

**UNIVERSIDAD TECNÓLOGICA PRIVADA DE SANTA CRUZ**

**FACULTAD DE CIENCIAS Y TECNOLOGÍAS**

CARRERA: **INGENIERÍA DE SISTEMAS**

**CASO DE ESTUDIO**:

**SISTEMA ADMINISTRATIVO BASADO EN EL MÉTODO SISTÉMICO DE CONTROL Y MEJORA ORGANIZACIONAL PARA MEJORAR LA GESTIÓN DE VENTAS DE LA EMPRESA DE TRANSPORTES “FLOTA MARISCAL SANTA CRUZ”**

**AUTOR: DANIEL GUILLERMO GORIANZ FERRUFINO**

**DEFENSA DEL CASO DE ESTUDIO DE EXAMEN DE GRADO PARA OPTAR AL TÍTULO DE LICENCIATURA EN INGENIERÍA DE SISTEMAS**

**SANTA CRUZ – BOLIVIA, 2021**

**ÍNDICE**

[CAPÍTULO I: ASPECTOS GENERALES 2](#_Toc82595102)

[TITULO 2](#_Toc82595103)

[1.1. Introducción 2](#_Toc82595104)

[1.2. Planteamiento del Problema 3](#_Toc82595105)

[1.3. Delimitaciones 3](#_Toc82595106)

[1.3.1. Delimitación Temporal 3](#_Toc82595107)

[1.3.2. Delimitación Espacial 3](#_Toc82595108)

[1.4. Objetivo General 3](#_Toc82595109)

[1.5. Objetivos Específicos 3](#_Toc82595110)

[CAPÍTULO II: MARCO TEORICO 3](#_Toc82595111)

[2.1. Metodologías 3](#_Toc82595112)

[2.1.1. Metodología XP 3](#_Toc82595113)

[2.1.2. Fases de XP 3](#_Toc82595114)

[2.1.3. Método sistémico de control y mejora organizacional 5](#_Toc82595115)

[2.2. Los valores de XP 5](#_Toc82595116)

[2.2.1. Comunicación 5](#_Toc82595117)

[2.2.2. Simplicidad 5](#_Toc82595118)

[2.2.3. Retroalimentación 6](#_Toc82595119)

[2.2.4. Coraje 6](#_Toc82595120)

[2.2.5. Aplicabilidad 6](#_Toc82595121)

[2.3. Marco de trabajo SCRUM 6](#_Toc82595122)

[2.4. Historias de usuario 7](#_Toc82595123)

[2.5. El Product Owner 7](#_Toc82595124)

[2.6. El Scrum Master 8](#_Toc82595125)

[2.7. Product backlog 8](#_Toc82595126)

[2.8. Planning Poker 9](#_Toc82595127)

[2.9. Arquitectura del sistema y/o patrones de desarrollo 10](#_Toc82595128)

[2.9.1. Modelo Vista Controlador 10](#_Toc82595129)

[2.9.2. Lenguajes 11](#_Toc82595130)

[2.9.3. Framework 12](#_Toc82595131)

[2.9.4. Plataforma 12](#_Toc82595132)

[2.9.5. Base de datos 13](#_Toc82595133)

[2.9.5. IDE 13](#_Toc82595134)

[CAPÍTULO III: INGENIERIA DEL PROYECTO 14](#_Toc82595135)

[3.1. Fase de Exploración 14](#_Toc82595136)

[3.1.1. Requisitos no funcionales 14](#_Toc82595137)

[3.1.2. Diagrama de clases 13](#_Toc82595138)

[3.1.3. Product Backlog – Historias de usuario 14](#_Toc82595139)

[3.2. Fase de planificación de la entrega (Release Plan) 15](#_Toc82595140)

[3.2.1. Priorización de las historias de usuario 15](#_Toc82595141)

[3.2.2. Estimación de esfuerzo por puntos de Historia 16](#_Toc82595142)

[3.3. Fase de iteraciones (Desarrollo) 17](#_Toc82595143)

[3.4. Fase de Producción 24](#_Toc82595144)

[3.4.1. Diagrama de clases de diseño 25](#_Toc82595145)

[3.4.2. Modelo de base de datos 26](#_Toc82595146)

[3.4.3. Diseño de Reportes 27](#_Toc82595147)

[3.4.4. Triggres 27](#_Toc82595148)

[3.4.5. Consultas más complejas 28](#_Toc82595149)

[3.4.6. Procedimientos almacenados 29](#_Toc82595150)

[3.5. Fase de mantenimiento 29](#_Toc82595151)

[3.5.1. Plan de backup de base de datos 29](#_Toc82595152)

[3.6. Fase de muerte del proyecto 30](#_Toc82595153)

[3.6.1. Rendimiento del sistema 30](#_Toc82595154)

[3.6.2. Confiabilidad del sistema 30](#_Toc82595155)

[Conclusiones 31](#_Toc82595156)

[Recomendaciones 31](#_Toc82595157)

[Bibliografía 32](#_Toc82595158)

[Anexos 33](#_Toc82595159)

[ENUNCIADO DEL CASO DE USO 33](#_Toc82595160)

**Índice de Figuras**

[**Figura 1:** Planning Poker 9](#_Toc82595161)

[**Figura 2:** Modelo Vista Controlador 11](#_Toc82595162)

[**Figura 3:** PostgreSQL 13](file:///C:\Users\dan\Desktop\DefExternaP.docx#_Toc82595163)

[**Figura 4:** Diagrama de clases (Modelo de Dominio) 14](#_Toc82595164)

[**Figura 5**: Burn Down Chart (Iteración II) 17](#_Toc82595165)

[**Figura 6**: Incremento (Iteración II) 18](#_Toc82595166)

[**Figura 7:** Burn Down Chart (Iteración IV) 19](#_Toc82595167)

[**Figura 8:** Incremento (Iteración IV) 19](#_Toc82595168)

[**Figura 9:** Burn Down Chart (Iteracion VI) 20](#_Toc82595169)

[**Figura 10:** Incremento (Iteración VI) 21](#_Toc82595170)

[**Figura 11:** Burn Down Chart (Iteración VII) 22](#_Toc82595171)

[**Figura 12:** Incremento (Iteración VII) 22](#_Toc82595172)

[**Figura 13**: Burn Down Chart (Iteración VIII) 23](#_Toc82595173)

[**Figura 14:** Incremento (Iteración VIII) 24](#_Toc82595174)

[**Figura 15:** Diagrama de Clases de Diseño 25](#_Toc82595175)

[**Figura 16**: Modelo de base de datos 26](#_Toc82595176)

[**Figura 17:** Diseño de Reportes 27](#_Toc82595177)

[**Figura 18:** Triggres 27](#_Toc82595178)

[**Figura 19:** Consulta Viajes 28](#_Toc82595179)

[**Figura 20:** Consulta Ventas 28](#_Toc82595180)

[**Figura 21:** Procedimientos almacenados 29](#_Toc82595181)

[**Figura 22:** Backup completo 29](#_Toc82595182)

[**Figura 23:** Backup Incremental 30](#_Toc82595183)

[**Figura 24:** Login de Usuario 35](#_Toc82595184)

[**Figura 25:** Pantalla Ciudades 35](#_Toc82595185)

[**Figura 26**: Eliminar Ciudad 36](#_Toc82595186)

[**Figura 27:** Pantalla Buses 36](#_Toc82595187)

[**Figura 28:** Pantalla Viajes 37](#_Toc82595188)

[**Figura 29:** Pantalla Ventas 37](#_Toc82595189)

[**Figura 30:** Pantalla Reportes 38](#_Toc82595190)

**Índice de Tablas**

[**Tabla 1**: Product Backlog (Historias de usuario) 14](#_Toc82595191)

[**Tabla 2:** Priorizacion de las historias de usuario 15](#_Toc82595192)

[**Tabla 3**: Estimación de esfuerzos por puntos de Historia 16](#_Toc82595193)

[**Tabla 4:** Iteración I 17](#_Toc82595194)

[**Tabla 5:** Iteración IV 18](#_Toc82595195)

[**Tabla 6:** Iteración VI 20](#_Toc82595196)

[**Tabla 7:** Incremento VII 21](#_Toc82595197)

[**Tabla 8:** Iteración VIII 23](#_Toc82595198)

# CAPÍTULO I: ASPECTOS GENERALES

## TÍTULO

SISTEMA ADMINISTRATIVO BASADO EN EL MÉTODO SISTÉMICO DE CONTROL Y MEJORA ORGANIZACIONAL PARA MEJORAR LA GESTIÓN DE VENTAS DE LA EMPRESA DE TRANSPORTES “FLOTA MARISCAL SANTA CRUZ”.

## Introducción

La automatización de procesos diarios mediante sistemas o servicios informáticos crece y se actualiza exponencialmente con el paso del tiempo, logrando así mejoras significativas en la atención de clientes, automatización de proceso y en las actividades diarias que cumple cualquier negocio y/o empresa ya sea grande, mediana o pequeña como por ejemplo en el caso de las empresas de transporte.

Según datos registrados en Bolivia por Fundempresa; del 2019 al 2021 la inscripción de empresas según actividad económica que se dedican a Servicios de Transporte y Almacenamiento es de un 14.39% siendo en 2019 un total de 14.379 empresas registradas y un total de 16.448 en 2021 hasta el mes de septiembre. (FUNDAEMPRESA, 2021)

Solo en el año 2021 según datos registrados por el INE (Instituto Nacional de Estadística) el flujo del transporte carretero de pasajeros por ruta en la red troncal que comprenden las ciudades de La Paz, Cochabamba y Santa Cruz es de 16.392.617 pasajeros. (INE, 2021)

En este sentido usando diversas técnicas de programación la tecnología informática que automatiza las tareas de las empresas de transporte y ha sido fundamental para mejorar el tiempo y atención al cliente dando paso a nuevas propuestas para mejorar la calidad de atención.

Ante esta demanda tecnológica, mediante el método sistémico de control y mejora organizacional, las herramientas y paradigmas del desarrollo Ágil “XP” y acompañado del marco de trabajo “SCRUM”, se desarrolló un sistema de administrativo para mejorar la gestión de ventas de la empresa de transportes “flota mariscal santa cruz”.

## Planteamiento del Problema

La empresa de transportes “Flota Mariscal Santa Cruz”, que en este momento es una empresa pequeña ha recibido una inyección de capital de un inversor que quiere escalar la empresa a nivel nacional ofreciendo viajes entre las principales ciudades de Bolivia en la red troncal y demás capitales. Esto obliga a la empresa a tener toda la información centralizada y disponible entre las principales Ciudades de Bolivia, abriendo la necesidad de un sistema informático que permita a la empresa cubrir la venta de pasajes a nivel nacional.

En este proceso se identificaron las siguientes dificultades a la hora de realizar la venta de un pasaje:

* Pérdida de información.
* Retraso a la hora de atender a los clientes
* El no tener una información centralizada tiene como consecuencia no poder realizar compras de pasajes desde ciudades intermedias.

## Delimitaciones

### Delimitación Temporal

El desarrollo del sistema administrativo para la empresa de transportes “Flota Mariscal Santa Cruz” será desarrollado e implementado en un periodo de 7 meses.

### Delimitación Espacial

El presente proyecto se desarrollará para la empresa de transportes “Flota Mariscal Santa Cruz” ubicada en la ciudad de Santa Cruz – Bolivia.

## Objetivo General

Implementar un sistema administrativo basado en el método sistémico de control y mejora organizacional para mejorar la gestión de ventas de la empresa de transportes “flota mariscal santa cruz”.

## Objetivos Específicos

* Recolectar información mediante entrevistas al personal para recolectar y entender los procesos de la empresa.
* Identificar una lista de los requerimientos para planificar y definir el alcance del software.
* Realizar estimaciones mediante puntos de Historia (en Horas) para calcular la duración del proyecto y el desarrollo.
* Diseñar y analizar el sistema administrativo en base a las tecnologías y metodologías escogidas para mejorar los procesos o actividades de la empresa.

# CAPÍTULO II: MARCO TEORICO

## 2.1. Metodologías

### 2.1.1. Metodología XP

La **metodología XP** es un conjunto de técnicas que dan agilidad y flexibilidad en la gestión de proyectos. También es conocida como **Programación Extrema** (Extreme Programming) y se centra crear un producto según los requisitos exactos del cliente. De ahí, que le involucre al máximo durante el método de gestión del desarrollo del producto. (www.sinnaps.com, n.d.)

### 2.1.2. Fases de XP

* **Fase I: Exploración.**

En esta fase, los clientes plantean a grandes rasgos las historias de usuario que son de interés para la primera entrega del producto. Al mismo tiempo el equipo de desarrollo se familiariza con las herramientas, tecnologías y prácticas que se utilizarán en el proyecto. Se prueba la tecnología y se exploran las posibilidades de la arquitectura del sistema construyendo un prototipo. (Letelier & Penadés, n.d.)

* **Fase II: Planificación de la Entrega.**

En esta fase el cliente establece la prioridad de cada historia de usuario, y correspondientemente, los programadores realizan una estimación del esfuerzo necesario de cada una de ellas. Se toman acuerdos sobre el contenido de la primera entrega y se determina un cronograma en conjunto con el cliente. Una entrega debería obtenerse en no más de tres meses. Esta fase dura unos pocos días.

La planificación se puede realizar basándose en el tiempo o el alcance. La velocidad del proyecto es utilizada para establecer cuántas historias se pueden implementar antes de una fecha determinada o cuánto tiempo tomará implementar un conjunto de historias. (Letelier & Penadés, n.d.)

* **Fase III: Desarrollo**

Esta fase incluye varias iteraciones sobre el sistema antes de ser entregado. El Plan de Entrega está compuesto por iteraciones de no más de tres semanas. En la primera iteración se puede intentar establecer una arquitectura del sistema que pueda ser utilizada durante el resto del proyecto. Esto se logra escogiendo las historias que fuercen la creación de esta arquitectura, sin embargo, esto no siempre es posible ya que es el cliente quien decide qué historias se implementarán en cada iteración (para maximizar el valor de negocio). Al final de la última iteración el sistema estará listo para entrar en producción. (Letelier & Penadés, n.d.)

* **Fase IV: Producción**

La fase de producción requiere de pruebas adicionales y revisiones de rendimiento antes de que el sistema sea trasladado al entorno del cliente. Al mismo tiempo, se deben tomar decisiones sobre la inclusión de nuevas características a la versión actual, debido a cambios durante esta fase.

Es posible que se rebaje el tiempo que toma cada iteración, de tres a una semana. Las ideas que han sido propuestas y las sugerencias son documentadas para su posterior implementación (por ejemplo, durante la fase de mantenimiento). (Letelier & Penadés, n.d.)

* **Fase V: Mantenimiento**

Mientras la primera versión se encuentra en producción, el proyecto XP debe mantener el sistema en funcionamiento al mismo tiempo que desarrolla nuevas iteraciones. Para realizar esto se requiere de tareas de soporte para el cliente. De esta forma, la velocidad de desarrollo puede bajar después de la puesta del sistema en producción. La fase de mantenimiento puede requerir nuevo personal dentro del equipo y cambios en su estructura. (Letelier & Penadés, n.d.)

* **Fase VI: Muerte del Proyecto**

Es cuando el cliente no tiene más historias para ser incluidas en el sistema. Esto requiere que se satisfagan las necesidades del cliente en otros aspectos como rendimiento y confiabilidad del sistema. Se genera la documentación final del sistema y no se realizan más cambios en la arquitectura. La muerte del proyecto también ocurre cuando el sistema no genera los beneficios esperados por el cliente o cuando no hay presupuesto para mantenerlo. (Letelier & Penadés, n.d.)

### 2.1.3. Método sistémico de control y mejora organizacional

El "Método sistémico de control y mejora organizacional”, es una herramienta de gestión que busca en primer lugar alcanzar el control de los procesos más relevantes de la organización, y luego basado en ello, el segundo objetivo es iniciar las tareas de mejora de los procesos a través de procedimientos sistémicos, sustentados en las denominadas "Bases para el control y la mejora organizacional”. (amp.ww.es.freejournal.org, n.d.)

## 2.2. Los valores de XP

XP se basa en cuatro valores, que deben estar presentes en el equipo de desarrollo para que el proyecto tenga éxito.

### 2.2.1. Comunicación

Muchos de los problemas que existen en proyectos de software se deben a problemas de comunicación entre las personas. La comunicación permanente es fundamental en XP. Dado que la documentación es escasa, el diálogo frontal, cara a cara, entre desarrolladores, gerentes y el cliente es el medio básico de comunicación. Una buena comunicación tiene que estar presente durante todo el proyecto. (ERNESTO, 2014)

### 2.2.2. Simplicidad

XP, como metodología ágil, apuesta a la sencillez, en su máxima expresión. Sencillez en el diseño, en el código, en los procesos, etc. La sencillez es esencial para que todos puedan entender el código, y se trata de mejorar mediante recodificaciones continuas. (ERNESTO, 2014)

### 2.2.3. Retroalimentación

La retroalimentación debe funcionar en forma permanente. El cliente debe brindar retroalimentación de las funciones desarrolladas, de manera de poder tomar sus comentarios para la próxima iteración, y para comprender, cada vez más, sus necesidades. Los resultados de las pruebas unitarias son también una retroalimentación permanente que tienen los desarrolladores acerca de la calidad de su trabajo. (ERNESTO, 2014)

### 2.2.4. Coraje

Cuando se encuentran problemas serios en el diseño, o en cualquier otro aspecto, se debe tener el coraje suficiente como para encarar su solución, sin importar que tan difícil sea. Si es necesario cambiar completamente parte del código, hay que hacerlo, sin importar cuanto tiempo se ha invertido previamente en el mismo. (ERNESTO, 2014)

### 2.2.5. Aplicabilidad

Por lo general, cada metodología tiene sus escenarios de aplicabilidad. Ninguna de las metodologías de desarrollo de software es buenas para todos los proyectos. Para proyectos que requieran varias decenas de desarrolladores, y en los que las especificaciones estén claramente determinadas desde el comienzo, los métodos en cascada o espiral pueden ser los más adecuados. Por el contrario, para proyectos medianos, y en los que las especificaciones no se puedan obtener hasta luego de comenzado el proyecto, XP puede ser la metodología recomendada. (ERNESTO, 2014)

## 2.3. Marco de trabajo SCRUM

Es el **marco de trabajo ágil** de desarrollo de software más utilizado en el mundo, **no** **una metodología.** Scrum es un **proceso de gestión** que reduce la complejidad en el desarrollo de productos para satisfacer las necesidades de los clientes. En el cual, se aplican de manera regular un **“conjunto de buenas prácticas para trabajar colaborativamente, en equipo, y obtener el mejor resultado posible de un proyecto”.**Estas prácticas se apoyan unas a otras y su selección tiene origen en un estudio de la manera de trabajar de equipos altamente productivos. (Djandrw, 2019)

## 2.4. Historias de usuario

Una historia de usuario es una representación de un [requisito](https://es.wikipedia.org/wiki/Requerimiento_(sistemas)) escrito en una o dos frases utilizando el lenguaje común del usuario. Las historias de usuario son utilizadas en las [metodologías de desarrollo ágiles](https://es.wikipedia.org/wiki/Metodolog%C3%ADa_%C3%A1gil) para la especificación de requisitos. Cada historia de usuario debe ser limitada, esta debería poderse escribir sobre una nota adhesiva pequeña.

Las historias de usuario son una forma rápida de administrar los requisitos de los usuarios sin tener que elaborar gran cantidad de documentos formales y sin requerir de mucho tiempo para administrarlos. Las historias de usuario permiten responder rápidamente a los requisitos cambiantes. (es.wikipedia.org, n.d.)

## 2.5. El Product Owner

Es uno de los roles dentro del equipo Scrum, encargado de maximizar el valor de trabajo entregado y del retorno de inversión. Es el representante del cliente dentro del equipo, su voz, representa a las partes interesadas internas y es responsable de entregar el valor más alto posible al negocio.

La responsabilidad principal del Product Owner es representar las necesidades del cliente manteniendo las capacidades clave y los resultados deseados.

Entre otras, el Product Owner es responsable de:

* El retorno de inversión del proyecto.
* Gestionar el Product Backlog en su totalidad, ordenando y priorizando las tareas para alcanzar los objetivos de la mejor forma.
* Optimizar el valor del trabajo del Equipo.
* Asegurarse de que el Product Backlog sea visible, transparente y claro para todos. (beagilemyfriend.com, n.d.)

## 2.6. El Scrum Master

**El Scrum Master es un rol muy sustancial dentro de un equipo**[Scrum](https://www.beagilemyfriend.com/que-es-scrum/), su participación en todos los eventos de un Sprint y haciendo de líder al servicio de los demás, lo convierten en un ingrediente esencial para la armonía del equipo.

**Se asegura de que el framework Scrum se desarrolle correctamente**. Es un líder al servicio de un equipo.

Elimina impedimentos, ayuda al equipo de desarrollo a ser más productivo y guía y enseña al Product Owner en la gestión del product backlog.

Además, debe participar en las reuniones y asegurarse de que el equipo Scrum cumpla el tiempo y los objetivos establecidos.

Entre otras, un Scrum Master debe:

* Planificar la implantación de Scrum junto con la organización.
* Ayudar a la organización a entender que interacciones con el equipo aportan valor y cuáles no.
* Asistir al Product Owner a saber cómo organizar el [product backlog](https://www.beagilemyfriend.com/product-backlog/) y maximizar el valor.
* Asegurarse de que haya una [definición de done.](https://www.beagilemyfriend.com/que-es-el-definition-of-done-dod/)
* Comprender y practicar la agilidad.
* Ayudar al equipo a crear productos de valor.
* Eliminar cualquier impedimento del equipo.
* Facilitar los eventos Scrum.
* Trabajar con otros Scrum Masters para aumentar la efectividad de Scrum en la organización. Junto con el equipo Scrum, actualizar el burndown chart. (beagilemyfriend.com, n.d.)

## 2.7. Product backlog

Es una lista priorizada y ordenada de requisitos del cliente (llamados Product Backlog items) de un proyecto. **Es gestionado por el**[product owner](https://www.beagilemyfriend.com/product-owner/), incluyendo su contenido, disponibilidad y peticiones, además es él (el product owner) quien ordena el Product Backlog en base al valor, riesgos, dependencias y necesidades de negocio.

Todos los requerimientos, funcionales y no funcionales, tareas y bugs deben ir en el Product Backlog. **Su contenido refleja todo el trabajo que el equipo de desarrollo tiene que hacer**. En otras palabras, el equipo de desarrollo no hace absolutamente nada que no se encuentre en este listado. (beagilemyfriend.com, n.d.)

## 2.8. Planning Poker

**Planning Poker es una técnica de estimación puesta en marcha por primera vez por James Grenning en un equipo Ágil utilizando XP en 2002.**

Desde luego no tiene relación alguna con el Poker. Se utiliza una baraja de cartas con una distribución de números muy parecida a la secuencia de Fibonacci (0, 1/2, 1, 2, 3, 5, 8, 13, etc).

Existen valores muy grandes como 20 o 40, usados para estimar historias de usuarios muy grandes, cuando en la sesión aparezcan estos valores, la historia de usuario debería ser partida en sub-historias más pequeñas.

Se utiliza en Scrum, pero puede ser aplicado en otros contextos. Favorece la participación y comunicación, esto último indispensable en [Scrum](https://www.beagilemyfriend.com/que-es-scrum/). Además de que los miembros del equipo aprendan unos de otros, y del producto que están desarrollando. (beagilemyfriend.com, n.d.)



**Figura 1:** Planning Poker

## 2.9. Arquitectura del sistema y/o patrones de desarrollo

### 2.9.1. Modelo Vista Controlador

Modelo Vista Controlador (MVC) es un estilo de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos.

* El **Modelo** que contiene una representación de los datos que maneja el sistema, su lógica de negocio, y sus mecanismos de persistencia.
* La **Vista**, o interfaz de usuario, que compone la información que se envía al cliente y los mecanismos interacción con éste.
* El **Controlador**, que actúa como intermediario entre el Modelo y la Vista, gestionando el flujo de información entre ellos y las transformaciones para adaptar los datos a las necesidades de cada uno.

El modelo es responsable de:

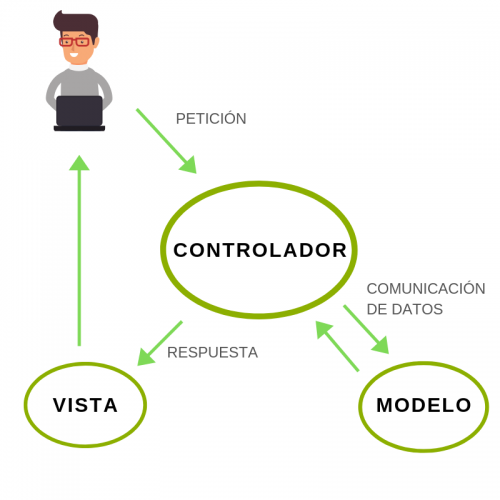
* Acceder a la capa de almacenamiento de datos. Lo ideal es que el modelo sea independiente del sistema de almacenamiento.
* Define las reglas de negocio (la funcionalidad del sistema). Un ejemplo de regla puede ser: "Si la mercancía pedida no está en el almacén, consultar el tiempo de entrega estándar del proveedor".
* Lleva un registro de las vistas y controladores del sistema.

El controlador es responsable de:

* Recibe los eventos de entrada (un clic, un cambio en un campo de texto, etc.).
* Contiene reglas de gestión de eventos, del tipo "SI Evento Z, entonces Acción W". Estas acciones pueden suponer peticiones al modelo o a las vistas. Una de estas peticiones a las vistas puede ser una llamada al método "Actualizar ()".

 Las vistas son responsables de:

* Recibir datos del modelo y mostrarlos al usuario.
* Tienen un registro de su controlador asociado (normalmente porque además lo instancia).
* Pueden dar el servicio de "Actualización ()", para que sea invocado por el controlador o por el modelo (cuando es un modelo activo que informa de los cambios en los datos producidos por otros agentes).



**Figura 2:** Modelo Vista Controlador

### 2.9.2. Lenguajes

* [**HTML**](https://developer.mozilla.org/es/docs/Glossary/HTML) es el lenguaje de marcado que usamos para estructurar y dar significado a nuestro contenido web, por ejemplo, definiendo párrafos, encabezados y tablas de datos, o insertando imágenes y videos en la página.
* [**CSS**](https://developer.mozilla.org/es/docs/Glossary/CSS) es un lenguaje de reglas de estilo que usamos para aplicar estilo a nuestro contenido HTML, por ejemplo, establecer colores de fondo y tipos de letra, y distribuir nuestro contenido en múltiples columnas.
* [**JavaScript**](https://developer.mozilla.org/es/docs/Glossary/JavaScript) es un lenguaje de secuencias de comandos que te permite crear contenido de actualización dinámica, controlar multimedia, animar imágenes y prácticamente todo lo demás. (Está bien, no todo, pero es sorprendente lo que puedes lograr con unas pocas líneas de código JavaScript).

### 2.9.3. Framework

Express JS es un Framework minimalista y te permite crear aplicaciones Web y APIs.  
Su API robusta permite a los usuarios configurar rutas para enviar y recibir solicitudes entre la vista Front y la Base de Datos actuando como un servidor HTTP, asimismo es compatible con muchos paquetes [NPM](https://www.npmjs.org/).

### 2.9.4. Plataforma

**Aplicación Web**

En la ingeniería de software se denomina aplicación web a aquellas herramientas que los usuarios pueden utilizar accediendo a un servidor web a través de internet o de una intranet mediante un navegador. En otras palabras, es un programa que se codifica en un lenguaje interpretable por los navegadores web en la que se confía la ejecución al navegador.

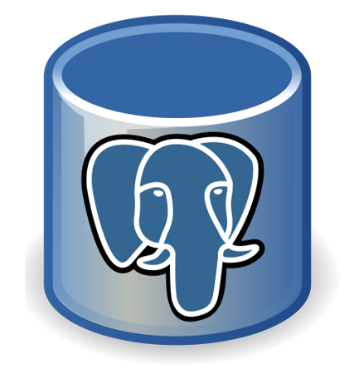
Las aplicaciones web son populares debido a lo práctico del navegador web como cliente ligero, a la independencia del sistema operativo, así como a la facilidad para actualizar y mantener aplicaciones web sin distribuir e instalar software a miles de usuarios potenciales. Existen aplicaciones como los correos web, wikis, blogs, tiendas en línea y la propia Wikipedia que son ejemplos bastante conocidos de aplicaciones web.

Es importante mencionar que una página web puede contener elementos que permiten una comunicación activa entre el usuario y la información. Esto permite que el usuario acceda a los datos de modo interactivo, gracias a que la página responderá a cada una de sus acciones, como por ejemplo rellenar y enviar formularios, participar en juegos diversos y acceder a gestores de base de datos de todo tipo.

### 2.9.5. Base de datos

PostgreSQL es un potente sistema de base de datos relacional de objetos de código abierto que utiliza y amplía el lenguaje SQL combinado con muchas características que almacenan y escalan de forma segura las cargas de trabajo de datos más complicadas.

PostgreSQL se ha ganado una sólida reputación por su arquitectura probada, confiabilidad, integridad de datos, conjunto de características robustas, extensibilidad y la dedicación de la comunidad

de código abierto detrás del software para brindar soluciones innovadoras y de alto rendimiento de manera consistente. ****

**Figura 3:** PostgreSQL

### 2.9.5. IDE

Visual Studio Code es un editor de código fuente ligero pero potente que se ejecuta en su escritorio y está disponible para Windows, macOS y Linux. Viene con soporte incorporado para JavaScript, TypeScript y Node.js y tiene un rico ecosistema de extensiones para otros lenguajes y tiempos de ejecución (como .NET y Unity).

# CAPÍTULO III: INGENIERIA DEL PROYECTO

## 3.1. Fase de Exploración

### 3.1.1. Requisitos no funcionales

**Eficiencia**

* Toda funcionalidad del sistema debe responder al usuario en menos de 5 segundos.
* El sistema debe ser capaz de operar adecuadamente con hasta 50 usuarios con sesiones concurrentes.
* Los datos modificados en la base de datos deben ser actualizados en la página en menos de 2 segundos para todos los usuarios conectados.

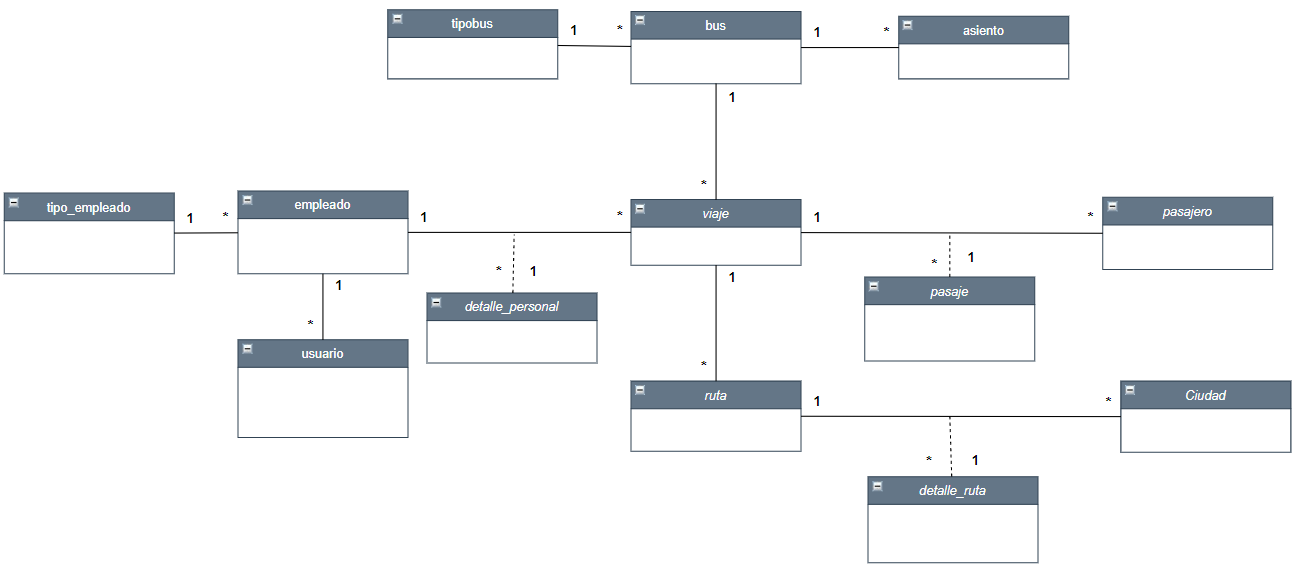
**Seguridad lógica y de datos**

* Los permisos de acceso al sistema podrán ser cambiados solamente por el administrador de acceso a datos.
* El nuevo sistema debe desarrollarse aplicando [patrones y recomendaciones de programación que incrementen la seguridad de datos](http://www.pmoinformatica.com/2014/01/consejos-seguridad-informatica.html).

**Usabilidad**

* El tiempo de aprendizaje del sistema por un usuario con conocimiento básico en informática deberá ser menor a 2 horas.
* La aplicación web debe poseer un [diseño “Responsive”](http://www.pmoinformatica.com/2015/04/consejos-diseno-web-movil.html) a fin de garantizar la adecuada visualización en múltiples computadores personales, dispositivos tableta y teléfonos inteligentes.
* El sistema debe poseer interfaces gráficas bien formadas.

### 3.1.2. Diagrama de clases



**Figura 4:** Diagrama de clases (Modelo de Dominio)

### 3.1.3. Product Backlog – Historias de usuario

**Tabla 1**: Product Backlog (Historias de usuario)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  | **Product Backlog** | | |
|  |  | **Hitorias de Usuario** | **Estado** | **Iteracion** |
| Autenticacion y Autorizacion | | Pantalla de Login | NO-TERMINADO | 1 |
| Servicio de Autenticacion | NO-TERMINADO |
| Servicio de Autorizacion | NO-TERMINADO |
| Redireccion de usuario | NO-TERMINADO |
| Gestion de Ciudades | | Registrar Ciudad | TERMINADO | 2 |
| Modificar Ciudad | TERMINADO |
| Eliminar Ciudad | TERMINADO |
| mostrar Ciudad | TERMINADO |
| Gestion de empleados | | Registrar Empleado | NO-TERMINADO | 3 |
| Modificar Empleado | NO-TERMINADO |
| Eliminar Empleado | NO-TERMINADO |
| Gesion de Buses | | Registrar Bus | TERMINADO | 4 |
| Modificar Bus | TERMINADO |
| Eliminar Bus | TERMINADO |
| mostrar Bus | TERMINADO |
| Gestion de Rutas | | Registrar ruta | NO-TERMINADO | 5 |
| Modificar ruta | NO-TERMINADO |
| Eliminar ruta | NO-TERMINADO |
| mostrar ruta | NO-TERMINADO |
| Gesion de asientos | | Interfaz de asientos | TERMINADO | 6 |
| mostrar Asiento disponibles | TERMINADO |
| cambiar estado Asiento | TERMINADO |
| Gestion de Viajes | | verificar pasajero | TERMINADO | 7 |
| registrar pasajero | TERMINADO |
| registrar viaje | TERMINADO |
| Modificar Viaje | TERMINADO |
| Eliminar Viaje | TERMINADO |
| Mostrar Viaje | TERMINADO |
| Gesion de Reportes | | reporte de viajes | TERMINADO | 8 |
| reporte de flotas | TERMINADO |
| reporte de pasajeros | NO-TERMINADO |
| reporte de ventas | TERMINADO |

*Elaboración Propia*

## 3.2. Fase de planificación de la entrega (Release Plan)

### 3.2.1. Priorización de las historias de usuario

**Tabla 2:** Priorizacion de las historias de usuario

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | **Priorizacion** | |
|  |  | **Hitoria de Usuario** | **Prioridad** |
| Autenticacion y Autorizacion | | Pantalla de Login | Media |
| Servicio de Autenticacion | Media |
| Servicio de Autorizacion | Media |
| Redireccion de usuario | Media |
| Gestion de Ciudades | | Registrar Ciudad | Alta |
| Modificar Ciudad | Baja |
| Eliminar Ciudad | Baja |
| mostrar Ciudad | Alta |
| Gestion de empleados | | Registrar Empleado | Alta |
| Modificar Empleado | Media |
| Eliminar Empleado | Media |
| Gestion de Buses | | Registrar Bus | Alta |
| Modificar Bus | Baja |
| Eliminar Bus | Baja |
| mostrar Bus | Alta |
| Gestion de Rutas | | Registrar ruta | Alta |
| Modificar ruta | Baja |
| Eliminar ruta | Baja |
| mostrar ruta | Alta |
| Gesion de asientos | | Interfaz de asientos | Alta |
| mostrar Asiento disponibles | Alta |
| cambiar estado Asiento | Alta |
| Gestion de Viajes | | verificar pasajero | Alta |
| registrar pasajero | Alta |
| registrar viaje | Alta |
| Modificar Viaje | Media |
| Eliminar Viaje | Baja |
| Mostrar Viaje | Media |
| Gesion de Reportes | | reporte de viajes | Media |
| reporte de flotas | Media |
| reporte de pasajeros | Media |
| reporte de ventas | Media |

*Elaboración Propia*

### 3.2.2. Estimación de esfuerzo por puntos de Historia

**Tabla 3**: Estimación de esfuerzos por puntos de Historia

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  | **Product Backlog** | |  |  |
|  |  | **Hitoria de Usuario** | **Esfuerzo** | **Total PH** |  |
| Autenticacion y Autorizacion | | Pantalla de Login | 10 | 34 | **264 PH** |
| Servicio de Autenticacion | 10 |
| Servicio de Autorizacion | 8 |
| Redireccion de usuario | 6 |
| Gestion de Ciudades | | Registrar Ciudad | 10 | 28 |
| Modificar Ciudad | 7 |
| Eliminar Ciudad | 5 |
| mostrar Ciudad | 6 |
| Gestion de empleados | | Registrar Empleado | 10 | 27 |
| Modificar Empleado | 10 |
| Eliminar Empleado | 5 |
| Gesion de Buses | | Registrar Bus | 10 | 31 |
| Modificar Bus | 10 |
| Eliminar Bus | 5 |
| mostrar Bus | 6 |
| Gestion de Rutas | | Registrar ruta | 10 | 28 |
| Modificar ruta | 5 |
| Eliminar ruta | 5 |
| mostrar ruta | 8 |
| Gesion de asientos | | Interfaz de asientos | 15 | 35 |
| mostrar Asiento disponibles | 15 |
| cambiar estado Asiento | 5 |
| Gestion de Viajes | | verificar pasajero | 3 | 42 |
| registrar pasajero | 8 |
| registrar viaje | 15 |
| Modificar Viaje | 8 |
| Eliminar Viaje | 5 |
| Mostrar Viaje | 3 |
| Gesion de Reportes | | reporte de viajes | 10 | 39 |
| reporte de flotas | 8 |
| reporte de pasajeros | 10 |
| reporte de ventas | 11 |

*Elaboración Propia*

## 3.3. Fase de iteraciones (Desarrollo)

1. **Iteración II**

**Tabla 4:** Iteración I

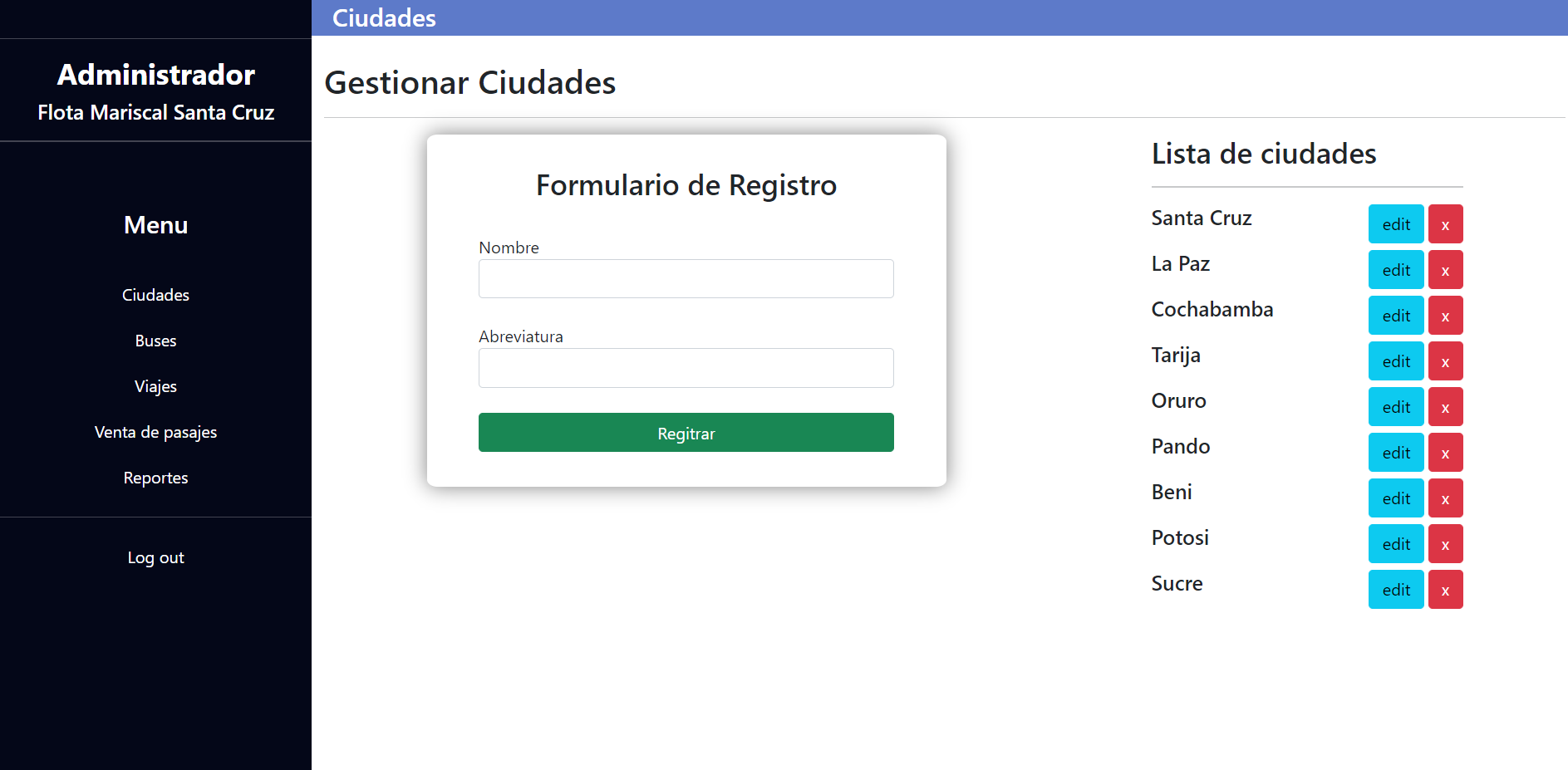
|  |  |  |  |
| --- | --- | --- | --- |
| **Fase de Iteraciones** | | | |
| **Sprint Backlog** | **Estado** | **Pruebas de aceptación** | **Comentarios** |
| Registrar Ciudad | TERMINADO | x | - |
| Modificar Ciudad | TERMINADO | x | - |
| Eliminar Ciudad | TERMINADO | x | - |
| mostrar Ciudad | TERMINADO | x | - |

*Elaboración Propia*



**Figura 5**: Burn Down Chart (Iteración II)





**Figura 6**: Incremento (Iteración II)

1. **Iteración IV**

**Tabla 5:** Iteración IV

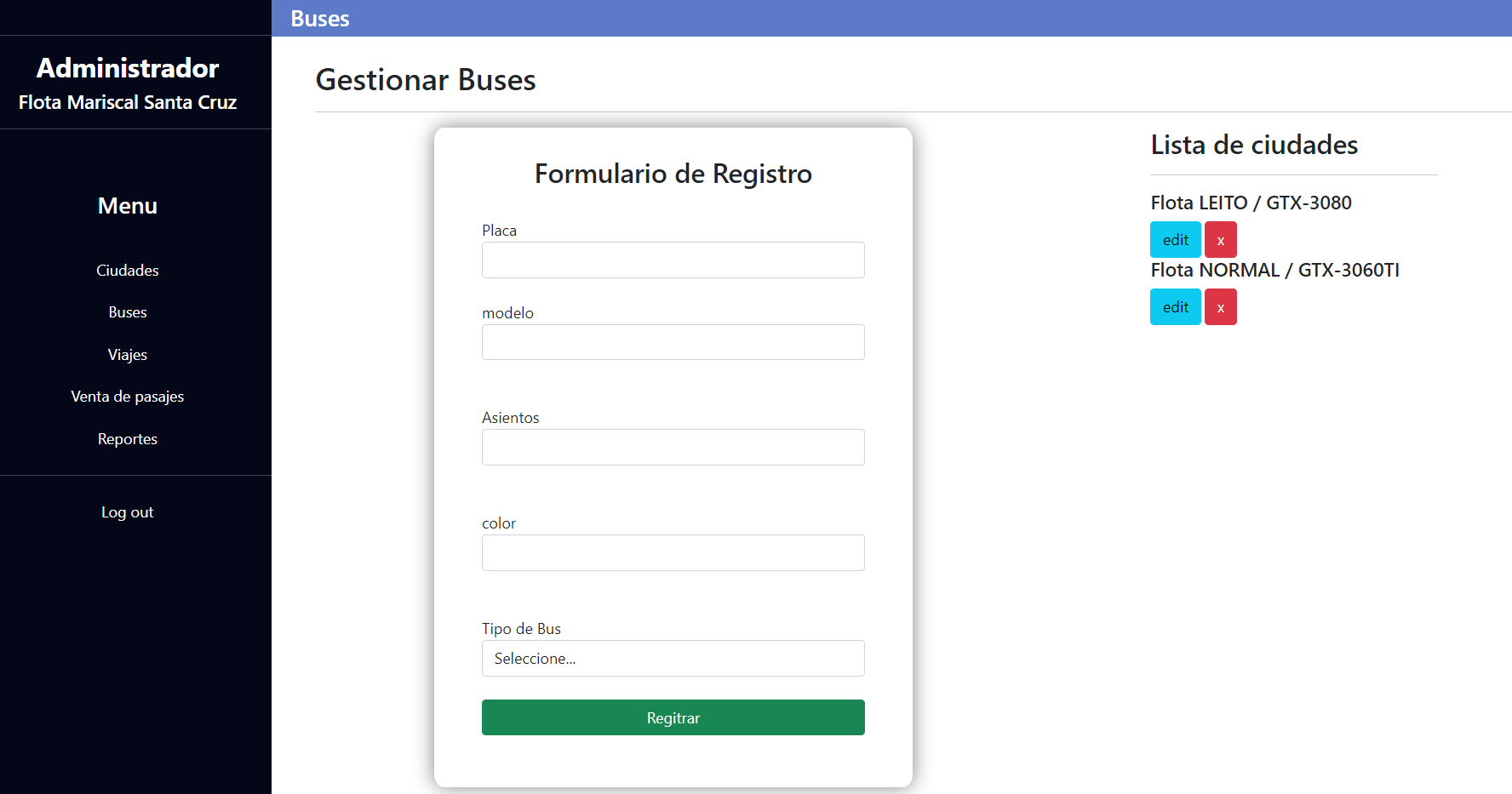
|  |  |  |  |
| --- | --- | --- | --- |
| **Fase de Iteraciones** | | | |
| **Sprint Backlog** | **Estado** | **Pruebas de aceptación** | **Comentarios** |
| Registrar Bus | TERMINADO | x | - |
| Modificar Bus | TERMINADO | x | - |
| Eliminar Bus | TERMINADO | x | - |
| mostrar Bus | TERMINADO | x | - |

*Elaboración Propia*



**Figura 7:** Burn Down Chart (Iteración IV)





**Figura 8:** Incremento (Iteración IV)

1. **Iteración VI**

**Tabla 6:** Iteración VI

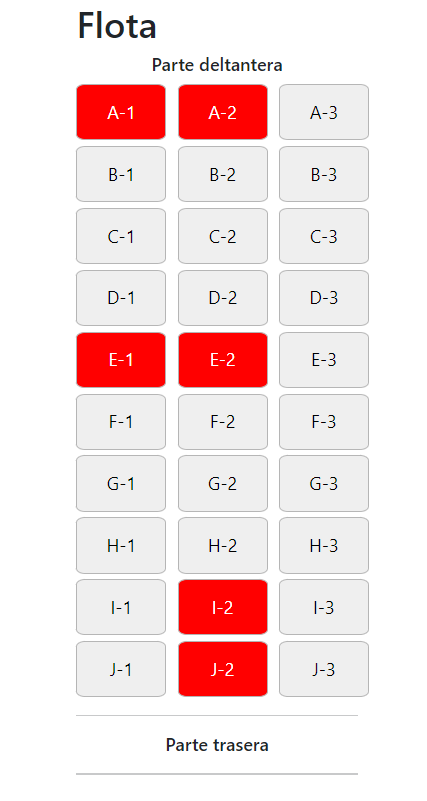
|  |  |  |  |
| --- | --- | --- | --- |
| **Fase de Iteraciones** | | | |
| **Sprint Backlog** | **Estado** | **Pruebas de aceptación** | **Comentarios** |
| Interfaz de asientos | TERMINADO | x | - |
| mostrar Asiento disponibles | TERMINADO | x | - |
| cambiar estado Asiento | TERMINADO | x | - |

*Elaboración Propia*



**Figura 9:** Burn Down Chart (Iteracion VI)





**Figura 10:** Incremento (Iteración VI)

1. **Iteración VII**

**Tabla 7:** Incremento VII

|  |  |  |  |
| --- | --- | --- | --- |
| **Fase de iteración** | | | |
| **Sprint Backlog** | **Estado** | **Pruebas de aceptación** | **Comentarios** |
| verificar pasajero | TERMINADO | x | - |
| registrar pasajero | TERMINADO | x | - |
| registrar viaje | TERMINADO | x | - |
| Modificar Viaje | TERMINADO | x | - |
| Eliminar Viaje | TERMINADO | x | - |
| Mostrar Viaje | TERMINADO | x | - |

*Elaboración Propia*



**Figura 11:** Burn Down Chart (Iteración VII)





**Figura 12:** Incremento (Iteración VII)

1. **Iteración VIII**

**Tabla 8:** Iteración VIII

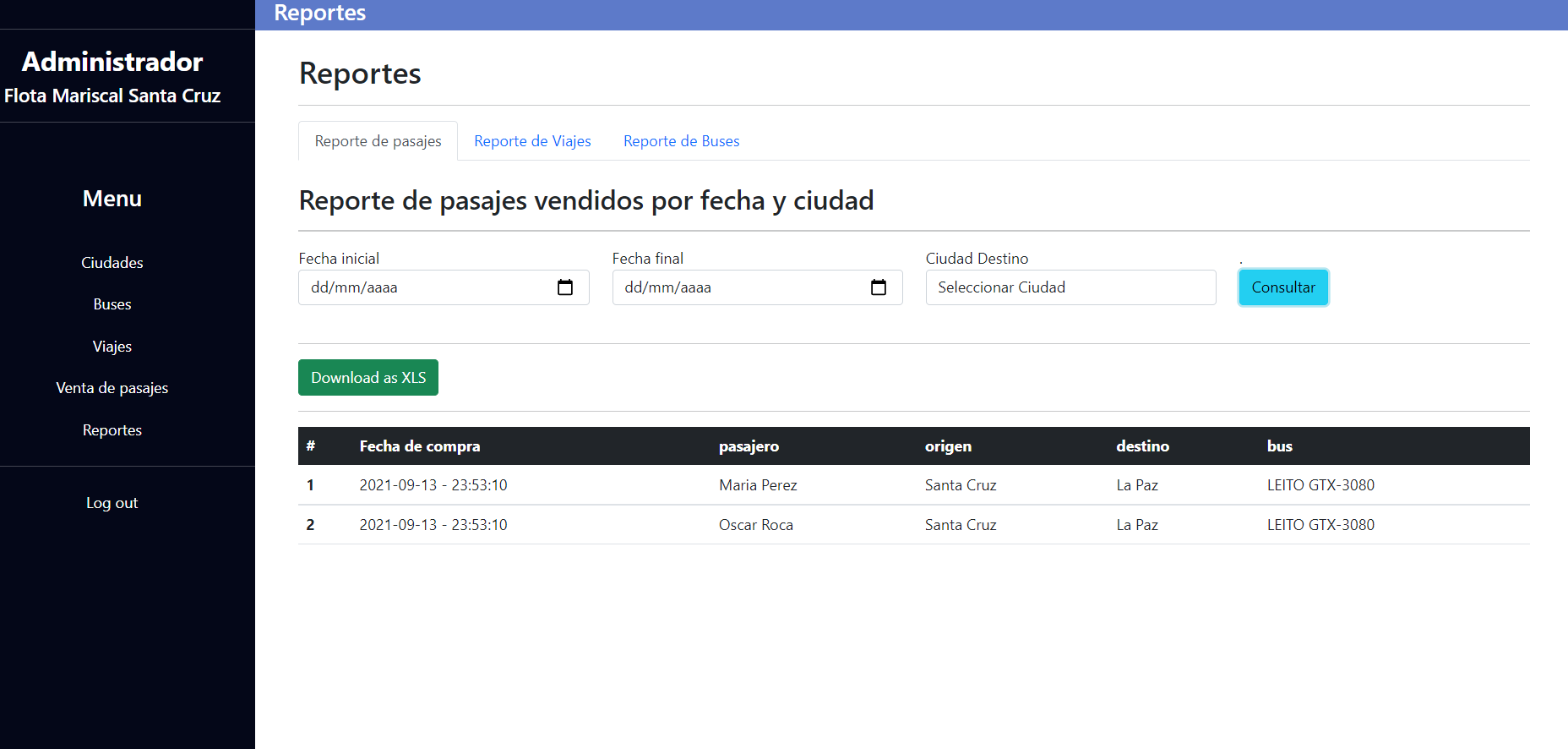
|  |  |  |  |
| --- | --- | --- | --- |
| **Fase de iteracion** | | | |
| **Sprint Backlog** | **Estado** | **Pruebas de aceptación** | **Comentarios** |
| reporte de viajes | TERMINADO | x | - |
| reporte de flotas | TERMINADO | x | - |
| reporte de pasajeros | TERMINADO | x | - |
| reporte de ventas | TERMINADO | x | - |

*Elaboración Propia*



**Figura 13**: Burn Down Chart (Iteración VIII)

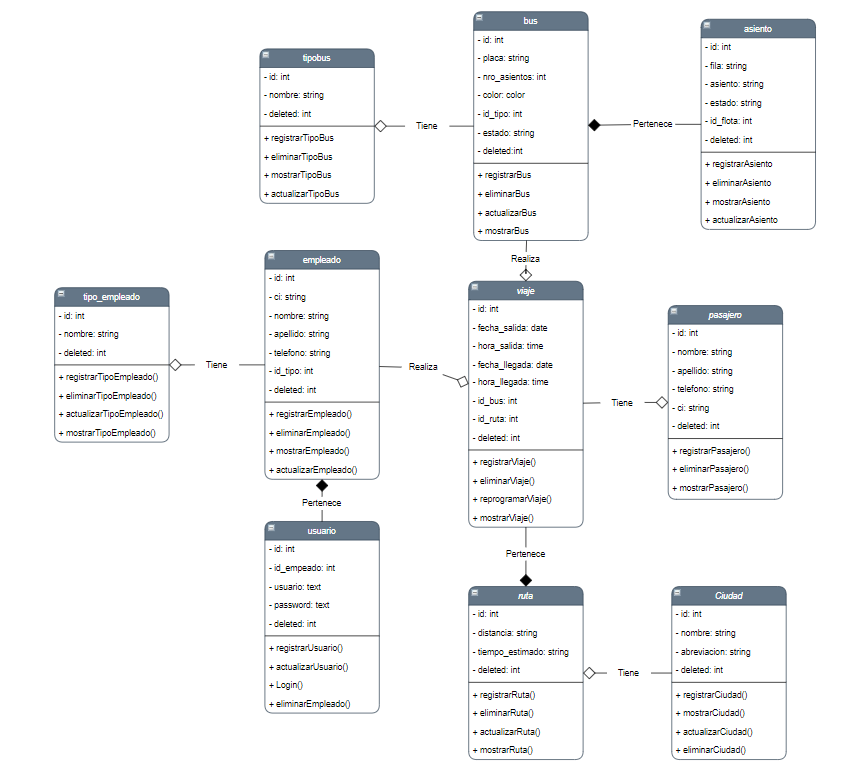




**Figura 14:** Incremento (Iteración VIII)

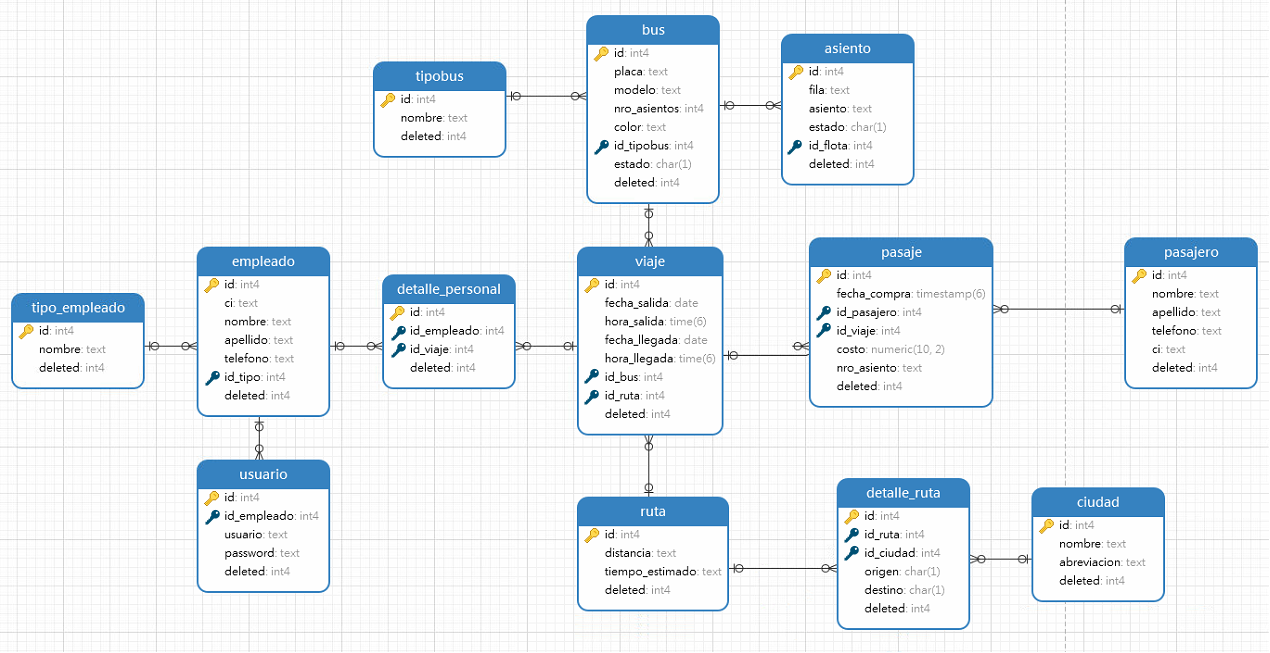
## 3.4. Fase de Producción

### 3.4.1. Diagrama de clases de diseño



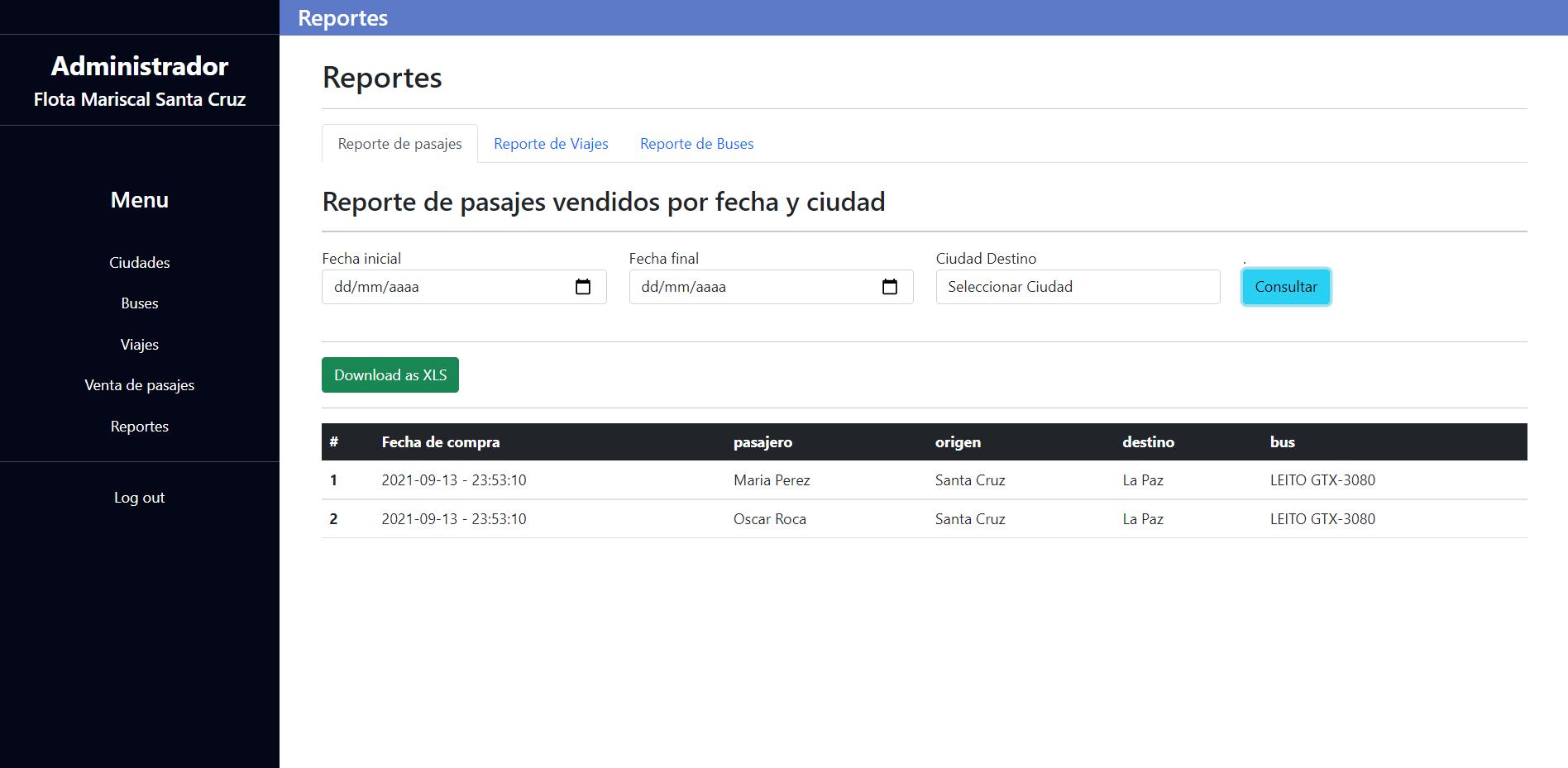
**Figura 15:** Diagrama de Clases de Diseño

### 3.4.2. Modelo de base de datos



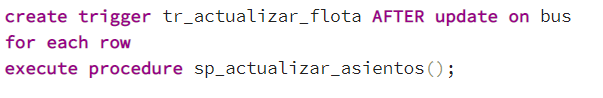
**Figura 16**: Modelo de base de datos

### 3.4.3. Diseño de Reportes



**Figura 17:** Diseño de Reportes

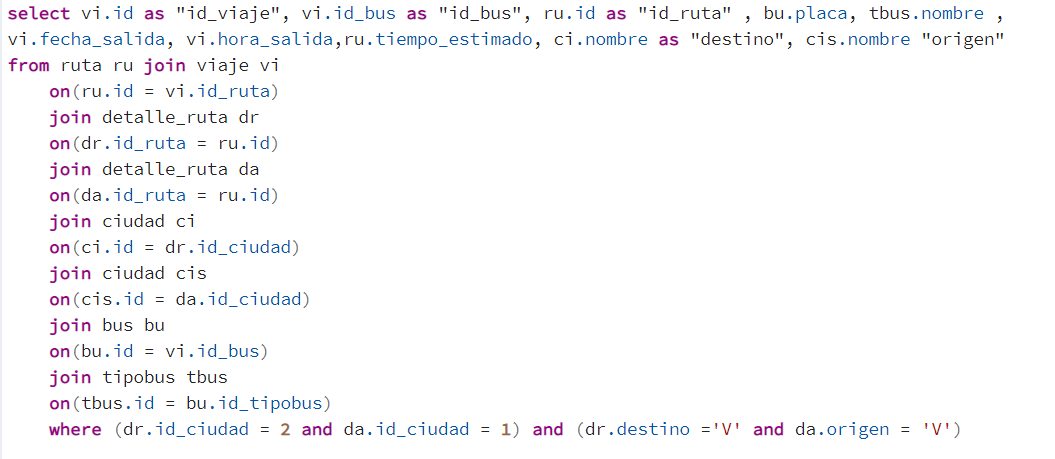
### 3.4.4. Triggres



**Figura 18:** Triggres

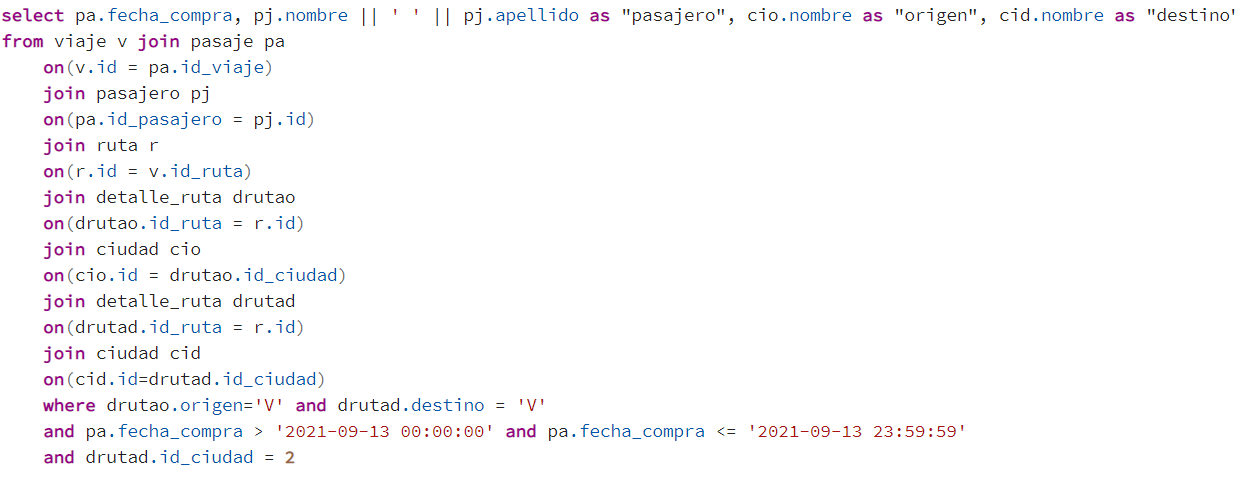
### 3.4.5. Consultas más complejas

* Consulta para traer los viajes con ciudad de origen y destino



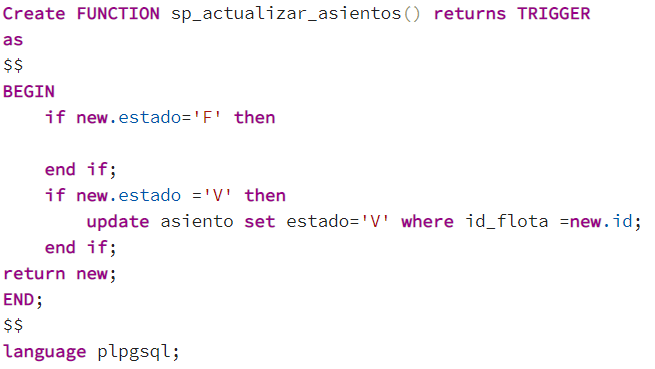
**Figura 19:** Consulta Viajes

* Consulta para traer el reporte las ventas de pasajes por fecha y ciudad



**Figura 20:** Consulta Ventas

### 3.4.6. Procedimientos almacenados

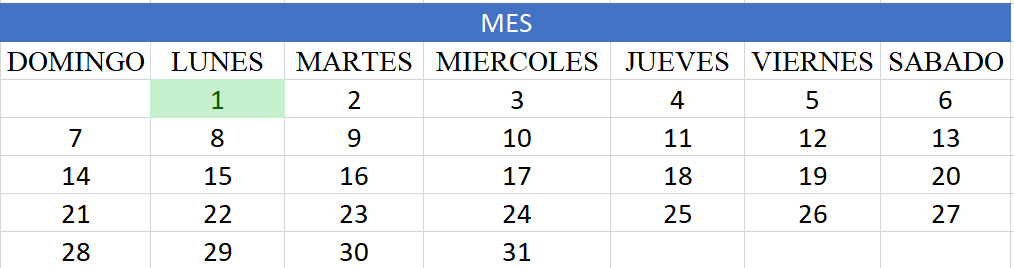


**Figura 21:** Procedimientos almacenados

## 3.5. Fase de mantenimiento

### 3.5.1. Plan de backup de base de datos

Se realizará un backup normal o completo cada primer día de mes para tener un respaldo de todos los movimientos de ese mes.



**Figura 22:** Backup completo

Se realizará un backup incremental 2 vez por semana para saber qué datos se modificaron o crearon desde el ultimo backup incremental para tener respaldo de los movimientos q se realizaron por semana.



**Figura 23:** Backup Incremental

## 3.6. Fase de muerte del proyecto

### 3.6.1. Rendimiento del sistema

Actualmente Google considera que el tiempo de respuesta del servidor, debe estar por debajo de los 200 milisegundos (0.2 segundos).

El sistema de gestión para la empresa de transporte “FlotaMariscal Santa Cruz” trae los datos del servidor aproximadamente en un tiempo de 200 milisegundos.

El sistema está desarrollado por React que es una librería JavaScript de Front End desarrollada por Facebook en 2011 y es una single page lo que aumenta el rendimiento del sistema.

Entonces podemos decir que el rendimiento de sistema de gestión para la empresa es bueno.

### 3.6.2. Confiabilidad del sistema

La disponibilidad del sistema, considerando la tecnología que utilizamos está basada completamente en la nube, dependerá del proveedor de servicios que en estos tiempos garantizan una alta disponibilidad.

Cada formulario del sistema posee validaciones lo cual nos evita que el sistema inserte datos incorrectos en la Base de datos.

Se realizaron pruebas con personas que no tenían conocimientos previos del funcionamiento del sistema, dando como resultado un tiempo estimado de 2 horas para el aprendizaje del sistema y por esto el usuario tendrá mayor confiabilidad en el sistema.

# Conclusiones

* Se identificó la lista de requerimientos para planificar y definir el alcance del proyecto.
* Se realizó la estimación mediante puntos de Historia para calcular duración del desarrollo.
* Se realizó revisiones de los procesos del software siguiendo lineamientos de revisión para garantizar y gestionar la calidad del software.
* Se desarrolló un sistema administrativo que logro automatizar las tareas administrativas de la empresa de transporte “Flota Mariscal Santa Cruz”, aplicando la metodología de desarrollo Ágil XP junto con su marco de trabajo SCRUM.
* Se realizó y analizo las historias de usuarios utilizando un Product backlog y una tabla de priorización mediante puntos de historia.
* Se logró centralizar el registro de datos, mediante la implementación de una base de datos bien analizada y estructurada, capaz de ser flexible y escalable con el tiempo.
* Se realizó el sistema escogiendo React como librería de desarrollo.
* Se realizó el diseño de interfaces amigables al usuario con parámetros bien definidos para un correcto registro de los datos.
* Se usó PostgreSQL para implementar la base de datos para gestionar la información.
* Se implementó las historias de usuario con mayor prioridad.
* Se realizaron pruebas del sistema para definir el rendimiento.

# Recomendaciones

* Realizar copias de seguridad de acuerdo al plan de backups indicado para evitar la pérdida de información y tener un mejor control del inventario.
* Se recomienda seguir con el diseño y la implantación de las demás historias de usuario, siguiendo la línea ya establecida en el documento.
* Se recomienda crear índices en las columnas principales de la base de datos para agilizar el proceso de búsqueda de la información de la empresa.
* También se recomienda tomar en cuenta los aspectos de diseño de la base de datos ya que la aplicación puede tener un crecimiento exponencial con el tiempo, y se puede llegar a requerir un aumento de funcionalidades.

# Bibliografía

amp.ww.es.freejournal.org. (s.f.). *https://amp.ww.es.freejournal.org/7793033/1/metodo-sistemico-de-control-y-mejora-organizacional.* Obtenido de https://amp.ww.es.freejournal.org/7793033/1/metodo-sistemico-de-control-y-mejora-organizacional.: https://amp.ww.es.freejournal.org/7793033/1/metodo-sistemico-de-control-y-mejora-organizacional.

beagilemyfriend.com. (s.f.). *https://beagilemyfriend.com/*. Obtenido de https://beagilemyfriend.com/: https://beagilemyfriend.com/product-owner/

Djandrw, A. (13 de Abril de 2019). *https://medium.com/@andrewdjandrw/qu%C3%A9-es-scrum-674c6b791af4*. Obtenido de https://medium.com/@andrewdjandrw/qu%C3%A9-es-scrum-674c6b791af4: https://medium.com/@andrewdjandrw/qu%C3%A9-es-scrum-674c6b791af4

ERNESTO, S. R. (2014). Santa Cruz.

es.wikipedia.org. (s.f.). *https://es.wikipedia.org/wiki/Historias\_de\_usuario*. Obtenido de https://es.wikipedia.org/wiki/Historias\_de\_usuario: https://es.wikipedia.org/wiki/Historias\_de\_usuario

FUNDAEMPRESA. (9 de Septiembre de 2021). *https://www.fundempresa.org.bo/*. Obtenido de https://www.fundempresa.org.bo/: https://www.fundempresa.org.bo/

Fundempresa. (Enero - 2020). *Estadisticas del Registro de Comercio de Bolivia.*

INE. (09 de Septiembre de 2021). *https://www.ine.gob.bo/*. Obtenido de https://www.ine.gob.bo/: https://www.ine.gob.bo/index.php/estadisticas-economicas/transportes/transporte-cuadros-estadisticos/

Letelier, P., & Penadés, M. C. (s.f.). *http://www.cyta.com.ar*. Obtenido de http://www.cyta.com.ar: http://www.cyta.com.ar/ta0502/v5n2a1.htm

www.sinnaps.com. (s.f.). *www.sinnaps.com*. Obtenido de www.sinnaps.com: https://www.sinnaps.com/blog-gestion-proyectos/metodologia-xp#que-es-la-metodologia-xp

# Anexos

# ENUNCIADO DEL CASO DE USO

**ÁREA: Bases de datos y sistemas de información**

**CASO: Sistema de gestión de venta de pasajes para la empresa de transportes “Flota Mariscal Santa Cruz”.**

Se necesita un sistema de gestión de venta de pasajes para la empresa de transporte “Flota Mariscal Santa Cruz”, que gestione los pasajeros, los pasajes, los viajes, las rutas y los choferes y asistentes asignados a al viaje.

La empresa de transportes “Flota Mariscal Santa Cruz”, que en este momento es una empresa pequeña ha recibido una inyección de capital de un inversor que quiere escalar la empresa a nivel nacional ofreciendo viajes entre las principales ciudades de Bolivia en la red troncal y demás capitales. Para poder lograr esto se ha visto la necesidad de un sistema informático que permita a la empresa automatizar los procesos informáticos de la empresa.

Al principio cuando se comience a usar el sistema lo primero que habrá que ingresar serán las ciudades con las que se podrán crear las rutas. También habrá que registrar en un principio los buses disponibles, así como el personal administrativo que viaja cada vez. El primer paso para poner en funcionamiento el sistema es crear un viaje, para el cual se elige una ruta previamente definida en el sistema. Al elegir el bus que hará el viaje se ponen en disponible todos los asientos del bus que fue elegido de manera gráfica y automáticamente y según las características de cada bus. A partir de este momento ya se pueden vender los pasajes.

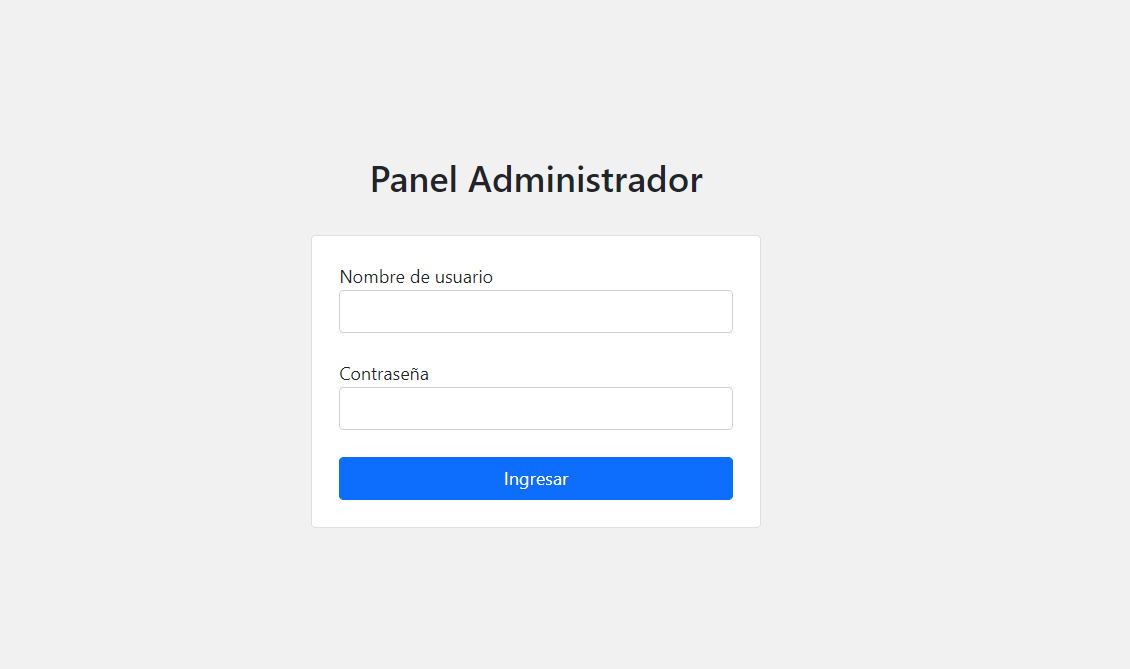
Para la venta del pasaje el usuario elegirá el o los lugares que quiere comprar. El operador hará click en estos lugares y les pondrá nombres, cambiando la disponibilidad de los asientos a “ocupado”.

Esta información le permitirá a la empresa vender los pasajes que queden disponibles en ciudades intermedias. Para eso es importante que el sistema sea web y pueda estar disponible desde cualquier lugar.

El sistema debe proveer una interface que permita a un operador llamar lista en un bus antes de que este inicie el viaje y constatar la presencia de todos los pasajeros en el bus.

El sistema deberá permitir el registro de todos los empleados que acompañaran el viaje. Estos son los choferes y ayudantes.

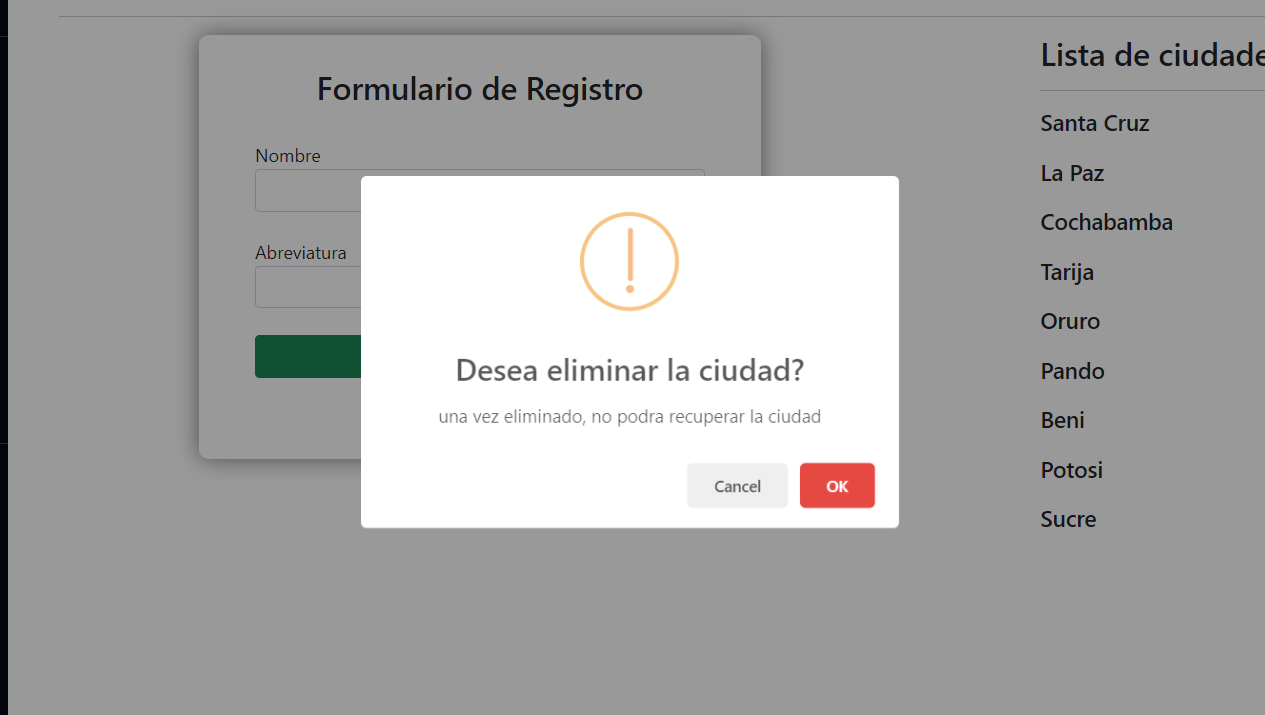
Al ser consultado, el encargado del desarrollo del sistema indica que lo mas urgente del sistema es la venta de pasajes ya que con eso la empresa puede funcionar. Para que se puedan vender pasajes deben estas registradas las ciudades, las rutas, los buses y los asistentes.



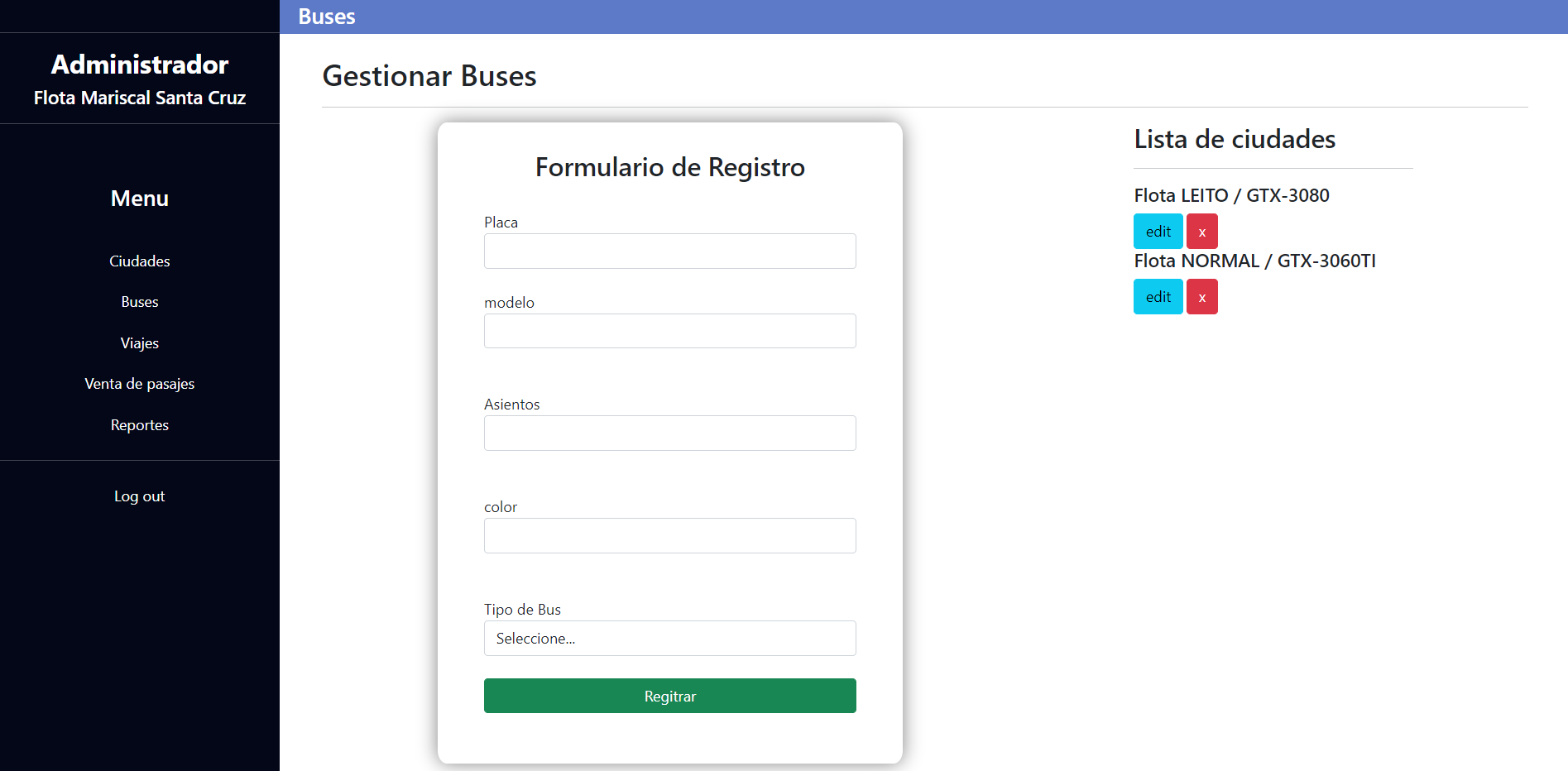
**Figura 24:** Login de Usuario



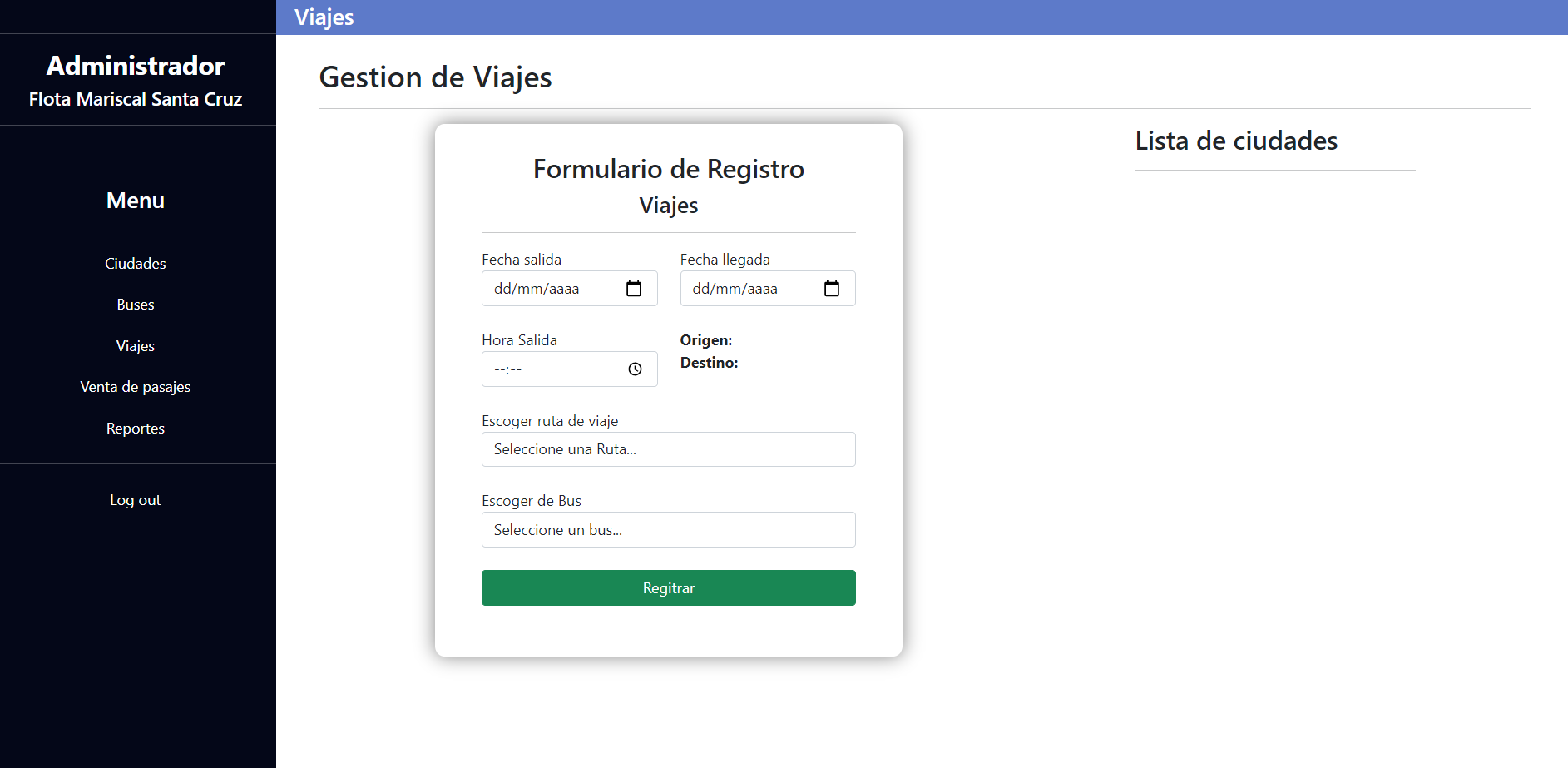
**Figura 25:** Pantalla Ciudades



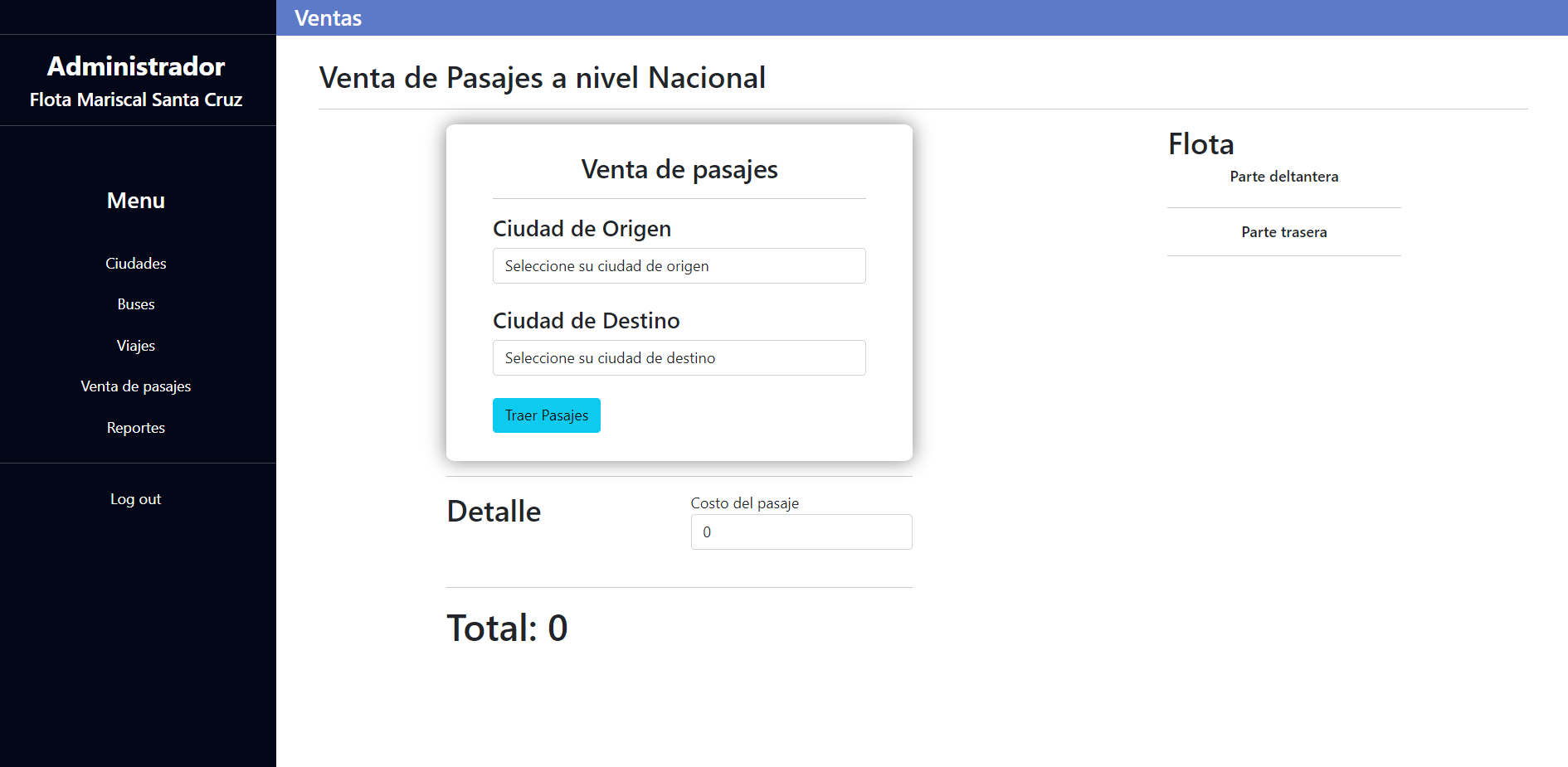
**Figura 26**: Eliminar Ciudad



**Figura 27:** Pantalla Buses



**Figura 28:** Pantalla Viajes



**Figura 29:** Pantalla Ventas



**Figura 30:** Pantalla Reportes