PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR

FACULTAD DE INGENIERÍA

ESCUELA DE SISTEMAS Y COMPUTACIÓN

TRABAJO PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO DE SISTEMAS Y COMPUTACIÓN

"ANÁLISIS Y DESARROLLO DE UN SISTEMA PARA GESTIONAR UN PARQUEADERO. CASO DE ESTUDIO: KENNEDY PARKING "

AUTOR: JAIME DAVID MALDONADO NÚÑEZ

DIRECTOR: ING. FABIÁN DE LA CRUZ

QUITO, ABRIL 2021

Dedicatoria

El presente trabajo va dedicado principalmente a mis padres, mi hermano que han sido el pilar fundamental que me ha ayudado a seguir adelante, a pesar de las caídas que tuve, siguieron confiando mí y nunca me falto sus consejos u apoyo.

A mi pareja, que ha sido la persona que ha estado para escucharme y sobre todo apoyarme en cualquier circunstancia.

Agradecimiento

Quiero agradecer a Dios por darme la oportunidad de estudiar en la PUCE, para mí es un honor haber conocido a excelentes profesionales, además de iluminar mi camino con personas correctas para cumplir los objetivos que me he planteado.

A mis padres y mi hermano, por el apoyo incondicional que nunca me falta y por ser mi guía a pesar de las victorias o derrotas que he pasado durante toda mi vida.

A mis amigos con quienes compartí veladas extensas para lograr un objetivo en común, que a pesar de cualquier circunstancia siempre me sacaban una sonrisa.

Al Ingeniero Fabián De La Cruz, por sus consejos y conocimiento tanto en el trabajo de titulación como en mi formación académica, a mis revisores por sus consejos en el presente trabajo.

Contenido

Tabla de Ilustraciones

CAPITULO I: INTRODUCCIÓN

Actualmente, se puede observar en la ciudad varios establecimientos, que simplemente no tienen automatizado su servicio de parqueadero, entre estos inconvenientes, surgió la idea de realizar el proyecto de investigación.

Se plantea el análisis y desarrollo de este sistema dentro del proyecto de disertación, puesto que la aplicación será usada por el administrador del negocio o encargado, además de ser quien controla todo el sistema y puede replicar en otros negocios de la misma naturaleza.

Mediante el desarrollo de este sistema, el propietario del establecimiento podrá tener un control en todos los aspectos de este negocio, evitando así la pérdida de la información y llevando un mejor control de sus ingresos.

Para llevar a cabo este proyecto de titulación se utilizará la metodología en cascada, usando orientación a objetos, además si el proceso que vamos a desarrollar está bien definido, no necesitamos de la interacción con el usuario final ni la necesidad de realizar iteraciones.

1.1 Datos del Caso de Estudio

Nombre: Kennedy Parking.

Ubicación: Avenida Pompilio Ulloa Reyes y Avenida 9 NO Guayaquil, Ecuador.

Descripción: Hace 12 años, este establecimiento abre sus puertas para el público, ofreciendo el servicio de estacionamiento para todo tipo de vehículos y motocicletas.

Horario de atención: De lunes a viernes de 8:00 hasta las 20:00.

1.3 Planteamiento del problema

La necesidad de automatizar algún proceso es algo clave, en cualquier tipo de empresa, ya sea pequeña o grande. En nuestro caso de estudio, se basa en implementar un software para la gestión de parqueaderos que, en muchos casos se lleva un control manual del mismo, por lo tanto, está sujeto a que se cometa errores e incluso que la información reportada manualmente sea incorrecta al momento de generar algún tipo de reporte.

Este problema no es solo para el administrador, sino también para el usuario final, ya que muchas veces al momento de registrar el ingreso de un cliente, se puede provocar errores inesperados, por lo cual con este proyecto de disertación se desea realizar una aplicación, que además de gestionar un parqueadero, nos informe antes de ingresar al parqueadero, verificar si hay cupos disponibles, con el objetivo de que no entren más vehículos de los que está permitido y evitar pérdida de tiempo por parte del usuario.

Este problema es muy frecuente en la ciudad, ya que actualmente hay demanda de parqueos, incluso por la medida “hoy no circula” que fue puesta en vigencia desde septiembre del año anterior, por lo tanto, debemos automatizar este proceso, ya que

ayudará a brindar un mejor servicio tanto para el administrador como para el usuario final.

Al momento de implementar un sistema de gestión, implica un cambio cultural y agregación de valor, además de no perder dinero en malas gestiones por hacer el proceso manualmente. También favorece un proceso de cambio y un cierto tipo de ventajas competitivas para la organización.

1.4 Alcance

El tema propuesto propone solucionar el registro manual de gestión de parqueaderos, por lo que el resultado de este proyecto será entregar un sistema que permita el registro de clientes que deseen pagar por mes, por hora o fracción. Aparte de generar un reporte que permita visualizar los ingresos que se obtuvo por una determinada fecha. Se realizarán estudios descriptivos para elaborar un modelo entidad-relación que satisfaga los requerimientos propuestos por el usuario.

El trabajo de titulación culminará con la entrega del sistema del proyecto de titulación.

1.5 Objetivos

General

- Análisis y desarrollo de un sistema para gestionar un parqueadero.

Específicos

- Analizar la infraestructura del establecimiento, para cumplir las expectativas del usuario con respecto al sistema.
- Desarrollar Implementar un sistema completo, funcional, en el caso de estudio correspondiente.
- Aplicar una metodología, como guía mediante los casos de uso.

La presente disertación está distribuida de la siguiente manera: en el Capítulo 1 se trata de presentar la problemática a resolver, además de la información del caso de estudio designado; en el Capítulo 2 se presenta los fundamentos teóricos donde se aborda las herramientas, metodologías, Frameworks, arquitecturas, entre otros; en el Capítulo 3, se presenta el Diseño de sistema y modelamiento mediante casos de uso, mediante el cual en esté capitulo se adjuntará la documentación y desarrollo de software; en el Capítulo 4 se presenta las conclusiones, recomendaciones y bibliografía.

CAPÍTULO II: FUNDAMENTOS TEÓRICOS

En el presente capítulo revisaremos las herramientas y metodologías de desarrollo de software, detallándolas de manera amplia. Además, se escogerán la apropiadas para satisfacer las necesidades del caso de uso correspondiente.

2.1 Metodologías de Desarrollo de Software

Al momento de desarrollar un proyecto, pensamos en cómo debemos organizar el proceso de desarrollo. Para esto, no existe un único camino, existen varias formas tanto de documentación como de codificación.

“Las metodologías de desarrollo son una batalla entre el dogmatismo y el pragmatismo.” (AllianceSoftware, 2018).

Se puede definir a una metodología desarrollo de software como un conjunto de procesos, normas que proveen un enfoque para interpretar la realidad y escoger la metodología más apropiada para cada proyecto.

2.1.1 Metodologías Agiles

Este tipo de metodologías promueven una disciplina en la gestión de proyectos para lograr una satisfacción del usuario final, además de fomentar el liderazgo y trabajo en equipo asignando roles a cada uno de los miembros del equipo. El desarrollo de software ágil hace referencia a un desarrollo iterativo, es decir, tanto los requisitos como las soluciones evolucionan mediante la colaboración de los equipos durante cada fase. A continuación, se hablará de algunas metodologías ágiles.

2.1.1.1 Scrum

Scrum es un marco de trabajo que se basa en el desarrollo de problemas complejos, cabe destacar que Scrum no se considera como un proceso o una técnica; este marco se basa en roles, eventos y artefactos. Los miembros que conforma Scrum son los siguientes:

Roles

- **Product Owner:** Es el dueño del producto, además de ser la voz del cliente para informar todos los avances y el valor de cada iteración.
- **Scrum Master:** Es la persona que se enfoca en que se cumpla Scrum en todo el proceso, enfocarse en la mejora continua y ser un apoyo y guía para el equipo Scrum

- **Development Team:** Está formado por un máximo de 9 personas, es un equipo autoorganizado, la responsabilidad es siempre compartida, además siempre se enfoca hacia el objetivo del proyecto.

Eventos

- **Sprint:** Es el corazón de Scrum, que se crea en un tiempo que debe ser de un mes o menos, en cada iteración, debe existir un incremento de valor y no puede pasar al siguiente sin terminar el anterior (Schwaber & Sutherland, 2017).

- **Sprint Planning:** Se produce el primer día del inicio del sprint, los miembros del equipo determinan cuantas tareas del Product Backlog podrán completar.

- **Daily Scrum:** Se lleva a cabo por la mañana y ayuda a establecer que trabajo se realizará al día siguiente. Estas reuniones de Scrum están estrictamente delimitadas a 15 minutos.

- **Sprint Retrospective:** En esta fase el equipo analiza que acciones de mejora deben tomar para el siguiente sprint. Es decir, es una oportunidad para observar la productividad y la calidad del producto. La duración de este evento es de máximo 45 minutos por cada semana que transcurrió desde que inicio el sprint.

- **Sprint Review:** En esta etapa se prioriza el Product Backlog, se observa cual fue el incremento de el presente sprint y que tareas van a ser las que se prioricen para el siguiente, el Product Owner es el encargado de potenciar el entendimiento de incremento.

Artefactos

- **Product Backlog:** Es una lista de requerimientos por parte de los clientes (stakeholders), deben estar escritos en un lenguaje no formal, no redundantes, que cubran todas las necesidades del producto.

- **Sprint Backlog:** Son las tareas que se decide realizar en un sprint en específico, usualmente las primeras tareas son las que generan mayor valor para el producto.

- **Product Increment:** Un incremento es un conjunto de trabajo realizado e inspeccionable que respalda el empirismo al final del Sprint, debe estar en condiciones utilizables.

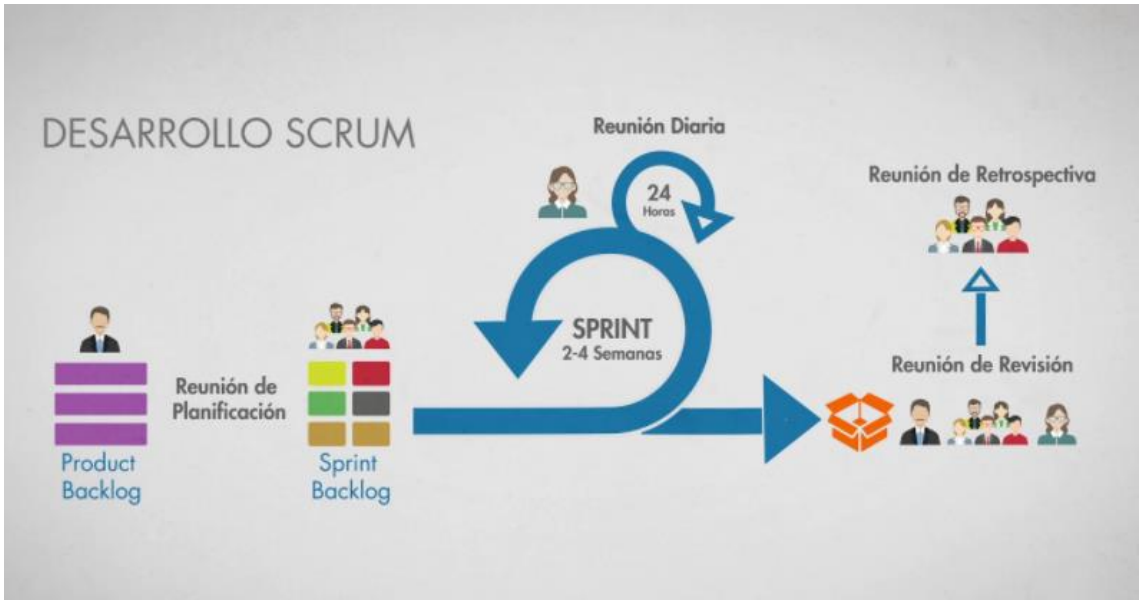


Ilustración 1 Metodología Scrum (Guzman, 2013)

En la ilustración 1 podemos observar cómo se desarrolla esta metodología. Para comenzar tenemos el Product Backlog, que es una lista clasificada de historias de usuario creadas por el Product Owner, la misma que contendrá todos los requerimientos por parte del usuario, una historia de usuario es un requerimiento del cliente escrito en un lenguaje no formal (Guzman, 2013).

Posteriormente pasamos al primer Sprint, el evento que es el Sprint Planning, el equipo escoge las tareas de mayor prioridad para realizarlas durante ese Sprint y lo almacenamos en el Sprint Backlog, usualmente se usa un tablero Kanban para saber el avance de estas planificaciones. El equipo se dedica a cumplir las tareas según la duración que tenga el sprint, que no puede ser más allá de 4 semanas, el encargado de que se cumpla los tiempos y apoyar al equipo es el Scrum Master. Al finalizar cada sprint pasamos al evento del Sprint Review, aquí se analiza cuáles fueron las tareas que se terminaron y cuales se van a priorizar para el siguiente sprint (incremento), en este punto intervienen todos los miembros del equipo y los stakeholders y para finalizar pasamos al Sprint Retrospective, donde el equipo analiza que acciones mejorar para el siguiente ciclo de desarrollo, con el fin de incrementar la productividad y generar mayor valor en cada ciclo.

Entre las desventajas de Scrum podemos encontrar lo siguiente:

- No se deben mover los plazos al momento de implementarla.
- Es una metodología que requiere una formación con bastante experiencia en la misma, ya que gran parte del éxito del proyecto radica en el aporte de los miembros del equipo.

2.1.2 Metodologías Estructuradas

Se centran básicamente en la especificación de requerimientos; el flujo de desarrollo es unidireccional desde los requisitos hasta el diseño, seguimos con el desarrollo, pruebas y mantenimiento.

Uno de los modelos más conocidos que emplea esta metodología es el modelo en cascada y que en cada fase tiene documentos entregables con documentación detallada.

Este tipo de enfoques se usa cuando los requerimientos y el dominio del problema se encuentran bien definidos, ya que no vamos a tener una retroalimentación por parte de ningún usuario, es decir que el responsable comprende claramente todo el proceso del producto final. Además, que el usuario tampoco podrá hacer evaluaciones en cada avance del proyecto. Las metodologías estructuradas se clasifican en:

Orientadas a procesos

Parte de la definición de proceso, que es el que de un determinado número de entradas produce salidas, esta se centra en el proceso.

Esta metodología emplea un método descendente de descomposición funcional en el cual se determina las condiciones del sistema empleando una serie de técnicas visuales para dar lugar a un concepto de especificación estructurada. (Universidad de Valladolid, 2009)

Las metodologías orientadas a procesos son las siguientes:

- DeMarco
- Gane y Sarson
- Yourdon

Orientadas a datos

Esta metodología parte el mismo concepto de proceso, pero se centra en las entradas y salidas. En esta estructura se escoge la estructura de datos, de las cuales se deriva la estructura de control, los componentes procedimentales. (Universidad de Valladolid, 2009)

Las metodologías orientadas a datos son las siguientes:

- Warnier
- Jackson
- Cameron

2.1.2.1 Metodología en Cascada

El modelo en cascada fue el primero modelo de desarrollo de software que se introdujo y además es muy sencillo de entender y utilizar, este modelo se divide en fases separadas, pero cada fase actúa como entrada de la siguiente fase, de manera secuencial, esto quiere decir que se debe completarse una fase para continuar con la siguiente, por esto también se dice que es bastante preciso cuando la inspección en todas las fases es realizada de manera adecuada.

Este proceso ilustra un proceso de manera lineal, por lo que también es llamado modelo de ciclo de vida lineal secuencial (Z, Lage, Pessacq, & García Martínez, 2017)

Este modelo consta de las siguientes fases:

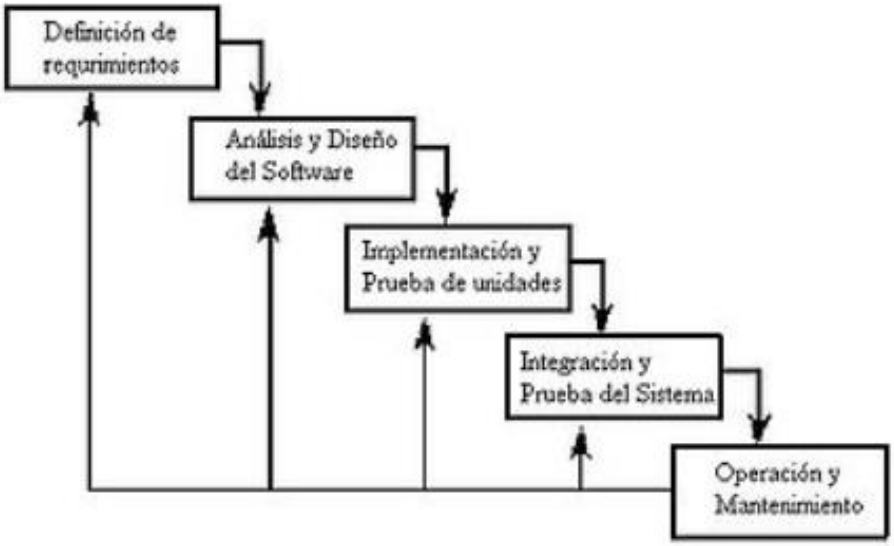


Ilustración 2 Modelo de cascada (Gutierrez, Modelos de Desarrollo, 2011)

En la Ilustración 2 se puede observar que consisten en:

Definición de Requerimientos: En esta fase se evalúa y se comprende el proceso a resolver, se trata de comprender todas las necesidades del cliente mediante preguntas, usualmente también se elaboran lluvias de ideas para luego acomodarlas a lo que el usuario necesite. En esta fase debemos comprobar que sean viables los requerimientos, no ambiguos. En esta fase se detalla los actores, diagramas de casos de uso detallado.

Análisis y Diseño del Software: Se describe el diseño computacional detallado y el diagrama de arquitectura para la aplicación. En esta fase se describirá cada módulo, como se implementará y probará.

Implementación y Prueba del Sistema: En esta fase se lleva a cabo la codificación del software de acuerdo con el diseño y a su vez se producen pruebas de cada módulo, con las cuales verificamos si se cumplió el funcionamiento acorde a los requerimientos previstos. En esta fase se entregará todos los módulos del sistema completo, con sus respectivas pruebas de módulos.

Integración y Prueba del Sistema: Se hace una prueba como un sistema completo y verificar el cumplimiento de los requerimientos solicitados por el cliente. En esta fase se entregan las pruebas de aceptación y del sistema.

Funcionamiento y Mantenimiento: Esta etapa consiste en corregir defectos que se inyectaron en fases anteriores, probar en ambiente de producción el sistema y verificar que cada módulo responda como fue previsto en el plan de pruebas.

Ventajas:

- Es un modelo fácil de entender y utilizar.
- Funciona bien en proyectos pequeños y bien definidos.
- Los resultados son documentados.

Desventajas:

- No es sujeto a cambios.
- No funciona con proyectos largos, ya que hay mucho riesgo al cambio.
- El cliente no obtiene un avance sino hasta cuando ya está elaborado el producto.

En conclusión, el modelo en cascada sirve siempre que tengamos los procesos bien definidos, el cliente no necesite una retroalimentación sino hasta el final del proyecto, además si los requisitos son sencillos y comprobables este modelo producirá mejores resultados.

2.2 Arquitecturas de Software

Las arquitecturas de Software se basan principalmente en dos: cliente servidor y arquitectura MVC.

2.2.1 Cliente Servidor

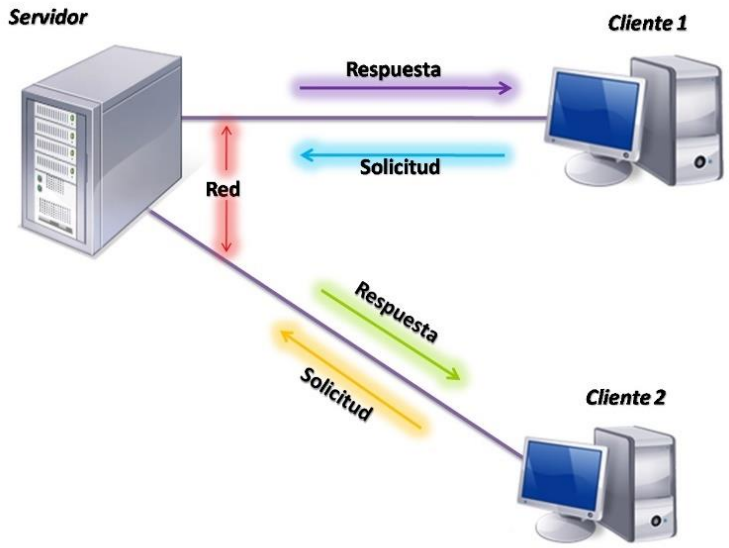


Ilustración 3 Arquitectura Cliente Servidor (Gomez, 2020)

Como se observa en la Ilustración 3 las aplicaciones cliente servidor, son aplicaciones que están a la espera de instrucciones de una computadora para comunicarse con el servidor, puede existir muchos clientes, sin embargo, la diferencia entre ambos es que el servidor tiene mucha más potencia, ya que se puede comunicar con una gran cantidad de clientes, sin embargo, el cliente es el que realiza el procesamiento, por lo que se necesita mayores recursos, usualmente se usa una conexión mediante ODBC (Bravo, 2019).

2.2.2Aplicaciones Web

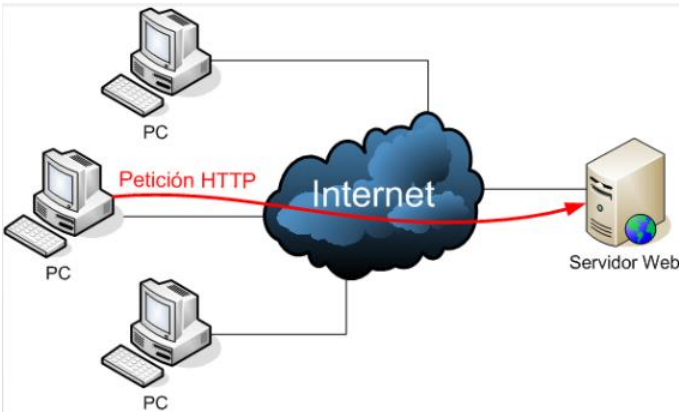


Ilustración 4 Aplicación Web (Anónimo, mrHouston, 2018)

Como se observa en la Ilustración 4, las aplicaciones web se destacan ya que el procesamiento está en el servidor, a diferencia que la anterior mencionada se ejecuta en el cliente por lo que no debemos preocuparnos por la infraestructura ni de disponer de equipos con un procesamiento grande, únicamente con una conexión a internet estable.

2.3 Patrón MVC

Un sistema web es aquel que utiliza tecnologías web que se basa en código html, css principalmente para entregar información y servicios, que se rige comúnmente a un marco MVC, además es un sistema basado en hipertexto.

Este patrón es el más utilizado, ya que separa al sistema en 3 componentes: el modelo, la vista y el controlador, tal como se presenta en la Ilustración 5.

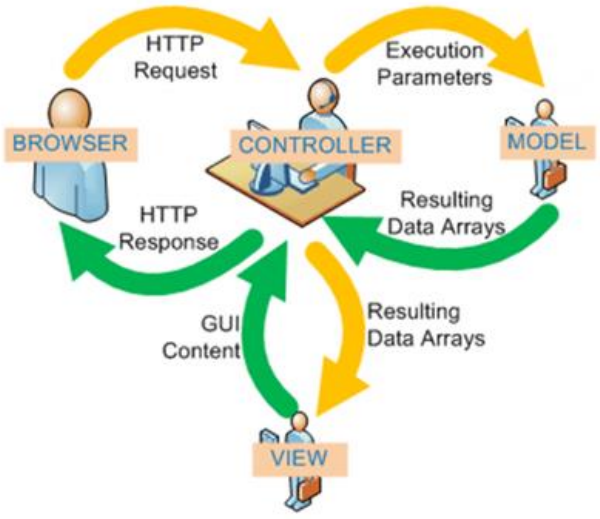


Ilustración 5 Arquitectura MVC (Alcant, 2020)

Vista: Es la capa que contiene todas las acciones del usuario(eventos).

Modelo: Es el que relaciona los datos con los cuales una aplicación va a operar, como: consultas, eliminación, modificación, entre otras. A esto también se le denomina lógica del negocio.

Controlador: El cual responde a eventos o acciones que realiza el usuario a la vista, también es responsable de elegir que vista tiene que mostrar al usuario según la manipulación del usuario, es decir es el vínculo entre el modelo con la vista. . (Edwar & Grace, 2008)

Al momento de usar este tipo de aplicaciones, este servicio web se aloja en un hosting con un dominio específico, con esto no necesitamos un servidor, sino basta con una conexión a internet. (EcuEed, 20)

2.4 Lenguajes de Desarrollo Web

El desarrollo web es un término que se utiliza para hacer énfasis al proceso de creación de sitios web, a esto nos referimos como el desarrollo de una página hasta la creación de un sistema entre otras cosas. Abarcando varios temas y acciones, entre las que podemos destacar el diseño, creación de contenido (programación), seguridad en la página, entre otras.

Un lenguaje de programación se divide en dos compontes que son la semántica y la sintaxis. Donde la sintaxis es la forma y la semántica es el significado; cabe destacar que cada lenguaje es muy diferente, sin embargo, la lógica es la misma.

Los lenguajes de programación web son lenguajes interpretados, es decir necesitan de un intérprete que ejecute el código fuente. (Escobar, 2017).

Entre los principales lenguajes de programación son: C#, Java, Python, Php, entre otros.

2.5 Marcos de Trabajo (Frameworks)

Desarrollar software es un proceso complejo, ya que nos debemos enfocar en el diseño, codificación, pruebas, sin embargo, una de las tareas en la que demanda más tiempo es la codificación, ya que demanda tiempo en escribir líneas de código, reparar errores, optimizar funciones, etc.

(Gutiérrez, 2014) menciona “una estructura software compuesta de componentes personalizables e intercambiables para el desarrollo de una aplicación.”

Es decir, cuando usamos este tipo de herramientas, debemos ocuparnos principalmente por las funcionalidades de alto nivel, ya que estos se encargan de las de bajo nivel.

Entre las ventajas que nos ofrece son:

- Se involucra a buenas prácticas según el patrón de diseño.
- Probar el código es más sencillo

- El tiempo de desarrollo se reduce significativamente

Sin embargo, muchos desarrolladores optan por realizar un desarrollo sin el uso de estos, ya que muchos de estos marcos de trabajo cambian muy rápido y ciertas funcionalidades quedan obsoletas, lo que significa que el desarrollo se vuelve más complejo, cuando se quiere implementar una funcionalidad a futuro.

2.6 Bases de Datos

(Pérez Valdez, 2007) menciona "la base de datos es un sistema formado por un conjunto de datos almacenados en discos que permiten el acceso directo a ellos", para organizar la información y mantener de manera que sea más confiable y que no sea de fácil acceso. Existen dos tipos: bases de datos relacionales y no relacionales.

2.6.1 Bases de Datos Relacionales

Los datos están estructurados en filas y columnas, cada fila de la tabla posee un id único llamado clave primaria, para manipular este tipo de bases de datos usamos el lenguaje SQL.

Los motores de bases de datos más populares son: MySQL, PosgreSQL, SQLite, Microsoft SQL Server, DB2, entre otras. (Pérez Valdez, 2007). Entre las ventajas de usar bases de datos relacionales son:

- Los datos se pueden almacenar en diferentes tablas.
- Fácil de acceder a los datos.
- Es un modelo consistente y fácil de entender.
- Los datos pueden ser compartidos.

2.6.2 Bases de Datos no Relacionales

También denominadas "No SQL", ya que no usan el lenguaje SQL para consultas, ya que no almacenan sus datos en tablas, sino como documentos, se dice que son base de datos pueden tener estructuras orientadas a documentos, grafos, objetos. (Morales, 2013).

Los motores más populares son: MongoDB, Redis, CuchGB. (Telefónica, 2018)

Entre las principales ventajas que podemos destacar son:

- Posee un alto rendimiento y bajo costo, ya que pueden ejecutarse con recursos muy bajos, por lo tanto, la inversión es baja.
- No hay inconveniente con el volumen de datos, ya que es muy fluido
- Posee mucha flexibilidad, ya que se ajustan a cualquier requerimiento.

2.7 Herramientas de Desarrollo y Metodología a usar

El criterio para seleccionar las fuentes de análisis y poderlas comparar, fue con ayuda del buscador Google, mediante palabras clave como: “sistema para parqueaderos”, “sistema de parqueaderos Ecuador”, ya que con estas frases clave, se desplegaron varios sistemas, aplicaciones, publicaciones relacionadas con el tema.

La mayoría de información se encontró en fuentes informales como son las páginas de internet. Entre las fuentes escogidas para el estudio, se encontraron varios tipos de aplicaciones, entre ellos se encontraron principalmente:

En relación con otros sistemas web, este se diferencia a que la mayoría de sistemas para gestión de parqueaderos son sistemas cliente/servidor que con el pasar del tiempo se vuelven sistemas obsoletos, ya que no se pueden conectar con otras tecnologías emergentes que se desarrollan a diario, al momento de hacer un sitio web, agregamos portabilidad para el usuario, ya que no necesita una infraestructura, al contrario solo un servicio de internet, además de cancelar un valor de hosting y para el usuario es más fácil interactuar con un sitio web, ya que cualquier tipo de actualización sin que el usuario necesite ayuda del administrador, por este motivo este proyecto será un sistema web.

2.7.1 Power Designer

Es una herramienta desarrollada por SAP, mediante la cual podemos modelar, diseñar diferentes tipos de diagramas, flujos, paquetes, así también proporcionar una estructura a las bases de datos relacionales.

Para el presente trabajo se usará para diseñar la base de datos, casos de uso, diagrama de paquetes, de actividades, entre otras cosas.

2.7.2 Visual Studio Code

Es un editor de código potente y bastante ligero, el motivo por el cual decidí seleccionar esta herramienta es porque posee plugins que nos ayuda a que la codificación sea más sencilla, autocompletando código redundante.

2.7.3 Laravel

Es un marco PHP, que fue lanzado en el 2011 por Taylow Otwell como un framework de código abierto creado para que el desarrollo de aplicaciones sea de manera limpia, reutilizable y elegante gracias a sus funciones integradas, con una buena curva de aprendizaje, lo que hace que sea uno de los más usado por los desarrolladores web. Este tipo de marco utiliza el patrón MVC, por lo que facilita la búsqueda de archivos en proyectos grandes.

Laravel incluye un motor de plantillas Blade, que es bastante potente y liviano, posee un desarrollo fluido y varios tipos de diseño, sin afectar el rendimiento y la velocidad de la aplicación. (Cíceri, 2018).

Para el presente trabajo se usará la versión 7, que es la más estable hasta el momento.

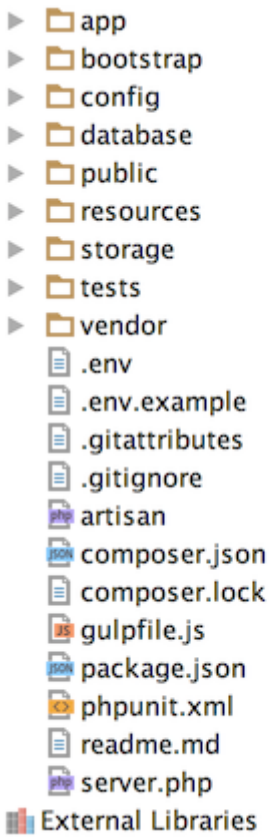


Ilustración 6 Estructura Laravel

2.7.4 MySQL

Es una de las bases de datos relacionales más populares del mundo, desarrollada por Sun MicroSystem en el año 2008 que luego fue comprado por Oracle en 2010, es compatible con la mayoría de los proveedores de alojamiento, podemos destacar su seguridad en los datos, además del sólido soporte transaccional, en el que se protege las transacciones en línea, cuenta con una licencia GPL (libre de copyleft), sin embargo posee otra versión gestionada por Oracle (pago) (Robledano, 2019).

2.7.5 UML

Es un enfoque moderno para modelar y documentar software, se puede decir que es una de las técnicas de modelamiento más usadas en la actualidad.

Se define principalmente para elaborar el análisis y el diseño de un sistema. No es una metodología, lo único que nos dice UML es una notación con la finalidad de plasmar tanto el análisis como el diseño del sistema mediante modelos.

Describe los componentes de software mediante representaciones visuales, ya que de esta manera se comprende mucho más fácil los procesos del software, así como también los errores que se inyecten en alguna etapa del ciclo de vida del desarrollo (Ferré Grau & Sánchez Segura, 2017).

CAPÍTULO III CASO DE ESTUDIO

Kennedy Parking, es un establecimiento que no posee una herramienta de automatización. Los clientes que llegan y salen son tarifados por una persona, así como los ingresos generados son llevados en cuadernos, los mismos que pueden ser extraviados y no tener evidencia.

Para el presente caso de estudio, se requiere automatizar la gestión de un parqueadero, lo que incluye: gestión de clientes, vehículos, caja, cochera, usuarios, roles y permisos, debido a que estos requerimientos con el tiempo es muy difícil que cambien, se decidió aplicar la metodología en cascada, puesto a que en una sola iteración se podrá automatizar estos procesos mencionados anteriormente, a continuación, se describirá las fases de la metodología mencionada.

3.1 Definición de requerimientos

3.1.1 Presentación del Sistema

Este sistema permitirá la gestión de parqueaderos, con roles y permisos, para evitar que el empleado modifique atributos que solo puede ser escritos por el administrador o dueño del establecimiento, además de llevar reportes de las ventas producidas por determinadas fechas, el cálculo de la tarifa automático, dependiendo de la hora de ingreso y de salida.

3.1.2 Tipos de usuarios

Administrador: Actor que tendrá todos los accesos al sistema

Empleado: Actor que tiene accesos restringidos como: abrir caja, eliminar un espacio de la cochera, crear usuarios, crear roles, entre otros que se detallaran en los casos de uso.

3.1.3 Alcance del Sistema

El principal alcance del sistema es, que el dueño del establecimiento tenga un control para sus ingresos, además que el cobro del servicio sea preciso, generar reportes, etc.

Para lo cual se describirán los siguientes requerimientos funcionales y no funcionales.

3.1.4 Requerimientos Funcionales y No Funcionales.

F0. Ingreso al Sistema.

F1. Administración de Clientes.

F2. Administración de Cajas.

F3. Administración de Tarifas

F4. Administración de Vehículos.

F5. Administración de Cocheras.

F6. Administración de Usuarios.

F7. Administración de Roles.

F8. Administración de Asignaciones.

NF1. Sistema fácil de utilizar.

NF2. Tenga disponibilidad 24/7.

La especificación de los requerimientos funcionales se los realizará a través de los modelos basados en Casos de Uso que se presentan a continuación

3.1.4.1 Diagrama de Casos de Uso

Los diagramas de casos de uso básicamente sirven para visualizar la interacción de los actores con el sistema en un alto nivel. Para construirlo se usa un conjunto de símbolos y asociaciones.

Actualmente se considera que los casos de uso son muy útiles en los proyectos de software, ya que representa las acciones y comportamiento de un sistema desde el punto de vista del usuario (Gutierrez, Casos de Uso, 2011)

Los casos de uso se representan con una forma ovalada, las figuras de los costados representan los actores.

En esta etapa, se utilizará los diagramas de casos de uso a nivel general y específico.

Caso de uso: Nivel General

Es un modelo que indica de manera gráfica todas las funcionalidades que tendrá el sistema, describiendo que actores (que pueden ser individuos o sistemas) intervienen en cada funcionalidad (Sommerville, 2011).

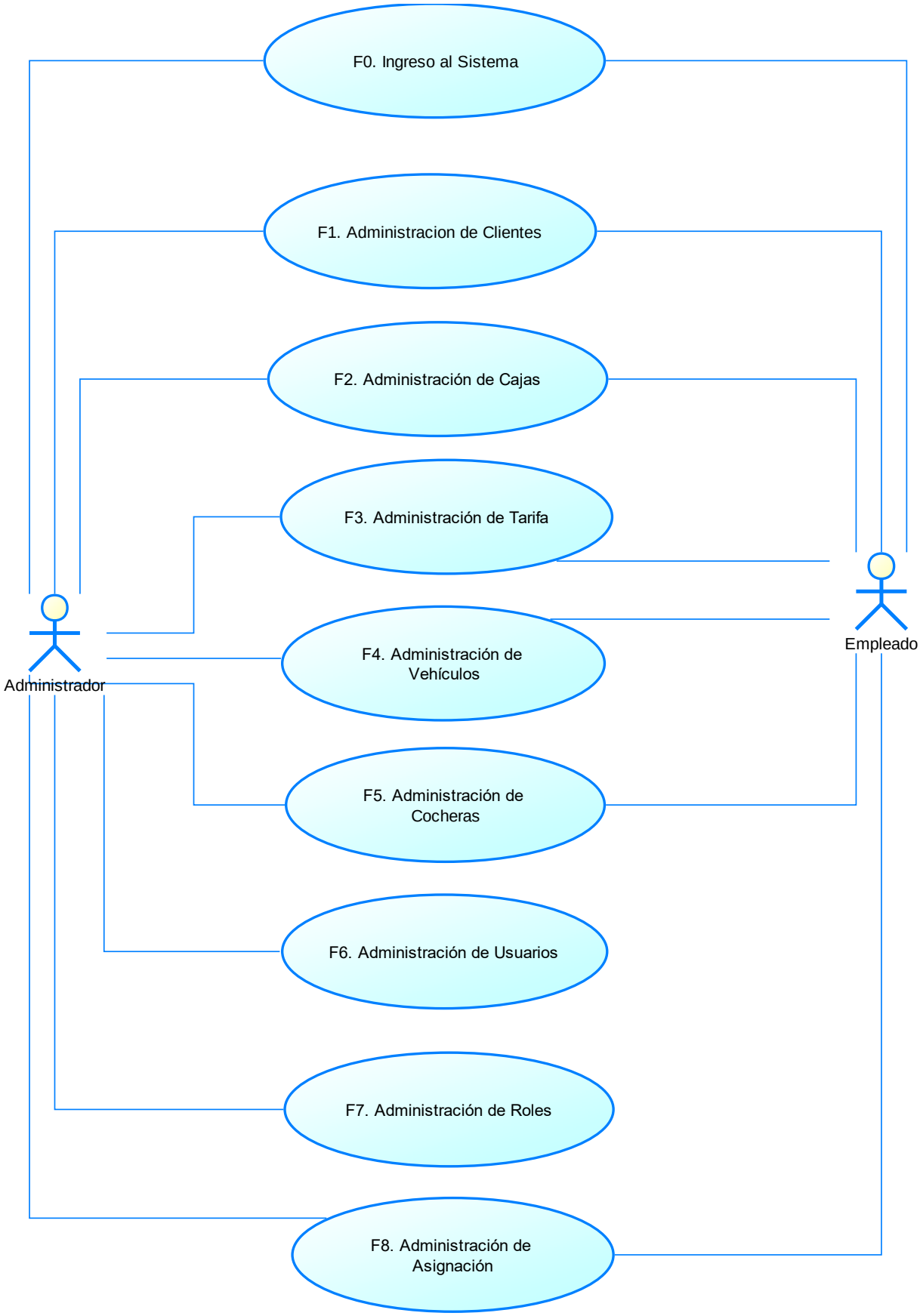


Ilustración 7 Diagrama casos de uso general

Casos de uso: A detalle

Estos modelos permiten describir de manera más detallada las funcionalidades que tendrá el sistema, con la interacción de los diferentes actores que posee.

F0. Ingreso al Sistema

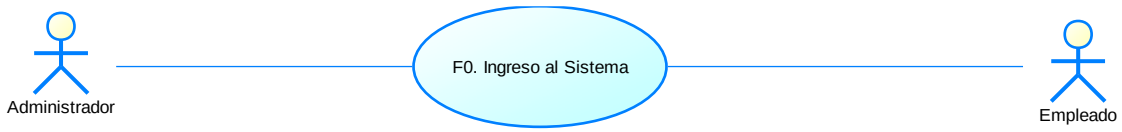


Ilustración 8 Caso de Uso a detalle Ingreso al Sistema

Descripción: Con esta funcionalidad el actor podrá ingresar al sistema.

Actores: Administrador, Empleado.

Flujo Principal:

1. El sistema despliega el menú de ingreso al sistema
2. El actor introduce el correo del usuario.
3. El actor introduce la contraseña del correo.
4. El actor presiona el botón Ingresar.
5. El sistema valida los datos ingresados (E1)
6. El sistema despliega la ventana principal

Flujo Alterno

1. Registrar usuario

Excepciones

Código	Causa	Mensaje
E1	Error de conexión con la base de datos	Contactar al soporte.

Considerando que el manejo de las funcionalidades F1, F2, F3, F5, F6, y F7 son similares, se presenta a continuación el Caso de Uso F1 Administrar Clientes, como el más representativo de este grupo. El resto están adjuntados en el Anexo 1.

F1. Administración de Clientes

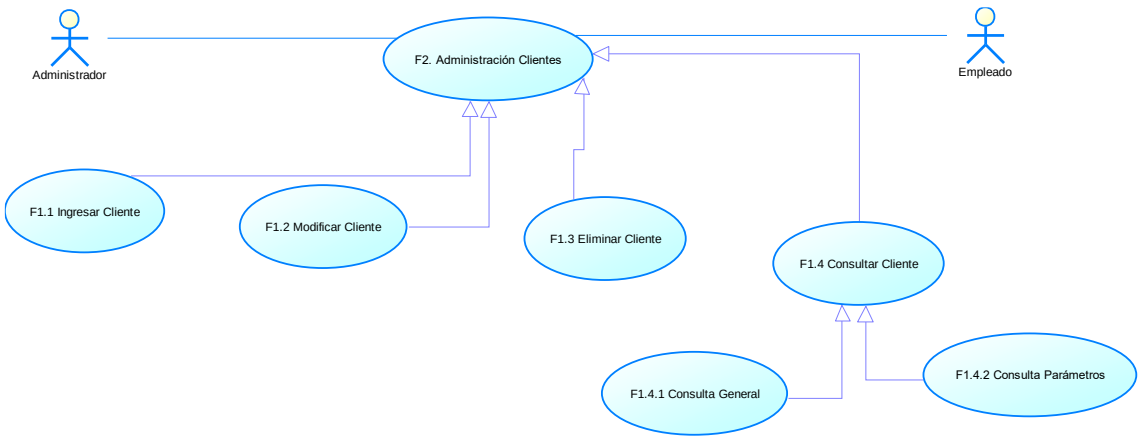


Ilustración 9 Caso de uso a detalle Administración Clientes

F1.1 Ingresar Cliente

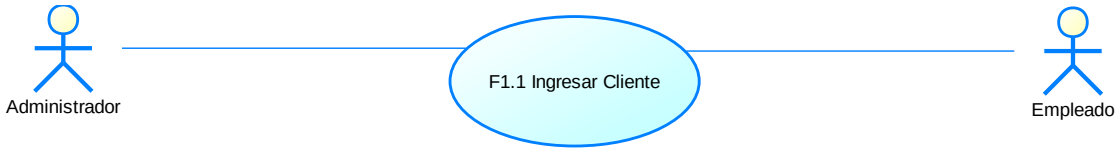


Ilustración 10 Ingresar Cliente

Descripción: Con esta funcionalidad, el actor podrá agregar un cliente al sistema con todos sus atributos.

Actores: Administrador, Empleado.

Flujo Principal:

1. El sistema despliega el menú principal.
2. El actor selecciona la opción clientes.
3. El sistema despliega la ventana de clientes (E1).
4. El actor presiona el botón Crear cliente.
5. El sistema despliega la ventana de creación de cliente.
6. El actor ingresa los campos solicitados.
7. El actor presiona el botón crear.
8. El sistema almacena la información (E1).
9. El sistema registra el nuevo cliente.

Flujo Alterno

1. Si ya existe, ver el caso de uso F1.2, F1.3.

Excepciones

Código	Causa	Mensaje
E1	Error de conexión con la base de datos	Contactar al soporte.

F1.2 Modificar Cliente

Ilustración 11 Modificar Cliente

Descripción: Con esta funcionalidad, el actor podrá modificar un cliente del sistema con todos sus atributos existentes.

Actores: Administrador, Empleado.

Flujo Principal:

1. El sistema despliega el menú principal.
2. El actor selecciona la opción clientes.
3. El sistema despliega la ventana de clientes (E1).
4. El actor presiona el botón Editar en el registro deseado.
5. El sistema despliega la ventana de edición de cliente.
6. El actor modifica los campos solicitados.
7. El actor presiona el botón guardar.
8. El sistema actualiza la información del cliente (E1).

Excepciones

Código	Causa	Mensaje
E1	Error de conexión con la base de datos	Contactar al soporte.

F1.3 Eliminar Cliente

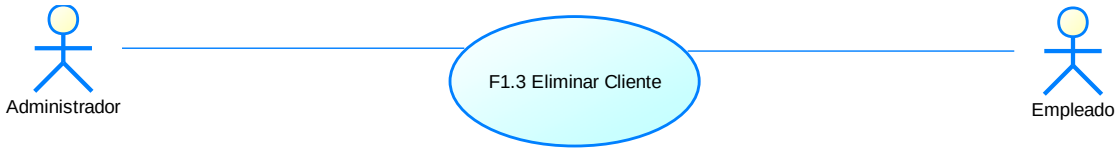


Ilustración 12 Eliminar Cliente

Descripción: Con esta funcionalidad, el actor podrá eliminar un cliente del sistema con todos sus atributos existentes.

Actores: Administrador, Empleado.

Flujo Principal:

1. El sistema despliega el menú principal.
2. El actor selecciona la opción clientes.
3. El sistema despliega la ventana de clientes (E1).
4. El actor presiona el botón Eliminar en el registro requerido.
5. El sistema despliega la ventana de confirmación.
6. El actor confirma la eliminación del ítem.
7. El sistema elimina el registro (E1).

Excepciones

Código	Causa	Mensaje
E1	Error de conexión con la base de datos	Contactar al soporte.

F1.4.1 Consulta General

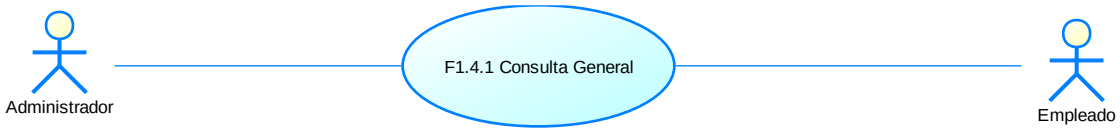


Ilustración 13 Consulta General Clientes

Descripción: Con esta funcionalidad, el actor podrá consultar todos los clientes del sistema con todos sus atributos existentes.

Actores: Administrador, Empleado.

Flujo Principal:

1. El sistema despliega el menú principal.
2. El actor selecciona la opción clientes.
3. El sistema despliega la información (E1).

Excepciones

Código	Causa	Mensaje
E1	Error de conexión con la base de datos	Contactar al soporte.

F1.4.2 Consulta por Parámetros

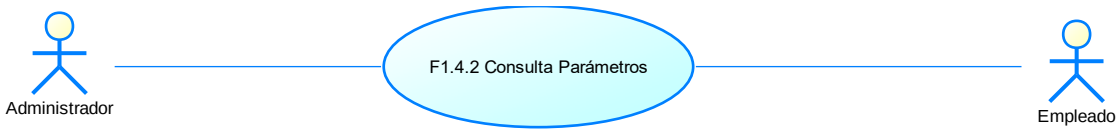


Ilustración 14 Consulta Parámetros Cliente

Descripción: Con esta funcionalidad, el actor podrá consultar clientes bajo parámetros con todos sus atributos existentes.

Actores: Administrador, Empleado.

Flujo Principal:

1. El sistema despliega el menú principal.
2. El actor selecciona la opción clientes.
3. El sistema despliega la ventana (E1).
4. El actor escribe la palabra de búsqueda.
5. El sistema despliega la información (E1).

Excepciones

Código	Causa	Mensaje
E1	Error de conexión con la base de datos	Contactar al soporte.

F4. Administración de Vehículos

El proceso de Administración de Vehículos cubre el siguiente diagrama de actividades y se complementa con su respectivo caso de uso.

Diagrama de actividades

Este tipo de diagrama describe los aspectos dinámicos del sistema, interactuando con los diferentes actores. Se puede decir que es una versión mejorada del diagrama de flujo (Sommerville, 2011).

A continuación, se presentará los diagramas de actividades de los procesos.

Diagrama de Actividades de Registro de vehículo

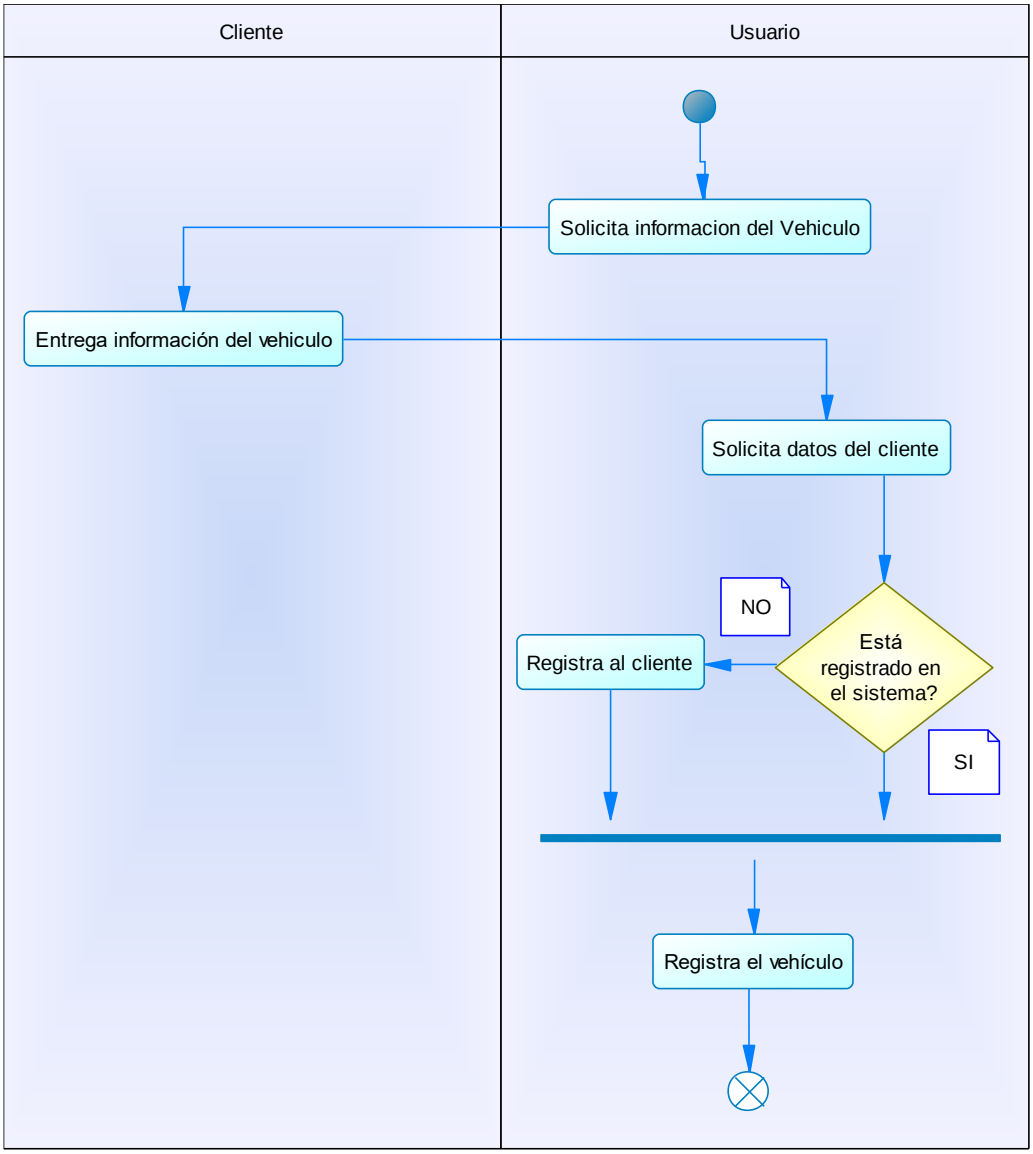


Ilustración 15 Diagrama de Actividades Registro Vehículo

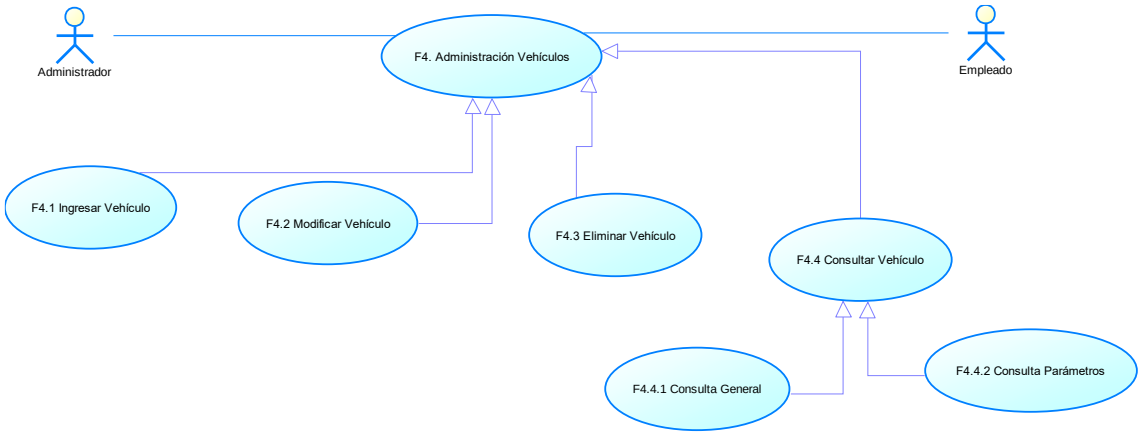


Ilustración 16 Caso de uso a detalle Vehículos

F4.1 Ingresar Vehículo

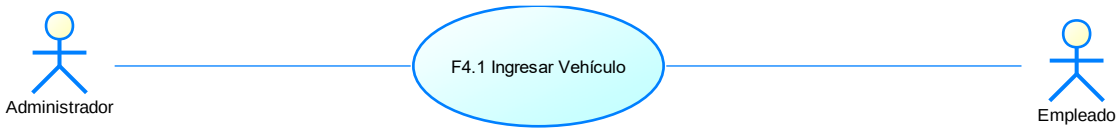


Ilustración 17 Ingresar Vehículo

Descripción: Con esta funcionalidad, el actor podrá agregar un vehículo al sistema con todos sus atributos.

Actores: Administrador, Empleado.

Flujo Principal:

1. El sistema despliega el menú principal.
2. El actor selecciona la opción vehículos.
3. El sistema despliega la ventana de vehículos (E1).
4. El actor presiona el botón Crear vehículos.
5. El sistema despliega la ventana de creación de vehículos.
6. El actor ingresa los campos solicitados.
7. El actor presiona el botón crear.
8. El sistema almacena la información (E1).
9. El sistema registra el nuevo cliente.

Flujo Alterno

1. Si ya existe, ver el caso de uso F2.2, F2.3.

Excepciones

Código	Causa	Mensaje
E1	Error de conexión con la base de datos	Contactar al soporte.

F4.2 Modificar Vehículo

Ilustración 18 Modificar Vehículo

Descripción: Con esta funcionalidad, el actor podrá modificar un vehículo del sistema con todos sus atributos existentes.

Actores: Administrador, Empleado.

Flujo Principal:

1. El sistema despliega el menú principal.
2. El actor selecciona la opción vehículos.
3. El sistema despliega la ventana de vehículos (E1).
4. El actor presiona el botón Editar en el registro deseado.
5. El sistema despliega la ventana de edición de vehículos.
6. El actor modifica los campos solicitados.
7. El actor presiona el botón guardar.
8. El sistema actualiza la información del vehículo (E1).

Excepciones

Código	Causa	Mensaje
E1	Error de conexión con la base de datos	Contactar al soporte.

F4.3 Eliminar Vehículo

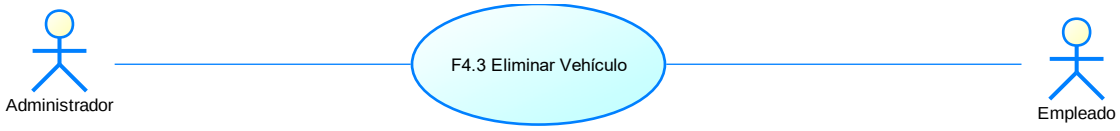


Ilustración 19 Eliminar Vehículo

Descripción: Con esta funcionalidad, el actor podrá eliminar un vehículo del sistema con todos sus atributos existentes.

Actores: Administrador, Empleado.

Flujo Principal:

1. El sistema despliega el menú principal.
2. El actor selecciona la opción vehículos.
3. El sistema despliega la ventana de vehículos. (E1).

4. El actor presiona el botón Eliminar en el registro requerido.
5. El sistema despliega la ventana de confirmación.
6. El actor confirma la eliminación del ítem.
7. El sistema elimina el registro (E1).

Excepciones

Código	Causa	Mensaje
E1	Error de conexión con la base de datos	Contactar al soporte.

F4.4.1 Consulta General

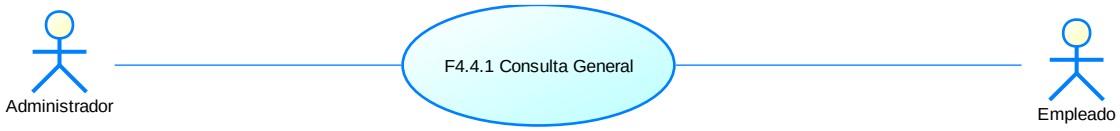


Ilustración 20 Consulta General Vehículos

Descripción: Con esta funcionalidad, el actor podrá consultar todos los vehículos del sistema con todos sus atributos existentes.

Actores: Administrador, Empleado.

Flujo Principal:

1. El sistema despliega el menú principal.
2. El actor selecciona la opción vehículos.
3. El sistema despliega la información (E1).

Excepciones

Código	Causa	Mensaje
E1	Error de conexión con la base de datos	Contactar al soporte.

F4.4.2 Consulta por Parámetros

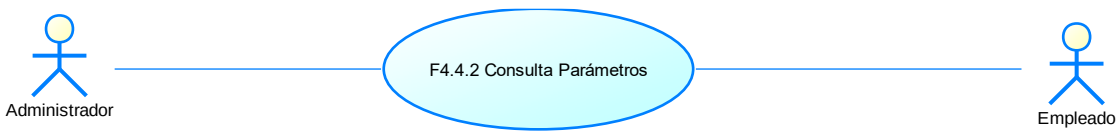


Ilustración 21 Consulta Parámetros Vehículo

Descripción: Con esta funcionalidad, el actor podrá consultar vehículos bajo parámetros con todos sus atributos existentes.

Actores: Administrador, Empleado.

Flujo Principal:

1. El sistema despliega el menú principal.
2. El actor selecciona la opción vehículos.
3. El sistema despliega la ventana (E1).
4. El actor escribe la palabra de búsqueda.
5. El sistema despliega la información (E1).

Excepciones

Código	Causa	Mensaje
E1	Error de conexión con la base de datos	Contactar al soporte.

F5. Administración de Cocheras

El proceso de Administración de Cocheras cubre el siguiente diagrama de actividades y se complementa con su respectivo caso de uso.

Diagrama de Actividades de Registro entrada de vehículo

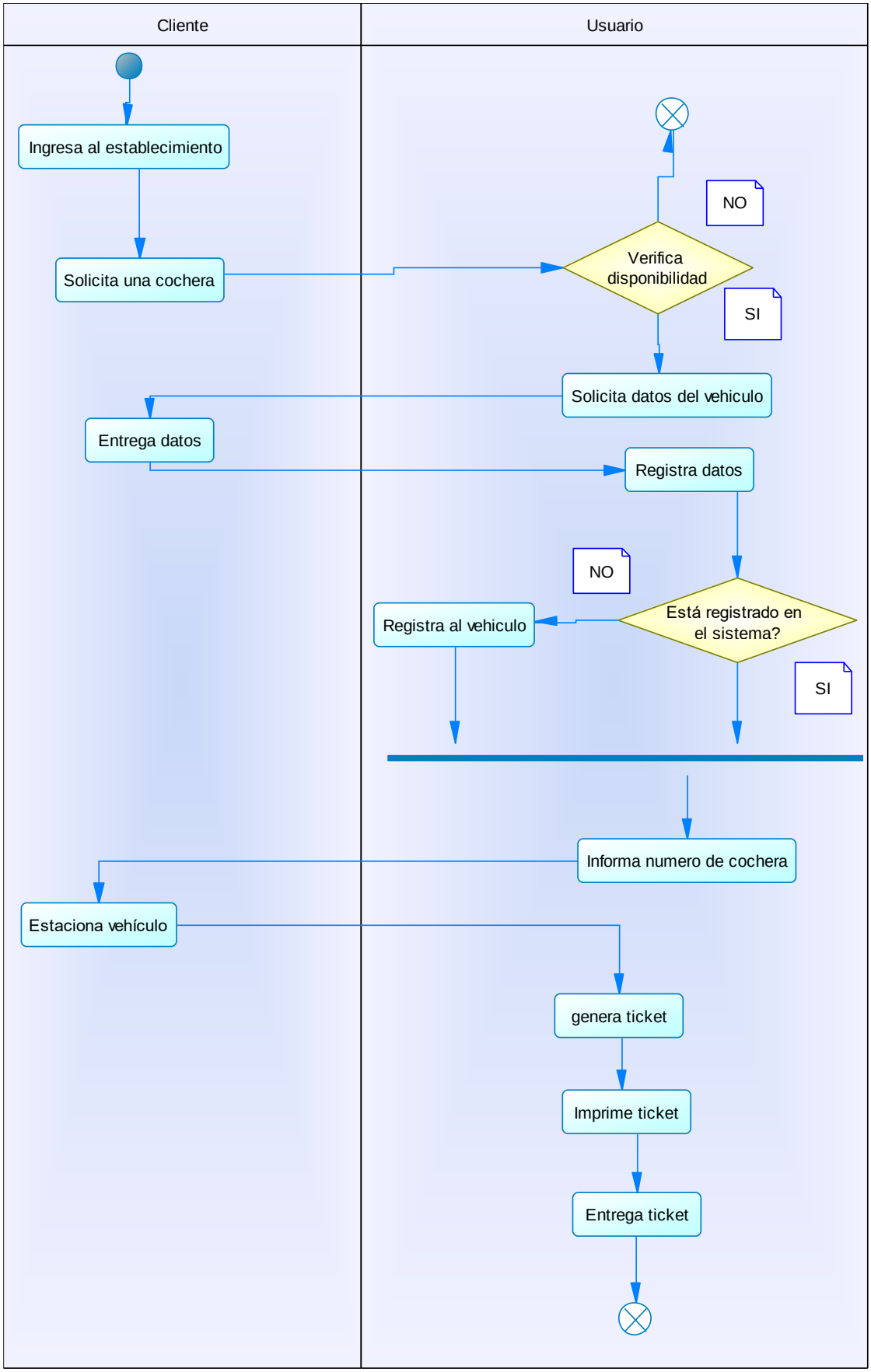


Ilustración 22 Diagrama de Actividades Registro Entrada

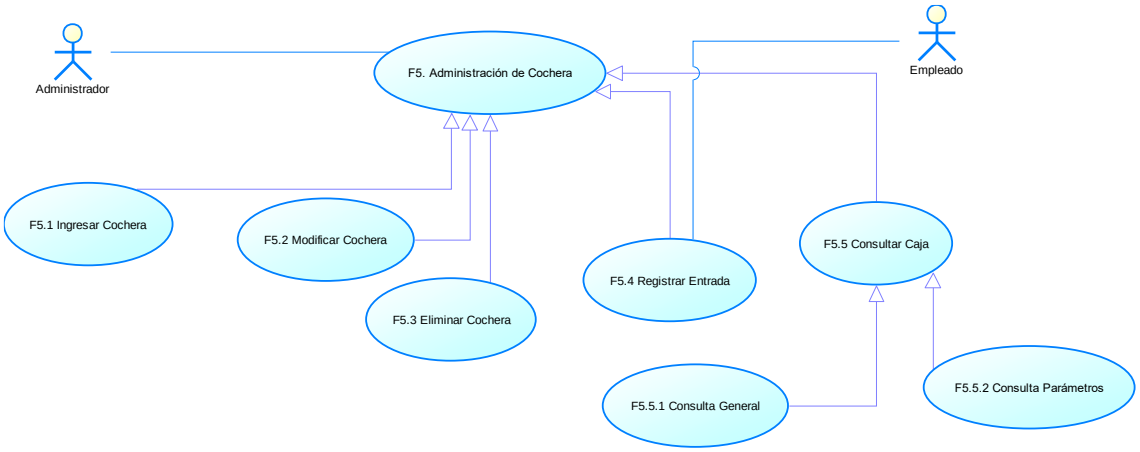


Ilustración 23 Caso de uso a detalle cocheras

F5.1 Ingresar Cochera

Ilustración 24 Ingresar cochera

Descripción: Con esta funcionalidad, el actor podrá agregar una cochera al sistema con todos sus atributos.

Actores: Administrador, Empleado.

Flujo Principal:

1. El sistema despliega el menú principal.
2. El actor selecciona la opción cocheras.
3. El sistema despliega la ventana de cocheras (E1).
4. El actor presiona el botón Crear cochera.
5. El sistema despliega la ventana de creación de cochera.
6. El actor ingresa los campos solicitados.
7. El actor presiona el botón crear.
8. El sistema almacena la información (E1).
9. El sistema crea una nueva cochera.

Flujo Alterno

1. Si ya existe, ver el caso de uso F5.2, F5.3

Excepciones

Código	Causa	Mensaje
E1	Error de conexión con la base de datos	Contactar al soporte.

F5.2 Modificar Cochera

Ilustración 25 Modificar cochera

Descripción: Con esta funcionalidad, el actor podrá modificar una cochera del sistema con todos sus atributos existentes.

Actores: Administrador, Empleado.

Flujo Principal:

1. El sistema despliega el menú principal
2. El actor selecciona la opción cochera s.
3. El sistema despliega la ventana de cocheras (E1).
4. El actor presiona el botón Editar en el registro deseado.
5. El sistema despliega la ventana de edición de cochera.
6. El actor modifica los campos solicitados.
7. El actor presiona el botón guardar.
8. El sistema actualiza la información de la cochera (E1).

Excepciones

Código	Causa	Mensaje
E1	Error de conexión con la base de datos	Contactar al soporte.

F5.3 Eliminar Cochera

Ilustración 26 Eliminar cochera

Descripción: Con esta funcionalidad, el actor podrá eliminar una cochera del sistema con todos sus atributos existentes.

Actores: Administrador.

Flujo Principal:

1. El sistema despliega el menú principal.
2. El actor selecciona la opción cocheras.
3. El sistema despliega la ventana de cocheras (E1).
4. El actor presiona el botón Eliminar en el registro requerido.
5. El sistema despliega la ventana de confirmación.
6. El actor confirma la eliminación del ítem.
7. El sistema elimina el registro (E1).

Excepciones

Código	Causa	Mensaje
E1	Error de conexión con la base de datos	Contactar al soporte.

F5.4 Registro Entrada

Ilustración 27 Registro entrada

Descripción: Con esta funcionalidad, el actor podrá registrar un vehículo en una cochera al sistema con todos sus atributos.

Actores: Administrador, Empleado.

Flujo Principal:

1. El sistema despliega el menú principal.
2. El actor selecciona la opción registrar entrada.
3. El sistema despliega la ventana.
4. El actor presiona el botón Ingresar.
5. El sistema despliega la ventana de vehículos disponibles (E1).
6. El actor selecciona el vehículo deseado.
7. El actor presiona el botón Ingresar.
8. El sistema almacena la información (E1).
9. El sistema registra un nuevo ingreso.
10. El sistema despliega la ventana del ticket (E1).

Flujo Alterno

1. Ver el caso de uso F1.1, F2.1

Excepciones

Código	Causa	Mensaje
E1	Error de conexión con la base de datos	Contactar al soporte.

F5.5.1 Consulta General

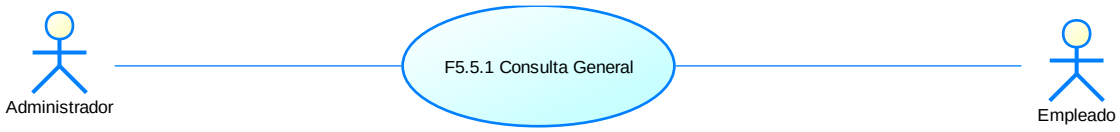


Ilustración 28 Consulta General cocheras

Descripción: Con esta funcionalidad, el actor podrá consultar todas las cocheras del sistema con todos sus atributos existentes.

Actores: Administrador, Empleado.

Flujo Principal:

1. El sistema despliega el menú principal.
2. El actor selecciona la opción cochera.
3. El sistema despliega la información (E1).

Excepciones

Código	Causa	Mensaje
E1	Error de conexión con la base de datos	Contactar al soporte.

F5.5.2 Consulta Parámetros

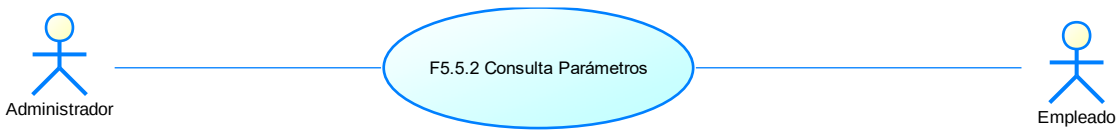


Ilustración 29 Consulta Parámetros cochera

Descripción: Con esta funcionalidad, el actor podrá consultar cocheras bajo parámetros con todos sus atributos existentes.

Actores: Administrador, Empleado.

Flujo Principal:

1. El sistema despliega el menú principal.
2. El actor selecciona la opción cocheras.
3. El sistema despliega la ventana (E1).
4. El actor escribe la palabra de búsqueda.
5. El sistema despliega la información (E1).

Excepciones

Código	Causa	Mensaje
E1	Error de conexión con la base de datos	Contactar al soporte.

F8. Administración de Asignaciones

El proceso de Administración de Asignaciones cubre el siguiente diagrama de actividades y se complementa con su respectivo caso de uso.

Diagrama de Actividades de Salida de vehículo

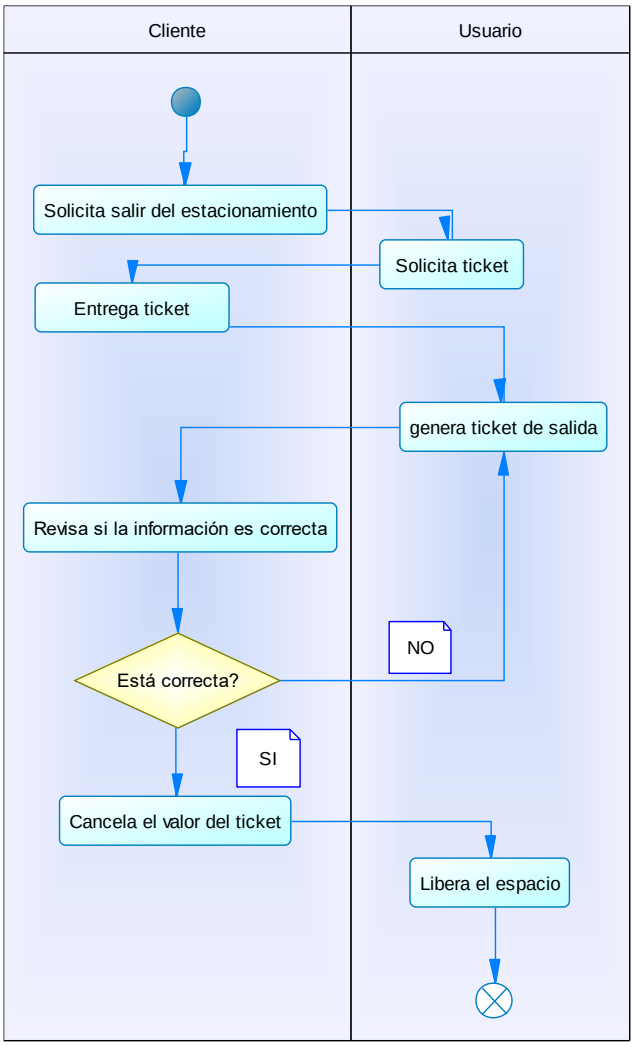


Ilustración 30 Diagrama de Actividades Salida Vehículo

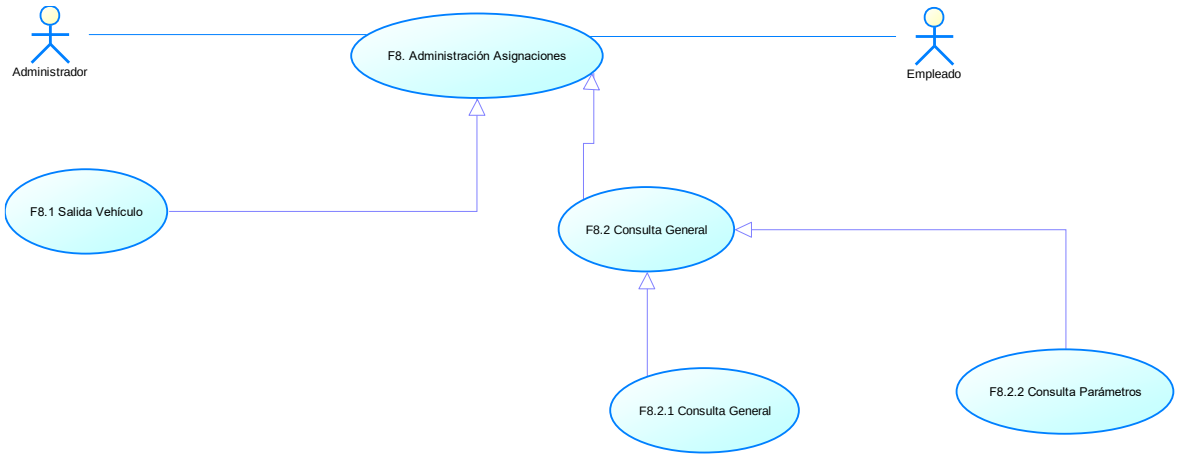


Ilustración 31 Caso de uso a detalle Asignaciones

F8.1 Salida Vehículo

Ilustración 32 Salida Vehículo

Descripción: Con esta funcionalidad, el actor podrá retirar un vehículo de una cochera

Actores: Administrador, Empleado.

Flujo Principal:

1. El sistema despliega el menú principal
2. El actor selecciona la opción cocheras.
3. El sistema despliega la ventana de cocheras (E1).
4. El actor presiona el botón Salida en el registro deseado.
5. El sistema despliega ventana de confirmación.
6. El actor presiona el botón Aceptar.
7. El sistema registra la salida del parqueadero (E1).
8. El sistema libera el espacio de la cochera (E1).
9. El sistema despliega el ticket de salida.

Excepciones

Código	Causa	Mensaje
E1	Error de conexión con la base de datos	Contactar al soporte.

F8.2.1 Consulta General

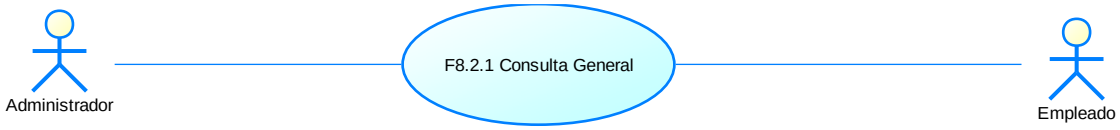


Ilustración 33 Consulta General Asignaciones

Descripción: Con esta funcionalidad, el actor podrá consultar todos los vehículos del sistema.

Actores: Administrador, Empleado.

Flujo Principal:

1. El sistema despliega el menú principal.
2. El actor selecciona la opción reportes.
3. El actor selecciona la opción todas las fechas.
4. El sistema despliega la información (E1).

Excepciones

Código	Causa	Mensaje
E1	Error de conexión con la base de datos	Contactar al soporte.

F8.2.2 Consulta por Parámetros

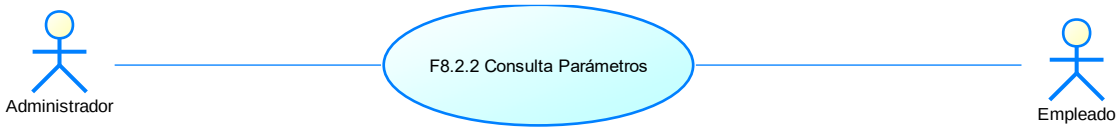


Ilustración 34 Consulta Parámetros Asignaciones

Descripción: Con esta funcionalidad, el actor podrá consultar cocheras bajo parámetros con todos sus atributos existentes.

Actores: Administrador, Empleado.

Flujo Principal:

1. El sistema despliega el menú principal.
2. El actor selecciona la opción reportes.
3. El actor selecciona la opción entre fechas.
4. El sistema despliega la ventana (E1).
5. El actor escribe el rango de fechas.

6. El sistema despliega la información (E1).

Excepciones

Código	Causa	Mensaje
E1	Error de conexión con la base de datos	Contactar al soporte.

3.2 Diseño del Sistema

3.2.1 Modelos de base de datos

Un modelo de base de datos es el que permite diseñar, visualizar las tablas que conforma una base de datos y a que están conectados (relaciones), con el fin de modelar el problema a resolver y entenderlo mejor, a manera de entidades y relaciones (Rojas, 2009).

Entre los diagramas de base de datos tenemos: conceptual, físico y lógico.

3.2.1.1 Diagrama Conceptual

Es una vista de conceptos de la base de datos y sus relaciones, el propósito de crearlo es establecer entidades, atributos y sus relaciones, no hay mucho detalle sobre las entidades, sin embargo, podemos observar de manera global el sistema (Sánchez, 2004) .

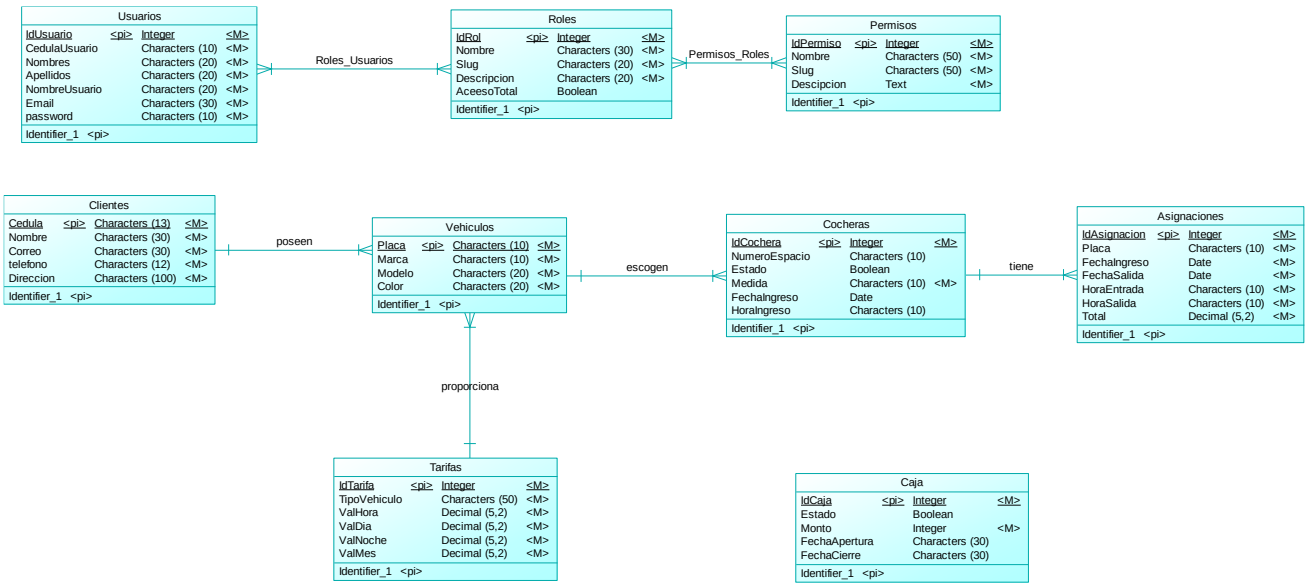


Ilustración 35 Diagrama Conceptual

3.2.1.2 Diagrama Lógico

Se utiliza para definir los atributos de las entidades, y sus relaciones entre sí. La ventaja de usar este tipo de modelo es que podemos construir el modelo físico con una buena base (Sánchez, 2004).

Este tipo de modelado no define una clave primaria o secundaria, como se presenta a continuación:

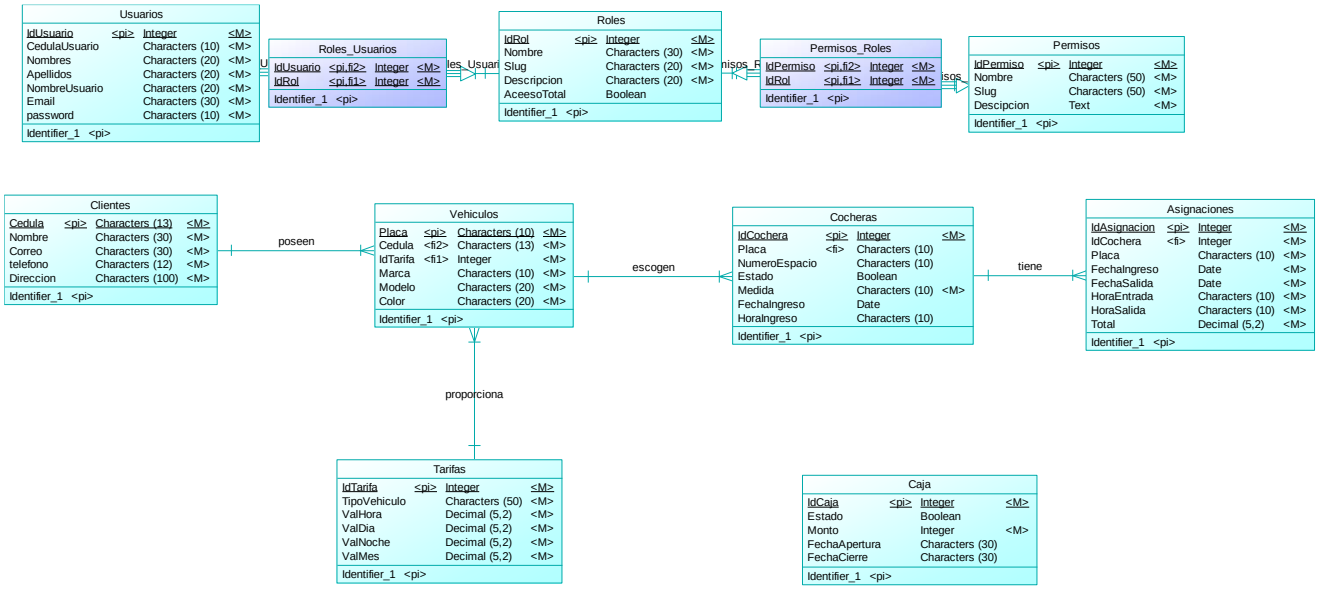


Ilustración 36 Diagrama Lógico

3.2.1.3 Diagrama Físico

Un modelo físico describe una implementación específica de la base de datos, a partir del modelo lógico, gracias a sus metadatos que posee, además que ayuda a visualizar la estructura (índices, disparadores, restricciones, entre otras) (Sánchez, 2004).

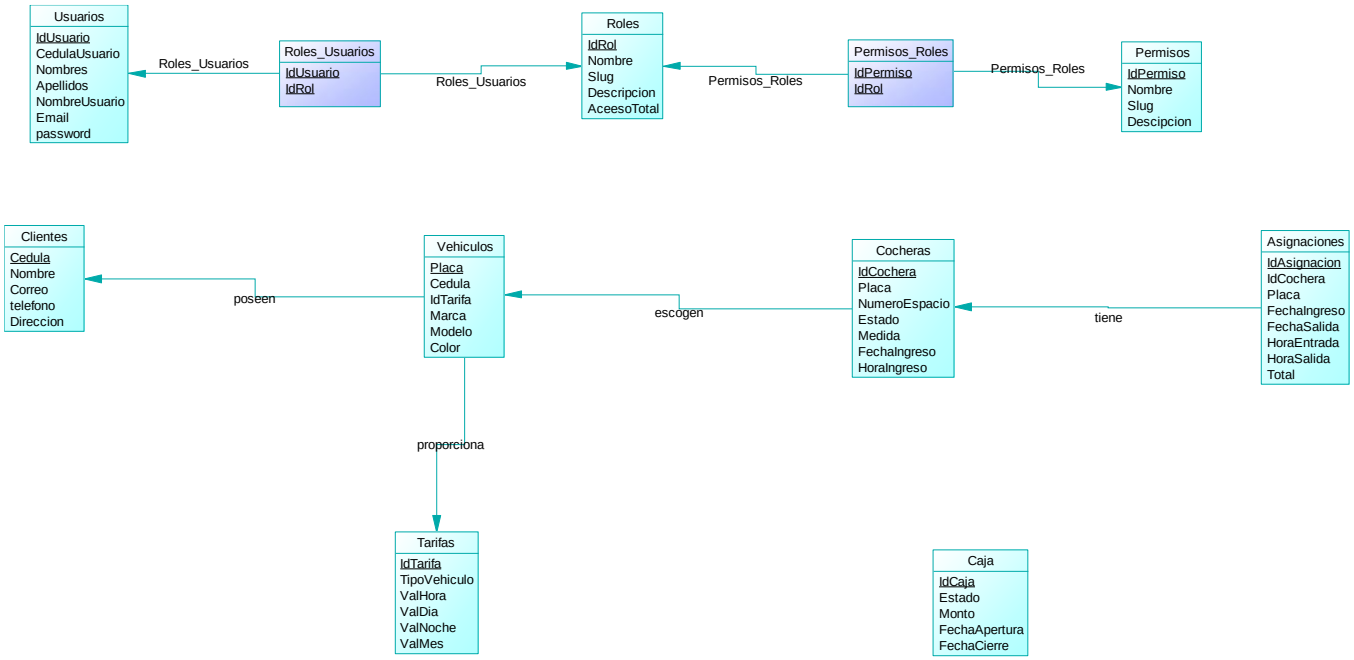


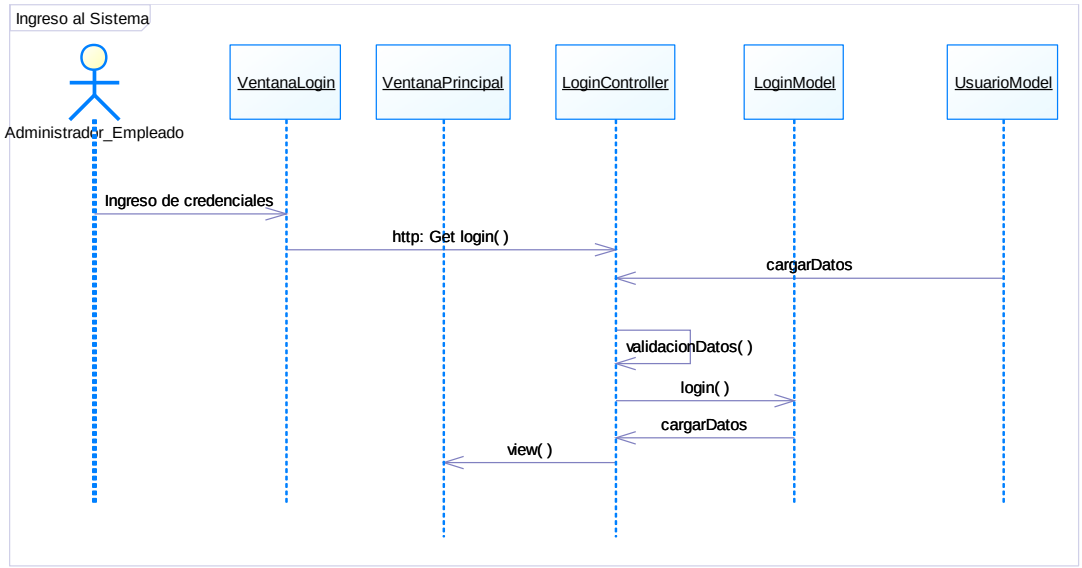
Ilustración 37 Diagrama Físico

3.2.2 Diagramas de Secuencia

Este tipo de diagrama define la interacción del usuario con el sistema para mostrar su comportamiento interactivo, es decir describe el cómo y en qué orden el grupo de objetos trabaja en conjunto de manera secuencial (Gutierrez, Diagramas de Secuencia, 2011). Entre sus elementos encontramos: el actor, líneas de vida, mensajes y objetos.

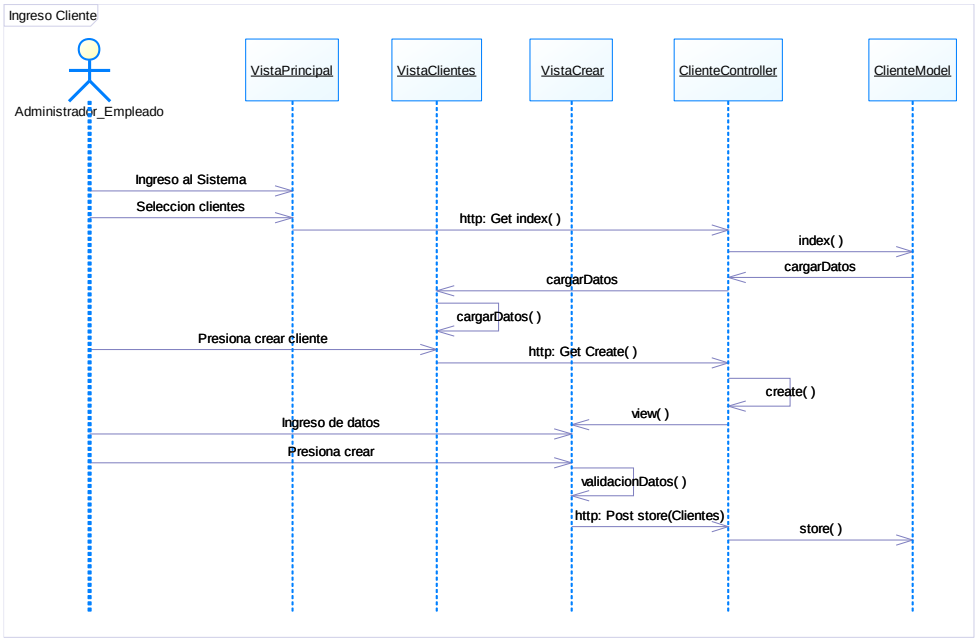
Al igual que en los casos de uso, se presenta los diagramas de F1 Administración de Clientes, el resto de los diagramas para F2, F3, F6, F7 y F8 están en el Anexo 2.

F0 Ingreso al Sistema

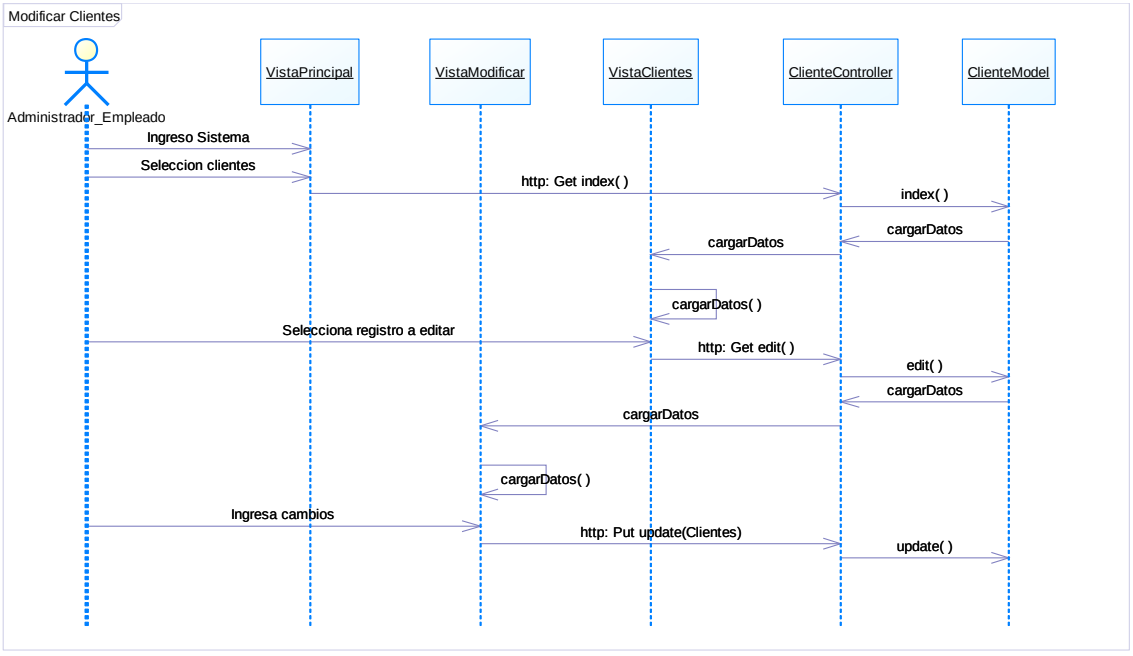


F1 Administración Clientes

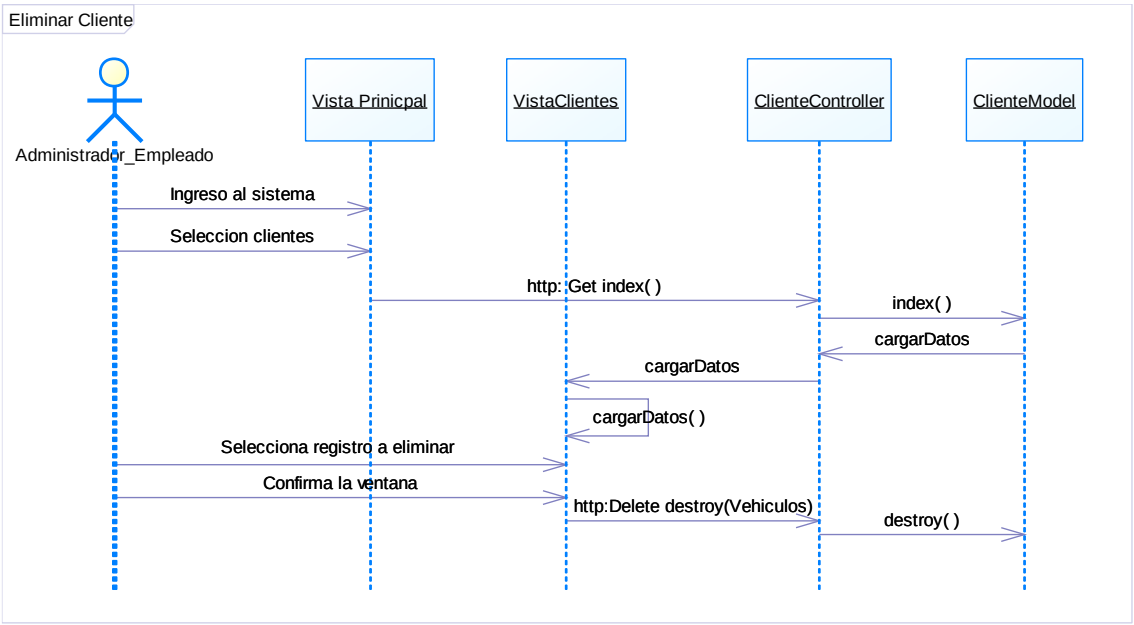
F1.1 Ingreso Cliente



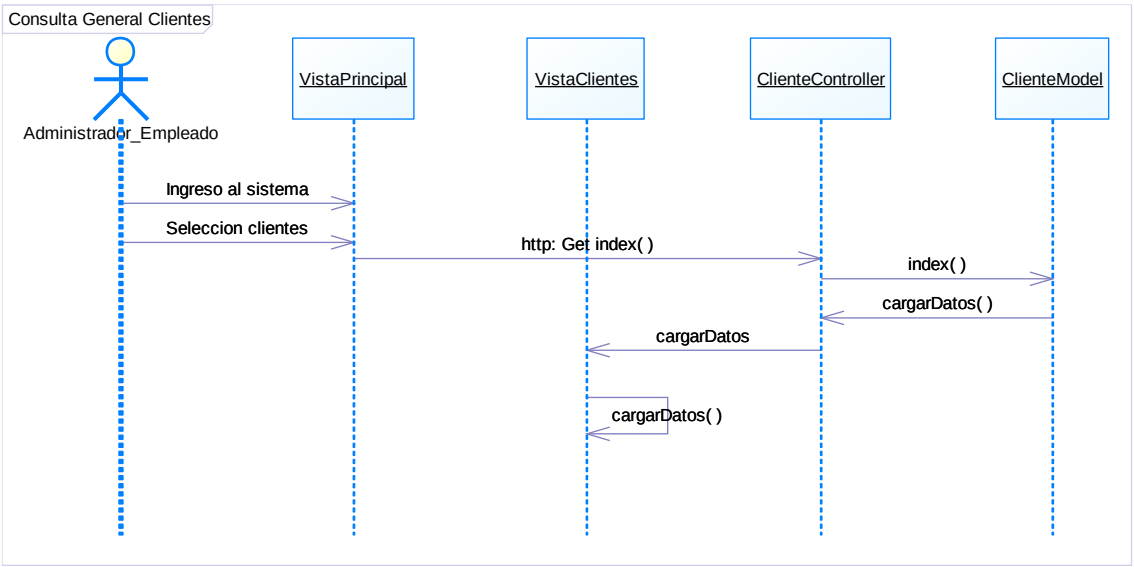
F1.2 Modificar Cliente



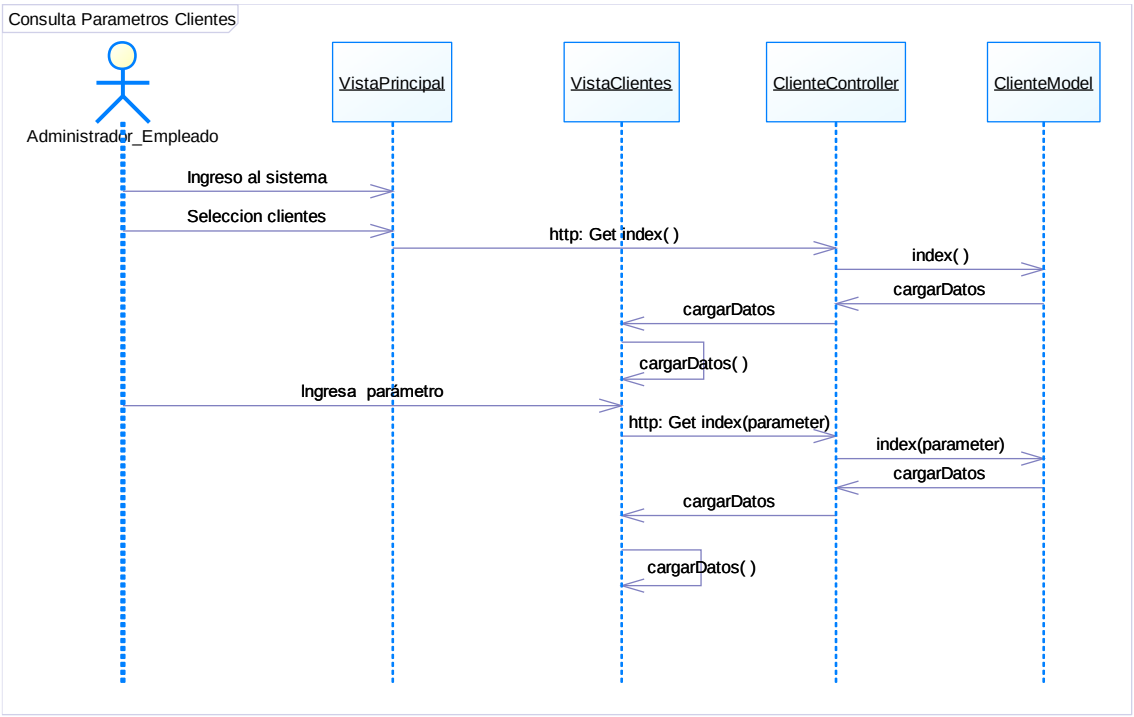
F1.3 Eliminar Cliente



F1.4.1 Consulta General

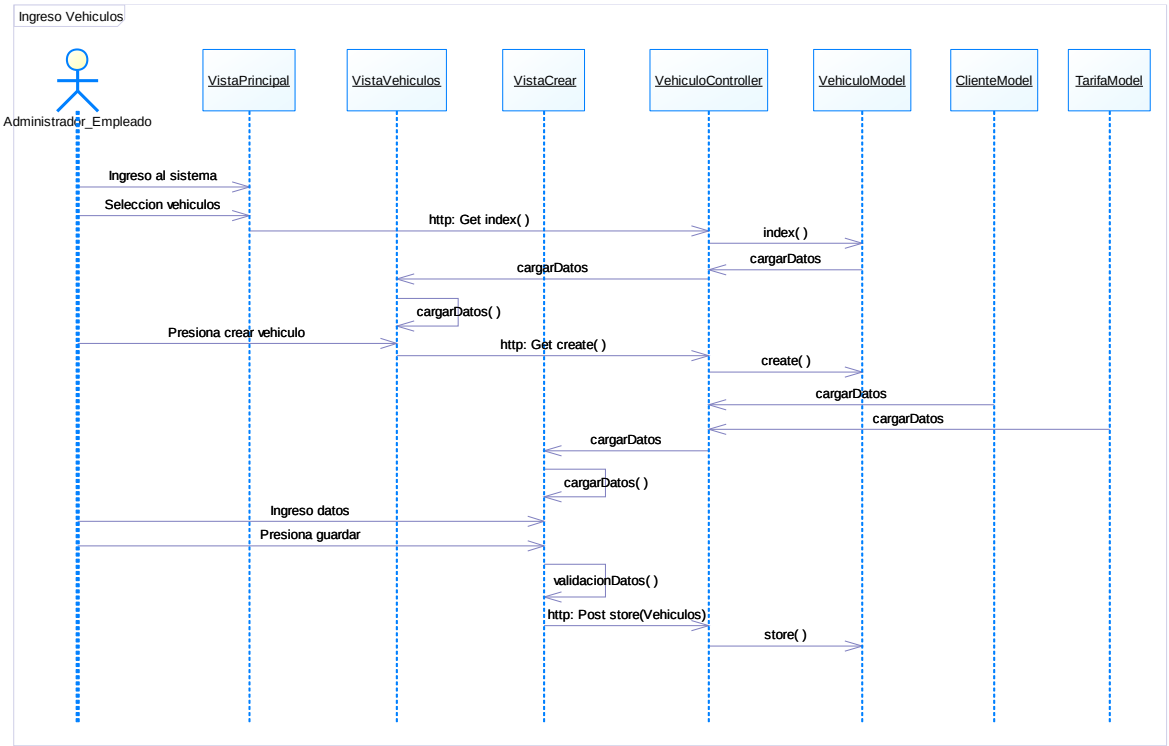


F1.4.2 Consulta Parámetros

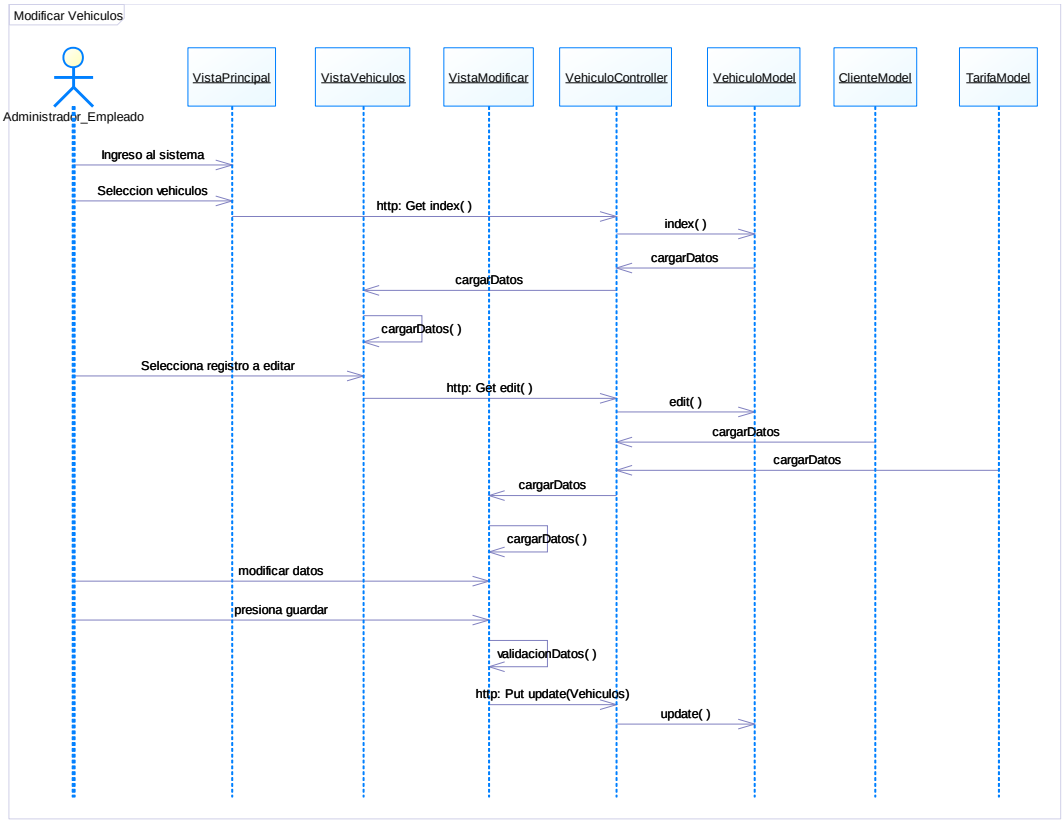


F4 Administración de Vehículos

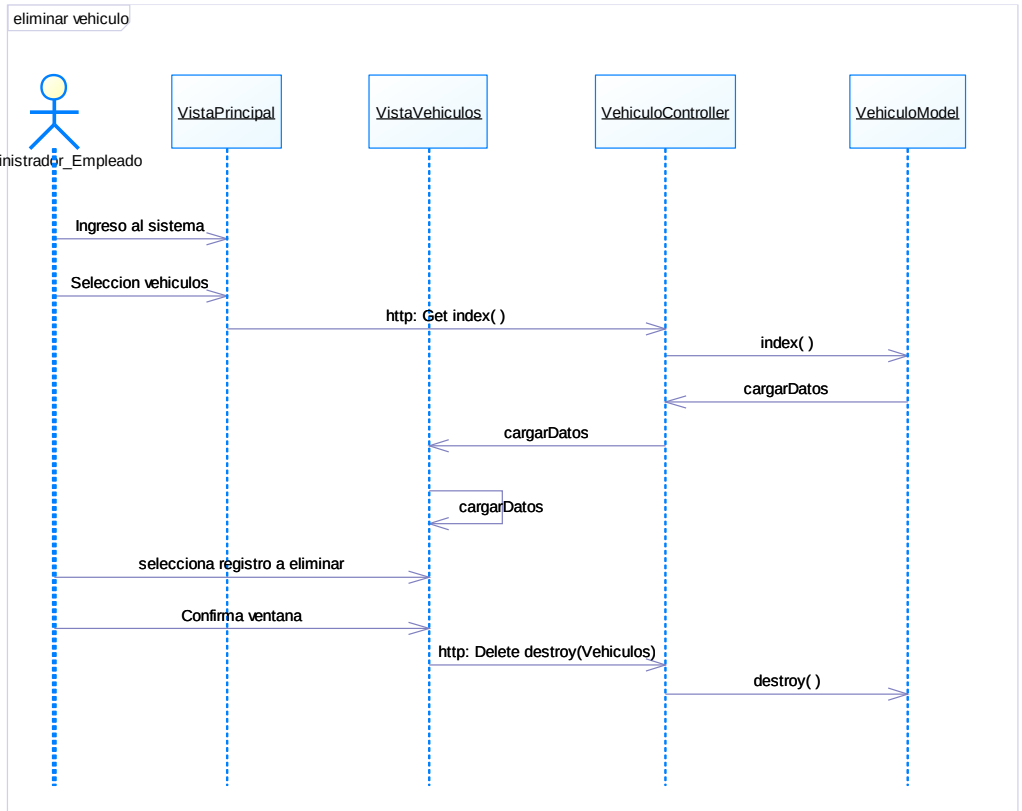
F4.1 Ingreso Vehículo



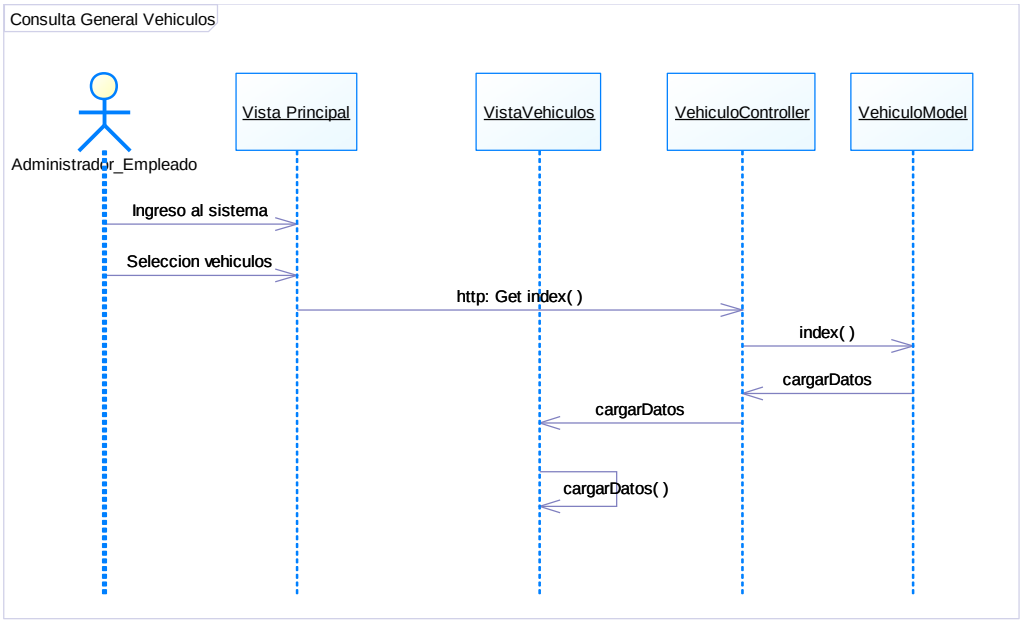
F4.2 Modificar Vehículo



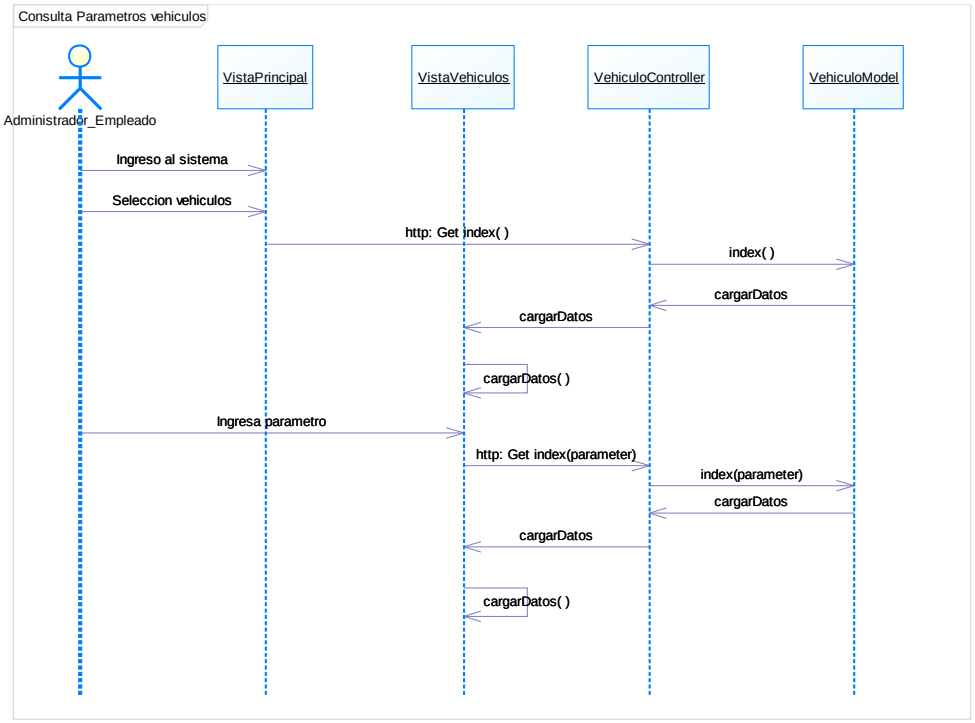
F4.3 Eliminar Vehículo



F4.4.1 Consulta General

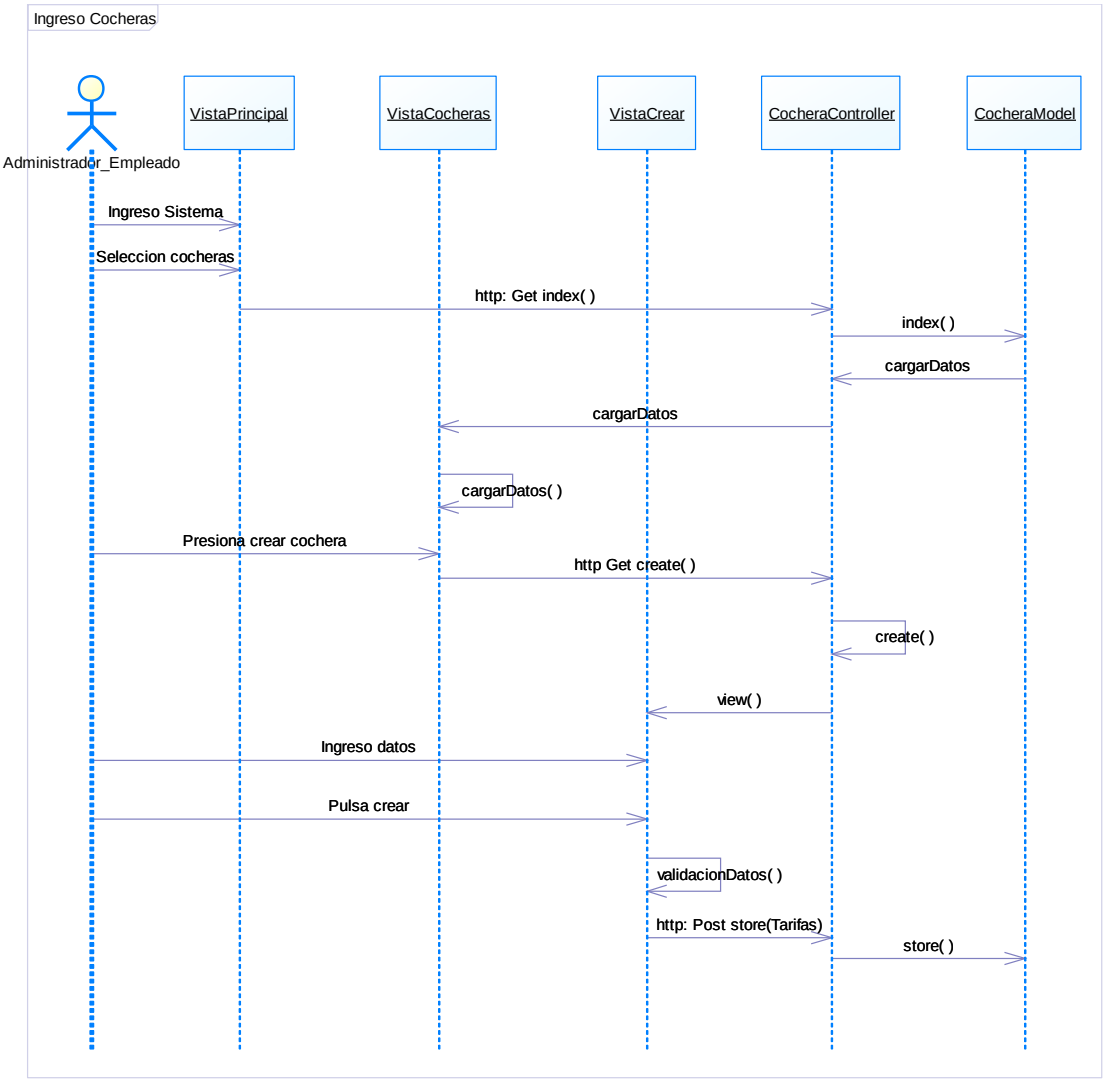


F4.4.2 Consulta Parámetros

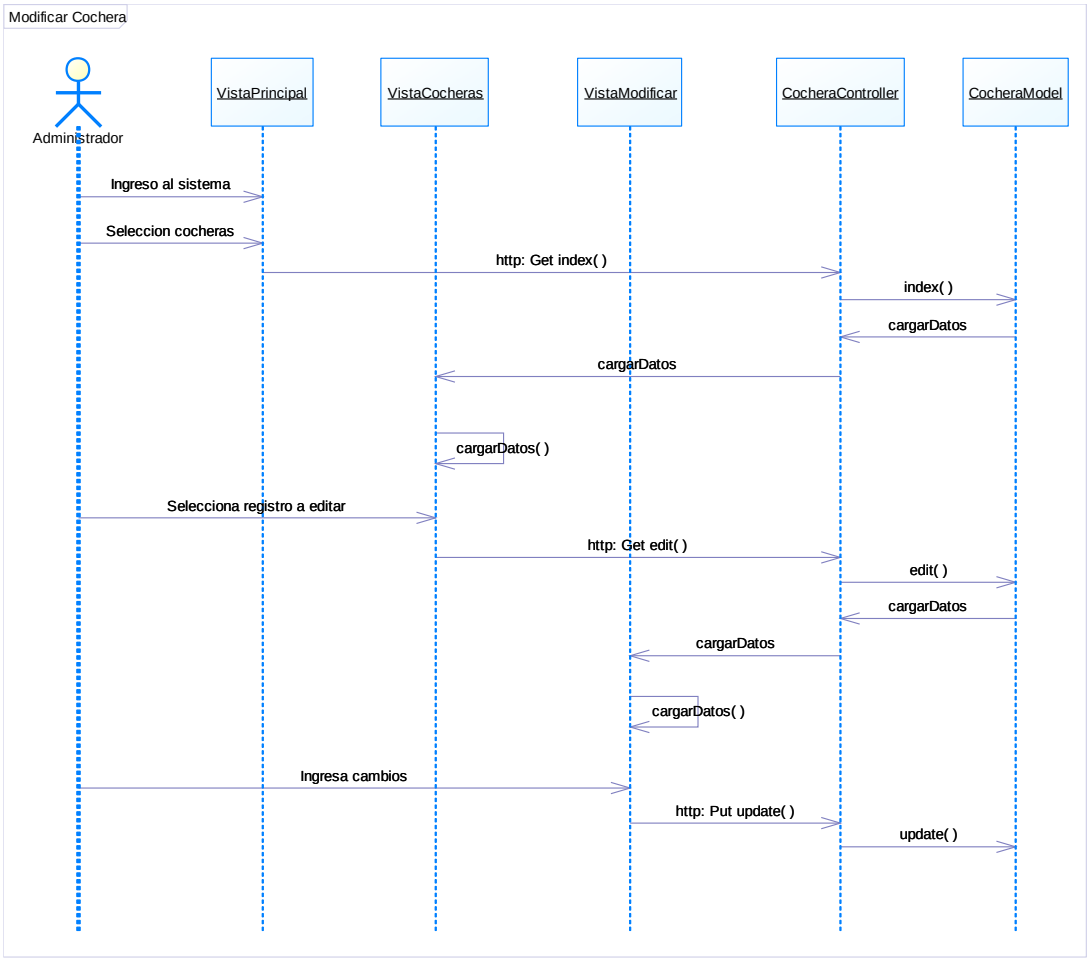


F5 Administración de Cocheras

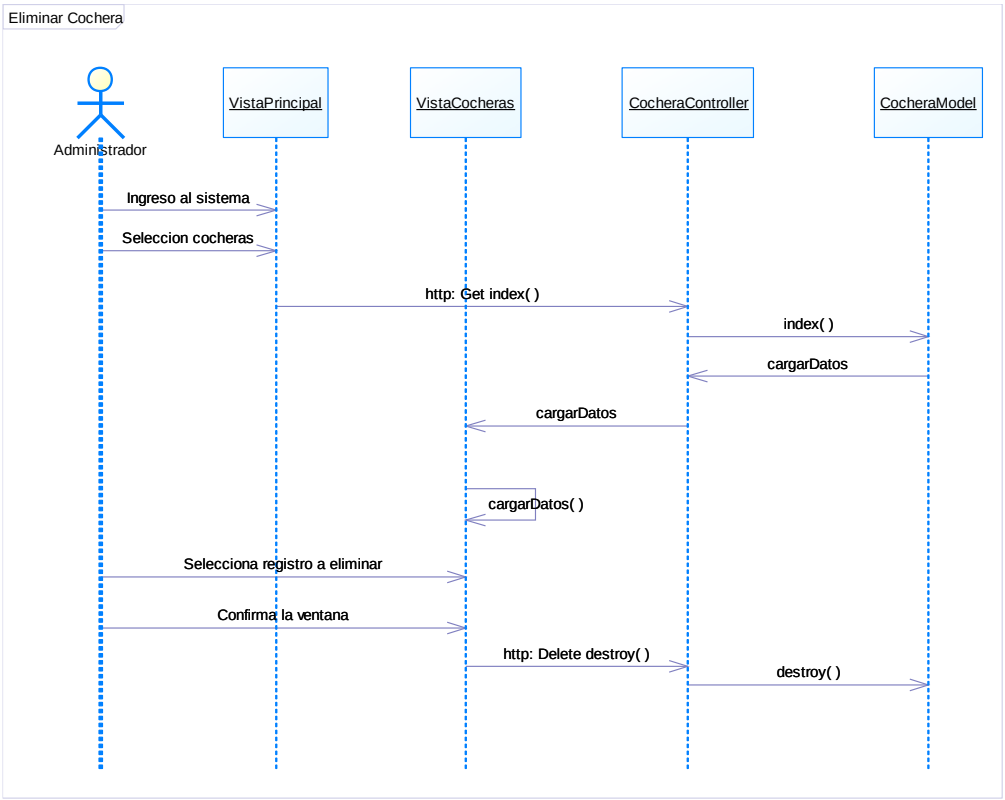
F5.1 Ingreso Cochera



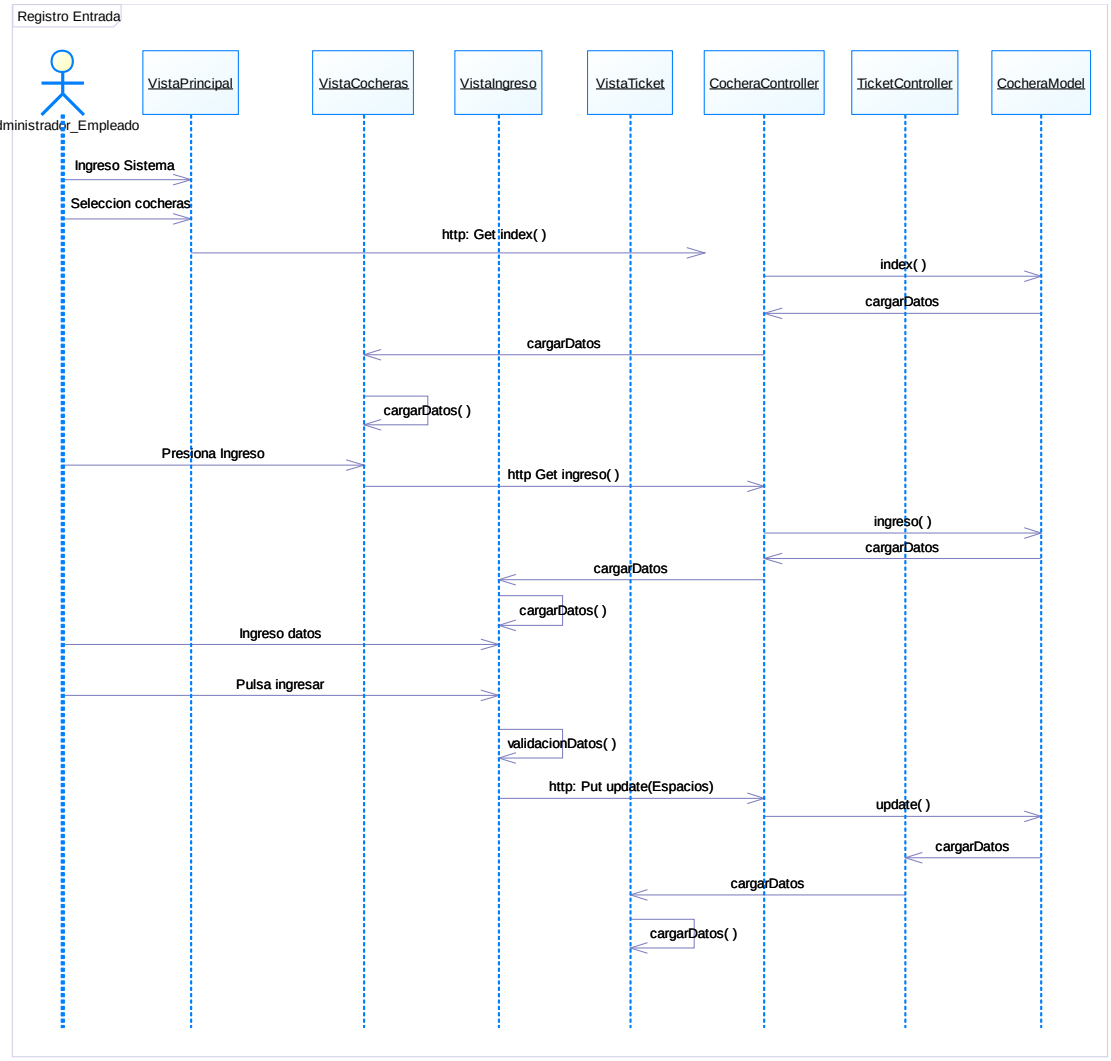
F5.2 Modificar Cochera



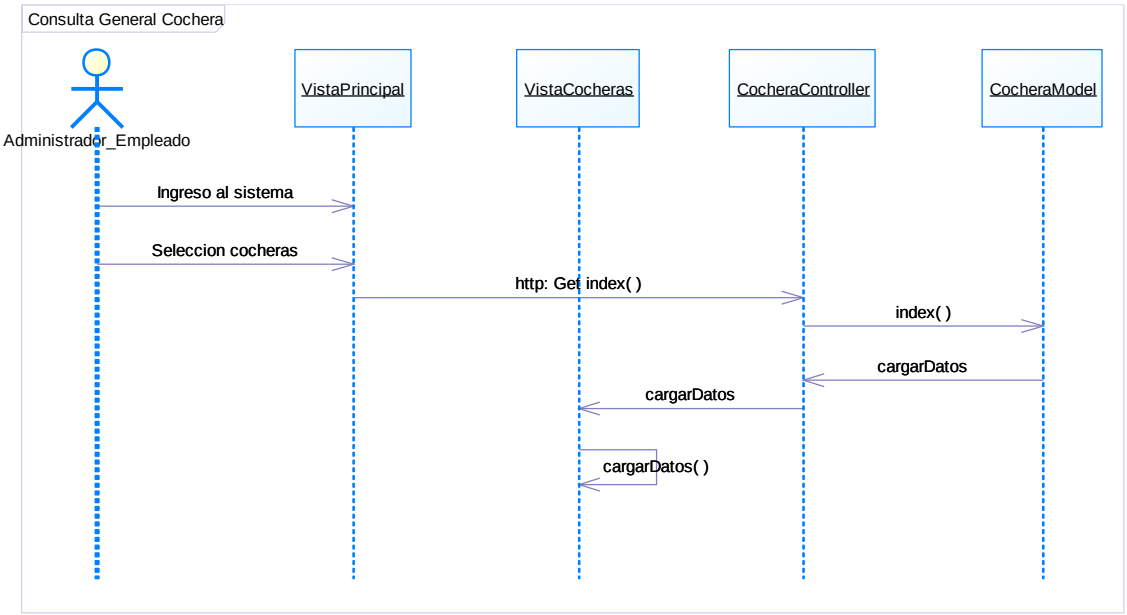
F5.3 Eliminar Cochera



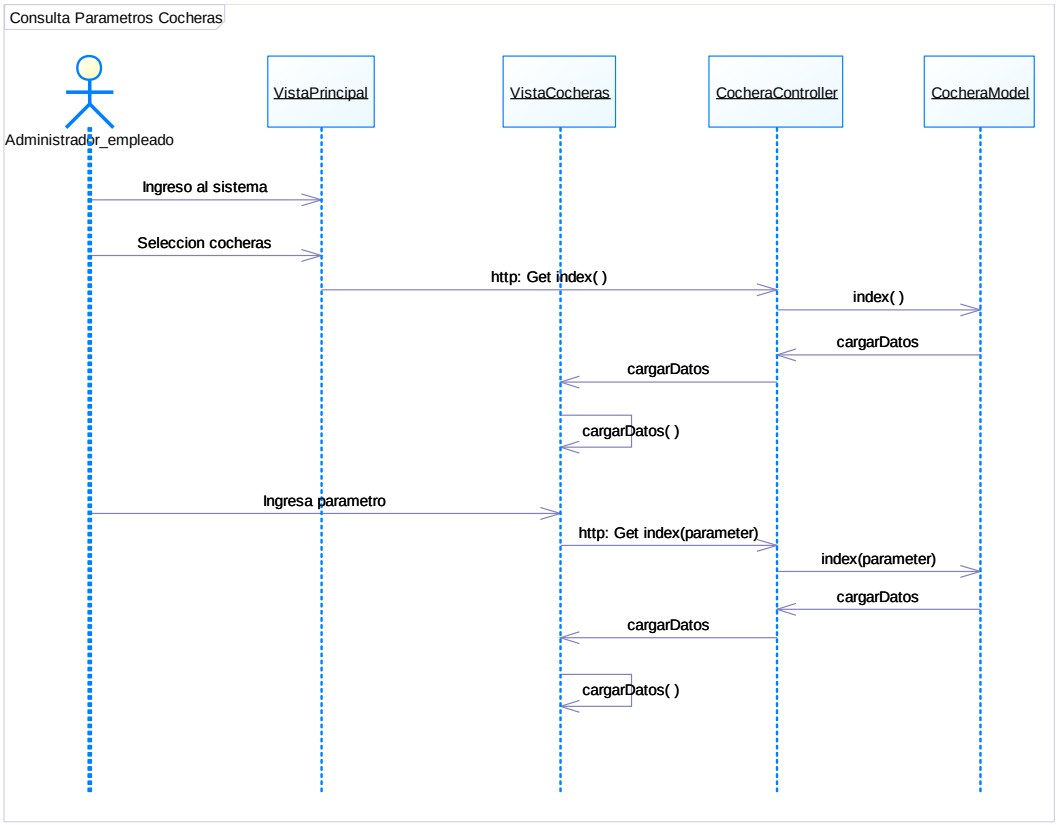
F5.4 Registro Entrada



F5.5.1 Consulta General

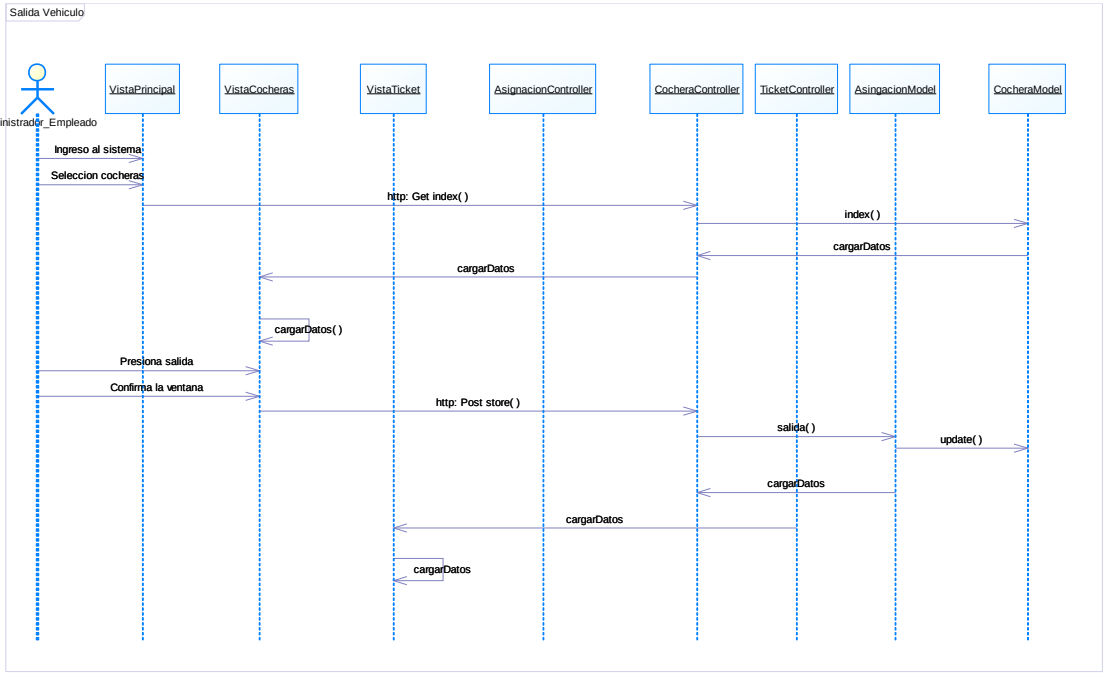


F5.5.2 Consulta Parámetros

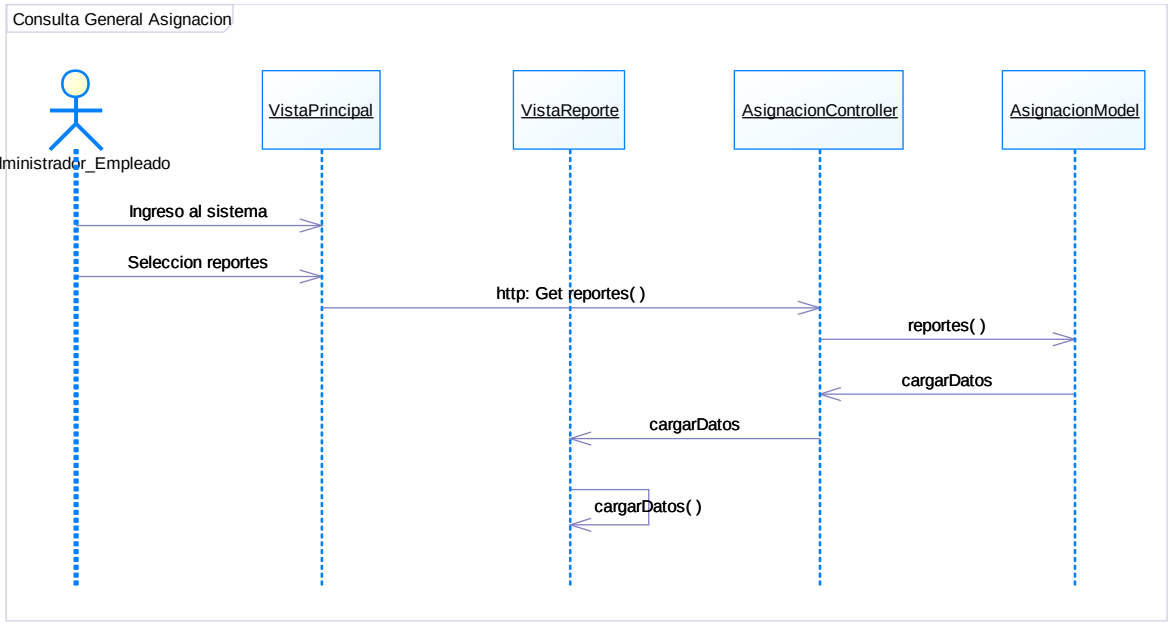


F8 Administración de Asignación

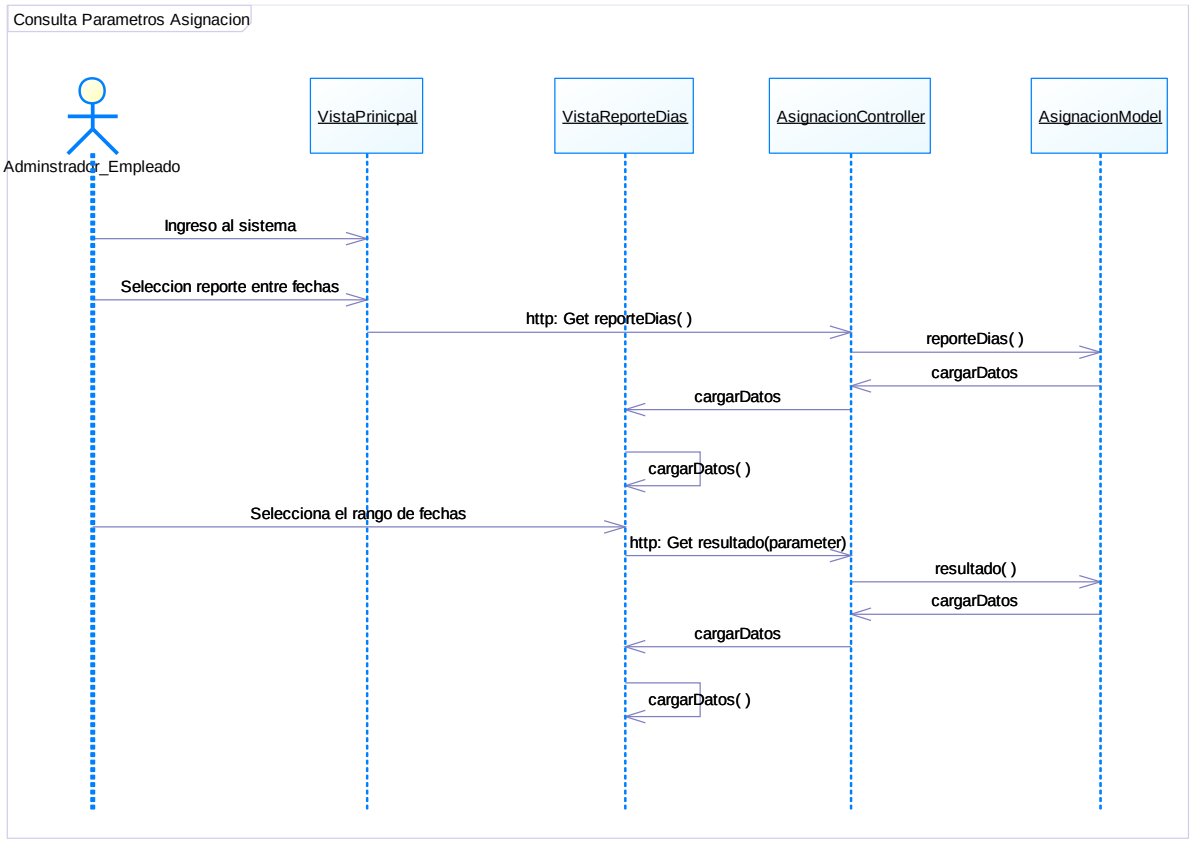
F8.1 Salida Vehículo



F8.2.1 Consulta General



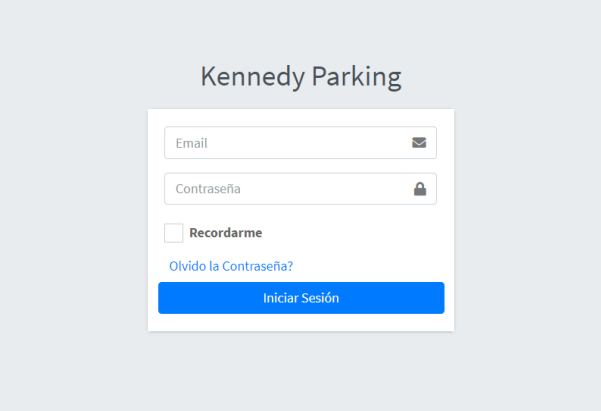
F8.2.2 Consulta Parámetros



3.3 Implementación

A continuación, se presentará las pantallas del sistema con scripts de código más representativo, indicando a que se refiere cada una. En el Anexo 3, están las pantallas para el manejo del resto de opciones del sistema y el código fuente de todas las funcionalidades estará en el CD que se anexa a esta disertación.

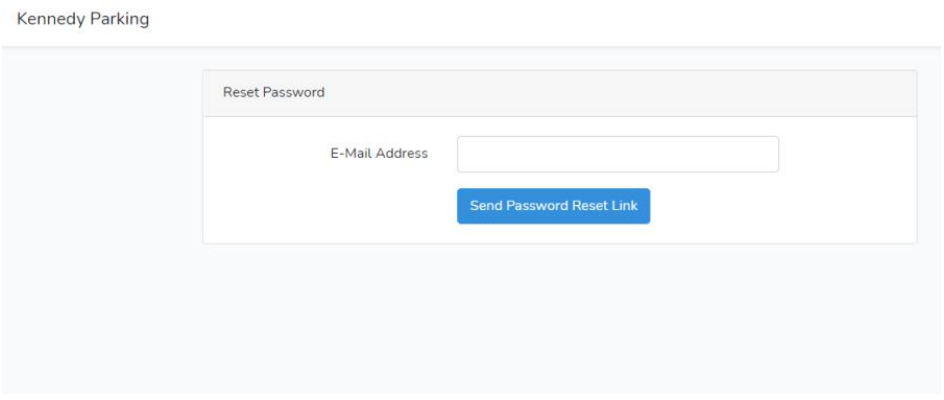
F0 Ingreso al Sistema




```php
public function login(Request $request)
{
    $this->validateLogin($request);


    if (method_exists($this, 'hasTooManyLoginAttempts') &&
        $this->hasTooManyLoginAttempts($request)) {
        $this->fireLockoutEvent($request);

        return $this->sendLockoutResponse($request);
    }

    if ($this->attemptLogin($request)) {
        return $this->sendLoginResponse($request);
    }

    $this->incrementLoginAttempts($request);

    return $this->sendFailedLoginResponse($request);
}
```

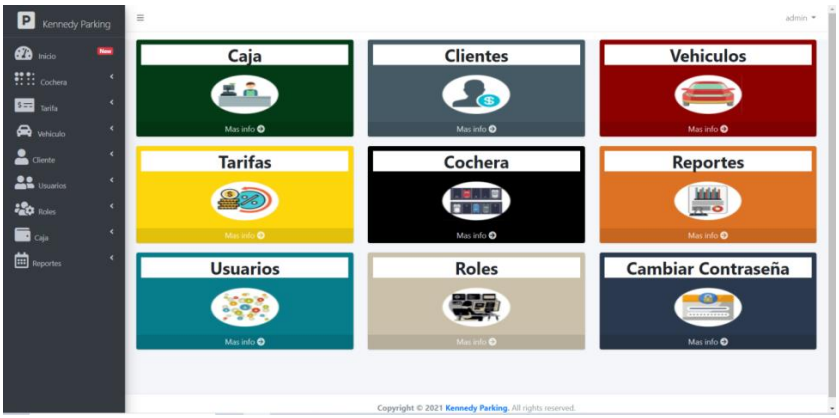
En esta funcionalidad lo más destacado es la conexión a la base de datos y se complementa con la siguiente pantalla:

Ventana de recuperación contraseña



```php
public function reset(Request $request)
{
    $request->validate($this->rules(), $this->validationErrorMessages());


    $response = $this->broker()->reset(
        $this->credentials($request), function ($user, $password) {
            $this->resetPassword($user, $password);
        }
    );


    return $response == Password::PASSWORD_RESET
                ? $this->sendResetResponse($request, $response)
                : $this->sendResetFailedResponse($request, $response);
}
```

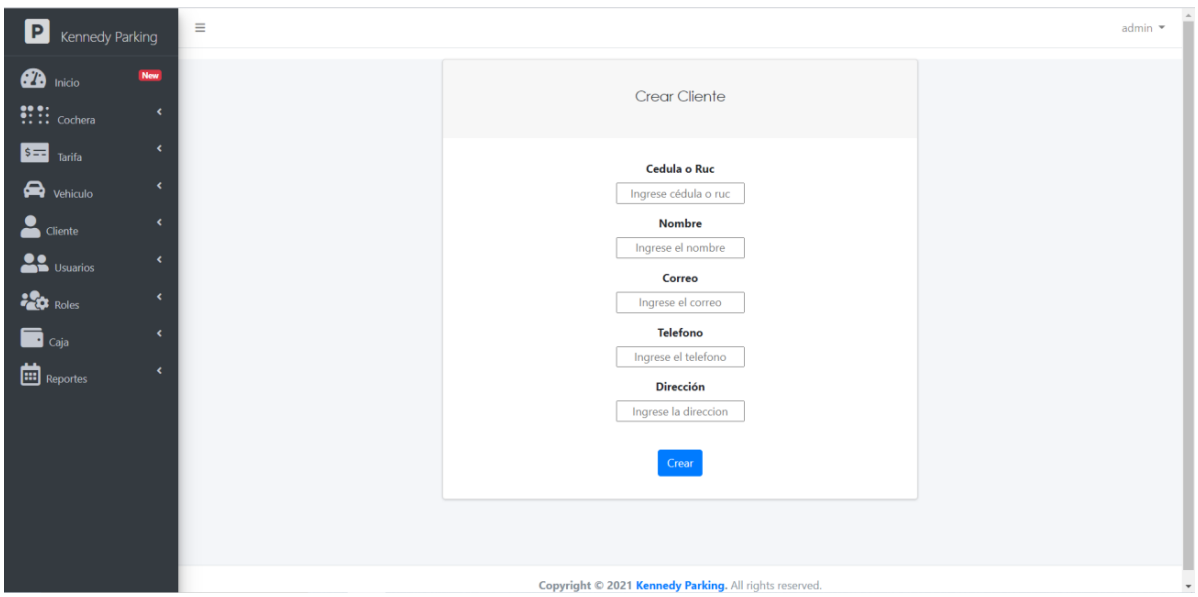
Pantalla Principal



Al igual que en los casos de uso, se presenta las ventanas de F1 Administrar Clientes, el resto de los diagramas para F2, F3, F5, F6, F7, están en el Anexo 2

F1. Administración de Clientes

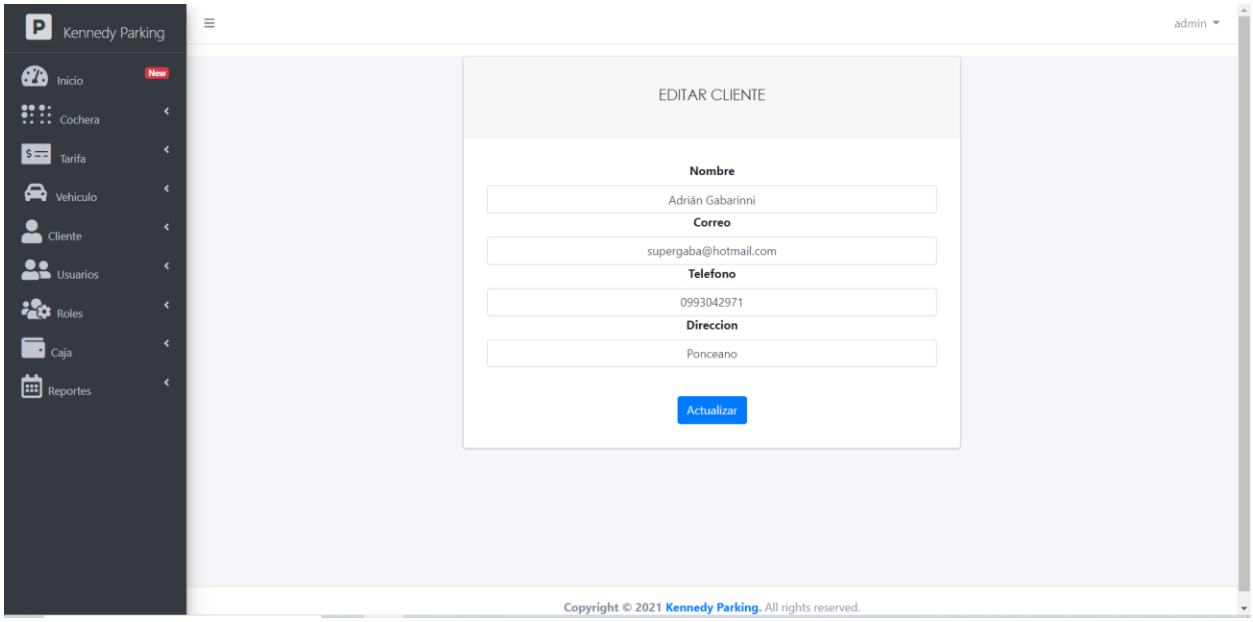
F1.1 Ingresar Cliente



```php
public function create()
{
    Gate::authorize('haveaccess','clientes.create');
    return view('clientes.create');
}

/**
 * Store a newly created resource in storage.
 *
 * @param  \Illuminate\Http\Request  $request
 * @return \Illuminate\Http\Response
 */
public function store(Request $request)
{
    Gate::authorize('haveaccess','clientes.create');
    $data = new \App\Cliente;
    DB::table('clientes')->insert(
        ['cedula' => $request['cedula'],
         'nombre' => $request['nombre'],
         'correo' => $request['correo'],
         'telefono' => $request['telefono'],
         'direccion' => $request['direccion']
         ]
    );
    return redirect('clientes');
}
```

En esta funcionalidad, lo más destacado es el ingreso de un nuevo cliente en el sistema con todos sus atributos

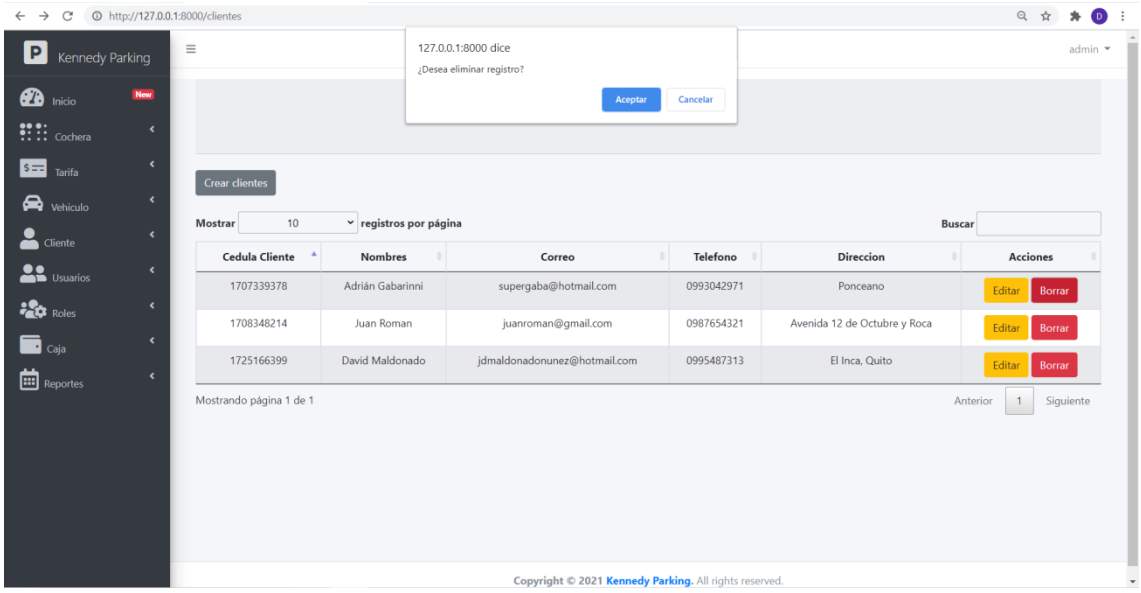
F1.2 Modificar Cliente



```php
public function edit($id)
{
    Gate::authorize('haveaccess','clientes.edit');
    $cliente = Cliente::findOrFail($id);
    return view('clientes.edit',compact('cliente'));
}

/**
 * Update the specified resource in storage.
 *
 * @param  \Illuminate\Http\Request  $request
 * @param  \App\Cliente  $cliente
 * @return \Illuminate\Http\Response
 */
public function update(Request $request, $id)
{
    Gate::authorize('haveaccess','clientes.edit');
    Cliente::where('cedula','=',$id)->update(
        ['cedula' => $id,
        'nombre' => $request['nombre'],
        'correo' => $request['correo'],
        'telefono' => $request['telefono'],
        'direccion' => $request['direccion']
        ]);
        return redirect('clientes');
}
```

En esta funcionalidad, lo más destacado la modificación de los atributos un cliente en el sistema.

F1.3 Eliminar Cliente

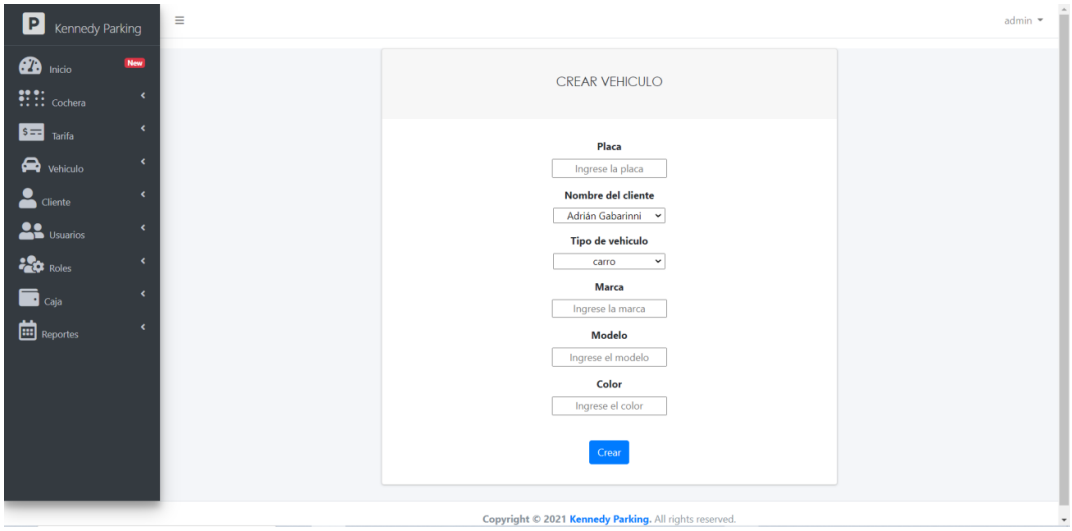



```php
public function destroy($id)
{
    Gate::authorize('haveaccess','clientes.destroy');
    DB::delete('DELETE FROM clientes WHERE CEDULA = ?', [$id]);
    return redirect('clientes');
}
```

En esta funcionalidad, lo más destacado la eliminación de un cliente en el sistema.

F4 Administración de Vehículos

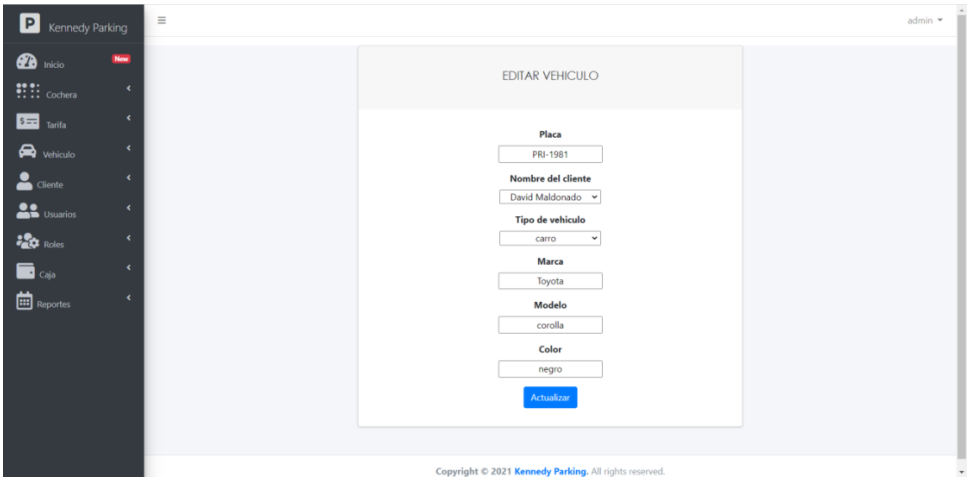
F4.1 Ingresar Vehículo




```php
public function create()
{
    Gate::authorize('haveaccess','vehiculos.create');
    $cliente = Cliente::all();
    $tarifa = Tarifa::all();
    return view('vehiculos.create', ['cliente' => $cliente],
     ['tarifa' => $tarifa] );
}

public function store(Request $request)
{
    Gate::authorize('haveaccess','vehiculos.create');
    $data = new \App\Vehiculo;
    Cliente::findOrFail($request['cliente_cedula']);
    Tarifa::findOrFail($request['tarifa_id']);
    DB::table('vehiculos')->insert(
        ['placa' => $request['placa'],
         'cedula' => $request['cliente_cedula'],
         'idtarifa' => $request['tarifa_id'],
         'marca' => $request['marca'],
         'modelo' => $request['modelo'],
         'color' => $request['color']
         ]
    );
    return redirect('vehiculos');
```

En esta funcionalidad, lo más destacado es el ingreso de un nuevo vehículo en el sistema con todos sus atributos

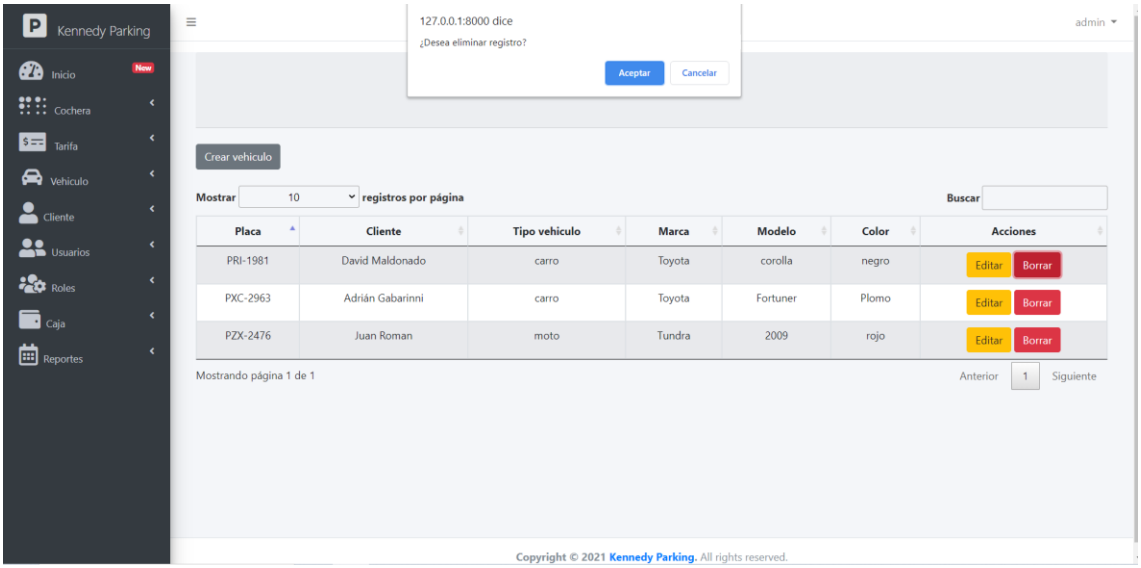
F4.2 Modificar Vehículo



```php
public function edit($id)
{
    Gate::authorize('haveaccess','vehiculos.edit');
    $vehiculo = Vehiculo::findOrFail($id);
    $cliente = Cliente::all();
    $tarifa = Tarifa::all();
    return view('vehiculos.edit',['cliente' => $cliente, 'tarifa' => $tarifa],
    compact('vehiculo'));

}


public function update(Request $request, $id)
{
    Gate::authorize('haveaccess','vehiculos.edit');
    Vehiculo::where('PLACA','=',$id)->update(
        ['PLACA' => $id,
        'cedula' => $request['cedula'],
        'idtarifa' => $request['idtarifa'],
        'marca' => $request['marca'],
        'modelo' => $request['modelo'],
        'color' => $request['color']
        ]);
        return redirect('vehiculos');
}
```

En esta funcionalidad, lo más destacado la modificación de los atributos de un cliente en el sistema.

F4.3 Eliminar Vehículo

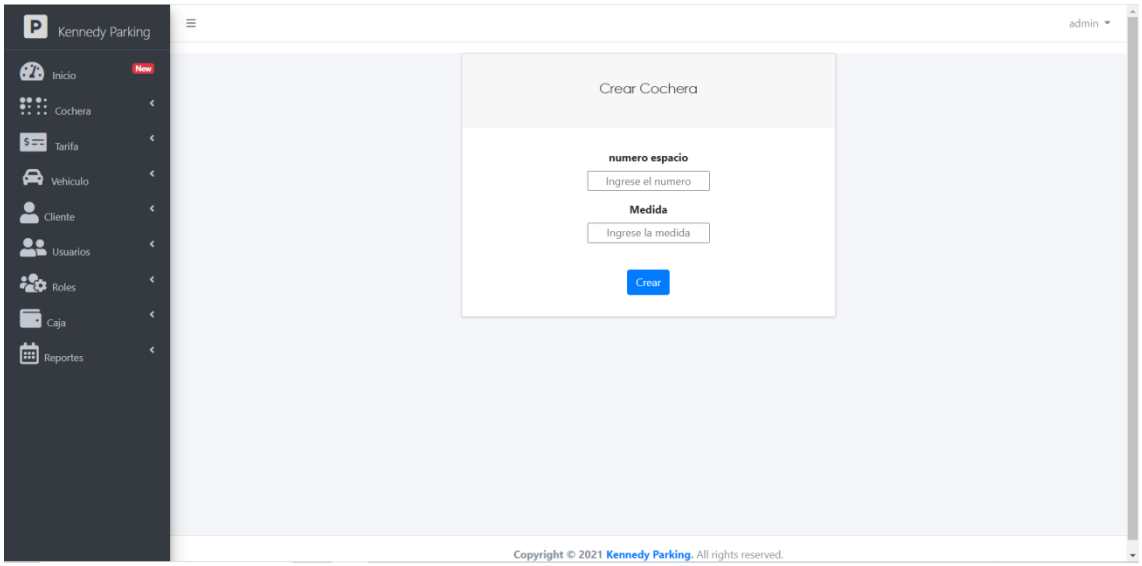


En esta funcionalidad, lo más destacado la eliminación de un cliente en el sistema.

```php
public function destroy($id)
{
    Gate::authorize('haveaccess','vehiculos.destroy');
    DB::delete('DELETE FROM vehiculos WHERE PLACA = ?', [$id]);
    return redirect('vehiculos');
}
```

F5 Administración de Cocheras

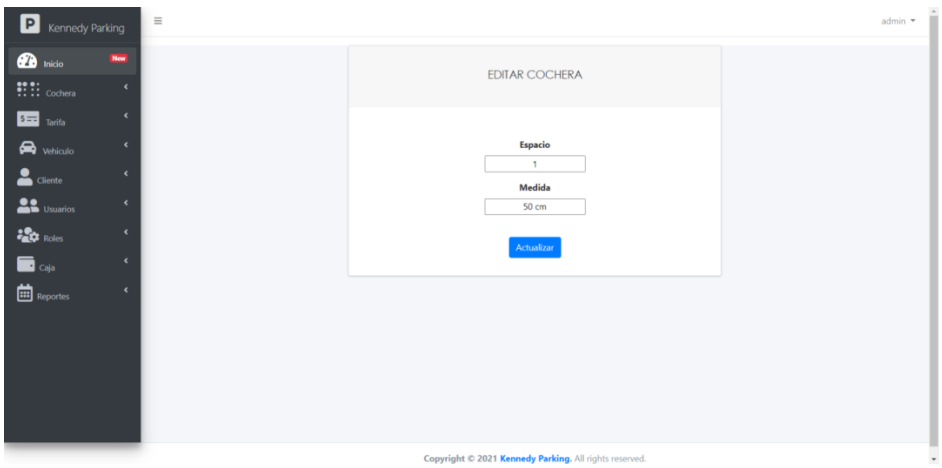
F5.1 Ingresar Cochera



En esta funcionalidad, lo más destacado es el ingreso de una nueva cochera en el sistema con todos sus atributos

```php
public function create()
{
    Gate::authorize('haveaccess','cocheras.create');
    return view('cocheras.create');
}

public function store(Request $request)
{
    Gate::authorize('haveaccess','cocheras.create');
    $data = new \App\Espacio;
    DB::table('cocheras')->insert(
        ['idcochera' => $request['idcochera'],
         'placa' => $request['placa'],
         'numerocochera' => $request['numerocochera'],
         'estado' => $request['estado'],
         'medida' => $request['medida'],
         'fechaingreso' => $request['fechaingreso'],
         'horaingreso' => $request['horaingreso']
        ]
    );
    return redirect('cocheras');
}
```

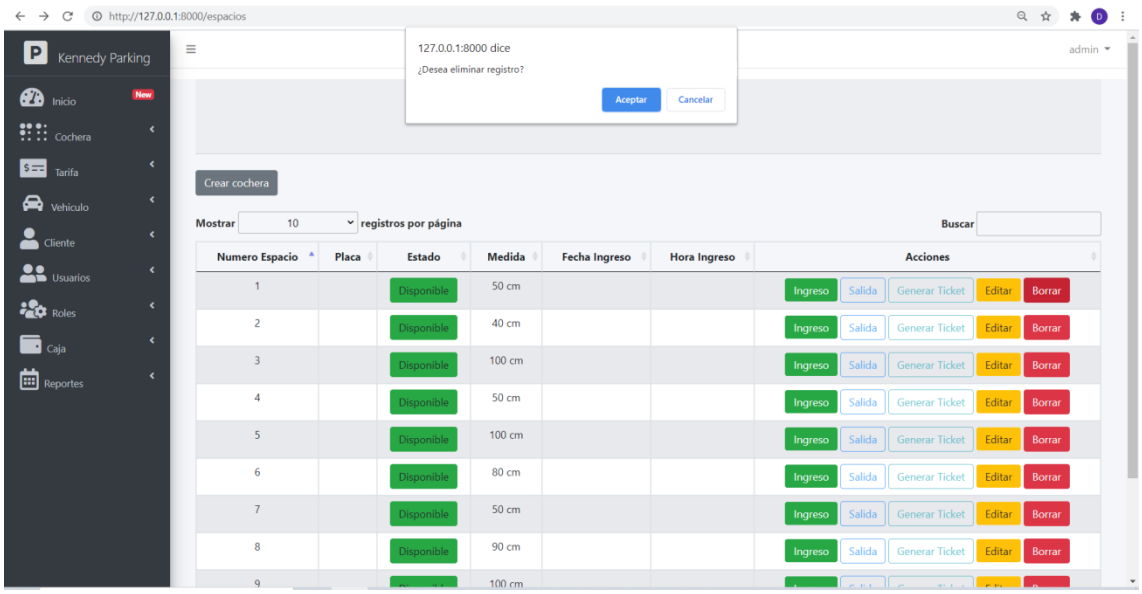
F5.2 Modificar Cochera



```php
public function edit($id)
{
    Gate::authorize('haveaccess','cocheras.edit');
    $cochera = Cochera::findOrFail($id);
    return view('cocheras.edit',compact('cochera'));
}
public function update(Request $request, $id)
{
Gate::authorize('haveaccess','cocheras.edit');
Cochera::where('IDCOCHERA','=',$id)->update(
    ['IDCOCHERA' => $id,
    'placa' => $request['placa'],
    'numerocochera' => $request['numerocochera'],
    'estado' => $request['estado'],
    'medida' => $request['medida'],
    'fechaingreso' => $request['fechaingreso'],
    'horaingreso' => $request['horaingreso']
    ]);
    return redirect('Cochera');
}
```

En esta funcionalidad, lo más destacado la modificación de los atributos una cochera en el sistema.

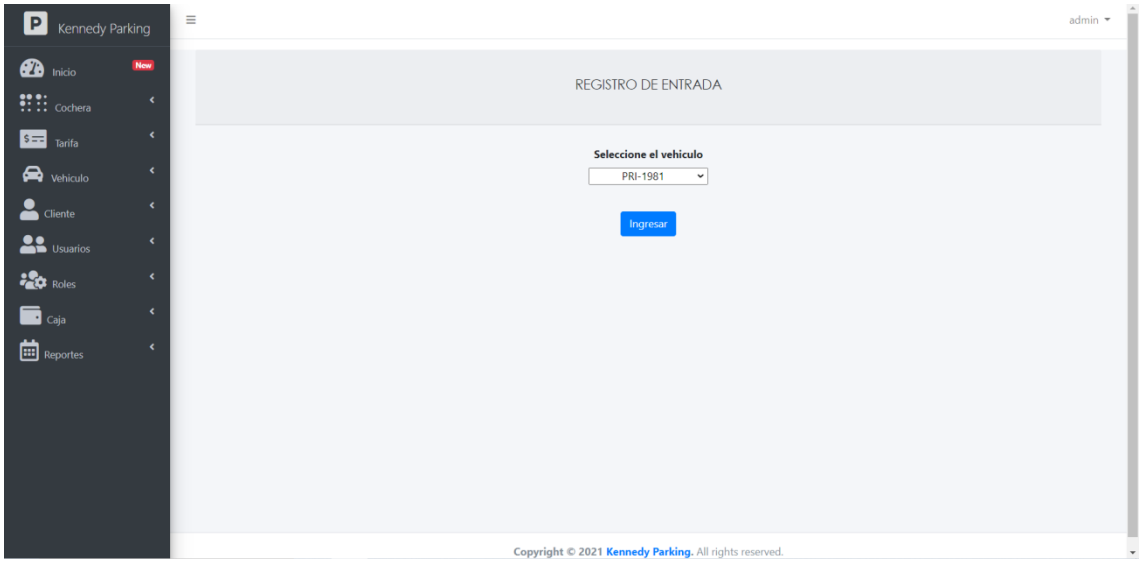
F5.3 Eliminar Cochera




```php
public function destroy($id)
{
    Gate::authorize('haveaccess','Cochera.destroy');
    DB::delete('DELETE FROM cochera WHERE IDCOCHERA = ?', [$id]);
    return redirect('cochera');
```

En esta funcionalidad, lo más destacado la eliminación de una cochera en el sistema.

F5.4 Registro Entrada



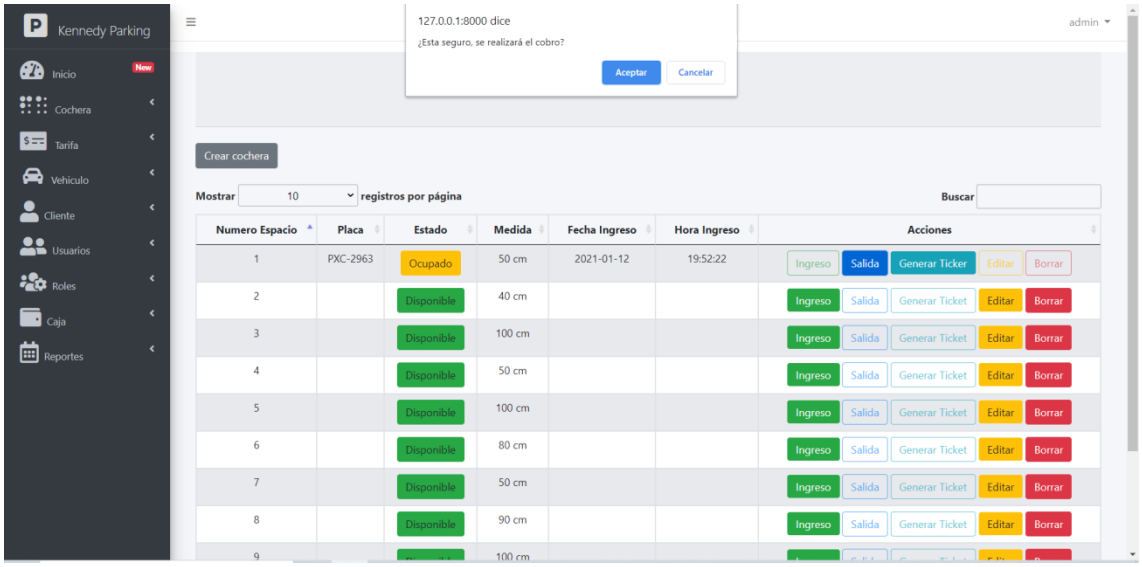

```php
public function ingreso($id)
{
    $cochera = Cochera::findOrFail($id);
    $vehiculo = Vehiculo::all();
    return view('cocheras.ingreso',['vehiculo' => $vehiculo], compact('cocheras'));

}
public function actualizar(Request $request, $id)
{
    $dato = DB::select('select * from Cochera where IDESPACIO = ?', [$id]);
    $mytime = Carbon::now();
    date_default_timezone_set('America/Guayaquil');
    Cochera::where('IDESPACIO','=',$id)->update(
        ['IDESPACIO' => $id,
        'placa' => $request['placa'],
        'numeroespacio' => $dato[0]->NUMEROESPACIO,
        'estado' => "1",
        'medida' => $dato[0]->MEDIDA,
        'fechaingreso' => Carbon::parse(now())->format('Y/m/d'),
        'horaingreso' => Carbon::parse(now())
        ]);
        return redirect('cochera');
}
```

En esta funcionalidad, lo más destacado es el ingreso de un vehículo en una determinada cochera, y se complementa con la siguiente pantalla:

Pantalla ticket entrada

F8 Administración de Asignaciones

F8.1 Salida Vehículo



```
date_default_timezone_set('America/Guayaquil');
$horaentr = Carbon::parse($data[0]->HORAINGRESO);
$horasal = Carbon::parse(now());
$date_diff=$horaentr->diffInMinutes($horasal);
$dia=false;//para verificar la tarifa del dia siguiente cuando entra en la noche
$noche=false;
do {
    //cobro por mes, si se pasa del mes, se cobra por cada dia que se pasa, se co
    if (($date_diff>7200) or ($horaentr->diffInMonths($horasal)==1)) {
        $dias = $horaentr->diffInDays($horasal);//calculo los dias que tiene el m
        if($dias<=31)
        {
            $date_diff=0;
            $total = $total + $tarifa[0]->VALMES;
        }
        else
        {
            $date_diff = $date_diff-44640;
            $total = $total + $tarifa[0]->VALMES;
        }
    }
    //por dia
    elseif (($date_diff >= 1440))
    {
        $date_diff = $date_diff - 1440 ;
        $total = $total + ($tarifa[0]->VALDIA + $tarifa[0]->VALNOCHE);
    }
```

En esta funcionalidad lo más destacado es la salida y cobro de un vehículo de una determinada cochera, y se complementa con la siguiente pantalla:

Pantalla ticket salida

3.4 Pruebas

Para la fase de pruebas, se crea una tabla, en la cual describirá todas las funcionalidades del sistema, mediante la cual evaluamos el comportamiento a las condiciones que pueden suceder y se comprueba la efectividad, confiabilidad y funcionamiento del sistema (Sommerville, 2011).

El Acta de Aceptación del sistema se encuentra en el Anexo 4.

1.5 Implantación

El sistema se puede implantar de dos maneras:

La primera, es alojar el servidor en la nube. En este caso se usaría únicamente como hardware el router y un pc (laptop) como dispositivo.

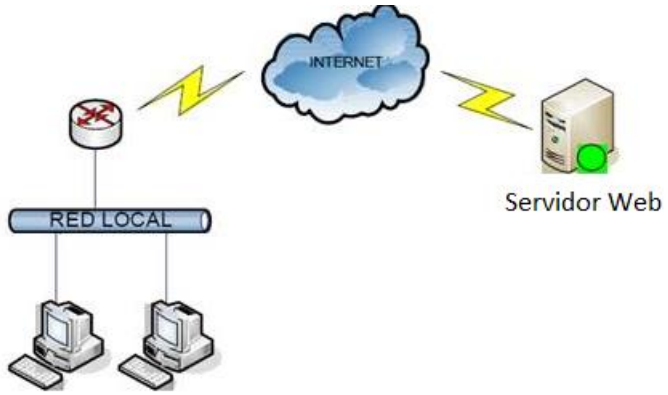


Ilustración 38 Servidor Web (Anónimo, El conquistador, 2015)

La segunda opción, es usar un servidor físico, con una conexión por red de área local

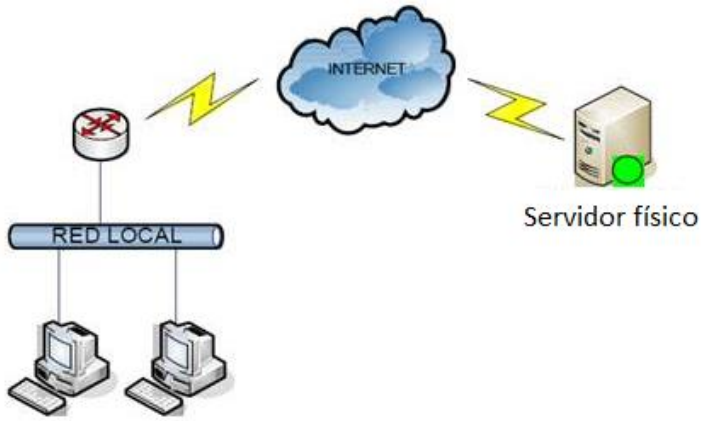


Ilustración 39 Servidor Físico (Anónimo, El conquistador, 2015)

3.6 Operación y Mantenimiento

Para que el sistema pueda funcionar se debe cumplir con los siguientes requisitos:

Requisitos mínimos

- Pc o laptop core i3 a 2.1 GHz de doble núcleo
- Conexión a internet, 1 proveedor de 10mbps
- Impresora térmica
- Generador eléctrico 2300W

Requisitos recomendados

- Pc o laptop core i5 a 3 GHz de doble núcleo
- Conexión a internet, 2 proveedores de 10mbps
- Impresora térmica
- Generador eléctrico 2300W

CAPÍTULO IV CONCLUSIONES Y RECOMENDACIONES

Conclusiones

- El desarrollo del sistema para el establecimiento Kennedy Parking ayuda a la automatización total de sus procesos, por ende, que exista un mejor control de entrada y salida de vehículos.

- El modelo en cascada resultó una metodología muy acertada, ya que desde que se puso en marcha la operación del establecimiento hasta la actualidad no ha existido cambios en los procesos, por lo que se utilizó una sola iteración, ayudando a reducir tiempo en el desarrollo de la disertación.

- El desarrollo de los casos de uso fue muy preciso, ya que en la etapa de diseño se puedo describir a detalle el funcionamiento de cada módulo para posteriormente en la codificación desarrollarlo según el diseño establecido.

- La etapa del levantamiento de requerimientos que se realizó con el propietario del establecimiento fue crucial, ya que ayudo a entender la operación del negocio para optimizar sus procesos principales, además la comunicación con el cliente fue muy importante, ya que, en esta metodología en específico, sirve de base para la construcción del producto que satisfaga las necesidades.

- El desarrollo de roles y permisos para los usuarios fue acertado, puesto que el cliente (administrador) puede restringir accesos hacia los diferentes usuarios y evitar que personal acceda a información propia de la empresa y/o vulnerable.

- El Framework Laravel además que nos ayuda a que el código sea estructurado está basado en el patrón MVC aporta una gran escalabilidad para el sistema en el caso de que se quiera implementar alguna funcionalidad extra.

- El sistema desarrollado, fue probado de manera local con equipos propios del establecimiento, de esta manera se pudo ahorrar dinero en adquisición de infraestructura del establecimiento.

- Para confirmar que el sistema funcione de manera esperada, el cliente hizo pruebas de cada módulo, además de firmar las pruebas de aceptación, de esta manera se confirma la correcta operación del sistema y cada uno de sus módulos.

- El desarrollo de este sistema permitirá al establecimiento tener mayor seguridad y confiabilidad en sus procesos, sobre todo en los datos, es decir, tener toda la información del establecimiento centralizada en un solo sitio, de esta manera los diferentes actores podrán acceder y almacenar fácilmente, garantizando la interoperabilidad.

Recomendaciones

- La selección de la metodología es muy importante, sobre todo si el cliente estará involucrado también en las diferentes fases del ciclo de vida.

- Usar métodos propios que nos brinda el Framework Laravel como lo son: autenticación, envío de mail, registro de usuarios, entre otros.

- Al momento de poner en producción un proyecto, debemos tomar en cuenta los requerimientos recomendados que se plantea, la correcta operación de este.

- Cuando se tiene más de un actor en el sistema es crucial crear roles y permisos para los usuarios, para restringir acceso y evitar que se acceda a información confidencial del administrador.

- El diseño de este sistema se puede aplicar a otro establecimiento de parqueadero tarifado, sin embargo, siempre va a existir pequeños cambios a pesar de que el proceso principal sea el mismo.

- Este sistema se podría mejorar utilizando un módulo de Arduino que en un futuro se podría utilizar como un proyecto complementario.

- A la hora de cerrar el establecimiento, es recomendable apagar el sistema, ya que reduciríamos el consumo de energía eléctrica y vida útil del equipo de computación.

- Tomar en cuenta los requisitos recomendados para la implantación del sistema, con esto garantizamos que la operación del sistema no se vea afectada bajo circunstancia alguna.

Anexo 1

F2. Administración de Cajas

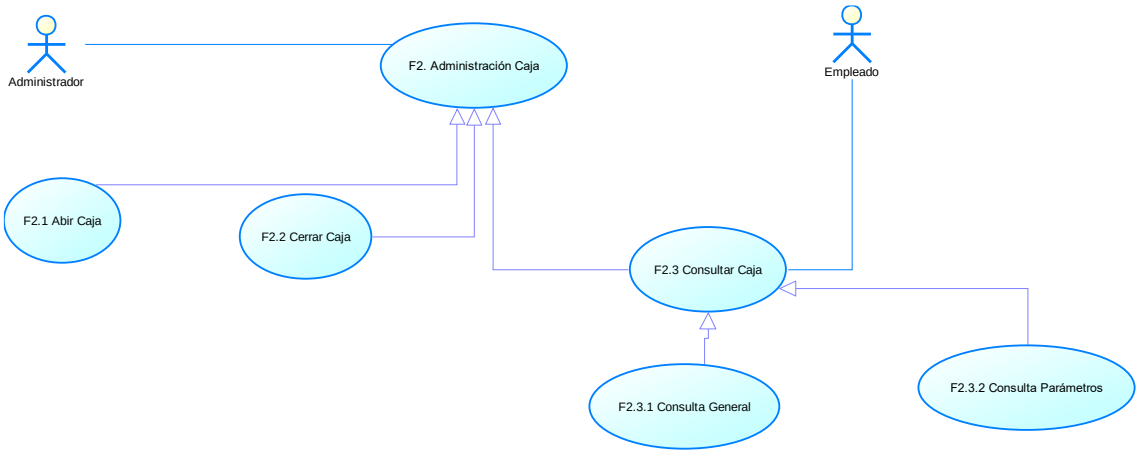


Ilustración 40 Caso de uso a detalle Cajas

Ilustración 41 Abrir Caja

F2.1 Abrir Caja

Descripción: Con esta funcionalidad, el actor podrá abrir una caja en el sistema con todos sus atributos.

Actores: Administrador.

Flujo Principal:

1. El sistema despliega el menú principal.
2. El actor selecciona la opción cajas.
3. El sistema despliega la ventana de cajas (E1).
4. El actor presiona el botón Crear caja.
5. El sistema despliega la ventana de creación de caja.
6. El actor ingresa los campos solicitados.
7. El actor presiona el botón crear.
8. El sistema almacena la información (E1).
9. El sistema crea una nueva caja.

Flujo Alterno

1. Si ya existe, ver el caso de uso F3.2, F3.3

Excepciones

Código	Causa	Mensaje
E1	Error de conexión con la base de datos	Contactar al soporte.

F2.2 Cerrar Caja

Ilustración 42 Cerrar Caja

Descripción: Con esta funcionalidad, el actor podrá cerrar una caja del sistema.

Actores: Administrador, Empleado.

Flujo Principal:

1. El sistema despliega el menú principal
2. El actor selecciona la opción cajas.
3. El sistema despliega la ventana de cajas (E1).
4. El actor presiona el botón Cerrar Caja en el registro deseado.
5. El actor confirma la ventana de cierre
6. E l sistema actualiza la información de la caja (E1).

Excepciones

Código	Causa	Mensaje
E1	Error de conexión con la base de datos	Contactar al soporte.

F2.3.1 Consulta General

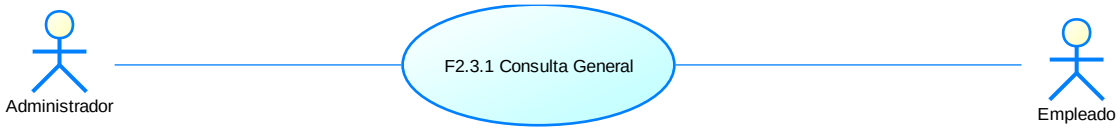


Ilustración 43 Consulta General Cajas

Descripción: Con esta funcionalidad, el actor podrá consultar todas las cajas del sistema con todos sus atributos existentes.

Actores: Administrador, Empleado.

Flujo Principal:

1. El sistema despliega el menú principal.
2. El actor selecciona la opción cajas.

3. El sistema despliega la información (E1).

Excepciones

Código	Causa	Mensaje
E1	Error de conexión con la base de datos	Contactar al soporte.

F2.3.2 Consulta por Parámetros

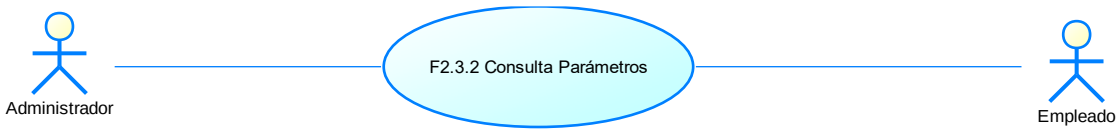


Ilustración 44 Consulta Parámetros Caja

Descripción: Con esta funcionalidad, el actor podrá consultar cajas bajo parámetros con todos sus atributos existentes.

Actores: Administrador, Empleado.

Flujo Principal:

1. El sistema despliega el menú principal.
2. El actor selecciona la opción cajas.
3. El sistema despliega la ventana (E1).
4. El actor escribe la palabra de búsqueda.
5. El sistema despliega la información (E1).

Excepciones

Código	Causa	Mensaje
E1	Error de conexión con la base de datos	Contactar al soporte.

F3. Administración de Tarifas

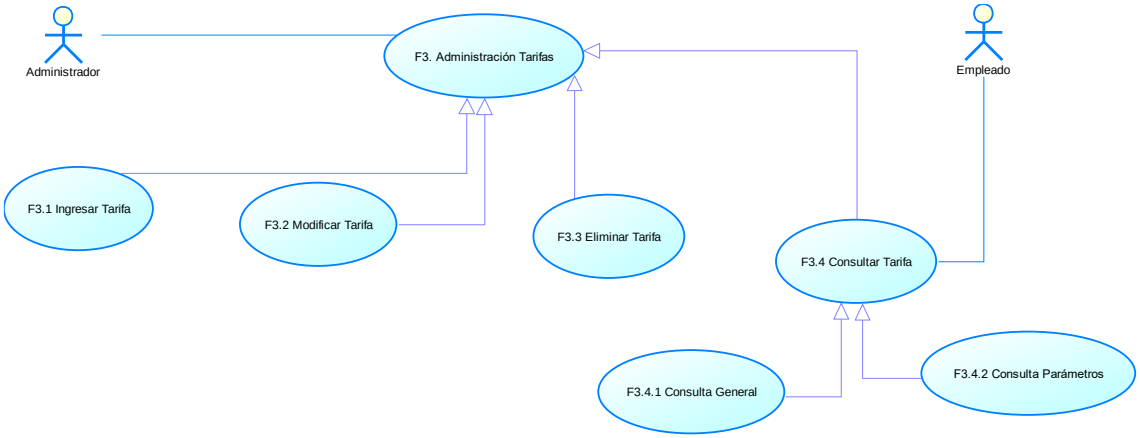


Ilustración 45 Caso de uso a detalle tarifas

Ilustración 46 Ingresar Tarifa

F3.1 Ingresar Tarifa

Descripción: Con esta funcionalidad, el actor podrá agregar una tarifa al sistema con todos sus atributos.

Actores: Administrador, Empleado.

Flujo Principal:

1. El sistema despliega el menú principal.
2. El actor selecciona la opción tarifas.
3. El sistema despliega la ventana de tarifas (E1).
4. El actor presiona el botón Crear tarifa.
5. El sistema despliega la ventana de creación de tarifa.
6. El actor ingresa los campos solicitados.
7. El actor presiona el botón crear.
8. El sistema almacena la información (E1).
9. El sistema crea una nueva tarifa.

Flujo Alterno

1. Si ya existe, ver el caso de uso F4.2, F4.3

Excepciones

Código	Causa	Mensaje
E1	Error de conexión con la base de datos	Contactar al soporte.

F3.2 Modificar Tarifa

Ilustración 47 Modificar Tarifa

Descripción: Con esta funcionalidad, el actor podrá modificar una tarifa del sistema con todos sus atributos existentes.

Actores: Administrador, Empleado.

Flujo Principal:

1. El sistema despliega el menú principal
2. El actor selecciona la opción tarifas.
3. El sistema despliega la ventana de tarifas (E1).
4. El actor presiona el botón Editar en el registro deseado.
5. El sistema despliega la ventana de edición de tarifa.
6. El actor modifica los campos solicitados.
7. El actor presiona el botón guardar.
8. El sistema actualiza la información de la tarifa (E1).

Excepciones

Código	Causa	Mensaje
E1	Error de conexión con la base de datos	Contactar al soporte.

F3.3 Eliminar Tarifa

Ilustración 48 Eliminar Tarifa

Descripción: Con esta funcionalidad, el actor podrá eliminar una tarifa del sistema con todos sus atributos existentes.

Actores: Administrador.

Flujo Principal:

1. El sistema despliega el menú principal.
2. El actor selecciona la opción tarifas.
3. El sistema despliega la ventana de tarifas. (E1).
4. El actor presiona el botón Eliminar en el registro requerido.
5. El sistema despliega la ventana de confirmación.
6. El actor confirma la eliminación del ítem.
7. El sistema elimina el registro (E1).

F3.4.1 Consulta General

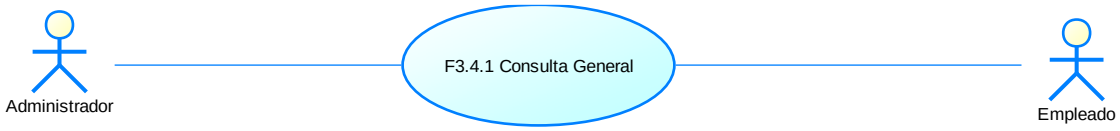


Ilustración 49 Consulta General Tarifas

Descripción: Con esta funcionalidad, el actor podrá consultar todas las tarifas del sistema con todos sus atributos existentes.

Actores: Administrador, Empleado.

Flujo Principal:

1. El sistema despliega el menú principal.
2. El actor selecciona la opción tarifas.
3. El sistema despliega la información (E1).

Excepciones

Código	Causa	Mensaje
E1	Error de conexión con la base de datos	Contactar al soporte.

F3.4.2 Consulta por Parámetros

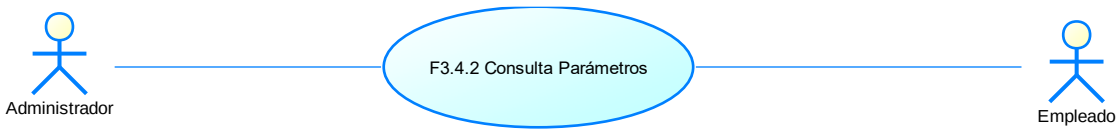


Ilustración 50 Consulta Parámetros Tarifa

Descripción: Con esta funcionalidad, el actor podrá consultar tarifas bajo parámetros con todos sus atributos existentes.

Actores: Administrador, Empleado.

Flujo Principal:

1. El sistema despliega el menú principal.
2. El actor selecciona la opción tarifas.
3. El sistema despliega la ventana (E1).
4. El actor escribe la palabra de búsqueda.
5. El sistema despliega la información (E1).

Excepciones

Código	Causa	Mensaje
E1	Error de conexión con la base de datos	Contactar al soporte.

F6. Administración de Usuarios

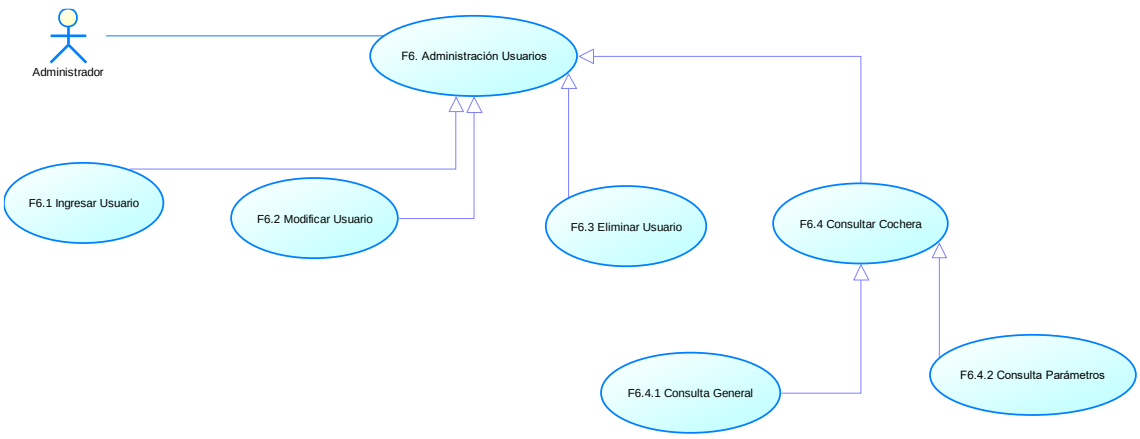


Ilustración 51 Caso de uso a detalle Usuarios

Ilustración 52 Ingresar Usuario

F6.1 Ingresar Usuario

Descripción: Con esta funcionalidad, el actor podrá agregar un usuario al sistema con todos sus atributos.

Actores: Administrador.

Flujo Principal:

1. El sistema despliega el menú principal.
2. El actor selecciona la opción usuarios.

3. El sistema despliega la ventana de usuarios (E1).
4. El actor presiona el botón Crear usuario.
5. El sistema despliega la ventana de creación de usuario.
6. El actor ingresa los campos solicitados.
7. El actor presiona el botón crear.
8. El sistema almacena la información (E1).
9. El sistema crea un nuevo usuario.

Flujo Alterno

1. Si ya existe, ver el caso de uso F7.2, F7.3

Excepciones

Código	Causa	Mensaje
E1	Error de conexión con la base de datos	Contactar al soporte.

F6.2 Modificar Usuario

Ilustración 53 Modificar Usuario

Descripción: Con esta funcionalidad, el actor podrá modificar un usuario del sistema con todos sus atributos existentes.

Actores: Administrador.

Flujo Principal:

1. El sistema despliega el menú principal
2. El actor selecciona la opción usuarios.
3. El sistema despliega la ventana de cocheras (E1).
4. El actor presiona el botón Editar en el registro deseado.
5. El sistema despliega la ventana de edición de usuario.
6. El actor modifica los campos solicitados.
7. El actor presiona el botón guardar.
8. El sistema actualiza la información de la cochera (E1).

Excepciones

Código	Causa	Mensaje
E1	Error de conexión con la base de datos	Contactar al soporte.

F6.3 Eliminar Usuario

Ilustración 54 Eliminar Usuario

Descripción: Con esta funcionalidad, el actor podrá eliminar un usuario del sistema con todos sus atributos existentes.

Actores: Administrador.

Flujo Principal:

1. El sistema despliega el menú principal.
2. El actor selecciona la opción usuarios.
3. El sistema despliega la ventana de usuarios (E1).
4. El actor presiona el botón Eliminar en el registro requerido.
5. El sistema despliega la ventana de confirmación.
6. El actor confirma la eliminación del ítem.
7. El sistema elimina el registro (E1).

Excepciones

Código	Causa	Mensaje
E1	Error de conexión con la base de datos	Contactar al soporte.

F6.4.1 Consulta General

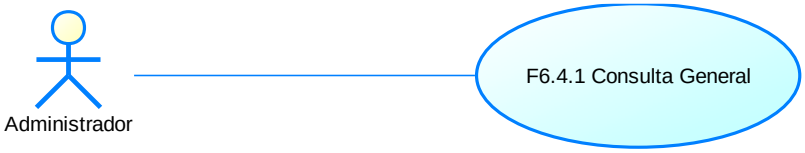


Ilustración 55 Consulta General Usuarios

Descripción: Con esta funcionalidad, el actor podrá consultar todos los usuarios del sistema con todos sus atributos existentes.

Actores: Administrador.

Flujo Principal:

1. El sistema despliega el menú principal.
2. El actor selecciona la opción usuarios.
3. El sistema despliega la información (E1).

Excepciones

Código	Causa	Mensaje
E1	Error de conexión con la base de datos	Contactar al soporte.

F6.4.2 Consulta por Parámetros

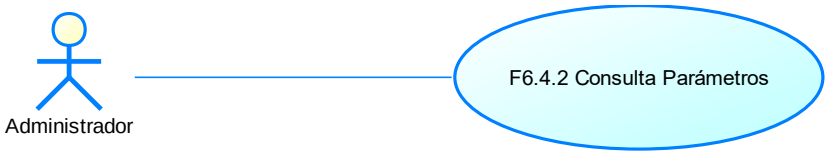


Ilustración 56 Consulta Parámetros Usuario

Descripción: Con esta funcionalidad, el actor podrá consultar usuarios bajo parámetros con todos sus atributos existentes.

Actores: Administrador.

Flujo Principal:

1. El sistema despliega el menú principal.
2. El actor selecciona la opción usuarios.
3. El sistema despliega la ventana (E1).
4. El actor escribe la palabra de búsqueda.
5. El sistema despliega la información (E1).

Excepciones

Código	Causa	Mensaje
E1	Error de conexión con la base de datos	Contactar al soporte.

F7. Administración de Roles

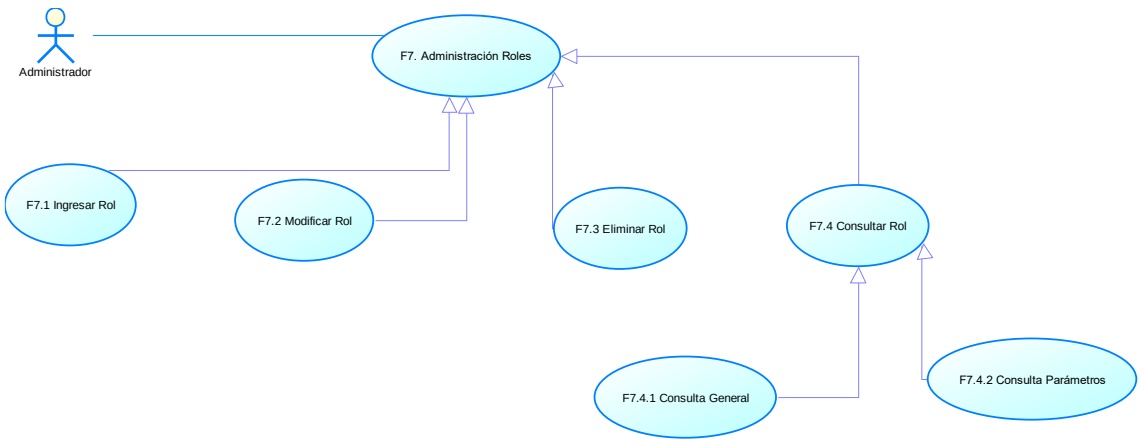


Ilustración 57 Caso de uso a detalle Roles

Ilustración 58 Ingresar Rol

F7.1 Ingresar Rol

Descripción: Con esta funcionalidad, el actor podrá agregar un rol al sistema con todos sus atributos.

Actores: Administrador.

Flujo Principal:

1. El sistema despliega el menú principal.
2. El actor selecciona la opción roles.
3. El sistema despliega la ventana de roles (E1).
4. El actor presiona el botón Crear rol.
5. El sistema despliega la ventana de creación de rol.
6. El actor ingresa los campos solicitados.
7. El actor presiona el botón crear.
8. El sistema almacena la información (E1).
9. El sistema crea un nuevo rol.

Flujo Alterno

1. Si ya existe, ver el caso de uso F8.2, F8.3

Excepciones

Código	Causa	Mensaje

E1	Error de conexión con la base de datos	Contactar al soporte.

F7.2 Modificar Rol

Ilustración 59 Modificar Rol

Descripción: Con esta funcionalidad, el actor podrá modificar un rol del sistema con todos sus atributos existentes.

Actores: Administrador.

Flujo Principal:

1. El sistema despliega el menú principal
2. El actor selecciona la opción roles.
3. El sistema despliega la ventana de roles (E1).
4. El actor presiona el botón Editar en el registro deseado.
5. El sistema despliega la ventana de edición de rol.
6. El actor modifica los campos solicitados.
7. El actor presiona el botón guardar.
8. El sistema actualiza la información del rol (E1).

Excepciones

Código	Causa	Mensaje
E1	Error de conexión con la base de datos	Contactar al soporte.

F7.3 Eliminar Rol

Ilustración 60 Eliminar Rol

Descripción: Con esta funcionalidad, el actor podrá eliminar un rol del sistema con todos sus atributos existentes.

Actores: Administrador.

Flujo Principal:

1. El sistema despliega el menú principal.
2. El actor selecciona la opción roles.
3. El sistema despliega la ventana de roles (E1).
4. El actor presiona el botón Eliminar en el registro requerido.
5. El sistema despliega la ventana de confirmación.
6. El actor confirma la eliminación del ítem.
7. El sistema elimina el registro (E1).

Excepciones

Código	Causa	Mensaje
E1	Error de conexión con la base de datos	Contactar al soporte.

F7.4.1 Consulta General

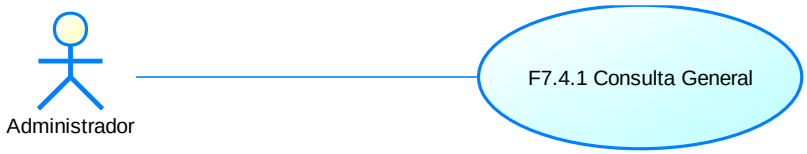


Ilustración 61 Consulta General Roles

Descripción: Con esta funcionalidad, el actor podrá consultar todos los roles del sistema con todos sus atributos existentes.

Actores: Administrador.

Flujo Principal:

1. El sistema despliega el menú principal.
2. El actor selecciona la opción roles.
3. El sistema despliega la información (E1).

Excepciones

Código	Causa	Mensaje
E1	Error de conexión con la base de datos	Contactar al soporte.

F7.4.2 Consulta por Parámetros

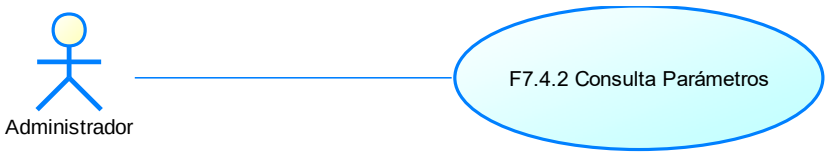


Ilustración 62 Consulta Parámetros Roles

Descripción: Con esta funcionalidad, el actor podrá consultar usuarios bajo parámetros con todos sus atributos existentes.

Actores: Administrador.

Flujo Principal:

1. El sistema despliega el menú principal.
2. El actor selecciona la opción roles.
3. El sistema despliega la ventana (E1).
4. El actor escribe la palabra de búsqueda.
5. El sistema despliega la información (E1).

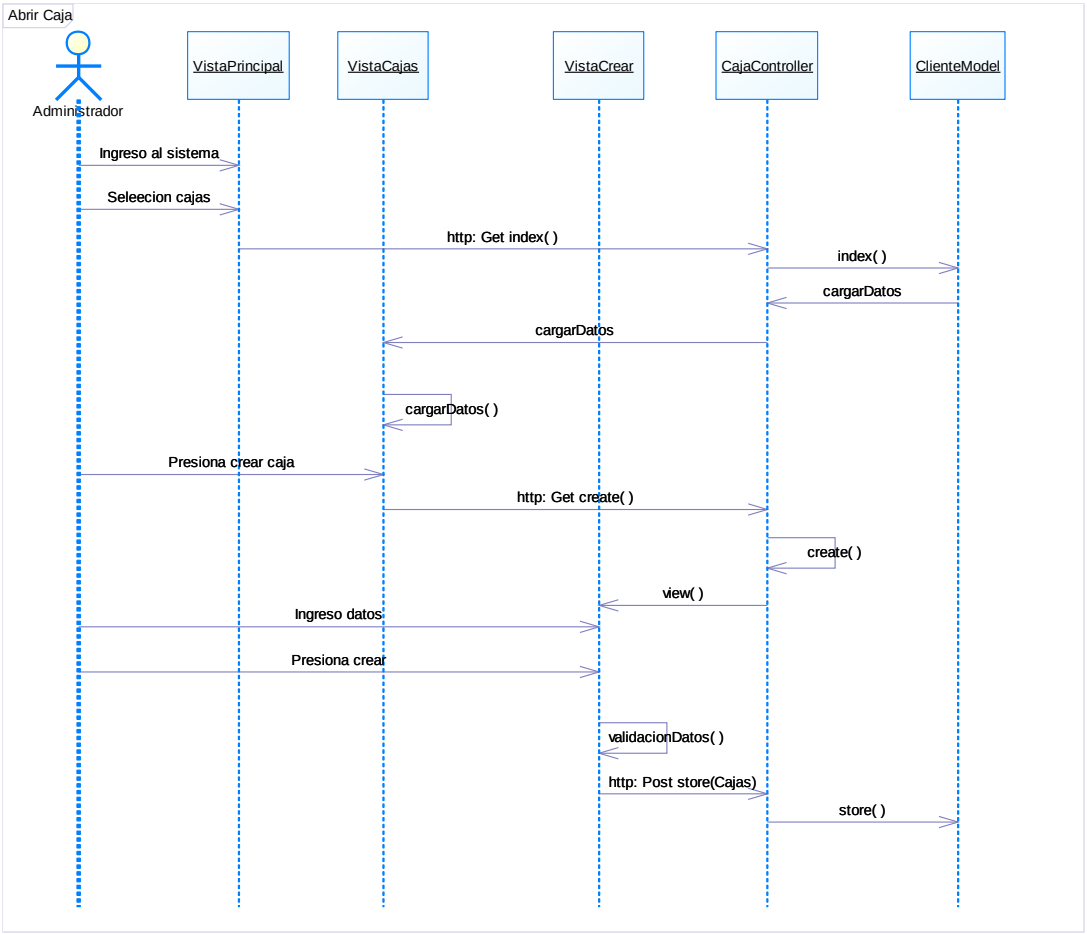
Excepciones

Código	Causa	Mensaje
E1	Error de conexión con la base de datos	Contactar al soporte.

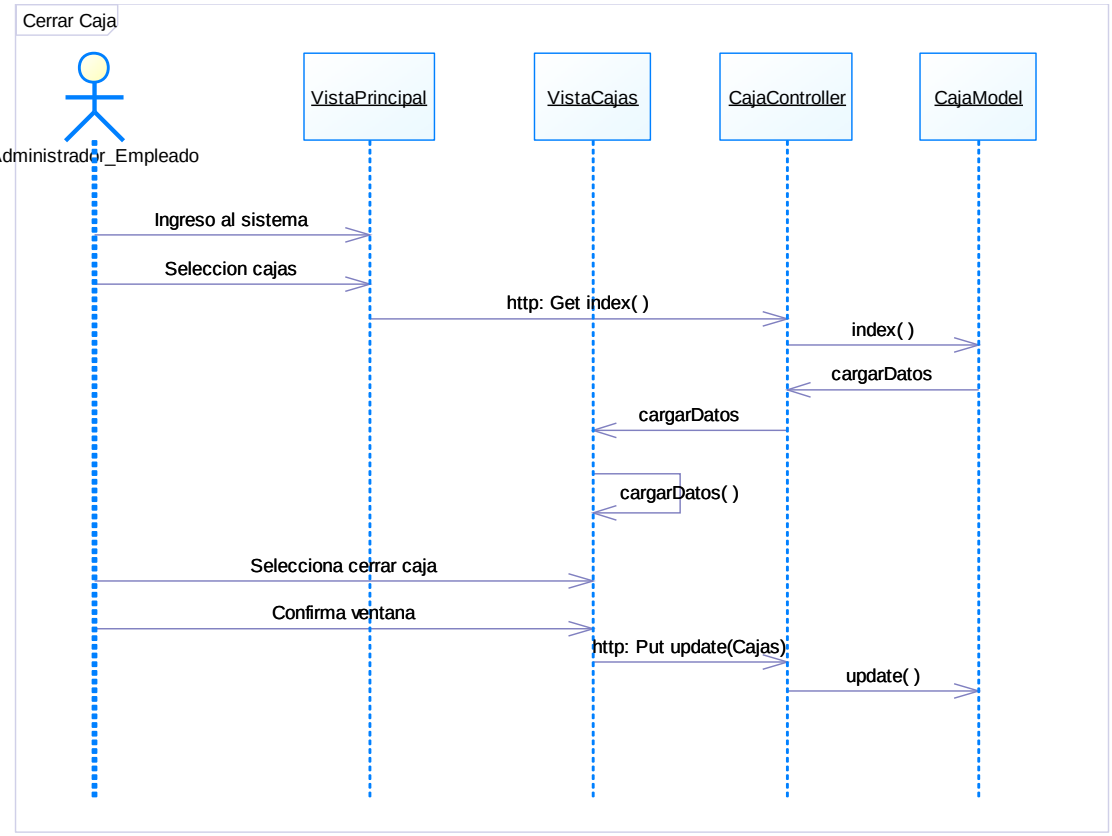
Anexo 2

F2 Administracion de Cajas

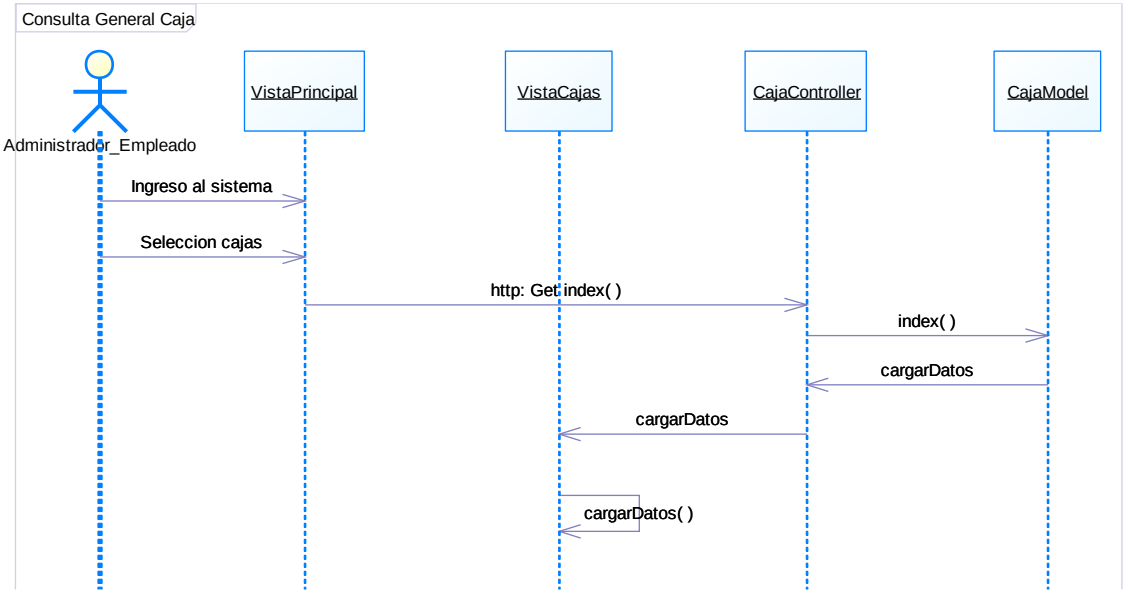
F2.1 Abrir Caja



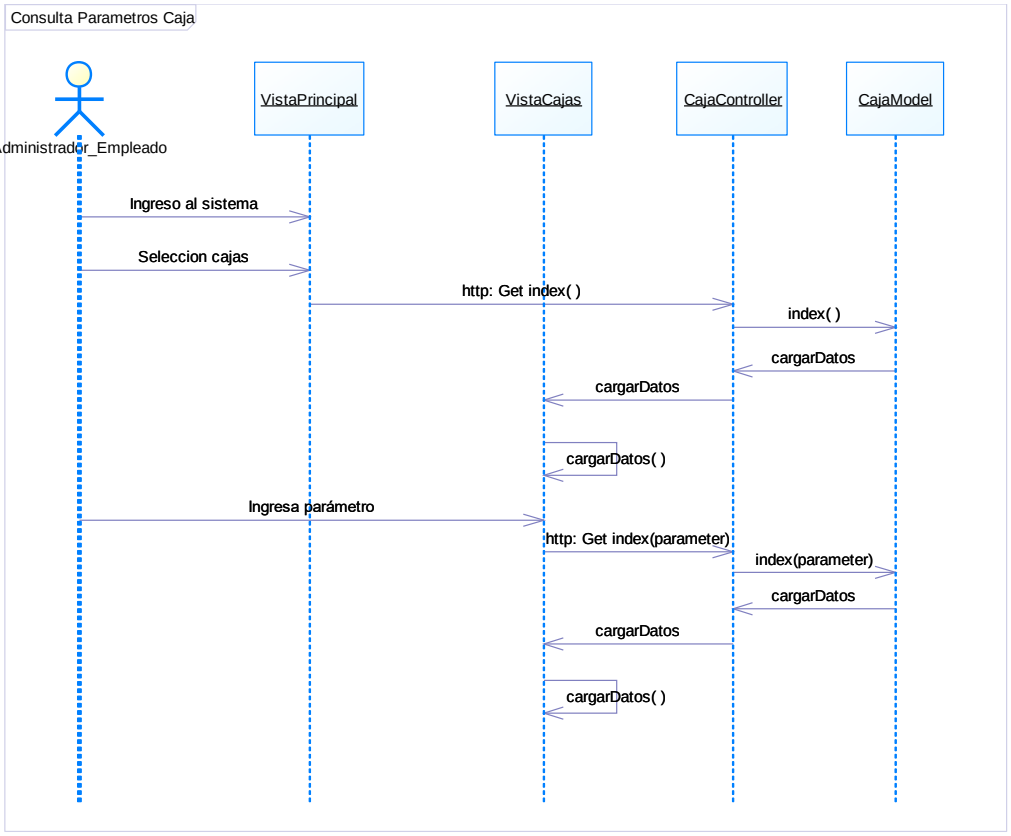
F2.2 Cerrar Caja



F2.3.1 Consulta General

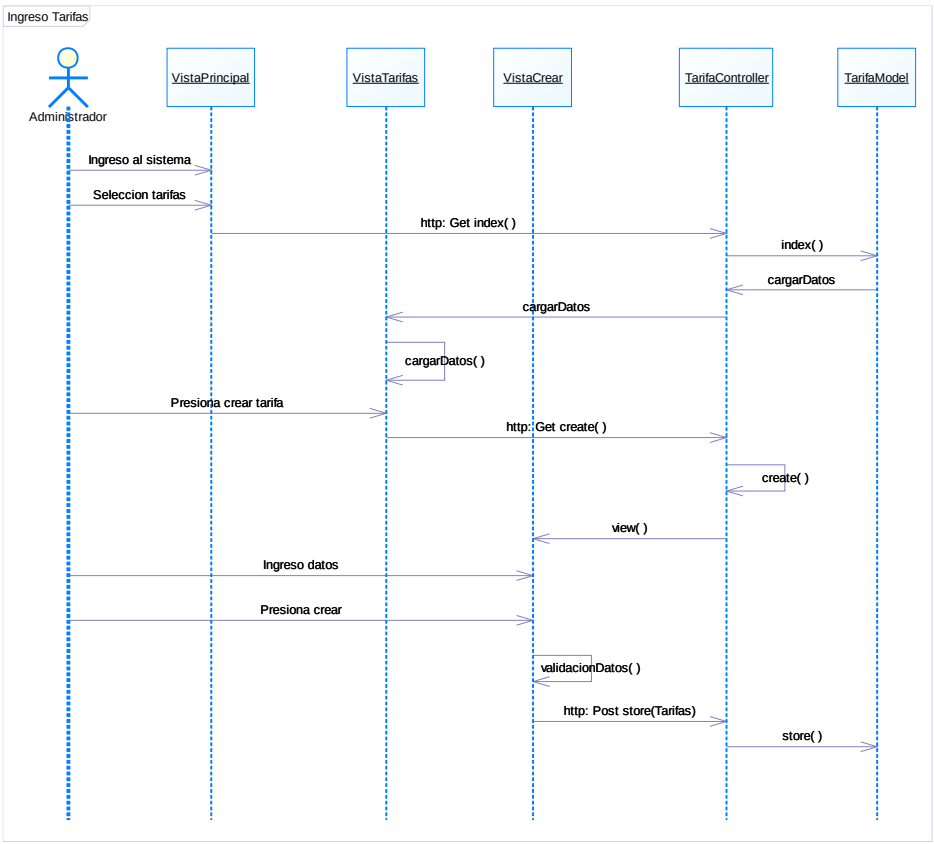


F2.3.2 Consulta Parámetros

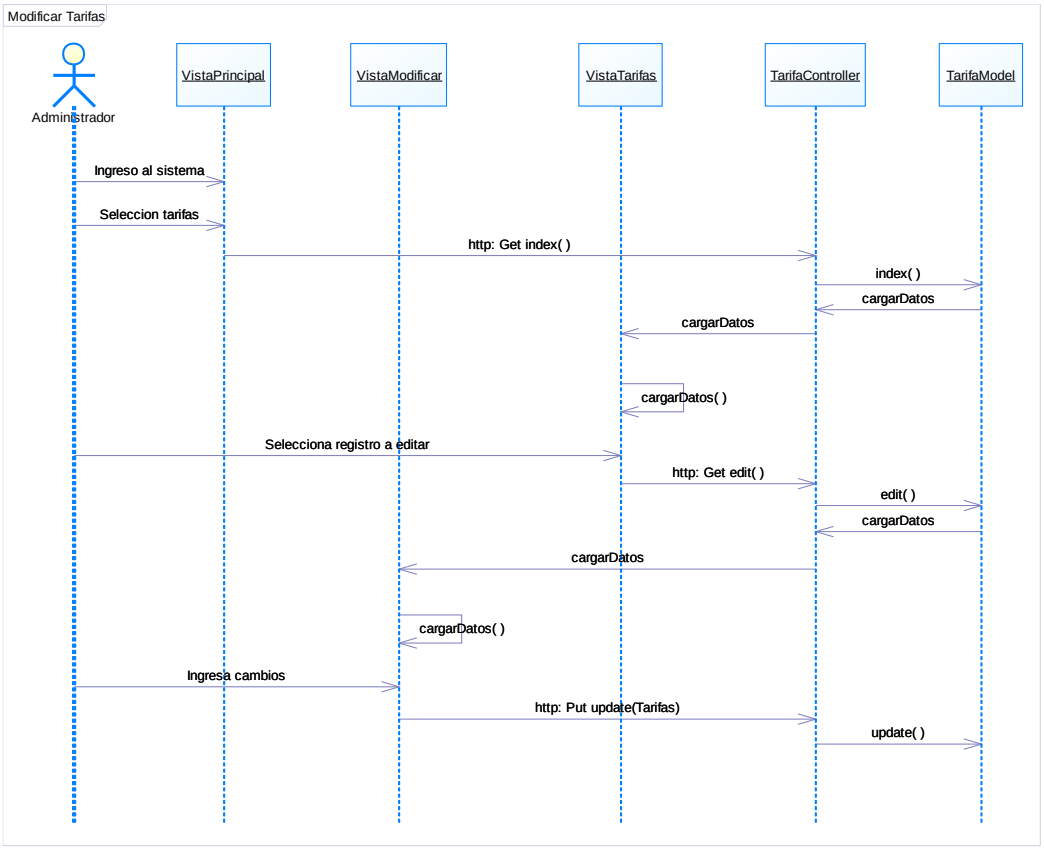


F3 Administración de Tarifas

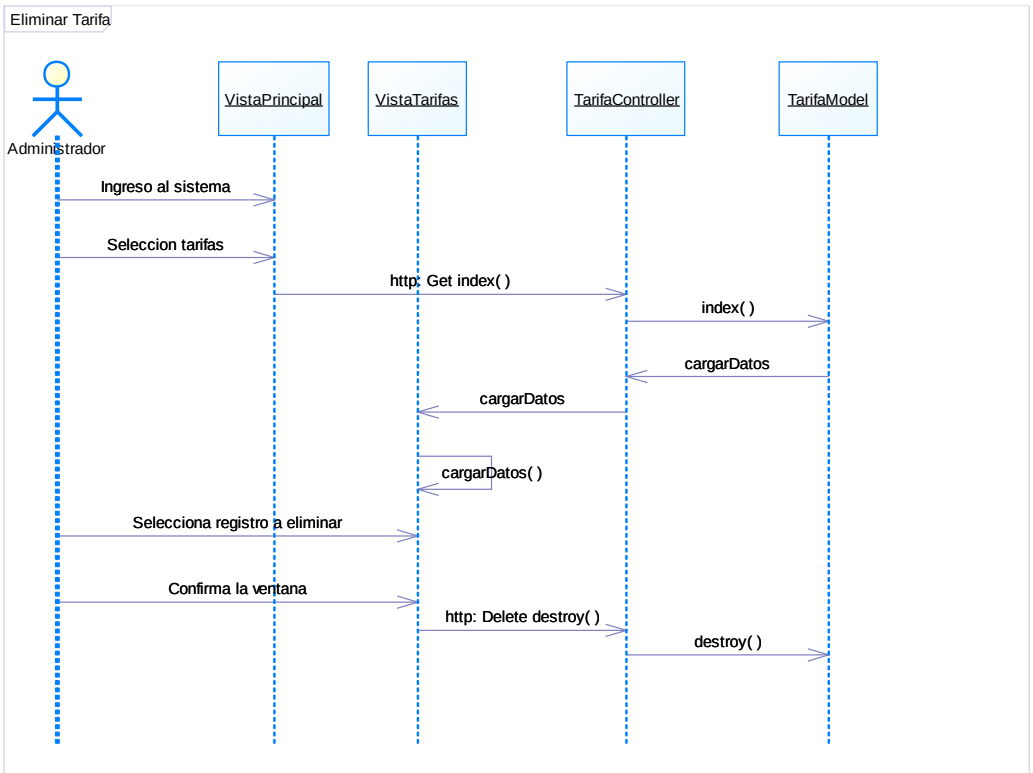
F3.1 Ingreso Tarifa



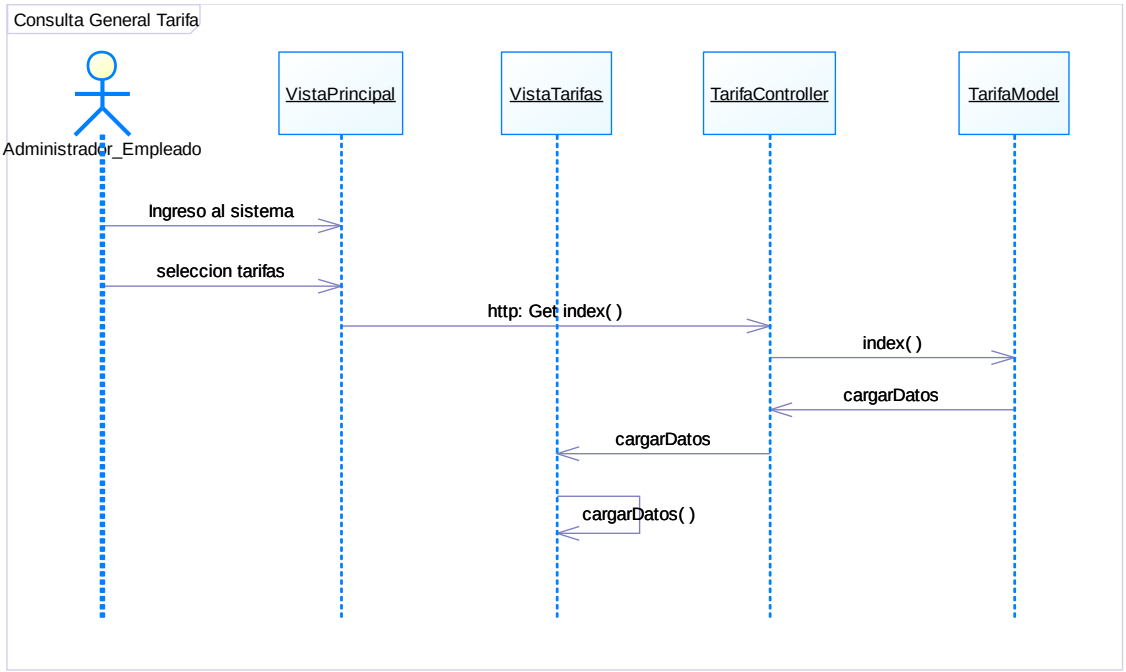
F3.2 Ingreso Modificar Tarifa



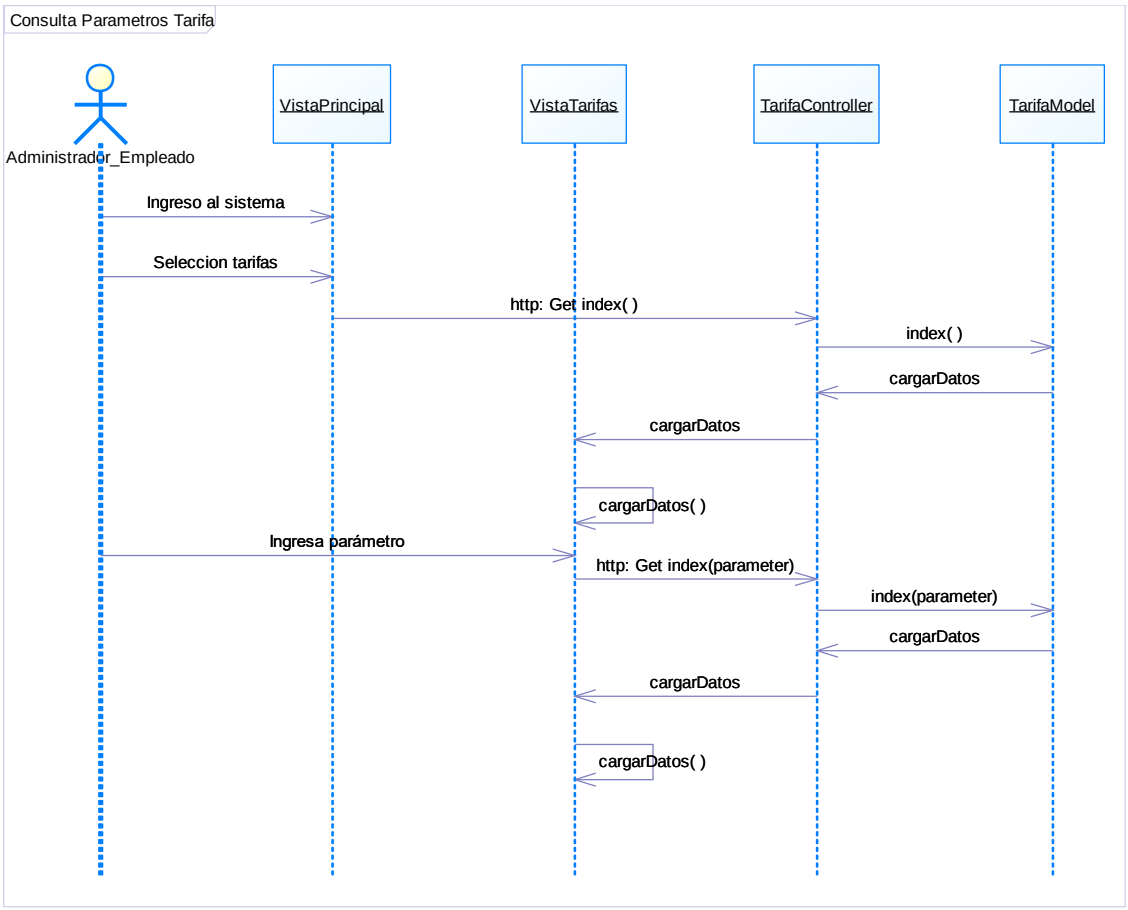
F3.3 Eliminar Tarifa



F3.4.1 Consulta General

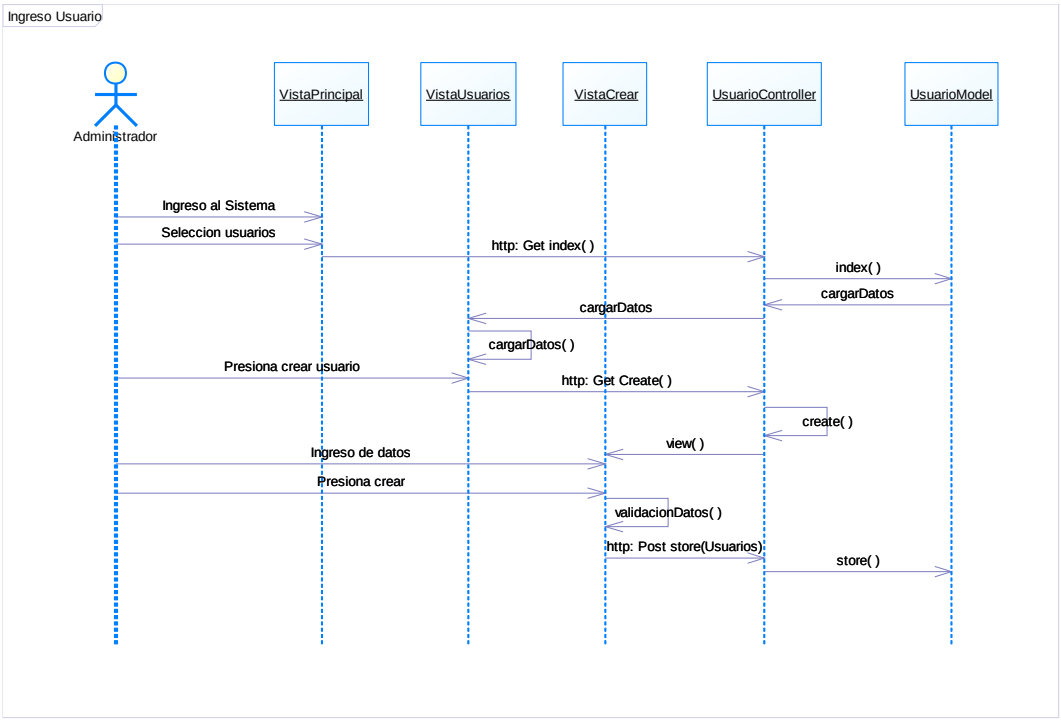


F3.4.2 Consulta Parámetros

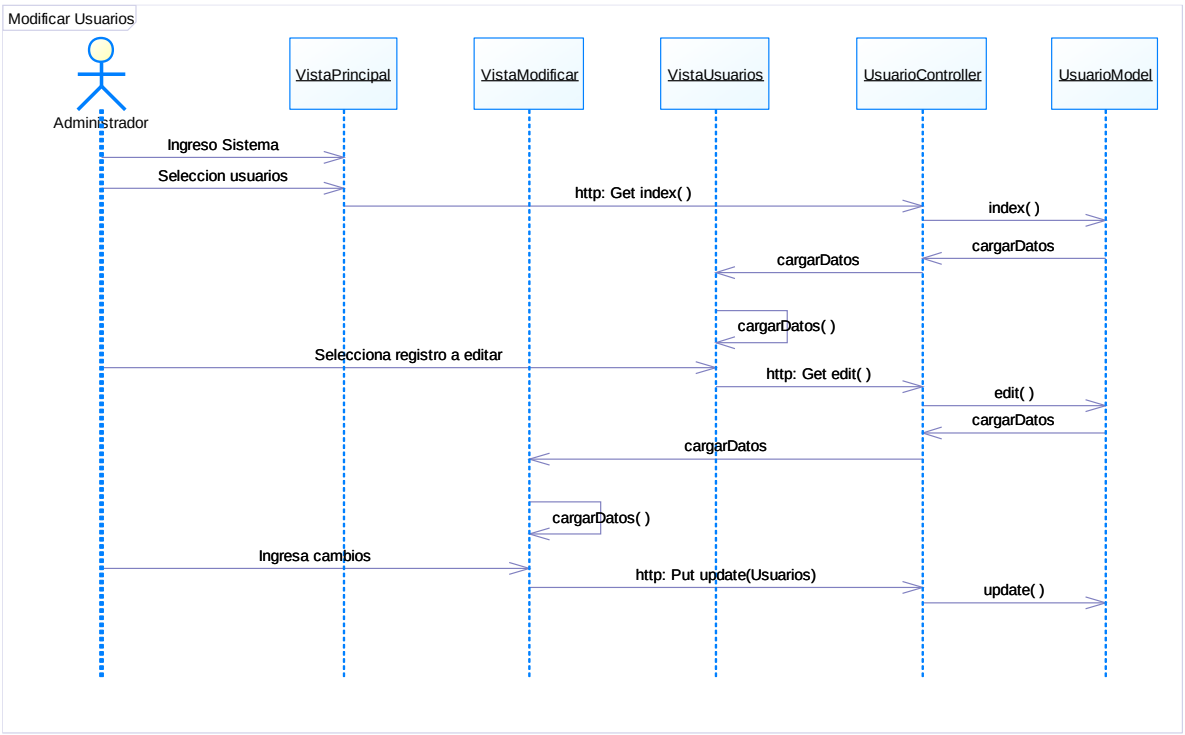


F6 Administración de Usuarios

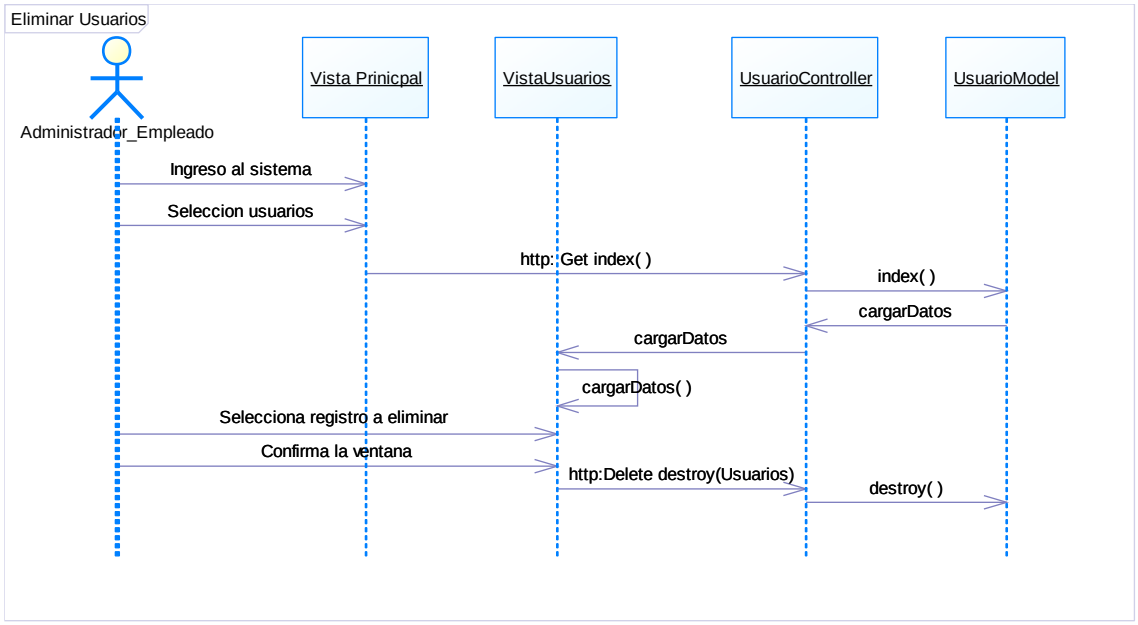
F6.1 Ingreso Usuario



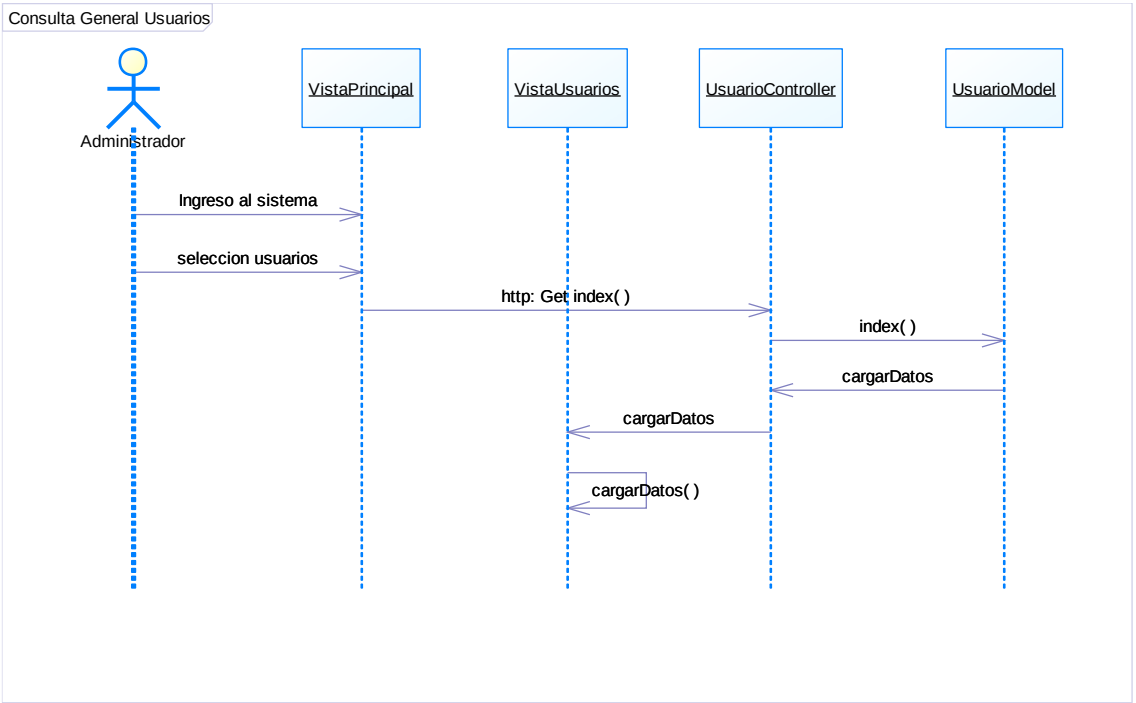
F6.2 Modificar Usuario



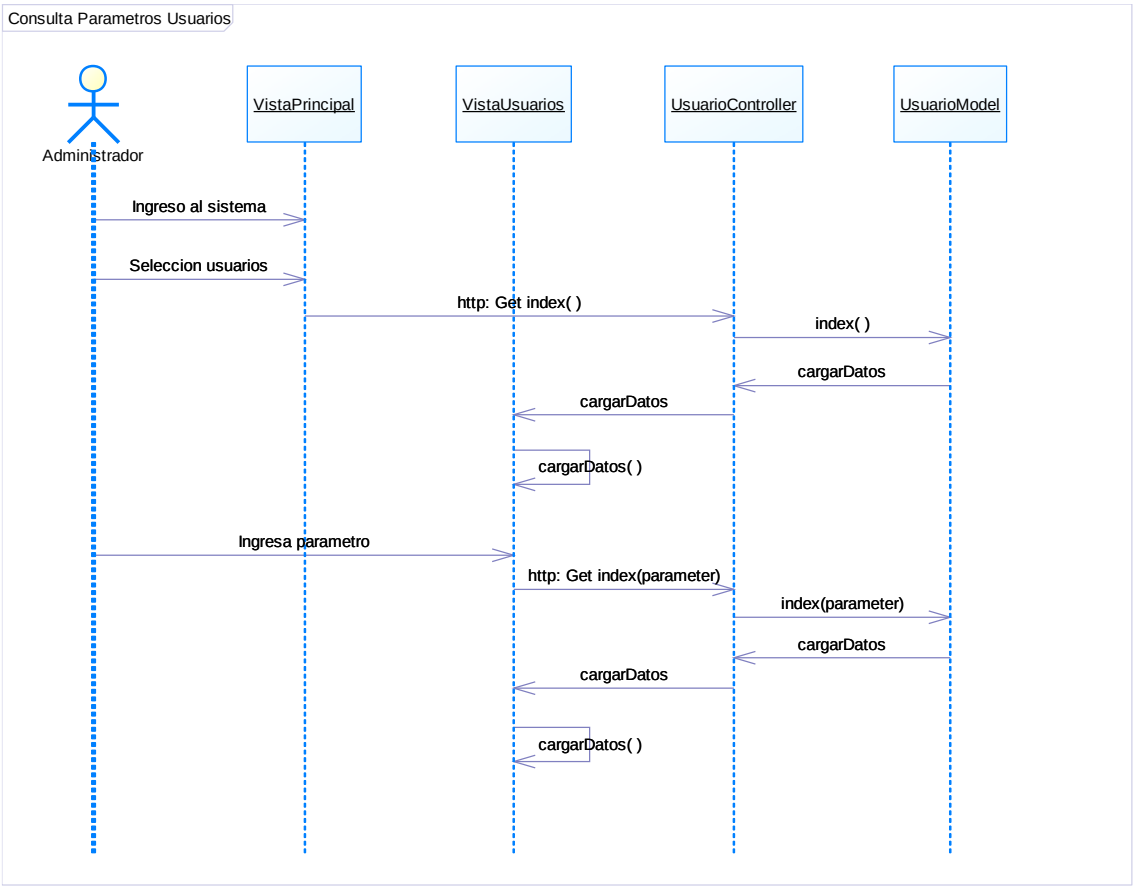
F6.3 Eliminar Usuario



F6.4.1 Consulta General

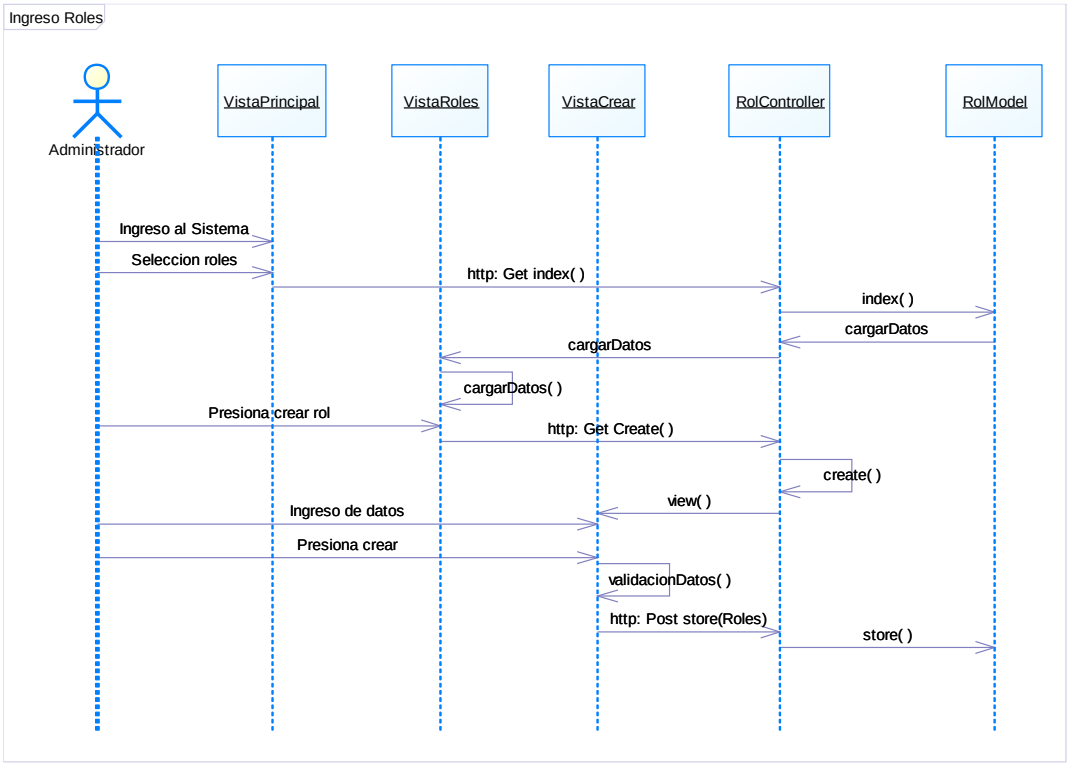


F6.4.2 Consulta Parámetros

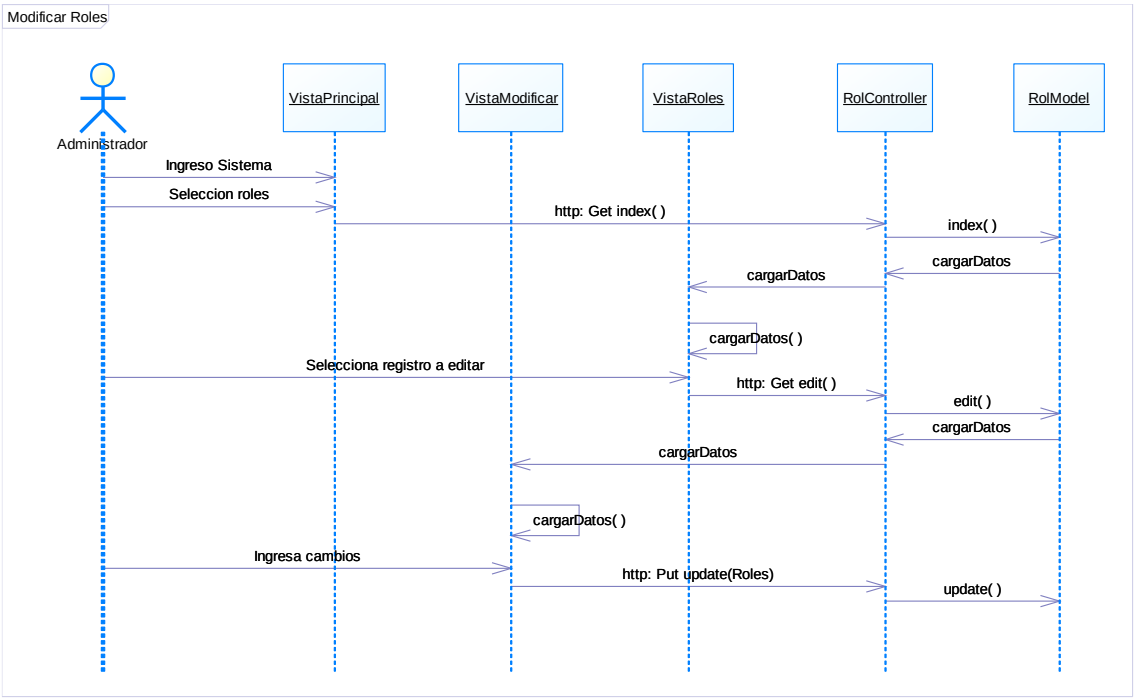


F7 Administración de Roles

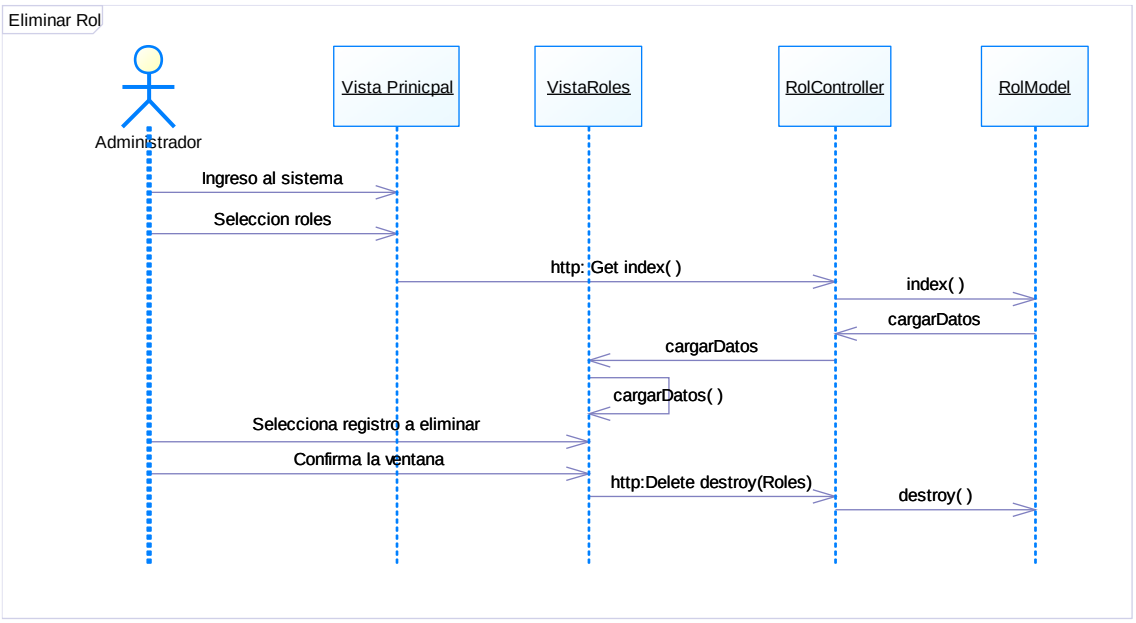
F7.1 Ingreso Rol



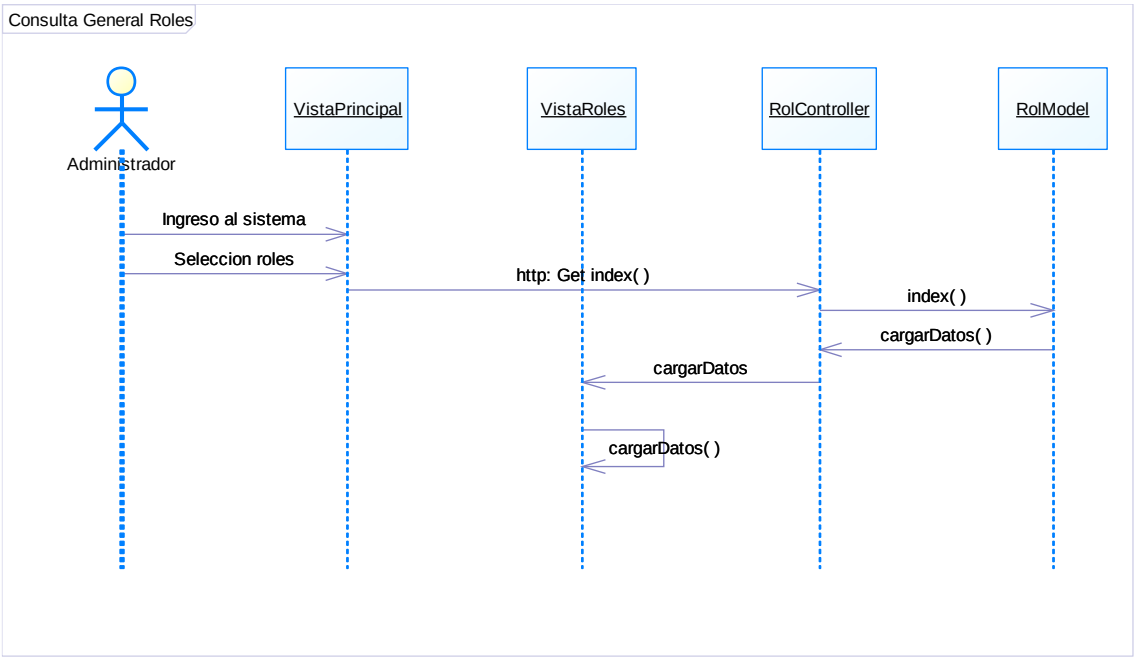
F7.2 Modificar Rol



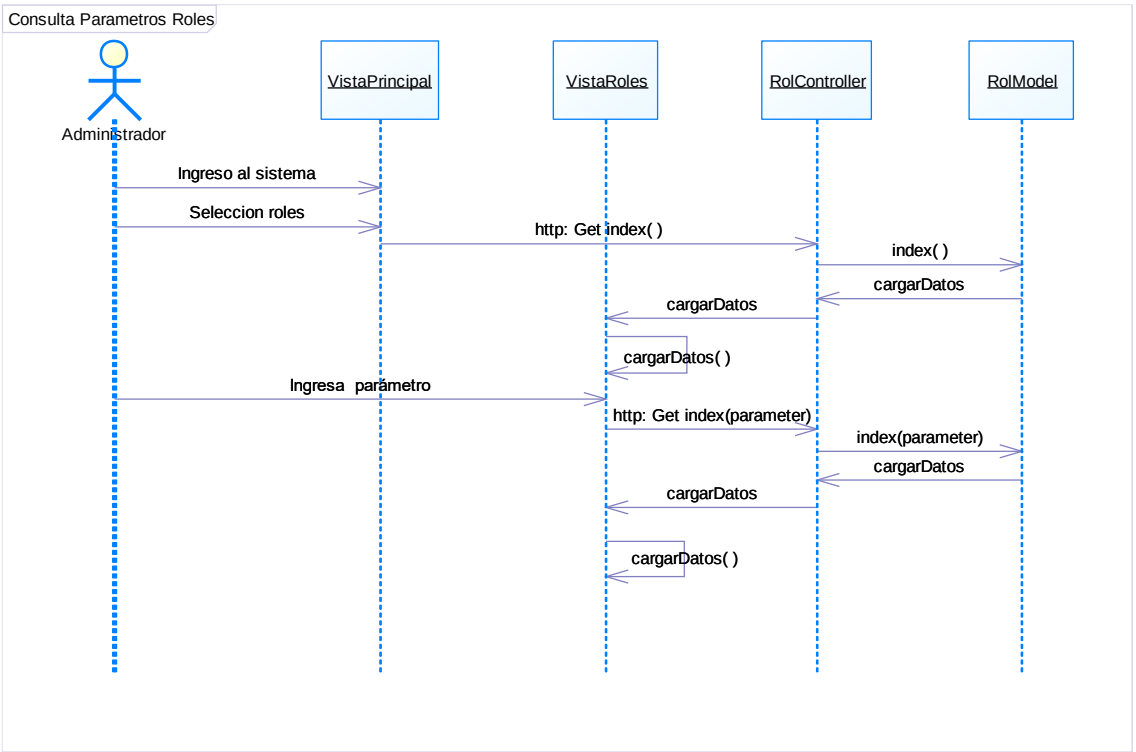
F7.3 Eliminar Rol



F7.4.1 Consulta General



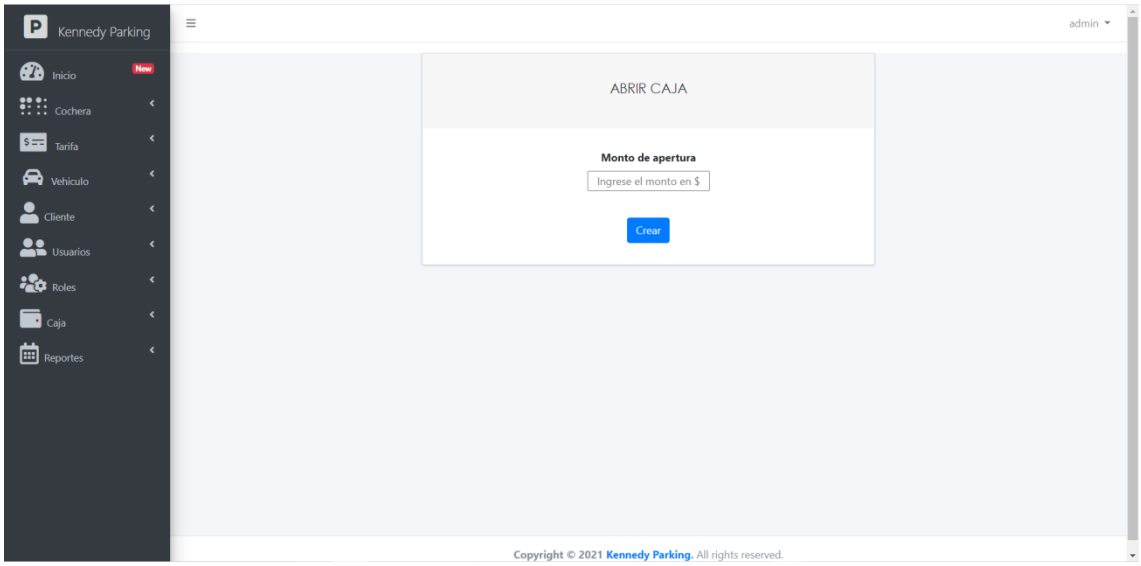
F7.4.2 Consulta Parámetros



Anexo 3

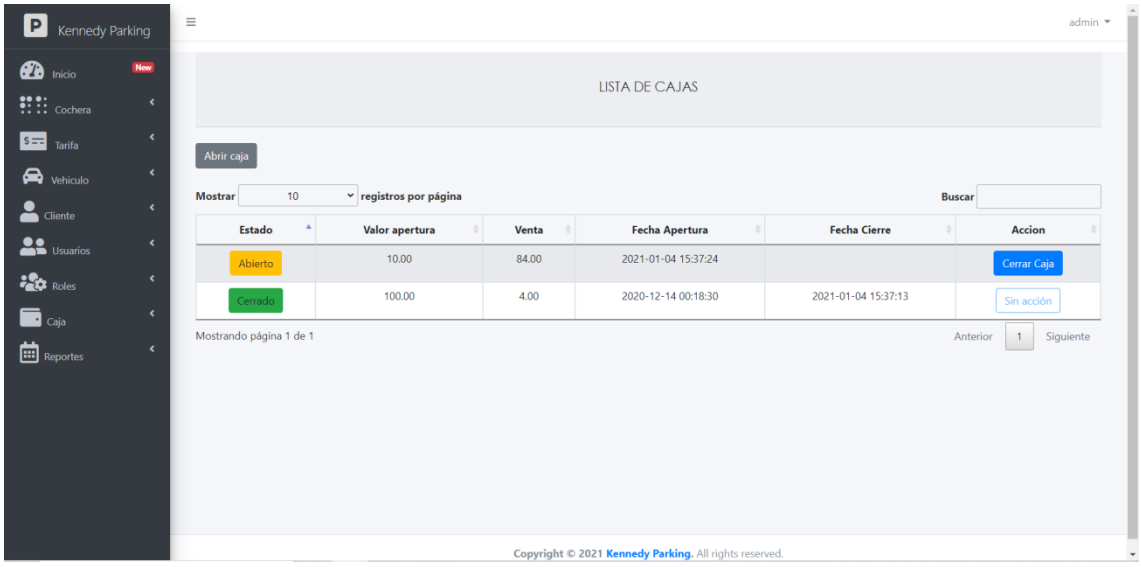
F2 Administración de Cajas

F2.1 Abrir Caja



En esta funcionalidad lo más destacado es la apertura de una nueva caja en el sistema con todos sus atributos

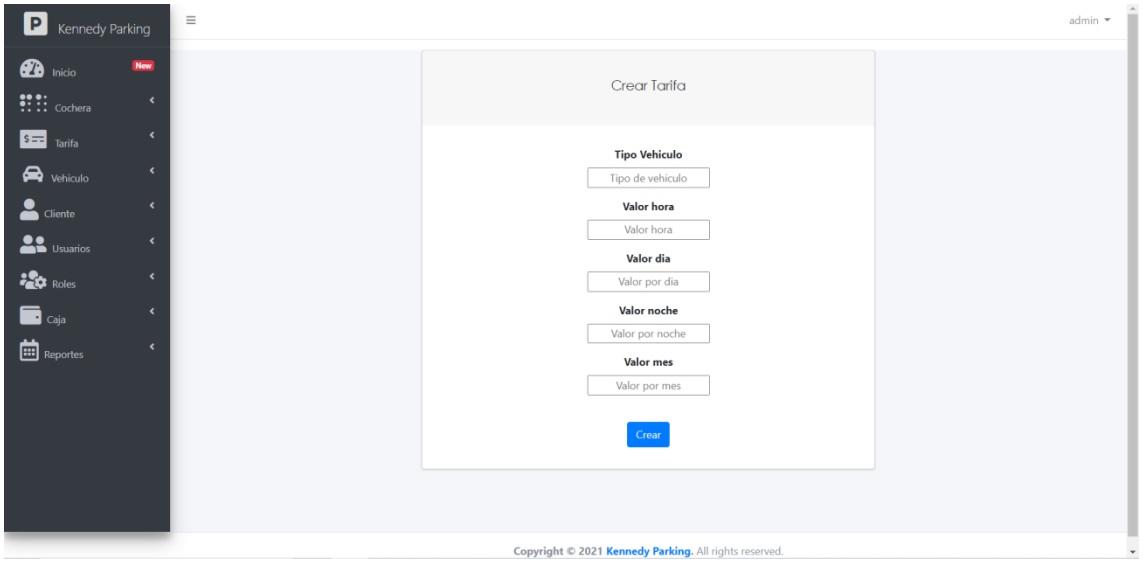
F2.2 Cerrar Caja



En esta funcionalidad lo más destacado el cierre de caja en el sistema.

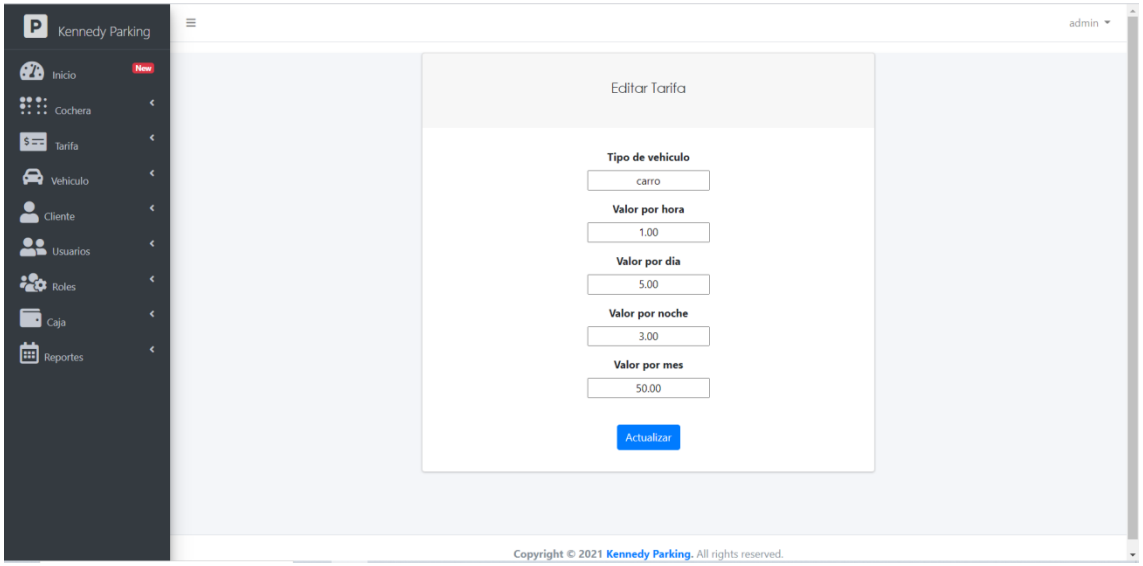
F3 Administración de Tarifa

F3.1 Ingresar Tarifa



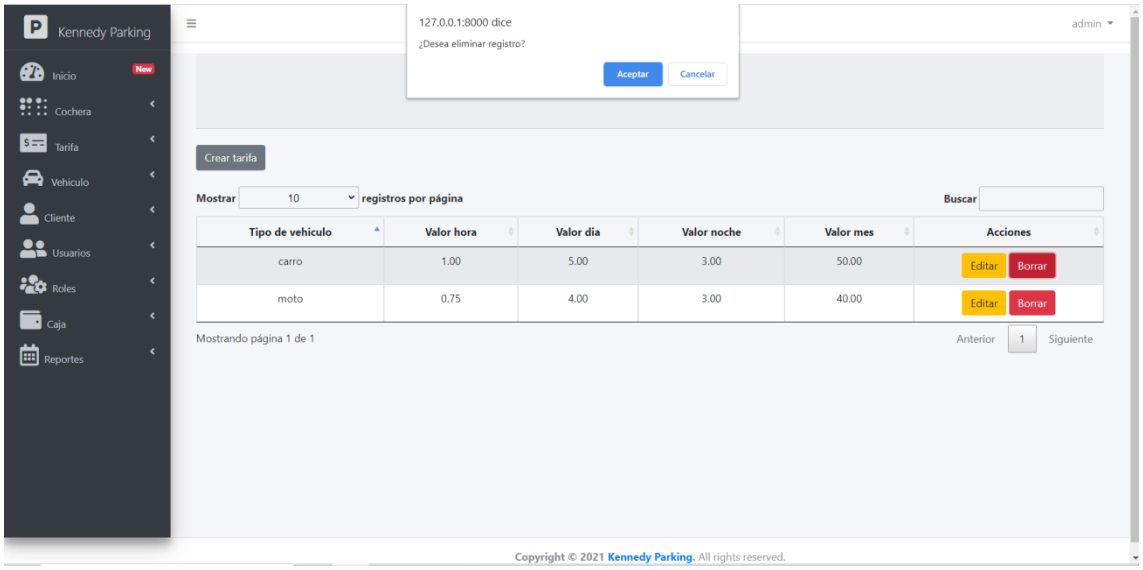
En esta funcionalidad lo más destacado es el ingreso de una nueva tarifa en el sistema con todos sus atributos

F3.2 Modificar Tarifa



En esta funcionalidad lo más destacado la modificación de los atributos un tarifa en el sistema.

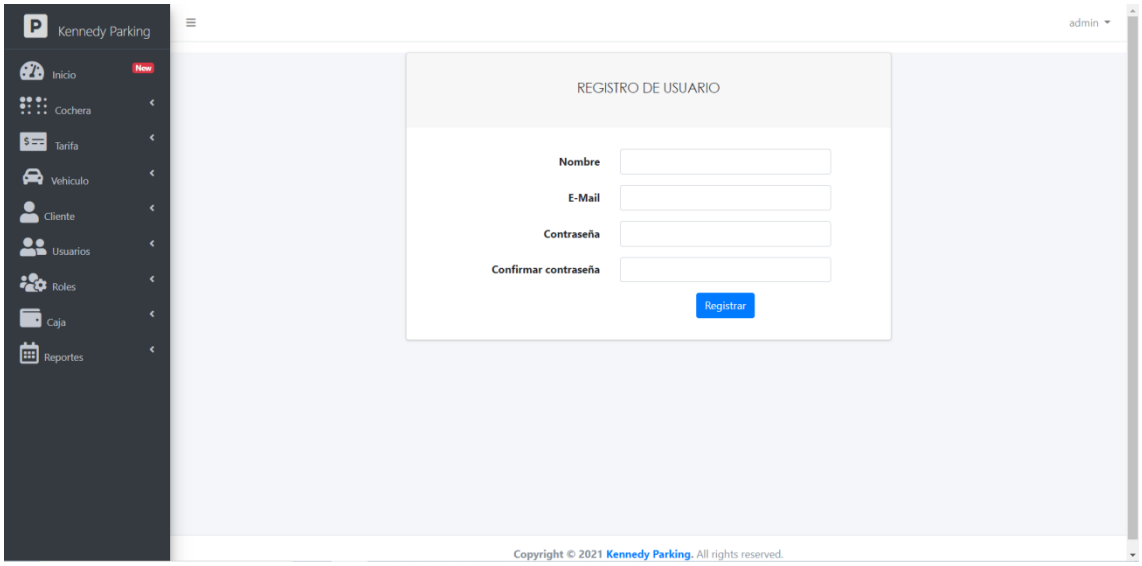
F3.3 Eliminar Tarifa



En esta funcionalidad lo más destacado la eliminación de una tarifa en el sistema.

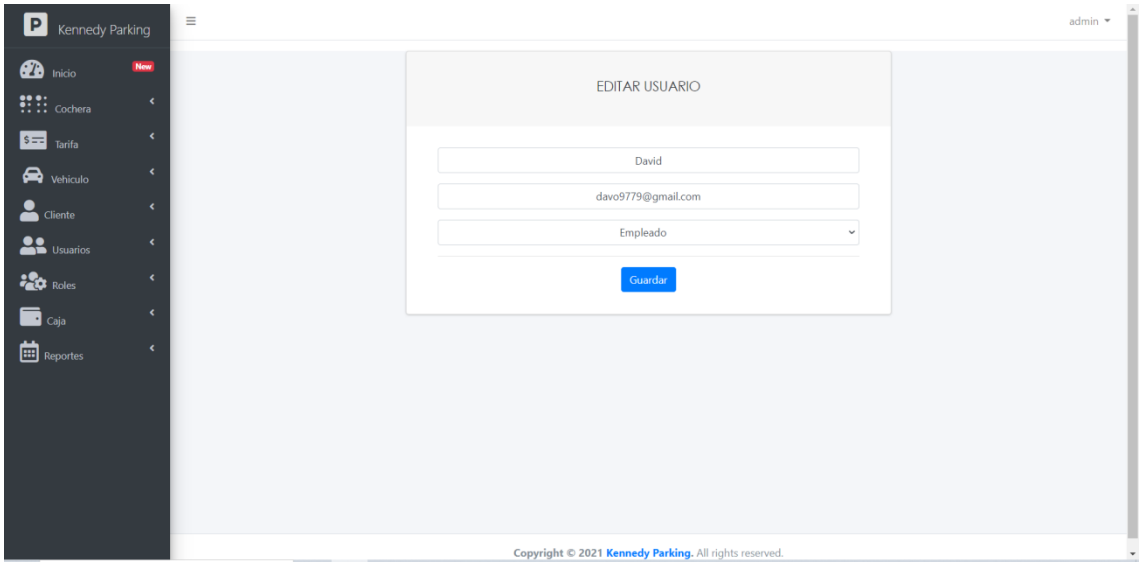
F6 Administración de Usuarios

F6.1 Ingresar Usuario



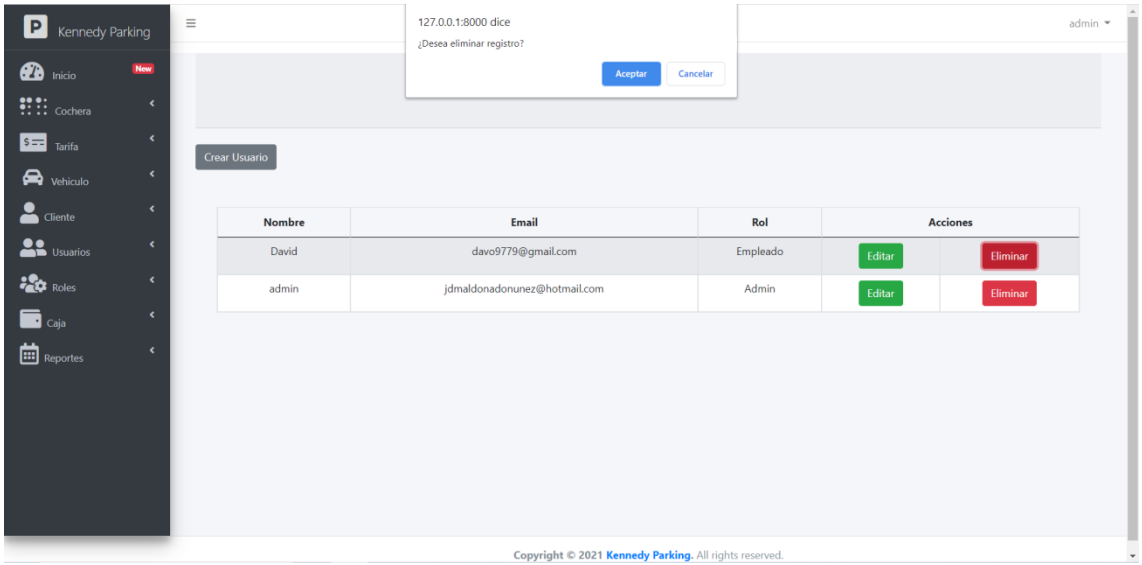
En esta funcionalidad lo más destacado es el ingreso de un nuevo usuario en el sistema con todos sus atributos.

F6.2 Modificar Usuario



En esta funcionalidad lo más destacado la modificación de los atributos un cliente en el sistema.

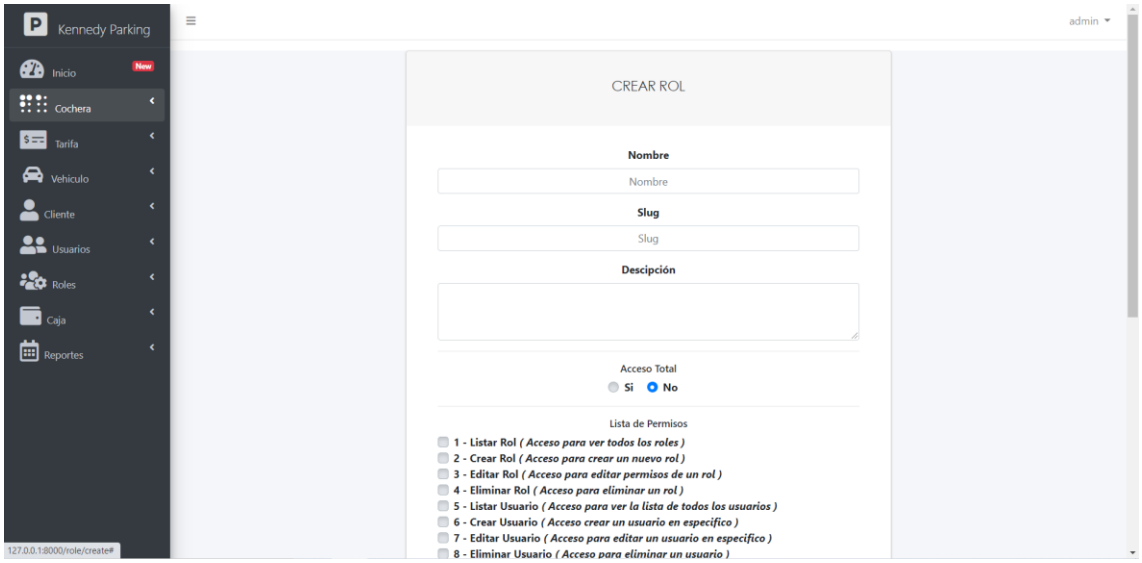
F6.3 Eliminar Usuario



En esta funcionalidad lo más destacado la eliminación de un usuario en el sistema.

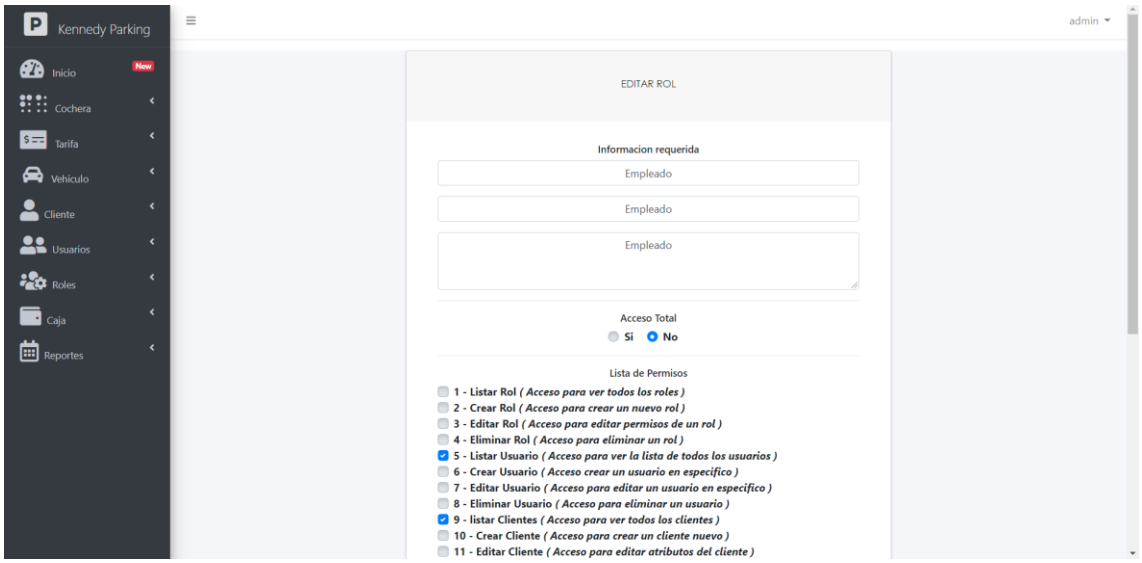
F7 Administración de Roles

F7.1 Ingresar Rol



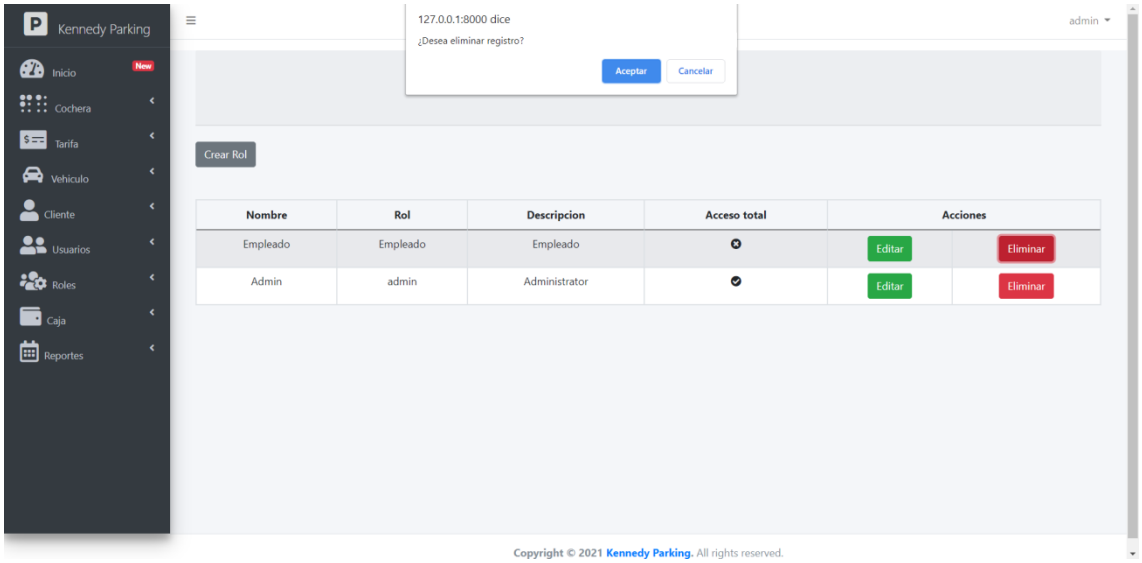
En esta funcionalidad lo más destacado es el ingreso de un nuevo rol en el sistema con todos sus atributos.

F7.2 Modificar Rol



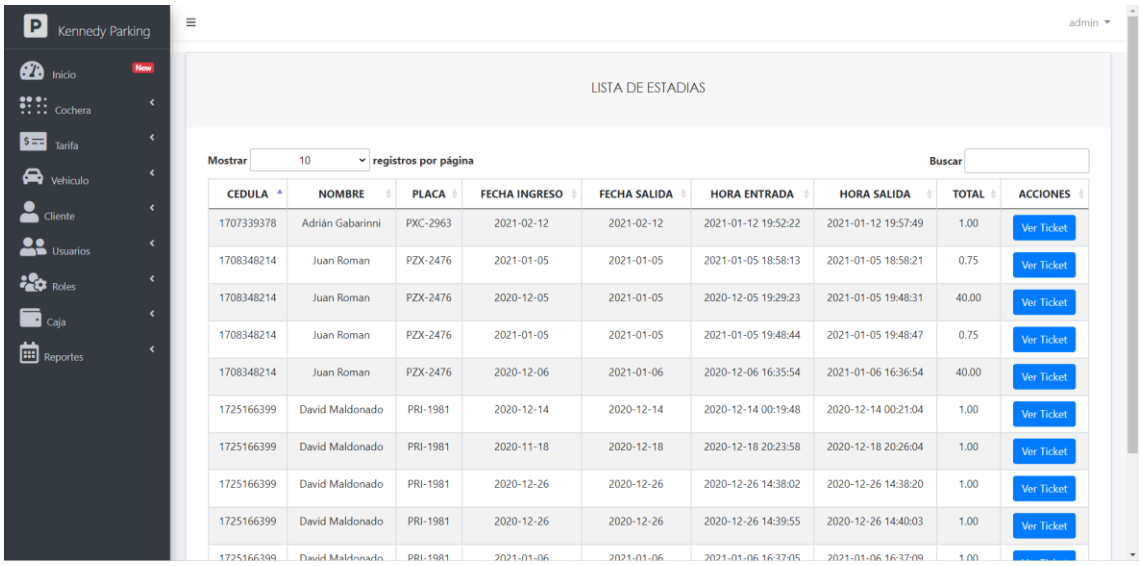
En esta funcionalidad lo más destacado la modificación de los atributos un rol en el sistema.

F7.3 Eliminar Rol



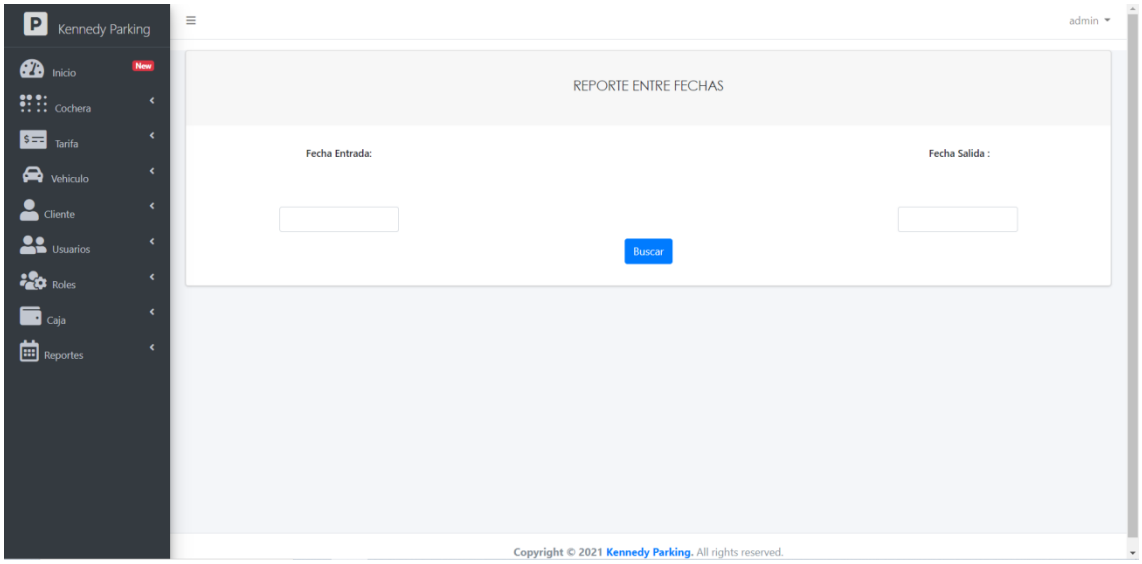
En esta funcionalidad lo más destacado la eliminación de un rol en el sistema.

Pantalla reporte estadías



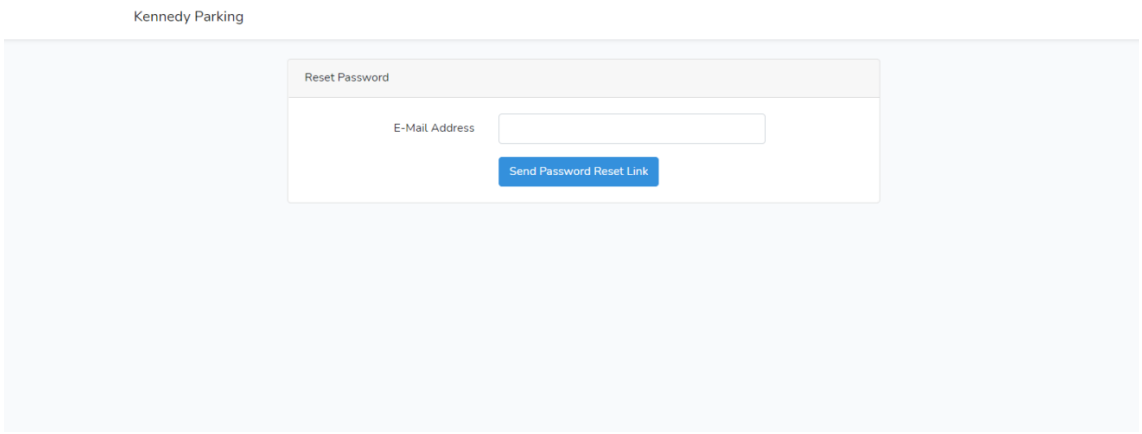
En esta pantalla se aprecia la lista de vehículos que ingresaron a una determinada cochera.

Ventana reporte estadías entre fechas



En esta pantalla se aprecia la lista de vehículos por parámetros que ingresaron a una determinada cochera

Pantalla recuperación contraseña



En esta pantalla se recupera la contraseña de un determinado usuario por su correo.

Anexo 4

Guayaquil, 22 de febrero del 2021

En reunión de trabajo se realizan las pruebas de aceptación del sistema, bajo el siguiente formato:

Funcionalidad	Resultado	¿Cumple?
F0. Ingreso al Sistema	Accede al sistema	Si

Funcionalidad	Resultado	¿Cumple?
F1.1 Ingreso Cliente	Ingresa cliente	Si
F1.2 Modificar Cliente	Modifica cliente	Si
F1.3 Eliminar Cliente	Elimina cliente	Si
F1.4.1 Consulta General	Consulta clientes	Si
F1.4.2 Consulta Parámetros	Consulta cliente	Si

Funcionalidad	Resultado	¿Cumple?
F2.1 Abrir Caja	Abre caja	Si
F2.2 Modificar Caja	Modifica caja	Si
F2.3.1 Consulta General	Consulta cajas	Si
F2.3.2 Consulta Parámetros	Consulta caja	Si

Funcionalidad	Resultado	¿Cumple?
F3.1 Ingreso Tarifa	Ingresa tarifa	Si
F3.2 Modificar Tarifa	Modifica tarifa	Si
F3.3 Eliminar Tarifa	Elimina tarifa	Si
F3.4.1 Consulta General	Consulta tarifas	Si
F3.4.2 Consulta Parámetros	Consulta tarifa	Si

Funcionalidad	Resultado	¿Cumple?
F4.1 Ingreso Vehículo	Ingresa vehículo	Si
F4.2 Modificar Vehículo	Modifica vehículo	Si
F4.3 Eliminar Vehículo	Elimina vehículo	Si
F4.4.1 Consulta General	Consulta vehículos	Si
F4.4.2 Consulta Parámetros	Consulta vehículo	Si

Funcionalidad	Resultado	¿Cumple?
F5.1 Ingreso Cochera	Ingresa cochera	Si
F5.2 Modificar Cochera	Modifica cochera	Si
F5.3 Eliminar Cochera	Elimina cochera	Si
F5.4 Registro Entrada	Entra vehículo en cochera	Si
F5.4.1 Consulta General	Consulta cocheras	Si
F5.4.2 Consulta Parámetros	Consulta cochera	Si

Funcionalidad	Resultado	¿Cumple?
F6.1 Ingreso Usuario	Ingresa usuario	si
F6.2 Modificar Usuario	Modifica usuario	si
F6.3 Eliminar Usuario	Elimina usuario	si
F6.4.1 Consulta General	Consulta usuarios	si
F6.4.2 Consulta Parámetros	Consulta usuario	si

Funcionalidad	Resultado	¿Cumple?
F7.1 Ingreso Rol	Ingresa rol	si
F7.2 Modificar Rol	Modifica rol	si
F7.3 Eliminar Rol	Elimina rol	si
F7.4.1 Consulta General	Consulta roles	si
F7.4.2 Consulta Parámetros	Consulta rol	si

Funcionalidad	Resultado	¿Cumple?
F8.1 Salida Vehiculo	Sale vehiculo de cochera	si
F8.2.1 Consulta General	Consulta asignaciones	si
F8.2.2 Consulta Parámetros	Consulta asignaciones	si

Galo Núñez L.

Dueño establecimiento

David Maldonado N.

Desarrollador

Bibliografía

Alcant, D. (15 de marzo de 2020). *Universidad Alicante*. Obtenido de Modelo Vista Controlador: https://si.ua.es/es/documentacion/asp-net-mvc-3/1-dia/modelo-vista-controlador-mvc.html

AllianceSoftware. (20 de mayo de 2018). *Introduccion a las metodologías de desarrollo de software*. Obtenido de https://www.alliancesoftware.com.au/introduction-software-development-methodologies/

Anónimo. (17 de abril de 2015). *El conquistador*. Obtenido de Cliente servidor: https://www.elconspirador.com/2015/04/17/como-funciona-una-pagina-web/

Anónimo. (18 de enero de 2018). *mrHouston*. Obtenido de Aplicaciones web: https://mrhouston.net/blog/que-son-las-aplicaciones-web/

Anónimo. (21 de mayo de 2020). *Honduras Digital Challange.* Obtenido de Metodológia Scrum una herramienta util para agilizar tus proyectos: https://hondurasdigitalchallenge.com/2020/05/21/metodologia-scrum-una-herramienta-util-para-agilizar-tus-proyectos/

Bravo, D. (12 de septiembre de 2019). *El comercio.* Obtenido de Demanda de parqueos en Quito aumentó: https://www.elcomercio.com/actualidad/demanda-parqueaderos-restriccion-vehicular-quito.html

Cíceri, M. (2018). *Introducción a Laravel Aplicaciones Robustas y a Gran Escala.* Buenos Aires: Six Ediciones.

EcuEed, C. d. (20 de 07 de 20). *Ecured*. Obtenido de Definicion cliente servidor: https://www.ecured.cu/Cliente-Servidor

Edwar, C., & Grace, P. (25 de 04 de 2008). *IEEE*. Obtenido de the Model-View-Controller Pattern: https://ieeexplore.ieee.org/abstract/document/4497770

Escobar, G. (6 de noviembre de 2017). *Make It Real*. Obtenido de Lenguajes compilados e interpretados: https://blog.makeitreal.camp/lenguajes-compilados-e-interpretados/

Ferré Grau, X., & Sánchez Segura, I. (junio de 2017). *Desarrollo Orientado a Objetos con UML.* Obtenido de Facultad de Informática – UPM: https://www.uv.mx/personal/maymendez/files/2011/05/umltotal.pdf

Gomez, Y. (07 de julio de 2020). *Tecnoinformatic*. Obtenido de Arquitectura cliente servidor: https://tecnoinformatic.com/c-informatica-basica/arquitectura-cliente-servidor-en-la-base-de-datos/

Gutierrez, D. (Abril de 2011). Casos de Uso. *Diagramas de Casos de Uso* (pág. 45). Mérida: Universidad de los Andes.

Gutierrez, D. (mayo de 2011). *Diagramas de Secuencia.* Obtenido de Universidad de los Andes: https://www.codecompiling.net/files/slides/UML_clase_06_UML_secuencia.pdf

Gutierrez, D. (12 de julio de 2011). *Modelos de Desarrollo*. Obtenido de Modelo en Cascada:

https://sites.google.com/site/proyectoadpmodelosdedesarrollo/home/modelo-en-cascada

Gutiérrez, J. J. (16 de agosto de 2014). *Departamento de Lenguajes y Sistemas Informáticos.* Obtenido de Que es un framework Web: http://www.lsi.us.es/~javierj/investigacion_ficheros/Framework.pdf

Guzman, A. (04 de Noviembre de 2013). *User Stories*. Obtenido de Scrum Alliance: https://www.scrumalliance.org/community/member-articles/523#:~:text=Una%20historia%20de%20usuario%20describe,son%20mucho%20m%C3%A1s%20que%20eso.

Morales, J. (14 de marzo de 2013). *Zainex.es.* Obtenido de Bases de datos orientadas a objetos y documentos: https://zainex.es/tags/nosql/bases-datos-orientadas-objetos-documentos-ddbb-nosql

Pedruelo, M. R. (Septiembre de 2000). *UESI.* Obtenido de UML. Una Metodología Orientada a Objetos: https://www.esic.edu/documentos/revistas/esicmk/060130_633111_E.pdf

Pérez Valdez, D. (26 de octubre de 2007). *Platzi.* Obtenido de Que es una base de datos: http://www.maestrosdelweb.com/que-son-las-bases-de-datos/

Robledano, A. (24 de septiembre de 2019). *Open webinars*. Obtenido de MySQL características: https://openwebinars.net/blog/que-es-mysql/

Rojas, Á. R. (2009). *Repositorio Académico Universidad de Chile*. Obtenido de Modelos de Base de Datos: http://repositorio.uchile.cl/handle/2250/102126

Sánchez, J. (2004). *Diseño conceptual de base de datos.* Stanford, California: creative commons.

Schwaber, K., & Sutherland, J. (Noviembre de 2017). *La Guía Definitiva de Scrum: Las Reglas del Juego.* Obtenido de Scrum.org: https://www.scrumguides.org/docs/scrumguide/v2017/2017-Scrum-Guide-Spanish-SouthAmerican.pdf

Sommerville, I. (2011). *Ingeniería del software.* Madrid: Pearson.

Telefónica, C. (1 de agosto de 2018). *Acens.* Obtenido de Bases de datos NoSQL. Que son y que tipos nos podemos encontrar: https://www.acens.com/wp-content/images/2014/02/bbdd-nosql-wp-acens.pdf

Universidad de Valladolid, D. d. (2009). *Ingeniería del Software. Tema 4.* Obtenido de CONCEPTO DE: https://www.infor.uva.es/~jvalvarez/docencia/ptema4is1.pdf

Z, C., Lage, F., Pessacq, R., & García Martínez, R. (24 de julio de 2017). *Facultad de Ingeniería. UNLP*. Obtenido de INGENIERIA DE SOFTWARE EDUCATIVO: https://d1wqtxts1xzle7.cloudfront.net/44903590/c-icie99-ingenieriasoftwareeducativo.pdf?1461117921=&response-content-disposition=inline%3B+filename%3DINGENIERIA_DE_SOFTWARE_EDUCATIVO.pdf&Expires=1602780110&Signature=fiH~2mLjwwvXcMEsgDHVkCfBQEASXP99oqHJeUt