éxito depende en gran medida de la eficiencia de los pasos previos. Según García (2016), este proceso implica la verificación de la identidad del destinatario, la obtención de una firma de recepción y la resolución de cualquier incidencia que pueda surgir. Tejero (2015) resalta la importancia de la puntualidad y la integridad de la entrega como factores clave en la satisfacción del cliente y la reputación de la empresa de transporte.

Sin embargo, la entrega puede enfrentarse a diversos retos, como la ausencia del destinatario en el momento de la entrega o dificultades de acceso en ciertas áreas geográficas. Para contrarrestar estos problemas, empresas líderes en logística han implementado políticas de entrega flexible, que permiten a los clientes seleccionar franjas horarias de entrega, puntos de recogida o reprogramar la entrega.

2.4. GESTIÓN DE BUSES

2.4.1. Asignación de rutas

La asignación de rutas es uno de los procesos clave en la gestión de transporte de pasajeros, ya que su correcta planificación puede mejorar significativamente la eficiencia operativa y la satisfacción del cliente. Según Molinero y Arellano (2005), este proceso implica la determinación de los recorridos óptimos que deben seguir los vehículos para cubrir la demanda de pasajeros de manera eficiente. Los autores señalan que una asignación de rutas efectiva debe considerar factores como la densidad poblacional, los patrones de viaje de los usuarios, la infraestructura vial disponible y las restricciones operativas de la empresa.

También Ballou (2004) destaca que la planificación de rutas tiene como objetivo maximizar la eficiencia operativa mediante la reducción de distancias y tiempos muertos. Ballou resalta que una asignación óptima de rutas no solo mejora la utilización de los vehículos, sino que también contribuye a una mejor satisfacción del cliente, al reducir los tiempos de entrega y los costos asociados.

2.4.2. Programación y asignación de conductores

La programación y asignación de conductores es un componente esencial en la gestión de buses, que busca optimizar el uso del recurso humano y garantizar la operación eficiente de los servicios. Según Mauttome et al. (2002), este proceso implica la optimización de recursos humanos para garantizar la cobertura eficiente de las rutas y horarios establecidos. Ibarra-Rojas y Rios-Solis

(2012) enfatizan la importancia de sincronizar los horarios de los conductores con los tiempos de llegada y salida de los vehículos, lo que no solo mejora la puntualidad del servicio, sino que también reduce los tiempos de espera para los pasajeros. Esta sincronización debe tener en cuenta factores como los períodos de descanso obligatorios, los cambios de turno y las variaciones en la demanda de pasajeros a lo largo del día.

Por otro lado, Molinero y Arellano (2005) añaden que la programación debe considerar no solo la eficiencia operativa, sino también el bienestar de los conductores, incluyendo aspectos como la fatiga, las preferencias personales y el equilibrio entre trabajo y vida personal. Esto subraya la necesidad de un enfoque holístico que balancee las necesidades operativas con las consideraciones humanas en la gestión del personal de transporte público.

2.5. MARCO LEGAL Y NORMATIVO

2.5.1. Autoridad de regulación y fiscalización de telecomunicaciones y transportes

La Autoridad de Regulación y Fiscalización de Telecomunicaciones y Transportes (ATT), es una institución pública, técnica y operativa, con personalidad jurídica y patrimonio propio, independencia administrativa, financiera, legal y técnica, transitoriamente dependiente del Ministerio de Obras Públicas, Servicios y Vivienda, de acuerdo a la Ley 164.

La ATT, tiene como objetivo regular las actividades que realizan las personas naturales y jurídicas, privadas, comunitarias, públicas, mixtas y cooperativas en los sectores de telecomunicaciones, transportes, tics y servicios postales, asegurando que se garantice los intereses y derechos de los consumidores y usuarios, los intereses del país y el desarrollo del sector, promoviendo la economía plural e inclusiva prevista en CPE, y brindando posibilidades para que más habitantes puedan acceder a los servicios (ATT, 2017).

2.5.2. Ley general del transporte

"Ley Nro. 165 de transporte, El Sistema de Transporte Integral - STI, en todo el Estado Plurinacional de Bolivia, se rige por la Constitución Política del Estado, los Tratados, Convenios e Instrumentos Internacionales, la Ley Marco de Autonomías y Descentralización, la presente Ley, normas sectoriales y otras normas específicas del ordenamiento jurídico del Estado Plurinacional."

(Ley General de Transporte, 2011)

Sus principios son:

Accesibilidad. Todas las usuarias y usuarios podrán acceder al Sistema de Transporte Integral - STI, por el medio y modalidad que escojan, los mismos que deben contar con facilidades de acceso y estar en condiciones de equidad, calidad y seguridad.

Calidad. El Sistema de Transporte Integral - STI, debe proveer un servicio en conformidad a los requisitos y estándares que garanticen un nivel de servicio adecuado de bienestar, eficiencia y eficacia, de acuerdo a la contraprestación autorizada.

Continuidad. El Sistema de Transporte Integral - STI, debe funcionar de manera permanente, regular y continua.

Eficacia. El servicio de transporte debe cumplir el propósito para el cual fue convenido.

Eficiencia. El Sistema de Transporte Integral - STI, debe prestar servicios en condiciones que garanticen el menor costo operacional y tiempo posible, contemplando un nivel de equidad, calidad y seguridad.

Participación y control social. Se garantizará y facilitará la participación y control social sobre la gestión pública por parte de la sociedad civil organizada.

Seguridad. El Sistema de Transporte Integral - STI, debe prestar servicios en condiciones que garanticen la integridad de personas y carga durante el traslado del lugar de origen al lugar de destino.

Sostenibilidad. El sistema de transporte debe prestar servicios que garanticen el menor impacto sobre la salud y el medio ambiente local y global. En el corto, mediano y largo plazo, sin comprometer el desarrollo de futuras generaciones.

Transparencia. Se garantiza la transparencia en el Sistema de Transporte Integral - STI.

Universalidad. Todas las usuarias y usuarios sin distinción alguna, tienen el derecho de utilizar el Sistema de Transporte Integral - STI, para su libre movilidad.

2.5.3. Reglamento regulatorio de transporte terrestre de pasajeros y carga

Resolución Ministerial N° 266 del 14 de agosto de 2017, emitida por el Ministerio de Obras Públicas, Servicios y Vivienda.

ARTÍCULO 1 (OBJETO.-) Reglamentar los aspectos regulatorios del servicio de transporte en la modalidad terrestre de pasajeros y carga en aplicación a la Ley N° 165 General de Transporte.

SECCIÓN III: DE LA INFORMACIÓN AL USUARIO Y LAS CONDICIONES PARA EL VIAJE

ARTÍCULO 14. (INFORMACIÓN AL USUARIO).

I. El operador, previamente a la compra del pasaje, debe informar al usuario, de forma clara, precisa y oportuna, sobre lo siguiente:

- a) Destinos, itinerario, rutas, tiempo de viaje, hora de salida y hora estimada de arribo del vehículo al lugar de destino.
- b) Capacidad del vehículo y asientos disponibles de acuerdo a numeración.
- c) Tarifas aprobadas por la Autoridad Regulatoria de acuerdo a las características del servicio de transporte terrestre interdepartamental.
- d) Peso, volumen y cantidad permitidos para el transporte de equipaje facturado y de mano, restricciones del equipaje considerado peligroso y/o nocivo a la salud.
- e) Condiciones del transporte requisitos y documentación necesaria para el viaje al país de destino; condiciones de reembolso en caso que el usuario desista del viaje; entre otros.

II. El operador tiene la obligación de informar al pasajero antes del inicio y durante el viaje lo siguiente:

- a) Características del servicio: destino, fecha, hora del viaje, número y categoría del bus y número de carril.
- b) Derechos y obligaciones de los usuarios.
- c) Demoras y/o cancelaciones, nueva hora de salida u otros aspectos relacionados al viaje.
- d) Rutas alternativas o desvíos y demoras del viaje por caso fortuito o fuerza mayor durante el viaje.

ARTÍCULO 15. (ACREDITACIÓN DE INFORMACIÓN PROPORCIONADA AL USUARIO).

El operador deberá aplicar diferentes mecanismos para brindar información confiable al usuario al momento de la compra del boleto, antes y durante la ejecución del servicio, y al momento de la entrega de carga y/o encomiendas, debiendo cumplir con esta obligación y acreditar tal situación.

ARTÍCULO 16. (MECANISMOS DE INFORMACIÓN).

En los monitores internos del vehículo del operador deben difundir material audiovisual, que informe y dé a conocer a los usuarios sus derechos, obligaciones y prohibiciones.

SECCIÓN IV: DEMORA, INTERRUPCIÓN Y/O CANCELACIÓN DEL VIAJE

ARTÍCULO 23. (ATENCIÓNES AL PASAJERO POR CAUSAS ATRIBUIBLES AL OPERADOR).

I. En casos de cancelaciones, interrupciones, demoras, duplicidad de boletos o ante cualquier otro evento que sea imputable al operador, éste deberá:

- a) Demoras: si la demora fuera mayor a los 15 minutos, informar a los pasajeros la causa y la nueva hora de salida del bus.

Si la demora fuera mayor a una (1) hora, deberán poner a disposición de los pasajeros otro bus de la misma categoría o acordar el transporte de éstos con otro operador.

- b) Cancelación: si el viaje es cancelado, deberá embarcar al pasajero en el siguiente bus disponible o de otro operador de idéntica categoría lo más rápidamente posible o en una fecha posterior que convenga al pasajero.
- c) Interrupción del viaje: si el viaje es interrumpido después de iniciado por fallas mecánicas y/o accidentes y no pueda continuar el recorrido, el operador se comunicará con su Centro de Contingencias e informará a los pasajeros las medidas a adoptar para auxiliarlos y el tiempo estimado en que llegará el auxilio para continuar el viaje, a fin de que el pasajero tome una determinación: esperar el bus de auxilio o tomar otro medio de transporte para llegar a destino.
- d) Duplicación de asientos: ante la venta de dos o más boletos para un solo espacio, deberá asignar al pasajero otro espacio en el bus o embarcarlo en otro bus de igual categoría. Solo a solicitud del usuario deberá rembolsar el 100 por ciento del valor del pasaje.
- e) Anticipación del viaje: en caso que el operador anticipa el viaje sin avisar al pasajero, deberá proporcionarle un espacio en el siguiente viaje que le resulte conveniente a su destino final. En estos casos, el pasajero no pagará ningún excedente si el nuevo espacio correspondiera a una tarifa superior.

II. En todos los casos, el operador está terminantemente prohibido de recurrir directamente a la devolución; deberá agotar todas las posibilidades para cumplir con el contrato; optará por la devolución únicamente si el pasajero lo requiere.

ARTÍCULO 56.- (PROHIBICIONES DEL CONDUCTOR)

Queda terminantemente prohibido a los conductores de las unidades de servicio de transporte automotor público terrestre en corresponsabilidad con el operador, lo siguiente:

- a) Presentarse al trabajo con síntomas de haber ingerido bebidas alcohólicas, o bajo la influencia de sustancias psicotrópicas o ingerirlas en horas de trabajo.
- b) Transportar pasajeros en los pasillos, buzones y cabina del bus.
- c) Realizar paradas no programadas o desviar el vehículo de su recorrido oficial, a menos que fuera instrucción de las autoridades correspondientes.
- d) Abandonar el vehículo en plena carretera.
- e) Efectuar paradas no autorizadas cuya duración exceda los 20 minutos.
- f) Agredir físicamente o psicológicamente a los usuarios, personal de la Autoridad Competente, Policía Boliviana u operador del servicio de Terminal Terrestre.
- g) Usar radios o parlantes con alto volumen de sonido.

2.6. INGENIERÍA DE SOFTWARE

La Ingeniería de Software es una disciplina dentro de la informática encargada de la aplicación de un enfoque sistemático y metodológico al desarrollo, operación y mantenimiento de sistemas de software de alta calidad. Este campo surgió en respuesta a la creciente complejidad de los sistemas informáticos y a la necesidad de garantizar que los programas y aplicaciones sean confiables, eficientes y ajustados a los requisitos del cliente.

Según Sommerville (2011), la ingeniería de software implica la aplicación de principios científicos y de ingeniería en el diseño, desarrollo, prueba y mantenimiento de software. Este enfoque abarca desde el análisis de requisitos, la creación de modelos, la implementación del software, hasta la prueba y verificación del mismo. La clave de la ingeniería de software radica en estructurar cada etapa del proceso de desarrollo de manera que permita crear sistemas que sean fácilmente mantenibles y adaptables a los cambios.

Además, Pressman (2010) enfatiza que la ingeniería de software no solo se ocupa del código, sino que integra el manejo de proyectos, la gestión de riesgos y la implementación de técnicas para

la verificación de la calidad del producto. Esta disciplina ayuda a gestionar el ciclo de vida completo del software, desde el diseño hasta el mantenimiento, lo cual es esencial en proyectos complejos que requieren altos estándares de calidad y funcionalidad.

2.7. MODELO EN CASCADA

El modelo en cascada es uno de los enfoques tradicionales más conocidos en la ingeniería de software. Se originó a partir de la necesidad de estructurar los proyectos de desarrollo de manera más controlada y predecible, lo cual fue posible gracias a su enfoque lineal y secuencial. Según Sommerville (2011), el modelo en cascada divide el proceso de desarrollo en una serie de fases bien definidas, donde cada etapa tiene que completarse antes de pasar a la siguiente. Esto proporciona un marco organizado y claro para proyectos con requisitos claramente establecidos desde el inicio.

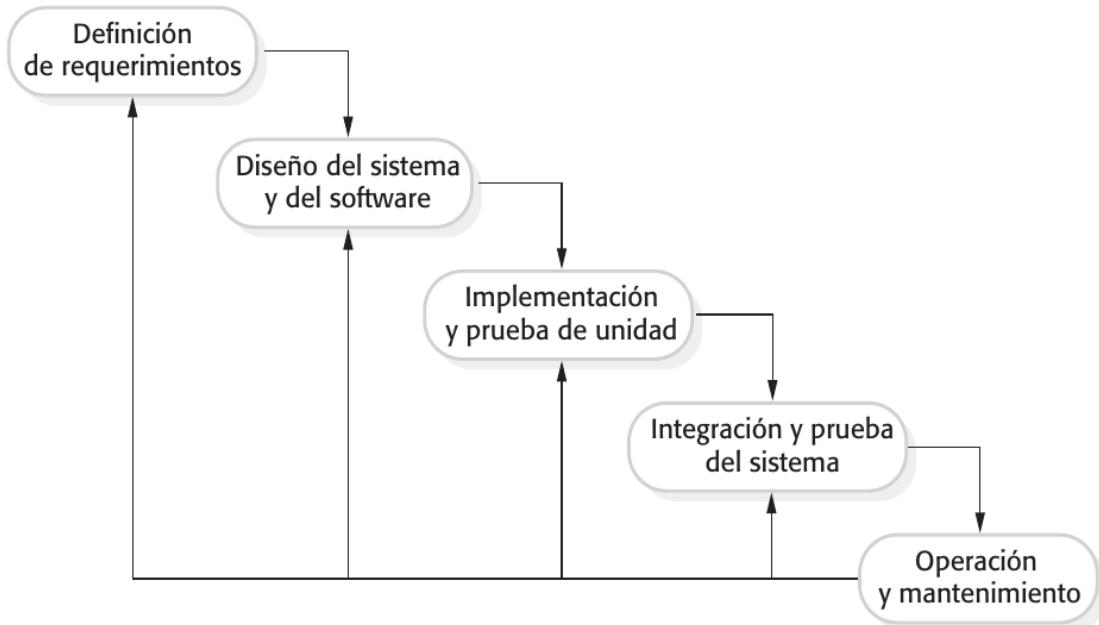
El modelo ha sido ampliamente utilizado en proyectos con entornos más controlados, como aquellos relacionados con sistemas críticos y empresariales, donde la precisión y la estabilidad son fundamentales. De acuerdo con Pressman (2010), una de las principales ventajas del modelo en cascada es su capacidad para proporcionar una excelente visibilidad de los avances, ya que permite una planificación minuciosa y detallada desde las etapas iniciales. Esto lo convierte en una opción favorable cuando los requisitos no cambian con frecuencia y el desarrollo debe seguir una ruta específica sin grandes variaciones.

Este modelo, sigue siendo un método relevante para proyectos que requieren control estricto y documentación exhaustiva, como los del ámbito gubernamental o grandes corporaciones, donde los cambios son poco frecuentes (Sommerville, 2011).

Las diferentes etapas del modelo en cascada, que abarcan desde el levantamiento de requisitos hasta el mantenimiento del sistema, estas se ilustran en la figura 2.1, donde se visualiza el flujo secuencial característico de este enfoque.

Figura 2.1

Etapas del Modelo en Cascada.



Nota. Obtenido de Sommerville (2011).

2.7.1. Análisis y definición de requerimientos

En esta fase inicial, se establecen los servicios, restricciones y objetivos del sistema en consulta con los usuarios. Todos estos elementos se definen en detalle y sirven como especificación del sistema (Sommerville, 2011).

Esta fase implicaría identificar las necesidades de la empresa de transporte, como la gestión de la venta de boletos, el registro de encomiendas y la asignación de rutas de buses. Se debe trabajar con los principales interesados, como administradores y empleados de la empresa, para definir claramente los procesos actuales y los problemas que el software debería resolver.

2.7.2. Diseño del sistema y del software

Esta etapa implica la transformación de requisitos en una representación del software que pueda ser evaluada por su calidad antes de comenzar la codificación, se establecen la arquitectura

general del sistema, la estructura de los datos, las interfaces principales y los algoritmos que proporcionarán la funcionalidad deseada, creando así un plan detallado para la construcción del producto (Pressman, 2010).

2.7.3. Implementación y prueba de unidad

Durante esta etapa, el diseño del software se lleva a cabo como un conjunto o unidades de programas. La prueba de unidades implica verificar que cada una cumpla con su especificación (Sommerville, 2011).

En esta etapa, se lleva a cabo la programación del software utilizando herramientas adecuadas como lenguajes de programación, así como frameworks que permitan la integración de las funcionalidades necesarias.

2.7.4. Integración y prueba de sistema

Es un proceso incremental donde los diferentes componentes o módulos del sistema se combinan y prueban como un sistema completo. Es una fase crítica que tiene dos objetivos distintos pero relacionados: por un lado, integrar los diferentes módulos o componentes en un sistema funcional completo, y por otro, verificar que el software satisface los requerimientos especificados y detectar errores de interacciones entre componentes, esta fase no solo se trata de encontrar errores en el software, sino también de demostrar al cliente que el sistema cumple con sus necesidades (Sommerville, 2011).

En esta fase, se realizarían pruebas funcionales para garantizar que los procesos de venta de boletos, la gestión de encomiendas y la asignación de buses operen correctamente. Las pruebas incluirían la simulación de transacciones, el seguimiento de encomiendas y la verificación de la correcta asignación de rutas y conductores.

2.7.5. Operación y mantenimiento

Normalmente, esta es la fase más larga del ciclo de vida. El sistema se instala y se pone en uso práctico. El mantenimiento implica corregir errores no descubiertos en las etapas anteriores del ciclo de vida, mejorar la implementación de las unidades del sistema y resaltar los servicios del sistema una vez que se descubren nuevos requerimientos (Sommerville, 2011).

2.8. INGENIERÍA DE REQUERIMIENTOS

Algunas definiciones de la ingeniería de requerimientos son:

Según Pressman (2010), la ingeniería de requerimientos proporciona el mecanismo apropiado para entender lo que desea el cliente, analizar las necesidades, evaluar la factibilidad, negociar una solución razonable, especificar la solución sin ambigüedades, validar la especificación y administrar los requerimientos a medida que se transforman en un sistema funcional.

De acuerdo a Sommerville (2011), la ingeniería de requerimientos establece la base para todas las fases posteriores del desarrollo de software, lo que la convierte en un proceso fundamental para garantizar el éxito del proyecto.

Dadas las definiciones anteriores podemos decir que la ingeniería de requerimientos es: una parte importante del desarrollo de software porque se puede utilizar para definir y comprender las expectativas y necesidades de las personas que utilizan el sistema. Este proceso incluye diversas actividades enfocadas en la investigación, documentación, uso y gestión de requisitos de software para garantizar que el producto final satisfaga las necesidades del negocio y de los clientes, lo que también se clasifica en requerimientos funcionales y requerimientos no funcionales.

2.8.1. Requerimientos funcionales

Los requerimientos funcionales son descripciones de las funciones y capacidades específicas que el sistema debe realizar, es decir, las tareas concretas que el software debe ejecutar para cumplir con los objetivos de negocio. Según Sommerville (2011), estos requerimientos "definen los servicios y

funciones del sistema, que son visibles y utilizables para los usuarios o clientes". Ejemplos comunes incluyen autenticación de usuario, procesamiento de transacciones o la generación de reportes.

2.8.2. Requerimientos no funcionales

Por otro lado, los requerimientos no funcionales se refieren a las cualidades y restricciones del sistema, tales como rendimiento, seguridad, y facilidad de uso. Estos determinan cómo debe comportarse el sistema y qué condiciones debe cumplir para ser eficiente y confiable, sin especificar funciones concretas. Para Pressman (2010), estos requerimientos son esenciales porque "determinan la calidad global del software" abarcando aspectos como tiempos de respuesta y compatibilidad. Sommerville (2011) también destaca que los requerimientos no funcionales suelen influir en la arquitectura y diseño general del sistema más que en sus funciones específicas.

2.8.3. Actividades de la ingeniería de requerimientos

Hay múltiples enfoques para definir las actividades de la ingeniería de requerimientos y la elección de un modelo particular a menudo depende del tamaño y la complejidad del software en desarrollo. Esto da lugar a la posibilidad de que se implementen diferentes actividades, las cuales pueden variar tanto en su cantidad como en su naturaleza.

A continuación, se detallan las actividades que forman parte de la ingeniería de requerimientos para el presente proyecto:

Paso 1: Levantamiento de requerimientos

Paso 2: Análisis de requerimientos

Paso 3: Especificación de requerimientos

Paso 4: Validación de requerimientos

Paso 5: Gestión de requerimientos

Paso 1: Levantamiento de requerimientos

En este primer paso se busca identificar, comprender y documentar las necesidades y

expectativas de los usuarios y las partes interesadas respecto al sistema a desarrollar. Esta actividad es el primer paso en el ciclo de vida del software y se fundamenta en la interacción continua entre los desarrolladores, los usuarios y otros actores del proyecto. El objetivo principal es asegurar que el sistema refleje adecuadamente las funciones y características que se requieren, permitiendo que su implementación sea exitosa.

La complejidad de esta fase radica en la diversidad de las fuentes de información y la posibilidad de que existan diferencias entre lo que los usuarios creen que necesitan y lo que realmente es necesario para el negocio. Según Sommerville (2011), uno de los mayores desafíos es interpretar correctamente las necesidades de los usuarios, ya que estos a menudo pueden tener una visión limitada del sistema o no ser conscientes de los detalles técnicos que implican sus solicitudes. Por ello, es fundamental contar con técnicas efectivas para recopilar esta información, como entrevistas, observación directa, cuestionarios y talleres participativos.

Algunas de las técnicas más comunes y efectivas para el levantamiento de requerimientos son:

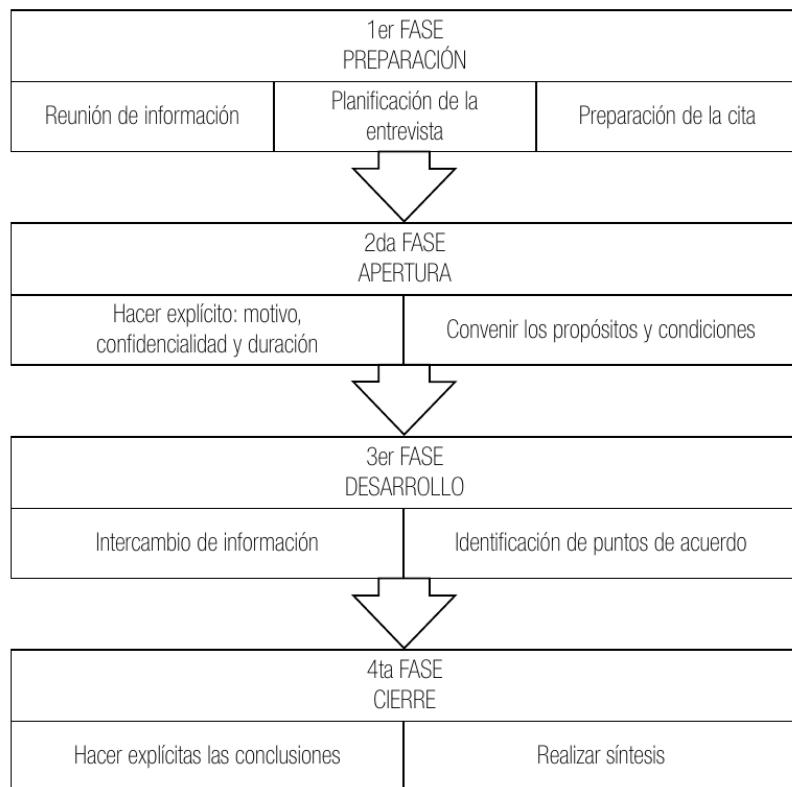
Entrevistas: Son útiles para obtener información detallada y directa de los usuarios y otros interesados. Pueden ser estructuradas, semiestructuradas o no estructuradas, dependiendo de la profundidad de la información que se deseé obtener. Las entrevistas permiten aclarar aspectos complejos y detectar necesidades particulares de cada usuario o área de la organización.

De acuerdo a Díaz-Bravo et al. (2013) en base a la clasificación antes mencionada se identifica que cada tipo de entrevista tiene su peculiaridad, sin embargo, en el momento de su desarrollo se presentan determinados momentos homogéneos. Estos momentos o fases de la entrevista los vemos en la figura 2.2.

En conclusión Díaz-Bravo et al. (2013) aclara que la entrevista es uno más de los instrumentos cuyo propósito es recabar datos, pero debido a su flexibilidad permite obtener información más

Figura 2.2

Fases de la entrevista.



Nota: Imagen tomada de Díaz-Bravo et al. (2013), *La entrevista, recurso flexible*.

profunda, detallada, que incluso el entrevistado y entrevistador no tenían identificada, ya que se adapta al contexto y a las características del entrevistado. Es valiosa en el campo de la investigación y más aún cuando se utiliza en estudios de tipo mixto como una visión complementaria del enfoque cuantitativo.

Talleres de trabajo: Se utilizan para reunir a diferentes partes interesadas en un mismo espacio, facilitando la discusión colaborativa sobre los requerimientos. Estos talleres ayudan a resolver posibles conflictos entre usuarios y a consolidar una visión conjunta del sistema a desarrollar.

Observación directa: Este enfoque implica observar a los usuarios en su entorno laboral para comprender mejor los procesos actuales, las dificultades que enfrentan y cómo un sistema

nuevo podría mejorar su desempeño. La observación es particularmente útil cuando los usuarios no pueden articular claramente sus necesidades o cuando los problemas son inherentes a las tareas que realizan.

Cuestionarios: Son una buena opción para obtener información de un amplio número de usuarios cuando no es posible realizar entrevistas individuales. Los cuestionarios permiten estructurar preguntas específicas sobre las necesidades del sistema, aunque son menos flexibles en cuanto a la obtención de respuestas detalladas.

Paso 2: Análisis de requerimientos

El análisis de requerimientos es donde se busca definir las necesidades y expectativas de los usuarios. Esta fase se enfoca en transformar las necesidades identificadas durante el levantamiento de requerimientos en especificaciones claras y comprensibles. Según Sommerville (2011), el análisis de requerimientos no solo implica la recopilación de datos, sino también su organización y priorización para asegurar que el producto final cumpla con las expectativas del cliente. Este proceso requiere un enfoque sistemático y la participación activa de todos los involucrados, lo que facilita la identificación de posibles problemas y la aclaración de dudas desde el inicio del proyecto.

Durante el análisis de requerimientos, se documenta estos requerimientos, esto se realiza mediante el uso de herramientas de modelado, como diagramas de casos de uso, que permiten visualizar las interacciones entre los usuarios y el sistema. Además, el análisis ayuda a identificar requerimientos funcionales y no funcionales. La correcta identificación y documentación de estos requerimientos son fundamentales para evitar malosentendidos y garantizar que el desarrollo se alinee con las necesidades del cliente.

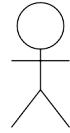
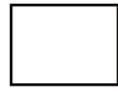
El **diagrama de casos de uso** es un diagrama uml (lenguaje de modelado unificado) que, a través de una representación gráfica del sistema, permite modelar los requerimientos funcionales de una aplicación, mostrándolos desde el punto de vista del usuario. Estos diagramas tienen

dos funciones importantes: 1) capturar los requisitos funcionales del sistema y 2) simplificar la construcción de los modelos de objetos. Estos diagramas son útiles también para que los usuarios realicen retroalimentación en la especificación de requerimientos, son insumos para posteriores etapas del ciclo de vida e incluso se utilizan como herramienta para la reingeniería de procesos (Patiño Martínez, 2022).

En la tabla 2.1 se observan los elementos de un diagrama de casos de uso.

Tabla 2.1

Elementos de un Diagrama de casos de uso

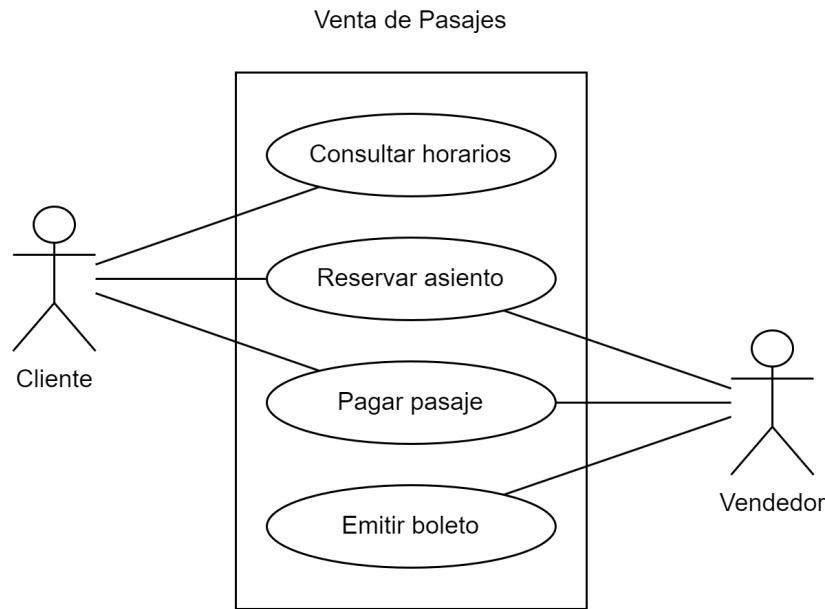
| Elemento | Definición | Símbolo |
|--------------------|---|---|
| Caso de uso | Representa el requisito funcional del sistema y describe un servicio prestado por el sistema. |  |
| Actores | Un actor representa cualquier objeto que intercambia información con el sistema. |  |
| Relaciones | Son asociaciones entre los casos de uso y los actores o entre casos de uso. |  |
| Límite del sistema | Representa la frontera entre el sistema (funcionalidades) y los usuarios (actores). |  |

En la figura 2.3 se ve un diagrama de casos de uso para un sistema de venta de pasajes que muestra la interacción entre dos actores principales (Cliente y Vendedor) con cuatro funcionalidades básicas del sistema representadas como casos de uso (Consultar horarios, Reservar asiento, Pagar pasaje y Emitir boleto), donde las líneas conectoras indican que el Cliente puede consultar horarios, reservar y pagar, mientras que el Vendedor puede reservar, cobrar y emitir el boleto, todo esto

encerrado dentro de un rectángulo que representa los límites del sistema.

Figura 2.3

Ejemplo de Diagrama de casos de uso.



Nota: Caso de uso para la venta de pasajes

El análisis también juega un papel esencial en la gestión del cambio, ya que a medida que se desarrolla el proyecto, pueden surgir nuevas necesidades o cambios en las prioridades del cliente. Por ello, es vital mantener una comunicación constante con las partes interesadas para ajustar los requerimientos de manera oportuna. A través de un análisis cuidadoso y dinámico, se pueden prevenir desviaciones en el proyecto y asegurar que el producto final se entregue dentro de los plazos y presupuestos establecidos.

Paso 3: Especificación de requerimientos

La especificación de requerimientos en el proceso de ingeniería de software, se trata de documentar y formalizar los requerimientos identificados durante la etapa de análisis. De acuerdo a Sommerville (2011), esta fase implica la creación de un documento que sirva como un contrato entre los desarrolladores y los interesados en el proyecto, asegurando que todas las partes tengan

una comprensión clara de lo que se espera del sistema. La especificación debe ser clara, completa y comprensible, lo que facilita su uso como guía para el diseño, desarrollo y pruebas del software.

La especificación se basa en un conjunto de descripciones detalladas que abordan cada requisito de manera integral y verificable. Descripción del sistema, propósito, alcance, entradas y salidas esperadas, limitaciones técnicas y operativas. La documentación debe ser clara para evitar cualquier ambigüedad o errores de implementación que puedan afectar el rendimiento del sistema (Pressman, 2010). Una buena documentación de requisitos ayuda a verificar y validar posteriormente, asegurando que el sistema esté alineado tanto con los requisitos del usuario como con los objetivos de la organización.

Es fundamental describir todos los requisitos de forma breve y estandarizada para que durante el desarrollo se pueda entender y utilizar fácilmente. Una jerarquía de documentación es una buena solución para proyectos complejos, ya que permite un seguimiento y control de los cambios más sencillo, lo que da como resultado una gestión más eficiente y organizada del desarrollo de software. Un desglose detallado de los criterios de aceptación de cada requisito para garantizar que cumplen con los estándares y objetivos establecidos, y verificarlos con precisión al final de la fase de desarrollo.

La especificación de requerimientos no es una actividad única; debe revisarse y actualizarse a medida que el proyecto avanza y se generan nuevos conocimientos. La inclusión de un proceso de revisión regular garantiza que los requerimientos sigan siendo relevantes y se adapten a cualquier cambio en las necesidades del negocio o en la tecnología disponible. Esta flexibilidad es esencial para el éxito del proyecto, ya que contribuye a la satisfacción del cliente y a la eficacia del equipo de desarrollo, minimizando el riesgo de retrabajo y asegurando una entrega más oportuna del producto final.

Paso 4: Validación de requerimientos

La validación de requerimientos es la etapa que tiene como objetivo asegurar que los requerimientos definidos son correctos, completos y viables antes de que el desarrollo del software continúe hacia etapas posteriores. Sommerville (2011) subraya que la validación es el proceso mediante el cual se comprueba que los requerimientos son consistentes con las necesidades del cliente y cumplen con los objetivos del sistema.

Entre las principales técnicas empleadas en la validación se encuentran las revisiones de los requerimientos, la construcción de prototipos, las simulaciones y el uso de casos de prueba. Estas actividades permiten identificar errores, inconsistencias o ambigüedades en la especificación, que podrían afectar la implementación del software. Según Pressman (2010), uno de los principales riesgos en esta fase es la ambigüedad de los requerimientos, que podría resultar en un producto que no satisface las necesidades del cliente. Por tanto, se debe realizar revisiones exhaustivas y asegurar que todos los involucrados tengan una comprensión clara y compartida de lo que se debe desarrollar.

La validación de requerimientos también busca garantizar que las funcionalidades propuestas sean técnicamente factibles y que el sistema final se ajuste a las restricciones legales y de seguridad. De acuerdo con Sommerville (2011), involucrar a los usuarios finales en el proceso de validación es vital, ya que son ellos quienes mejor pueden evaluar si los requerimientos reflejan sus necesidades reales. Esto puede lograrse mediante la creación de prototipos o modelos preliminares del sistema que permitan a los usuarios interactuar con el producto antes de su desarrollo completo.

Paso 5: Gestión de requerimientos

La gestión de requerimientos es una actividad que asegura que las necesidades, deseos y expectativas de las partes interesadas sean correctamente documentadas, monitoreadas y controladas a lo largo del ciclo de vida del proyecto. Esta tarea no finaliza una vez que los requerimientos han sido formalizados y especificados; en cambio, se extiende durante todo el desarrollo para asegurar

que los cambios que surjan sean controlados adecuadamente, manteniendo la coherencia con los objetivos del proyecto.

Una parte importante es el control de cambios, a lo largo del desarrollo de software, es común que los interesados soliciten modificaciones basadas en nuevas necesidades o la evolución del negocio. Aquí es donde la gestión de requerimientos entra en acción para evaluar, priorizar y aprobar o rechazar dichas modificaciones. Este proceso evita la introducción de cambios que puedan desestabilizar el proyecto, ayudando a prevenir sobrecostos o desviaciones del alcance.

Otro aspecto importante es la trazabilidad de los requerimientos, que asegura que cada requerimiento pueda ser rastreado desde su origen, a través de su implementación y pruebas, hasta su entrega final. Por su parte Sommerville (2011), la trazabilidad facilita el seguimiento de cómo se cumplen los requerimientos durante las diferentes etapas del proyecto, lo que es esencial para garantizar que el software final cumpla con las expectativas originales de los interesados.

Finalmente, la gestión de requerimientos también implica asegurar que la documentación de los requerimientos esté actualizada y disponible para todos. Mantener una buena comunicación y visibilidad sobre el estado de los requerimientos es fundamental para que los desarrolladores puedan alinearse con los objetivos del proyecto. Así, una correcta gestión de requerimientos contribuye a minimizar riesgos, optimizar recursos y garantizar el éxito del desarrollo del software.

2.9. MODELO ENTIDAD - RELACIÓN

Conforme a la segunda etapa del modelo en cascada, en su punto 2.7.2, se aborda el diseño de la base de datos, que es fundamental para la estructuración y almacenamiento eficiente de la información del sistema.

El modelo entidad-relación (ER) es una herramienta fundamental en el diseño de bases de datos, que se utiliza para representar de manera gráfica la estructura de los datos y las relaciones entre ellos. Propuesto por Peter Chen en 1976, este modelo permite a los diseñadores de bases de

datos conceptualizar y organizar la información de manera intuitiva, facilitando la comprensión de los requisitos del sistema antes de la implementación de la base de datos (Chen, 1976).

Componentes del modelo

Entidades: Se trata de cualquier objeto u elemento (real o abstracto) acerca del cual se pueda almacenar información en la base de datos. Es decir cualquier elemento informativo que tenga importancia para una base de datos y es representada por un rectángulo, dentro del cual se escribe lo que representa.

Relaciones: Es una correspondencia o asociación entre dos o más entidades. Cada relación tiene un nombre que describe su función. Las relaciones se representan gráficamente mediante rombos y su nombre aparece en el interior.

En el modelo entidad-relación, existen varios tipos de relaciones que describen cómo las entidades se conectan entre sí:

Relación uno a uno (1:1): una entidad está relacionada exactamente con una instancia de otra entidad, y viceversa.

Relación uno a muchos (1:N): una entidad está relacionada con varias instancias de otra entidad, pero una instancia de la segunda entidad está relacionada con solo una instancia de la primera entidad.

Relación muchos a uno (N:1): muchas instancias de una entidad están relacionadas con una instancia de otra entidad.

Relación muchos a muchos (N:N): muchas instancias de una entidad pueden estar relacionadas con muchas instancias de otra entidad, gestionada mediante una tabla de unión.

Atributos: Estos son en esencia descripciones de las entidades o relaciones, que describen elementos y características básicos de éstas. Son fundamentales y establecen la información que

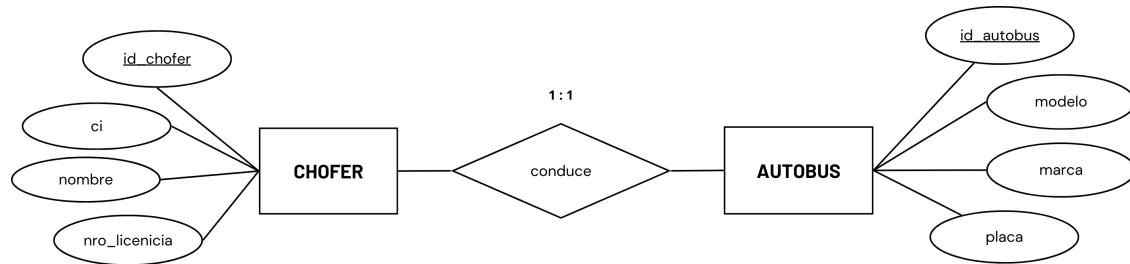
deseamos almacenar de cada objeto de la base de datos.

Clave primaria: Cada entidad tiene una clave primaria que la identifica de manera única dentro de la base de datos. La clave primaria es un atributo (o conjunto de atributos) que garantiza que no haya duplicados en la información.

A continuación en la figura 2.4 podemos observar en ejemplo básico de un modelo entidad-relación.

Figura 2.4

Ejemplo de Modelo entidad-relación.



Nota: Modelo entidad-relación de dos entidades con sus atributos.

2.10. BASE DE DATOS

La base de datos es un conjunto organizado de datos que se almacenan de manera estructurada y se gestionan a través de un sistema de gestión de bases de datos (SGBD). Su principal objetivo es facilitar el almacenamiento, recuperación y manipulación eficiente de grandes cantidades de información, asegurando la integridad y disponibilidad de los datos. La creación de una base de datos implica la definición de su estructura, basada en el modelo de datos elegido, que puede ser relacional, orientado a objetos, jerárquico, entre otros. El modelo relacional, propuesto por Edgar F. Codd en 1970, es el más utilizado hoy en día y organiza los datos en tablas compuestas por filas y columnas, conocidas como relaciones (Codd, 1970).

2.10.1. Base de datos relacional

Una base de datos relacional es un sistema de almacenamiento de información que organiza los datos en tablas interrelacionadas, conocidas como relaciones. Estas tablas están compuestas por filas (tuplas) y columnas (atributos), donde cada fila representa un registro único y cada columna un tipo de dato específico. Este modelo fue propuesto por Edgar F. Codd en 1970, en un esfuerzo por optimizar la gestión y manipulación de grandes cantidades de información, y se ha convertido en el enfoque predominante en la mayoría de las aplicaciones empresariales y científicas Codd (1970).

Los Sistemas de Gestión de Bases de Datos Relacionales son software diseñado para gestionar bases de datos relacionales.

2.10.2. Modelo relacional

El modelo relacional, para el modelado y la gestión de bases de datos, es un modelo de datos basado en la lógica de predicados y en la teoría de conjuntos. Este modelo utiliza el lenguaje SQL (Structured Query Language) para realizar consultas, actualizar y gestionar los datos. Los componentes principales del modelo relacional son: Estructura de datos: Relaciones o tablas. Operadores: Conjunto de operaciones para manipular los datos (álgebra relacional). Restricciones de integridad: Reglas para mantener la consistencia de los datos.

2.10.3. Normalización

La normalización es el proceso de organización de una base de datos para reducir la redundancia y mejorar la integridad de los datos. Las formas normales más comunes son:

Primera Forma Normal (1FN): Una tabla está en 1FN si todos sus atributos contienen valores atómicos (indivisibles) y cada entrada de la tabla tiene un único valor por columna. Es decir, no debe haber grupos repetidos ni atributos con listas de valores, como se puede ver en la figura 2.5

Figura 2.5*Ejemplo de Primera Forma Normal*

The diagram illustrates the decomposition of a first normal form (1FN) table into two second normal form (2FN) tables. It consists of three parts: a top table, a middle arrow, and a bottom section.

Top Table:

| id | Apellido | Nombre | Dirección | Ubicación |
|-----------|-----------------|---------------|------------------|------------------|
| 1 | Chavez | Juan | C/Costas | La Paz, Bolivia |
| 2 | Marca | Mayra | C/Rivas | Santiago, Chile |

Middle Arrow:

A large blue downward-pointing arrow is positioned between the top table and the bottom section.

Bottom Section:

| id | Apellido | Nombre | Dirección | Ciudad | País |
|-----------|-----------------|---------------|------------------|---------------|-------------|
| 1 | Chavez | Juan | C/Costas | La Paz | Bolivia |
| 2 | Marca | Mayra | C/Rivas | Santiago | Chile |

Nota: Tabla donde el atributo Ubicación se divide en dos atributos que son Ciudad y País.

Segunda Forma Normal (2FN): Para estar en 2FN, una tabla debe cumplir con la 1FN y, además, todos los atributos no clave deben depender completamente de la clave primaria. En otras palabras, elimina la dependencia parcial de los atributos en caso de que la clave primaria sea compuesta, a continuación en la figura 2.6 observamos un ejemplo.

Figura 2.6*Ejemplo de Segunda Forma Normal*

The diagram illustrates the decomposition of a second normal form (2FN) table into two third normal form (3FN) tables. It consists of three parts: a top table, a middle arrow, and a bottom section.

Top Table:

| id | Apellido | Nombre | Código de vuelo | Origen | Destino |
|-----------|-----------------|---------------|------------------------|---------------|----------------|
| 1 | Chavez | Juan | 101 | La Paz | Beni |
| 2 | Marca | Mayra | 102 | Oruro | Tarija |

Middle Arrow:

A large blue downward-pointing arrow is positioned between the top table and the bottom section.

Bottom Section:

| id | Apellido | Nombre | Código de vuelo |
|-----------|-----------------|---------------|------------------------|
| 1 | Chavez | Juan | 101 |
| 2 | Marca | Mayra | 102 |

| Código de vuelo | Origen | Destino |
|------------------------|---------------|----------------|
| 101 | La Paz | Beni |
| 102 | Oruro | Tarija |

Nota: En la primera tabla los atributos Origen y Destino dependen del atributo Código de vuelo, entonces lo dividimos en dos tablas.

Tercera Forma Normal (3FN): Una tabla está en 3FN si cumple con 2FN y no tiene dependencias transitivas. Esto significa que los atributos no clave no dependen de otros atributos no clave,

así como se observa en la figura 2.7.

Figura 2.7

Ejemplo de Tercera Forma Normal



| CI | Apellido | Nombre | Dirección | Teléfono | Materia | Sigla | id_carrera |
|----|----------|--------|-----------|----------|---------------|-------|------------|
| 1 | Chavez | Juan | C/Costas | 123456 | Programacion | PRG | Sistemas |
| 2 | Marca | Mayra | C/Rivas | 456789 | Base de datos | BD | Sistemas |
| 1 | Chavez | Juan | C/Costas | 123456 | Base de datos | BD | Sistemas |
| 3 | Ruiz | Carlos | C/Nueve | 123789 | Algebra | ALG | Mecánica |

| CI | Apellido | Nombre | Dirección | Teléfono |
|----|----------|--------|-----------|----------|
| 1 | Chavez | Juan | C/Costas | 123456 |
| 2 | Marca | Mayra | C/Rivas | 456789 |
| 1 | Chavez | Juan | C/Costas | 123456 |
| 3 | Ruiz | Carlos | C/Nueve | 123789 |

| CI | Sigla |
|----|-------|
| 1 | PRG |
| 1 | BD |
| 2 | BD |
| 3 | ALG |

| Id_carrera | Carrera |
|------------|----------|
| 123 | Sistemas |
| 456 | Mecánica |

| Sigla | Materia |
|-------|---------------|
| PRG | Programación |
| BD | Base de datos |
| ALG | Algebra |

Nota: En la primera tabla se observa que los atributos Materia, Sigla, id_carrera no dependen de CI, lo que hace que subdividamos la tabla en otras tablas más.

2.11. PATRÓN DE DISEÑO WEB

De acuerdo con el modelo en cascada, en su segunda etapa, que corresponde al punto 2.7.2 sobre el Diseño del sistema y del software, en esta etapa se desarrolla también el diseño de la interfaz, que será el medio principal de interacción del usuario con el sistema.”

Los patrones de diseño web se refieren a un conjunto de principios que guían el desarrollo de páginas web y la creación de interfaces de usuario, abarcando desde elementos individuales hasta componentes más completos. A diferencia de las plantillas, que sirven como un punto de partida para diseñar páginas con una estructura similar pero con contenidos distintos, los patrones ofrecen un enfoque basado en buenas prácticas para el diseño. Utilizar estos patrones conlleva dos beneficios clave:

Primero, aseguran una experiencia de usuario óptima al proporcionar directrices sobre la disposición y organización de los distintos elementos en una página, adaptándose así a las necesidades y expectativas de los usuarios.

En segundo lugar, contribuyen a agilizar y simplificar el proceso de diseño, abordando problemas comunes en el desarrollo web que ya han encontrado soluciones efectivas.

2.12. DISEÑO EN FORMA DE F

El diseño en forma de "F" es un patrón de diseño web basado en el comportamiento natural de lectura de los usuarios, quienes escanean y consumen el contenido de las páginas web de una manera similar a la letra "F". Según Nielsen (2006), los usuarios tienden a leer primero horizontalmente, luego realizan una segunda mirada horizontal más corta y finalmente escanean hacia abajo de manera vertical, lo que crea la forma de la letra "F". Este comportamiento es clave para organizar el contenido en páginas de manera que optimice la visibilidad y la usabilidad como se puede ver en la figura 2.8.

Figura 2.8

Resultados de los estudios de seguimiento de los ojos.



Eyetracking by Nielsen Norman Group nngroup.com NN/g

Nota: Obtenido de <https://www.nngroup.com/>

En una web de administración, este modelo puede aplicarse eficazmente cuando se dispone un encabezado en la parte superior, un menú lateral izquierdo y el contenido principal en el centro:

Encabezado superior: El encabezado se ubica en la parte superior de la página, el primer punto de atención visual de los usuarios, donde deberían colocarse elementos importantes como la identidad de la empresa, accesos a configuraciones o funciones críticas como el logout, ya que es la zona donde los usuarios enfocan su mirada inicialmente.

Menú lateral izquierdo: El menú de navegación en el lado izquierdo está alineado con la columna vertical de la "F", lo que lo convierte en el lugar ideal para ubicar opciones de navegación. Las funciones más importantes deben estar en la parte superior del menú, donde los usuarios tienden a hacer un recorrido visual de arriba hacia abajo.

Contenido central: El contenido principal de la página debería alinearse con las barras horizontales del patrón en forma de "F". La información más importante debe situarse en la parte superior, ya que esta área recibe más atención, mientras que las secciones menos cruciales pueden colocarse más abajo.

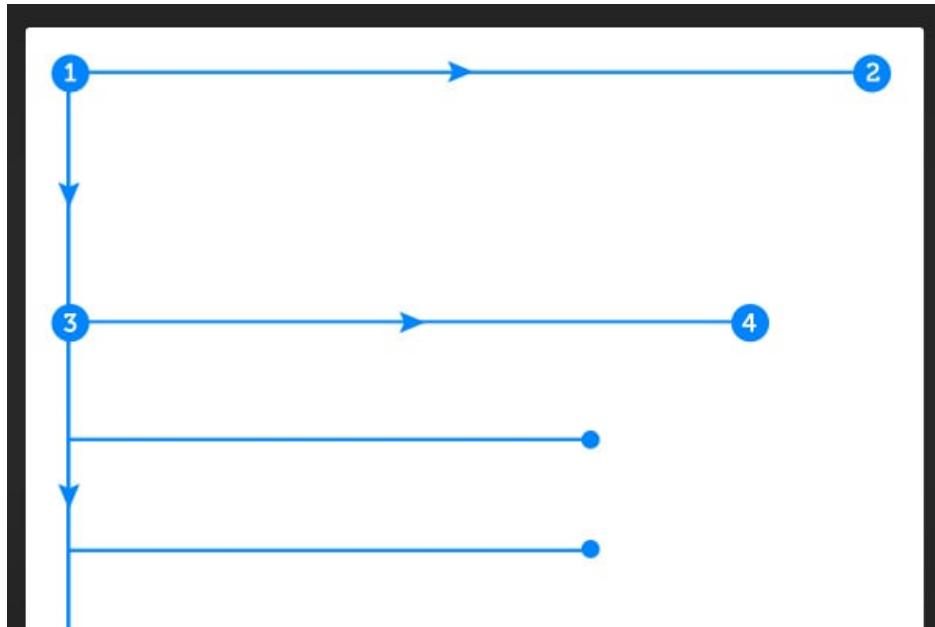
En la figura 2.9 se ve que este diseño mejora la experiencia del usuario al seguir su comportamiento visual natural, lo que resulta en una mayor usabilidad y eficiencia en tareas administrativas. Según Nielsen (2006), el diseño en forma de F asegura que las áreas más importantes reciban la mayor atención, facilitando la navegación y el acceso a la información esencial:

2.13. PATRÓN DE ARQUITECTURA

Los patrones de arquitectura son soluciones generales y reutilizables para problemas comunes en el diseño de software. Estos patrones proporcionan un conjunto de subsistemas predefinidos, especifican sus responsabilidades e incluyen reglas y pautas para organizar las relaciones entre ellos. De acuerdo a Cervantes et al. (2016), los patrones de arquitectura se caracterizan por su capacidad de reutilización, abstracción, escalabilidad y mantenibilidad, lo que los convierte en herramientas

Figura 2.9

Diseño en forma de F para la estructura de una web de administración.



Nota: Obtenido de <https://webdesign.tutsplus.com/> fundamentales para el diseño de sistemas de software robustos y flexibles.

La importancia de los patrones de arquitectura radica en su capacidad para representar la estructura fundamental de un sistema. Como señala Pressman (2010), estos patrones "proporcionan un conjunto de subsistemas predefinidos, especifican sus responsabilidades e incluyen reglas y guías para organizar las relaciones entre los subsistemas". Estos patrones ofrecen un marco que guía la organización del código y la estructura de la aplicación, promoviendo la reutilización y facilitando el mantenimiento y la escalabilidad.

La elección de un patrón de arquitectura adecuado influye en la calidad del software, su rendimiento y su facilidad de uso, el uso de patrones de arquitectura ayuda a los desarrolladores a afrontar la complejidad del software mediante la identificación de soluciones probadas y confiables, lo que contribuye a un desarrollo más ágil y eficiente. Este enfoque permite que los equipos de desarrollo colaboren de manera más efectiva, al tener una base común sobre la cual construir.

Un patrón particularmente relevante en el desarrollo web es el Model-Template-View (MTV).

Este patrón se adapta a las necesidades específicas de diferentes frameworks y lenguajes de programación, permitiendo una mayor modularidad y flexibilidad en el diseño de aplicaciones web.

2.14. PATRÓN MODEL-TEMPLATE-VIEW (MTV)

El patrón Model-Template-View (MTV), es un framework de desarrollo web de código abierto escrito en el lenguaje de programación Python. El patrón MTV tiene la particularidad de separar claramente la lógica de la aplicación en tres componentes principales:

- **Model (Modelo):** se refiere a la estructura de datos en Django, se define esta estructura creando una clase por cada tabla en la base de datos. Cada modelo de Django contiene los campos y comportamientos de los datos que desea almacenar.
- **Template (Plantilla):** Es responsable de la presentación de los datos. Los templates permiten generar las vistas que el usuario final verá en la interfaz de la aplicación.
- **View (Vista):** Actúa como intermediario entre el modelo y los templates, gestionando la lógica de la aplicación y determinando qué datos deben ser enviados a los templates.

La aplicación del patrón MTV en Django facilita la separación de responsabilidades, mejorando la mantenibilidad y la escalabilidad del sistema. Esta separación también permite que diferentes equipos puedan trabajar de manera independiente sobre cada componente, favoreciendo una mayor eficiencia en el desarrollo.

2.15. HERRAMIENTAS DE DESARROLLO

En la tabla 2.2, tabla 2.3 y tabla 2.4 se identifican las herramientas de desarrollo que se utilizan para el proyecto.

Tabla 2.2

Herramientas a utilizar en el desarrollo del front-end del proyecto

| Logo | Herramienta | Versión |
|---|---------------------|---------|
|  | HTML | 5 |
|  | CSS | 3 |
|  | JavaScript | ES14 |
|  | Framework Bootstrap | 4.4 |

Tabla 2.3

Herramientas a utilizar en el desarrollo del back-end del proyecto

| Logo | Herramienta | Versión |
|---|------------------|---------|
|  | Python | 3.12.5 |
|  | Framework Django | 5.1.1 |
|  | PostgreSQL | 16.3 |
|  | pgAdmin | 4 |

Tabla 2.4

Herramientas a utilizar en programación y el control de versiones del proyecto

| Logo | Herramienta | Versión |
|---|--------------------|---------|
|  | Visual Studio Code | 1.94.2 |
|  | Github | 2.46.1 |

2.16. PRUEBAS

La etapa de pruebas se realiza para identificar, corregir errores y asegurar que el sistema funcione según las especificaciones planteadas en los requerimientos. De acuerdo con Sommerville (2011), las pruebas de software deben ser meticulosas, comprobando tanto las funciones internas del sistema como su comportamiento desde la perspectiva del usuario final. Las pruebas no solo permiten detectar defectos, sino también asegurar que el software opere correctamente en diferentes escenarios, reduciendo el riesgo de errores que afecten la experiencia del usuario o la operación de la empresa.

Pruebas unitarias

Las pruebas unitarias verifican el funcionamiento de las partes más pequeñas del software, llamadas “unidades”. Cada unidad, generalmente una función o un módulo, se prueba de forma aislada para asegurar que su comportamiento sea correcto. Como destaca Pressman (2010), estas pruebas son fundamentales para detectar errores en los componentes básicos del sistema, lo cual facilita una construcción más sólida del sistema completo.

También las pruebas unitarias permiten detectar errores rápidamente, lo que reduce el costo y el tiempo necesarios para corregir problemas en etapas avanzadas del desarrollo. Estas pruebas

se enfocan en verificar la funcionalidad de métodos o funciones específicas en lugar de todo el sistema, utilizando herramientas como unittest en Python. Además, su ejecución frecuente fomenta la confianza en la estabilidad del sistema a medida que se realizan cambios en el código.

2.17. MODELO DE CALIDAD BOEHM

El modelo de calidad de Boehm fue propuesto por Barry Boehm en los años 70 como un enfoque exhaustivo para evaluar la calidad del software de forma estructurada. Su objetivo principal es ofrecer un marco de referencia que descompone el concepto de calidad en atributos y sub-atributos específicos, permitiendo a los ingenieros de software identificar y medir aspectos clave para mejorar la calidad del software desarrollado. Según Pressman (2010), este modelo proporciona una guía que facilita el cumplimiento de los objetivos de calidad, estableciendo métricas y criterios que ayudan en la toma de decisiones a lo largo del proceso de desarrollo.

2.17.1. Estructura jerárquica del modelo de calidad Boehm

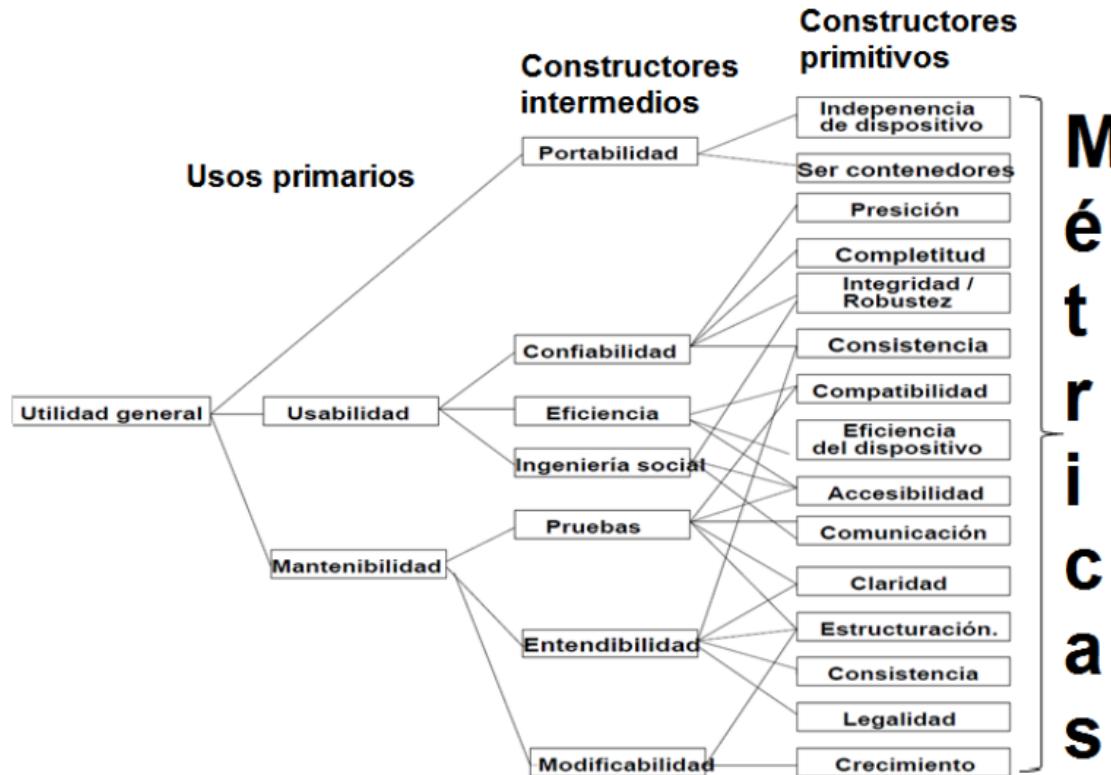
El Modelo de Calidad de Boehm establece un marco de trabajo para evaluar la calidad de un software mediante una estructura jerárquica de características. En la figura 2.10, el modelo se organiza en tres niveles: usos primarios, constructores intermedios y constructores primitivos. Cada uno de estos niveles desglosa aspectos de la calidad que se vuelven más específicos a medida que se profundiza en la jerarquía.

En el primer nivel define tres utilidades como factores para la calidad del software que son las siguientes:

- 1) Utilidad del software tal y como está en el momento de la evaluación, identificar la facilidad de uso, fiabilidad y eficiencia.
- 2) La facilidad de mantenimiento identificar lo que es modificable para realizar sus modificaciones pertinentes y las pruebas adecuadas.
- 3) Portabilidad facilidad que tiene el software de ser utilizado en entorno distinto.

Figura 2.10

Estructura jerárquica del Modelo de Calidad Boehm



Nota: Obtenido de <https://evaluacionred-g3-2019.fandom.com/>

2.17.2. Usabilidad

La usabilidad en el modelo de calidad de Boehm se centra en la facilidad y eficiencia con la que los usuarios pueden interactuar con un sistema de software. Esta cualidad es evaluada en varios aspectos clave, que Boehm estructura en componentes específicos para garantizar una experiencia de usuario satisfactoria y una operatividad óptima del sistema. Entre los factores principales se encuentran:

- Facilidad de aprendizaje:** El software debe permitir a los nuevos usuarios comprender y utilizar sus funciones esenciales rápidamente. Esto es particularmente relevante en entornos con rotación frecuente de personal o en sistemas diseñados para usuarios ocasionales

(Boehm, 1978).

- b) **Eficiencia operativa:** Evalúa qué tan eficazmente los usuarios logran sus objetivos con un esfuerzo mínimo, lo que incluye una navegación clara e interfaces intuitivas. Este aspecto asegura que las tareas sean completadas con rapidez y precisión (Sommerville, 2011).
- c) **Satisfacción del usuario:** Considera la percepción subjetiva sobre la facilidad de uso, el diseño de la interfaz y la respuesta general del sistema, lo que resulta crucial para la adopción del software y su aceptación a largo plazo (Pressman, 2010).
- d) **Minimización de errores:** Se enfoca en reducir la probabilidad de errores de usuario y en proporcionar mecanismos claros para corregirlos. Esto incluye mensajes de error comprensibles y funciones para mitigar las consecuencias de los errores (Boehm, 1978).
- e) **Flexibilidad y adaptabilidad:** Examina cómo el sistema puede ajustarse a diferentes necesidades y preferencias de los usuarios, lo que es importante en entornos donde las condiciones cambian frecuentemente (Pressman, 2010).

2.17.3. Portabilidad

La portabilidad, según el modelo de calidad de Boehm, representa la capacidad del software para trasladarse y operar en diferentes plataformas o entornos tecnológicos con modificaciones mínimas o nulas. Este atributo es importante en un mundo donde los sistemas y dispositivos evolucionan constantemente, las empresas buscan soluciones escalables y adaptables que puedan implementarse en múltiples configuraciones técnicas.

- a) **Adaptabilidad:** La adaptabilidad hace referencia a la facilidad con la que el software puede ajustarse a diferentes entornos operativos. Esto incluye compatibilidad con múltiples sistemas operativos, arquitecturas de hardware y dispositivos. Según Sommerville (2011), esta característica asegura que el sistema pueda evolucionar sin ser limitado por la infraes-

tructura tecnológica. En el diseño, se priorizan estándares abiertos y el desacoplamiento del software de elementos específicos del hardware.

- b) **Instalación y configuración sencilla:** Otro componente fundamental de la portabilidad es la facilidad con que el software puede ser instalado en nuevos entornos. Esto implica minimizar las dependencias técnicas y ofrecer procesos de configuración claros, automáticos o guiados. Pressman (2010) señala que un sistema portable debe ofrecer una experiencia de instalación optimizada, lo cual es esencial para empresas que operan en múltiples ubicaciones o que implementan actualizaciones frecuentes.
- c) **Sustituibilidad:** Este aspecto se enfoca en la capacidad del software para reemplazar a otro sistema similar sin interrumpir el funcionamiento normal. Se busca garantizar la compatibilidad con los datos existentes, minimizar el tiempo de migración y evitar la pérdida de funcionalidad. Boehm (1978) enfatiza que esta característica es particularmente importante para organizaciones que dependen de software integrado con procesos comerciales críticos.

3 MARCO APLICATIVO

3.1. ANÁLISIS Y DEFINICIÓN DE REQUERIMIENTOS

En la primera etapa del desarrollo del software, se realiza el Análisis y definición de requerimientos, una fase esencial donde se identifican y documentan las necesidades y expectativas del cliente mediante la ingeniería de requerimientos. Esta etapa es importante para asegurar que el desarrollo posterior esté alineado con las metas del proyecto y que el sistema cumpla efectivamente con las expectativas planteadas, evitando problemas en fases avanzadas del desarrollo.

3.1.1. Levantamiento de requerimientos

Para la realización del levantamiento de requerimientos se realizó un cronograma de reuniones que se observa en la tabla 3.1.

Tabla 3.1

Cronograma de reuniones y entrevistas

| Reunión | Fecha | Tarea |
|----------------|--------------|---|
| 1 | 09/12/2024 | Reunión con el Administrador |
| 2 | 03/01/2025 | Entrevista con el Administrador |
| 3 | 07/02/2025 | Entrevista con el Personal de Atención al Cliente |

Nota: El cronograma muestra las fechas para las reuniones y entrevistas realizadas durante el desarrollo del proyecto.

Como parte inicial al proceso de levantamiento de requerimientos se realizó la **técnica**

de observación directa sobre las operaciones diarias en la empresa de transportes durante la semana del 16 al 20 de diciembre de 2024. Esta técnica permitió identificar puntos críticos iniciales para preparar las entrevistas posteriores al personal de la empresa.

Durante la observación se identificaron los siguientes aspectos:

1. Procesos que requieren optimización:

- Verificación de disponibilidad de asientos
- Registro de encomiendas
- Control de embarque de pasajeros
- Gestión de caja y turnos

2. Puntos críticos identificados:

- Tiempos de espera prolongados en horas pico
- Proceso manual propenso a errores
- Falta de información en tiempo real

3. Oportunidades de mejora:

- Automatización del proceso de venta
- Gestión digital de asientos
- Control automatizado de embarque

Las entrevistas fueron programadas en base al cronograma detallado en la tabla 3.1, logrando reuniones efectivas tanto con el administrador y con el personal de atención al cliente. A continuación, se detallan las consultas realizadas en estas sesiones.

Entrevista al Administrador

Información del Entrevistado

- Cargo: Administrador General

- Fecha: 03/01/2025
- Lugar: Empresa de Transportes Cali Internacional
- Duración estimada: 60 minutos

Preguntas:

1. ¿Cuáles son los principales problemas o dificultades que enfrenta la empresa con el manejo actual?
2. ¿Qué procesos considera que son los más críticos y necesitan mayor atención?
3. ¿Cómo se gestiona actualmente el control de buses y encomiendas en la empresa?
4. ¿Cuántos buses tiene actualmente la empresa en operación?
5. ¿Cuántas rutas manejan y cuáles son sus principales destinos?
6. ¿Qué problemas son los más frecuentes en el servicio de encomiendas?
7. ¿Qué volumen promedio de encomiendas manejan diariamente?
8. ¿Qué funcionalidades específicas necesita que tenga el módulo de gestión de buses para optimizar las operaciones?
9. ¿Cómo se realiza actualmente la asignación de conductores a las unidades?
10. ¿Cuántos empleados necesitarán acceso al sistema?
11. ¿Qué roles o niveles de acceso considera necesarios para el personal?
12. ¿Qué tipos de reportes son esenciales para la toma de decisiones?
13. ¿Cómo le gustaría que se maneje el sistema de reservas y venta de pasajes?
14. ¿Qué sistema de tarifas manejan y cómo les gustaría que se implemente en el software?
15. ¿Qué tipo de reportes financieros necesita generar periódicamente?

16. ¿En qué plazo espera que el sistema esté completamente operativo?

Entrevista al Personal de atención al cliente

Información del Entrevistado

- Vendedor de pasajes y recepcionista de encomiendas
- Fecha: 07/02/2025
- Lugar: Empresa de Transportes Cali Internacional
- Duración estimada: 60 minutos

Preguntas:

1. ¿Cuál es el proceso actual que sigue para vender un boleto de viaje?
2. ¿Qué información del cliente es obligatoria registrar al momento de la venta?
3. ¿Cómo maneja las reservaciones de asientos?
4. ¿Qué problemas son los más frecuentes durante el proceso de venta de boletos?
5. ¿Cómo gestiona actualmente los diferentes tipos de tarifas?
6. ¿Cuál es el procedimiento actual para registrar una encomienda?
7. ¿Qué información necesita registrar sobre las encomiendas?
8. ¿Cómo realiza el seguimiento de una encomienda cuando un cliente lo solicita?
9. ¿Qué problemas son los más comunes en el servicio de encomiendas?
10. ¿Cómo maneja las quejas por pérdida o retraso de encomiendas?
11. ¿Cuáles son las preguntas más frecuentes de los clientes?
12. ¿Qué información necesita tener a mano para responder rápidamente a las consultas de los clientes?
13. ¿Cómo gestiona los cambios o cancelaciones de pasajes?

14. ¿Cómo realiza el cierre de caja de sus ventas?
15. ¿Qué tipo de reportes necesita generar durante su turno?
16. ¿Cómo verifica la disponibilidad de asientos en los buses?
17. ¿Qué reportes le facilitarían su trabajo diario?
18. ¿Qué proceso sigue cuando un cliente pierde su boleto?
19. ¿Qué experiencia tiene en el uso de sistemas informáticos?
20. ¿Qué aspectos considera importantes incluir en la capacitación del nuevo sistema?
21. ¿Qué información proporciona a los clientes sobre el viaje?

Las entrevistas permiten complementar la información obtenida en la observación directa y proporcionar una visión más clara de los procesos actuales y las necesidades reales de automatización.

3.1.2. Análisis de requerimientos

Durante la fase de Análisis de requerimientos del proyecto, se realizó un proceso integral para definir y estructurar los casos de uso que abarcarán las funcionalidades esenciales del sistema, esta etapa se centró en organizar las interacciones clave que los usuarios tendrán con el sistema, permitiendo establecer una visión clara de cómo deben funcionar los distintos módulos.

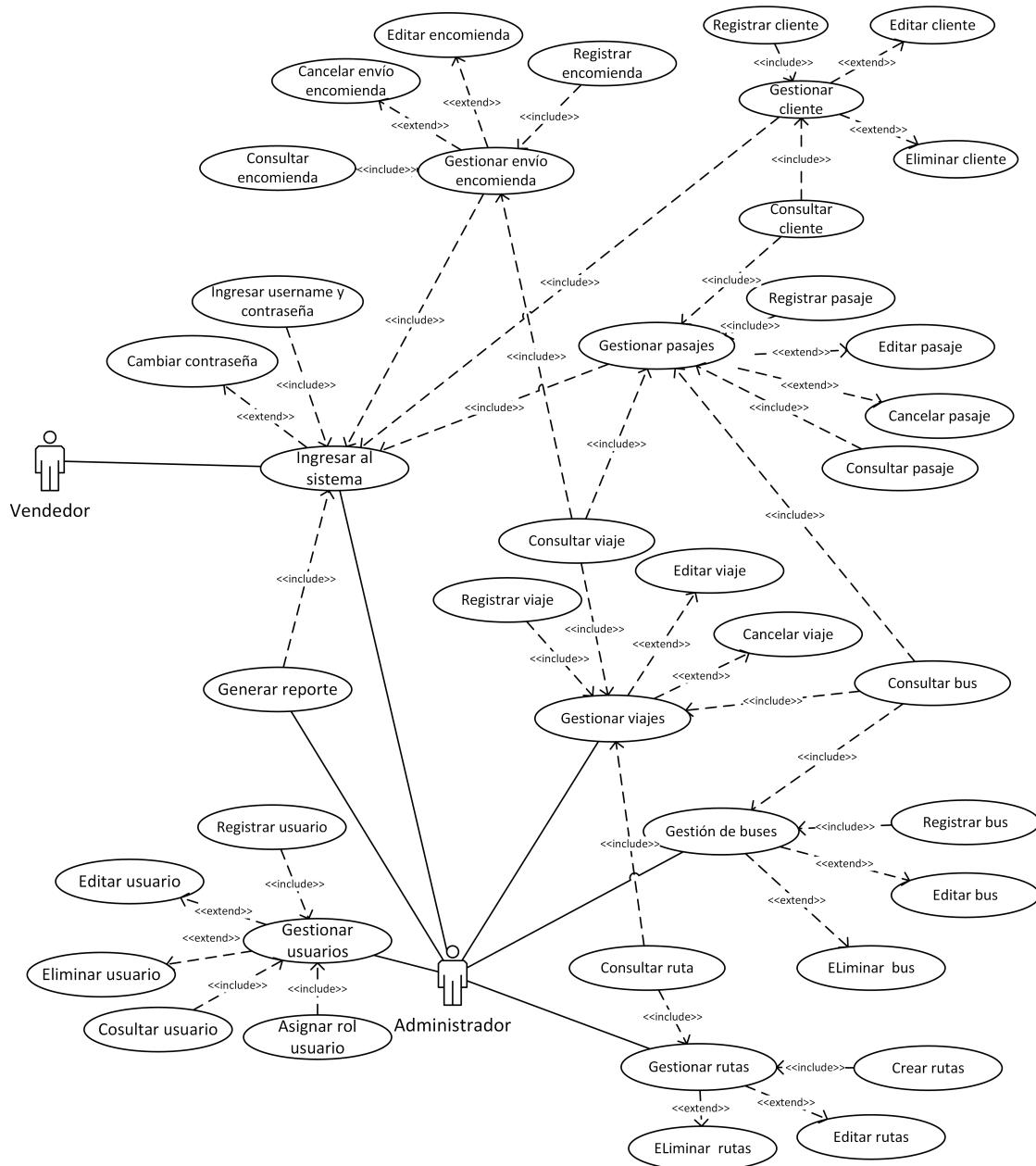
En esta fase se realizó la organización y documentación de los casos de uso, lo que facilitó el establecimiento de una visión clara de las funcionalidades a desarrollar, se logró crear una especificación detallada de cada caso de uso, incluyendo actores, flujos principales, flujos alternativos y condiciones específicas de ejecución. Este nivel de detalle proporciona una guía clara para las fases subsecuentes del proyecto, también permite establecer un entendimiento común con los interesados sobre cómo el sistema debe comportarse ante las diferentes interacciones de los usuarios.

Diagrama de casos de uso

De acuerdo al análisis de requerimientos en la figura 3.1, se presenta el Diagrama de casos de uso para el sistema.

Figura 3.1

Diagrama de Casos de Uso del sistema de logística y gestión de buses.



Nota: El diagrama representa los principales actores y casos de uso del sistema.

Especificaciones de casos de uso

El presente apartado desde la tabla 3.2 a la tabla 3.9 contiene la especificación detallada de los casos de uso del sistema, organizados por módulos funcionales, para cada caso se describe el alcance, los objetivos específicos y las interacciones necesarias para completar las funcionalidades requeridas, proporcionando una visión clara del comportamiento esperado.

Tabla 3.2

Especificación de casos de uso: Autenticación del usuario

| | |
|---|--|
| Caso de Uso: Autenticación del usuario | Actores: Administrador, Personal de atención al cliente |
| Descripción: La página de autenticación de usuarios permite al administrador y al personal de atención al cliente acceder al sistema y realizar funciones de cada usuario. | |
| Secuencia Normal: <ul style="list-style-type: none"> El usuario y la contraseña son validados en la base de datos. Se verifica en la base de datos el tipo de usuario que se ha autenticado y se lo dirige a las opciones pertinentes. Se visualizan las opciones que tiene cada usuario. | |
| Precondiciones: El usuario debe contar con un nombre de usuario y una contraseña para poder acceder a las opciones del sistema web. | Postcondiciones: Los datos del usuario se mantienen mientras su sesión esté abierta después de que se ha autenticado en el sistema. |
| Excepciones: Si el usuario y la contraseña no existen en la base de datos o si la contraseña no corresponde al usuario, se muestra una notificación de error. | |

Nota: Este caso de uso describe el proceso de autenticación de los usuarios.

Tabla 3.3

Especificación de casos de uso: Gestionar bus

| | |
|---|----------------------------------|
| Caso de Uso: Gestionar bus | Actores: Administrador |
| Descripción: Este caso de uso hace referencia al registro de los datos de un bus. | |
| Secuencia Normal: | |
| El administrador debe elegir la opción “Configurar Bus” del ítem “Configurar”. | |
| El administrador debe consultar la existencia del bus a registrar. | |
| El administrador debe completar la placa del bus, la capacidad de personas, modelo y año. | |
| El administrador guarda el registro realizado. | |
| El administrador podrá editar los datos del registro y eliminar el registro realizado. | |
| Precondiciones: El administrador debe acceder al sistema con su usuario y contraseña. | Postcondiciones: Ninguna. |
| Excepciones: | |
| Si el usuario y la contraseña no existen en la base de datos o si la contraseña no corresponde al usuario, se muestra una notificación de error solicitando nuevamente los datos. | |
| De no completar los datos en los registros, se mostrará un mensaje mencionando qué datos están vacíos y cuáles no han sido seleccionados. | |

Nota: El caso de uso cubre el proceso de registro de un bus en el sistema, incluyendo los detalles como la placa, capacidad, modelo y año, así como la opción de editar o eliminar estos datos.

Tabla 3.4

Especificación de casos de uso: Gestionar rutas

| | |
|--|----------------------------------|
| Caso de Uso: Gestionar rutas | Actores: Administrador |
| Descripción: Este caso de uso hace referencia al registro de los lugares de origen y destino. | |
| Secuencia Normal: | |
| <p>El administrador debe elegir la opción “Registrar rutas” del ítem “Registro”.</p> <p>El administrador debe consultar la existencia de las rutas de origen y de destino a registrar.</p> <p>El administrador debe completar los datos del lugar de origen como el nombre de la ciudad.</p> <p>Por defecto el estado muestra como activo. De igual manera, el administrador deberá completar los mismos datos para el lugar de destino.</p> <p>El administrador guarda el registro realizado.</p> <p>El administrador podrá editar los datos del registro, eliminar el registro realizado y cambiar el estado del registro.</p> | |
| Precondiciones: El administrador debe acceder al sistema con su usuario y contraseña. | Postcondiciones: Ninguna. |
| Excepciones: | |
| <p>Si el usuario y la contraseña no existen en la base de datos o si la contraseña no corresponde al usuario, se muestra una notificación de error solicitando nuevamente los datos.</p> <p>De no completar los datos en los registros, se mostrará un mensaje mencionando qué datos están vacíos y cuáles no han sido seleccionados.</p> | |
| Nota: Este caso de uso se refiere al registro de rutas, en el cual el administrador debe ingresar los lugares de origen y destino, con la opción de editar o eliminar estos registros posteriormente. | |

Tabla 3.5*Especificación de casos de uso: Gestionar viajes*

| | |
|---|-------------------------------|
| Caso de Uso: Gestionar viajes | Actores: Administrador |
| <p>Descripción: Este caso de uso hace referencia a la programación de los viajes con sus fechas respectivas para cada uno. Se asignará un viaje a un bus y, por defecto, se indicarán los lugares de origen y destino. Tendrá un estado de viaje.</p> | |
| <p>Secuencia Normal:</p> <p>El administrador debe elegir la opción “Registrar viajes” del ítem “Registro”.</p> <p>El administrador debe consultar el bus que realizará un viaje.</p> <p>El sistema muestra el lugar de ubicación del bus y por defecto lo asigna al campo de ciudad de origen. También se completan los campos de capacidad de personas y límite de carga.</p> <p>El administrador debe completar la ciudad de destino, debe programar la fecha de salida y de llegada del viaje. Debe elegir un estado del viaje que, por defecto, el sistema muestra como PENDIENTE.</p> <p>El administrador guarda el registro realizado.</p> <p>El administrador podrá editar los datos del registro y eliminar el registro realizado.</p> | |
| <p>Precondiciones: El administrador debe acceder al sistema con su usuario y contraseña.</p> <p>Postcondiciones: Ninguna.</p> | |
| <p>Excepciones: De no completar los datos en los registros, se mostrará un mensaje mencionando qué datos están vacíos y cuáles no han sido seleccionados.</p> | |
| <p>Nota: Este caso de uso trata sobre la programación de viajes, donde el administrador asigna un bus y establece la fecha y hora de salida.</p> | |

Tabla 3.6*Especificación de casos de uso: Gestionar pasaje*

| | |
|---|---|
| Caso de Uso: Gestionar venta y reserva de pasajes | Actores: Personal de atención al cliente |
| Descripción: Este caso de uso hace referencia a la venta y reservas de pasajes para los buses. | |
| Secuencia Normal: | |
| <p>El vendedor debe elegir la opción “Pasajes”.</p> <p>El vendedor consulta el día del viaje, el sistema mostrará la lista de viajes programados según la fecha actual del sistema. Por defecto se completan los datos de la ciudad de origen y de destino y el bus programado.</p> <p>El vendedor debe consultar los asientos libres en los buses disponibles.</p> <p>El vendedor consulta la existencia del cliente; si existe, se obtienen los datos y se completan en los campos vacíos; si no existe, procede a registrar los datos del cliente.</p> <p>Por defecto se completan los datos en los demás campos según el registro de los datos de las personas. El sistema consulta, calcula el monto del pago y completa los campos.</p> <p>Se imprime el boleto del pasaje con los datos del pasajero, el destino, el monto de pago, el tipo de pasaje y un número de ticket.</p> <p>El vendedor también podrá editar los datos del registro del pasaje y cancelar el registro.</p> | |
| Precondiciones: El vendedor debe acceder al sistema con su usuario y contraseña. | Postcondiciones: Ninguna. |

Excepciones: De no completar los datos en los registros, se mostrará un mensaje mencionando qué datos están vacíos y cuáles no han sido seleccionados.

Nota: Este caso de uso describe el proceso de venta y reserva de pasajes, incluyendo la consulta de disponibilidad, registro de datos del cliente y emisión del boleto.

Tabla 3.7

Especificación de casos de uso: Gestionar rol de usuario

Caso de Uso: Gestionar rol de usuario

Actores: Administrador

Descripción: Este caso de uso hace referencia al registro de un rol de usuario.

Secuencia Normal:

El administrador debe elegir la opción “Configurar roles”.

El administrador debe consultar la existencia del rol a registrar.

El administrador debe completar los campos de nombre del rol y registrar una descripción del mismo.

El administrador debe seleccionar los accesos que tendrá el rol al sistema.

El administrador podrá editar los datos del registro y eliminar el registro realizado.

Precondiciones: El administrador debe acceder al sistema con su usuario y contraseña.

Postcondiciones: Ninguna.

Excepciones: De no completar los datos en los registros, se mostrará un mensaje mencionando qué datos están vacíos y cuáles no han sido seleccionados.

Nota: Caso de uso que describe el proceso de asignación de roles a los usuarios del sistema.

Tabla 3.8

Especificación de casos de uso: Gestionar encomienda

| | |
|--|---|
| Caso de Uso: Gestionar encomienda | Actores: Personal de atención al cliente |
| Descripción: Este caso de uso hace referencia al registro de las encomiendas. | |
| Secuencia Normal: | |
| <p>El recepcionista debe elegir la opción “Enviar encomienda”.</p> <p>El recepcionista debe consultar el día del viaje. Por defecto se completan los campos del lugar de origen y destino y el bus programado.</p> <p>El recepcionista realiza una consulta de la existencia de los clientes (remitente y destinatario); si existen se obtienen los datos y se completan los campos vacíos. Si no existen, se procede a registrar sus datos.</p> <p>El recepcionista debe registrar los datos de la encomienda o carga, llenar una descripción de la encomienda, la cantidad, el peso y el monto a pagar. El estado del encargo por defecto se guarda como “PENDIENTE”.</p> <p>El recepcionista debe imprimir el comprobante del registro del envío.</p> <p>Cuando la encomienda llega a su destino y se realiza el recojo, el recepcionista debe constatar que la persona que recoge sea la registrada en el sistema. Luego debe actualizar el estado del encargo a “ENTREGADO” y hacer entrega de la encomienda.</p> | |
| Precondiciones: El recepcionista debe acceder al sistema con su usuario y contraseña. Postcondiciones: Ninguna. | |

Excepciones: De no completar los datos en los registros, se mostrará un mensaje mencionando qué datos están vacíos y cuáles no han sido seleccionados.

De no actualizar el estado del encargo entregado, éste se mantendrá como pendiente.

Nota: Caso de uso que hace referencia al registro y gestión de encomiendas.

Tabla 3.9

Especificación de casos de uso: Generar reporte

| | |
|-------------------------------------|--|
| Caso de Uso: Generar reporte | Actores: Administrador, personal de atención al cliente |
|-------------------------------------|--|

Descripción: Este caso de uso hace referencia al proceso de generación de reportes necesarios para la gestión administrativa de la empresa.

Secuencia Normal:

El usuario puede ver dentro del sistema sólo los reportes que su perfil le permita. Cabe mencionar que el perfil con todos los permisos u opciones es el del Administrador.

El usuario selecciona el reporte que desea generar.

El usuario ingresa los parámetros de búsqueda antes de generar el reporte.

El sistema muestra los datos del reporte.

El sistema imprime el reporte seleccionado.

Precondiciones: Los usuarios deben acceder al sistema con su usuario y contraseña.

Postcondiciones: Ninguna.

Excepciones: Si el usuario y la contraseña no existen o si la contraseña no corresponde al usuario, se muestra una notificación de error solicitando nuevamente los datos.

Nota: Caso de uso que describe el procedimiento para la generación de reportes.

3.1.3. Especificación de requerimientos

En esta etapa del proyecto, se presenta la especificación detallada de los requerimientos funcionales y no funcionales identificados para el sistema. Esta sección tiene como objetivo proporcionar una descripción completa de las capacidades y restricciones que deberá cumplir la solución, estableciendo así las bases para su posterior diseño e implementación.

Requerimientos funcionales

Tabla 3.10

Lista de Requerimientos Funcionales

| ID | Requerimiento |
|-------|---|
| RF-01 | El sistema contará con un módulo de gestión de perfiles de usuarios, distinguiendo entre administradores y empleados. |
| RF-02 | El sistema permitirá la gestión de rutas y horarios de los servicios de transporte. |
| RF-03 | El sistema contará con un módulo de venta y reserva de pasajes para los usuarios. |
| RF-04 | El sistema realizará la asignación de asientos en los vehículos de transporte. |
| RF-05 | El sistema permitirá el registro de las encomiendas transportadas. |
| RF-06 | El sistema gestionará la flota de buses, incluyendo información de los vehículos y conductores. |

Nota: Esta tabla enumera los principales requerimientos funcionales del sistema propuesto, abarcando aspectos relacionados con usuarios, servicios de transporte, ventas y operaciones logísticas.

Requerimientos no funcionales

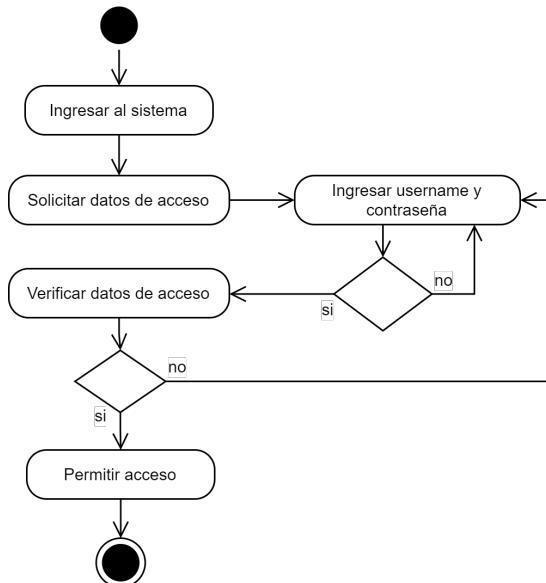
Tabla 3.11*Lista de Requerimientos No Funcionales*

| ID | Requerimiento |
|--------|--|
| RNF-01 | El sistema debe ser accesible desde múltiples navegadores (Chrome, Firefox) y celulares. |
| RNF-02 | Capacidad de exportar reportes en formatos estándar (PDF, CSV, Excel). |
| RNF-03 | Arquitectura de software modular y escalable para facilitar actualizaciones. |

Nota: Esta tabla enumera los principales requerimientos no funcionales del sistema.

3.1.4. Gestión de requerimientos

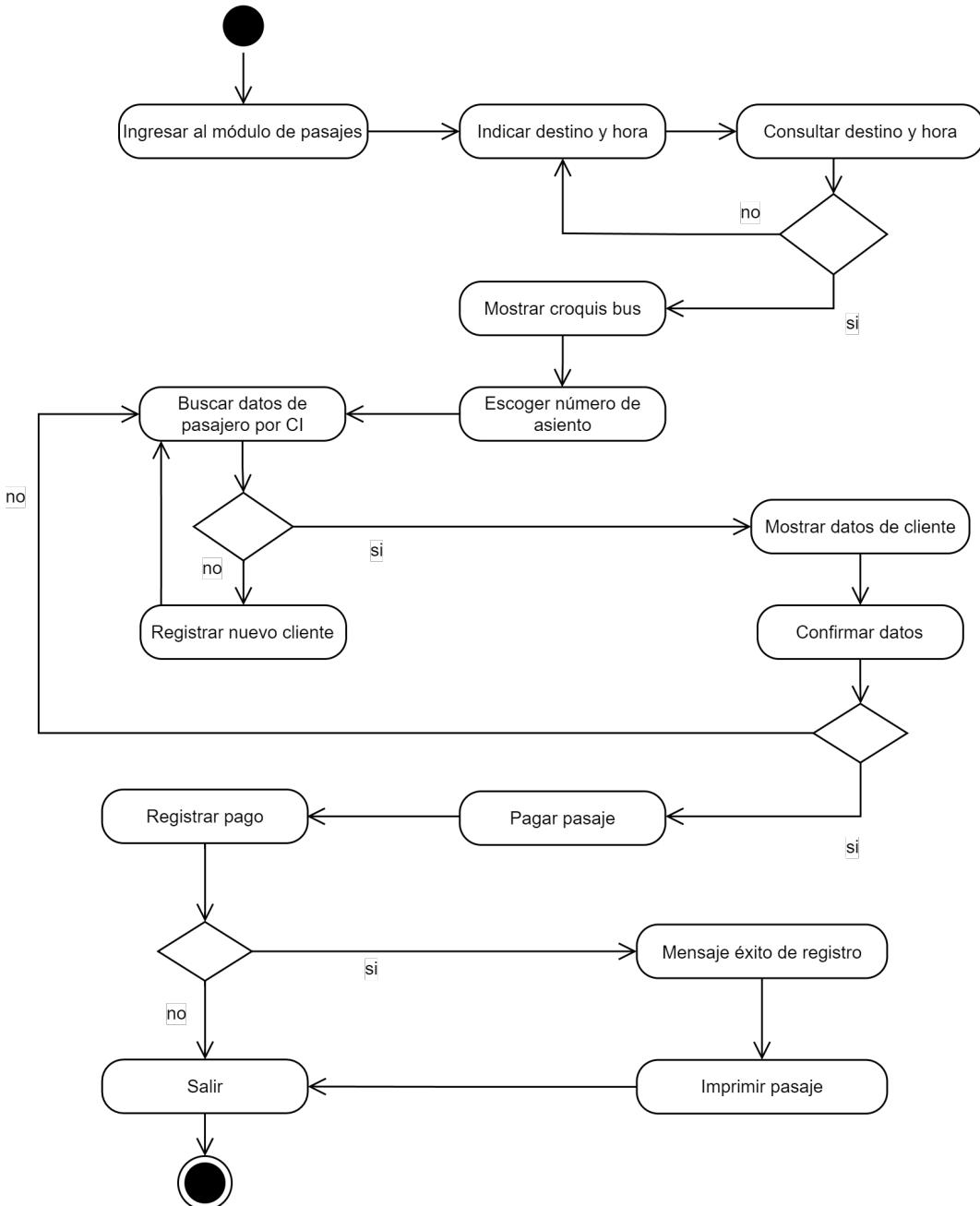
En esta etapa del proyecto, se presentan los diagramas de actividades que se utilizan para representar los flujos de trabajo relacionados con el manejo y control de los requerimientos.

Figura 3.2*Diagrama de actividades del proceso de ingreso al sistema.*

Nota. El diagrama describe el flujo de actividades que realiza el usuario para iniciar sesión en el sistema.

Figura 3.3

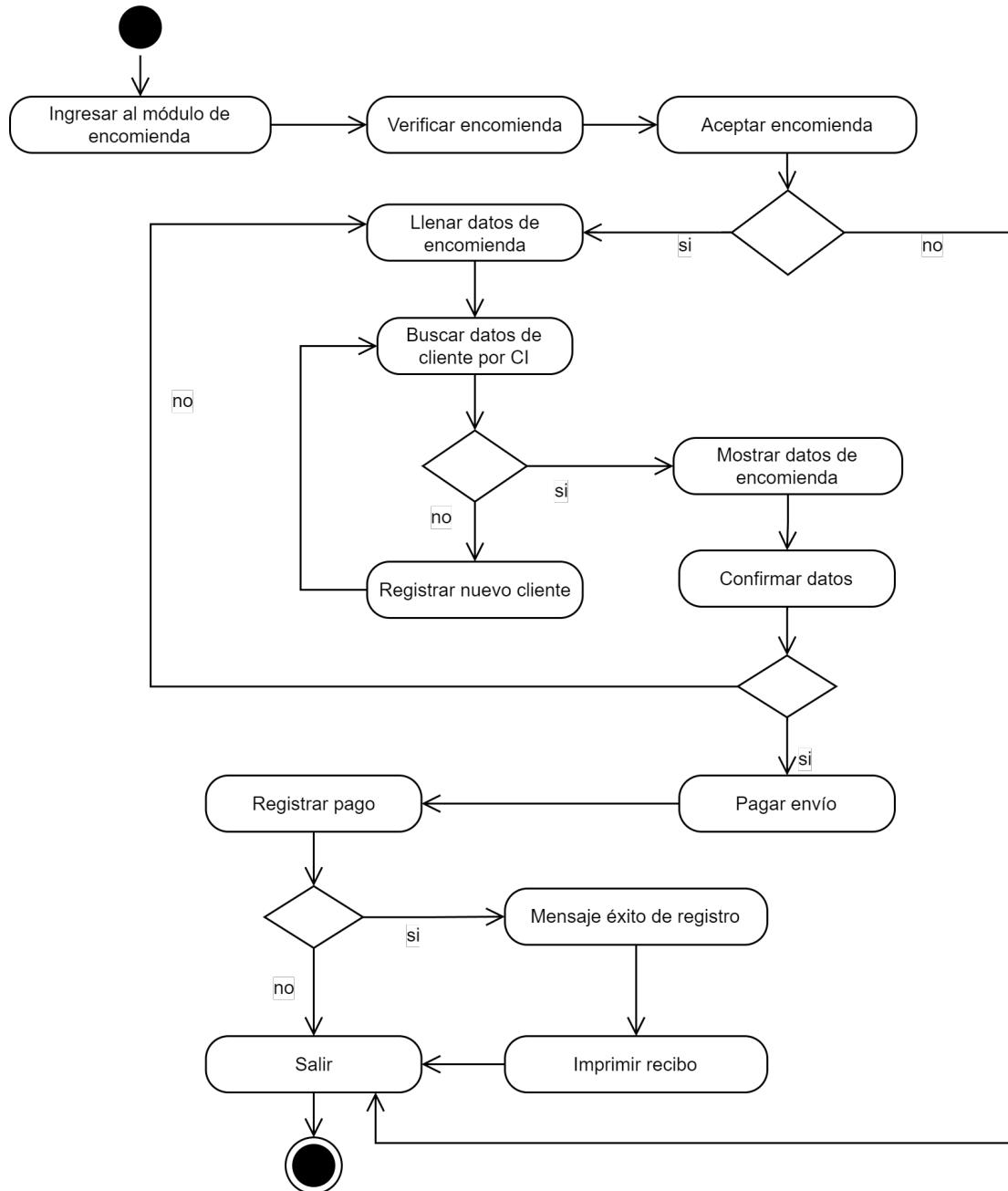
Diagrama de actividades del proceso de venta y reserva de pasajes.



Nota. El diagrama muestra el flujo de actividades desde el ingreso al módulo de venta - reserva de pasajes hasta la confirmación de la venta o reserva.

Figura 3.4

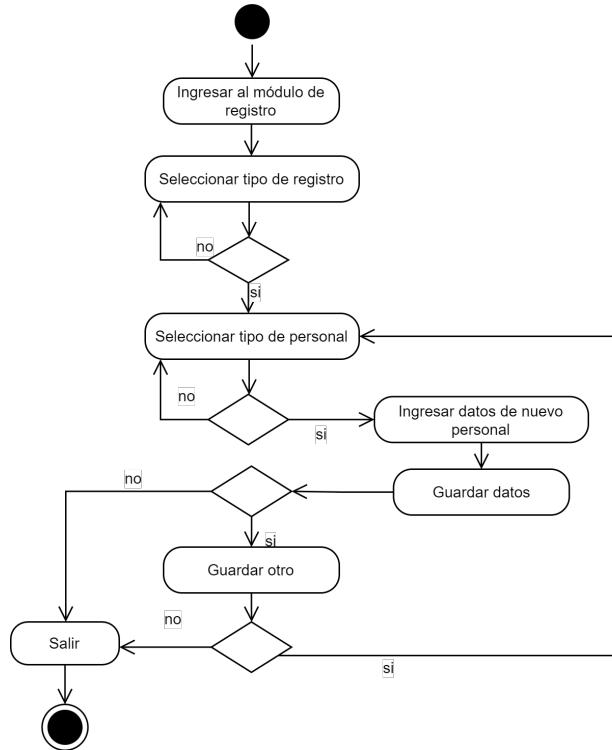
Diagrama de actividades del proceso de registro del envío de encomiendas.



Nota. El diagrama representa las acciones necesarias para registrar una encomienda.

Figura 3.5

Diagrama de actividades del proceso de registro de personal - clientes.



Nota. El diagrama describe las actividades requeridas para registrar en el sistema a los usuarios, tanto personal administrativo como clientes.

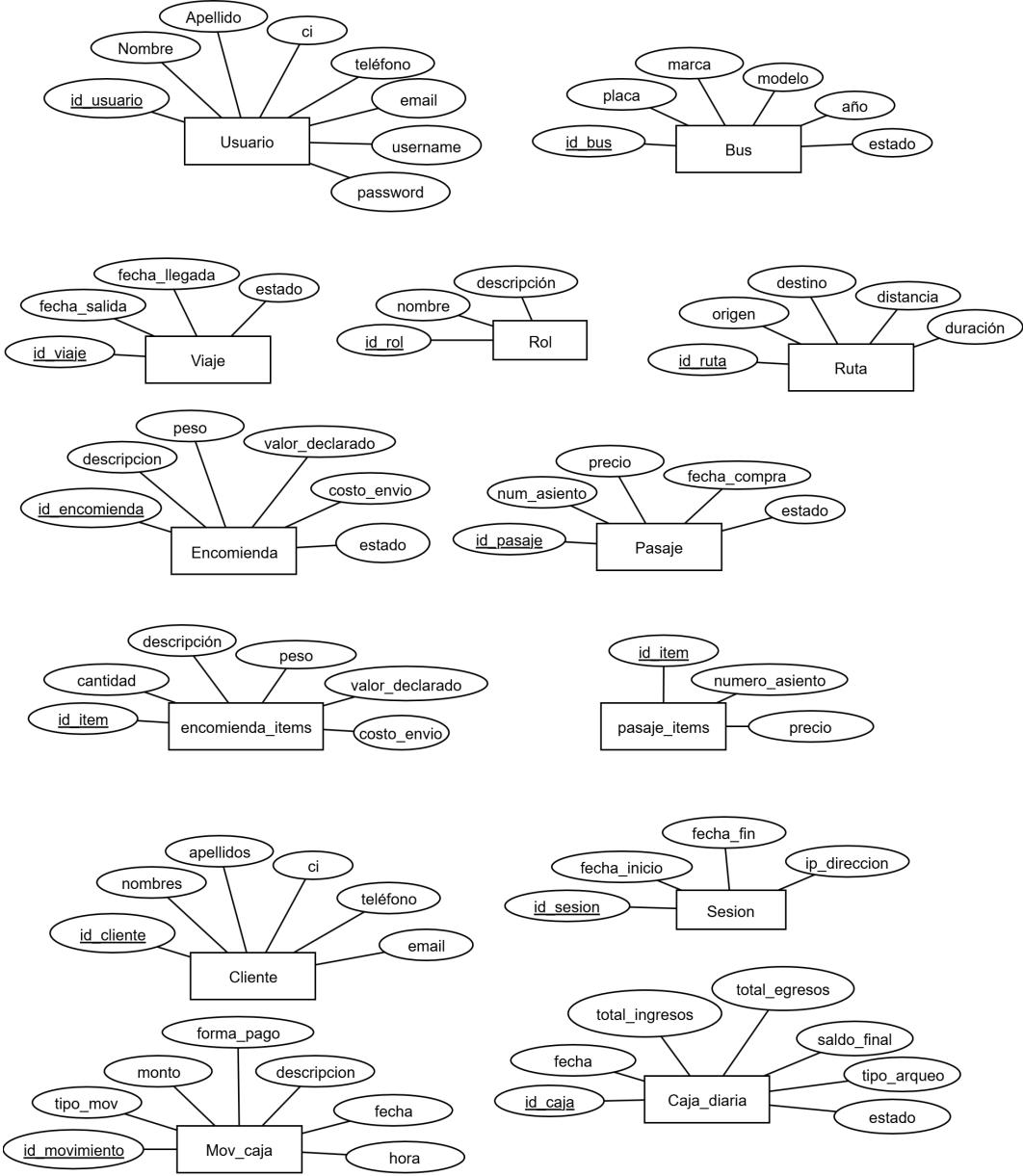
En las figuras 3.2, 3.3, 3.4, y 3.5, se muestran los diagramas de actividades con los flujos correspondientes a cada actividad.

3.2. DISEÑO DEL SISTEMA Y DEL SOFTWARE

Para el modelado de datos ahora se procede a realizar el modelo entidad - relación que permite visualizar la estructura inicial de la base de datos.

3.2.1. Diagrama entidad - relación

A continuación en la figura 3.6 se muestran las entidades del sistema, cada entidad se representa con sus atributos específicos, se distinguen las claves primarias y los atributos descriptivos, estableciendo así la base para el almacenamiento de la información del sistema.

Figura 3.6*Diagrama Entidad-Relación (Entidades y atributos)*

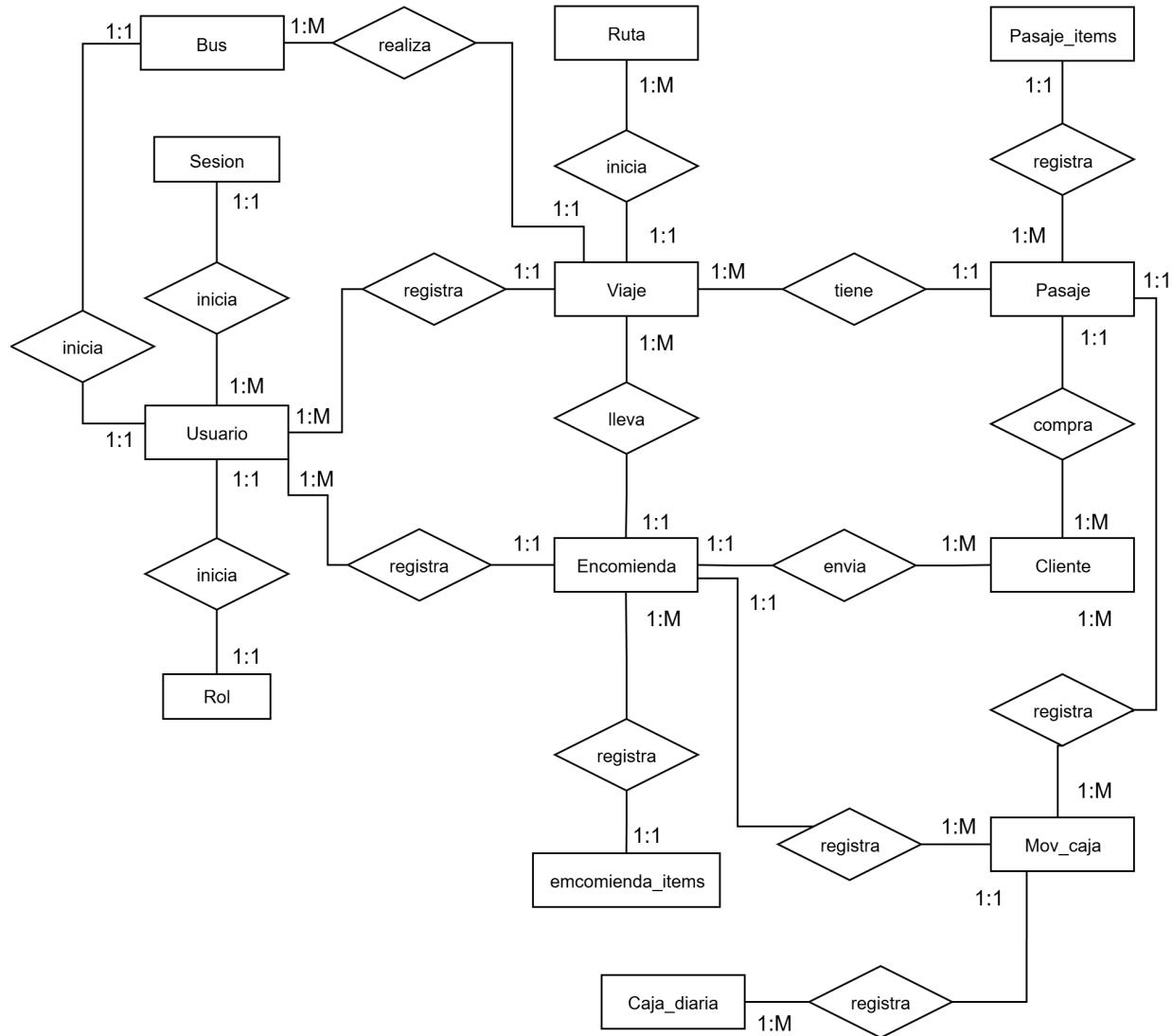
Nota: El diagrama muestra las entidades del sistema y sus atributos, sin incluir las relaciones entre ellas.

La figura 3.7 representa el modelo entidad-relación, donde se visualizan las conexiones entre las diferentes entidades del sistema. Las líneas de relación muestran cómo las entidades interactúan entre sí, y la cardinalidad especificada en cada relación define las reglas de negocio y

las restricciones del sistema.

Figura 3.7

Diagrama Entidad-Relación

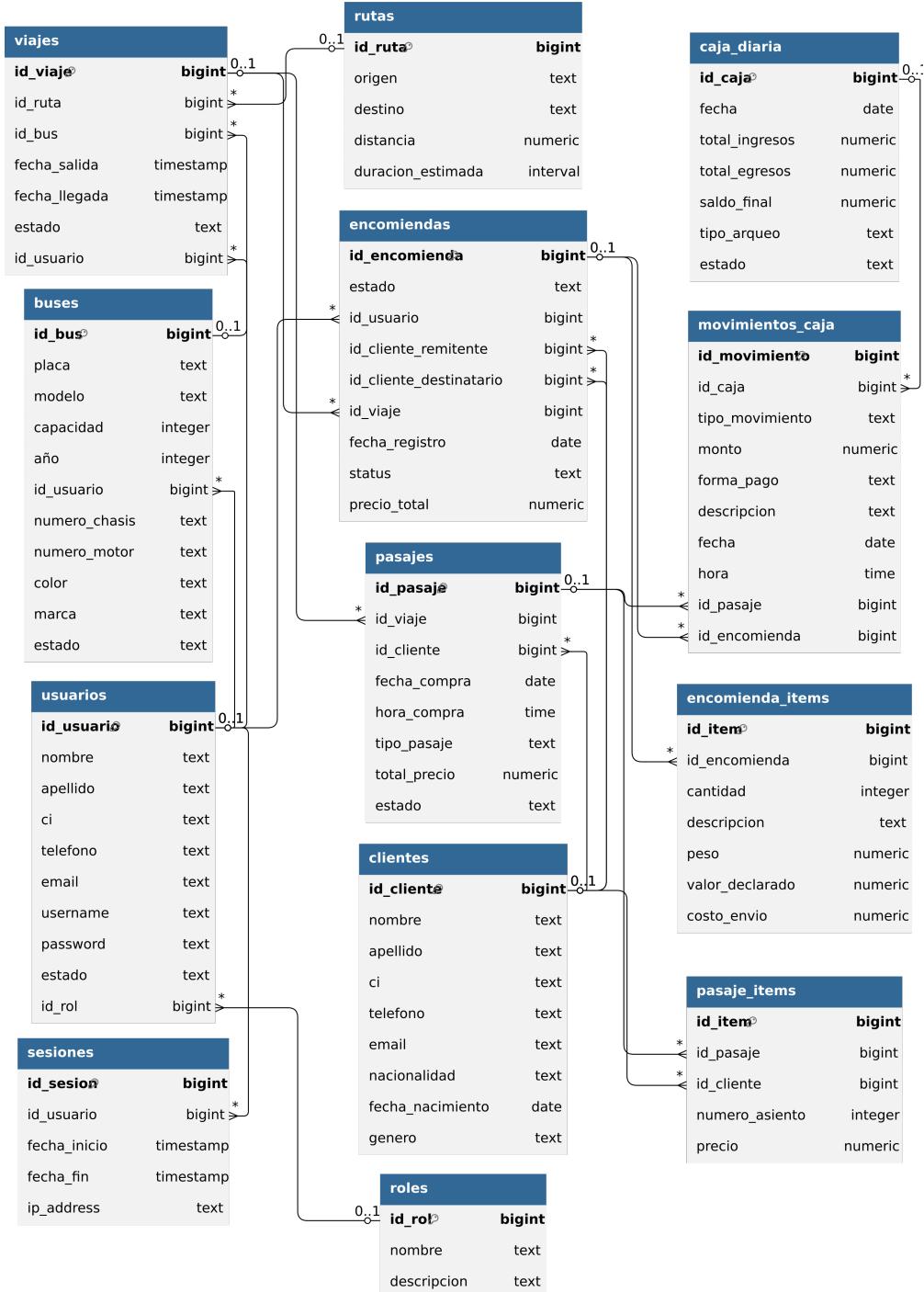


Nota: El diagrama presenta la estructura lógica de la base de datos del sistema.

3.2.2. Diagrama relacional

En la figura 3.8 se presenta el diagrama relacional derivado del modelo entidad-relación mostrado en la figura 3.6 y figura 3.7.

Este diagrama refleja la estructura final de la base de datos, organizando las tablas y sus

Figura 3.8*Diagrama Relacional.*

Nota. El diagrama representa el modelo relacional derivado del diseño entidad-relacion.

relaciones de acuerdo con las reglas de normalización. A través de este diagrama, se puede visualizar cómo se distribuyen los datos en las distintas tablas, asegurando la eficiencia en el almacenamiento y el manejo adecuado de las relaciones entre las entidades del sistema.

3.2.3. Diseño de la interfaz

El diseño de la interfaz se ha desarrollado para ofrecer una experiencia clara, intuitiva y accesible. Se ha tomado en cuenta la estructura lógica de los procesos del sistema, facilitando que el usuario recorra visualmente los elementos más importantes.

Asimismo, la disposición de botones, formularios, menús y tablas responde a la lógica del escaneo visual en forma de "F", facilitando una rápida adaptación por parte de los usuarios. Este enfoque busca mejorar la eficiencia en la operación diaria, también reducir errores y aumentar la satisfacción del usuario con la herramienta.

Figura 3.9

Diseño de la página web principal del sistema.



Nota. Se presenta una estructura clara con accesos directos accesibles.

La página web principal, como se muestra en la figura 3.9, presenta una estructura clara y visualmente atractiva para el usuario.

Figura 3.10

Diseño del inicio de sesión.

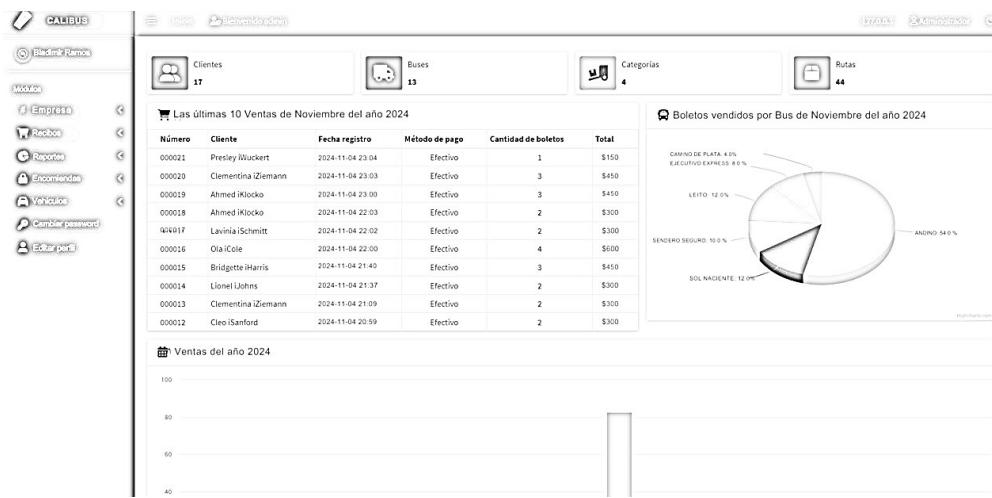


Nota. Diseño preliminar del inicio de sesión.

La figura 3.10, representa el diseño propuesto para la pantalla de inicio de sesión, enfocado en facilitar el ingreso de usuarios al sistema de forma segura.

Figura 3.11

Diseño de la pantalla principal



Nota. Prototipo inicial de la pantalla principal con enfoque en la visualización de reportes.

En la figura 3.11, se observa el diseño de la vista principal, donde se plantean los elementos visuales para la consulta de reportes de ventas.

Figura 3.12

Diseño interfaz Venta de pasajes.

Nota. Vista conceptual destinada a la venta de pasajes.

Como se muestra en la figura 3.12, el diseño contempla una interfaz para la gestión de pasajes, organizada para agilizar el proceso de venta.

Figura 3.13

Diseño interfaz Envío de encomiendas.

Nota. Muestra la organización del formulario para gestionar las encomiendas.

La figura 3.13, corresponde al diseño preliminar de la pantalla destinada al registro y control de envíos de encomiendas.

3.3. IMPLEMENTACIÓN Y PRUEBA DE UNIDAD

El sistema ha sido desarrollado utilizando el lenguaje de programación Python junto con el framework Django, el cual se seleccionó por su arquitectura robusta basada en el patrón MVT (Model-View-Template). Esta estructura permitió una separación clara entre la lógica del negocio, la gestión de datos y la presentación visual, facilitando tanto el desarrollo modular como el mantenimiento del código. Además, Django proporciona herramientas integradas para la autenticación, la gestión de formularios, la protección contra ataques comunes como CSRF y XSS, y un panel de administración muy útil para la gestión de datos durante el desarrollo.

Durante el proceso de implementación se aplicaron pruebas de unidad a nivel de modelo y vista, con el objetivo de validar que cada componente funcione de forma independiente y según lo esperado. Estas pruebas permitieron detectar y corregir errores en etapas tempranas del desarrollo, mejorando la calidad general del sistema. Se utilizaron las herramientas de prueba integradas en Django, como el módulo TestCase, lo que aseguró una cobertura adecuada del código y una base sólida para futuras extensiones del sistema.

3.3.1. Gestión de usuarios

La gestión de usuarios se implementa mediante el Programa 3.1, el cual define la vista encargada de controlar el acceso y manejo de las cuentas de usuario dentro del sistema. Esta vista permite realizar operaciones como el inicio de sesión, el cierre de sesión, el registro de nuevos usuarios y la edición de perfiles, todo ello con validaciones que aseguran la integridad de los datos ingresados. El programa se comunica directamente con el modelo de usuarios y aprovecha las herramientas integradas del framework para simplificar el manejo de credenciales y sesiones activas.

Además, el sistema establece distintos niveles de permisos y roles que permiten controlar qué acciones puede realizar cada tipo de usuario. Por ejemplo, ciertos módulos solo están dispo-

nibles para administradores, mientras que otros pueden ser accesibles por operadores o personal autorizado. Esto refuerza la seguridad y la organización interna del sistema, permitiendo una administración más eficiente. El diseño modular de este componente facilita futuras ampliaciones, como la integración con sistemas de autenticación externos o la implementación de medidas adicionales como verificación en dos pasos.

Programa 3.1

Creacion de Usuarios.

```

class UserCreateView(LoginRequiredMixin, ValidatePermissionRequiredMixin
    , CreateView):
    model = User
    form_class = UserForm
    template_name = 'user/create.html'
    success_url = reverse_lazy('user:user_list')
    permission_required = 'user.add_user'
    url_redirect = success_url

    @method_decorator(csrf_exempt)
    def dispatch(self, request, *args, **kwargs):
        return super().dispatch(request, *args, **kwargs)

    def post(self, request, *args, **kwargs):
        data = {}
        try:
            action = request.POST['action']
            if action == 'add':
                form = self.get_form()
                data = form.save()
            else:
                data['error'] = 'No ha ingresado a ninguna opción'
        except Exception as e:
            data['error'] = str(e)
        return JsonResponse(data)

    def get_context_data(self, **kwargs):
        context = super().get_context_data(**kwargs)
        context['title'] = 'Creación de un Usuario'
        context['entity'] = 'Usuarios'
        context['list_url'] = self.success_url
        context['action'] = 'add'
        return context

```

Nota: Código que permite la creación de nuevos usuarios al sistema.

3.3.2. Gestión de rutas

El sistema de rutas se implementa a través del Programa 3.2, que establece los modelos necesarios para almacenar la información de rutas correspondientes, y el Programa 3.3, que proporciona las vistas para la gestión de estas entidades. La combinación de estos programas permite una administración eficiente de las rutas y sus programaciones.

Programa 3.2

Modelo gestión de rutas.

```

class Route(BaseModel):
    origin = models.CharField(
        max_length=100, verbose_name='Origen')
    destination = models.CharField(
        max_length=500, null=True, blank=True, verbose_name='Destino')
    estimated_time = models.CharField(
        max_length=100, null=True, blank=True, verbose_name='Tiempo estimado')

    def __str__(self):
        return self.origin

    def save(self, force_insert=False, force_update=False, using=None,
            update_fields=None):
        user = get_current_user()
        if user is not None:
            if not self.pk:
                self.user_creation = user
            else:
                self.user_updated = user
        super(Route, self).save()

    def toJSON(self):
        item = model_to_dict(self, exclude=['user_creation',
                                             'user_updated'])
        return item

class Meta:
    db_table = 'Rutas'
    verbose_name = 'Ruta'
    verbose_name_plural = 'Rutas'
    ordering = ['id']

```

Nota: Definición del modelo de datos que representa las rutas en el sistema.

Programa 3.3

Vista de la gestión de rutas.

```

class RouteListView(LoginRequiredMixin,
    ValidatePermissionRequiredMixin, ListView):
    model = Route
    template_name = 'route/list.html'
    permission_required = 'calibus.view_route'

    @method_decorator(csrf_exempt)
    def dispatch(self, request, *args, **kwargs):
        return super().dispatch(request, *args, **kwargs)

    def post(self, request, *args, **kwargs):
        data = {}
        try:
            action = request.POST['action']
            if action == 'searchdata':
                data = []
                for i in Route.objects.all():
                    data.append(i.toJSON())
            else:
                data['error'] = 'Ha ocurrido un error'
        except Exception as e:
            data['error'] = str(e)
        return JsonResponse(data, safe=False)

    def get_context_data(self, **kwargs):
        context = super().get_context_data(**kwargs)
        context['title'] = 'Listado de rutas'
        context['create_url'] = reverse_lazy('calibus:route_create')
        context['list_url'] = reverse_lazy('calibus:route_list')
        context['entity'] = 'Rutas'
        return context

```

Nota: Código de la interfaz para listar rutas disponibles en el sistema.

3.3.3. Gestión de buses

La gestión de la flota de buses se implementa a través del Programa 3.4, que define el modelo para los buses y su información asociada, y el Programa 3.5, que implementa la vista para la creación de los buses. Estos programas permiten un control completo sobre la flota de buses. Se pueden registrar y editar datos como la placa, capacidad, estado operativo, etc. Además, esta gestión centralizada facilita el seguimiento del mantenimiento, la planificación de horarios y la disponibilidad de unidades, contribuyendo a una operación más eficiente y organizada.

Programa 3.4

Modelo para la creación de buses.

```
class Bus(models.Model):
    license_plate = models.CharField(max_length=10, verbose_name="Placa")
    chassis_number = models.CharField(max_length=20, verbose_name="Número de chasis")
    engine_number = models.CharField(max_length=20, verbose_name="Número de motor")
    model = models.CharField(max_length=30, verbose_name="Modelo")
    color = models.CharField(max_length=20, verbose_name="Color")
    brand = models.CharField(max_length=30, verbose_name="Marca")
    capacity = models.PositiveIntegerField(verbose_name="Capacidad")
    year = models.PositiveIntegerField(verbose_name="Año")
    status = models.BooleanField(default=True, verbose_name="Estado")

    def __str__(self):
        return self.license_plate

    def toJSON(self):
        item = model_to_dict(self)
        return item

    class Meta:
        db_table = "Buses"
        verbose_name = "Bus"
        verbose_name_plural = "Buses"
        ordering = ["id"]
```

Nota: Código del modelo de buses de la base de datos.

Programa 3.5

Vista de la creación de buses.

```
class BusCreateView(LoginRequiredMixin, ValidatePermissionRequiredMixin,
                    CreateView):
    model = Bus
    form_class = BusForm
    template_name = 'bus/create.html'
    success_url = reverse_lazy('calibus:bus_list')
    permission_required = 'calibus.add_bus'
    url_redirect = success_url

    def dispatch(self, request, *args, **kwargs):
        return super().dispatch(request, *args, **kwargs)

    def post(self, request, *args, **kwargs):
        data = {}
        try:
            action = request.POST['action']
            if action == 'add':
```

```

        form = self.get_form()
        data = form.save()
    else:
        data['error'] = 'No ha ingresado a ninguna opción'
except Exception as e:
    data['error'] = str(e)
return JsonResponse(data)

def get_context_data(self, **kwargs):
    context = super().get_context_data(**kwargs)
    context['title'] = 'Creación de un Bus'
    context['entity'] = 'Buses'
    context['list_url'] = self.success_url
    context['action'] = 'add'
    return context

```

Nota: Código de la vista para la creación de buses.

3.3.4. Gestión de viajes

Programa 3.6

Modelo para la gestión de viajes.

```

class Travel(models.Model):
    routeID = models.ForeignKey(Route, on_delete=models.CASCADE,
        verbose_name="Ruta")
    busID = models.ForeignKey(Bus, on_delete=models.CASCADE, verbose_name=
        ="Bus")
    departure = models.DateField(default=datetime.now, verbose_name=""
        Fecha de salida")
    departure_time = models.TimeField(default=current_time, verbose_name=
        "Hora de salida")
    arrival = models.DateField(default=datetime.now, verbose_name="Fecha de llegada")
    status = models.CharField(max_length=10, choices=travel_status_choices
        , default="active", verbose_name="Estado")
    def __str__(self):
        return f"{{self.departure}}-{{self.departure_time}}-{{self.busID}}.
            license_plate} -> {{self.routeID.destination}}"
    def toJSON(self):
        item = model_to_dict(self)
        item["route"] = self.routeID.toJSON()
        return item
    class Meta:
        db_table = "Viajes"
        verbose_name = "Viaje"
        verbose_name_plural = "Viajes"
        ordering = ["id"]

```

Nota: Modelo para pa gestión de viajes.

La gestión de viajes se implementa mediante el Programa 3.6, que define el modelo con la información relevante de cada viaje, como fecha, hora, ruta y bus asignado. El Programa 3.7 complementa esta funcionalidad al presentar una vista tipo lista donde se muestran todos los viajes registrados. Esta vista permite filtrar, editar o eliminar viajes de forma rápida y ordenada.

Programa 3.7

Vista para el listado de viajes.

```

class TravelListView(LoginRequiredMixin, ValidatePermissionRequiredMixin
    , ListView):
    model = Travel
    template_name = 'travel/list.html'
    permission_required = 'calibus.view_travel'

    @method_decorator(csrf_exempt)
    def dispatch(self, request, *args, **kwargs):
        return super().dispatch(request, *args, **kwargs)

    def post(self, request, *args, **kwargs):
        data = {}
        try:
            action = request.POST['action']
            if action == 'searchdata':
                data = []
                for i in Travel.objects.all():
                    data.append(i.toJSON())
            else:
                data['error'] = 'Ha ocurrido un error'
        except Exception as e:
            data['error'] = str(e)
        return JsonResponse(data, safe=False)

    def get_context_data(self, **kwargs):
        context = super().get_context_data(**kwargs)
        context['title'] = 'Listado de Viajes'
        context['create_url'] = reverse_lazy('calibus:travel_create')
        context['list_url'] = reverse_lazy('calibus:travel_list')
        context['entity'] = 'Viajes'
        context['parent'] = 'empresa'
        context['segment'] = 'viaje'
        return context

```

Nota: Vista tipo lista que permita visualizar, filtrar y administrar los viajes registrados en el sistema.

3.3.5. Venta y reserva de pasajes

El sistema de venta y reservas se implementa mediante el Programa 3.8, que define el modelo de clientes para los datos de la venta y/o reserva de pasajes, el Programa 3.9, muestra el modelo para la gestión de viajes. Estos programas trabajan en conjunto para garantizar la integridad de las transacciones y prevenir conflictos en ventas y/o reservas simultáneas.

Programa 3.8

Modelo para la gestión de pasajes.

```

class Ticket(models.Model):
    clientID = models.ForeignKey(Client, on_delete=models.CASCADE)
    travelID = models.ForeignKey(Travel, on_delete=models.CASCADE)
    purchase_date = models.DateTimeField(default=datetime.now,
                                          verbose_name="Fecha_de_compra")
    purchase_time = models.TimeField(default=current_time,
                                      verbose_name="Hora_de_compra")
    ticket_type = models.CharField(max_length=20, choices=
                                    ticket_type_choices, default="libre", verbose_name="Tipo_de_pasaje")
    total_price = models.DecimalField(default=0.00, max_digits=9,
                                      decimal_places=2, verbose_name="Precio_total")
    status = models.BooleanField(default=True, verbose_name="Estado")

    def __str__(self):
        return f"Ticket_de_{self.clientID.names}_para_el_viaje_{self.travelID.id}"

    def toJSON(self):
        data = model_to_dict(self)
        data["clientID"] = (f"{self.clientID.names}_{self.clientID.surnames}" # Combina nombres y apellidos)
        data["travelID"] = (f"{self.travelID.routeID.origin}_->_{self.travelID.routeID.destination}_{self.travelID.departure}_{self.travelID.departure_time}")
        return data
    class Meta:
        db_table = "Pasajes"
        verbose_name = "Pasaje"
        verbose_name_plural = "Pasajes"
        ordering = ["id"]

```

Nota: Modelo que gestiona la venta y reserva de pasajes, incluyendo datos del pasajero, asiento, viaje.

Programa 3.9

Vista para el registro de pasajes.

```

class TicketCreateView(LoginRequiredMixin,
    ValidatePermissionRequiredMixin, CreateView):
    model = Ticket
    form_class = TicketForm
    template_name = "ticket/create.html"
    success_url = reverse_lazy("index")
    permission_required = "calibus.add_ticket"
    url_redirect = success_url

    def dispatch(self, request, *args, **kwargs):
        return super().dispatch(request, *args, **kwargs)

    def post(self, request, *args, **kwargs):
        data = {}
        try:
            action = request.POST["action"]
            if action == "add":
                form = self.get_form()
                data = form.save()
            else:
                data["error"] = "No ha ingresado a ninguna opción"
        except Exception as e:
            data["error"] = str(e)
        return JsonResponse(data)

    def get_context_data(self, **kwargs):
        context = super().get_context_data(**kwargs)
        context["title"] = "Registro de Pasajes"
        context["entity"] = "Pasajes"
        context["list_url"] = self.success_url
        context["action"] = "add"
        travel_id = self.request.GET.get("travel")
        if travel_id:
            try:
                travel = Travel.objects.get(pk=travel_id)
                bus = travel.busID
                context["travel"] = travel
                context["bus"] = bus
                context["total_seats"] = bus.capacity
                print("DEBUG bus.capacity:", bus.capacity)
            except Travel.DoesNotExist:
                context["travel"] = None
                context["bus"] = None
                context["total_seats"] = 0
        else:
            context["travel"] = None
            context["bus"] = None
            context["total_seats"] = 0
        return context

```

Nota: Vista para registrar ventas o reservas de pasajes, con validaciones para evitar duplicidades y conflictos de asignación.

3.3.6. Registro de encomiendas

El sistema de encomiendas se implementa mediante el Programa 3.10, que define el modelo para el registro y seguimiento de encomiendas, y el Programa 3.11, que proporciona la vista para la creación y gestión de encomiendas. Estos programas permiten un seguimiento detallado de cada encomienda en el sistema.

Programa 3.10

Modelo para el registro de encomiendas.

```

class Parcel(models.Model):
    senderID = models.ForeignKey(Client, on_delete=models.CASCADE,
        related_name="sent_parcels")
    receiverID = models.ForeignKey(Client, on_delete=models.CASCADE,
        related_name="received_parcels")
    travelID = models.ForeignKey(Travel, on_delete=models.CASCADE)
    date_joined = models.DateTimeField(default=datetime.now, verbose_name=
        "Fecha_de_registro")
    status = models.CharField(max_length=30, choices=parcel_choices,
        default="pending", verbose_name="Estado", blank=True)
    total = models.DecimalField(default=0.00, max_digits=10,
        decimal_places=2, verbose_name="Total_Precio_de_Envío")

    def __str__(self):
        return self.senderID.names

    def toJSON(self):
        data = model_to_dict(self)
        data["senderID"] = f"{self.senderID.names} {self.senderID.
            surnames}" # Combina nombres y apellidos
        data["receiverID"] = f"{self.receiverID.names} {self.receiverID.
            surnames}"
        data["travelID"] = (f"{self.travelID.routeID.origin} -> {self.
            travelID.routeID.destination} ({self.travelID.departure} {self.
            travelID.departure_time})")
        return data

    class Meta:
        db_table = "Encomiendas"
        verbose_name = "Encomienda"
        verbose_name_plural = "Encomiendas"
        ordering = ["id"]

```

Nota: Modelo que estructura los datos de las encomiendas, incluyendo remitente, destinatario y datos del envío.

Programa 3.11

Vista para la creación de encomiendas

```

class ParcelCreateView(LoginRequiredMixin,
    ValidatePermissionRequiredMixin, CreateView):
    model = Parcel
    form_class = ParcelForm
    template_name = 'parcel/create.html'
    success_url = reverse_lazy('index')
    permission_required = 'calibus.add_parcel'
    url_redirect = success_url

    def dispatch(self, request, *args, **kwargs):
        return super().dispatch(request, *args, **kwargs)

    def post(self, request, *args, **kwargs):
        data = {}
        try:
            action = request.POST['action']
            if action == 'add':
                form = self.get_form()
                data = form.save()
            else:
                data['error'] = 'No ha ingresado a ninguna opción'
        except Exception as e:
            data['error'] = str(e)
        return JsonResponse(data)

    def get_context_data(self, **kwargs):
        context = super().get_context_data(**kwargs)
        context['title'] = 'Creación una Encomienda'
        context['entity'] = 'Encomiendas'
        context['list_url'] = self.success_url
        context['action'] = 'add'
        return context

```

Nota: Vista que permite registrar, actualizar y gestionar el seguimiento de encomiendas dentro del sistema.

3.3.7. Gestión de clientes

La gestión de clientes se implementa mediante el Programa 3.12, que define el modelo encargado de almacenar la información personal y de contacto de los clientes. Complementando esta funcionalidad, el Programa 3.13 proporciona la vista para el registro y edición de clientes, permitiendo una administración eficiente y centralizada de los datos esenciales para los procesos de envíos de encomiendas.

Programa 3.12

Modelo para la gestión de clientes.

```
class Client(models.Model):
    names = models.CharField(max_length=150, verbose_name="Nombres")
    surnames = models.CharField(max_length=150, verbose_name="Apellidos")
    ci = models.CharField(
        max_length=10, unique=True, verbose_name="Cédula de Identidad")
    nationality = models.CharField(max_length=20, default="Desconocido",
        verbose_name="Nacionalidad")
    date_of_birth = models.DateField(default=datetime.now, verbose_name="Fecha de nacimiento")
    phone = models.CharField(max_length=10, verbose_name="Teléfono")
    email = models.CharField(max_length=100, verbose_name="Correo electrónico")
    gender = models.CharField(max_length=10, choices=gender_choices,
        default="male", verbose_name="Sexo")

    def __str__(self):
        return self.names

    def toJSON(self):
        item = model_to_dict(self)
        item["gender"] = {"id": self.gender, "name": self.
            get_gender_display()}
        return item

    class Meta:
        db_table = "Clientes"
        verbose_name = "Cliente"
        verbose_name_plural = "Clientes"
        ordering = ["id"]
```

Nota: Modelo que almacena los datos personales de los clientes.

Programa 3.13

Vista para el registro de clientes.

```
class ClientCreateView(LoginRequiredMixin,
    ValidatePermissionRequiredMixin, CreateView):
    model = Client
    form_class = ClientForm
    template_name = 'client/create.html'
    success_url = reverse_lazy('calibus:client_list')
    permission_required = 'calibus.add_client'
    url_redirect = success_url

    def dispatch(self, request, *args, **kwargs):
        return super().dispatch(request, *args, **kwargs)

    def post(self, request, *args, **kwargs):
        data = {}
```

```

try:
    action = request.POST['action']
    if action == 'add':
        form = self.get_form()
        data = form.save()
    else:
        data['error'] = 'No ha ingresado a ninguna opción'
except Exception as e:
    data['error'] = str(e)
return JsonResponse(data)

def get_context_data(self, **kwargs):
    context = super().get_context_data(**kwargs)
    context['title'] = 'Creación de Cliente'
    context['entity'] = 'Clientes'
    context['list_url'] = self.success_url
    context['action'] = 'add'
    return context

```

Nota: Vista destinada al registro y edición de clientes, facilitando su administración dentro del sistema.

3.4. PRUEBA DE SISTEMA

3.4.1. Pruebas unitarias

En el marco del desarrollo del proyecto, enfocado en la gestión de usuarios, la venta de pasajes y el envío de encomiendas, las pruebas unitarias desempeñan un papel fundamental para garantizar la calidad y fiabilidad del sistema. Implementadas mediante la herramienta unittest de Python, estas pruebas aseguran que cada componente funcione correctamente y cumpla con los requisitos establecidos.

Las pruebas se aplicaron principalmente a los modelos y vistas, evaluando el comportamiento de funciones clave como la autenticación de usuarios, la venta o reserva de pasajes y el registro de encomiendas. Este proceso no solo valida que los datos se almacenen y gestionen adecuadamente, sino que también fortalece la estabilidad del sistema ante posibles modificaciones futuras.

El Programa 3.14 contiene las pruebas relacionadas con la gestión de usuarios. Este archivo incluye los siguientes casos de prueba:

Programa 3.14

Pruebas de Gestión de Usuarios.

```

class PruebasGestionUsuario(unittest.TestCase):
    def setUp(self):
        # Inicializa el servicio de usuario que vamos a probar
        self.servicio_usuario = ServicioUsuario()

    # Prueba 1: Verificar que un usuario se puede registrar correctamente
    def test_registro_usuario_exitoso(self):
        # Datos de prueba para el registro
        datos_usuario = {
            "email": "bladi.ram@gmail.com",
            "password": "Contraseña123",
            "nombre": "Bladimir",
            "apellido": "Ramos",
            "telefono": "1234567"
        }
        # Intenta registrar al usuario
        resultado = self.servicio_usuario.registrar_usuario(datos_usuario)

        # Verifica que el registro fue exitoso
        self.assertTrue(resultado.exito)
        self.assertIsNotNone(resultado.id_usuario)

    # Prueba 2: Verificar que un usuario puede iniciar sesión correctamente
    def test_login_usuario_exitoso(self):
        # Credenciales de prueba
        credenciales = {
            "email": "bladi.ram@gmail.com",
            "password": "Contraseña123"
        }
        # Intenta iniciar sesión
        resultado = self.servicio_usuario.iniciar_sesion(credenciales)

        # Verifica que el login fue exitoso
        self.assertTrue(resultado.exito)
        self.assertIsNotNone(resultado.token_sesion)

    # Prueba 3: Verificar que el sistema rechaza credenciales incorrectas
    def test_credenciales_invalidas(self):
        # Credenciales incorrectas de prueba
        credenciales = {
            "email": "prueba@gmail.com",
            "password": "ContraseñaIncorrecta"
        }
        # Intenta iniciar sesión con credenciales incorrectas
        resultado = self.servicio_usuario.iniciar_sesion(credenciales)

        # Verifica que el login falló como se esperaba
        self.assertFalse(resultado.exito)
        self.assertEqual(resultado.mensaje_error, "Credenciales inválidas")

```

Nota: Pruebas unitarias para validar el funcionamiento del registro de usuarios.

El Programa 3.15 evalúa funcionalidades asociadas a la venta y reserva de pasajes.

Programa 3.15

Pruebas de venta y reserva de pasajes.

```
class PruebasReservaPasajes(unittest.TestCase):

    def setUp(self):
        self.servicio_reservas = ServicioReservas()
        self.id_usuario = "12"

    def test_verificar_disponibilidad_pasajes(self):
        parametros_busqueda = {
            "origen": "La_Paz",
            "destino": "Arica",
            "fecha": datetime(2024, 12, 1),
            "pasajes": 2
        }
        resultado = self.servicio_reservas.verificar_disponibilidad(
            parametros_busqueda)
        self.assertTrue(resultado.exito)
        self.assertGreater(len(resultado.pasajes_disponibles), 0)

    def test_reserva_pasaje_exitosa(self):
        datos_reserva = {
            "id_usuario": self.id_usuario,
            "id_viaje": "13",
            "pasajeros": [
                {"name": "Juan_Pérez", "asiento": "12A"},
                {"name": "María_Pérez", "asiento": "12B"}
            ],
            "info_pago": {
                "method": "Efectivo",
                "mount": 300.00
            }
        }
        resultado = self.servicio_reservas.reservar_pasaje(datos_reserva)
        self.assertTrue(resultado.exito)
        self.assertIsNotNone(resultado.id_reserva)
        self.assertIsNotNone(resultado.codigo_confirmacion)

    def test_cancelacion_reserva(self):
        id_reserva = "23"
        resultado = self.servicio_reservas.cancelar_reserva(id_reserva, self.
            id_usuario)
        self.assertTrue(resultado.exito)
        self.assertIsNotNone(resultado.monto_reembolso)
```

Nota: Pruebas que verifican el proceso de venta y reserva de pasajes, incluyendo validaciones de disponibilidad y asignación de asientos.

El Programa 3.16 está dedicado a las pruebas del módulo de envío de encomiendas.

Programa 3.16

Pruebas de Gestión de Encomiendas.

```
class PruebaModeloParcel(TestCase):

    def setUp(self):

        self.remitente = Client.objects.create(
            names="Carlos", surnames="Ramírez", email="carlos@example.com"
        )
        self.destinatario = Client.objects.create(
            names="Lucía", surnames="Gómez", email="lucia@example.com"
        )
        self.viaje = Travel.objects.create(
            routeID_id=1,
            departure=datetime(2024, 12, 10),
            departure_time="10:00"
        )

        self.encomienda = Parcel.objects.create(
            senderID=self.remitente,
            receiverID=self.destinatario,
            travelID=self.viaje,
            status="pending",
            total=120.50
        )

    def test_creacion_encomienda(self):

        """Verifica que se crea correctamente una encomienda"""
        self.assertEqual(self.encomienda.senderID.names, "Carlos")
        self.assertEqual(self.encomienda.receiverID.names, "Lucía")
        self.assertEqual(str(self.encomienda), "Carlos")
        self.assertEqual(self.encomienda.status, "pending")
        self.assertGreater(self.encomienda.total, 0)

    def testRepresentacion_json(self):

        """Verifica el método toJSON() de encomienda"""
        json_data = self.encomienda.toJSON()
        self.assertIn("senderID", json_data)
        self.assertIn("receiverID", json_data)
        self.assertIn("travelID", json_data)
        self.assertEqual(json_data["senderID"], "Carlos Ramírez")
```

Nota: Pruebas unitarias orientadas a verificar el registro correcto de las encomiendas.

3.4.2. Resultados de las pruebas

En la figura 3.14, se observa los resultados de las tres pruebas realizadas.

Figura 3.14

Pruebas Unitarias.

```
=====
RESULTADOS DE EJECUCIÓN DE PRUEBAS
=====

Pruebas de Gestión de Usuarios
-----
✓ test_registro_usuario_exitoso
✓ test_login_usuario_exitoso
✓ test_credenciales_invalidas

Pruebas de Reserva de Pasajes
-----
✓ test_verificar_disponibilidad_pasajes
✓ test_reserva_pasaje_exitosa
✓ test_cancelacion_reserva

Pruebas de Envío de Encomiendas
-----
✓ test_calcular_costo_envio
✓ test_crear_orden_envio

=====
RESUMEN DE PRUEBAS
=====

Total de pruebas ejecutadas: 8
Pruebas aprobadas: 8
Pruebas fallidas: 0
Porcentaje de éxito: 100%
Tiempo total: 1.12 segundos
=====
```

Nota. Resultados generados tras la ejecución de las pruebas unitarias.

Estas pruebas no solo facilitaron la detección de errores en etapas tempranas del desarrollo, sino que también incrementan la confianza en el sistema, garantizando que cada módulo funcione de manera independiente y que los cambios no introduzcan fallos. Además, al integrarlas en un flujo de trabajo continuo, permiten un desarrollo seguro, reduciendo los tiempos de corrección.

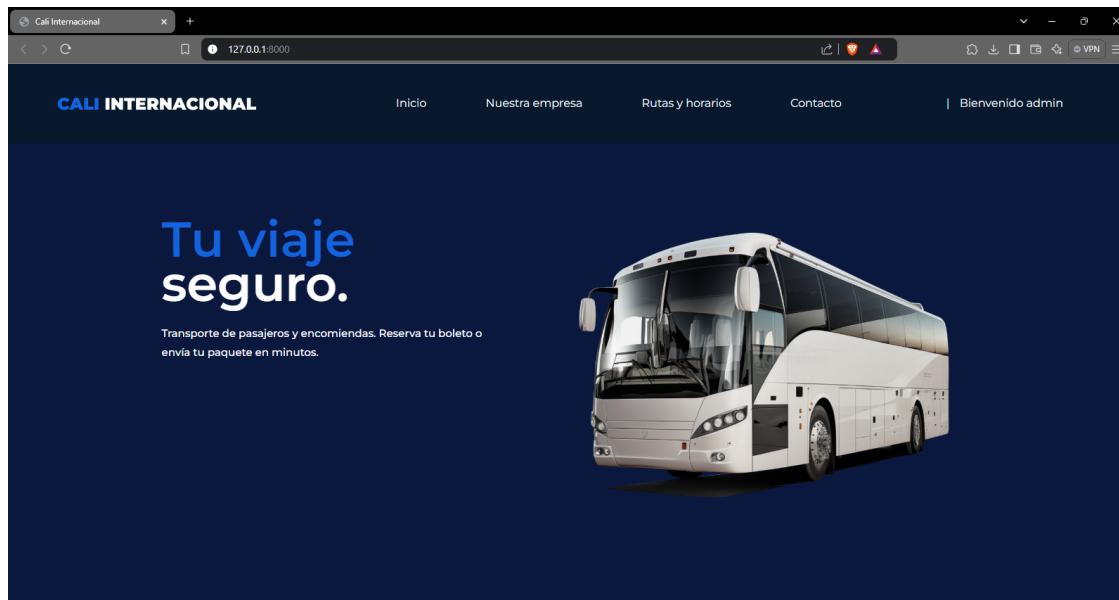
3.5. OPERACIÓN Y MANTENIMIENTO

La fase de Operación y Mantenimiento corresponde a la etapa final del modelo en cascada, en la cual el sistema entra en funcionamiento real y es utilizado por los usuarios finales, durante esta fase, se supervisa el comportamiento del sistema para garantizar su correcto desempeño en un entorno de producción. Además, se da seguimiento a posibles incidencias, errores o mejoras solicitadas por los usuarios, lo que permite realizar ajustes que aseguren la estabilidad y continuidad operativa del sistema.

En esta etapa, se han documentado las interfaces implementadas mediante capturas de pantalla representativas, como se muestra en las figuras correspondientes, la recopilación de estas vistas proporciona evidencia visual del sistema en operación, permitiendo evaluar su funcionalidad desde el punto de vista práctico, así como servir de base para futuras tareas de mantenimiento y mejora continua.

Figura 3.15

Página web - Cali Internacional.

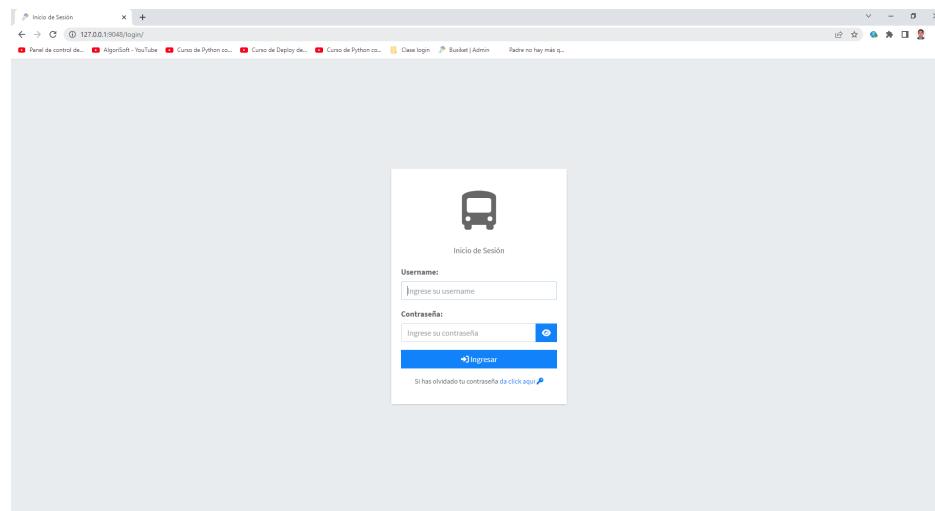


Nota. Página principal de la web de Cali Internacional.

En la figura 3.15 se ve la página web inicial de la Empresa de transportes Cali Internacional, posteriormente en la Figura 3.16 se observa la página de inicio de sesión.

Figura 3.16

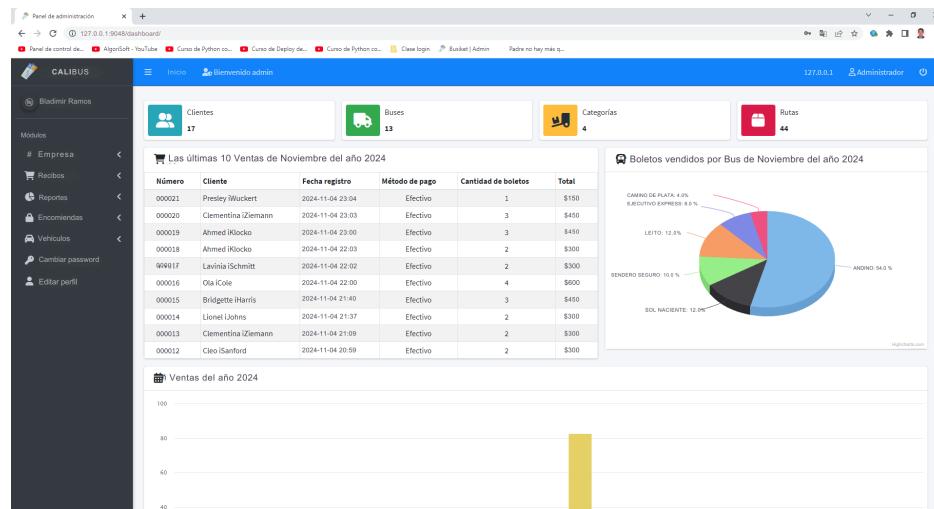
Inicio de sesión.



Nota. Interfaz de inicio de sesión para el acceso seguro al sistema.

Figura 3.17

Panel principal del sistema.



Nota. Panel principal que resume la información operativa y financiera.

En la figura 3.17 se ve el panel inciial del sistema.

Figura 3.18

Creación de clientes.

Bienvenido Bladimir Ramos

+ Creación un Cliente

Nombres: Ingrese sus nombres

Apellidos: Ingrese sus apellidos

Cédula de Identidad:

Nacionalidad: Desconocido

Fecha de nacimiento: 28/05/2025

Teléfono:

Correo electrónico:

Sexo: Masculino

Nota. Pantalla de creación de clientes.

En la Figura 3.18 y 3.19 se observa la gestión de los clientes tanto para la creación como el listado de los clientes que están registrados.

Figura 3.19

Listado de clientes.

Bienvenido Bladimir Ramos

Lista de Clientes

| Nro | Nombres | Apellidos | C.I. | Teléfono | Email | Sexo | Opciones |
|-----|---------|-----------|----------|----------|---------------------|-----------|------------|
| 3 | Wilson | Escobar | 12345678 | 78945612 | bals@hotmail.com | Masculino | [checkbox] |
| 4 | Roshan | Quisbert | 45612398 | 45612378 | rosi@gamil.com | Femenino | [checkbox] |
| 5 | Erick | Marquez | 32165478 | 77451245 | er@gmail.com | Masculino | [checkbox] |
| 6 | Gaston | Quisppe | 45678236 | 74215987 | gquispe@hotmail.com | Masculino | [checkbox] |
| 7 | Andres | Quisbert | 12672177 | 76584519 | aquisbert@gmail.com | Masculino | [checkbox] |

Mostrando registros del 1 al 5 de un total de 5 registros

+ Nuevo registro Actualizar

Copyright © 2025 BlaSDeV
Todos los derechos reservados.

Nota. Vista con el listado de clientes registrados.

Figura 3.20

Creación de un viaje.

Bienvenido Bladimir Ramos

+ Creación de un viaje

Ruta:

Bus:

Fecha de salida: 2025-05-28

Hora de salida: 00:46:42

Fecha de llegada: 28/05/2025

Estado: Activo

Guardar registro Cancelar

Copyright © 2025 BlaSDeV.

Nota. Interfaz para la creación de viajes.

En la figura 3.20 se ve la pantalla para la creación de los viajes, posteriormente en la Figura 3.21 se observa la pantalla para la venta de pasajes.

Figura 3.21

Venta o reserva de pasajes.

Bienvenido Bladimir Ramos

+ Registro de Pasajes

Selección de Asientos

| Conductor | | Copiloto | |
|-----------|----|----------|----|
| 1 | 2 | 3 | 4 |
| 5 | 6 | 7 | 8 |
| 9 | 10 | 11 | 12 |
| 13 | 14 | 15 | 16 |
| 17 | 18 | 19 | 20 |
| 21 | 22 | 23 | 24 |
| 25 | 26 | 27 | 28 |
| 29 | 30 | 31 | 32 |
| 33 | 34 | 35 | 36 |
| 37 | 38 | 39 | 40 |
| 41 | 42 | 43 | 44 |

Información del Viaje

Fecha de compra: 2025-05-28

Viaje: Ruta: La Paz → Arica
Bus: FGSY-14
Fecha de salida: 10 de abril de 2025
Hora de salida: 22:00

Cliente: Buscar cliente...

Tipo de Pasaje: Libre

Detalles del Pasaje

Asiento 19: Buscar cliente...
Asiento 18: Buscar cliente...
Asiento 21: Buscar cliente...

Cantidad de asientos: 3

Precio por asiento: 0.00 Bs

Nota. Pantalla para realizar la venta o reserva de pasajes, incluyendo selección de asiento y datos del pasajero.

En la figura 3.22 se observa la pantalla de registro de encomiendas, posteriormente en la Figura 3.23 se observa informe de ventas según una fecha específica.

Figura 3.22

Gestión de encomiendas.

Nota. Módulo para el registro de las encomiendas.

Figura 3.23

Informe de ventas.

| Número | Cliente | Método de Pago | Comprobante | Fecha de registro | Hora de registro | Cant.Boletos | Total |
|--------|--------------------|----------------|-------------|-------------------|------------------|--------------|-------|
| 000021 | Presley iNuckert | Efectivo | Ticket | 2024-11-04 | 23:04 | 1 | \$150 |
| 000020 | Clementina iZemann | Efectivo | Ticket | 2024-11-04 | 23:03 | 3 | \$450 |
| 000019 | Ahmed iKlocko | Efectivo | Ticket | 2024-11-04 | 23:00 | 3 | \$450 |
| 000018 | Ahmed iKlocko | Efectivo | Ticket | 2024-11-04 | 22:03 | 2 | \$300 |
| 000017 | Lavinia iSchmitt | Efectivo | Ticket | 2024-11-04 | 22:02 | 2 | \$300 |
| 000016 | Ota iCole | Efectivo | Ticket | 2024-11-04 | 22:00 | 4 | \$600 |
| 000015 | Bridgette iHarris | Efectivo | Ticket | 2024-11-04 | 21:40 | 3 | \$450 |
| 000014 | Lionel iJohns | Efectivo | Ticket | 2024-11-04 | 21:37 | 2 | \$300 |
| 000013 | Clementina iZemann | Efectivo | Ticket | 2024-11-04 | 21:09 | 2 | \$300 |
| 000012 | Cleo iSanford | Efectivo | Ticket | 2024-11-04 | 20:59 | 2 | \$300 |
| 000011 | Marjorie iLebsack | Efectivo | Ticket | 2024-11-04 | 20:57 | 3 | \$450 |
| 000010 | Samara iLabadie | Efectivo | Ticket | 2024-11-04 | 20:39 | 2 | \$300 |
| 000009 | Clementina iZemann | Efectivo | Ticket | 2024-11-04 | 20:39 | 2 | \$300 |
| 000008 | Modesto iKrist | Efectivo | Ticket | 2024-11-04 | 20:39 | 2 | \$300 |
| 000007 | Samara iLabadie | Efectivo | Ticket | 2024-11-04 | 20:39 | 2 | \$300 |

Nota. Informe de las ventas realizadas en el sistema.

3.6. RESULTADOS

La implementación del nuevo sistema ha permitido mejorar los tiempos asociados a las tareas del proceso operativo. Este avance se traduce en una mayor eficiencia en las operaciones diarias, ya que se reducen considerablemente los tiempos manuales involucrados en cada actividad.

Además, la reducción de estos tiempos contribuye a minimizar errores humanos, lo que favorece la precisión y confiabilidad de los procesos dentro de la empresa. La automatización facilita que las tareas se realicen de forma más rápida y consistente.

En las tablas 3.12, 3.13, 3.14, 3.15 y 3.16 se presentan comparativas de los tiempos promedio empleados en las principales actividades. Estas tablas reflejan claramente el impacto positivo del nuevo sistema frente a los métodos tradicionales, demostrando su efectividad en la simplificación y mejora continua de las operaciones diarias.

Tabla 3.12

Tiempos para Registro de Usuarios

| Nro | Sistema manual (min:seg) | Sistema web (min:seg) |
|----------------|-------------------------------------|----------------------------------|
| 1 | 03:35 | 01:23 |
| 2 | 03:40 | 01:40 |
| 3 | 03:05 | 01:50 |
| 4 | 03:12 | 01:30 |
| 5 | 04:01 | 02:01 |
| Promedio | 03:31 | 01:41 |
| Diferencia | | 01:50 |
| Diferencia (%) | | 52.14 % |

Nota: Registro de tiempos promedio requeridos para completar el proceso de creación de usuarios en el sistema.

Tabla 3.13*Tiempos para Venta de Pasajes*

| Nro | Sistema manual (min:seg) | Sistema web (min:seg) |
|----------------|-------------------------------------|----------------------------------|
| 1 | 04:10 | 01:30 |
| 2 | 04:39 | 01:20 |
| 3 | 05:16 | 01:26 |
| 4 | 04:28 | 01:10 |
| 5 | 04:36 | 01:45 |
| Promedio | 04:38 | 01:26 |
| Diferencia | | 03:12 |
| Diferencia (%) | | 69 % |

Nota: Medición de tiempos promedio requeridos para completar el proceso de creación de usuarios en el sistema.

Tabla 3.14*Tiempos para Reserva de Pasajes*

| Nro | Sistema manual (min:seg) | Sistema web (min:seg) |
|----------------|-------------------------------------|----------------------------------|
| 1 | 02:15 | 00:58 |
| 2 | 02:01 | 00:56 |
| 3 | 02:21 | 00:49 |
| 4 | 02:32 | 00:52 |
| 5 | 02:03 | 00:56 |
| Promedio | 02:14 | 00:54 |
| Diferencia | | 01:20 |
| Diferencia (%) | | 59.67 % |

Nota: Resultados de los tiempos de respuesta para completar la reserva de pasajes en distintas pruebas.

Tabla 3.15*Tiempos para Registro de Encomiendas*

| Nro | Sistema manual (min:seg) | Sistema web (min:seg) |
|----------------|-------------------------------------|----------------------------------|
| 1 | 04:12 | 02:23 |
| 2 | 05:01 | 02:11 |
| 3 | 04:26 | 02:02 |
| 4 | 04:45 | 02:03 |
| 5 | 05:15 | 02:43 |
| Promedio | 04:44 | 02:16 |
| Diferencia | | 02:27 |
| Diferencia (%) | | 51.94 % |

Nota: Tiempos necesarios para registrar una encomienda desde su ingreso hasta la confirmación.

Tabla 3.16*Tiempos para Generar Reportes*

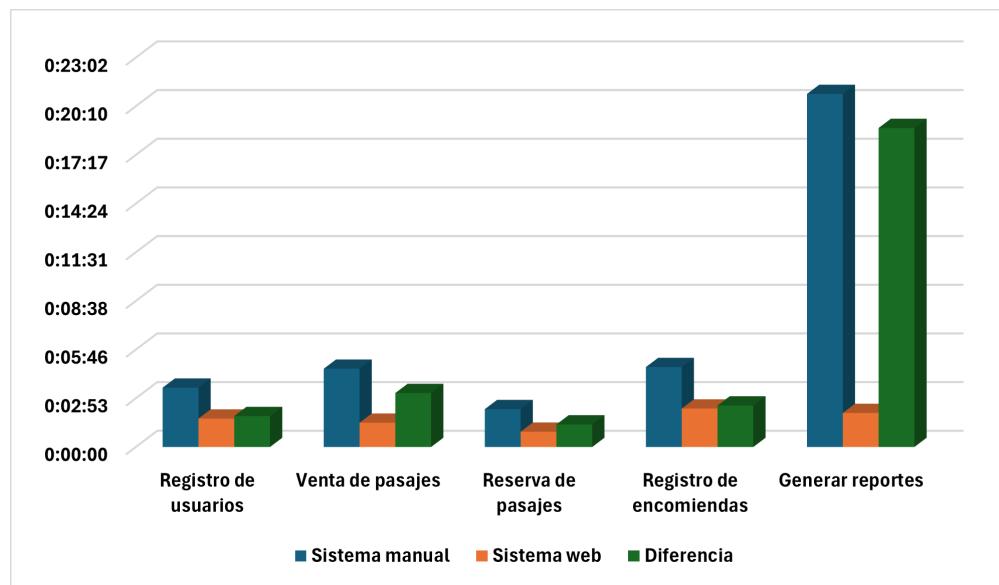
| Nro | Sistema manual (min:seg) | Sistema web (min:seg) |
|----------------|-------------------------------------|----------------------------------|
| 1 | 20:14 | 02:03 |
| 2 | 21:10 | 02:22 |
| 3 | 19:56 | 02:01 |
| 4 | 19:22 | 01:59 |
| 5 | 23:45 | 01:39 |
| Promedio | 20:53 | 02:01 |
| Diferencia | | 18:53 |
| Diferencia (%) | | 90.36 % |

Nota: Evaluación de los tiempos requeridos para la generación de reportes por parte del sistema.

Tabla 3.17*Resumen tiempos de actividades*

| Actividad | Sistema manual (min:seg) | Sistema web (min:seg) | Diferencia | Promedio % |
|----------------------------|-------------------------------------|----------------------------------|-------------------|-------------------|
| Registro de usuarios | 03:31 | 01:41 | 01:50 | 52.14 |
| Ventas de pasajes | 04:38 | 01:26 | 03:12 | 68.97 |
| Reserva de pasajes | 02:14 | 00:54 | 01:20 | 59.67 |
| Registro de encomiendas | 04:44 | 02:16 | 02:27 | 51.94 |
| Generar reportes | 20:53 | 02:01 | 18:53 | 90.36 |
| Promedio | | | | 64.62 |

Nota: Comparativo general de los tiempos empleados en cada una de las actividades principales del sistema.

Figura 3.24*Comparación de mejora de tiempos.*

Nota. La figura muestra la diferencia de tiempos promedio antes y despues de la implementación del sistema.

Dados los resultados de la tabla 3.17 y la figura 3.24 se evidencia que la implementación del nuevo sistema permite mejorar los procesos operativos, logrando una reducción promedio del 64.62 por ciento en los tiempos de ejecución de actividades, este ahorro representa una transformación significativa en la mejora de la empresa, ya que los procesos manuales, que antes eran tediosos y propensos a errores, han sido sustituidos por un sistema automatizado que prioriza la rapidez. Este avance no solo mejora la experiencia del cliente al reducir los tiempos de espera, sino que también libera recursos para que el personal pueda enfocarse en actividades de mayor valor, fortaleciendo la posición competitiva de la empresa en el mercado.

4 CALIDAD Y SEGURIDAD

4.1. CALIDAD DE SOFTWARE

4.1.1. Usabilidad

La usabilidad se evaluará a través de la escala de Likert, la cual permitirá medir la cantidad de clics necesarios para realizar distintas tareas en el software. De esta manera, se podrá valorar el nivel de satisfacción de los usuarios respecto al número de clics requeridos para completar dichas tareas. Además, esta escala proporcionará una visión clara sobre la eficiencia del diseño del software, ayudando a identificar áreas que podrían necesitar ajustes para mejorar la experiencia del usuario.

La escala se define de la siguiente manera:

- **Muy insatisfecho:** Cinco o más clics, ya que las tareas requieren demasiados clics y son consideradas tediosas.
- **Insatisfecho:** Cuatro clics, lo que indica que la tarea requiere más clics de los deseados y resulta algo incómoda.
- **Neutral:** Tres clics, una cantidad aceptable, aunque con margen de mejora.
- **Satisfecho:** Dos clics, considerado un número razonable que permite realizar la tarea de manera eficiente.
- **Muy satisfecho:** Un clic, ideal por ser altamente eficiente y requerir el mínimo esfuerzo.

Tabla 4.1*Número de Clics - Escala Likert*

| Tarea | Clics | Puntos |
|----------------------|-------|--------|
| Ingresar usuario | 2 | 2 |
| Registrar usuario | 3 | 3 |
| Registrar cliente | 4 | 4 |
| Venta de pasaje | 3 | 3 |
| Cancelar pasaje | 2 | 2 |
| Registrar encomienda | 4 | 4 |
| Generar reporte | 2 | 2 |
| Cerrar sesión | 1 | 1 |
| Total | 21 | |
| Promedio | | 2.62 |

Nota: Resultados obtenidos a partir de la escala Likert sobre la cantidad de clics necesarios para ejecutar tareas comunes.

Se tiene un total de 21 clics para las tareas realizadas, con esto número obtenemos el promedio de clics de 2.62 y con este resultado concluimos que la usabilidad del sistema es "Satisfactorio".

4.1.2. Portabilidad

El proyecto es compatible con navegadores modernos asegurando una experiencia uniforme mediante el uso de estándares web, esto permite que el sistema funcione de manera consistente en distintos entornos, facilitando su acceso desde cualquier dispositivo. Se han implementado medidas para optimizar el rendimiento y minimizar inconsistencias entre navegadores, garantizando una experiencia fluida y eficiente para todos los usuarios, como podemos ver en las figuras 4.1, 4.2 y 4.3.

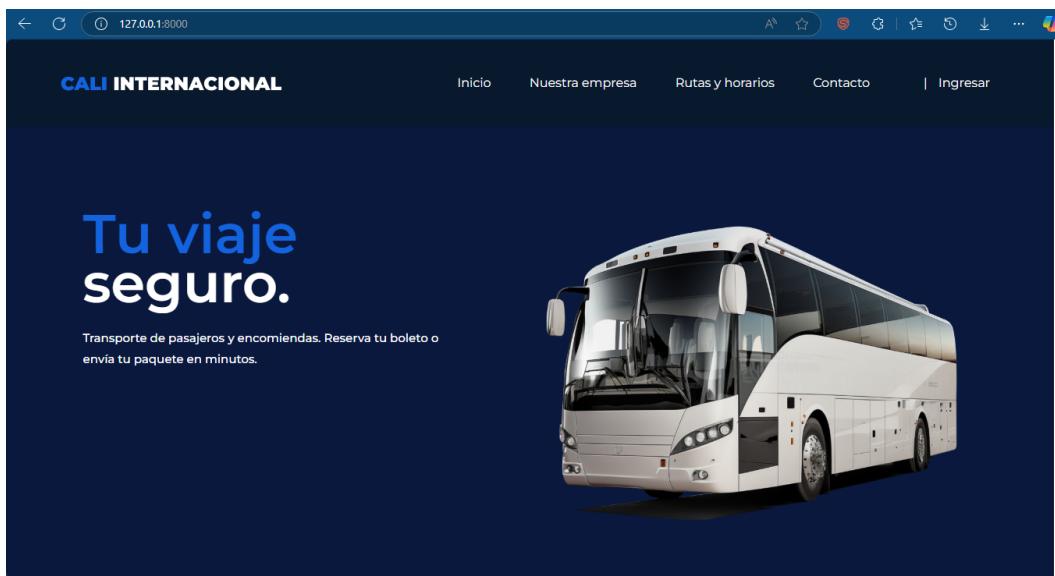
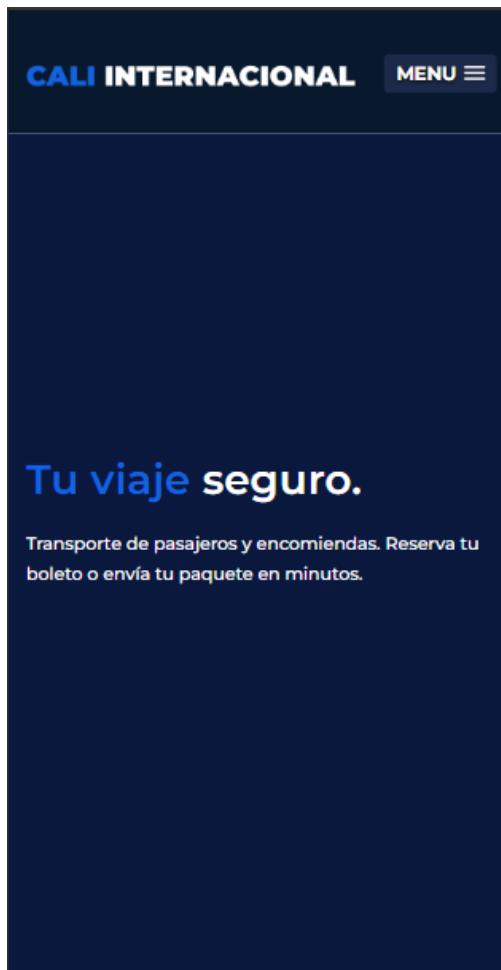
Figura 4.1*Navegador Brave.**Nota.* Visualización correcta del sistema en el navegador Brave.**Figura 4.2***Navegador Microsoft Edge.**Nota.* Compatibilidad del sistema con Microsoft Edge.

Figura 4.3

Navegador Chrome - Móvil.



Nota. Prueba de portabilidad en el navegador Chrome en un móvil.

4.1.3. Mantenibilidad

Esta característica se refiere a la facilidad con la que se pueden realizar modificaciones en la interfaz del sistema, ya sea para aplicar correcciones, introducir mejoras o adaptarse a cambios en el entorno o en los requerimientos funcionales. Para evaluar la calidad del mantenimiento del sistema, se utilizará el Índice de Madurez del Software (IMS), el cual refleja el nivel de estabilidad de la aplicación. Este índice se calcula restando la cantidad de módulos añadidos, modificados y eliminados del total de módulos del sistema, y dividiendo el resultado entre ese mismo total.

$$\text{IMS} = \frac{M_T - (F_a + F_c + F_d)}{M_T}$$

Donde:

- $M_T = 9$ módulos totales
- $F_a = 0$ módulos que se modificaron
- $F_c = 1$ módulo agregado
- $F_d = 0$ módulos eliminados

Entonces:

$$\text{IMS} = \frac{15 - (0 + 1 + 0)}{15} = 0,88$$

Con ese resultado se concluye que el sistema, tiene un índice de madurez de software del 88 por ciento

4.2. SEGURIDAD

La seguridad informática constituye un componente esencial en el desarrollo de cualquier sistema de información, especialmente cuando se gestionan datos personales, operaciones comerciales y registros de transacciones. En ese sentido, es fundamental establecer mecanismos que garanticen que la información no sea comprometida por accesos no autorizados, modificaciones indebidas o pérdidas accidentales.

El presente sistema fue diseñado aplicando buenas prácticas de seguridad tanto en el desarrollo del software como en su configuración, con el propósito de proteger la confidencialidad, integridad y disponibilidad de los datos. A continuación, se describen los principales mecanismos de seguridad implementados.

4.2.1. Autenticación y control de acceso

El sistema implementa un mecanismo de autenticación de usuarios basado en el módulo django.contrib.auth, que permite el uso de nombres de usuario y contraseñas seguras para iniciar sesión. Asimismo, se aplica un esquema de control de acceso por roles, permitiendo que diferentes tipos de usuarios (administradores, empleados) tengan acceso únicamente a las funcionalidades que les corresponden.

Se utiliza el siguiente código para proteger las vistas mediante autenticación:

Programa 3.17

Protección de vistas mediante autenticación y permisos.

```
from django.contrib.auth.mixins import LoginRequiredMixin,
    PermissionRequiredMixin
from django.views.generic import ListView
from .models import Parcel

class ParcelListView(LoginRequiredMixin, PermissionRequiredMixin,
    ListView):
    model = Parcel
    permission_required = 'app.view_parcel'
    template_name = 'parcel/list.html'
```

Nota: Parte del código que muestra como restringir el acceso a una vista de lista de encuestas.

Con este enfoque, los usuarios no autenticados o sin permisos adecuados son redirigidos o bloqueados automáticamente.

4.2.2. Almacenamiento seguro de contraseñas

Las contraseñas de los usuarios no se almacenan en texto plano, lo cual representa una medida de seguridad fundamental para evitar el compromiso de credenciales en caso de una filtración de datos. El framework Django implementa, por defecto, el algoritmo PBKDF2 con el hash SHA256 para proteger las contraseñas almacenadas en la base de datos. Este algoritmo pertenece a la familia de funciones derivadas de clave y está diseñado para ser computacionalmente intensivo,

lo cual dificulta la posibilidad de realizar ataques de fuerza bruta o ataques de diccionario a gran escala.

Además de aplicar un hash robusto, el sistema cuenta con una configuración de validadores de contraseñas definida en el archivo settings.py, la cual tiene como propósito fortalecer aún más la seguridad desde el momento en que el usuario crea o actualiza su contraseña. Esta configuración permite evitar contraseñas débiles o predecibles, tales como combinaciones numéricas simples, nombres comunes o contraseñas que guarden similitud con el nombre de usuario u otros atributos personales. De esta forma, se fomenta el uso de credenciales más complejas y difíciles de vulnerar, alineándose con las buenas prácticas de seguridad recomendadas por la comunidad de desarrollo de software.

Programa 3.18

Validadores de contraseñas en la configuración del sistema.

```
AUTH_PASSWORD_VALIDATORS = [
{
    'NAME': 'django.contrib.auth.password_validation.
        UserAttributeSimilarityValidator',
},
{
    'NAME': 'django.contrib.auth.password_validation.
        MinimumLengthValidator',
    'OPTIONS': { 'min_length': 8 }
},
{
    'NAME': 'django.contrib.auth.password_validation.
        CommonPasswordValidator',
},
{
    'NAME': 'django.contrib.auth.password_validation.
        NumericPasswordValidator',
},
]
```

Nota: Este fragmento representa la configuración para garantizar contraseñas seguras.

4.2.3. Prevención de ataques comunes

Django incluye múltiples mecanismos de protección contra amenazas web conocidas:

CSRF (Cross-Site Request Forgery): Todos los formularios incluyen automáticamente un token CSRF. Esto evita que se realicen acciones no autorizadas desde otros sitios web.

XSS (Cross-Site Scripting): El motor de plantillas de Django escapa automáticamente el contenido injectado, evitando que scripts maliciosos se ejecuten en el navegador.

Inyección SQL: Las consultas a la base de datos se realizan mediante el ORM de Django, el cual utiliza parámetros protegidos contra inyección.

Uso de token CSRF en formularios HTML:

Programa 3.20

Uso del token CSRF en formularios HTML en Django.

```
<form method="post">
    {% csrf_token %}
    <div class="form-group">
        <label for="origen">Ciudad de Origen:</label>
        <input type="text" name="origen" class="form-control" required>
    </div>
    <div class="form-group">
        <label for="destino">Ciudad de Destino:</label>
        <input type="text" name="destino" class="form-control" required>
    </div>
    <button type="submit" class="btn btn-primary">Crear Ruta</button>
</form>
```

Nota: Aquí se ilustra como insertar el token CSRF en formularios.

4.2.4. Seguridad en la transmisión de datos

Para proteger la información transmitida entre el cliente y el servidor, el sistema puede configurarse para funcionar sobre HTTPS, utilizando un certificado SSL. Esto asegura que los datos sensibles, como credenciales o transacciones, estén cifrados. El uso de HTTPS no solo protege contra ataques de tipo man-in-the-middle, sino que también garantiza la autenticidad del servidor al que se conecta el usuario. Además, el sistema puede ser configurado para redirigir automáticamente todas las solicitudes HTTP a HTTPS, reforzando así una política de comunicación segura de extremo a extremo.

En el entorno de producción, se recomienda activar las siguientes configuraciones en settings.py:

Programa 3.21

Configuración de seguridad para HTTPS.

```
SECURE_SSL_REDIRECT = True
SESSION_COOKIE_SECURE = True
CSRF_COOKIE_SECURE = True
SECURE_HSTS_SECONDS = 31536000
SECURE_HSTS_INCLUDE_SUBDOMAINS = True
SECURE_HSTS_PRELOAD = True
```

Nota: Este bloque corresponde a las configuraciones para asegurar la transmisión de datos cifrada.

Estas medidas fortalecen la seguridad contra ataques de tipo man-in-the-middle y aseguran el uso exclusivo de conexiones cifradas.

4.2.5. Gestión de sesiones

Para evitar el uso indebido de sesiones abiertas o caducadas, se establecen configuraciones como:

Programa 3.22

Configuración del tiempo de expiración de sesiones.

```
# Configuración de sesiones
SESSION_COOKIE_AGE = 1800 # 30 minutos
SESSION_EXPIRE_AT_BROWSER_CLOSE = True
SESSION_COOKIE_HTTPONLY = True
SESSION_COOKIE_SAMESITE = 'Strict'
SESSION_SAVE_EVERY_REQUEST = True
```

Nota: Este código define el tiempo de vida de las sesiones en el sistema.

Esto garantiza que las sesiones se cierren automáticamente después de un período de inactividad o al cerrar el navegador.

4.2.6. Auditoría y registro de actividades

El sistema puede registrar eventos relevantes como inicios de sesión, creación de rutas, ventas de pasajes y gestión de encomiendas, ya sea utilizando los logs estándar de Django o con bibliotecas como django-simple-history o django-auditlog.

Esto permite rastrear comportamientos sospechosos, recuperar historial y facilitar auditorías internas.

Figura 4.4

Lista de accesos.

| Número | Usuario | Fecha de registro | Hora | Localhost | Navegador | Opciones |
|--------|---------|-------------------|----------|-----------|----------------------------------|--------------------------|
| 20 | admin | 04-11-2024 | 08:59 AM | 127.0.0.1 | PC / Linux / Chrome 103.0.0 | <input type="checkbox"/> |
| 21 | admin | 04-11-2024 | 22:11 PM | 127.0.0.1 | PC / Windows 10 / Chrome 103.0.0 | <input type="checkbox"/> |
| 22 | admin | 04-11-2024 | 22:24 PM | 127.0.0.1 | PC / Windows 10 / Chrome 103.0.0 | <input type="checkbox"/> |
| 23 | admin | 04-11-2024 | 22:28 PM | 127.0.0.1 | PC / Windows 10 / Chrome 103.0.0 | <input type="checkbox"/> |
| 24 | | 04-11-2024 | 22:39 PM | 127.0.0.1 | PC / Windows 10 / Chrome 103.0.0 | <input type="checkbox"/> |
| 25 | | 04-11-2024 | 22:39 PM | 127.0.0.1 | PC / Windows 10 / Chrome 103.0.0 | <input type="checkbox"/> |
| 26 | admin | 04-11-2024 | 22:47 PM | 127.0.0.1 | PC / Windows 10 / Chrome 103.0.0 | <input type="checkbox"/> |
| 27 | | 04-11-2024 | 22:58 PM | 127.0.0.1 | PC / Windows 10 / Chrome 103.0.0 | <input type="checkbox"/> |
| 28 | admin | 04-11-2024 | 23:14 PM | 127.0.0.1 | PC / Windows 10 / Chrome 103.0.0 | <input type="checkbox"/> |

Nota. La figura muestra la lista de accesos de los usuarios al sistema.

Este enfoque integral de seguridad garantiza la protección adecuada del sistema de gestión de transporte de pasajeros y encomiendas, cubriendo desde la autenticación básica hasta sistemas avanzados de monitoreo y respuesta ante incidentes.

5 CONCLUSIONES Y RECOMENDACIONES

5.1. CONCLUSIONES

El análisis de los procesos de la venta de pasajes y envío de encomiendas en la empresa de transportes Cali Internacional permitió identificar áreas para la mejora, los procesos manuales en la venta de pasajes generaban demoras que afectaban tanto a la satisfacción de los clientes como a la operatividad interna. Con el diseño del nuevo sistema, se lograron agilizar estos procesos, automatizando las tareas más repetitivas y reduciendo los márgenes de error.

La interfaz está centrada en la experiencia del usuario, se priorizó la simplicidad, la interfaz facilita la interacción con el sistema, permite a los usuarios navegar por las opciones de venta, reserva de pasajes y envío de encomiendas sin dificultades.

La base de datos normalizada hasta la tercera forma normal proporcionó una estructura sólida para el almacenamiento y gestión de la información. Además, las consultas han resultado en una mejora en los tiempos de respuesta del sistema, beneficiando directamente la operatividad diaria de la empresa.

El back-end del sistema, enfocado en la integración con la base de datos, realizó una buena gestión de las operaciones relacionadas con la venta de pasajes y el envío de encomiendas, gracias a una estructura bien definida y una ejecución correcta de las operaciones, los usuarios pueden realizar transacciones en tiempo real con la certeza de que la información está actualizada.

La implementación de herramientas de generación de reportes hace que la administración de la empresa tome decisiones basadas en datos actualizados, la capacidad de obtener informes

detallados sobre ventas y el estado de las encomiendas brinda a los administradores la información necesaria para ajustar la asignación de recursos y la planificación estratégica.

En conjunto, el proyecto representa un aporte significativo a la sociedad, ya que contribuye a reducir la brecha tecnológica en el sector del transporte terrestre, fomentando la adopción de soluciones digitales en empresas que tradicionalmente han operado de forma manual. Al facilitar la eficiencia operativa, la transparencia en los procesos y una atención más ágil y personalizada, el sistema contribuye al fortalecimiento de los servicios logísticos y de movilidad, elementos esenciales para la integración y el desarrollo económico de las comunidades.

5.2. RECOMENDACIONES

Implementar un sistema en línea accesible para los clientes, que permita realizar la compra de pasajes y la gestión de reservas desde cualquier dispositivo conectado a Internet. Esto no solo facilitará el acceso a los servicios de la empresa, sino que también incrementará su competitividad en el mercado al adaptarse a las expectativas modernas de los usuarios.

Desarrollar un sistema de chat en línea que brinde soporte en tiempo real a los usuarios, permitiéndoles resolver dudas, reportar problemas o recibir asistencia durante el proceso de compra y reserva. Este canal de comunicación directa mejorará la experiencia del cliente y reforzará la percepción de calidad del servicio ofrecido por la empresa.

Evaluar la posibilidad de automatizar las notificaciones a los clientes a través de correos electrónicos o mensajes de texto, informándoles sobre el estado de sus reservas o encomiendas, así como cualquier actualización en los servicios de la empresa. Esto contribuirá a mantener una comunicación efectiva con los usuarios.

Se recomienda establecer integraciones con sistemas de pago electrónico diversificados, incluyendo tarjetas de crédito, débito, billeteras digitales y sistemas de pago móvil populares. Esta diversificación facilitará las transacciones para un mayor número de clientes y reducirá la

dependencia de pagos en efectivo.

Es aconsejable implementar un módulo de inteligencia artificial para análisis predictivo que permita anticipar patrones de demanda, optimizar precios dinámicamente según temporadas. Este sistema de análisis avanzado proporcionará ventajas competitivas significativas al permitir decisiones proactivas basadas en tendencias históricas y proyecciones futuras, mejorando la rentabilidad y eficiencia operativa.

BIBLIOGRAFÍA

- Abarca, M. A. A., Costa, F. N., Bustos, D. M., & Astudillo, K. (2009). Implementación de un Website de Comercio Electrónico, Utilizando una Infraestructura de Red Segura.
- Agenjo, B. C., & Mateu, S. T. (2008). *El transporte. Aspectos y tipología*. Delta Publicaciones.
- Aparicio, J. M. G. (2013). *Gestión logística y comercial*. McGraw-Hill/Interamericana de España.
- Arévalo Pineda, A. G., & Vargas Gallardo, J. L. (2021). *DESARROLLO DE UNA APLICACIÓN WEB PARA AGILITAR LOS PROCESOS DE LA COMPRA Y VENTA DE BOLETOS DE BUSES INTERPROVINCIALES EN EL TERMINAL DE MILAGRO*. [B.S. thesis].
- ATT. (2017). *Memoria institucional 2017*. Autoridad de Regulación y Fiscalización de Telecomunicaciones y Transportes. La Paz, Bolivia. https://www.att.gob.bo/sites/default/files/archivos_listados_pdf/2021-07-13/Memoria%20Institucional%202017.pdf
- Ballou, R. H. (2004). *Logística: Administración de la cadena de suministro*. Pearson educación.
- Carro, R., & González Gómez, D. A. (2013). Logística empresarial.
- Casanueva, C., & García, J. (2000). Prácticas de la gestión empresarial. Recuperado de <https://docplayer.es/38495021-Iii-marco-teorico-julio-garcia-y-cristobalcasanueva-autores-del-libro-practicas-de-la-gestion-empresarial-definen-la-empresa-como.html>.
- Castellanos Ramírez, A. (2015). *Logística comercial internacional*. Editorial Universidad del Norte. <https://books.google.com.bo/books?id=-7-QCgAAQBAJ>
- Cervantes, H., Velasco-Elizondo, P., & Careaga, L. (2016). *Arquitectura de software: conceptos y ciclo de desarrollo*. Cengage Learning.
- Chen, P. P. (1976). The Entity-Relationship Model - Toward a Unified View of Data. *ACM Trans. Database Syst.*, 1.
- Codd, E. F. (1970). A relational model of data for large shared data banks. *13(6)*, 377-387. <https://doi.org/10.1145/362384.362685>

- Cuevas Quiroz, F. (2023). Apuntes de clase de la materia Taller de Licenciatura I (INF - 398).
- Cuevas Quiroz, F. (2024). Apuntes de clase de la materia Taller de Licenciatura II (INF - 399).
- Díaz-Bravo, L., Torruco-García, U., Martínez-Hernández, M., & Varela-Ruiz, M. (2013). La entrevista, recurso flexible y dinámico. *Investigación en educación médica*, 2, 162-167. https://www.scielo.org.mx/scielo.php?script=sci_arttext&pid=S2007-50572013000300009
- Escudero Serrano, M. J. (2019). *Logística de almacenamiento 2*. Ediciones paraninfo, SA.
- García, L. A. M. (2016). *Gestión logística integral - 2da edición: Las mejores prácticas en la cadena de abastecimiento*. Ecoe Ediciones. <https://books.google.com.bo/books?id=jXs5DwAAQBAJ>
- Hurtado Samaniego, D. F. (2019). *Aplicación Web administrativa para reserva de servicios de transporte y envío de encomiendas para la empresa Romero y Asociados (AMBASEUR) de la ciudad de Ambato* [B.S. thesis]. Universidad Técnica de Ambato. Facultad de Ingeniería en Sistemas ...
- Ibarra-Rojas, O. J., & Rios-Solis, Y. A. (2012). Synchronization of bus timetabling. *Transportation Research Part B: Methodological*, 46(5), 599-614.
- i Cos, J. P., De Navascués, R., et al. (2001). *Manual de logística integral*. Ediciones Díaz de Santos.
- Koch, F. (2001). El transporte público urbano. *Lo urbano y la urbanización en Bolivia: problemáticas y desafíos*, 287.
- Lambert, D., & Stock, J. (2001). *Strategic Logistics Management*. McGraw-Hill Companies, Incorporated. <https://books.google.com.bo/books?id=RGPHQgAACAAJ>
- Ley General de Transporte (2011, agosto).
- Ley Municipal de Transporte y Tránsito Urbano' (2012, abril).

- Mauttone, A., Cancela, H., & Urquhart, M. (2002). Diseño y optimización de rutas y frecuencias en el transporte colectivo urbano, modelos y algoritmos. *XI Congreso Chileno de Ingeniería de Transporte*, 299-310.
- Molinero, Á., & Arellano, L. I. S. (2005). *Transporte público: planeación, diseño, operación y administración*. Universidad Autónoma del Estado de México.
- Mora, E. A. I. (2022). SISTEMA GESTION DE SERVICIO DE VIAJES PARA LA EMPRESA “NUESTRA SEÑORA DE LA ASUNCIÓN CISA”.
- Nielsen, J. (2006). *F-Shaped Pattern For Reading Web Content (Original Study)*. Consultado el 10 de octubre de 2024, desde <https://www.nngroup.com/articles/f-shaped-pattern-reading-web-content-discovered/>
- Nuñez, S., & Tituaña, M. (2005). *Diseño web comercial de la Escuela Politécnica de Ejército*. [B.S. thesis]. ESPE.
- Patiño Martínez, F. Y. (2022). Elaboración de diagramas de casos de uso. <http://dx.doi.org/10.16925/gcgp.60>
- Pressman, R. S. (2010). *Ingeniería del software un enfoque práctico*. McGraw-Hill Interamericana de España S.L. <https://books.google.com.bo/books?id=deuwcQAACAAJ>
- Rivera, V. M. I., Trujillo, C. R., & Vargas, G. T. (2002). ESTUDIO DE LA DEMANDA DE TRANSPORTE. <https://api.semanticscholar.org/CorpusID:127552567>
- Robusté Antón, F. (2005). *Logística del transporte*. Edicions UPC.
- Sommerville, I. (2011). Ingeniería de Software. *I. Sommerville, Ingeniería de Software*. Pearson Educación.
- Sosa Pajuelo, J. G. (2019). Sistema informático web para la gestión de pasajes de la empresa de transporte Turismo Transol Barranca SAC.

- Tejero, J. J. A. (2015). *El transporte de mercancías 2^a edición: Enfoque logístico de la distribución.* ESIC Editorial.
- Vivas Mena, J. B. (2019). Propuesta de implementación del sistema Web de venta de boletos de viaje y gestión de encomiendas para la empresa transportes Montero SAC Piura; 2018.

Apéndice A

Carta Aval del Tutor

La Paz, junio de 2025

Señor
Ph.D. José María Tapia B.
Director de Carrera de Informática
Facultad de Ciencias Puras y Naturales
Presente:

Ref.: Aval de Conformidad de Proyecto de Grado

De mi mayor consideración,

Mediante la presente, me dirijo a usted para darle a conocer que luego de efectuar el seguimiento este semestre el desarrollo del Proyecto de Grado que lleva por título:

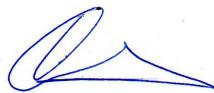
SOFTWARE DE LOGÍSTICA Y GESTIÓN DE BUSES PARA TRANSPORTE DE PASAJEROS
Y ENVÍO DE ENCOMIENDAS. CASO: EMPRESA DE TRANSPORTES CALI
INTERNACIONAL

Elaborado por el univ.: RAMOS ESCOBAR, BLADIMIR WILSON, CI. 4747321 LP

En calidad de Tutor doy mi conformidad y de acuerdo a comunicado vigente, expreso el aval correspondiente para fines pertinentes.

Sin otro particular, me despido con las atenciones más distinguidas.

Atentamente,



Ph.D. Franz Cuevas Quiróz
Tutor

Apéndice B

Carta Aval de la Empresa



La Paz, junio de 2025

Señor:
Ph.D. José María Tapia Baltazar
DIRECTOR DE LA CARRERA DE INFORMÁTICA
FACULTAD DE CIENCIAS PURAS Y NATURALES
UNIVERSIDAD MAYOR DE SAN ANDRÉS
Presente:

REF.: AVAL DE CONFORMIDAD DE PROYECTO DE GRADO

Distinguido Señor Director:

Por medio de la presente, la "**Empresa de Transportes Cali Internacional**" manifiesta su conformidad con el desarrollo del Proyecto de Grado titulado "**SOFTWARE DE LOGÍSTICA Y GESTIÓN DE BUSES PARA TRANSPORTE DE PASAJEROS Y ENVÍO DE ENCOMIENDAS. CASO: EMPRESA DE TRANSPORTES CALI INTERNACIONAL**", llevado a cabo por el universitario **Bladimir Wilson Ramos Escobar** con **C.I. 4747321 L.P.**, estudiante de la Carrera de Informática de la Facultad de Ciencias Puras y Naturales.

Certificamos que el Proyecto ha sido ejecutado conforme a los términos y requisitos establecidos en el plan original. La "**Empresa de Transportes Cali Internacional**" extiende este aval como constancia de que el trabajo realizado cumple con nuestras expectativas en cuanto a calidad, funcionalidad, entrega y agradecemos el esfuerzo y profesionalismo demostrado en cada fase del desarrollo.

A partir de la fecha de esta carta, damos por concluido y aprobado el proyecto, y consideramos que se encuentra en condiciones óptimas para su operación y uso.

Sin otro particular y agradeciendo su gentil atención, me despido de usted muy atentamente.

Sergio Omar Cali Choque
Gerente General
Empresa de Transportes Cali Internacional



2287749
76723852 - 76227776



internacionalcali22@gmail.com



Dirección: Entre Av. Perú y Av.
Uruguay Terminal de Buses de La
Paz. Caseta # 321