UNIVERSIDAD TÉCNICA DE AMBATO

FACULTAD DE INGENIERÍA EN SISTEMAS ELECTRÓNICA E INDUSTRIAL

CARRERA DE INGENIERÍA EN SISTEMAS COMPUTACIONALES E INFORMÁTICOS

TEMA:

Sistema de reservas web para el servicio de transporte puerta a puerta aplicando web services.

Trabajo de Graduación. Modalidad: Proyecto de Investigación, presentado previo la obtención del título de Ingeniero en Ingeniero en Sistemas Computacionales e Informáticos

SUBLÍNEA DE INVESTIGACIÓN: Aplicación Web.

AUTOR: Jaime Alfredo López Romero
TUTOR: Ing. Franklin Mayorga Mg.

Ambato - Ecuador
2018

CERTIFICACIÓN DEL TUTOR

En mi calidad de Tutor del Trabajo de Investigación sobre el Tema:

"Sistema de reservas web para el servicio de transporte puerta a puerta aplicando web services", de la señor, estudiante Jaime Alfredo López Romero de la Carrera de Ingeniería en Sistemas Computacionales e Informáticos, de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial, de la Universidad Técnica de Ambato, considero que el informe investigativo reúne los requisitos suficientes para que continúe con los trámites y consiguiente aprobación de conformidad con el Art. 16 del Capítulo II, del Reglamento de Graduación para Obtener el Título Terminal de Tercer Nivel de la Universidad técnica de Ambato

Ambato, Agosto de 2018

Ing. Franklin Mayorga Mg.

EL TUTOR

AUTORÍA DEL TRABAJO

El presente trabajo de investigación titulado: Sistema de reservas web para el servicio de transporte puerta a puerta aplicando web services. Es absolutamente original, auténtico y personal, en tal virtud, el contenido, efectos legales y académicos que se desprenden del mismo son de exclusiva responsabilidad del autor.

Ambato, Agosto de 2018

Jaime Alfredo López Romero

CC: 1804677951

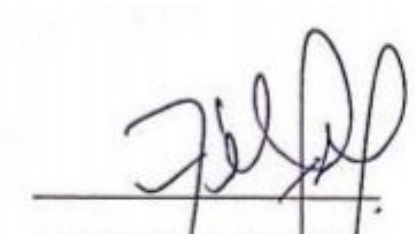
APROBACIÓN DEL TRIBUNAL DE GRADO

La Comisión Calificadora del presente trabajo conformada por los señores docentes Ing. David Guevara Mg. e Ing. Félix Fernández PhD., revisó y aprobó el Informe Final del trabajo de graduación titulado Sistema de reservas web para el servicio de transporte puerta a puerta aplicando web services, presentado por el señor Jaime Alfredo López Romero de acuerdo al Art. 17 del Reglamento de Graduación para obtener el título Terminal de tercer nivel de la Universidad Técnica de Ambato.

Ing. Elsa Pilar Urrutia Urritia Mg.

PRESIDENTE DEL TRIBUNAL

Ing. David Guevara Mg.
DOCENTE CALIFICADOR



Ing. Félix Fernández PhD.
DOCENTE CALIFICADOR

DERECHOS DE AUTOR

Autorizo a la Universidad Técnica de Ambato, para que haga uso de este trabajo de titulación sobre el tema: "SISTEMA DE RESERVAS WEB PARA EL SERVICIO DE TRANSPORTE PUERTA A PUERTA APLICANDO WEB SERVICES" como un documento disponible para su lectura, consulta y procesos de investigación. Cedo los Derechos de mi Trabajo de Titulación, con fines de difusión pública, además autorizo su reproducción dentro de las regulaciones de la Universidad, respetando mis derechos de autor y no se utilice con fines de lucro.

Jaime Alfredo López Romero

C.C. 1804677951

AUTOR

DEDICATORIA

Este trabajo va dedicado a todas las personas cercanas a mi, que siempre estuvieron apoyándome para que continúe con mis estudios.

A mi madre que siempre me daba su apoyo moral en esta etapa de estudio haciendo que llegue al final sin importar las circunstancias que se hayan presentado o que tan difíciles hayan sido, superándolas con las mejores decisiones.

A mi padre quien ha sido el pilar fundamental gracias a su apoyo incondicional logre uno de mis objetivos que es culminar la carrera, y para que así tenga mejores oportunidades en el mundo laboral.

A mi esposa que su apoyo fue el que me mantuvo firmes con la visión de llegar a culminar mi etapa estudiantil en la universidad alentándome confianza en mi mismo.

En especial a mi hijo, porque el fue el mayor motivo para perseverar en todo momento, levantarme cada día y ser mejor en todas las cosas que hago.

Jaime López

AGRADECIMIENTO

Expreso mi mas sincero agradecimiento a toda mi familia por su apoyo en todo momento en el transcurso de toda esta etapa de formación profesional.

A mis compañeros y amigos con quienes he podido confiar, por su apoyo en los momentos en los que he necesitado, por todos los momentos vividos en estos años en los que he estado en la universidad y poder hacer grandes amistades que durarán toda la vida.

A los profesores e ingenieros que guiaron con la mejor enseñanza para poder sobresalir en el ámbito profesional.

A la Universidad Técnica de Ambato, en especial a la facultad de Ingeniería en Sistemas Electrónica e Industrial, que me ha permitido formarme profesionalmente.

Jaime López

ÍNDICE

APROBACIÓN DEL TUTOR ii

AUTORÍA iii

APROBACIÓN COMISIÓN CALIFICADORA iv

DEDICATORIA v

AGRADECIMIENTO vi

RESUMEN xv

SUMMARY xvi

INTRODUCCIÓN xvii

CAPÍTULO 1 El problema 1

1.1 Tema de Investigación 1

1.2 Planteamiento del problema 1

1.3 Delimitación 2

1.3.1 De contenidos 2

1.3.2 Espacial 2

1.3.3 Temporal 2

1.4 Justificación 2

1.5 Objetivos 3

1.5.1 General 3

1.5.2 Específicos 3

CAPÍTULO 2 Marco Teórico 4

2.1 Antecedentes Investigativos 4

2.2 Fundamentación teórica 4

2.2.1 Sistema web 4

2.2.2 Sistema de reservas 5

2.2.3 Framework 5
2.2.4 Entity Framework 10
2.2.5 Servicio web 10
2.2.6 OWIN 16
2.2.7 Token de seguridad 17
2.2.8 RUP 17
2.2.9 Metodología XP 21
2.2.10 Metodología 3p 21
2.2.11 Retrofit 22
2.2.12 Android 23
2.3 Propuesta de Solución 23

CAPÍTULO 3 Metodología **24**
3.1 Modalidad de la investigación 24
3.2 Recolección de información 24
3.3 Procesamiento y análisis de datos 24
3.4 Desarrollo del Proyecto 24

CAPÍTULO 4 Desarrollo de la propuesta. **26**
4.1 Introducción. 26
4.2 Requisitos 29
4.2.1 Requisitos funcionales y no funcionales. 29
4.2.2 Visión. 30
4.2.3 Glosario. 30
4.2.4 Modelado del negocio. 32
4.3 Análisis y diseño. 45
4.3.1 Modelo lógico. 45
4.3.2 Diccionario de datos. 45
4.3.3 Interfaz plantilla máster. 50
4.3.4 Interfaz inicio de sesión. 51
4.3.5 Formularios de mantenimiento. 52
4.3.6 Área de visualización y búsqueda de información. 53
4.3.7 Arquitectura. 54
4.4 Implementación. 54
4.4.1 Creación de los servicios web. 54
4.5 Pruebas de funcionamiento. 82
4.5.1 Caso de prueba registrar usuario. 82
4.5.2 Caso de prueba Ingreso al sistema. 82

4.5.3 Caso de prueba Roles. 83
4.5.4 Caso de prueba Asignar Rol. 83
4.5.5 Caso de prueba Vehículos. 84
4.5.6 Caso de prueba Ciudad. 85
4.5.7 Caso de prueba Horarios. 85
4.5.8 Caso de prueba Clientes. 86
4.5.9 Caso de prueba Reserva. 86
4.5.10 Caso de prueba Asignar Reserva. 90
4.5.11 Caso de prueba Reporte de viajes. 91
4.5.12 Caso de prueba Reporte de clientes frecuentes. 91
4.5.13 Caso de prueba Vehículos. 92
4.5.14 Caso de prueba asignación de clientes. 92
4.6 Pruebas de concurrencia de usuarios. 93

CAPÍTULO 5 Conclusiones y Recomendaciones. 102
5.1 Conclusiones. 102
5.2 Recomendaciones. 103

Bibliografia 104

ANEXOS 108

ÍNDICE DE TABLAS

1 Comparación de licencias. 6
2 Comparación de soporte. 7
3 Comparación de plataformas. 7
4 Comparación de ventajas. 8
5 Comparación de desventajas. 9
6 Resultados selección de Framework 9
7 Cuadro comparativo entre los métodos de SOAP y REST 12

8 Ficha Cliente 26
9 Ficha vehículo. 26
10 Ficha reserva. 27
12 Entrevista. 28
13 Observación del proceso de reserva 29
14 Sentencia que define el problema. 30
15 Descripción del proceso de una reserva de viaje puerta a puerta. . 33
16 Gestionar roles 34
17 Gestionar usuarios 34
18 Gestionar horarios y precios 35
19 Gestionar vehículos 35
20 Emitir reportes. 35
21 Gestionar reservas del asistente. 36
22 Emitir reportes. 37
23 Gestionar reservas chofer. 38
24 Gestionar vehículos. 38
25 Gestión de reservas 39
26 Emitir reportes. 39
27 Diccionario de datos entidad Provincia. 46
28 Diccionario de datos entidad Cantón. 46
29 Diccionario de datos entidad ciudad 46
30 Diccionario de datos entidad horario. 46
31 Diccionario de datos entidad AspNetRoles. 46

32 Diccionario de datos entidad AspNetUsers. 47
33 Diccionario de datos entidad AspNetuserRoles 47
34 Diccionario de datos entidad vehículo. 48
35 Diccionario de datos entidad cliente. 48
36 Diccionario de datos entidad reserva. 49
37 Diccionario de datos entidad Asignar_Reserva 49
38 Prueba registrar un usuario. 82
39 Prueba ingreso al sistema. 82
40 Prueba gestión de roles. 83
41 Prueba asignación de roles. 83
42 Prueba gestión de vehículos 84
43 Prueba gestión de ciudades 85
44 Prueba gestión de horarios 85
45 Prueba gestión de clientes. 86
46 Prueba gestión de Reservas posibilidad 1. 87
47 Prueba gestión de Reservas posibilidad 2. 89
48 Prueba gestión de asignación de reservas. 90
49 Prueba reporte de viajes. 91
50 Prueba reporte de viajes. 91
51 Prueba gestión de vehículos con rol de chofer. 92
52 Prueba gestión de vehículos con rol de chofer. 92

ÍNDICE DE FIGURAS

1 Entity Framework 10
2 Request servicio web rest 13
3 Ejemplo request usando el software Fiddler. 13
4 Ejemplo request programando en c#. 14
5 Response servicio web rest. 14
6 Ejemplo Response servicio web rest usando el software Fiddler. . . 15
7 Cualquier usuario. 15
8 Usuarios autorizados. 16
9 Usuario o roles específicos autorizados. 16
10 Funcionamiento de un token de seguridad. 17
11 Estructura del proceso RUP 19
12 Arquitectura de retrofit. 22

13 Modelado del proceso de una reserva de viaje puerta a puerta. . . 32
14 Caso de uso - Administrador 33
15 Caso de uso - Asistente 36
16 Caso de uso - Chofer 37
17 Caso de uso - Cliente 39
18 Diagrama de secuencia Administrador. 41
19 Diagrama secuencia Asistente 42
20 Diagrama secuencia Chofer. 43
21 Diagrama secuencia Cliente 44
22 Modelo lógico de la base de datos. 45
23 Página maestra. 50
24 Página de inicio de sesión. 51
25 Formularios de mantenimiento. 52
26 Área de visualización y búsqueda de información. 53
27 Arquitectura del sistema. 54
28 Creación de un nuevo proyecto. 54
29 Seleccionar Aplicación web ASP.NET. 55
30 Seleccionar Web API MVC. 56

31 Agregar nuevo elemento 56
32 Agregar nuevo elemento ADO.NET Entity Data Model 57
33 Seleccionar EF Designer desde base de datos 58
34 Configurar conexión a Base de Datos. 59
35 Guardar cadena de conexión. 60
36 Seleccionar los objetos para EF Designer 61
37 Diagrama generado con EF Designer 62
38 Agregar un nuevo elemento con Scaffolding. 62
39 Seleccionar Controlador de Web API 2 con acciones que usan Entity Framework. 63
40 Agregar controlador para entidad generada con EF Designer. . . . 63
41 Clase generada con los métodos del servicio web. 64
42 Verificación del funcionamiento web. 64
43 Pruebas del funcionamiento del servicio web Insert con Fiddler . . 65
44 Resultado del servicio web getAll. 65
45 Pruba del servicio web getById 66
46 Prueba Update 66
47 Prueba servicio web Delete. 66
48 Creación de la clase del servicio web. 67
49 Serializar los datos. 68
50 Request del servicio web, ... 68
51 Response del Request. 69
52 Mostrar los datos en un grid view 69
53 Actualizar los datos. 70
54 Eliminar un dato. 70
55 Prueba conexión a IIS 71
56 Clase token. 72
57 Clase vehículo. 73
58 Clase tipo interfaz parte superior. 74
59 Clase tipo interfaz parte inferior 75
60 Clase adaptador. 75
61 Método para obtener el token de sesión. 76
62 Método para obtener listado de vehículos. 77
63 Método para insertar un vehículo. 77
64 Método actualizar. 78
65 Método eliminar. 78
66 Pantalla inicial. 79

67 Consumiendo el servicio web. 80
68 Prueba con usuario incorrecto. 81
69 Configuración usuarios. 93
70 Configuración por defecto. 94
71 Obtener el token de sesión. 94
72 Insertar vehículo. 95
73 Seleccionar vehículos. 95
74 Resultados obtenidos 96
75 Resumen de los resultados. 96
76 Página de Inicio. 97
77 Interfaz menú. 98
78 Formulario vehículos. 99
79 Formulario ciudad parte superior. 100
80 Formulario ciudad parte inferior. 101

RESUMEN

Con el fin de mejorar los procesos para realizar una reserva a la hora de viajar en un servicio de transporte puerta a puerta, esta investigación propone un sistema web para facilitar el proceso de reservas a los usuarios.
Con la ayuda del proceso RUP y UML como metodología se a logrado llevar a cabo el desarrollo de la aplicación, desde el análisis, creación de la base de datos, creación de los servicios web y desarrollo del sitio web de una manera eficaz.
La aplicación web permite que un usuario reservar eligiendo su destino, marcando la ubicación de origen y destino en un mapa, un horario y obteniendo la información del vehículo que le llevará a su destino.
El diseño de la aplicación con diseño amigable, responsivo e intuitivo para el usuario utilizando herramientas como boostrap, Asp.Net c#, Entity Framework, y servicios web tipo Rest, dando como resultado una aplicación ligera e intuitiva.

SUMMARY

In order to improve the processes to make a reservation when traveling in a door-to-door transport service, this research proposes a web system to facilitate the reservation process to users.

With the help of the process RUP and UML make a methodology it has been possible to carry out the development of the application, from the analysis, creation of the database, creation of the web services and development of the website in an efficient way.

The web application allows a user to reserve by choosing their destination, marking the location of origin and destination on a map, a schedule and obtaining the information of the vehicle that will take you to your destination.

The design of the application with friendly, responsive and intuitive design for the user using tools such as boostrap, Asp.Net c #, Entity Framework, and web services type Rest, resulting in a light and intuitive application.

INTRODUCCIÓN

En la actualidad el uso de las TICS es una necesidad para realizar cualquier gestión vía Internet, por lo tanto se recomienda automatizar procesos en los cuales se puedan poner a disposición en Internet, es así el caso que se recomienda implementar un sistema de reservas web para los servicios de transporte puerta a puerta, el proyecto se realizó por la necesidad de las personas que suelen ser usuarios de este tipo de transporte no tienen a su disposición un sistema web que le permita realizar este tipo de reservas.

Con la ayuda de la el proceso RUP y UML como metodología, permite desarrollar fácilmente el sistema, la metodología se basa en tres características que son el modelado del negocio, el mismo que con diagramas de caso de uso que facilitan el entendimiento de los procesos que realizan para una reserva, la segunda característica se basa en la arquitectura, en este caso permite el diseño de la implicación web y la manera como se consume los servicios web que estos a su vez hace uso de Entity Framework para mapear las tablas de la base de datos y facilitar el desarrollo de los servicios web tipo rest, y la última característica es el proceso iterativo incremental el cual ayuda a dividir el proyecto en proyectos de menor tamaño.

La aplicación web permitirá que los usuarios que hacen el uso de este tipo de transporte y tengan a su alcance un computador o un teléfono puedan realizar una reserva fácilmente y las empresas que ofrecen este servicio tengan un correcto control de clientes y choferes.

CAPÍTULO 1

El problema

1.1. Tema de Investigación

Sistema de reservas web para el servicio de transporte puerta a puerta aplicando web services.

1.2. Planteamiento del problema

Las reservaciones de viajes en la actualidad son muy importantes para las personas que frecuentemente están viajando por distintos países del mundo, y cada vez existe la necesidad de hacer las cosas fácil y rápido, todo a través de un computador o un teléfono inteligente, las personas a la hora de realizar alguna actividad el primer paso es buscar información en Internet, y si un negocio o empresa quiere ser encontrado fácilmente debe tener una página web con toda la información necesaria de los servicios o productos que ofrece para que puedan ponerse en contacto. En el país existen varias formas de trasladarse de un lugar a otro, como son: cooperativas de buses interprovinciales, cantonales, urbanos, taxis camionetas, taxis exprés, etc. Los usuarios muchas veces sienten molestia a la hora de viajar especialmente fuera de la provincia ya que tienen que hacer colas para comprar los boletos de viaje, esto se debe a que la mayoría de cooperativas no tienen un sitio web en donde el usuario pueda hacer una reserva de viaje en línea, o consultar el horario del recorrido y pueda acercarse al terminar ya cuando la unidad tenga que salir a su destino. En la zona centro del país las empresas que ofrecen el servicio de transporte puerta a puerta la mayoría no tienen a la disposición de los clientes una aplicación web, y mucho menos con la funcionalidad que puedan hacer reservas online, en consecuencia los usuarios se sienten desconformes por el servicio, y deben recurrir al proceso tradicional para obtener un cupo de viaje haciendo llamadas telefónicas que muchas veces se presenta inconvenientes de cobertura móvil, el teléfono está ocupado, batería descargada, etc., además esto implica gasto económico debido a que tiene que dar explicaciones de la fecha, hora, lugar, y alguna referencia para que el chofer de la unidad le pueda llegar al lugar sin problemas.

1.3. Delimitación

1.3.1. De contenidos

Área Académica: Software
Línea de investigación: Desarrollo de Software
Sublíneas de investigación: Aplicación web

1.3.2. Espacial

La aplicación web no está delimitado para un área geográfica específica, el objetivo es vender la aplicación web a las empresas que ofrecen este servicio, por lo que la delimitación espacial del proyecto es a nivel Regional.

1.3.3. Temporal

La presente investigación se desarrollará en el periodo Abril 2017 - Septiembre 2017

1.4. Justificación

Un elemento muy importante hoy en día, para desarrollar aplicaciones que interactúen con diferentes servicios y plataformas es un servicio web, las aplicaciones como Facebook, WhatsApp, twitter, etc. Son ejemplos claros que usan este tipo de tecnologías para el desarrollo de sus aplicaciones. Al usar este tipo de tecnología en el sitio web, se tiene la principal ventaja que, si se quiere implementar una aplicación móvil, solo se deberá consumir estos servicios. El presente proyecto busca mejorar los servicios que prestan las empresas de transportes puerta a puerta la manera en que estas guardan la información y en sí los procesos más frecuentes como: registro de clientes, registro de reservaciones, consultas de asientos disponibles, obtención de informes de los choferes. Para mejorar el problema de información se realizará un sistema web consumiendo servicios web que guardará los datos de una manera adecuada, permitirá una rápida búsqueda; además de poseer un mejor control de clientes frecuentes para ofrecer promociones y descuentos.

1.5. Objetivos

1.5.1. General

Desarrollar un sistema de reservas web para el servicio de transporte puerta a puerta aplicando servicios web.

1.5.2. Específicos

- Analizar los procesos que se realizan para hacer una reserva de viaje con los servicios de transporte puerta a puerta.
- Realizar un estudio de los principales Frameworks existentes en el mercado tecnológico para el desarrollo de aplicaciones web y servicios web.
- Comparar diferentes tecnologías de servicios web para usar en el desarrollo de la aplicación web.
- Seleccionar una metodología ágil aplicada al campo del desarrollo de aplicaciones web que facilite el desarrollo del proyecto planteado.
- Implementar el sistema de reservas web para el servicio de transporte puerta a puerta aplicando servicios web.

CAPÍTULO 2

Marco Teórico

2.1. Antecedentes Investigativos

Realizando un estudio sobre las aplicaciones web de reservas se encontraron las siguientes:
Juan Carlos Salán Villena realizó un sistema de reservas para una cooperativa de transporte, Con la implementación de la aplicación web se tiene una gran ventaja de dar un mejor servicio a 24 horas a los clientes. La aplicación Web brinda soluciones a los procesos de gestión de rutas, poniendo a disposición una mayor facilidad de realizar los procesos de la cooperativa. La implementación de la aplicación generará un control exhaustivo, acorde a las funciones y lineamientos que necesita la cooperativa, partiendo de la necesidad de información de dicha cooperativa. De las encuestas realizadas al personal administrativo, la mayor parte de ellos estuvieron de acuerdo de que era necesario el desarrollo de una aplicación Web para así mejorar en los procesos de gestión de rutas [1].
José Cedamano, Marwin Cedamano, Edison Lascano y Jenny Ruiz, que desarrollaron un sistema web, para el control de rutas, encomiendas, reservación y venta de tickets para la cooperativa de transportes "pullman Carchi", concluyen que con esto se garantiza más aún que el software en cuestión sea una solución a la problemática de facturación, reservación de pasajes, control de rutas, control de clientes y forma de pago, y que sea desarrollado con estándares de calidad adecuados. La necesidad de mejorar procesos en las cooperativas es importante. La tecnología debe apoyar estos cambios, optimizarlos, para que sean ejecutados con rapidez, dando como resultado el aumento en la eficiencia de la empresa, lo que significa menos gasto en tiempo y recursos, obteniendo más rentabilidad y una mejor atención al cliente [2].

2.2. Fundamentación teórica

2.2.1. Sistema web

Es un sistema multimedia de acceso a diferente información distribuida por toda la red en forma de hipertexto, el sistema web apareció a principios de los noventa

con el objetivo de facilitar la distribución de la información, en el CERN (Centre Europeen de Recherche Nucleaaire), en definitiva trataban de que la información esté disponible para los grupos de investigación dispersos, para que cada terminal pueda acceder de una forma rápida a los documentos en una red hipertextual. Un año más tarde la red creada en la CERN se abre al Internet con la aparición del navegador Mosaic [3].
Un Sistema web o también conocido como aplicación web, está alojado en un hosting, o en una red local; la aplicación web utiliza bases de datos para poder procesar las peticiones de los usuarios [4].

2.2.2. Sistema de reservas

Según Eugenio Fernández e Ignacio Olmeda, un sistema de reservas debe contener recursos necesarios que dependerán de la aplicación final que se pretenda crear, y se clasifican en los siguientes:

- Infraestructura: Periféricos de red, Conexiones, Personal de servicio técnico, Servidores, Equipos terminales.
- Software para la edición: Todo el software necesario para diseñar y programar un sitio web, gestores de bases de datos, y aplicaciones para la edición multimedia.
- Fuentes de información: Información que se presentará en la aplicación web para disposición del usuario.

Además, para abordar la implementación y puesta en marcha de estos sistemas se requiere una selección apropiada de los servicios ofrecidos por Internet que permiten una comunicación entre clientes [5].

2.2.3. Framework

Son herramientas que facilitan su desarrollo y su implementación de aplicaciones web, los frameworks permiten al desarrollador usar modelos o patrones para el desarrollo.
Existen diferentes frameworks dependiendo las necesidades del sitio web y además dependiendo el estilo de desarrollo algunas características de los frameworks son:

- Seguridad.
- Robustez.

- Facilidad de uso [6].

Para seleccionar el lenguaje de programación y a su vez el framework se usará la metodología funcional de benchmarking.

- Planificación.

  - Se va a medir características como:

    - Licencia
    - Soporte y documentación.
    - Plataformas
    - Ventajas
    - Desventajas

  - Se va a medir a los principales frameworks:

    - .NET (ASP.NET)
    - PHP (LARAVEL)
    - JAVA (SPRING)

- Resultado.

  - Se va ha calificar entre 1 y 2 puntos cada característica.

- Datos.

Framework		
Licencia		
ASP.NET	LARAVEL	SPRING
No necesitan licencias, se necesita de licencias para el IDE de desarrollo, pero se puede usar sin licencia en su versión express , si no es para empresa grande.	Es software libre por lo tanto no se necesita de una licencia.	Al igual que LARAVEL es software libre.

Tabla 1: Comparación de licencias.

Framework		
Soporte y documentación.		
ASP.NET	LARAVEL	SPRING
Cuenta con variedad de información en páginas oficiales de los creadores y ayuda en foros de las misma que facilitan la programación.	Existe información en paginas y foros, cómo fue creado en el 2011 aún no se tiene información mas puntual.	Carece de información en páginas oficiales que ayuden a solucionar temas puntuales.

Tabla 2: Comparación de soporte.

Framework		
Plataformas.		
ASP.NET	LARAVEL	SPRING
Windows Linux	Windows Linux Unix	Windows Linux Unix

Tabla 3: Comparación de plataformas.

Framework	
Ventajas	
ASP.NET	- Admiten cualquier lenguaje compatible con Common Language Runtime de .NET, incluidos Microsoft Visual Basic, Microsoft Visual C#, Microsoft J# y Microsoft JScript.NET. - Facilidad en el mantenimiento y programación ya que separa la parte del cliente y la parte del servidor - Compatible con bases de datos como SqlServer, Mysql, PostgreSql, Oracle, Firebird, etc. - Es orientado a objetos. - Permite crear servicios web tipo Soap, Rest, y Wcf. - Existen varios componentes que permiten crear servicios web sin escribir nada de código.
LARAVEL	- Compatible con php 5 y php 7. - Utiliza patrón de desarrollo MVC - Crea código php simple, reduciendo la cantidad de código. - Utiliza un ORM (Object Relational Mapper) que permite mapear los objetos de la Base de Datos. - Uso de layouts con blade, esto es igual a usar una página maestra. - Permite crear servicios web tipo Soap y Rest.
SPRING	- Todo se puede configurar con archivos xml, como puede ser conexión a BDD, login, acciones a ejecutar. - Utiliza el patrón de desarrollo MVC - Existen adaptadores para que realicen funciones, que son fáciles integrarles. - Permite crear servicios web tipo Soap y Rest

Tabla 4: Comparación de ventajas.

Framework	
Desventajas	
ASP.NET	-Muchos hostings no lo soportan por su alto coste.
LARAVEL	- No es compatible con la mayoría de las bases de datos. - Se puede utilizar solamente plantillas blade y fabric. - Como es más reciente que los otros frameworks no es muy utilizado para el desarrollo, porque no tiene mucha información. - La creación de los servicios web se hace escribiendo código.
SPRING	- Se debe hacer una configuración por cada servicio. - Escasos libros y documentación sobre sprint. - Es inevitable utilizar muchos archivos XML. - Es necesario escribir mucho código en los archivos JSP a pesar de que genera tags automáticamente. - La creación de los weeb services se hace escribiendo código.

Tabla 5: Comparación de desventajas.

- Análisis.

  Para la selección del Framework de desarrollo de aplicaciones web y servicios web, el que tiene mayor calificación es ASP.NET.

	ASP.NET	LARAVEL	SPRING
DETALLE	Puntos	Puntos	Puntos
Licencia	2	2	2
Soporte y documentación	2	1	1
Plataformas	1	2	2
ventajas	2	2	1
Desventajas	2	1	1
Total	9	8	7

Tabla 6: Resultados selección de Framework

- Acción.

  Se va a desarrollar en ASP.NET los servicios web y la aplicación web.

2.2.4. Entity Framework

Es un ORM (Object Relational Mapping) que reduce el tiempo en programación de los métodos básicos de acceso a los datos (CRUD) a través de una transformación de las tablas de una base de datos en una serie de entidades [7] .

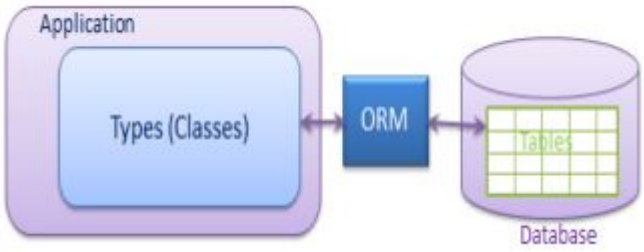


Figura 1: Entity Framework

2.2.5. Servicio web

El término servicio web se define como "Aplicaciones que utilizan estándares para el transporte, codificación y protocolo de intercambio de información. Los servicios Web permiten la intercomunicación entre sistemas de cualquier plataforma y se utilizan en una gran variedad de escenarios de integración, tanto dentro de las organizaciones como con parthners de negocios" [8].

Tipos de servicios web:
Existen 2 tipos de servicios web más usados que son SOAP y REST

SOAP

SOAP (Simple Object Access Protocol) es un estándar de comunicación entre los servicios web y las aplicaciones. y sus mensajes se definen en formato XML a su vez viajan por lo general a través de HTTP/HTTPS ya que además puede usar los protocolos como FTP y SMPT [9].
Estándares de SOAP:

- XML: (Extensible Markup Language) Lenguaje de Marcado Extensible, es el formato estándar para intercambiar datos.
- WSDL: (Web Services Description Language) es el Lenguaje de Descripción de los Servicios Web, en otras palabras, es la forma estándar de detallar los

servicios web basados en SOAP, estos detalles deben especificar la forma que debe tomar los mensajes, hacia donde van ser enviados y la respuesta del mensaje [10].

- UDDI: (Universal Description, Discovery and Integration) Descripción, Descubrimiento e Integración Universales es el registro e información de los servicios web [10].
- WS-Security: Permite optimizar la seguridad de los servicios web mediante algoritmos de encriptación [10].

Ventajas [11]:

- La depuración es posible.
- Las operaciones complejas pueden ser escondidas detrás de una fachada.
- Envolver APIs existentes es sencillo Incrementa la privacidad.
- Herramientas de desarrollo.

Desventajas [11]:

- Los clientes necesitan saber las operaciones y su semántica antes del uso.
- Los clientes necesitan puertos dedicados para diferentes tipos de notificaciones.
- Las instancias del proceso son creadas implícitamente.

REST

REST (Representational State Transfer) Transferencia de estado Representacional es servicio que se integra de una mejor manera con HTTP que el servicio web SOAP, ya que REST no requiere archivos XML para sus mensajes, estos servicio además tienen un infraestructura ligera que permite su rápida implementación y consumo, la dirección URL del recurso sirve como identificador de recursos y GET, PUT, DELETE, POST y HEAD son operaciones estándar, además REST puede enviar y recibir datos como JSON, XML o incluso texto sin formato [12]. Estándares:

- HTTP: (Hypertext Transfer Protocol) Protocolo de transferencia de hipertexto, es el protocolo de transferencia de información en Internet.

- URI: (Uniform Resource Identifier) es la dirección para acceder a la representación de un recurso.
- MIME: (Multiple Internet Mail Extension) es el identificador que describe el formato de un recurso en Internet.

Ventajas [11]:

- Bajo consumo de recursos.
- Las instancias del proceso son creadas explícitamente.
- El cliente no necesita información de enrutamiento a partir de la URI inicial.
- Los clientes pueden tener una interfaz "listener" (escuchadora) genérica para las notificaciones.
- Generalmente fácil de construir y adoptar.

Desventajas [11]:

- Gran número de objetos.
- Manejar el espacio de nombres (URIs) puede ser engorroso.
- La descripción sintáctica/semántica muy informal (orientada al usuario).
- Pocas herramientas de desarrollo.

Operaciones	SOAP	REST
Crear	CrearNuevo(string id, string titulo)	/nuevo.svc/{id} Método HTTP: POST
Actualizar	Actualizar(string id)	/nuevo.svc/{id} Método HTTP: PUT
Seleccionar	Seleccionar()	/nuevo.svc/ Método HTTP: GET
Seleccionar por ID	Seleccionar(String id)	/nuevo.svc/{id} Método HTTP: GET
Eliminar	Eliminar(String id)	/nuevo.svc/ Método HTTP: DELETE

Tabla 7: Cuadro comparativo entre los métodos de SOAP y REST

Estructura del servicio web tipo Rest:

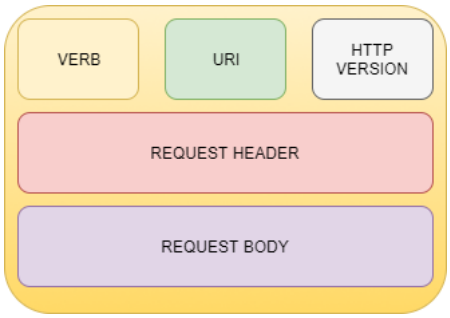


Figura 2: Request servicio web rest

Una solicitud HTTP consta de cinco partes principales:

- Verb: Indicar los métodos HTTP como GET, POST,PUT, DELETE.
- URI: Contiene el URI, Uniform Resource Identifier para identificar el recurso en el servidor
- HTTP Version: Indica la versión de HTTP, por ejemplo HTTP v1.1.
- Request Header: Contiene metadatos para el mensaje de petición HTTP como pares de valores clave. Por ejemplo, el cliente (or browser) el tipo, formato soportado por el cliente, el formato del cuerpo del mensaje, la configuración de caché, etc.
- Request Body: Contenido del mensaje o representación de recursos [12].

Figura 3: Ejemplo request usando el software Fiddler.

En caso de que el servicio web necesite de un token en el Header se debe colocar lo siguiente: Authorization: Bearer “token” en donde “token” sin comillas representa a un código del token.

```
//Uri
 WebRequest request = WebRequest.Create(Configuraciones.url + "/api/VEHICULOes");
 //Verb
 request.Method = "POST";

 //Header
 // Token
 request.Headers.Add("Authorization", "Bearer " + Session["tokenSession"].ToString());
 // Set the ContentType property of the WebRequest.
 request.ContentType = "application/json";

 //Body
 byte[] byteArray = Encoding.UTF8.GetBytes(json);
 // Set the ContentLength property of the WebRequest.
 request.ContentLength = byteArray.Length;
 // Get the request stream.
 Stream dataStream = request.GetRequestStream();
 // Write the data to the request stream.
 dataStream.Write(byteArray, 0, byteArray.Length);
 // Close the Stream object.
 dataStream.Close();
 // Get the response.
 WebResponse response = request.GetResponse();
```

Figura 4: Ejemplo request programando en c#.

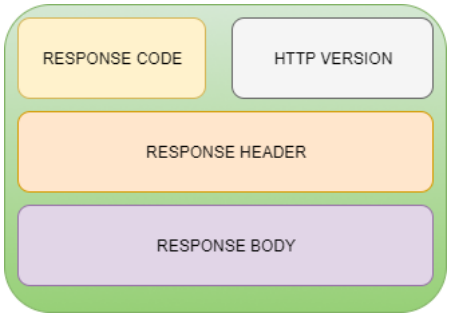


Figura 5: Response servicio web rest.

Una respuesta HTTP consta de cuatro partes principales:

- Status/Response Code: Indica el estado del servidor para el recurso solicitado. Por ejemplo 404 medios Recurso no encontrado y 200 medios de respuesta es ok.
- HTTP Version: Indica la versión de HTTP, por ejemplo HTTP v1.1.
- Response Header: Contiene metadatos para el mensaje de respuesta HTTP como pares de valores clave. Por ejemplo, longitud del contenido, tipo de contenido, fecha de respuesta, el tipo de servidor etc.
- Response Body: contenido del mensaje de respuesta o la representación de recursos [12].

Figura 6: Ejemplo Response servicio web rest usando el software Fiddler.

Autentificación y autorización.

La autentificación es el proceso en donde valida si un usuario ingresó a la aplicación por medio de un usuario y contraseña [13].
Por otro lado, la autorización determina si un usuario es autorizado para que pueda consumir el servicio web o no, los tipos de autorización son los siguientes:

- Para que cualquiera pueda consumir el servicio web.

```
public class ValuesController : ApiController
{
    public HttpResponseMessage Get(int id) { ... }
    public HttpResponseMessage Post() { ... }
}
```

Figura 7: Cualquier usuario.

- Para que solo personas autentificadas puedan consumir el servicio web, eso quiere decir que hayan obtenido un token.

```
// Require authorization for all actions on the controller.
[Authorize]
public class ValuesController : ApiController
{
    public HttpResponseMessage Get(int id) { ... }
    public HttpResponseMessage Post() { ... }
}
```

Figura 8: Usuarios autorizados.

- Para usuarios o roles específicos puedan consumir el servicio web.

```
// Restrict by user:
[Authorize(Users="Alice,Bob")]
public class ValuesController : ApiController
{
}

// Restrict by role:
[Authorize(Roles="Administrators")]
public class ValuesController : ApiController
{
}
```

Figura 9: Usuario o roles específicos autorizados.

SOAP es un formato de mensaje XML utilizado en interacciones de servicios web. Los mensajes SOAP habitualmente se envían sobre HTTP o JMS (Java Message Service), pero se pueden utilizar otros protocolos. El uso de SOAP en un servicio web específico se describe mediante la definición WSDL [14].
Una API REST describe un conjunto de recursos y un conjunto de operaciones a las que se puede llamar en estos recursos. Se puede llamar a las operaciones en una API REST desde cualquier cliente HTTP, incluido el código JavaScript del cliente que se ejecuta en un navegador web [14].
Es notable que los servicios web tipo REST son los más óptimos para implementar ya que son más ligeros para transferir información, y son en la actualidad los que más se utilizan, por ende, es el tipo de servicios web que se utilizará en este proyecto de investigación.

2.2.6. OWIN

OWIN (Open Web Interface form .NET) Es una interfaz de web abierta, es un intermediario entre el servidor de .NET y las aplicaciones web, su principal

función es brindar un acceso a un determinado recurso que se encuentra en el servidor por medio de un Token [15].

2.2.7. Token de seguridad

En un servidor lo mas importante es el acceso que tienen los usuarios, y existen varias maneras de acceder a los recursos de un servidor, el mismo servidor debe brindar los mecanismos necesarios para acceder a un sistema, determinando los proceso que puede hacer el usuario dependiendo el rol.
Un token es un grupo de datos que contiene información que verifica la identificación de una persona al momento de acceder a una aplicación [14]. Los tokens de seguridad generan claves que se van renovando cada cierto tiempo según sea configurado el tiempo de caducidad [14]. El uso de los tokens necesita que las aplicaciones sean capaces de leer el token y trasladar la información que este contiene al servidor para posteriormente autentificar al usuario, muchas veces este procedimiento no es posible sin la implementación de un sistema Single Sign On debido a que no todas las aplicaciones son compatibles entre sí [14]. Un sistema single sing on, es una manera de autentificarse en una aplicación y esta utilizar para acceder a varias aplicaciones como es el caso Google, Facebook que permiten acceder a diferentes sitios con las mismas credenciales.

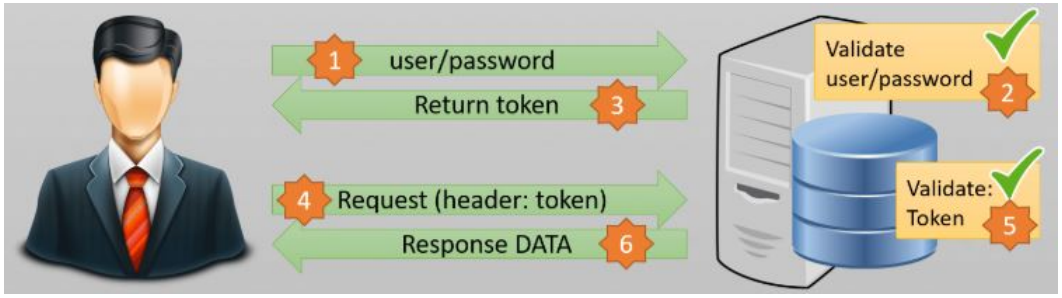


Figura 10: Funcionamiento de un token de seguridad.

Metodologías

2.2.8. RUP

El Proceso Racional Unificado o RUP (por sus siglas en inglés de Rational Unified Process), es un proceso de desarrollo de software desarrollado por la empresa Rational Software, actualmente propiedad de IBM tiene como objetivo asegurar el desarrollo de software de alta calidad a su vez haciendo que satisfaga las necesidades de sus usuarios finales dentro del plazo y con el presupuesto predecible, RUP junto con el Lenguaje Unificado de Modelado (UML), constituye

la metodología estándar más utilizada para el análisis, diseño, implementación y documentación de sistemas orientados a objetos, RUP no es un sistema con pasos firmemente establecidos, sino un conjunto de metodologías adaptables al contexto y necesidades de cada organización. También se conoce por este nombre al software, también desarrollado por Rational, que incluye información entrelazada de diversos artefactos y descripciones de las diversas actividades. Está incluido en el Rational Method Composer (RMC), que permite la personalización de acuerdo con las necesidades [16].
RUP se basa en tres principios básicos que son:

- Dirigido en UML para el modelado de los procesos.

  La RUP junto con el lenguaje unificado de modelado UML hacen una metodología estándar para el análisis, diseño, implementación, y documentación de software.

- Centrado en la arquitectura.

  La arquitectura es el plano del sistema de cómo podrá funcionar, constituye la descripción de sus partes más importantes, además se puede comprender, organizar y evolucionar el software.

  La arquitectura se comprende por plataformas de software, sistemas operativos, gestores de base de datos, protocolos, etc. y también ayuda a todos los implicados a tener una idea completa de lo que se está construyendo.

- Proceso iterativo incremental

  Para que un proyecto sea manejable se debe dividir en ciclos haciendo que el proyecto se vea que esta constituido por proyectos pequeños, los cuales están constituidos por uno o mas iteraciones de las actividades principales [16].

Fases de RUP.

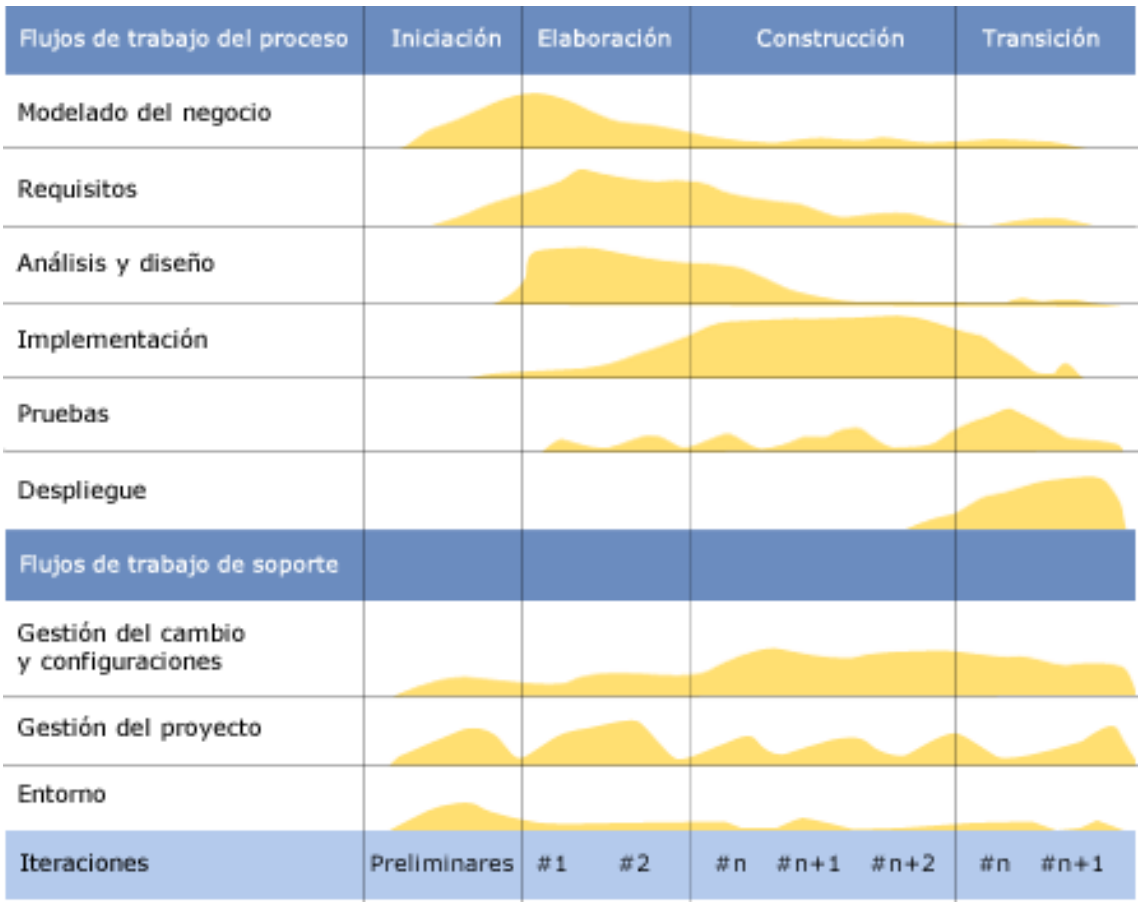


Figura 11: Estructura del proceso RUP

- Las faces de RUP en el eje vertical son:
  1. Flujos de trabajo del proceso de ingeniería.
     - Introducción. - Es el levantamiento de información inicial de una empresa.
     - Modelado del negocio. - Se usará los diagramas UML, para modelar el negocio de la empresa.
     - Requisitos. - Es la parte más importante ya que se define que es lo que debe hacer el sistema, y consta de los siguientes ítems:
       - Requisitos funcionales y no funcionales.
       - Visión.
       - Glosario.
       - Diagramas de caso de uso.
     - Análisis y diseño. - Es una traducción de los requisitos a especificaciones de cómo hacer el sistema.

- Implementación. - Es la codificación y desarrollo del sistema.
- Pruebas. - Son las pruebas de funcionamiento del sistema.
- Despliegue. - Es la distribución del sistema a los usuarios.

2. Flujos de trabajo de soporte.

- Gestión del proyecto. - En esta parte se debe equilibrar los objetivos, administrar los riesgos, y superar las restricciones del proyecto.
- Gestión del cambio y configuraciones. - Mantiene la integridad de todo lo que va creando durante el desarrollo, es como el controlador de versiones del sistema.
- Entorno. - Se encarga de preparar el entorno de trabajo para el desarrollo del proyecto.

- En el eje horizontal:

  En cada proceso se realizan una serie de artefactos los cuales son:

  - Análisis o iniciación.
  - Diseño o elaboración.
  - Desarrollo (también llamado Implementación, Construcción).
  - Cierre o transición

Características

- Desarrollar software iterativamente.
- Administrar los requerimientos.
- Utilizar arquitecturas basadas en componentes.
- Modelar software visualmente.
- Verificar la calidad del software.
- Controlar los cambios del software.

Se puede simplificar algunas de los muchos fases que ofrece RUP como recomienda IBM Rational, para que se pueda hacer un proceso ágil (AUP), que desde el punto de vista de los desarrolladores de RUP tradicional se puede decir que es más optimizada, mientras que los desarrolladores que utilizan XP dirán que es más pesada, quedando AUP como intermedio entre RUP y XP [17].
Filosofía de AUP
AUP se basa en los siguientes principios:

- Simplicidad.
- Agilidad.
- Concentrarse en actividades de alto valor.
- Independencia de la herramienta.
- Querrá adaptar este producto para satisfacer sus propias necesidades.

Tomando en cuenta algunas características mas importantes entre las siguientes metodologías:

2.2.9. Metodología XP

La metodología XP está centrada en las relaciones interpersonales promoviendo el trabajo en equipo, se basa en la retroalimentación con el cliente y el equipo de desarrollo, es especialmente adecuada para proyectos con requisitos imprecisos, cambiantes y un alto riesgo técnico [18].

- Ligero
- Cercano al desarrollo
- Se basa en historias de usuario.
- Fuerte comunicación con el cliente
- El código fuente pertenece a todos
- Programación por parejas
- Solo el mínimo de organización
- Pobre en cuanto a documentación

2.2.10. Metodología 3p

Los principios que sustentan el modelo son:
El personal. - Los individuos y sus iteraciones son mas importante que los procesos y las herramientas.
El problema. - La comunicación con el cliente debe construir una solución.
El proceso. - El funcionamiento del sistema es mas importante que la documentación exhaustiva.
Características:

- Ágil
- Cercano al desarrollo, pero sin olvidar el diseño.
- Se basa en 3 principios: Personal, Problema, Proceso.
- Gran interacción con el cliente.
- Logra alcanzar un control y organización del proceso.
- Logra un equilibrio en cuanto a la generación de documentación[19].

Se optó por escoger la metodología RUP, por sus características que son modelado UML, proceso iterativo incremental, y se basa en la arquitectura [20].
Los procesos que se realizan en una reserva de viaje se entienden mejor haciendo diagramas UML, la arquitectura que conforma el sistema esta divido en la parte de servicios web y la aplicación web que consume los mismos, además el sistema es mejor dividir en subprocesos incrementales.

2.2.11. Retrofit

Retrofit es un cliente REST para Android desarrollado por Square, permite hacer peticiones como GET, POST, PUT, DELETE, gestionando múltiples parámetros y convirtiendo a una respuesta a un objeto java [21].

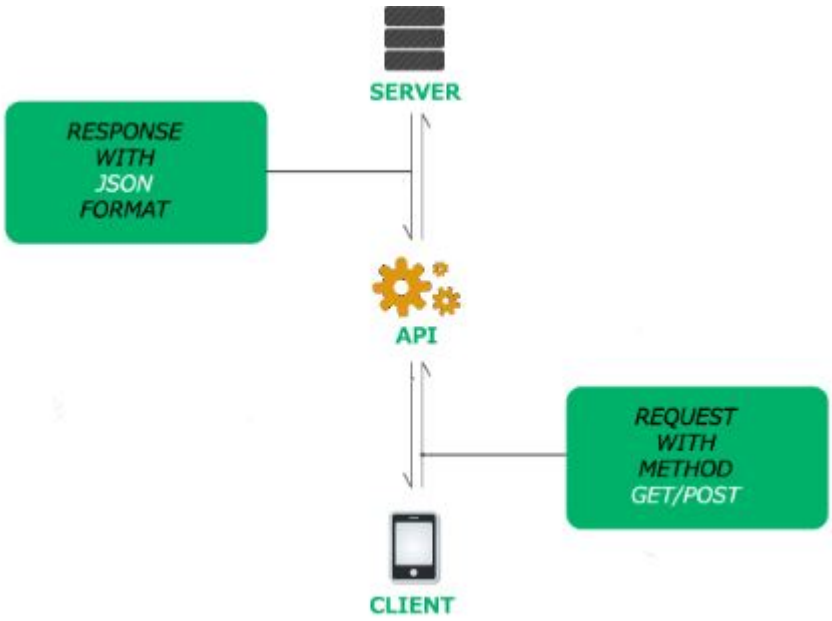


Figura 12: Arquitectura de retrofit.

2.2.12. Android

Es un sistema operativo para dispositivos móviles como teléfonos inteligentes, tabletas, relojes inteligentes, televisiones, etc. Android bajo la definición de Google se considera un software stack, debido a que se conforma de [21]:

- Sistema operativo.
- El middleware para la conexión de redes.
- Las aplicaciones que se pueden ejecutar.

Sus principales características son [21]:

- Plataforma totalmente libre basado en Linux.
- Multitarea.
- Compatibilidad en dispositivos móviles.
- Portal para descargar aplicaciones.

2.3. Propuesta de Solución

Desarrollar una aplicación web para realizar reservas en empresas de servicios de transporte puerta a puerta aplicando servicios web se busca principalmente dar facilidades a los clientes al momento de viajar, y al usar servicios web se podrá consumir los mismos en diferentes plataformas de desarrollo como aplicaciones web.

CAPÍTULO 3

Metodología

3.1. Modalidad de la investigación

El presente trabajo tiene las siguientes modalidades de investigación:
Modalidad Bibliográfica o Documentada Se considera esta modalidad ya que se recurre a diferentes fuentes obtenidas de libros, artículos científicos, tesis desarrolladas en Universidades para profundizar enfoques con respecto al tema de la investigación.
Modalidad aplicada Por la utilización de los conocimientos adquiridos a lo largo de la carrera universitaria.

3.2. Recolección de información

Para la recolección de información se utilizará la técnica de investigación documental de tipo informativa (expositiva), se buscará información relevante de diferentes fuentes confiables, la información analizada y sintetizada servirá de apoyo para el desarrollo de este trabajo.

3.3. Procesamiento y análisis de datos

Para el procesamiento y análisis de la información se aplicará los siguientes procedimientos:
1. Obtener Información sobre los procesos.
2. Analizar la información obtenida.
3. Diseño de procesos.
4. Desarrollo de sistema web.
5. Implementación del sistema.

3.4. Desarrollo del Proyecto

1. Definición del concepto y aspectos técnicos de la aplicación web.
2. Determinación los requerimientos de un sistema de reservas online.
3. Diseño de la base de datos para el sitio web.

4. Diseño del sitio web.
5. Desarrollo del sitio web.
6. Pruebas Finales de aceptación
7. Documentación del manual técnico y el manual de usuario.

CAPÍTULO 4

Desarrollo de la propuesta.

4.1. Introducción.

Para facilitar el desarrollo se utilizará las plantillas basadas en la metodología.

Ficha Cliente	
Nombre	Pablo
Apellido	Barrionuevo
Ciudad	Ambato
Calle principal	Cevallos
Calle secundaria	Montalvo
N° de casa	3-41
Teléfono	0984785122

Tabla 8: Ficha Cliente

Ficha vehículo	
Placa	TCU-293
Modelo	Sedan
Color	Blanco
Marca	Chevrolet
Cilindraje	1.8
Capacidad	4
Año	2010

Tabla 9: Ficha vehículo.

Ficha Reserva	
Cliente	Pablo Barrionuevo
N° asientos	2
N° maletas	1
Ciudad Origen	Ambato
Calle Principal Origen	Cevallos
Calle secundaria Origen	Quito
Referencia Origen	Al frente del Colegio
Ciudad destino	Quito
Calle principal destino	10 de agosto
Calle secundaria destino	Eloy Alfaro
Referencia destino	Universidad Central
Teléfono	0984785122
Fecha y hora	17/07/2017 8:00

Tabla 10: Ficha reserva.

La entrevista se realizó de la empresa de trasporte Confort Tour de la ciudad de Ambato, además la observación y del proceso que se realiza se realizó en base a las experiencias de reservas de viaje en la misma empresa debido a los viajes frecuentes que yo realizo a la ciudad de Quito.

Entrevista.

N	PREGUNTA	RESPUESTA
1	Cuál es la manera en que registran una reserva de viaje?	Las reservas se las realiza mediante llamadas, mensajes de texto, Whatsapp, redes sociales.
2	Cuántas reservas en promedio tiene por día?	35 reservas.
3	Tiene horarios definidos para viajar?	Los horarios son a cada hora.
4	De cuantos vehículos dispone la empresa para realizar los viajes?	Nueve vehículos.
5	Cómo notifica a los choferes que tienen una reserva de viaje, en que lugares están esperando los clientes y hacia dónde van?	Mediante mensajes de texto.
6	Como guarda la información de los clientes?	En archivos de excel, y también en el equipo de la central como contacto.

7	Qué métodos de pago dispone para cancelar un viaje?	Solo en efectivo.
8	Cuando ya no dispone de asientos disponibles que alternativas toma?	Avisa a otra empresa de transporte.
9	Con cuántos pasajeros mínimo sale al destino?	Mínimo con dos pasajeros.
10	Cuando no cumple con los pasajeros requeridos para salir de viaje al destino, que hace usted?	Pide ayuda a otro compañero con el mismo turno.
11	Como hace el proceso para consultar cuántos asientos están reservados y hacia qué destino se dirigen?	Revisando los apuntes, llamadas, o mensajes, y en ocasiones, recordando las llamadas
12	Cómo sabe si un cliente es frecuente?	Es difícil saberlo, ya que no se puede recordar muchos clientes.
13	Aplican descuentos o promociones a clientes frecuentes?	Por el momento no, porque no se puede saber cuántos clientes son frecuentes.
14	Qué información o reportes usted necesita?	- Cuantas reservas exactamente se hicieron al día, mes y año. - Clientes - Viajes por cada vehículo.

Tabla 12: Entrevista.

Observación.

GUÍA DE OBSERVACIÓN
Objetivo. Conocer los procesos que se realizan en viaje a través de un servicio de trasporte puerta a puerta.
Tipo de observación: Directa.
Indicadores a observar.
1. Medios para registrar una reserva.
- Libreta de apuntes. - Archivo de Excel. - Hojas volantes.
2. Medios para almacenar información.
- Archivos de Excel. - Archivadores. - Carpetas físicas.
3. Disponibilidad de la información.
- Búsqueda en hojas electrónicas. - Búsqueda en archivos físicos almacenados.
4. Demanda de reservas.
- Se realizan un promedio de 35 reservas diarias.

Tabla 13: Observación del proceso de reserva

4.2. Requisitos

4.2.1. Requisitos funcionales y no funcionales.

- Requisitos funcionales:
  - El sistema deberá permitir registrar un usuario.
  - El sistema debe manejar control de accesos de acuerdo a roles.
  - Cuando se registre un usuario se asignará un rol por defecto.
  - Debe permitir registrar varios clientes por cada usuario.
  - El cliente podrá realizar varias reservas.
  - El sistema debe asignar un vehículo cuando el cliente hace una reserva.
  - El cliente podrá actualizar su información.
  - El sistema mostrará reportes para los clientes, choferes y administradores.
  - El chofer y asistente podrán actualizar el estado de la reserva de un cliente según la ocasión.

- No se podrá eliminar una reserva, sino a su vez dar de baja.
- El sistema mostrará un reporte con la cantidad de viajes que ha realizado un vehículo.
- El sistema mostrará los clientes que fueron asignados a un vehículo para el traslado.
- El sistema no debe dejar ingresar datos incorrectos según sea la necesidad.
- Para consumir un servicio web debe pedir autentificación por el usuario.

- Requisitos no funcionales:
  - El sistema debe ser capaz de procesar un promedio de 5 transacciones por cada 35 usuarios en un día.
  - El sistema debe ser intuitivo para que el cliente pueda hacer una reserva fácilmente.
  - El diseño debe ser responsivo.
  - El sistema debe mostrar mensajes de error o alertas.

4.2.2. Visión.

El problema de	Controlar las reservas de viajes realizadas por los clientes, de forma que se puedan gestionar las reservas con los vehículos. Gestionar las reservas por los clientes. Gestionar las reservas por los choferes.
afecta a	Al cliente que realiza la reserva. Chofer del vehículo. Asistente o administrador de la empresa.
el impacto asociado es	Al momento de guardar información de los clientes, y la información de cada vehículo que realiza una reserva.
Una solución óptima es	Desarrollar un sistema, para que el cliente guarde su información y pueda hacer una reserva, además guardar la información de cada vehículo con la reserva que realice.

Tabla 14: Sentencia que define el problema.

4.2.3. Glosario.

- Cliente. - La persona que realiza la reserva de servicios de transporte puerta a puerta.

- Chofer. - Persona encargada de conducir el vehículo que trasladará al cliente de un punto a otro.
- Asistente. - Persona encargada de registrar una reserva, será la persona que atenderá a los clientes en la oficina.
- Administrador. - Persona con acceso a todo el sistema.
- Marker. - Punto en el mapa que indica una ubicación específica.
- Latitud. - Cifras decimales entre 0 y 90 que indican una distancia en un mapa.
- Longitud. - Cifras decimales entre 0 y 180 que indican una distancia en un mapa.
- Usuario. - Persona que interactúa con el sistema.
- Rol. - Característica de un usuario que restringe a ciertas partes del sistema.

4.2.4. Modelado del negocio.

Procesos de una reserva de viaje puerta a puerta.

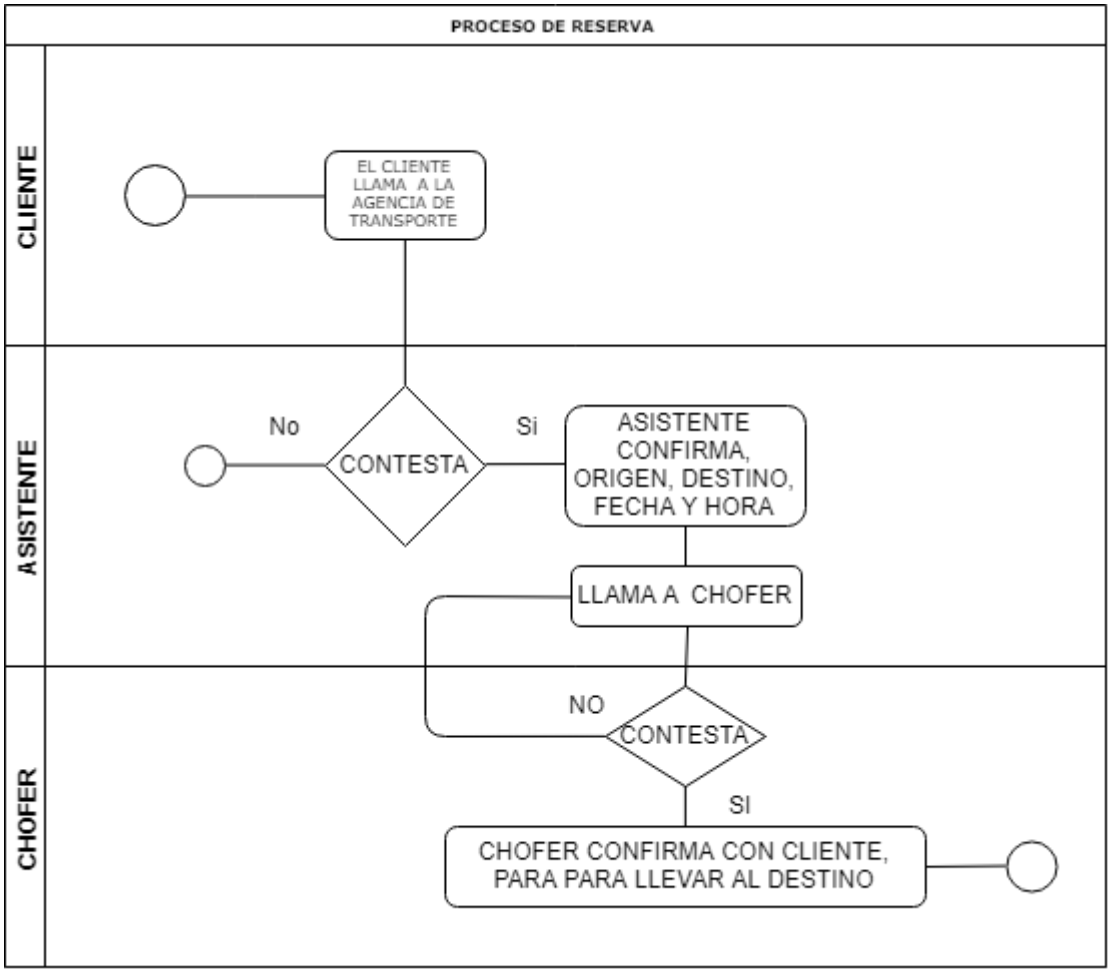


Figura 13: Modelado del proceso de una reserva de viaje puerta a puerta.

Procesos para la reserva de un viaje a través de un servicio de transporte puerta a puerta.
CLIENTE:
- El cliente realiza una reserva de viaje.
ASISTENTE DE VIAJE:
El o la asistente confirma con el cliente y obtiene: - El destino. - El origen. - Hora de viaje.
CHOFER
- Realiza el transporte.

Tabla 15: Descripción del proceso de una reserva de viaje puerta a puerta.

Diagrama de caso de uso Administrador

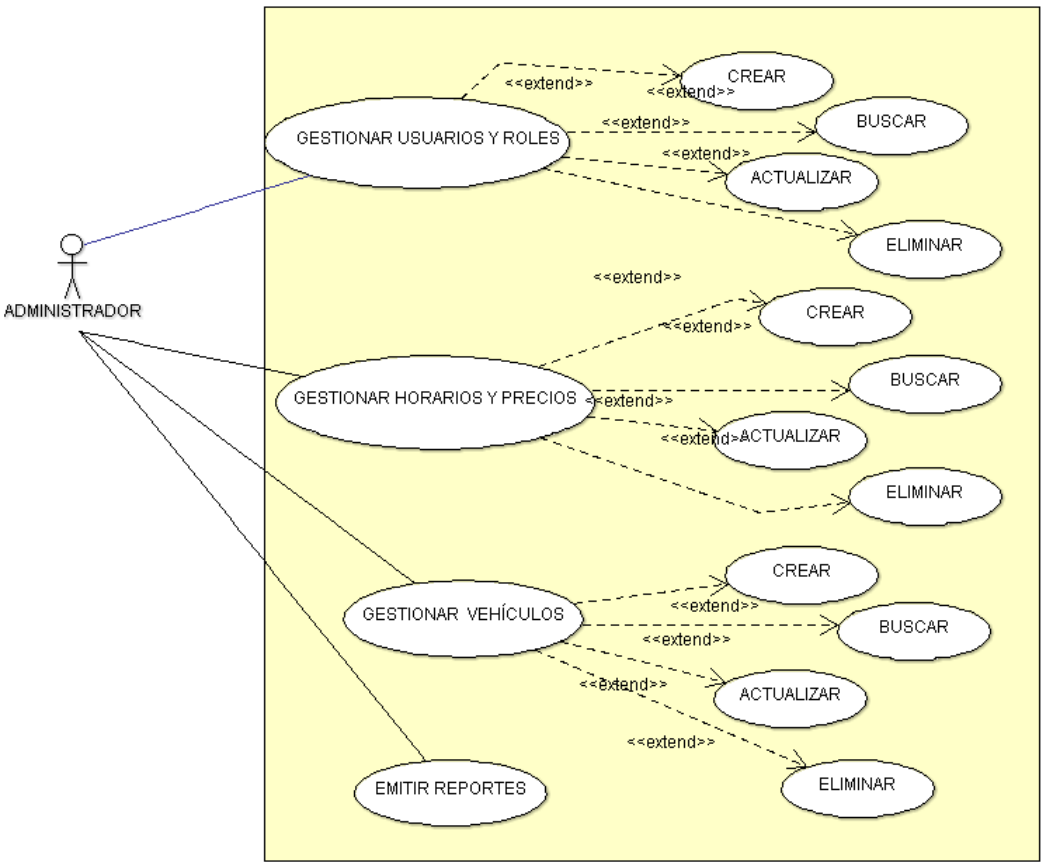


Figura 14: Caso de uso - Administrador

CASO DE USO	Gestionar roles.
ACTOR	Administrador.
DESCRIPCIÓN	Son los roles que se configuran para determinar el acceso al sistema, de ello depende que menú se muestra a cada usuario dependiendo el rol.
PRECONDICIÓN	Se debe ingresar con usuario administrador al sistema.
CURSO NORMAL	El administrador ingresa los roles al sistema, según las necesidades del módulo y los campos a solicitar.
CURSO ALTERNO	El sistema realiza la validación con respecto a los roles ingresados y a los roles que ya están registrados en el sistema para no permitir duplicidad.
POSTCONDICIÓN	Los Datos han sido almacenados en el sistema Se puede continuar con las demás operaciones.

Tabla 16: Gestionar roles

CASO DE USO	Gestionar usuarios.
ACTOR	Administrador.
DESCRIPCIÓN	Son los usuarios ingresados al sistema para su validación de acceso al mismo.
PRECONDICIÓN	Ingresar al sistema con un usuario administrador para la gestión de usuarios.
CURSO NORMAL	El administrador ingresa los usuarios al sistema, según las necesidades del módulo y los campos a solicitar.
CURSO ALTERNO	El sistema realiza la validación con respecto a los usuarios ingresados y a los usuarios que ya están registrados en el sistema para no permitir duplicidad.
POSTCONDICIÓN	Los Datos han sido almacenados en el sistema Se puede continuar con las demás operaciones.

Tabla 17: Gestionar usuarios

CASO DE USO	Gestionar de horarios y precios.
ACTOR	Administrador.
DESCRIPCIÓN	Es la lista de precios y horarios disponibles para realizar la reserva de viaje, los mismos que se muestran a los clientes.
PRECONDICIÓN	Ingresar con un usuario administrador al sistema.
CURSO NORMAL	El administrador ingresa los horarios y precios al sistema, según las necesidades del módulo y los campos a solicitar.
CURSO ALTERNO	El sistema realiza la validación con respecto a los horarios ingresados y a los horarios que ya están registrados en el sistema para no permitir duplicidad.
POSTCONDICIÓN	Los Datos han sido almacenados en el sistema Se puede continuar con las demás operaciones.

Tabla 18: Gestionar horarios y precios

CASO DE USO	Gestionar Vehículos.
ACTOR	Administrador.
DESCRIPCIÓN	Son los vehículos que prestan el servicio de transporte para los clientes.
PRECONDICIÓN	Ingresar con usuario administrador al sistema.
CURSO NORMAL	El administrador ingresa los vehículos al sistema, según las necesidades del módulo y los campos a solicitar.
CURSO ALTERNO	El sistema realiza la validación con respecto a los vehículos
POSTCONDICIÓN	Los Datos han sido almacenados en el sistema Se puede continuar con las demás operaciones.

Tabla 19: Gestionar vehículos

CASO DE USO	Emitir reportes.
ACTOR	Administrador.
DESCRIPCIÓN	Son los reportes de número de reservas por vehículo, clientes más frecuentes, y asignación de asientos al cliente.
PRECONDICIÓN	Ingresar con usuario administrador al sistema.
CURSO NORMAL	El administrador genera los reportes requeridos.
CURSO ALTERNO	No existen datos para generar el reporte.
POSTCONDICIÓN	El reporte se ha generado correctamente.

Tabla 20: Emitir reportes.

Especificaciones del caso de uso

Diagrama de caso de uso Asistente

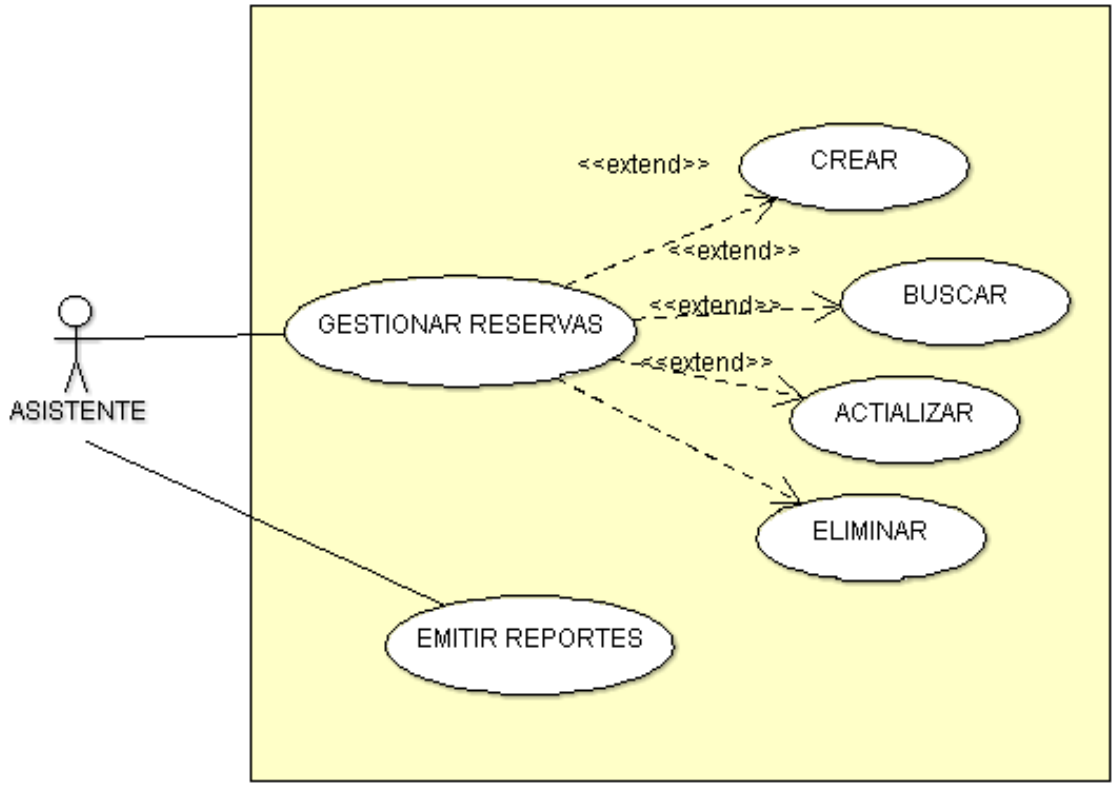


Figura 15: Caso de uso - Asistente

CASO DE USO	Gestionar reservas.
ACTOR	Asistente.
DESCRIPCIÓN	Realiza todo el proceso de reservas desde que se contacta con el cliente hasta que asigna a un vehículo.
PRECONDICIÓN	Ingresar con usuario asistente al sistema.
CURSO NORMAL	El asistente ingresa las reservas al sistema, según las necesidades del módulo y los campos a solicitar.
CURSO ALTERNO	El sistema realiza la validación con respecto a las reservas
POSTCONDICIÓN	Los Datos han sido almacenados en el sistema Se puede continuar con las demás operaciones.

Tabla 21: Gestionar reservas del asistente.

CASO DE USO	Emitir reportes.
ACTOR	Asistente.
DESCRIPCIÓN	Son los reportes de número de reservas por vehículo, clientes más frecuentes, y asignación de asientos al cliente.
PRECONDICIÓN	Ingresar con usuario asistente al sistema.
CURSO NORMAL	El asistente genera los reportes requeridos.
CURSO ALTERNO	No existen datos para generar el reporte.
POSTCONDICIÓN	El reporte se ha generado correctamente.

Tabla 22: Emitir reportes.

Especificaciones del caso de uso

Diagrama de caso de uso Chofer

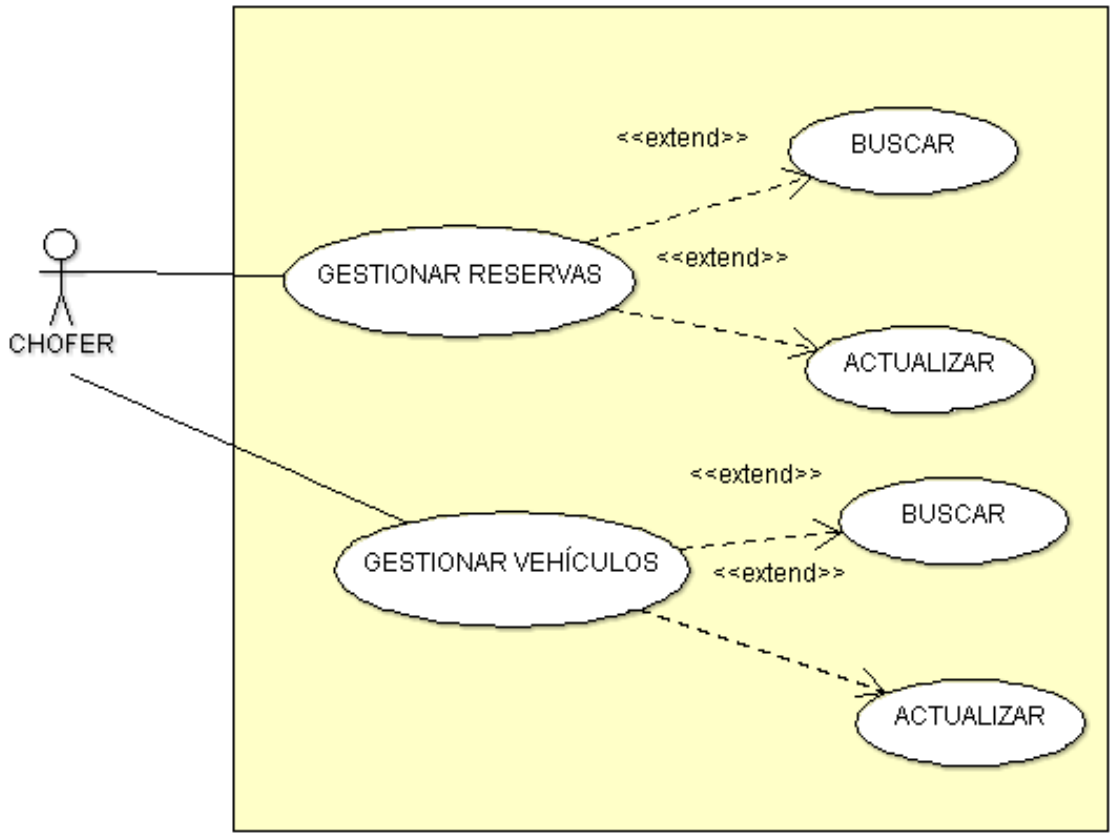


Figura 16: Caso de uso - Chofer

CASO DE USO	Gestionar reservas.
ACTOR	Chofer.
DESCRIPCIÓN	Cuando ya llega a su destino con el o los pasajeros actualiza el estado de la reserva.
PRECONDICIÓN	Debe ingresar con un usuario chofer, y tener asignado una reserva y un vehículo.
CURSO NORMAL	El chofer ingresa actualiza el estado de la reserva y el vehículo.
CURSO ALTERNO	El sistema realiza la validación con respecto a las reservas
POSTCONDICIÓN	Los Datos han sido actualizados en el sistema Se puede continuar con las demás operaciones.

Tabla 23: Gestionar reservas chofer.

CASO DE USO	Gestionar vehículos.
ACTOR	Chofer.
DESCRIPCIÓN	El chofer puede hacer las modificaciones del vehículo que le pertenece según sea la circunstancia.
PRECONDICIÓN	Debe ingresar con un usuario chofer y tener asignado un vehículo.
CURSO NORMAL	El asistente ingresa las reservas al sistema, según las necesidades del módulo y los campos a solicitar.
CURSO ALTERNO	El sistema realiza la validación con respecto a los vehículos.
POSTCONDICIÓN	Los Datos han sido actualizados en el sistema Se puede continuar con las demás operaciones.

Tabla 24: Gestionar vehículos.

Especificaciones del caso de uso

Diagrama de caso de uso Cliente

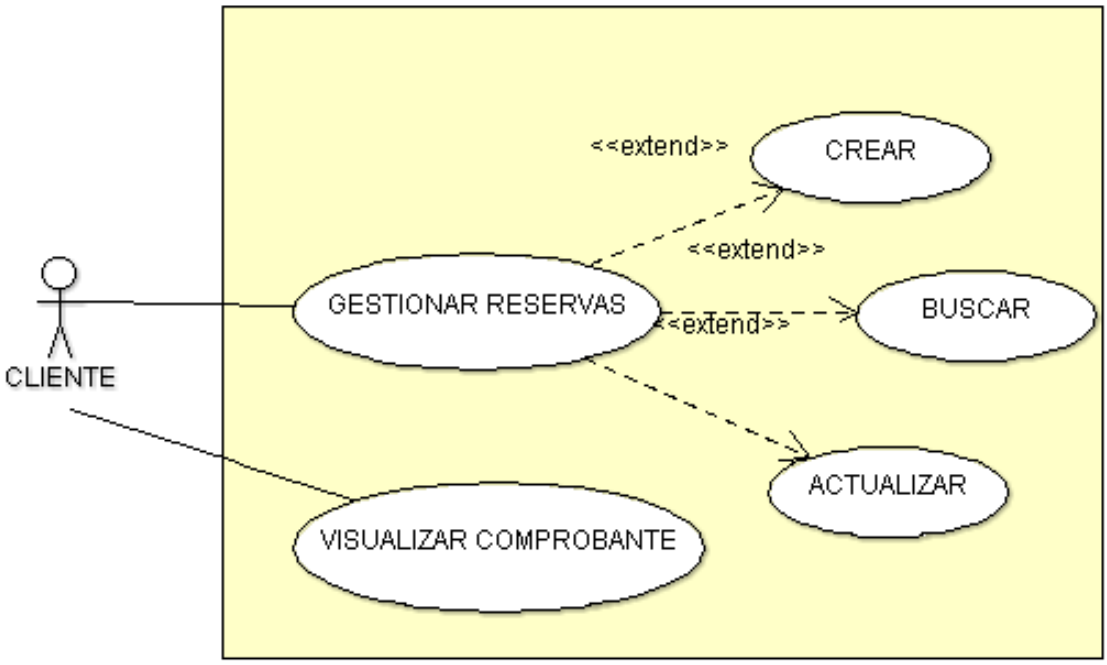


Figura 17: Caso de uso - Cliente

CASO DE USO	Gestión de reservas.
ACTOR	Cliente.
DESCRIPCIÓN	El cliente realiza el proceso para llevar a cabo una reserva
PRECONDICIÓN	Debe ingresar con un usuario cliente.
CURSO NORMAL	El cliente ingresa las reservas al sistema, según las necesidades del módulo y los campos a solicitar.
CURSO ALTERNO	El sistema realiza la validación con respecto a las reservas
POSTCONDICIÓN	Los Datos han sido almacenados en el sistema Se puede continuar con las demás operaciones.

Tabla 25: Gestión de reservas

CASO DE USO	Emitir reportes.
ACTOR	Cliente.
DESCRIPCIÓN	Asignación de asientos al cliente.
PRECONDICIÓN	Ingresar con usuario cliente al sistema.
CURSO NORMAL	El cliente genera los reportes requeridos.
CURSO ALTERNO	No existen datos para generar el reporte.
POSTCONDICIÓN	El reporte se ha generado correctamente.

Tabla 26: Emitir reportes.

Especificaciones del caso de uso

Diagrama de secuencias.

Diagrama de secuencias del usuario administrador.

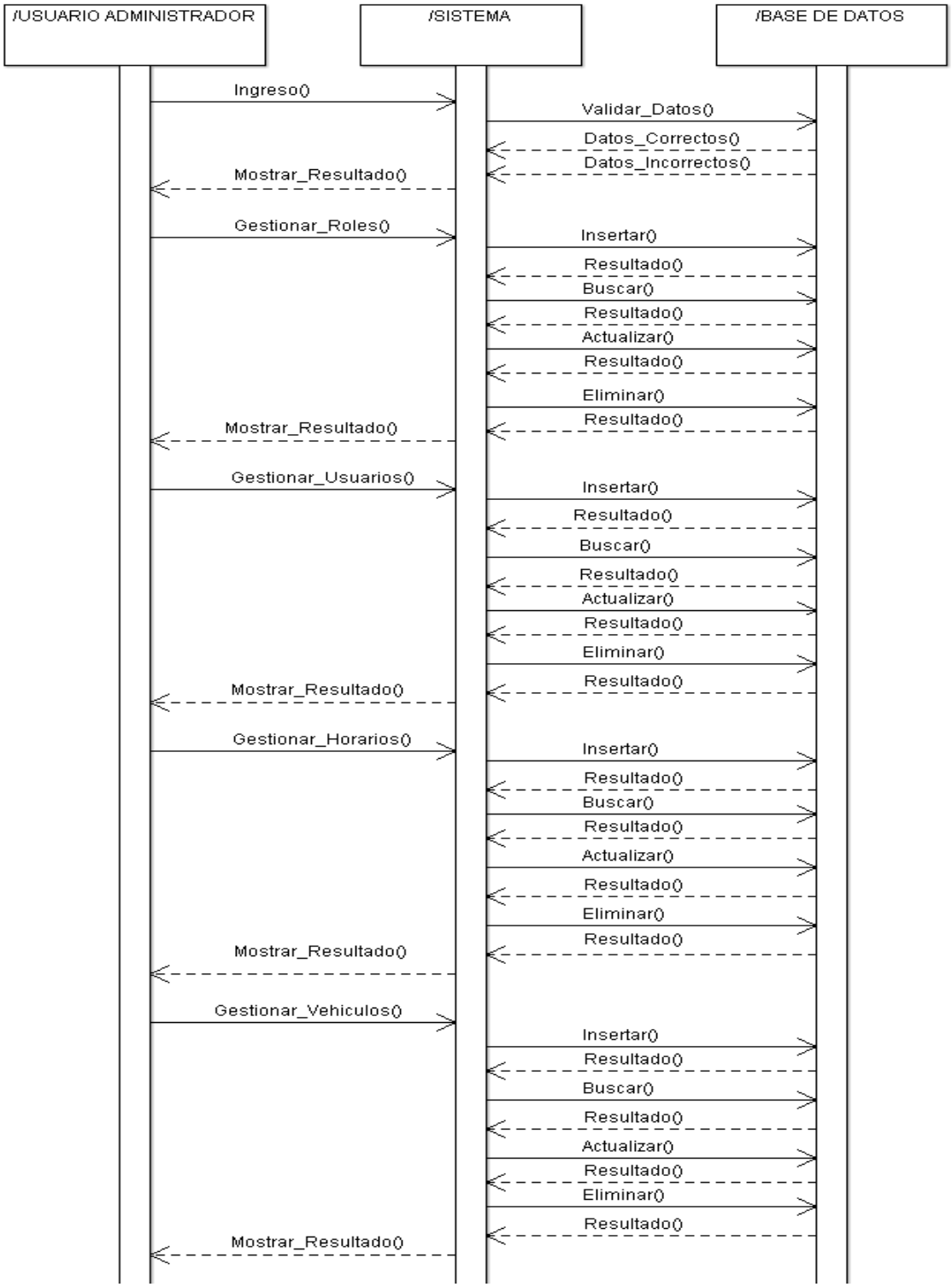


Figura 18: Diagrama de secuencia Administrador.

Diagrama de secuencias del usuario asistente.

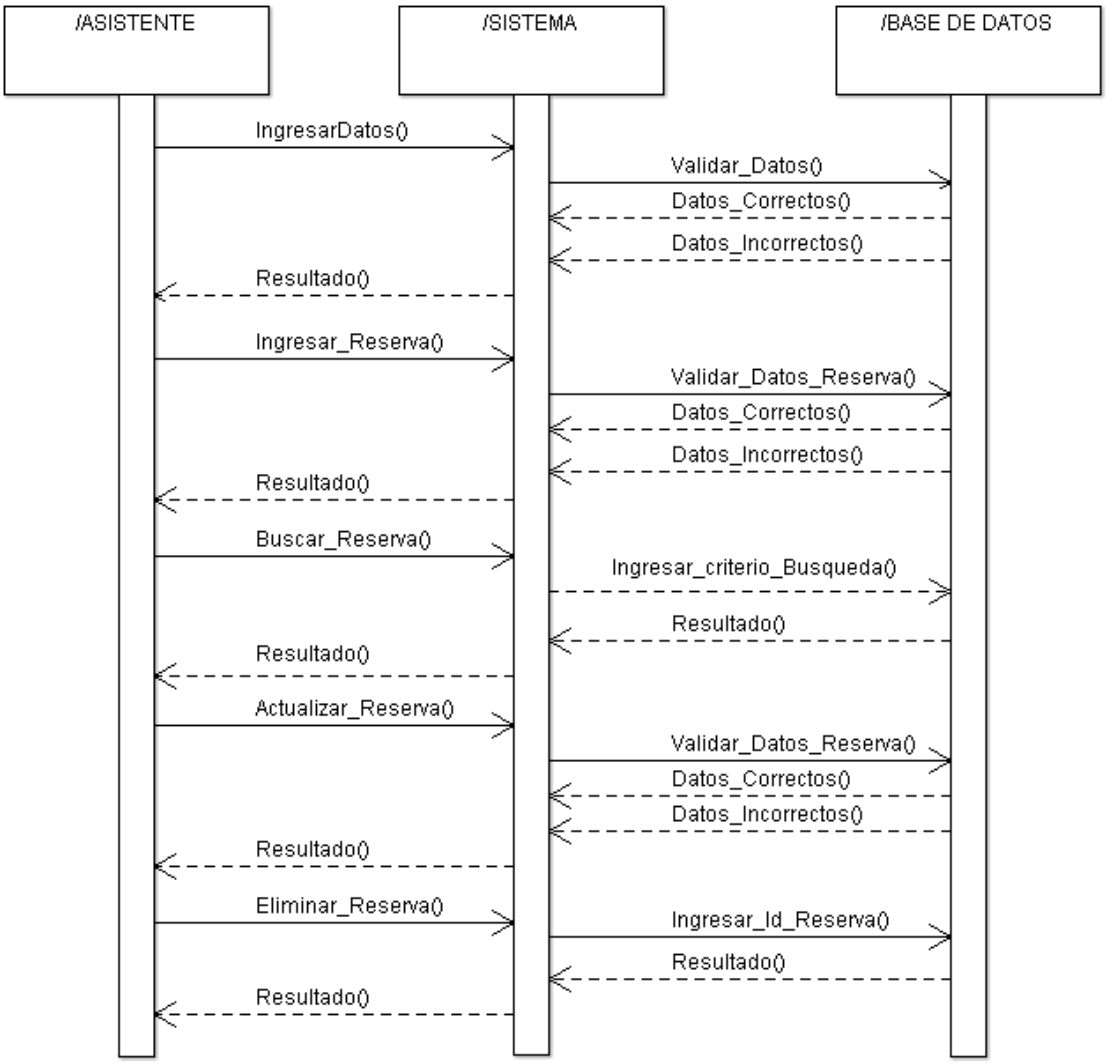


Figura 19: Diagrama secuencia Asistente

Diagrama de secuencias del usuario Chofer.

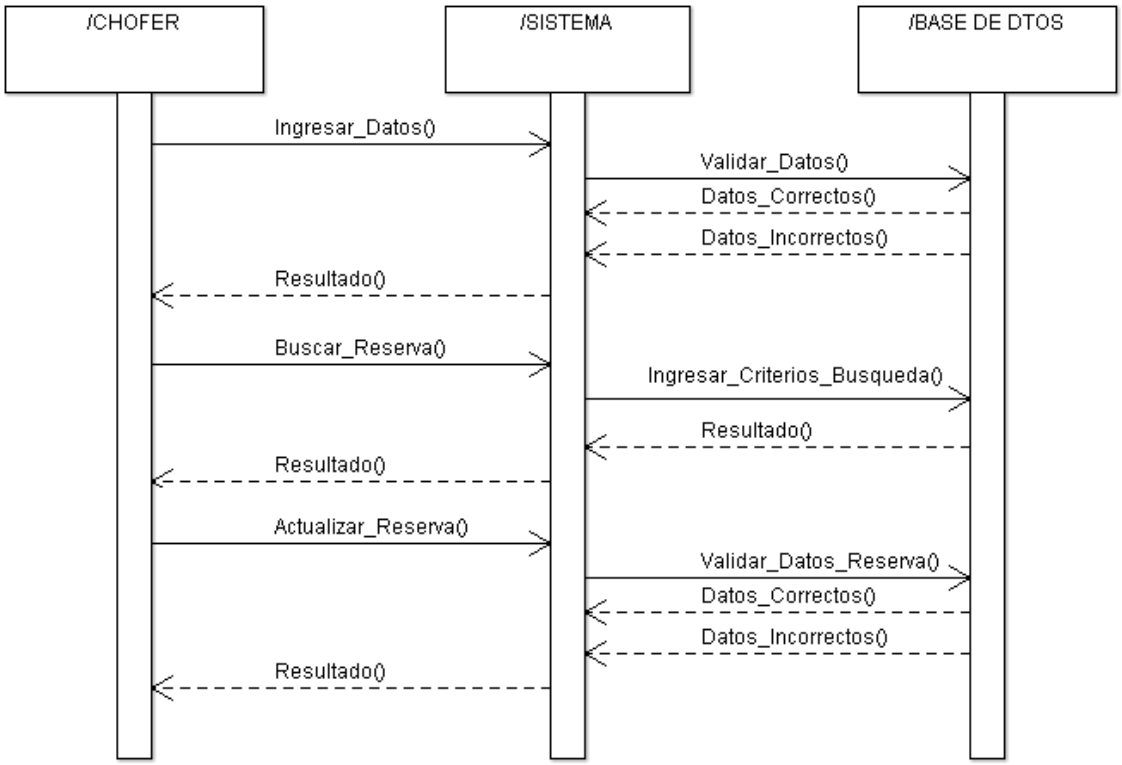


Figura 20: Diagrama secuencia Chofer.

Diagrama de secuencias del usuario cliente.

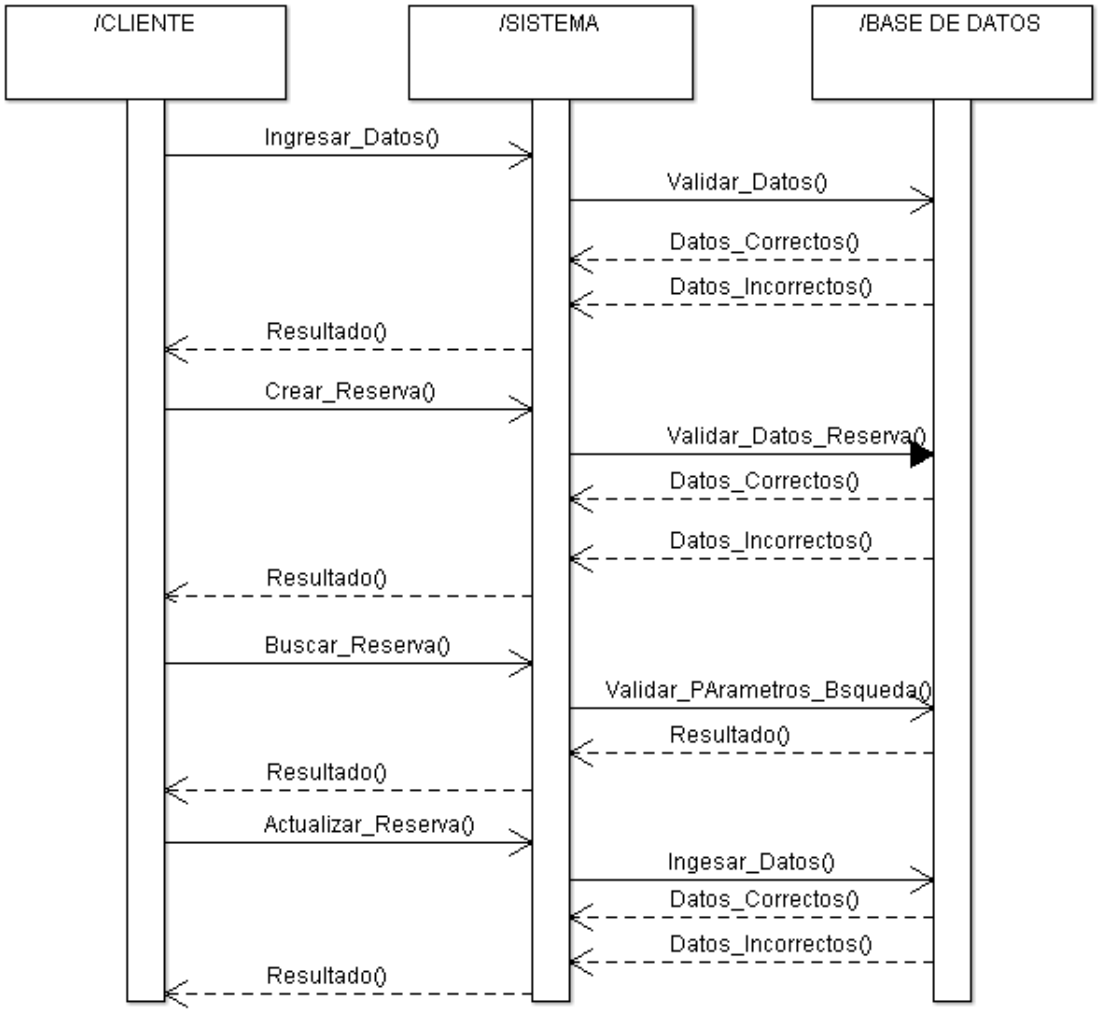


Figura 21: Diagrama secuencia Cliente

4.3. Análisis y diseño.

4.3.1. Modelo lógico.

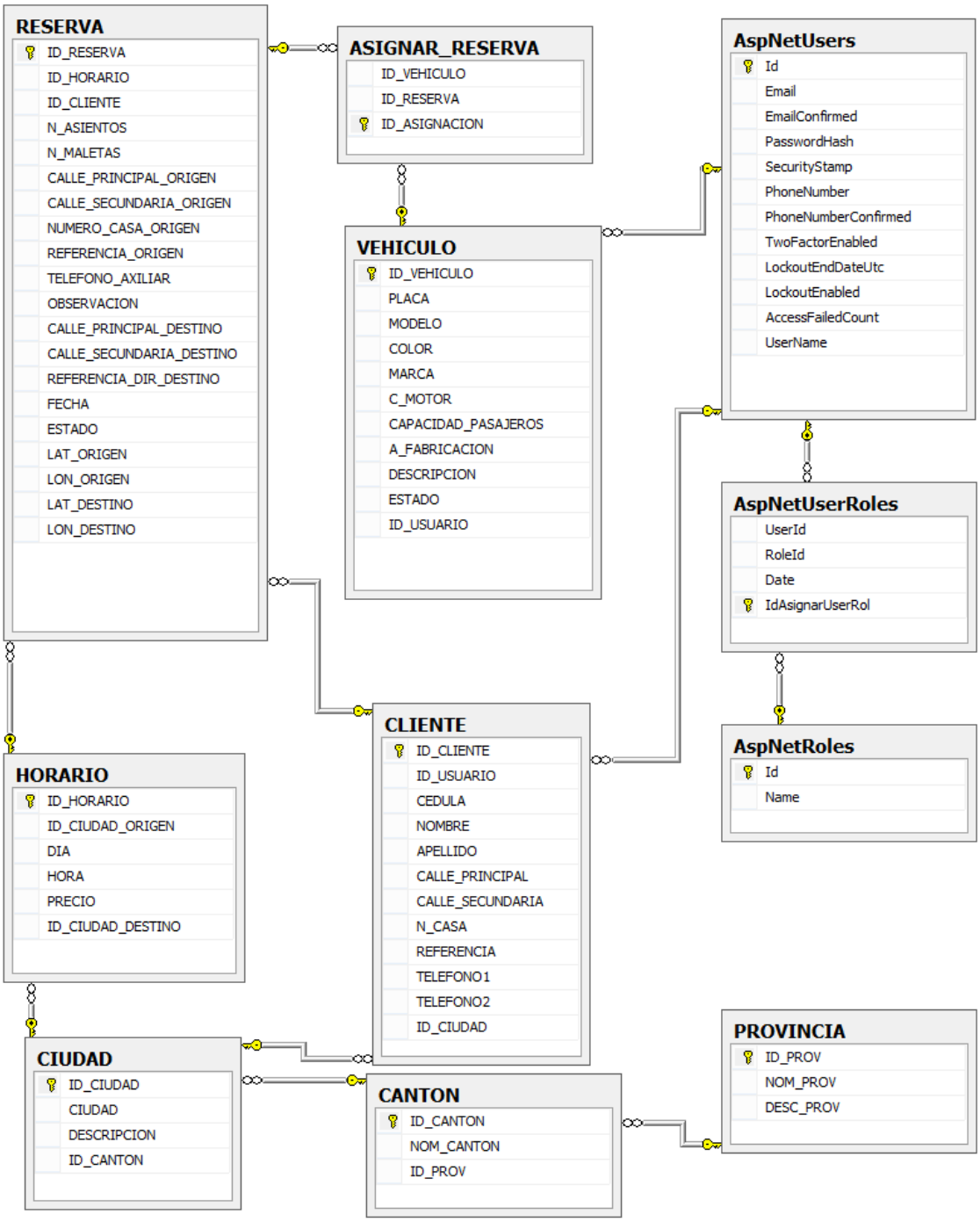


Figura 22: Modelo lógico de la base de datos.

4.3.2. Diccionario de datos.

Entidad: Provincia

ATRIBUTO	TIPO	NULO	OBSERVACIÓN
ID_PROV	VARCHAR(10)	No	Id de provincia. (pk)
NOM_RPOV	varchar(100)	No	Nombre de provincia.
DESC_PROV	varchar(100)	Si	Descripción de la provincia.

Tabla 27: Diccionario de datos entidad Provincia.

Entidad: Cantón.

ATRIBUTO	TIPO	NULO	OBSERVACIÓN
ID_CANTON	VARCHAR(10)	No	Id de cantón. (pk)
NOM_CANTON	varchar(100)	No	Nombre de cantón.
ID_PROV	varchar(10)	No	Id de provincia (fk).

Tabla 28: Diccionario de datos entidad Cantón.

Entidad: Ciudad.

ATRIBUTO	TIPO	NULO	OBSERVACIÓN
ID_CIUDAD	int	No	Id de ciudad. (pk)
CIUDAD	varchar(50)	No	Nombre de la ciudad.
DESCRIPCIÓN	varchar(100)	Si	Descripción de la ciudad.
ID_CANTON	varchar(10)	No	Id Cantón (fk)

Tabla 29: Diccionario de datos entidad ciudad

Entidad: Horario.

ATRIBUTO	TIPO	NULO	OBSERVACIÓN
ID_HORARIO	int	No	Id del horario (pk).
ID_CIUDAD_ORIGEN	int	No	Id ciudad origen (fk).
ID_CIUDAD_DESTINO	int	No	Id ciudad destino (fk).
DIA	varchar(15)	No	Día de la semana.
HORA	char(5)	No	Hora del horario.
PRECIO	money	No	Precio del horario.

Tabla 30: Diccionario de datos entidad horario.

Entidad: AspNetRoles.

ATRIBUTO	TIPO	NULO	OBSERVACIÓN
ID	nvarchar(128)	No	Id del rol (pk).
NAME	nvarchar(256)	No	Nombre de rol.

Tabla 31: Diccionario de datos entidad AspNetRoles.

Entidad: AspNetUsers.

ATRIBUTO	TIPO	NULO	OBSERVACIÓN
ID	nvarchar(128)	No	Id del rol (pk).
EMAIL	nvarchar(256)	No	Nombre de rol.
EMAILCONFIRMED	bit	No	Confirmar email
PASSWORDHASH	nvarchar(max)	Si	Contraseña
SECURITYSTAMP	nvarchar(max)	Si	Valor aleatorio.
PHONENUMBER	nvarchar(max)	Si	Número de teléfono
PHONENUMBER CONFIMRMED	bit	No	Confirmación del numero.
TWOFACTOR ENABLE	bit	No	Confirmar acceso en dos pasos.
LOCKOUTENDDATE UTC	datetime	Si	Fecha de bloqueo del usuario.
LOCKOUTENABLE	bit	No	Habilitado
ACCESSFAILED COUNT	bit	No	Número de intentos válidos para ingresar.
USERNAME	nvarchar(256)	No	Nombre de usuario.

Tabla 32: Diccionario de datos entidad AspNetUsers.

Entidad: AspNetuserRoles.

ATRIBUTO	TIPO	NULO	OBSERVACIÓN
IDASIGNAR USERROL	int	No	Id de Asignación (pk)
USERID	int	No	Id de usuario (fk).
ROLEID	varchar(50)	No	Id de rol (fk).
DATE	varchar(100)	Si	Fecha de asignación.

Tabla 33: Diccionario de datos entidad AspNetuserRoles

Entidad: Vehículo.

ATRIBUTO	TIPO	NULO	OBSERVACIÓN
ID_VEHICULO	int	No	Id de vehículo (pk).
PLACA	varchar(10)	No	Placa del vehículo.
MODELO	varchar(20)	No	Modelo del vehículo.
COLOR	varchar(30)	No	Color del vehículo.
MARCA	varchar(50)	No	Marca del vehículo.
C_MOTOR	decimal(18)	No	Cilindraje del motor.
CAPACIDAD_ PASAJEROS	int	No	Capacidad del vehículo sin el chofer.
A_FABRICACION	int	No	Año de fabricación.
DESCRIPCION	varchar(10)	Si	Descripción del vehículo.
ESTADO	varchar(20)	No	Estado activo o inactivo.
ID_USUARIO	nvarchar(128)	No	Propietario o chofer (fk).

Tabla 34: Diccionario de datos entidad vehículo.

Entidad: Cliente.

ATRIBUTO	TIPO	NULO	OBSERVACIÓN
ID_CLIENTE	int	No	Id de cliente (pk).
ID_USUARIO	nvarchar(128)	No	Id usuario (fk).
CEDULA	varchar(13)	No	Cédula o RUC del cliente.
NOMBRE	varchar(50)	No	Nombre del cliente.
APELLIDO	varchar(50)	No	Apellido del cliente.
CALLE_PRINCIPAL	varchar(100)	No	Calle principal de residencia.
CALLE_SECUNDARIA	varchar(100)	No	Calle secundaria de residencia.
N_CASA	varchar(20)	Si	Numero de casa de residencia.
REFERENCIA	nvarchar(128)	Si	Referencia de residencia.
TELEFONO1	char(10)	No	Teléfono celular.
TELEFONO2	char(10)	Si	Teléfono opcional 2.
ID_CIUDAD	int	No	Id de ciudad (fk)

Tabla 35: Diccionario de datos entidad cliente.

Entidad: Reserva.

ATRIBUTO	TIPO	NULO	OBSERVACIÓN
ID_RESERVA	int	No	Id de reserva (pk).
ID_HORARIO	int	No	Id horario (fk).
ID_CLIENTE	int	No	Id cliente (fk).
N_ASIENTOS	int	No	Número de asientos.
N_MALETAS	int	Si	Número de equipaje.
CALLE_PRINCIPAL _ORIGEN	varchar(100)	No	Calle principal. De origen.
CALLE_ SECUNDARIA _ORIGEN	varchar(100)	No	Calle secundaria de origen.
NUMERO_ CASA_ORIGEN	char(10)	Si	Numero de casa de residencia.
REFERENCIA _ORIGEN	varchar(200)	Si	Referencia de origen.
TELEFONO _AXILIAR	char(10)	Si	Teléfono auxiliar del cliente.
OBSERVACION	varchar(150)	Si	Observación.
CALLE_PRINCIPAL _DESTINO	nvarchar(128)	Si	Calle principal de destino.
CALLE _SECUNDARIA _DESTINO	varchar(100)	No	Calle secundaria de destino
REFERENCIA_ DIR_DESTINO	varchar(200)	Si	Referencia del lugar de destino.
FECHA	datetime	No|	Fecha del viaje.
ESTADO	varchar(20)	No	Estado de la reserva.
LAT_ORIGEN	varchar(20)	Si	Latitud de origen.
LON_ORIGEN	varchar(20)	Si	Longitud de origen.
LAT_DESTINO	varchar(20)	Si	Latitud destino.
LON_DESTINO	varchar(20)	Si	Longitud destino.

Tabla 36: Diccionario de datos entidad reserva.

Entidad: Asignar_Reserva.

ATRIBUTO	TIPO	NULO	OBSERVACIÓN
ID_ASIGNACION	int	No	Id de asignación. (pk).
ID_RESERVA	int	No	Id reserva (fk).
ID_VEHICULO	int	No	Id vehículo (fk).

Tabla 37: Diccionario de datos entidad Asignar_Reserva

4.3.3. Interfaz plantilla máster.

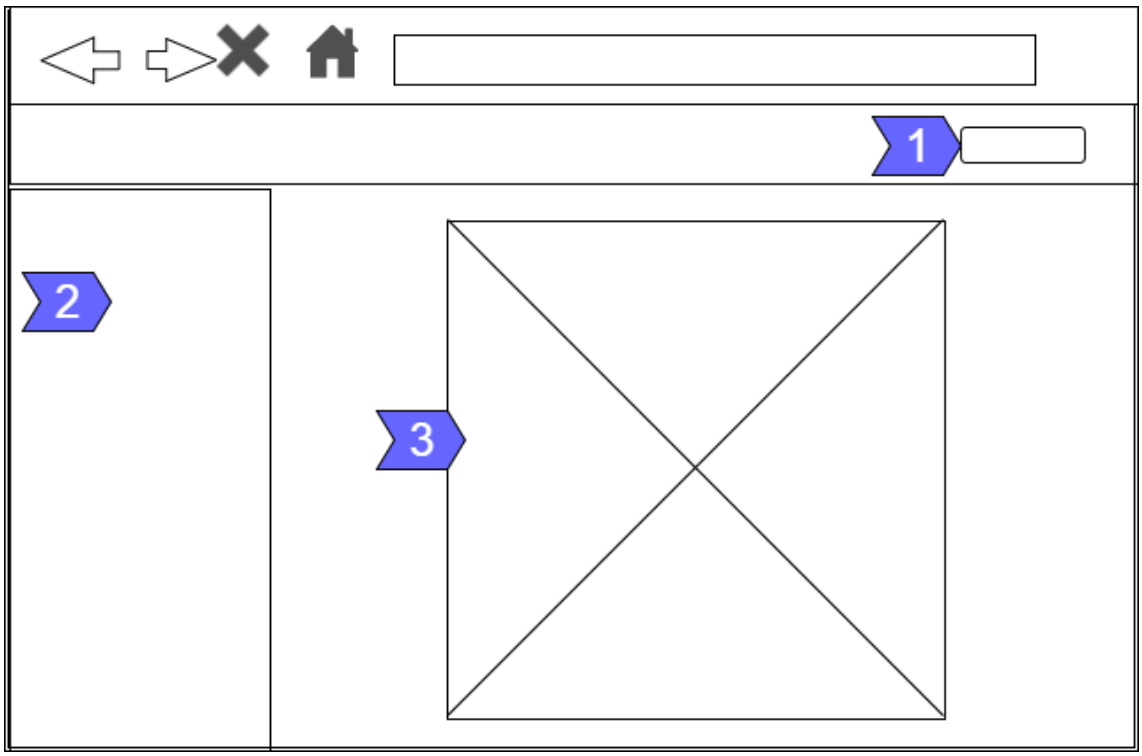


Figura 23: Página maestra.

1. Información de inicio de sesión: Permite observar quien inició sesión en el sitio web.
2. Menú: Contiene botones que redirecciona a las distintas páginas del sitio web, los menús se mostraran de acuerdo al rol que tenga el usuario.
3. Área: En esta área se mostrarán el contenido de cada una de las páginas.

4.3.4. Interfaz inicio de sesión.

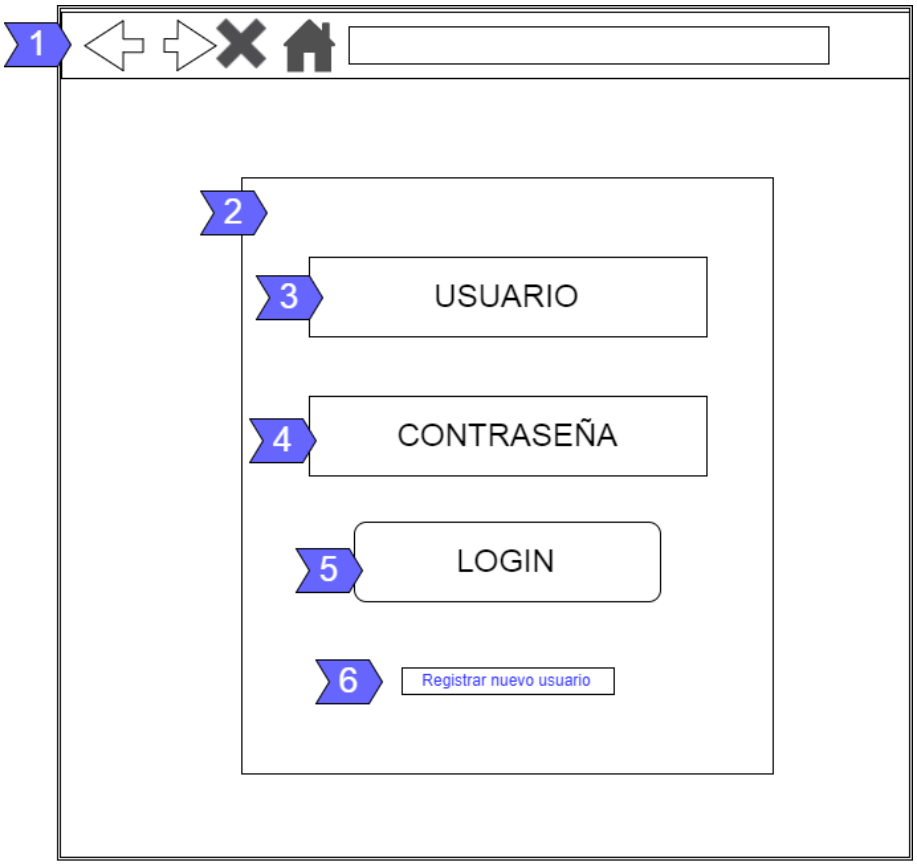


Figura 24: Página de inicio de sesión.

1. Barra de navegación: Componente propio de cada navegador.
2. Área de inicio de sesión: Área que contiene los elementos de inicio de sesión.
3. Ingreso de usuario: Ingreso del usuario registrado en el sistema.
4. Ingreso de contraseña: Ingreso de la contraseña de usuario registrado en el sistema.
5. Botón login: Botón para enviar la petición al servidor web.
6. Enlace: enlace que redirige a una página para registrar un nuevo usuario.

4.3.5. Formularios de mantenimiento.

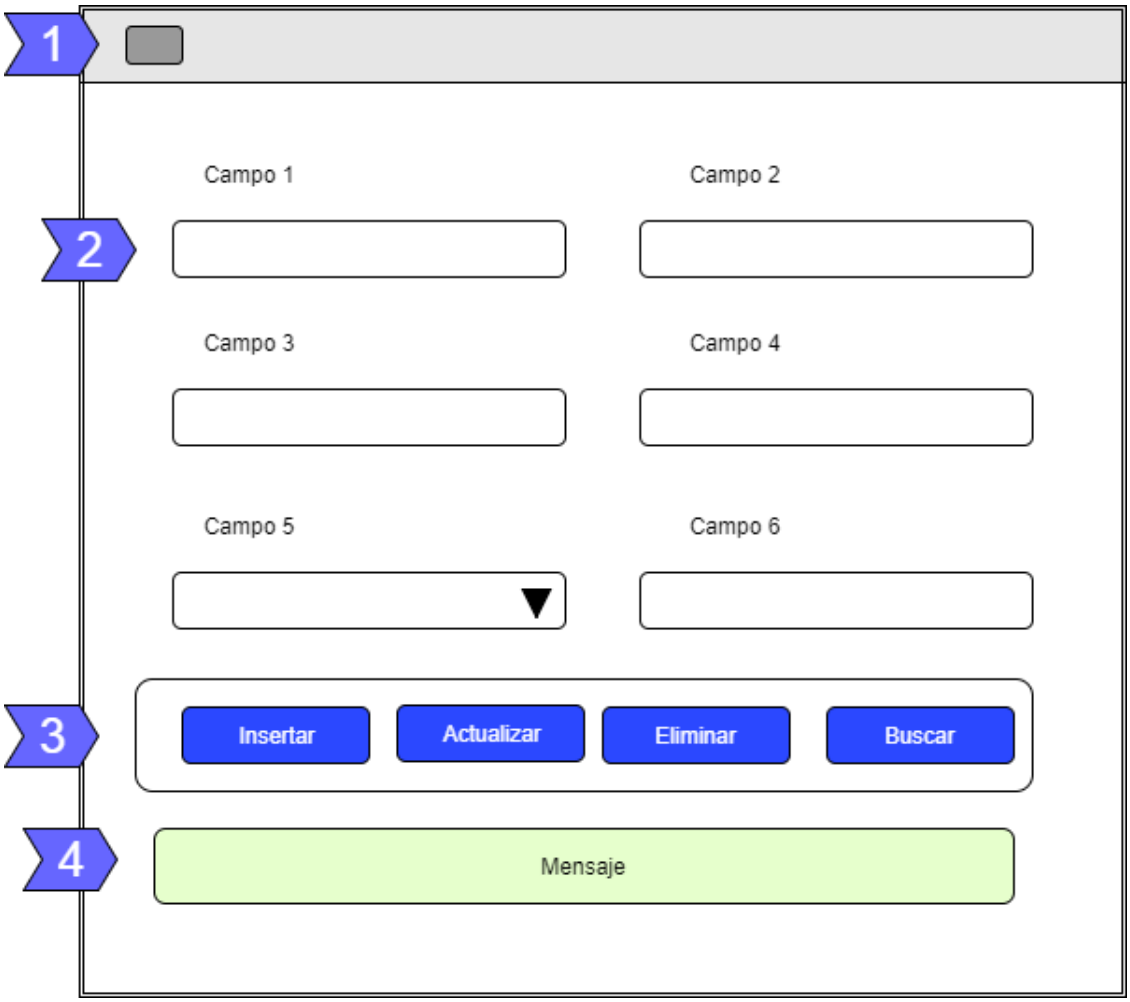


Figura 25: Formularios de mantenimiento.

1. Cabecera formulario: Contiene icono y descripción por cada formulario.
2. Componentes: Por cada página diferente se podrá encontrar Labels, Textboxs, Dropdow List, etc. dependiendo la necesidad de cada formulario.
3. Botones de control: En cada formulario se tendrá botones los mismos que cumplirán una función específica.
4. Área de mensaje: Se mostrarán los mensajes según los resultados de la acción ejecutada.

4.3.6. Área de visualización y búsqueda de información.

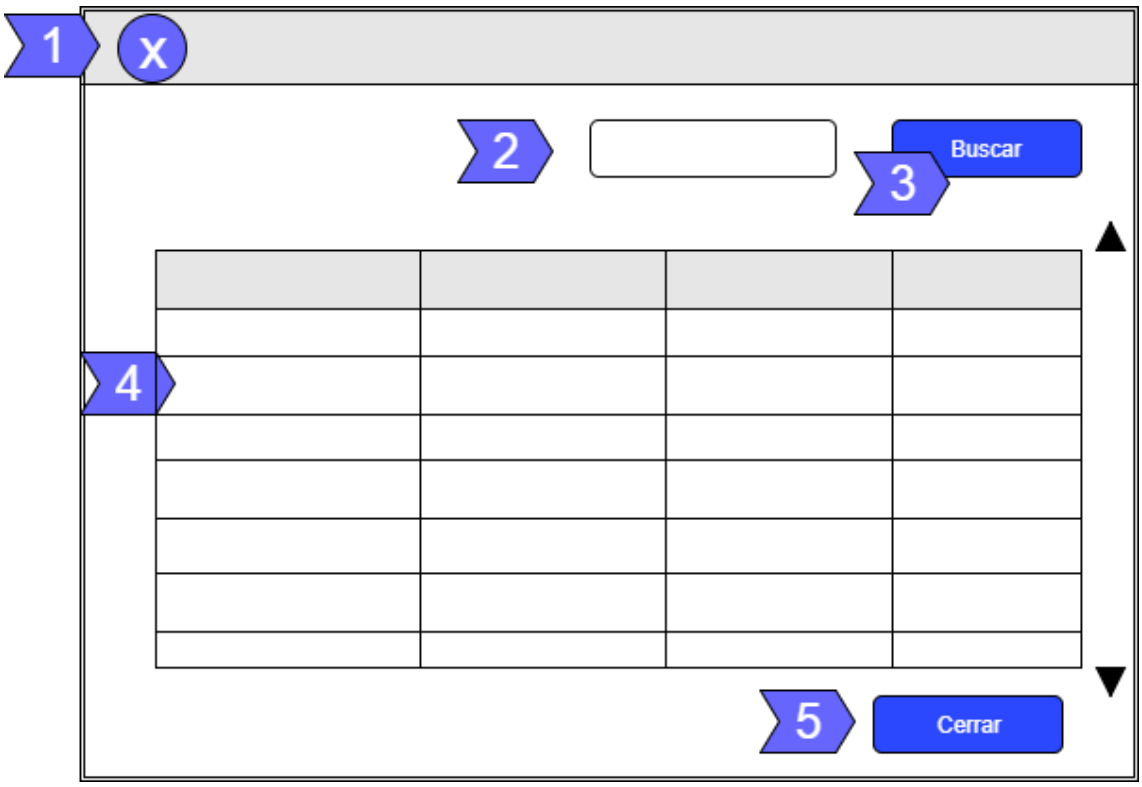


Figura 26: Área de visualización y búsqueda de información.

1. Cabecera del área: se tiene un botón para cerrar y el título del área.
2. Cuadro de texto: Cuadro de texto para filtrar la información por el campo más importante.
3. Botón: Botón para filtrar la información de acuerdo al texto a buscar.
4. Grid View: Muestra la información.

4.3.7. Arquitectura.

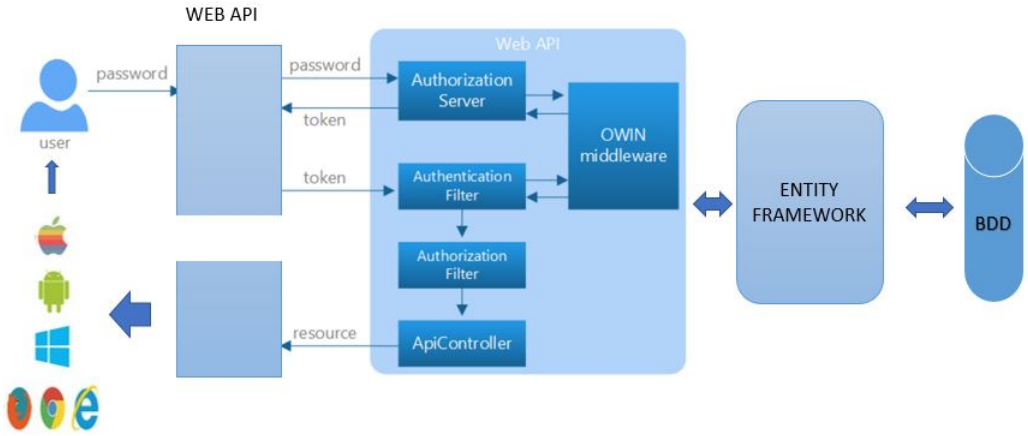


Figura 27: Arquitectura del sistema.

4.4. Implementación.

4.4.1. Creación de los servicios web.

Luego de la creación de la base de datos en Sql Server se procede con la creación de los servicios web.
Para la creación de los servicios web, se utilizó servicios web REST, los servicios web se desarrollaron con Visual Studio 2013 y Entity Framework.

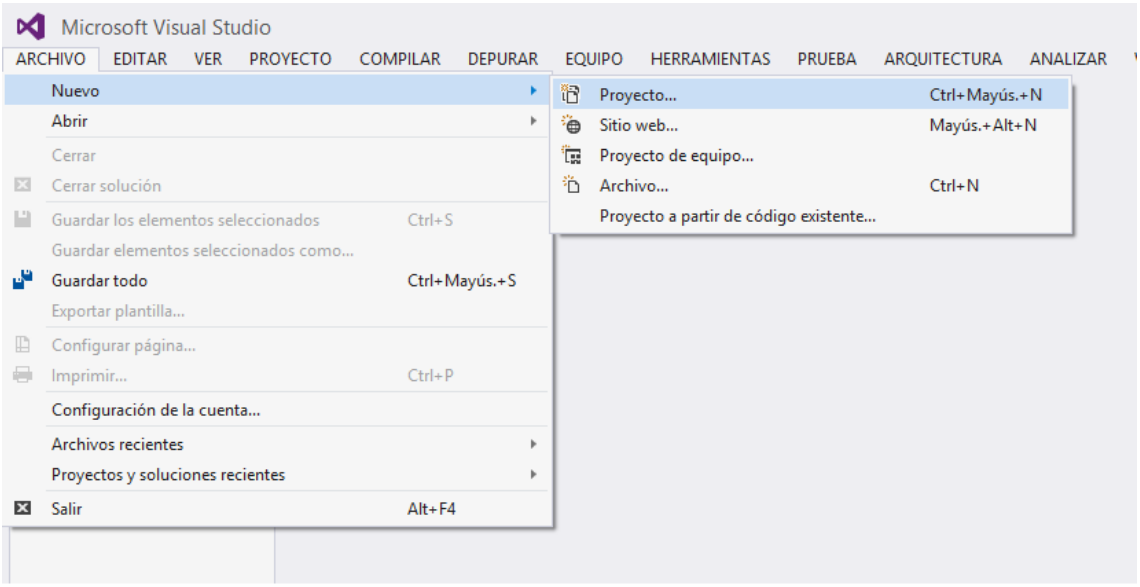


Figura 28: Creación de un nuevo proyecto.

Se crea un nuevo proyecto con Aplicación web ASP.NET.

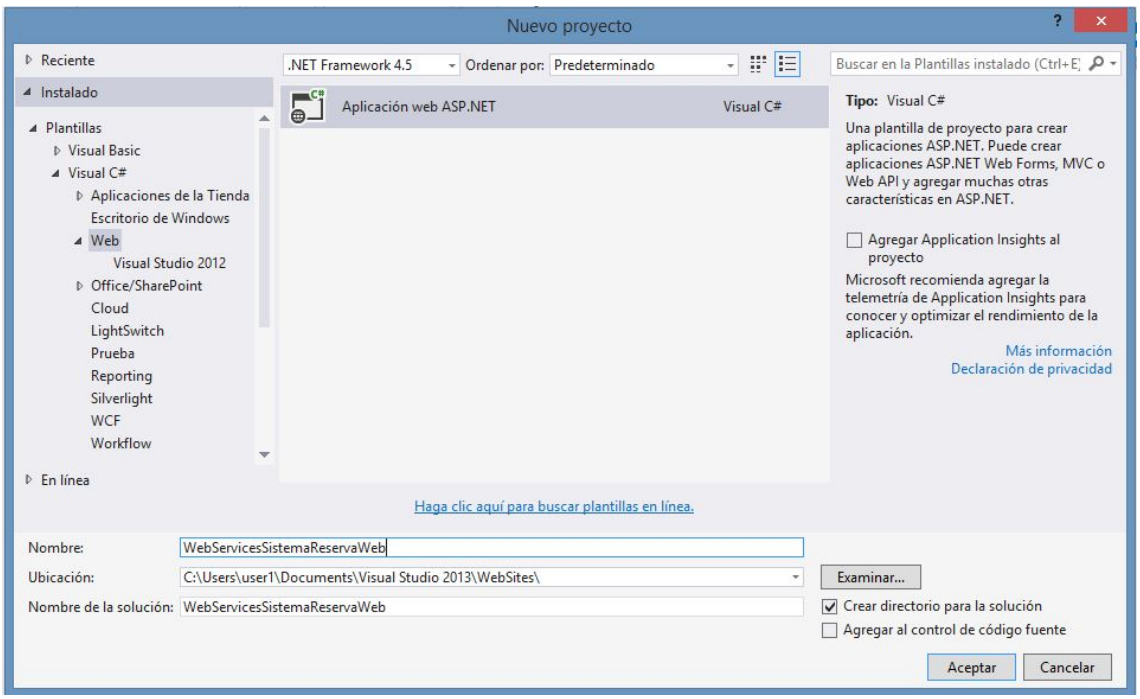


Figura 29: Seleccionar Aplicación web ASP.NET.

Seleccionar Web API después agregar carpetas y referencias principales para MVC (Modelo Vista Controlador) y Web API.

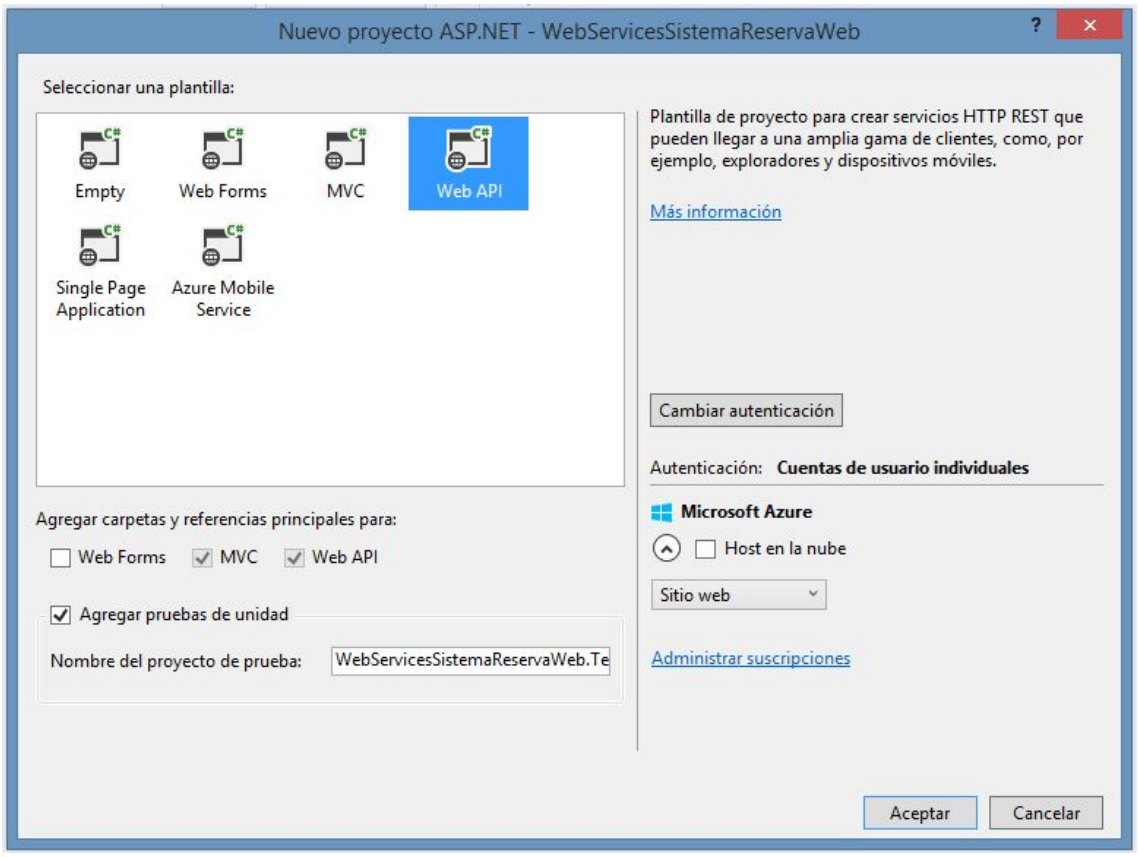


Figura 30: Seleccionar Web API MVC.

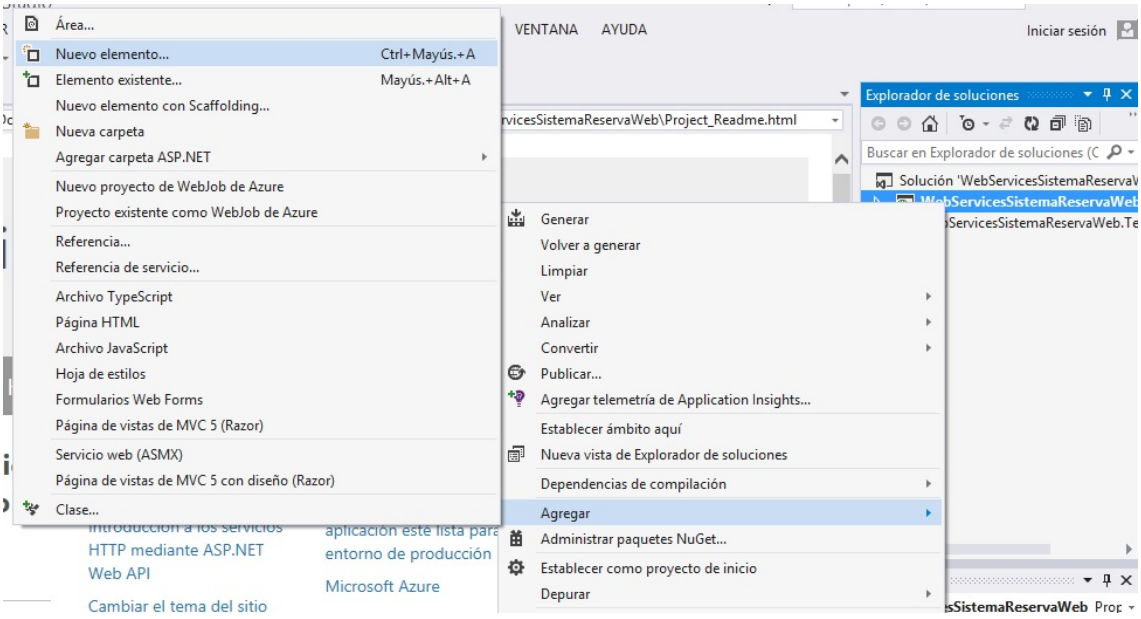


Figura 31: Agregar nuevo elemento

Agregar un nuevo elemento ADO.NET Entity Data Model, el cual se encargará de mapear todas las tablas y relaciones de la base de datos.

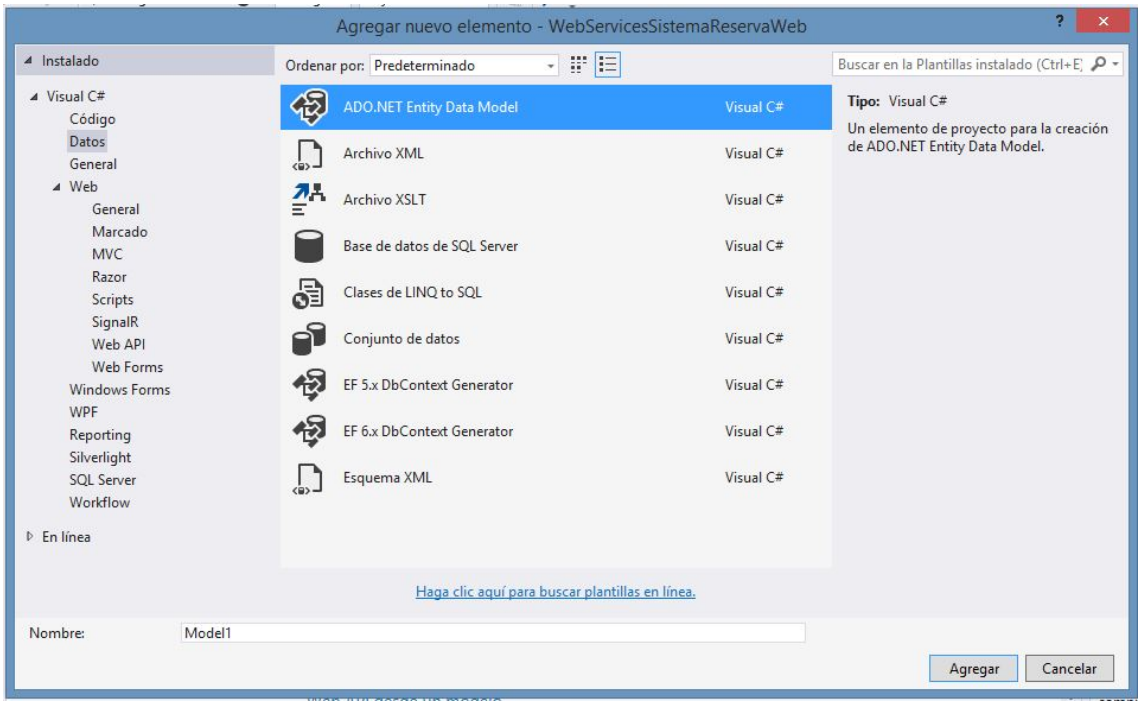


Figura 32: Agregar nuevo elemento ADO.NET Entity Data Model

Seleccionar que se diseñe el esquema desde una base de datos.

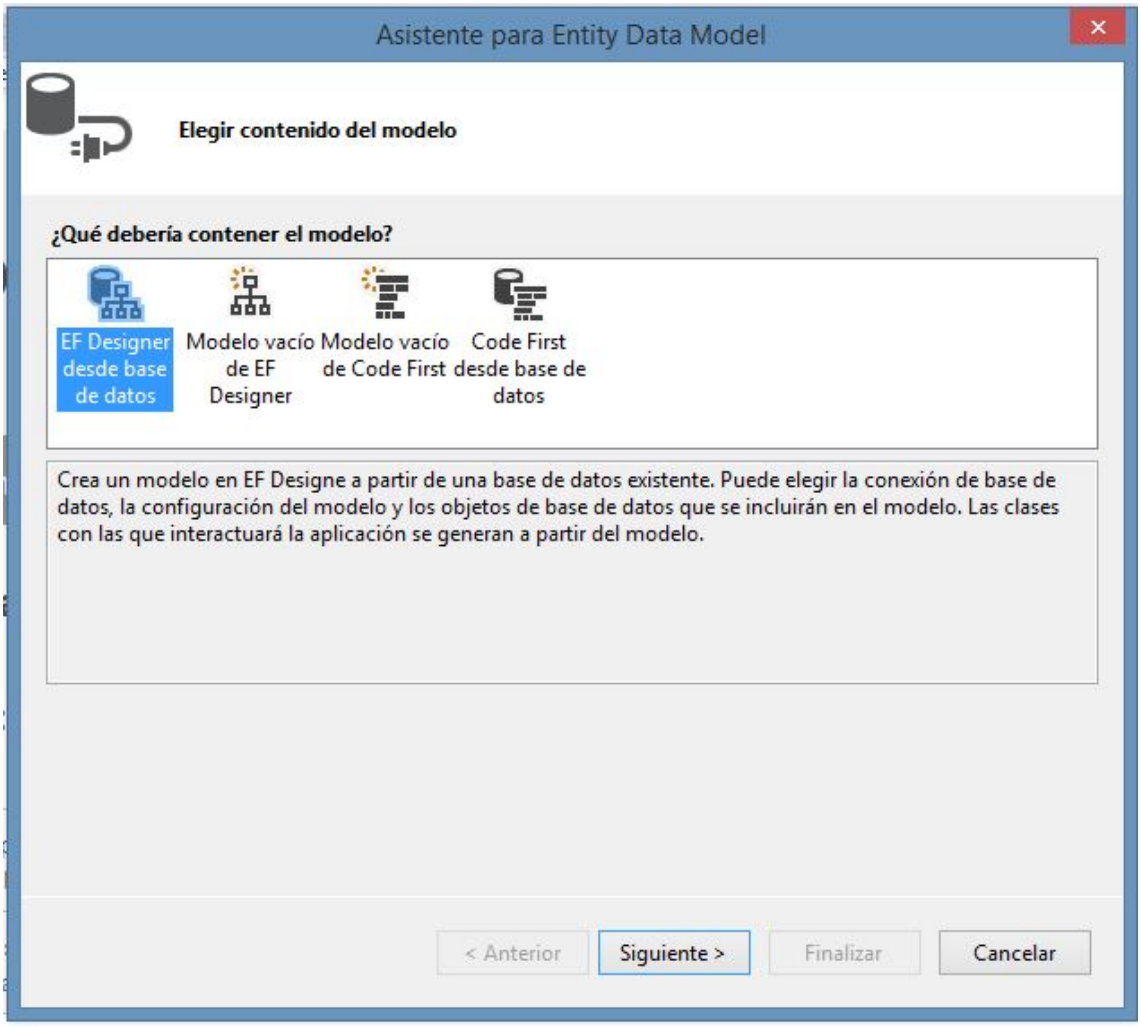


Figura 33: Seleccionar EF Designer desde base de datos

Configurar una conexión a la base de datos, dependiendo de la base de datos se debe elegir el cliente, ingresar nombre del servidor, usuario, contraseña y seleccionar la base de datos.

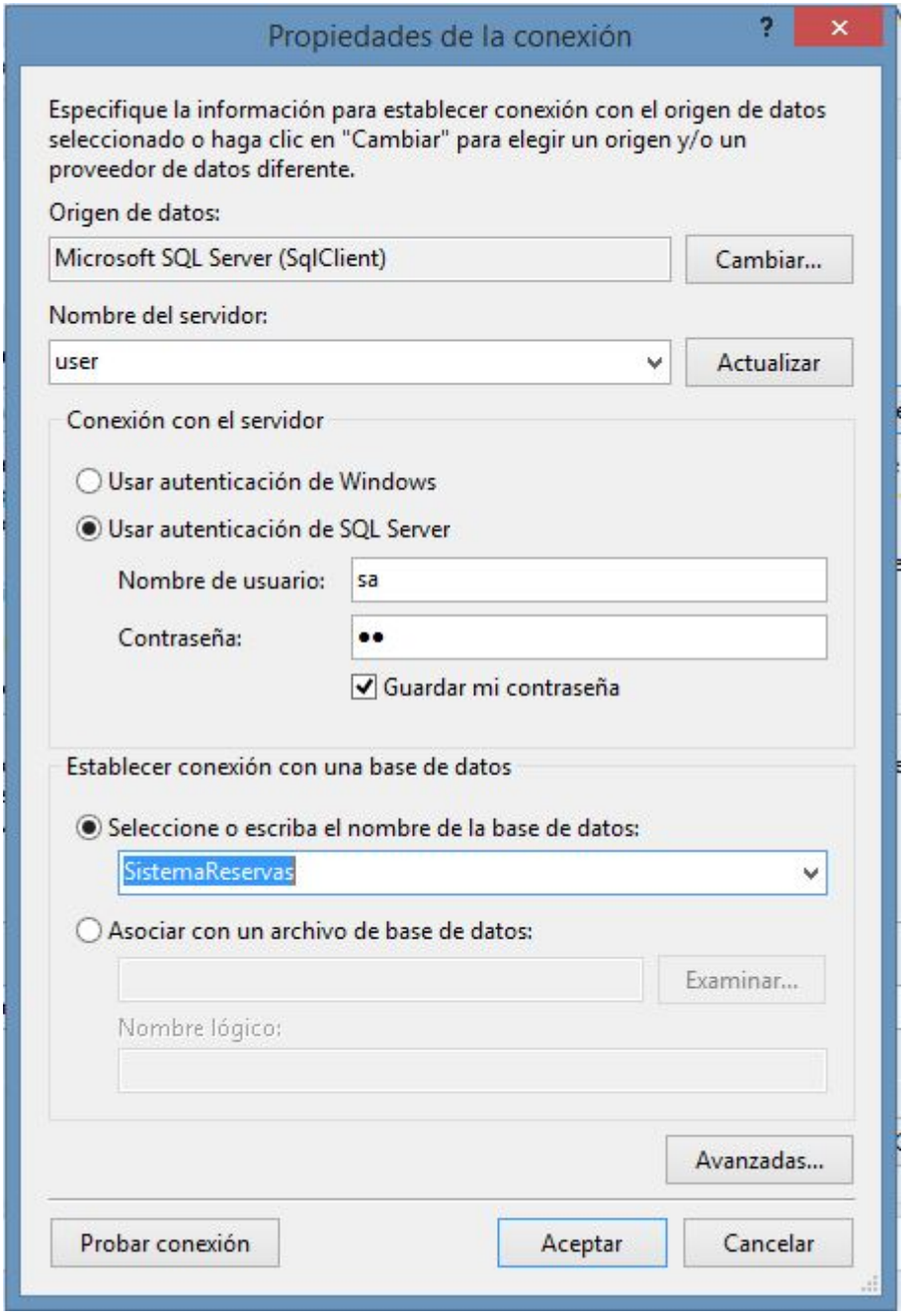


Figura 34: Configurar conexión a Base de Datos.

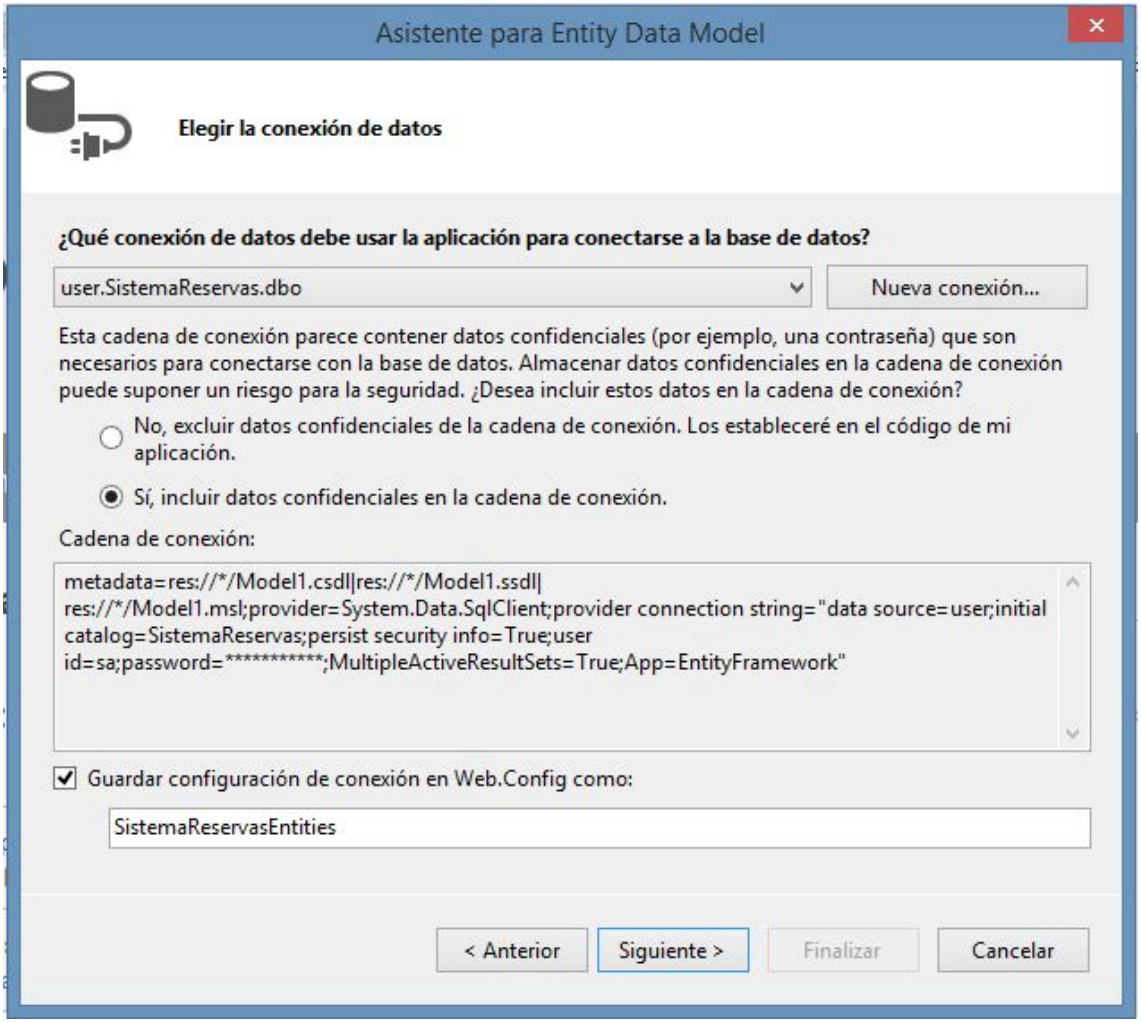


Figura 35: Guardar cadena de conexión.

Seleccionar los elementos, en este caso solo tablas.

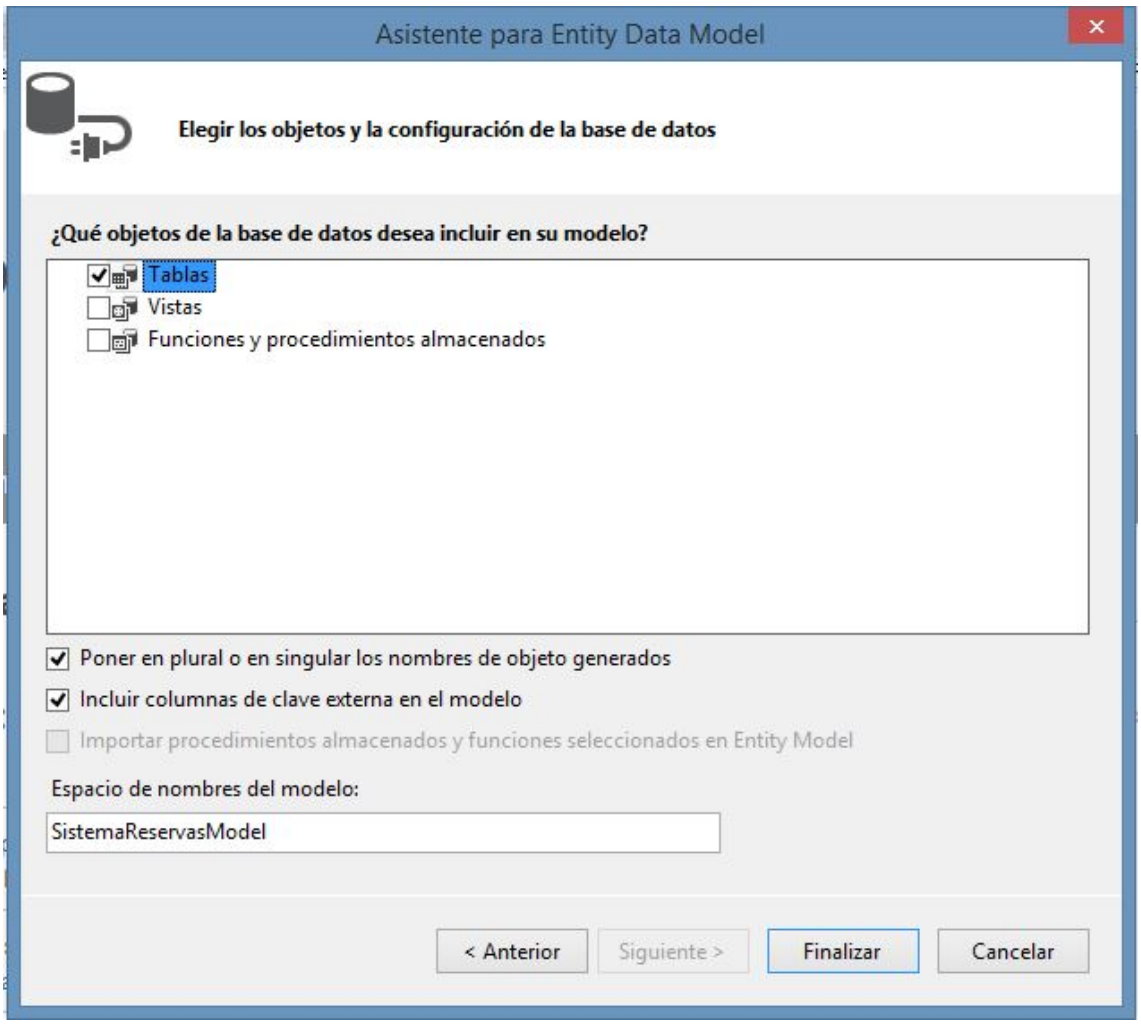


Figura 36: Seleccionar los objetos para EF Designer

Se muestra el diagrama que se diseña.

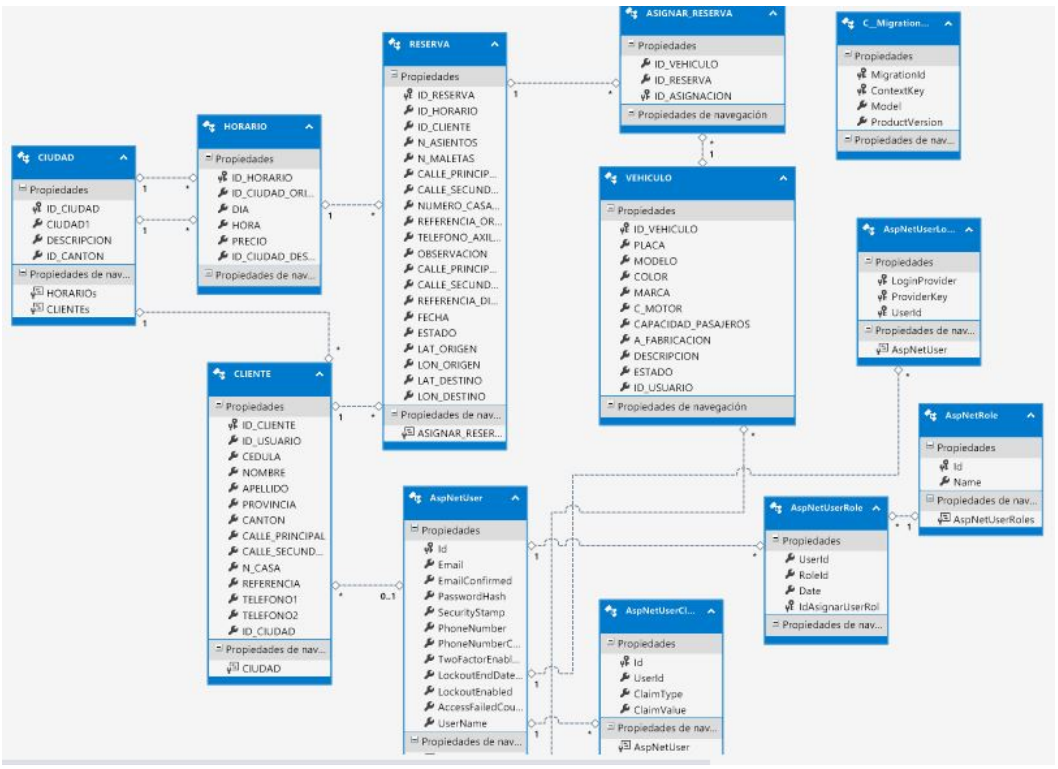


Figura 37: Diagrama generado con EF Designer

Agregar un nuevo elemento con Scaffolding.

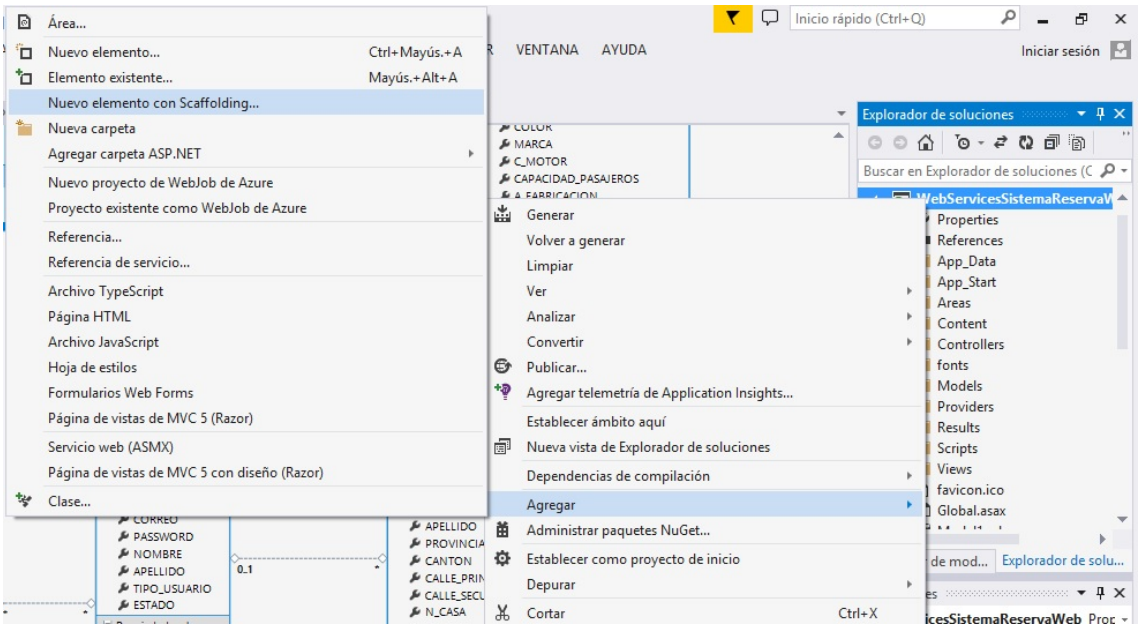


Figura 38: Agregar un nuevo elemento con Scaffolding.

Seleccionar controlador de web API 2 con acciones que usan Entity Framework, con esto se tendrá las acciones del servicio web tipo REST insertar, seleccionar,

actualizar y eliminar datos de la base de datos.

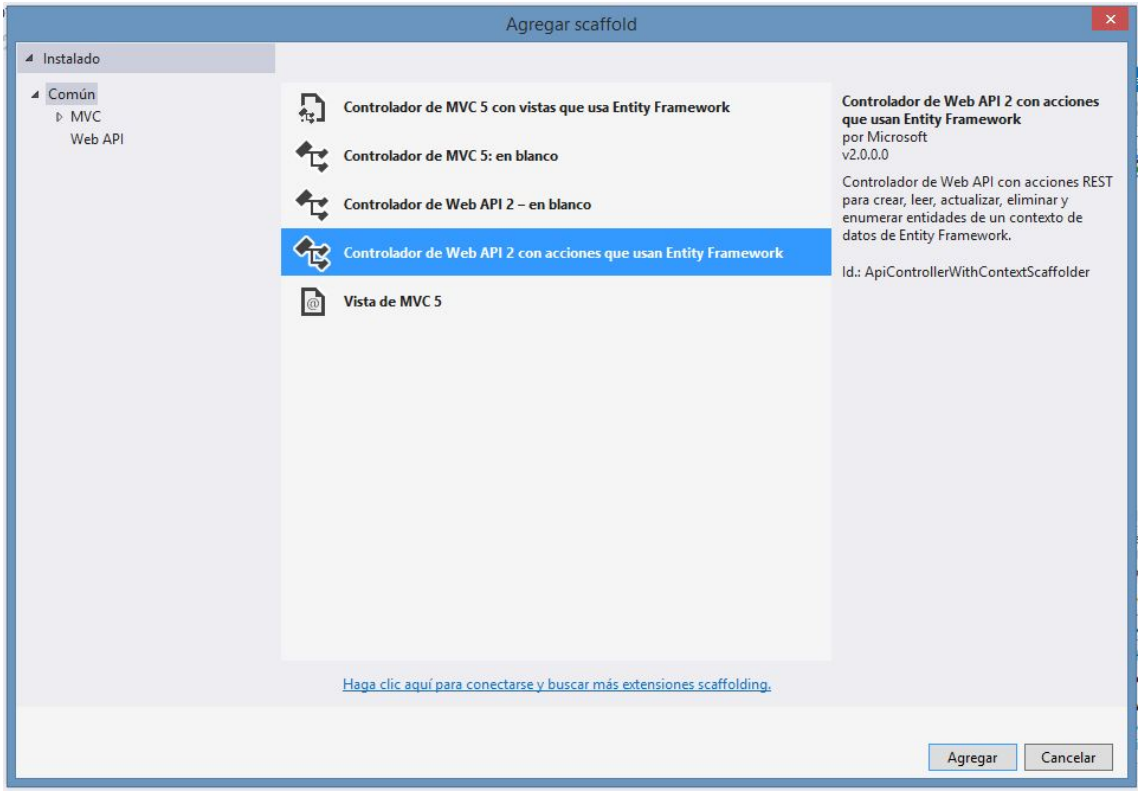


Figura 39: Seleccionar Controlador de Web API 2 con acciones que usan Entity Framework.

Agregar el controlador para cada entidad, primero seleccionar la clase del modelo, la clase contexto.

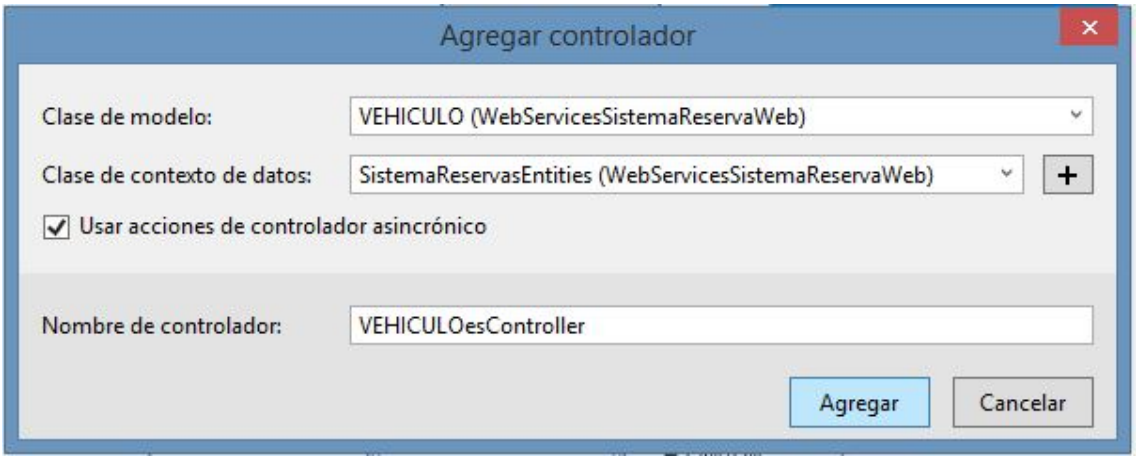


Figura 40: Agregar controlador para entidad generada con EF Designer.

Luego muestra la clase como se genera automáticamente con las distintas acciones del servicio web.

```
using System.Web.Http.Description;

namespace WebServicesSistemaReservaWeb
{
    0 referencias
    public class VEHICULOesController : ApiController
    {
        private SistemaReservasEntities db = new SistemaReservasEntities();

        // GET: api/VEHICULOes
        0 referencias
        public IQueryable<VEHICULO> GetVEHICULOes()
        {
            return db.VEHICULOes;
        }

        // GET: api/VEHICULOes/5
        [ResponseType(typeof(VEHICULO))]
        0 referencias
        public async Task<IHttpActionResult> GetVEHICULO(int id)...

        // PUT: api/VEHICULOes/5
        [ResponseType(typeof(void))]
        0 referencias
        public async Task<IHttpActionResult> PutVEHICULO(int id, VEHICULO vEHICULO)...

        // POST: api/VEHICULOes
        [ResponseType(typeof(VEHICULO))]
        0 referencias
```

Figura 41: Clase generada con los métodos del servicio web.

Ejecutar para ver que todo funcione perfectamente.

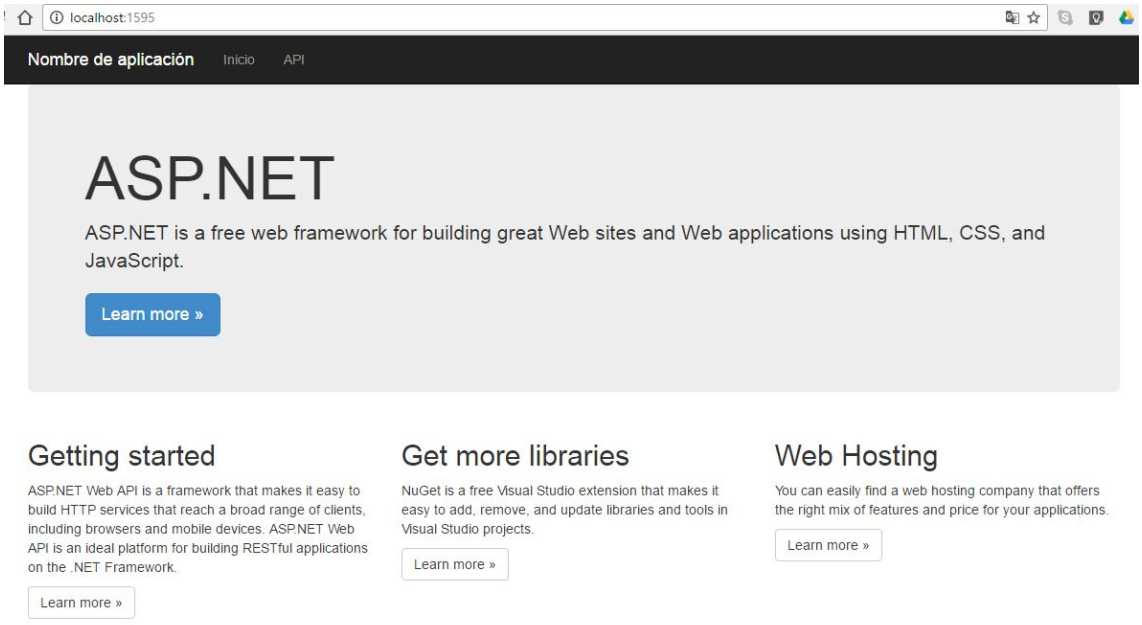


Figura 42: Verificación del funcionamiento web.

Con la ayuda de la herramienta Fiddler se puede hacer el test de los servicios web sin consumir en alguna aplicación, en este caso seleccionar tipo POST para insertar datos tipo JSON a la base de datos.

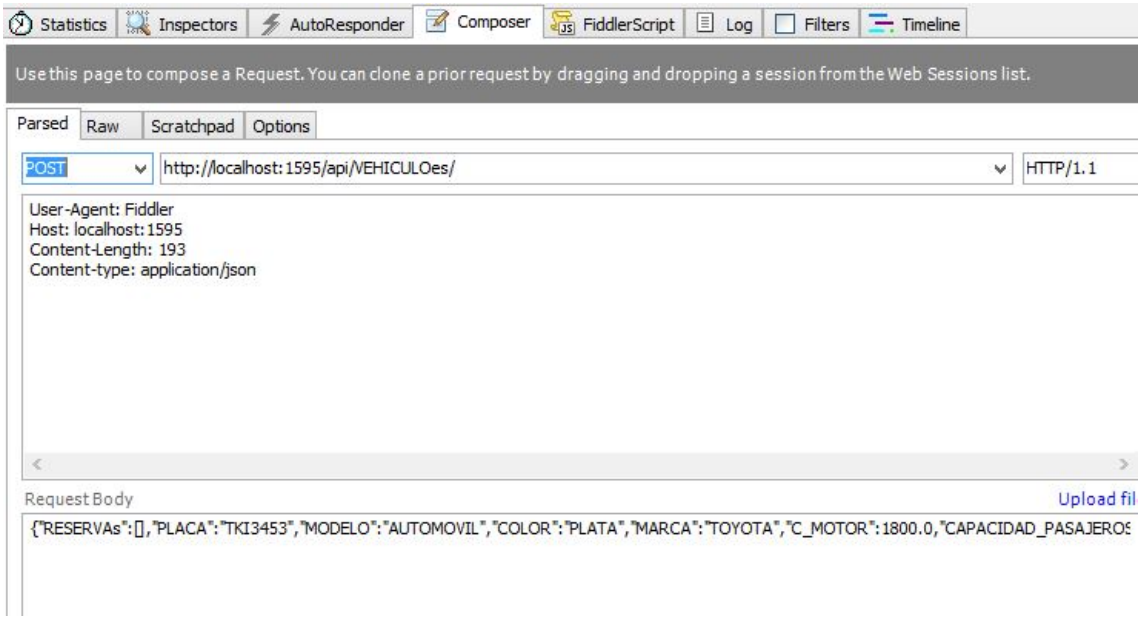


Figura 43: Pruebas del funcionamiento del servicio web Insert con Fiddler

Con GET para seleccionar los datos de la base de datos, en este caso getAll.

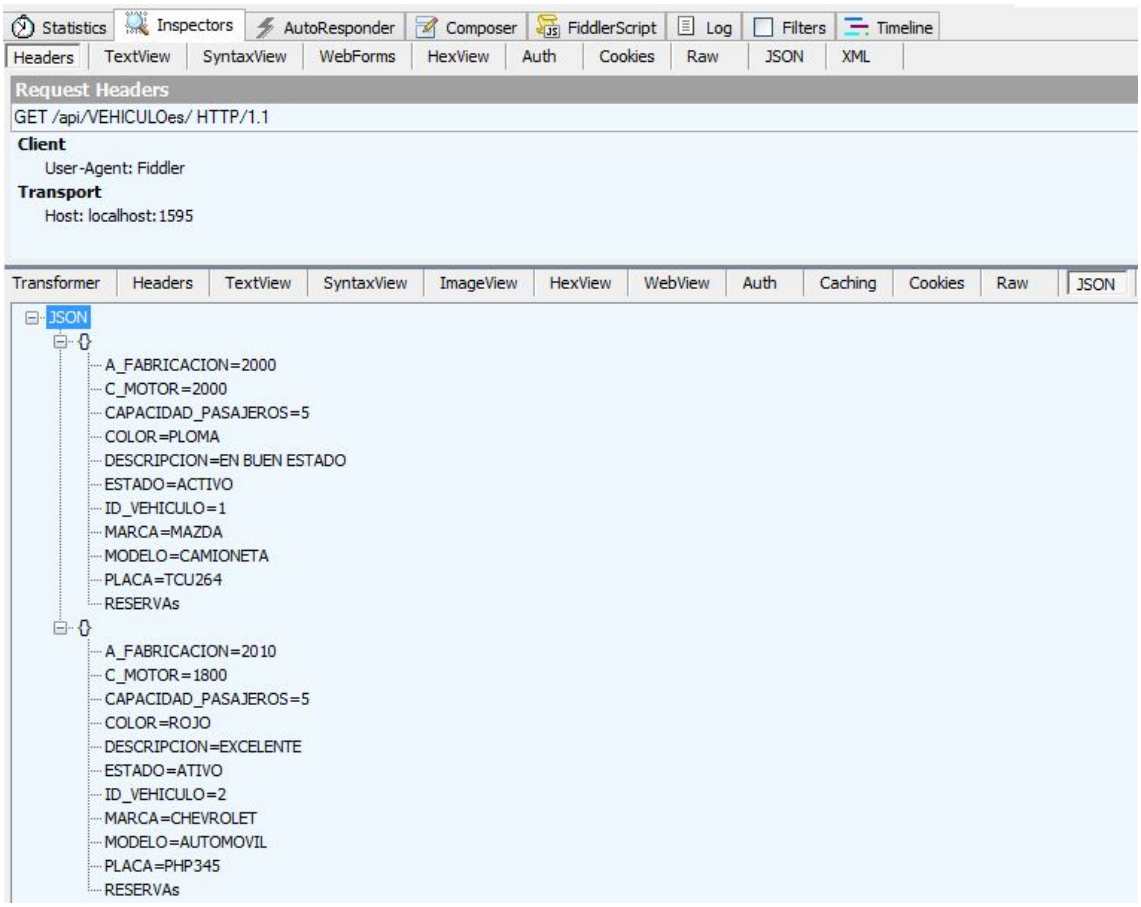


Figura 44: Resultado del servicio web getAll.

Prueba de getById.

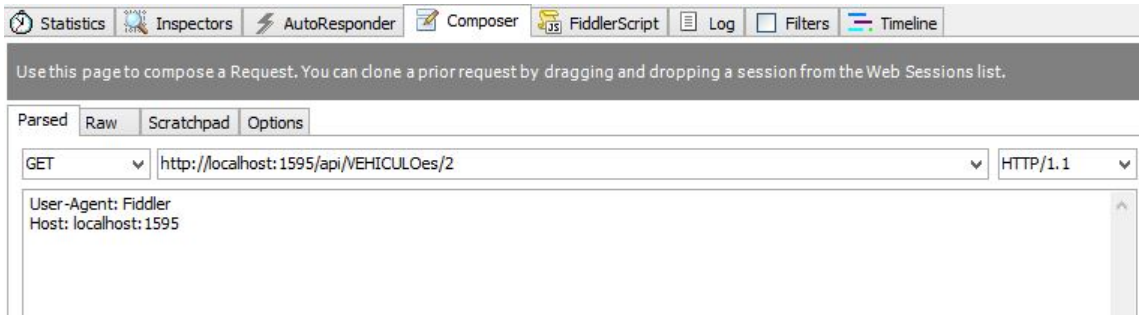


Figura 45: Pruba del servicio web getById

Prueba para actualizar un registro.

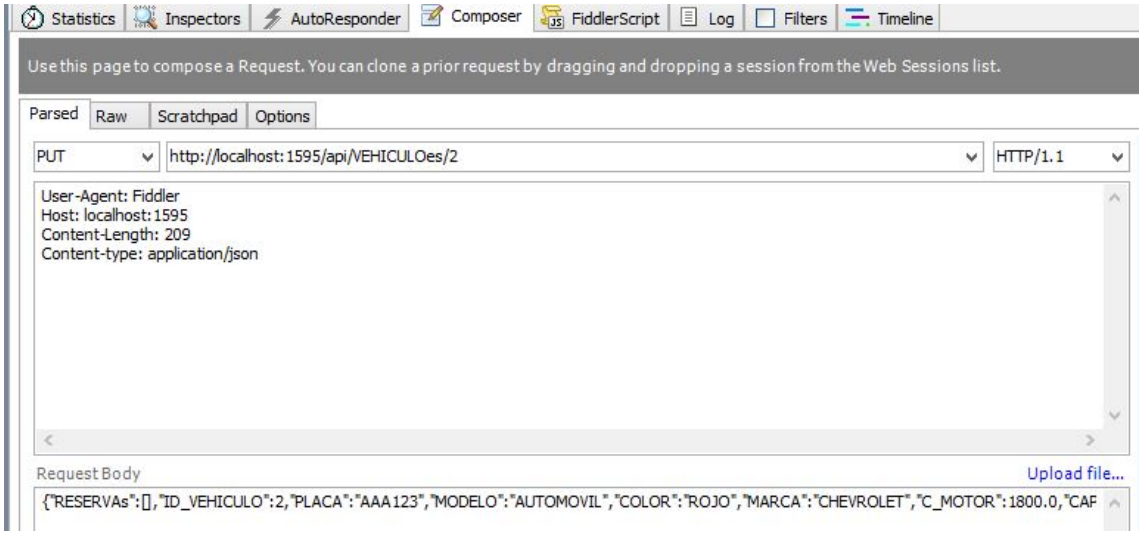


Figura 46: Prueba Update

Prueba para eliminar un registro.

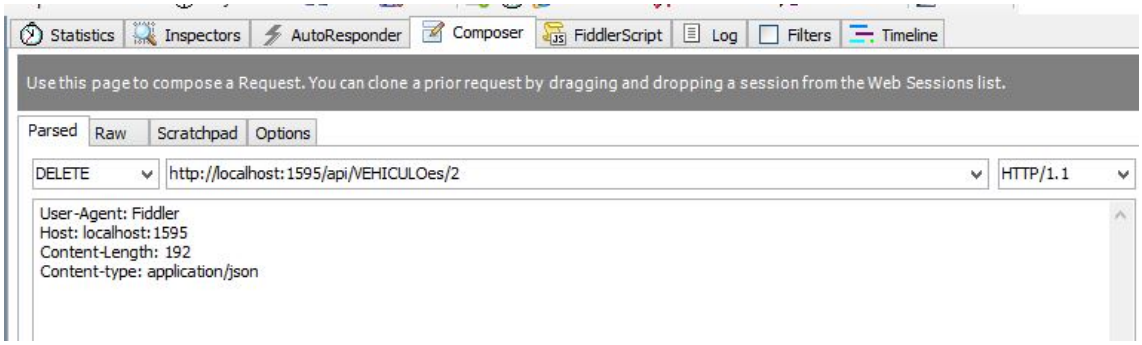


Figura 47: Prueba servicio web Delete.

Consumir los servicios web en ASP.NET.

Una vez creado los servicios web se procede a consumir, lo primero que se hacer es insertar un vehículo, para ello se necesita de una clase de tipo del servicio web que se vaya a consumir.

```
public class Clase_Vehiculo
{
        2 referencias
        public int ID_VEHICULO { get; set; }
        2 referencias
        public string PLACA { get; set; }
        2 referencias
        public string MODELO { get; set; }
        2 referencias
        public string COLOR { get; set; }
        2 referencias
        public string MARCA { get; set; }
        2 referencias
        public double C_MOTOR { get; set; }
        2 referencias
        public int CAPACIDAD_PASAJEROS { get; set; }
        2 referencias
        public int A_FABRICACION { get; set; }
        2 referencias
        public string DESCRIPCION { get; set; }
        2 referencias
        public string ESTADO { get; set; }
        2 referencias
        public string ID_USUARIO { get; set; }
}
```

Figura 48: Creación de la clase del servicio web.

Para insertar los datos por medio del servicio web se hace uso de la clase para serializar los datos en tipo json para eso se usa el ensamblado Newtonsoft.

```
var vh = new Clase_Vehiculo()
{
    PLACA = txtPlaca.Text.Trim(),
    MODELO = txtModelo.Text.Trim(),
    COLOR = txtColor.Text.Trim(),
    MARCA = txtMarca.Text.Trim(),
    C_MOTOR = Convert.ToInt32(txtCilindrajeMotor.Text.Trim()),
    CAPACIDAD_PASAJEROS = Convert.ToInt32(txtCapacidad.Text.Trim()),
    A_FABRICACION = Convert.ToInt32(txtAnioFabricacion.Text.Trim()),
    DESCRIPCION = txtDescripcion.Text.Trim(),
    ESTADO = cbEstado.SelectedItem.Text,
    ID_USUARIO=conf.getIduserByUsername(txtPropietario.Text.Trim())
};
var json = Newtonsoft.Json.JsonConvert.SerializeObject(vh);
```

Figura 49: Serializar los datos.

Crear la estructura que debe tener un servicio web, como son el uri, verb, header que contiene el token y body en donde estarán los datos que se van a serializar.

```
// Uri
   WebRequest request = WebRequest.Create(Configuraciones.url + "/api/VEHICULOes");
//Verb
   request.Method = "POST";
//Header
   //Token
   request.Headers.Add("Authorization", "Bearer " + Session["tokenSession"].ToString());
//Body
   byte[] byteArray = Encoding.UTF8.GetBytes(json);
   request.ContentType = "application/json";
   request.ContentLength = byteArray.Length;
   // Get the request stream.
   Stream dataStream = request.GetRequestStream();
   // Write the data to the request stream.
   dataStream.Write(byteArray, 0, byteArray.Length);
   // Close the Stream object.
   dataStream.Close();
```

Figura 50: Request del servicio web,

Se obtiene el resultado o response del request, en este caso se obtiene el código 200 que significa que se insertaron los datos correctamente u otros códigos como el 500 que significa lo contrario.

```
// Obtenemos el resultado.
WebResponse response = request.GetResponse();
// Se muestra el estado.
lblEstado.Text = (((HttpWebResponse)response).StatusDescription);
// Obtenemos el contenido del resultado.
dataStream = response.GetResponseStream();
// Accedemos al contenido.
StreamReader reader = new StreamReader(dataStream);
// Se lee el contenido.
string responseFromServer = reader.ReadToEnd();
// Se cierra el la lectura.
reader.Close();
dataStream.Close();
response.Close();
```

Figura 51: Response del Request.

Para seleccionar los datos y mostrarlos en un listo se debe hacer de la siguiente forma:

```
//uri
WebRequest request = WebRequest.Create(Configuraciones.url+"/api/VEHICULOes");
// verb
request.Method = "GET";
//header
request.ContentType = "application/json";

request.Headers.Add("Authorization", "Bearer " + Session["tokenSession"].ToString());
//body

//Response
WebResponse response = request.GetResponse();
Stream dataStream = response.GetResponseStream();
StreamReader reader = new StreamReader(dataStream);
// Leer el contenido
string responseFromServer = reader.ReadToEnd();
//Deserializamos el contenido
List<Clase_Vehiculo> result = JsonConvert.DeserializeObject<List<Clase_Vehiculo>>(responseFromServer);
// Convertimos a un datatable la informacón para mostrar un gridView
DataTable dt = new DataTable();
dt = metodosVarios.ToDataTable<Clase_Vehiculo>(result);
//Mostramos el resultado
dgvVehiculo.DataSource = dt;
dgvVehiculo.DataBind();
```

Figura 52: Mostrar los datos en un grid view

Para actualizar los datos de un servicio web, es la misma estructura como para insertar, lo único que se diferencia es el VERB y el URI.

```
var vh = new Clase_Vehiculo()
{
    ID_VEHICULO = Convert.ToInt32(txtId.Text.Trim()),
    PLACA = txtPlaca.Text.Trim(),
    MODELO = txtModelo.Text.Trim(),
    COLOR = txtColor.Text.Trim(),
    MARCA = txtMarca.Text.Trim(),
    C_MOTOR = Convert.ToInt32(txtCilindrajeMotor.Text.Trim()),
    CAPACIDAD_PASAJEROS = Convert.ToInt32(txtCapacidad.Text.Trim()),
    A_FABRICACION = Convert.ToInt32(txtAnioFabricacion.Text.Trim()),
    DESCRIPCION = txtDescripcion.Text.Trim(),
    ESTADO = cbEstado.SelectedItem.Text,
    ID_USUARIO = conf.getIduserByUsername(txtPropietario.Text)
};
var json = Newtonsoft.Json.JsonConvert.SerializeObject(vh);

// URI
WebRequest request = WebRequest.Create(Configuraciones.url + "/api/VEHICULOes" + "/" + txtId.Text);
// VERB
request.Method = "PUT";
// TOKEN
request.Headers.Add("Authorization", "Bearer " + Session["tokenSession"].ToString());
//BODY
// byte[] byteArray = Encoding.UTF8.GetBytes(postData);
byte[] byteArray = Encoding.UTF8.GetBytes(json);
```

Figura 53: Actualizar los datos.

Para eliminar un registro por medio de un servicio web se debe hacer de la siguiente manera.

```
var vh = new Clase_Vehiculo()
{
    ID_VEHICULO = Convert.ToInt32(txtId.Text.Trim())

};
var json = Newtonsoft.Json.JsonConvert.SerializeObject(vh);
// URI
WebRequest request = WebRequest.Create(Configuraciones.url + "/api/VEHICULOes" + "/" + txtId.Text);
// VERB
request.Method = "DELETE";
// TOKEN
request.Headers.Add("Authorization", "Bearer " + Session["tokenSession"].ToString());
// BODY
byte[] byteArray = Encoding.UTF8.GetBytes(json);
// Set the ContentType property of the WebRequest.
request.ContentType = "application/json";
// Set the ContentLength property of the WebRequest.
request.ContentLength = byteArray.Length;
// Get the request stream.
Stream dataStream = request.GetRequestStream();
// Write the data to the request stream.
dataStream.Write(byteArray, 0, byteArray.Length);
// Close the Stream object.
dataStream.Close();
```

Figura 54: Eliminar un dato.

Consumir los servicios web en sistema operativo android.

Para consumir los servicios web en el sistema operativo Android se debe tener acceso al servidor de IIS, para eso se hace una prueba simple ingresando a la URL

en el navegador móvil.

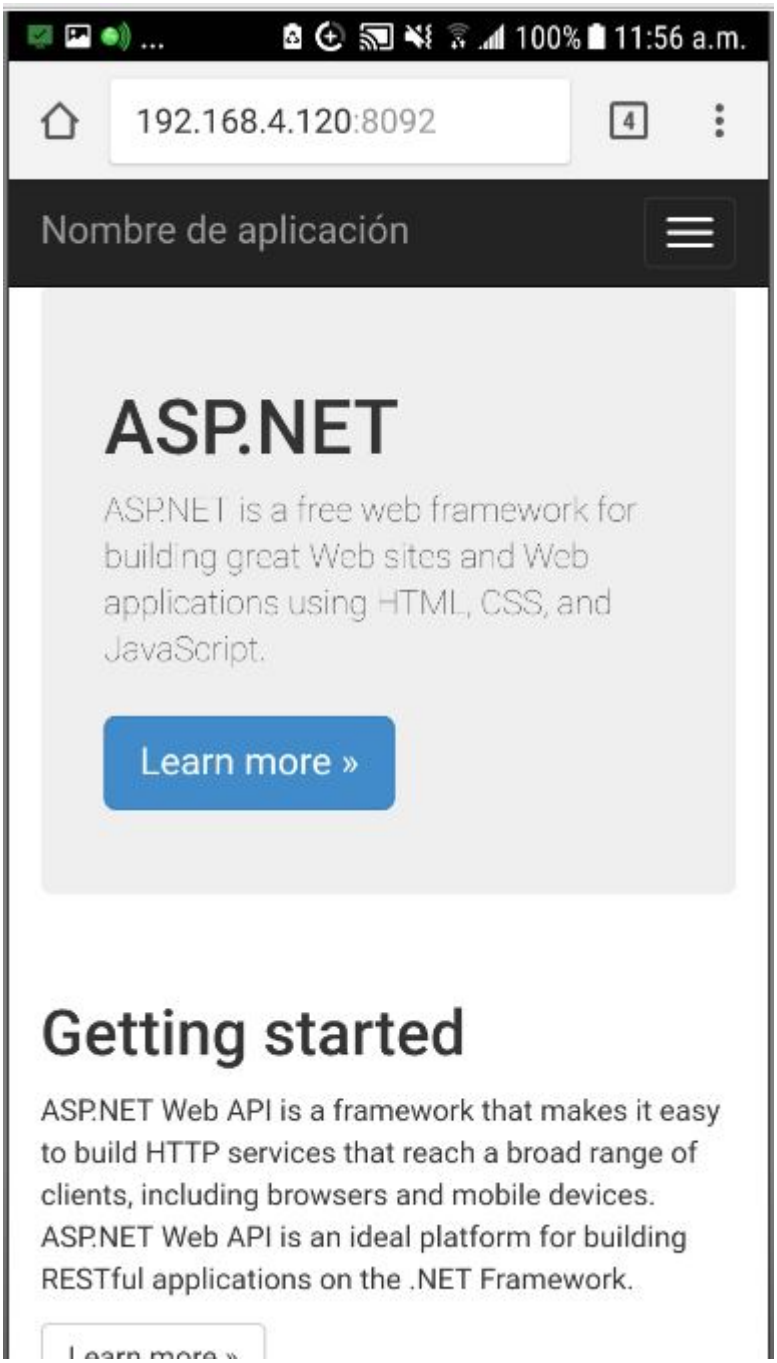


Figura 55: Prueba conexión a IIS

En android con la ayuda de la librería retrofid permite hacer la conexión hacia un web servicio tipo rest de una manera fácil, para esto después de agregar las librerías, se agrega una clase para cada servicio web, en este caso la clase del token.

```
public class OAuthToken {
    private String access_token;
    private String token_type;
    private int expires_in;
    private String userName;
    private String issued;
    private String expires;

    public String getAccess_token() { return access_token; }
    public void setAccess_token(String access_token) { this.access_token = access_token; }
    public String getToken_type() { return token_type; }
    public void setToken_type(String token_type) { this.token_type = token_type; }
    public int getExpires_in() { return expires_in; }

    public void setExpires_in(int expires_in) { this.expires_in = expires_in; }

    public String getUserName() { return userName; }

    public void setUserName(String userName) { this.userName = userName; }

    public String getIssued() { return issued; }

    public void setIssued(String issued) { this.issued = issued; }

    public String getExpires() { return expires; }

    public void setExpires(String expires) { this.expires = expires; }
```

Figura 56: Clase token.

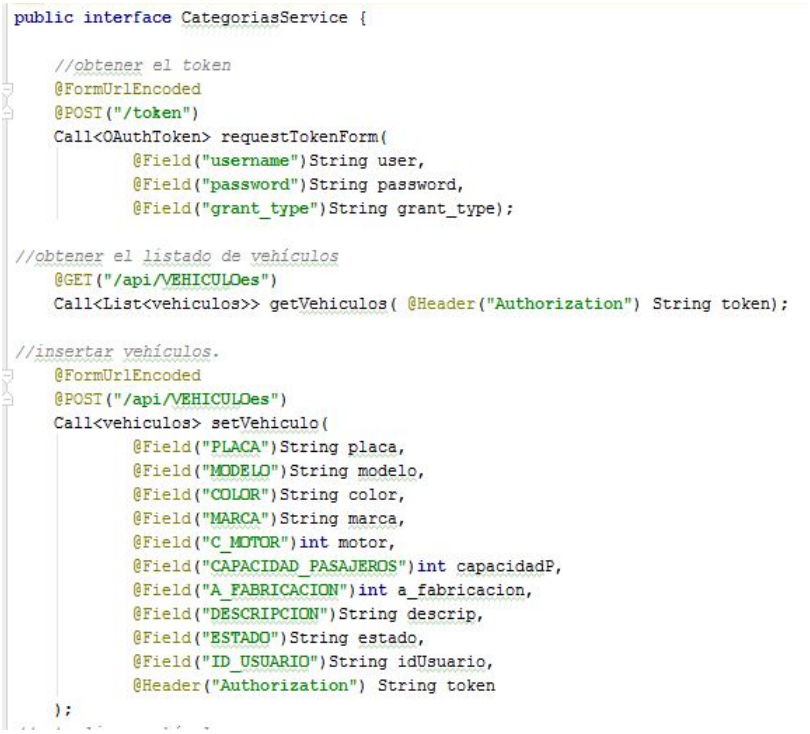
```java
public class vehiculos {

    private int ID_VEHICULO;
    private String PLACA;
    private String MODELO;
    private String COLOR;
    private String MARCA;
    private double C_MOTOR;
    private int CAPACIDAD_PASAJEROS;
    private int A_FABRICACION;
    private String DESCRIPCION;
    private String ESTADO;
    private String ID_USUARIO;

    public int getID_VEHICULO() { return ID_VEHICULO; }
    public void setID_VEHICULO(int ID_VEHICULO) { this.ID_VEHICULO = ID_VEHICULO; }
    public String getPLACA() { return PLACA; }
    public void setPLACA(String PLACA) { this.PLACA = PLACA; }
    public String getMODELO() { return MODELO; }
    public void setMODELO(String MODELO) { this.MODELO = MODELO; }
    public String getCOLOR() { return COLOR; }
    public void setCOLOR(String COLOR) { this.COLOR = COLOR; }
    public String getMARCA() { return MARCA; }
    public void setMARCA(String MARCA) { this.MARCA = MARCA; }
    public double getC_MOTOR() { return C_MOTOR; }
    public void setC_MOTOR(double c_MOTOR) { C_MOTOR = c_MOTOR; }
    public int getCAPACIDAD_PASAJEROS() { return CAPACIDAD_PASAJEROS; }
    public void setCAPACIDAD_PASAJEROS(int CAPACIDAD_PASAJEROS) {
        this.CAPACIDAD_PASAJEROS = CAPACIDAD_PASAJEROS;    }
    public int getA_FABRICACION() { return A_FABRICACION; }
```

Figura 57: Clase vehículo.

Se crea una clase de tipo interfaz que permitirá hacer la petición a cada servicio web, en la cual consta de dos métodos uno para obtener el token de sesión, y la segunda para obtener el listado de vehículos, que será necesario enviar el token de sesión en la cabecera de la petición para obtener el resultado



```
public interface CategoriasService {

    //obtener el token
    @FormUrlEncoded
    @POST("/token")
    Call<OAuthToken> requestTokenForm(
            @Field("username")String user,
            @Field("password")String password,
            @Field("grant_type")String grant_type);

//obtener el listado de vehículos
    @GET("/api/VEHICULOes")
    Call<List<vehiculos>> getVehiculos( @Header("Authorization") String token);

//insertar vehículos.
    @FormUrlEncoded
    @POST("/api/VEHICULOes")
    Call<vehiculos> setVehiculo(
            @Field("PLACA")String placa,
            @Field("MODELO")String modelo,
            @Field("COLOR")String color,
            @Field("MARCA")String marca,
            @Field("C_MOTOR")int motor,
            @Field("CAPACIDAD_PASAJEROS")int capacidadP,
            @Field("A_FABRICACION")int a_fabricacion,
            @Field("DESCRIPCION")String descrip,
            @Field("ESTADO")String estado,
            @Field("ID_USUARIO")String idUsuario,
            @Header("Authorization") String token
    );
```

Figura 58: Clase tipo interfaz parte superior.

```
//actualizar vehículos
    @FormUrlEncoded
    @PUT("/api/VEHICULOes/{ID}")
    Call<vehiculos> actualizarVehiculo(
            @Path("ID") int id,
            @Field("PLACA")String placa,
            @Field("MODELO")String modelo,
            @Field("COLOR")String color,
            @Field("MARCA")String marca,
            @Field("C_MOTOR")int motor,
            @Field("CAPACIDAD_PASAJEROS")int capacidadP,
            @Field("A_FABRICACION")int a_fabricacion,
            @Field("DESCRIPCION")String descrip,
            @Field("ESTADO")String estado,
            @Field("ID_USUARIO")String idUsuario,
            @Header("Authorization") String token
    );
//eliminar vehículos.
    @FormUrlEncoded
    @DELETE("/api/VEHICULOes/{ID}")
    Call<vehiculos> eliminarVehiculo(
            @Path("ID") int id,
            @Header("Authorization") String token
    );

}
```

Figura 59: Clase tipo interfaz parte inferior

Se crea una clase adaptador que en donde se tendrá la configuración de la dirección del servio web, el interceptor, el cliente y la conversión del texto tipo json del servicio web.

```
public class RedemnorteApiAdapter {
    private static CategoriasService API_SERVICE;

    public static CategoriasService getApiService()
    {
        HttpLoggingInterceptor logging = new HttpLoggingInterceptor();
        logging.setLevel(HttpLoggingInterceptor.Level.BODY);
        OkHttpClient.Builder httClient = new OkHttpClient.Builder();
        httClient.addInterceptor(logging);
        String baseUrl="http://192.168.4.120:8092";
        if(API_SERVICE==null)
        {
            Retrofit retrofit = new Retrofit.Builder()
                    .baseUrl(baseUrl)
                    .addConverterFactory(GsonConverterFactory.create())
                    .client(httClient.build())
                    .build();
            API_SERVICE=retrofit.create(CategoriasService.class);
        }
        return API_SERVICE;
    }
}
```

Figura 60: Clase adaptador.

Método para obtener el token, en esta parte se recibe el usuario y contraseña para hacer la petición a la clase adaptador, que invocará al método requestTokenForm de la interfaz enviando los parámetros del usuario contraseña y tipo, para luego obtener el token de sesión si el usuario y contraseña son correctos, y se guarda en una variable estática para luego usar en las peticiones futuras.

```
private void obtenerToken(String usuario, String contrasenia )
{
    try {
        Call<OAuthToken> call = RedemnorteApiAdapter.getApiService().requestTokenForm(usuario,contrasenia,"password");
        call.enqueue(new Callback<OAuthToken>() {
            @Override
            public void onResponse(Call<OAuthToken> call, Response<OAuthToken> response) {
                if (response.isSuccessful()) {
                    String resultado = response.body().getAccess_token();
                    MetodosAuxiliares.setToken(resultado);
                    if(resultado.length()>=10)
                    tvLogin.setText("Usuario correcto");
                } else {
                    tvLogin.setText("Usuario y contaseña incorrecto");
                }
            }
            @Override
            public void onFailure(Call call, Throwable t) {
                Toast.makeText(consumirWsRest.this, t.getMessage(), Toast.LENGTH_LONG);
            }
        });
    }
    catch (Exception e)
    {
        Toast.makeText(consumirWsRest.this, e.getMessage(), Toast.LENGTH_LONG);
    }
}
```

Figura 61: Método para obtener el token de sesión.

Método para obtener el listado de vehículos enviando el token de sesión.

```
private void obtenerVehiculos(String token)
{
    try {
        Call<List<vehiculos>> call = RedemnorteApiAdapter.getApiService().getVehiculos("Bearer "+token);
        call.enqueue(new Callback<List<vehiculos>>() {
            @Override
            public void onResponse(Call<List<vehiculos>> call, Response<List<vehiculos>> response) {
                tvListaVehiculo.setText(response.message());
                if (response.isSuccessful()) {
                    if (response.body().toString().length() >= 3) {
                        List<vehiculos> vehiculos =  response.body();
                        pasarListaAMatriz(vehiculos);
                    } else {
                        tvListaVehiculo.setText("No se encontraron datos");
                    }
                } else {
                    Toast.makeText(consumirWsRest.this, "error", Toast.LENGTH_LONG);
                }
            }
            @Override
            public void onFailure(Call call, Throwable t) {
                Toast.makeText(consumirWsRest.this, t.getLocalizedMessage(), Toast.LENGTH_LONG);
            }
        });
    }
    catch (IllegalStateException e)
    {
        Toast.makeText(consumirWsRest.this, e.getMessage(), Toast.LENGTH_LONG);
    }
}
```

Figura 62: Método para obtener listado de vehículos.

```
//METDOD PARA INSERTAR VEHICULO
private void InsertVehiculos(String token)
{
    try {
        Call<vehiculos> call = RedemnorteApiAdapter.getApiService().setVehiculo("WWWAAA","SEDAN",
                "PURPURA","BMW",2000,4,2018,"EXECELENTEESTADO","DISPONIBLE",
                "d7aecfe1-b0d6-46a2-bee5-1927bb2e3beb", "Bearer "+token);
        call.enqueue(new Callback<vehiculos>() {
            @Override
            public void onResponse(Call<vehiculos> call, Response<vehiculos> response) {
                tvListaVehiculo.setText(response.message());
                if (response.isSuccessful()) {
                    Toast.makeText(consumirWsRest.this, "Se insertó correctamente", Toast.LENGTH_LONG);
                } else {
                    Toast.makeText(consumirWsRest.this, "error", Toast.LENGTH_LONG);
                }
            }
            @Override
            public void onFailure(Call call, Throwable t) {
                Toast.makeText(consumirWsRest.this, t.getLocalizedMessage(), Toast.LENGTH_LONG);
            }
        });
    }
    catch (IllegalStateException e)
    {
        Toast.makeText(consumirWsRest.this, e.getMessage(), Toast.LENGTH_LONG);
    }
}
```

Figura 63: Método para insertar un vehículo.

```
//METDOD PARA ACTUALIZAR VEHICULO
private void actualizarVehiculos(String token)
{
    try {
        Call<vehiculos> call = RedemnorteApiAdapter.getApiService().actualizarVehiculo(2945,"WWWAAA",
                "SEDAN","PURPURA","BMW",2001,2,2019,"actualizados",
                "DISPONIBLE","d7aecfe1-b0d6-46a2-bee5-1927bb2e3beb", "Bearer "+token);
        call.enqueue(new Callback<vehiculos>() {
            @Override
            public void onResponse(Call<vehiculos> call, Response<vehiculos> response) {
                tvListaVehiculo.setText(response.message());
                if (response.isSuccessful()) {
                    Toast.makeText(consumirWsRest.this, "Se insertó correctamente", Toast.LENGTH_LONG);
                } else {
                    Toast.makeText(consumirWsRest.this, "error", Toast.LENGTH_LONG);
                }
            }
            @Override
            public void onFailure(Call call, Throwable t) {
                Toast.makeText(consumirWsRest.this, t.getLocalizedMessage(), Toast.LENGTH_LONG);
            }
        });
    }
    catch (IllegalStateException e)
    {
        Toast.makeText(consumirWsRest.this, e.getMessage(), Toast.LENGTH_LONG);
    }
}
```

Figura 64: Método actualizar.

```
//METDOD PARA ELIMINAR VEHICULO
private void eliminarVehiculos(String token)
{
    try {
        Call<vehiculos> call = RedemnorteApiAdapter.getApiService().eliminarVehiculo(2945, "Bearer "+token);
        call.enqueue(new Callback<vehiculos>() {
            @Override
            public void onResponse(Call<vehiculos> call, Response<vehiculos> response) {
                tvListaVehiculo.setText(response.message());
                if (response.isSuccessful()) {
                    Toast.makeText(consumirWsRest.this, "Se eliminó correctamente", Toast.LENGTH_LONG);
                } else {
                    Toast.makeText(consumirWsRest.this, "error", Toast.LENGTH_LONG);
                }
            }
            @Override
            public void onFailure(Call call, Throwable t) {
                Toast.makeText(consumirWsRest.this, t.getLocalizedMessage(), Toast.LENGTH_LONG);
            }
        });
    }
    catch (IllegalStateException e)
    {
        Toast.makeText(consumirWsRest.this, e.getMessage(), Toast.LENGTH_LONG);
    }
}
```

Figura 65: Método eliminar.

En la manera que se ve en el dispositivo móvil es la siguiente.

Figura 66: Pantalla inicial.

Una prueba ingresando el usuario y contraseña y haciendo clic en mostrar.

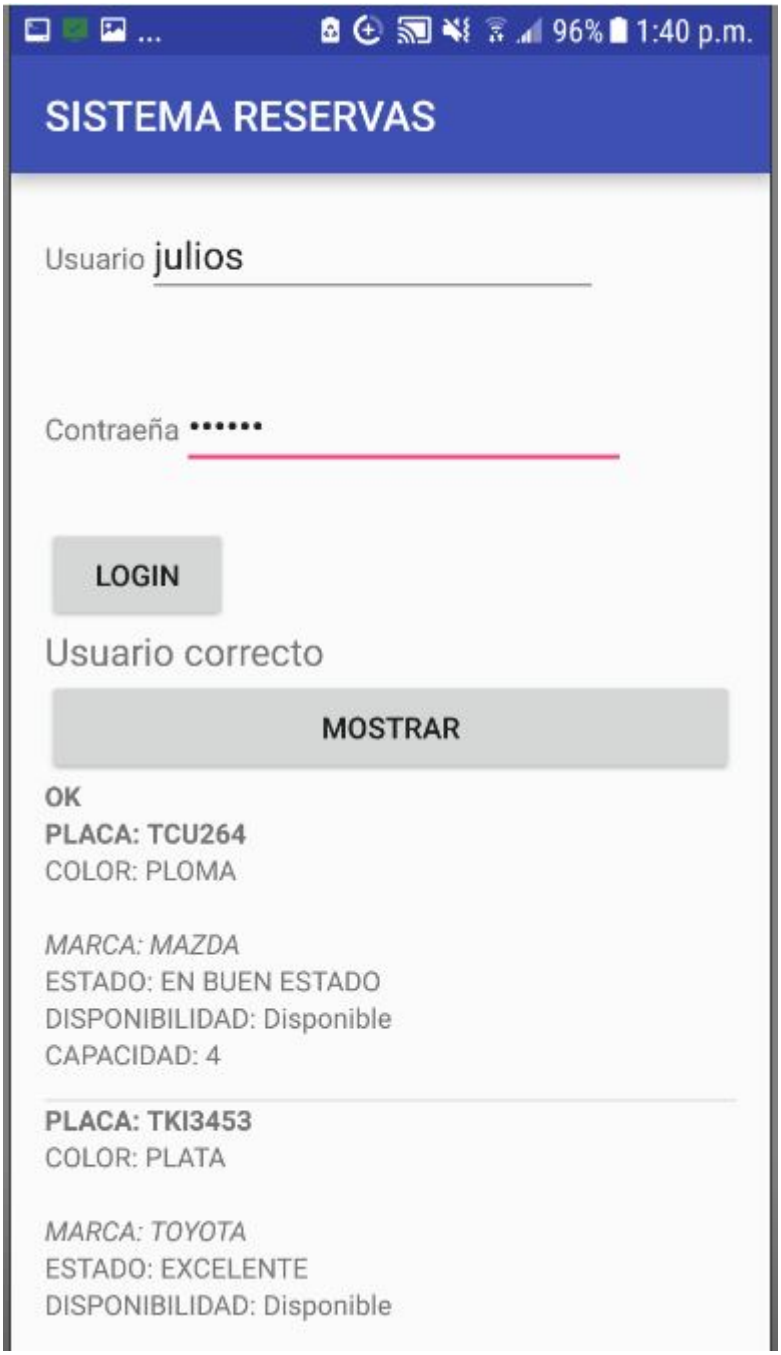


Figura 67: Consumiendo el servicio web.

Ahora una prueba ingresando el usuario y contraseña incorrecto y haciendo clic en mostrar.

Figura 68: Prueba con usuario incorrecto.

4.5. Pruebas de funcionamiento.

4.5.1. Caso de prueba registrar usuario.

Descripción.	Ingresar al sitio web y registrar un usuario para interactuar con el sistema.
Flujo de eventos.	Ingresar al sistema, hacer clic en reservar, se redirecciona al login, y en la parte de abajo hacemos clic en registrar nuevo usuario.
Precondiciones.	Acceder al sitio web.
Poscondiciones.	- Se debe permitir ingresar un email correcto. - El usuario debe ser único. - Si las contraseñas no son iguales mostrar un mensaje. - Mostrar un mensaje si se crea el usuario. - Asociar a un rol por defecto para el ingreso al sistema.
Entrada.	Email: tonyplaza@gmail.com
	Usuario: tonyplaza123
	Contraseña: tn243Qj. %
	Repetir Contraseña: tn243Qj. %
Resultado esperado.	- El usuario se creó correctamente.
Evaluación de la prueba.	Prueba superada exitosamente.

Tabla 38: Prueba registrar un usuario.

4.5.2. Caso de prueba Ingreso al sistema.

Descripción.	Ingresar al sitio web con un usuario para interactuar con el sistema.
Flujo de eventos.	Ingresar al sistema, hacer clic en reservar, luego en el login, ingresar usuario y contraseña.
Precondiciones.	Acceder al sitio web y hacer clic en reservar.
Poscondiciones.	- Mostrar un mensaje de error en el caso de no ingresar correctamente el usuario y contraseña. - Después de ingresar al sistema mostrar el menú de acuerdo al rol al que pertenece.
Entrada.	Usuario: jaimejlr
	Contraseña: jaimelopezJ.1 %
Resultado esperado.	- Ingresar al sistema y mostrar el menú según el rol.
Evaluación de la prueba.	Prueba superada exitosamente.

Tabla 39: Prueba ingreso al sistema.

4.5.3. Caso de prueba Roles.

Descripción.	Ingresaremos al sistema como administrador, y gestionaremos el módulo de roles ingresando, actualizando y eliminando un rol.
Flujo de eventos.	En la pantalla principal dar clic en el menú roles y en insertar.
Precondiciones.	Estar loggeado con usuario administrador.
Entrada.	Nombre Rol: Chofer.
Resultado esperado.	- Se insertó correctamente. - Se actualizó correctamente. - Se eliminó correctamente. - Se actualizó en la tabla que se muestran todos los roles.
Evaluación de la prueba.	Prueba superada exitosamente.

Tabla 40: Prueba gestión de roles.

4.5.4. Caso de prueba Asignar Rol.

Descripción.	Ingresaremos al sistema como administrador, y gestionaremos el módulo de asignar roles
Flujo de eventos.	En la pantalla principal dar clic en el menú asignar rol.
Precondiciones.	Estar loggeado con usuario administrador. - Buscar un usuario. Seleccionar el nuevo rol que se asignará. - Eliminar rol.
Entrada.	Usuario: tonyplaza Rol: chofer.
Resultado esperado.	- Se insertó correctamente. - Se eliminó correctamente. - Se actualizó en la tabla que se muestran todos los roles.
Evaluación de la prueba.	Prueba superada exitosamente.

Tabla 41: Prueba asignación de roles.

4.5.5. Caso de prueba Vehículos.

Descripción.	Ingresaremos al sistema como administrador y gestionaremos el módulo vehículos ingresando, actualizando y eliminando un vehículo.
Flujo de eventos.	En la pantalla principal dar clic en el menú vehículos.
Precondiciones.	Estar loggeado con usuario administrador.
Poscondiciones.	- Se mostrará un mensaje con la acción realizada. - Si ocurre un error se mostrará un mensaje.
Entrada.	- Placa:TCU264 - Modelo: SEDAN - Color: ROJO - Cilindraje: 1500 - Capacidad: 4 - Año de fabricación: 2015 - Descripción: DIBUJOS EN LAS PUERTAS -Propietario: tonyplaza
Resultado esperado.	- Se insertó correctamente. - Se actualizó correctamente. - Se eliminó correctamente. - Al presionar en el botón buscar se muestran los vehículos
Evaluación de la prueba.	Prueba superada exitosamente.

Tabla 42: Prueba gestión de vehículos

4.5.6. Caso de prueba Ciudad.

Descripción.	Ingresaremos al sistema como administrador y gestionaremos el módulo ciudades ingresando, actualizando y eliminando una ciudad.
Flujo de eventos.	En la pantalla principal dar clic en el menú ciudades.
Precondiciones..	Estar loggeado con usuario administrador.
Poscondiciones.	- Se mostrará un mensaje con la acción realizada. - Se actualizará en la tabla que se muestran las ciudades. - Si ocurre un error se mostrará un mensaje.
Entrada.	- Ciudad: AMBATO. - Descripción: CENTRO DEL PAÍS.
Resultado esperado.	- Se insertó correctamente. - Se actualizó correctamente. - Se eliminó correctamente. - Se actualizó en la tabla que se muestran las ciudades.
Evaluación de la prueba.	Prueba superada exitosamente.

Tabla 43: Prueba gestión de ciudades

4.5.7. Caso de prueba Horarios.

Descripción.	Ingresaremos al sistema como administrador y gestionaremos el módulo horario ingresando, actualizando y eliminando un horario.
Flujo de eventos.	En la pantalla principal dar clic en el menú horarios.
Precondiciones.	Estar loggeado con usuario administrador.
Poscondiciones.	- Se mostrará un mensaje con la acción realizada. - Se actualizará en la tabla que se muestran los horarios. - Si ocurre un error se mostrará un mensaje.
Entrada.	Ciudad origen: AMBATO Ciudad destino: QUITO Día: LUNES Hora: 07:00 Precio: 12
Resultado esperado.	- Se insertó correctamente. - Se actualizó correctamente. - Se eliminó correctamente. - Se actualizó en la tabla que se muestran los horarios.
Evaluación de la prueba.	Prueba superada exitosamente.

Tabla 44: Prueba gestión de horarios

4.5.8. Caso de prueba Clientes.

Descripción.	Ingresaremos al sistema como administrador, asistente o cliente y gestionaremos el módulo clientes ingresando, actualizando y eliminando un Cliente.
Flujo de eventos.	En la pantalla principal dar clic en el menú clientes.
Precondiciones.	Estar loggeado con usuario administrador o asistente.
Poscondiciones.	- Se mostrará un mensaje con la acción realizada. - Ingresar cédula valida. - Se actualizará en la tabla que se muestran los clientes. - Si ocurre un error se mostrará un mensaje.
Entrada.	Cédula: 1804677951 Nombre: CARLOS Apellido: ARAUJO Provincia: TUNGURAHUA Cantón: AMBATO Calle principal: AV CEVALLOS Calle secundaria: QUITO Número de casa: 10-49 Referencia: COLEGIO UELAM. Teléfono1: 0984785826 Teléfono2: 0978568423
Resultado esperado.	- Se insertó correctamente. - Se actualizó correctamente. - Se eliminó correctamente. - Se actualizó en la tabla que se muestran todos los clientes.
Evaluación de la prueba.	Prueba superada exitosamente.

Tabla 45: Prueba gestión de clientes.

4.5.9. Caso de prueba Reserva.

Posibilidad de flujo 1, Esta prueba se realiza con un rol de administrador o asistente, se utiliza los datos del cliente para el origen de viaje.

Descripción.	Ingresaremos al sistema como administrador, asistente o cliente y gestionaremos el módulo reservas ingresando, actualizando y eliminando una reserva.
Flujo de eventos.	En la pantalla principal dar clic en el menú reserva.
Precondiciones.	Estar loggeado con usuario administrador o asistente.
Poscondiciones.	- Se mostrará un mensaje con la acción realizada. - Se actualizará en la tabla que se muestran las reservas. - Si ocurre un error se mostrará un mensaje.
Entrada.	Cliente: CARLOS ARAUJO Datos para el viajar al origen: FALSO Horario: AMBATO-QUITO 07:00 Número de asientos: 1 Número de equipaje: 0 Datos de origen del viaje: Calle principal: AV CEVALLOS Calle secundaria: QUITO Número de casa: 10-49 Referencia: COLEGIO UELAM. Teléfono1: 0984785826 Teléfono2: 0978568423 Observación: SER PUNTUAL Latitud: -1.24398 Longitud: -78.62937 Datos del Destino del viaje: Calle principal: AV DE LOS GRANADOS Calle secundaria: AV. ELOY ALFARO Fecha: 02/04/2017 Referencia: Referencia del destino. QUITO NORTE Latitud: -0.16613 longitud: -78.46845 Estado: PENDIENTE
Resultado esperado.	- Se insertó correctamente. - Se actualizó correctamente. - Se eliminó correctamente. - Se actualizó en la tabla que se muestran todas las reservas.
Evaluación de la prueba.	Prueba superada exitosamente.

Tabla 46: Prueba gestión de Reservas posibilidad 1.

Posibilidad de flujo 2, Esta prueba se realiza con un rol de administrador o asistente, se utiliza los datos del cliente para el destino de viaje.

Descripción.	Ingresaremos al sistema como administrador, asistente o cliente y gestionaremos el módulo reservas ingresando, actualizando y eliminando una reserva.
Flujo de eventos.	En la pantalla principal dar clic en el menú reserva.
Precondiciones.	Estar loggeado con usuario administrador o asistente.
Poscondiciones.	- Se mostrará un mensaje con la acción realizada. - Se actualizará en la tabla que se muestran las reservas. - Si ocurre un error se mostrará un mensaje.
Entrada.	Cliente: CARLOS ARAUJO Datos para el viajar al origen: VERDADERO Horario: QUITO - AMBATO 18:00 Número de asientos:1 Número de equipaje: 1 Datos de origen del viaje: Calle principal: AV DE LOS GRANADOS Calle secundaria: AV ELOY ALFARO Número de casa: SN Referencia: QUITO NORTE Teléfono1: 0984785826 Teléfono2: 0978568423 Observación: A LAS 6PM Latitud: -0.16613 longitud: -78.46845 Datos del Destino del viaje: Calle principal : AV CEVALLOS Calle secundaria : CALLE QUITO Fecha: 02/04/2018 Referencia:COLEGIO UELAM Latitud: -1.24398 Longitud: -78.62937 Estado: CONFIRMADO
Resultados esperado.	- Se insertó correctamente. - Se actualizó correctamente. - Se eliminó correctamente. - Se actualizó en la tabla que se muestran las reservas.
Evaluación de la prueba.	Prueba superada exitosamente.

Tabla 47: Prueba gestión de Reservas posibilidad 2.

4.5.10. Caso de prueba Asignar Reserva.

Descripción.	Ingresaremos al sistema como administrador, o como asistente y gestionaremos el módulo de asignar reservas y asignar clientes a un vehículo.
Flujo de eventos.	En la pantalla principal dar clic en el menú asignar reserva.
Precondiciones.	Estar loggeado con usuario administrador o asistente. - Al dar clic en el botón Asignar reserva a un vehículo, y si no se ha buscado un vehículo no debe grabar ningún cambio. - Al buscar una reserva primero se debe buscar un vehículo. - Si una reserva ya fue asignada a otro vehículo no debe permitir asignar a otro. - La sumatoria de asientos en la reserva no debe superar al limite de asientos disponible del vehículo.
Entrada.	Vehículo: TCU264 Reserva: CARLOS ARAUJO
Resultado esperado.	- Se insertó correctamente. - Se actualizó correctamente. - Se eliminó correctamente. - Se actualizó en la tabla que se muestran todas las reservas.
Evaluación de la prueba.	Prueba superada exitosamente.

Tabla 48: Prueba gestión de asignación de reservas.

4.5.11. Caso de prueba Reporte de viajes.

Descripción.	Ingresaremos al sistema como administrador o asistente y gestionaremos el módulo reportes de reservas por vehículo generando un reporte.
Flujo de eventos.	En la pantalla principal dar clic en reportes y reporte de reservas por vehículo.
Precondiciones.	- Estar loggeado con usuario administrador o asistente. - Tener asignado por lo menos una reserva.
Poscondiciones.	- Se debe seleccionar un vehículo. - Seleccionar un intervalo de fechas.
Entrada.	- Vehículo: TCU264 - Fecha inicio: 1/01/2018 - Fecha fin: 1/02/2018
Resultados esperado.	- Se muestra el reporte perteneciente al vehículo.
Evaluación de la prueba.	Prueba superada exitosamente.

Tabla 49: Prueba reporte de viajes.

4.5.12. Caso de prueba Reporte de clientes frecuentes.

Descripción.	Ingresaremos al sistema como administrador o asistente y gestionaremos el módulo reportes de clientes frecuentes generando un reporte.
Flujo de eventos.	En la pantalla principal dar clic en reportes y reporte de clientes mas frecuentes.
Precondiciones.	- Estar loggeado con usuario administrador o asistente. - Tener un cliente que haya realizado una reserva.
Poscondiciones.	- Se debe seleccionar ciudad de origen y destino. - Seleccionar un intervalo de fechas.
Entrada.	- Ciudad origen: Ambato. - Ciudad destino: Quito. - Fecha inicio: 1/01/2018 - Fecha fin: 1/02/2018
Resultado esperado.	- Se muestra el reporte de los clientes frecuentes en esa fecha.
Evaluación de la prueba.	Prueba superada exitosamente.

Tabla 50: Prueba reporte de viajes.

4.5.13. Caso de prueba Vehículos.

Descripción.	Ingresaremos al sistema como chofer y gestionaremos el módulo vehículos revisando y actualizando el estado del o los vehículos.
Flujo de eventos.	En la pantalla principal dar clic en el menú vehículos.
Precondiciones.	- Estar loggeado con usuario chofer. - Tener asignado por lo menos un vehículo.
Poscondiciones.	- Se listará todos los vehículos a su nombre. - Para actualizar el estado se debe seleccionar. - Se mostrará un mensaje cuando se haga una actualización.
Entrada.	- Vehículo: TCU264 -Estado: DISPONIBLE
Resultado esperado.	- Se actualizó correctamente.
Evaluación de la prueba.	Prueba superada exitosamente.

Tabla 51: Prueba gestión de vehículos con rol de chofer.

4.5.14. Caso de prueba asignación de clientes.

Descripción.	Ingresaremos al sistema como chofer y gestionaremos el módulo gestión de clientes actualizando el estado del o los vehículos.
Flujo de eventos.	En la pantalla principal dar clic en asignación de clientes.
Precondiciones.	- Estar loggeado con usuario chofer. - Tener asignado por lo menos un cliente.
Poscondiciones.	- Se listará todos los clientes asignados al vehículo. - Para actualizar el estado se debe seleccionar un cliente. - Se mostrará un mensaje cuando se haga una actualización.
Entrada.	- Cliente: Seleccionar un cliente. -Estado: Seleccionar un estado
Resultado esperado.	- Se actualizó correctamente.
Evaluación de la prueba.	Prueba superada exitosamente.

Tabla 52: Prueba gestión de vehículos con rol de chofer.

4.6. Pruebas de concurrencia de usuarios.

El equipo en el que está instalado el servicio web y la base de datos tiene las siguientes características:
Sistema operativo: Windows 8.1 x 64
Memoria Ram: 8GB
Procesador: AMD 2.0 GHz
Para hacer las pruebas de concurrencia, se utilizó el software JMETER, que es un simulador de un cliente haciendo varias transacciones hacia el servidor, el primer paso es crear las configuraciones de usuarios, ingresando el número de usuario, el periodo de tiempo en segundos y las veces a repetir el proceso, después se debe agregar las peticiones http para cada servicio web.
Para este ejemplo se configuró para 1000 usuarios haciendo 5 transacciones de 4 de seleccionar y 1 de insertar en un periodo de 60 segundos.

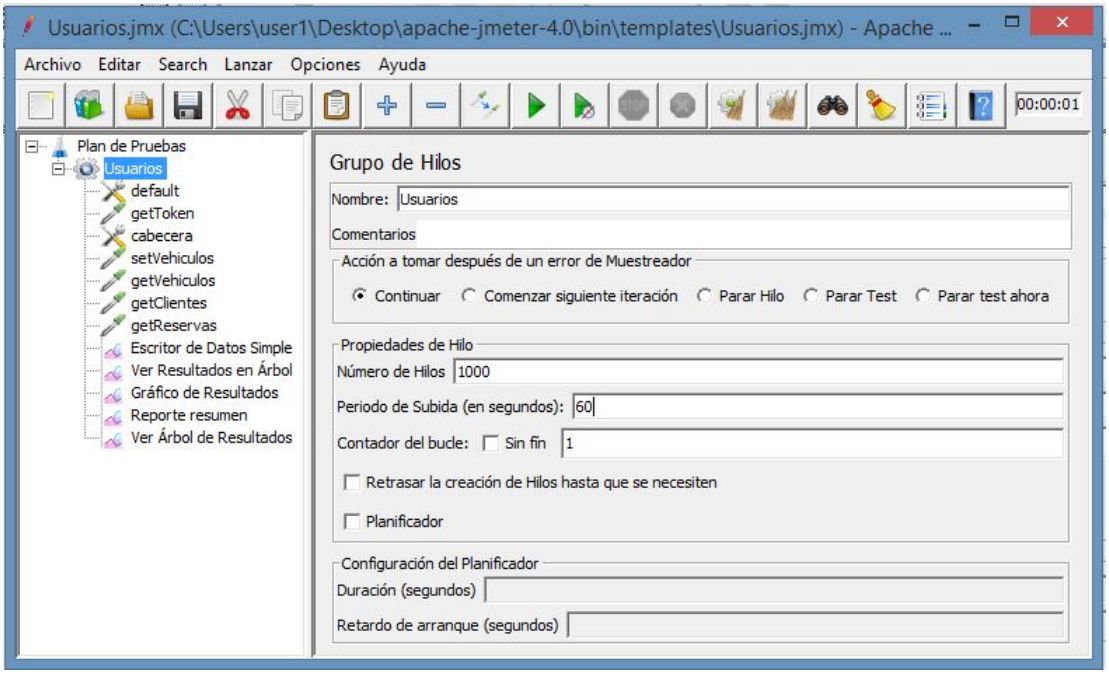


Figura 69: Configuración usuarios.

Se debe agregar la configuración de petición http por defecto, en el cual se ingresa el servidor y el puerto.

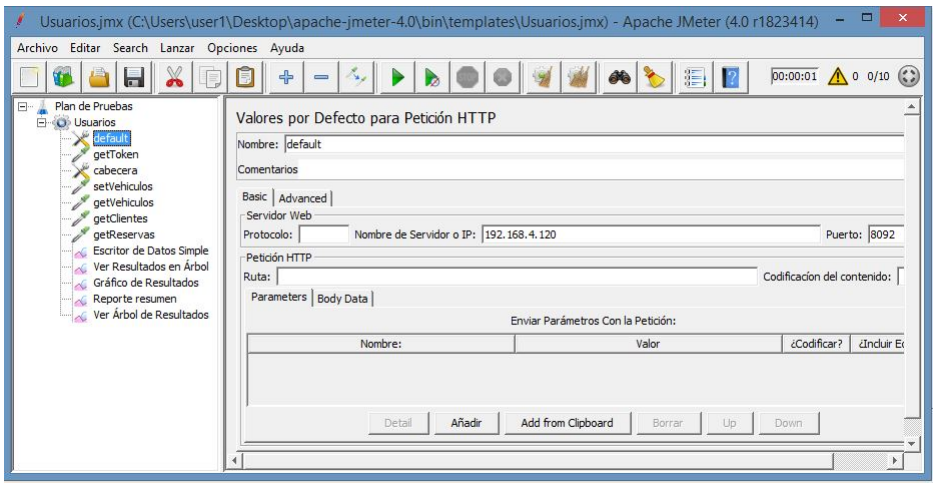


Figura 70: Configuración por defecto.

La petición para obtener el token de sesión, se hace con el método POST, y se hereda las configuraciones del servidor de la petición anterior.

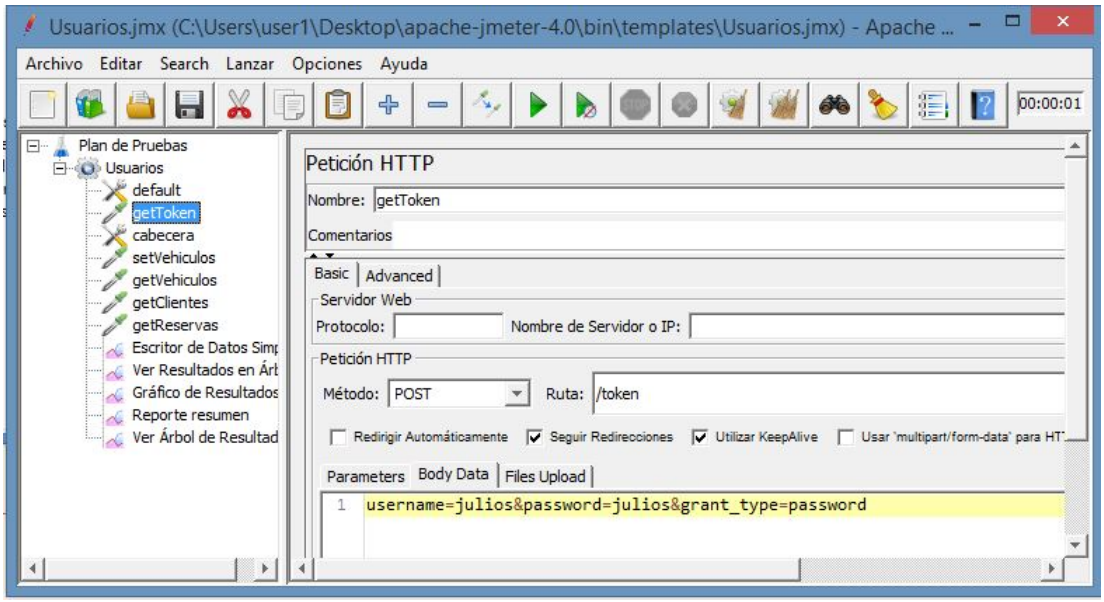


Figura 71: Obtener el token de sesión.

Configuración para agregar vehículos.

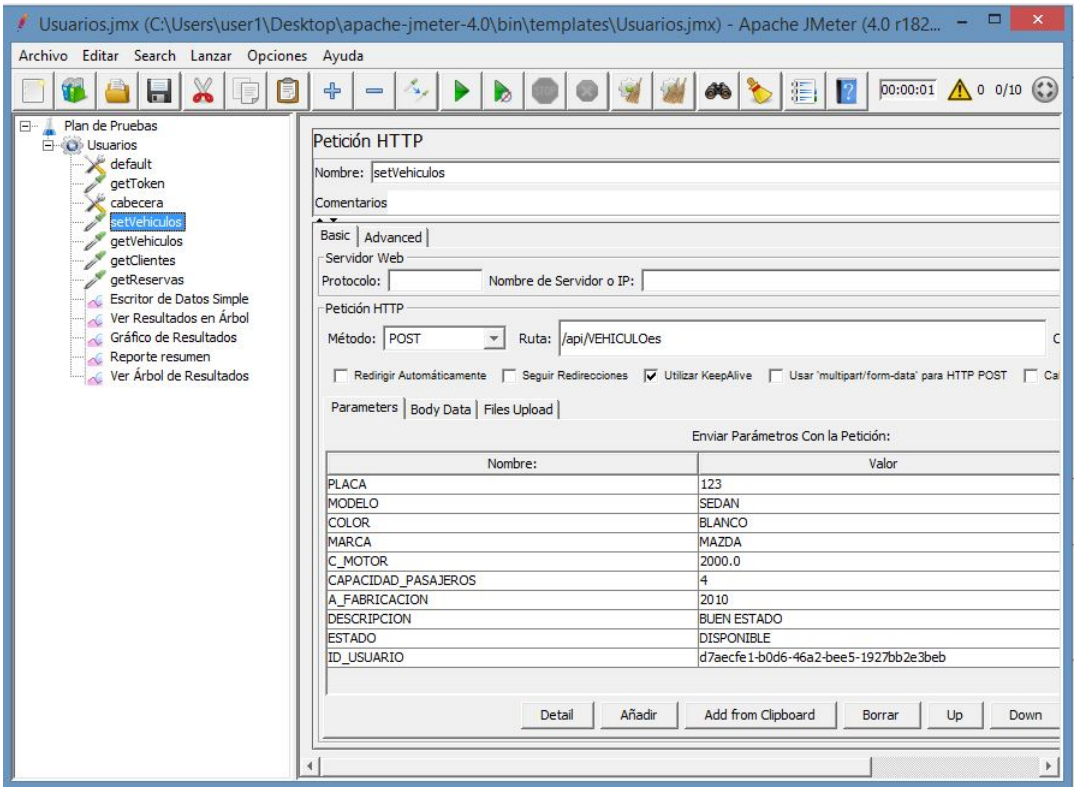


Figura 72: Insertar vehículo.

Petición para seleccionar vehículos.

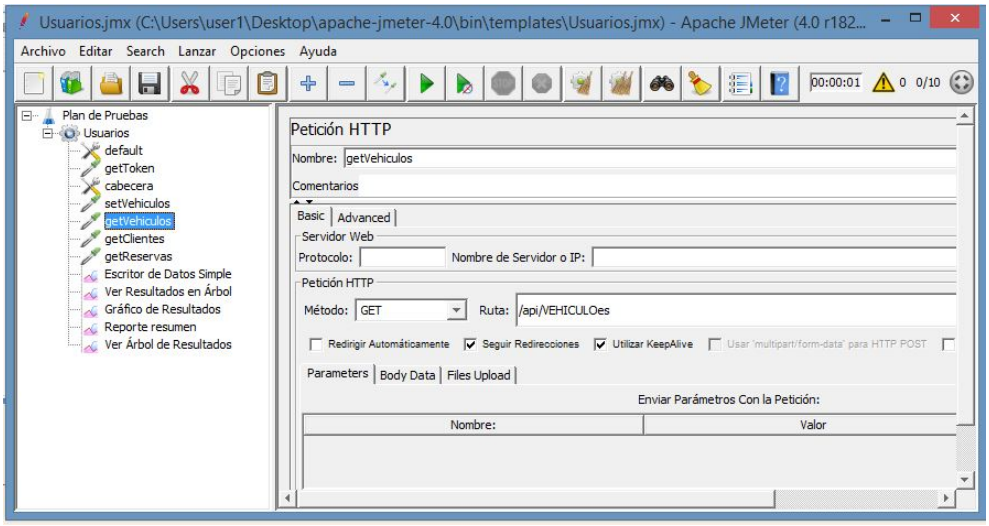


Figura 73: Seleccionar vehículos.

Resultados obtenidos.

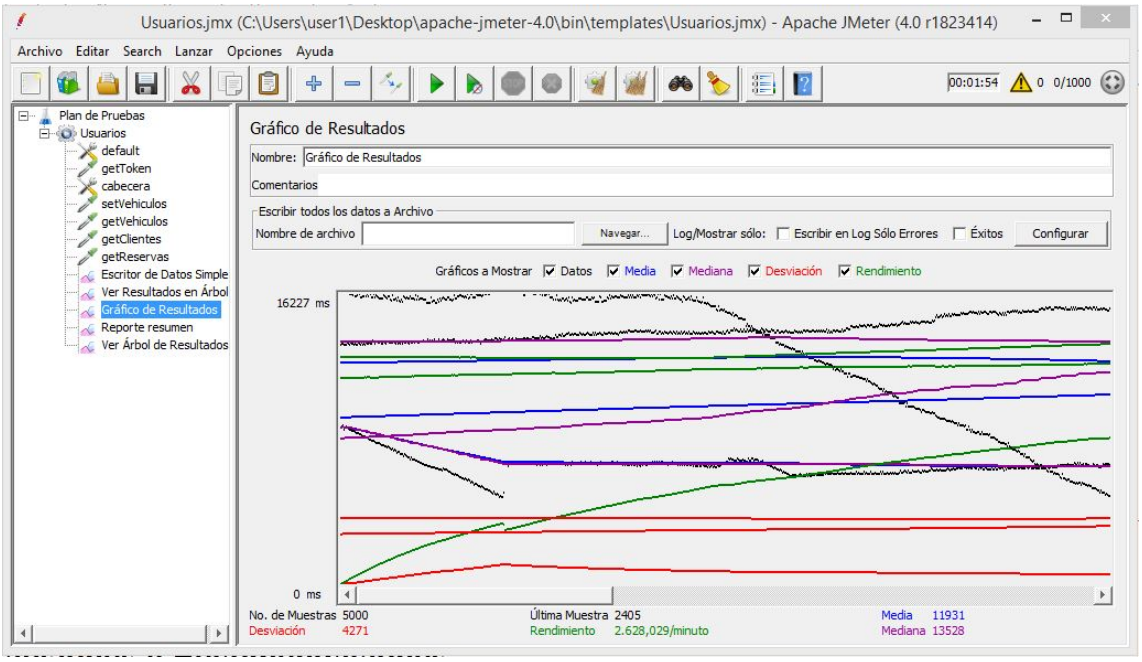


Figura 74: Resultados obtenidos

Resumen de los resultados.

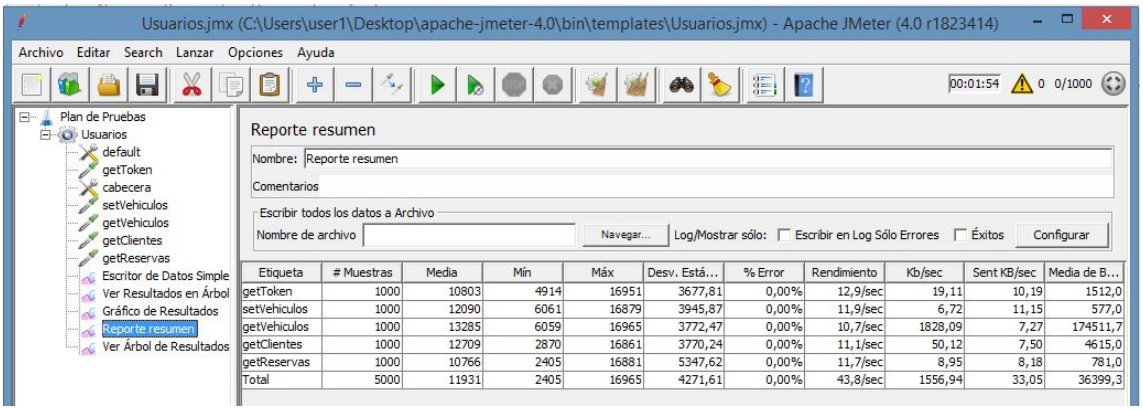


Figura 75: Resumen de los resultados.

Interfaz responsive en dispositivos móviles.

Figura 76: Página de Inicio.

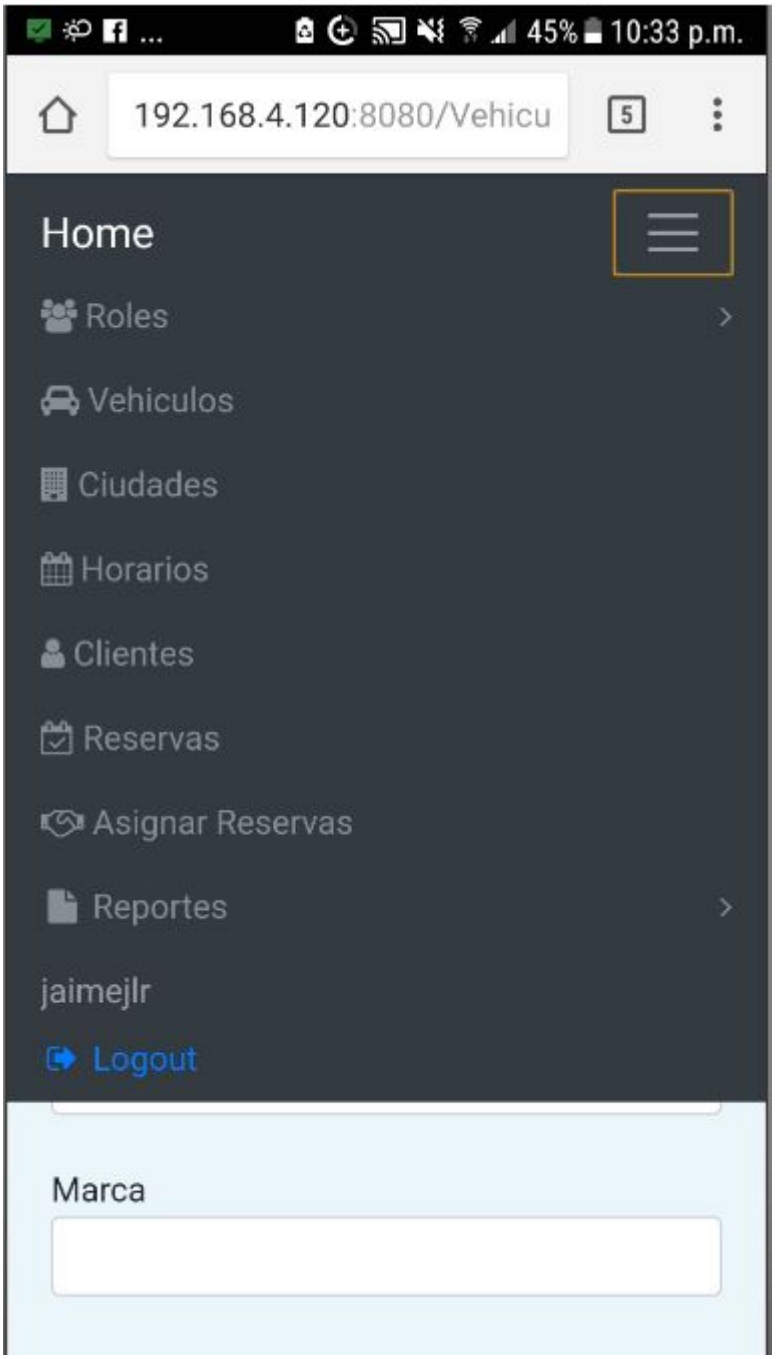


Figura 77: Interfaz menú.

Figura 78: Formulario vehículos.

Figura 79: Formulario ciudad parte superior.

Figura 80: Formulario ciudad parte inferior.

CAPÍTULO 5

Conclusiones y Recomendaciones.

5.1. Conclusiones.

- Para la parte de los procesos de reserva de los servicios de transporte puerta, se utilizó técnicas como el modelado del negocio en caso de uso, entrevista y observación resultando eficientes para tener claro el procedimiento.
- El optar por el servicio web tipo rest, permitió una comunicación rápida con el servidor debido a que su información está representado en formato JSON.
- La elección de Asp.Net C# y Entity Framework Data Model, con sus múltiples herramientas facilitó la creación de los servicios web tipo rest y el desarrollo de la aplicación web.
- El uso de Owin Midleware en los servicios web, permitió que se pueda controlar el nivel de acceso a los recursos de los mismos, el sistema generó un token único por cada usuario que fue utilizado para validar la autentificación cada vez que se consumió la información de un servicio web.
- RUP junto con UML hacen una metodología de desarrollo, además por su flexibilidad se pudo simplificar las fases, ayudando al desarrollo de la aplicación web de una forma ágil.
- Los servicios web son compatibles para consumir en dispositivos con sistema operativo android.
- Después de hacer las pruebas de concurrencia con 1000 usuarios haciendo 5 transacciones cada usuario en periodo de 60 segundos en un servidor de prueba con las características descritas soportó hasta 43.8 transacciones por segundo o 2628 transacciones por minuto, lo que significa que un servidor de producción que tenga mínimo 2 procesadores, 64 GB de memoria RAM, soportará 5000 transacciones en 60 segundos.

5.2. Recomendaciones.

- Al utilizar Entity Framework Data Model, es recomendable para la creación de los servicios web no usar campos compuestas como claves primarias, porque crea únicamente el servicio web con una sola clave primaria, y esto impide actualizar y eliminar por id.

- Para hacer una consulta compleja es recomendable crear un procedimiento almacenado y un controlador para llamar desde la aplicación y si tiene parámetros estos pasar por url.

- Para crear una metodología de desarrollo se debe combinar RUP con UML, y para que sea ágil se debe adaptar a RUP simplificando sus fases según sea las necesidades de acuerdo al contexto y necesidades del proyecto.

- Para el control de acceso para el consumo de un servicio web tipo rest siempre se debe hacer privado poniendo la definición en la cabecera del controlador, para que solo las peticiones que tengan un token generado por un usuario al iniciar sesión lo puedan consumir.

- Para que la información viaje segura a través de internet, se debe instalar un certificados SSL para que se pueda encriptar la información entre el cliente y el servidor.

Bibliografia

[1] J. Salan, "Aplicación web para la gestión de rutas y pre-reservas de la cooperativa de transportes el dorado.," Master's thesis, UNIANDES, 2015.

[2] J. C. y Marwin Cedamano y otros, "Análisis, dise ñ o y desarrollo del sistema para el control de rutas, encomiendas, reservaciones y venta de tickets para la cooperativa de transportes pullman carchi.," Master's thesis, ESPE, 2010.

[3] Á. Cobo, *PHP y MySQL: Tecnología para el desarrollo de aplicaciones web.* Ediciones Díaz de Santos, 2005.

[4] A. C. y Elsa Urrutia, "Sistema web para el registro de reservaciones y control de hospedaje en el hotel acapulco de la ciudad de ambato," Master's thesis, UNIVERSIDAD TÉ'CNICA DE AMBATO, 2015.

[5] E. Fernández and I. Olmeda, "Servidores de información y reservas a través de internet," *TURITEC*, vol. 99, pp. 195–208, 1999.

[6] F. Sierra, J. Acosta, J. Ariza, and M. Salas, "Estudio y análisis de los framework en php basados en el modelo vista controlador para el desarrollo de software orientado a la web," *Revista Investigación y Desarrollo en TIC*, vol. 4, no. 2, 2017.

[7] P. B. Torre, "Gestión de los datos para aplicaciones generales usando tecnoloías .net y metodologías de tipo desifn pattern.," Master's thesis, UNIVERSIDAD DE CANTABRIA, 2015.

[8] J. C. y Elsa Urrutia, "Plataforma informática orientada a servicios web para la recopilación de información de las áreas de terapia física en las unidades educativas especializadas.," Master's thesis, UNIVERSIDAD TÉCNICA DE AMBATO, 2016.

[9] V. Carrasco, "Servicios web en dispositivos móviles para el soporte de aplicaciones colaborativas.," Master's thesis, Universidad de Chile, 2007.

[10] N. Chase, "Comprender las especificaciones de los servicios web, parte 1: Soap."

https://www.ibm.com/developerworks/ssa/webservices/tutorials/ws-understand-web-services1/index.html, Aug. 2011.

[11] R. Navarro, "Modelado, diseño e implementación de servicios web," tech. rep., UPV, 2006.

[12] D. C. de la Computación e IA, "Introducción a los servicios web. invocación de servicios web soap." http://www.jtech.ua.es/j2ee/publico/servc-web-2012-13/sesion01-apuntes.pdf.

[13] M. Wasson, "Authentication and authorization in asp.net web api." 2012.

[14] S. Delgado and B. Alexander, *Análisis de factibilidad de procedimiento de autenticación única para acceder a los servicios informáticos de La Pontificia Universidad Católica del Ecuador-Sede Esmeraldas.* PhD thesis, Ecuador-PUCESE-Escuela de Sistemas y Computación, 2018.

[15] M. Wasson, "Getting started with owin and katana," 2013.

[16] A. P. ñ. a. Diana López, "Desarrollo de software utilizando la metodología rup caso práctico: Sistema escólastico parametrizable.," Master's thesis, ESCUELA POLITECNICA DEL EJERCITO, 2004.

[17] P. Torrecilla, "El proceso unificado ágil: fases y disciplinas." http://nosolopau.com/2012/06/07/mas-sobre-el-proceso-unificado-agil-fases-y-disciplinas/.

[18] P. L. y Carmen Penads, "Metodologías ágiles para el desarrollo de software: extreme programming (xp)," Master's thesis, Universidad Politécnica de Valencia, 2012.

[19] J. Araníbar, "Metodología 3p," *Ingeniería del Software*, 2015.

[20] E. Delgado, "Metodologías de desarrollo de software. Â¿cuál es el camino?," 2008.

[21] L. A. M. Pilamunga, "Aplicación móvil de gestión administrativa y académica en la escuela de formación y capacitación de conductores profesionales de Ambato utilizando la plataforma android.," Master's thesis, Universidad Técnica de Ambato., 2017.

[22] E. R. CAÑARTE MONTALVÁN, C. TUMBACO, and A. DORALISA, "Aplicación móvil para la ubicación y reserva de canchas múltiples en

la ciudad de manta.," Master's thesis, UNIVERSIDAD LAICA ELOY ALFARO DE MANABÍ, 2018.

Anexos y Apéndices

MANUAL DE USUARIO

Sistema de reservas web para el servicio de transporte puerta a puerta.

2018

ÍNDICE

Ingreso al Sistema. **1**

Registrar un nuevo usuario. **2**

Crear usuario. 2

Login **3**

Roles. **5**

Insertar Roles. 5

Asignar Roles 6

Vehículos. **7**

Ciudad **10**

Horarios. **12**

Clientes **13**

Reservas. **15**

Asignar reservas. **19**

Reportes. **21**

Reporte Reserva por vehículo 21

Reporte clientes más frecuentes. 22

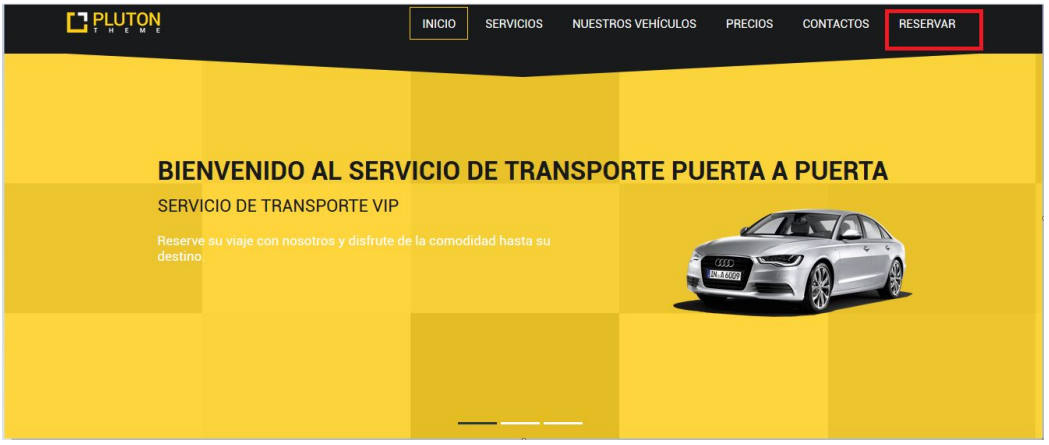
Menú de asistente **23**

Menú de cliente **23**

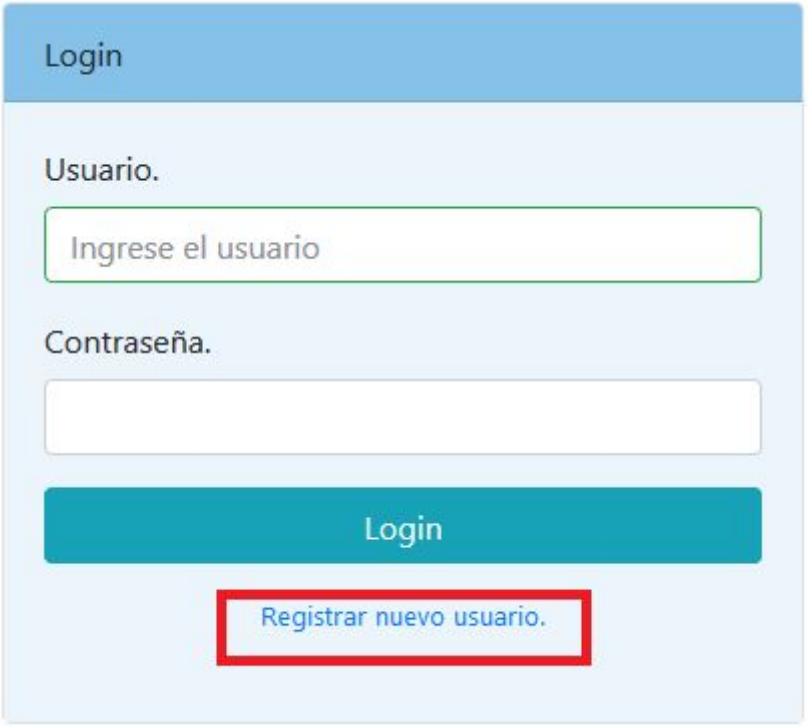
Menú chofer **25**

1. Ingreso al Sistema.

Ingresar al sistema, se mostrará toda la información correspondiente, y hacer clic en reservar.

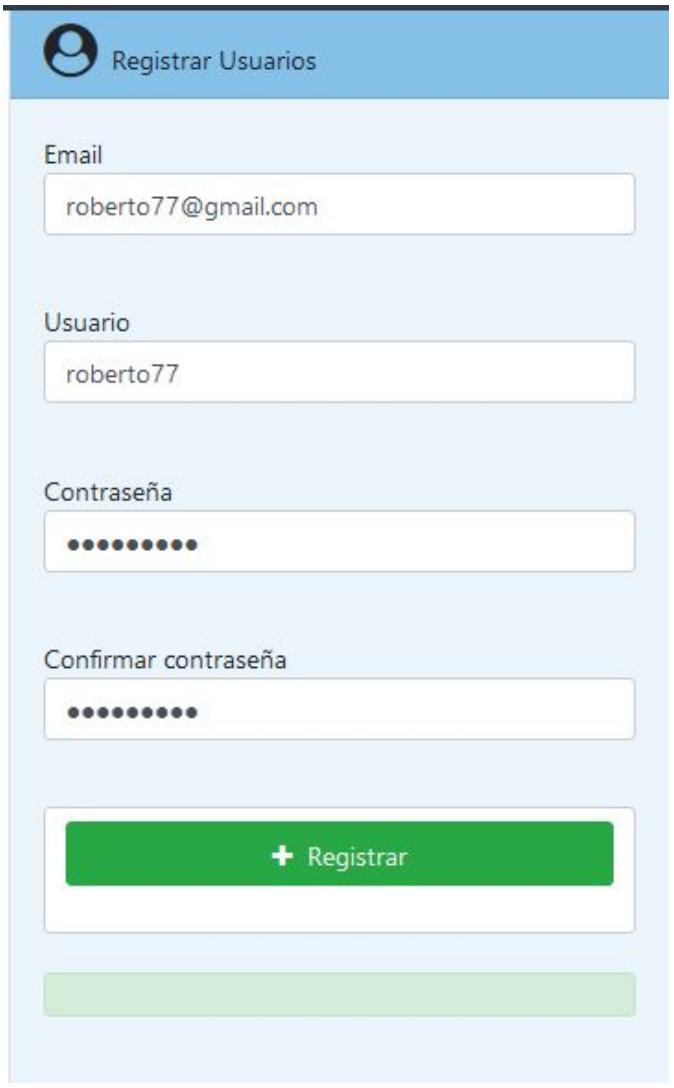


2. Registrar un nuevo usuario.



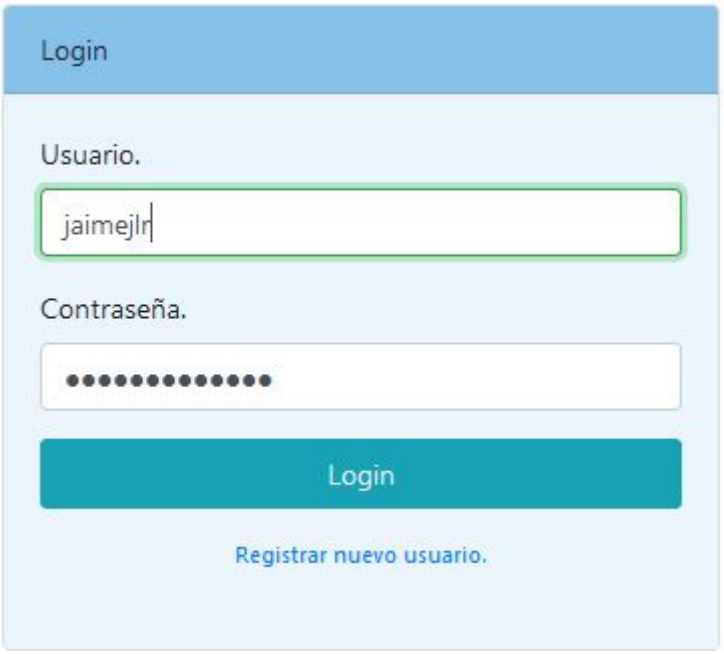
2.1. Crear usuario.

Ingresar los datos correspondientes y hacer clic en Registrar, cuando se crea un usuario por defecto se crea como cliente.

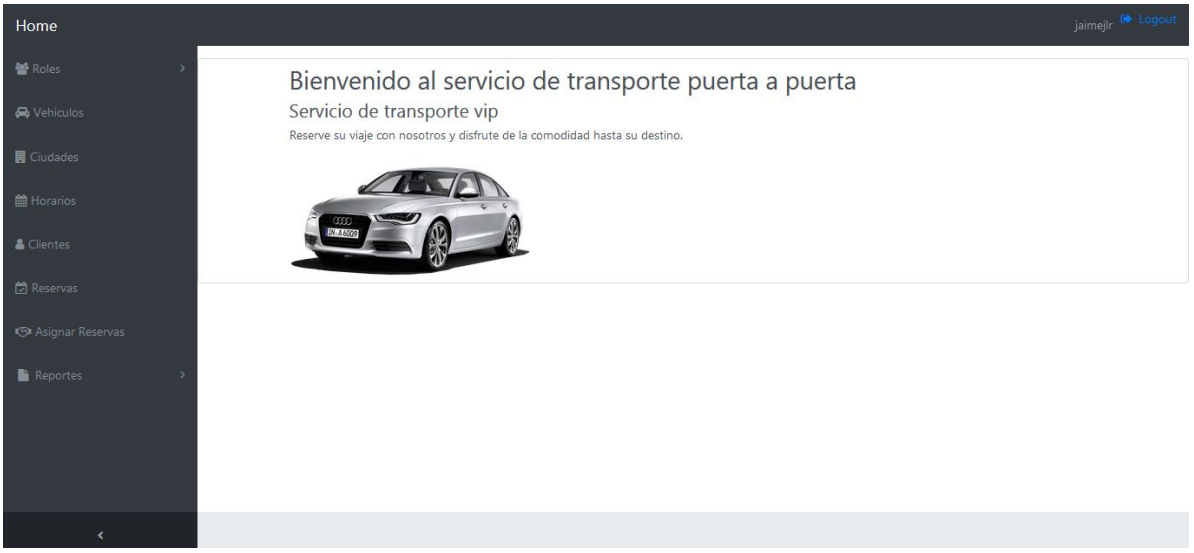


3. **Login.**

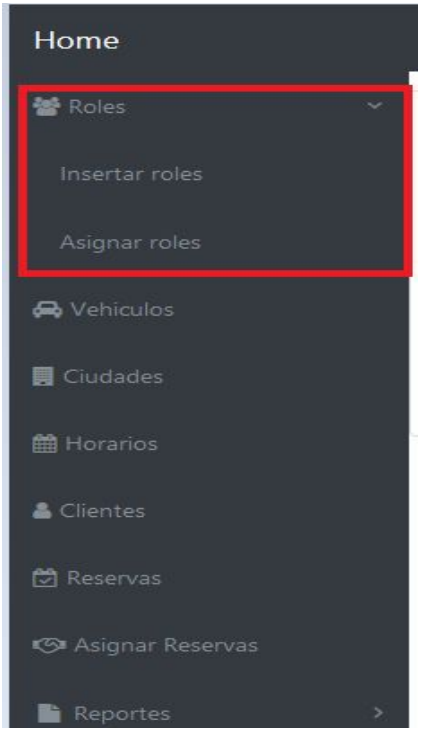
   Para el login hacemos clic en reservar, y nos mostrará el formulario login, en donde ingresamos el usuario y contraseña correspondiente, en este caso ingresar como usuario administrador.



Si las credenciales son correctas, se mostrará la siguiente pantalla.

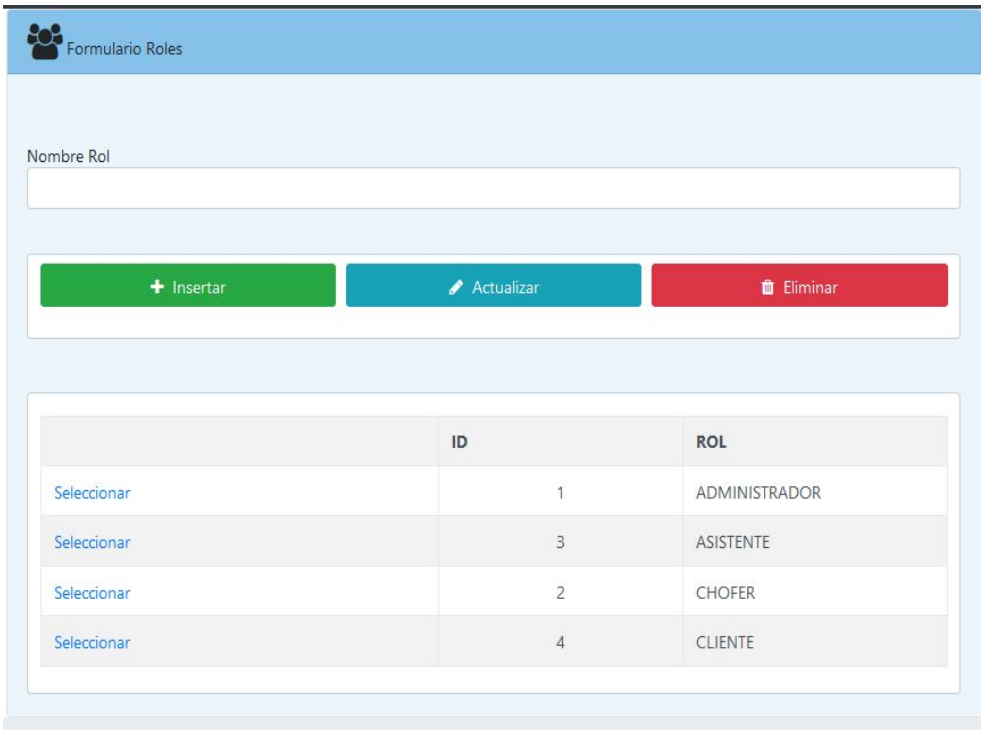


4. Roles.



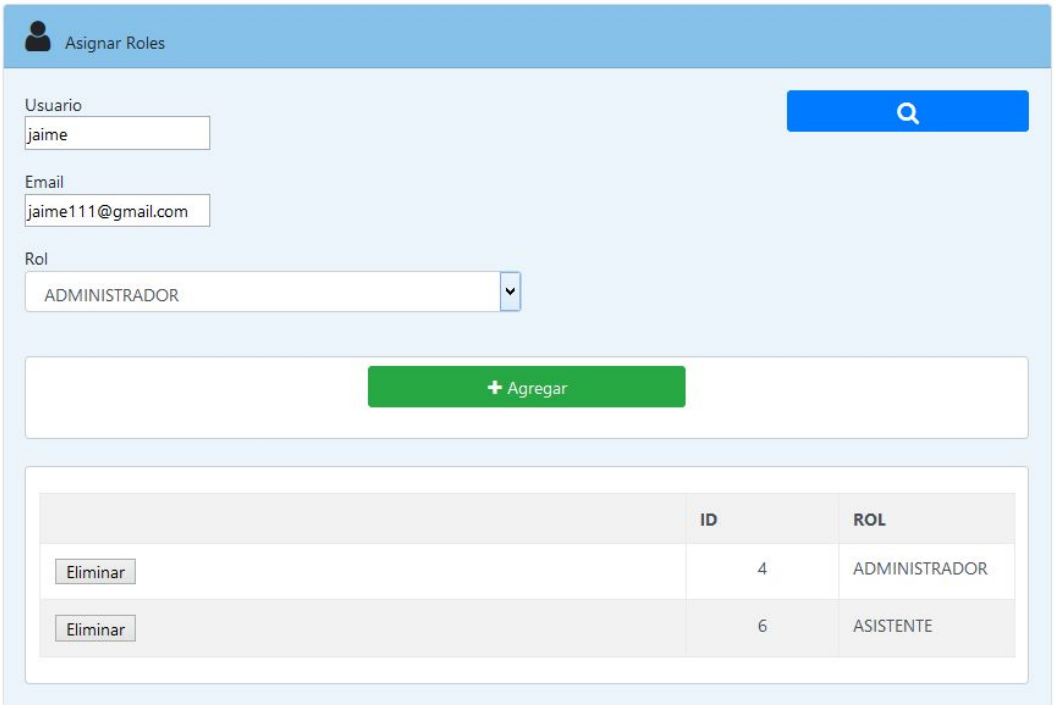
4.1. Insertar Roles.

Para insertar un rol hacemos clic en roles y después en insertar roles. Ingresamos el nombre del rol y hacemos clic en Insertar, para actualizar o eliminar primero debemos seleccionar un rol.



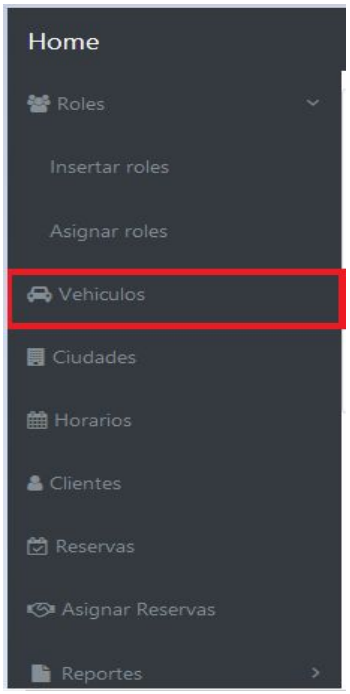
4.2. Asignar Roles

Para asignar roles hacemos clic en Asignar Roles, Buscamos un usuario haciendo clic en el ícono de buscar, y seleccionamos el rol o roles correspondientes, y hacemos clic en Agregar, para eliminar hacemos clic en el botón eliminar en el listado.

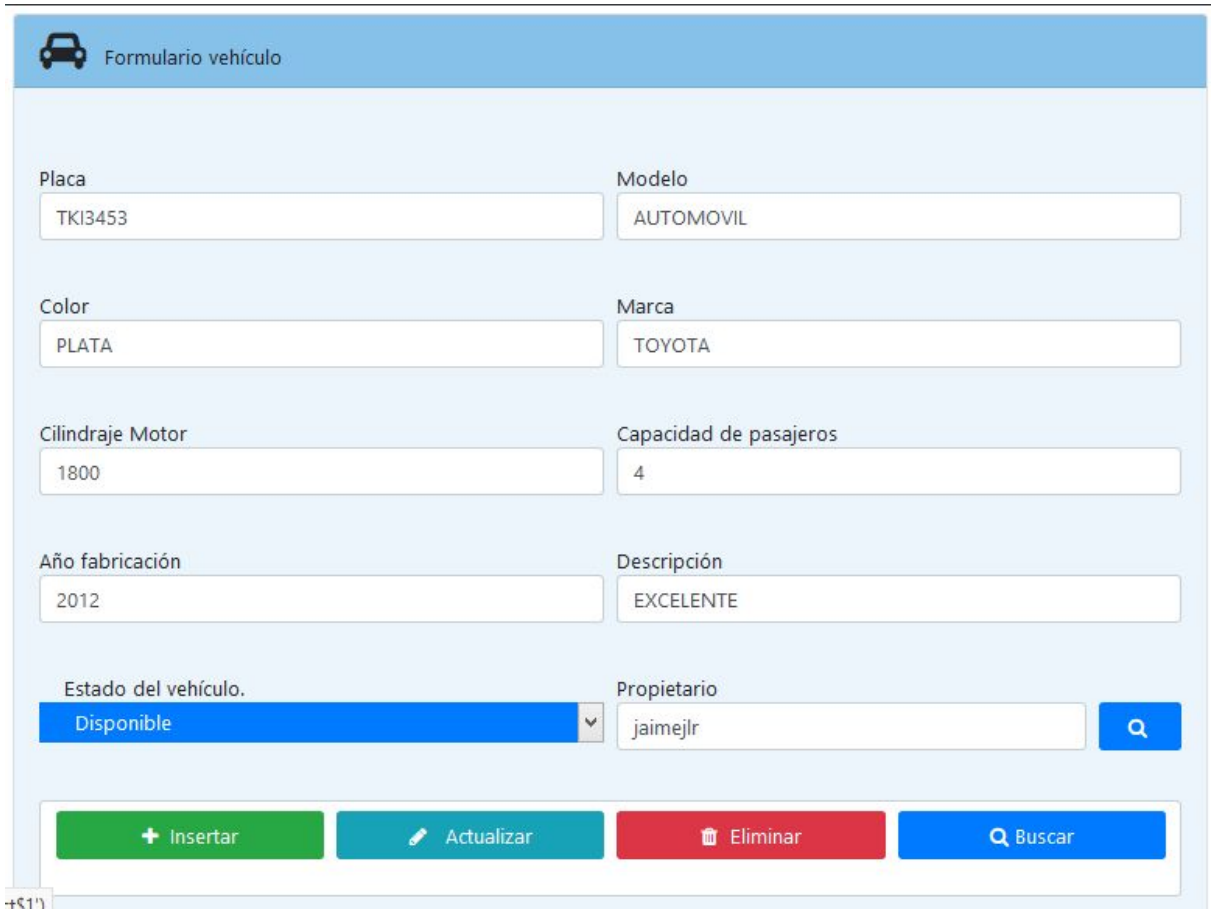


5. Vehículos.

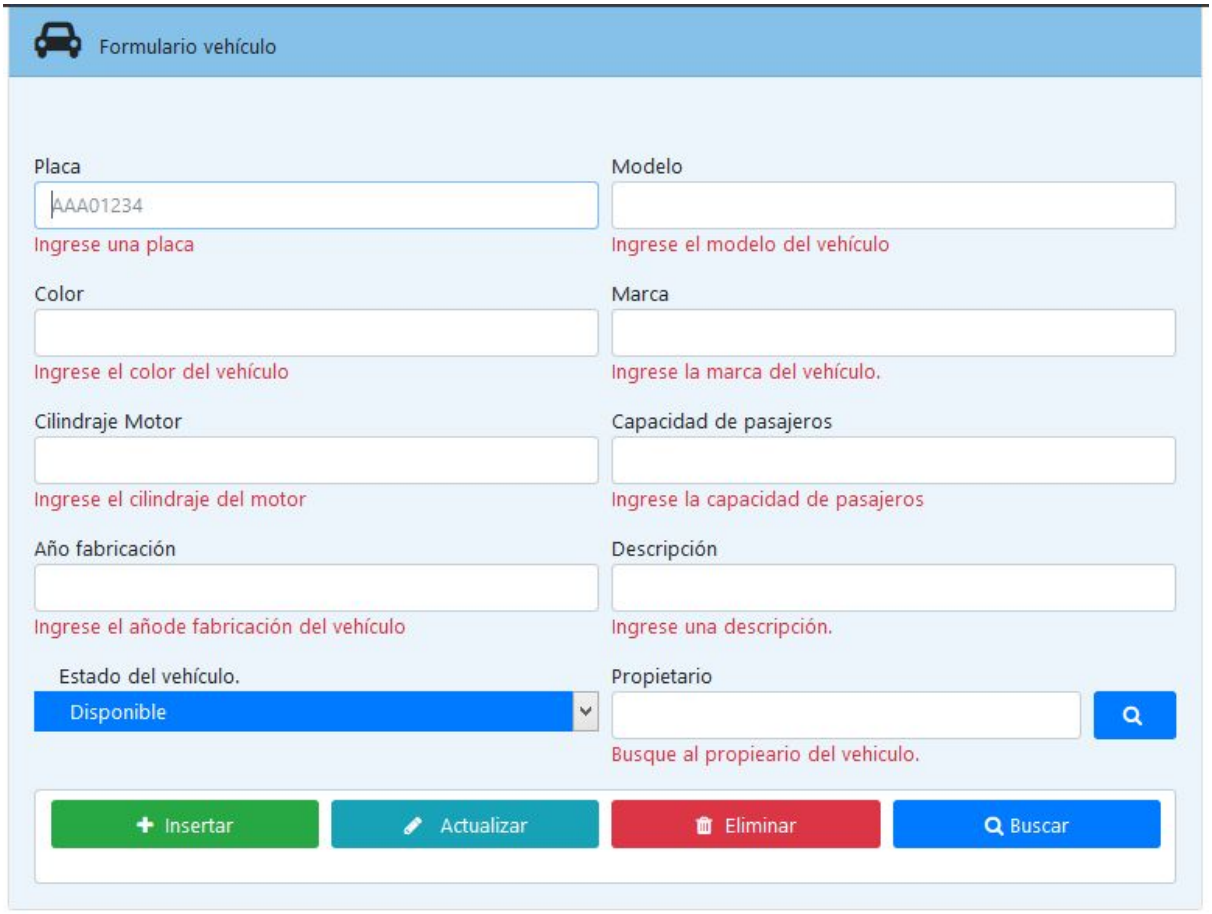
Hacer clic en vehículos.



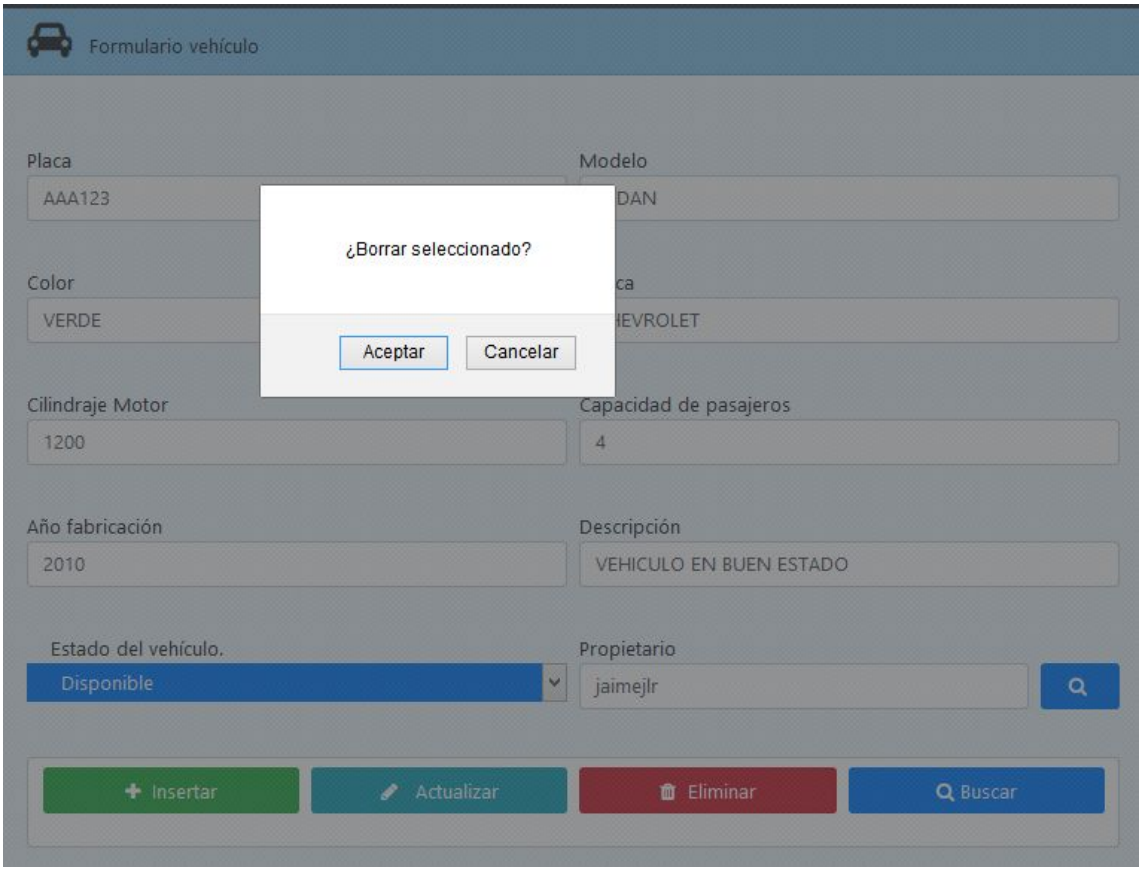
Completar todos los datos que sean necesarios y hacer clic en agregar.
Para actualizar o eliminar primero buscamos el vehículo y hacer clic en la acción correspondiente.



Al hacer clic en Insertar o actualizar el sistema notificará los campos que se deben llenar.

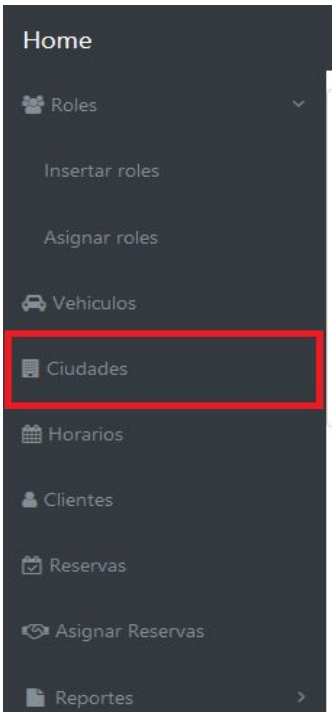


Para eliminar el sistema mostrará un mensaje de confirmación en donde podrá aceptar o cancelar.



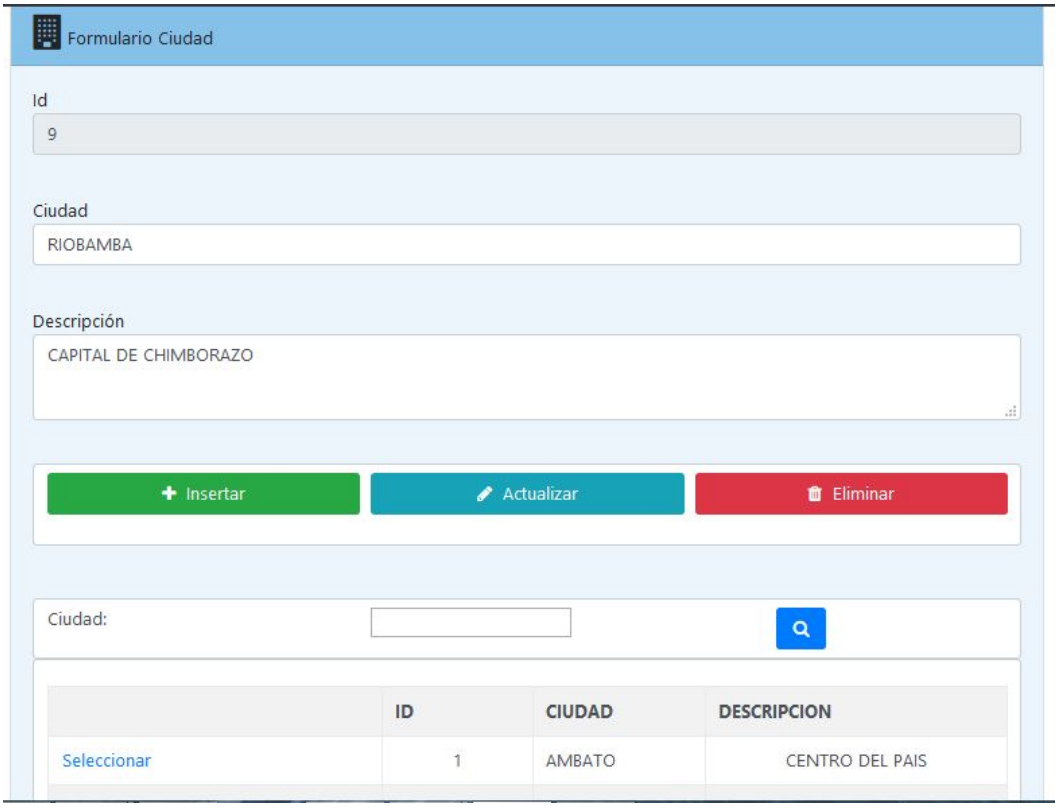
6. Ciudad

Para ingresar a gestionar las ciudades hacemos clic en el menú principal en ciudad.



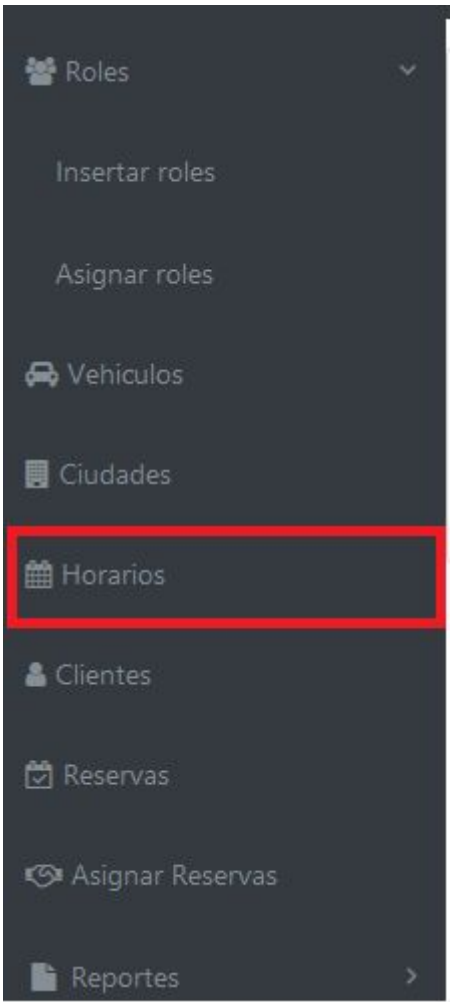
En la pantalla principal podemos Ingresar una nueva ciudad con el nombre y la descripción de la misma.
Para editar o eliminar se debe seleccionar una ciudad del listado.

Las ciudades que se ingresen serán los destinos que se ofertarán para el servicio de transporte.

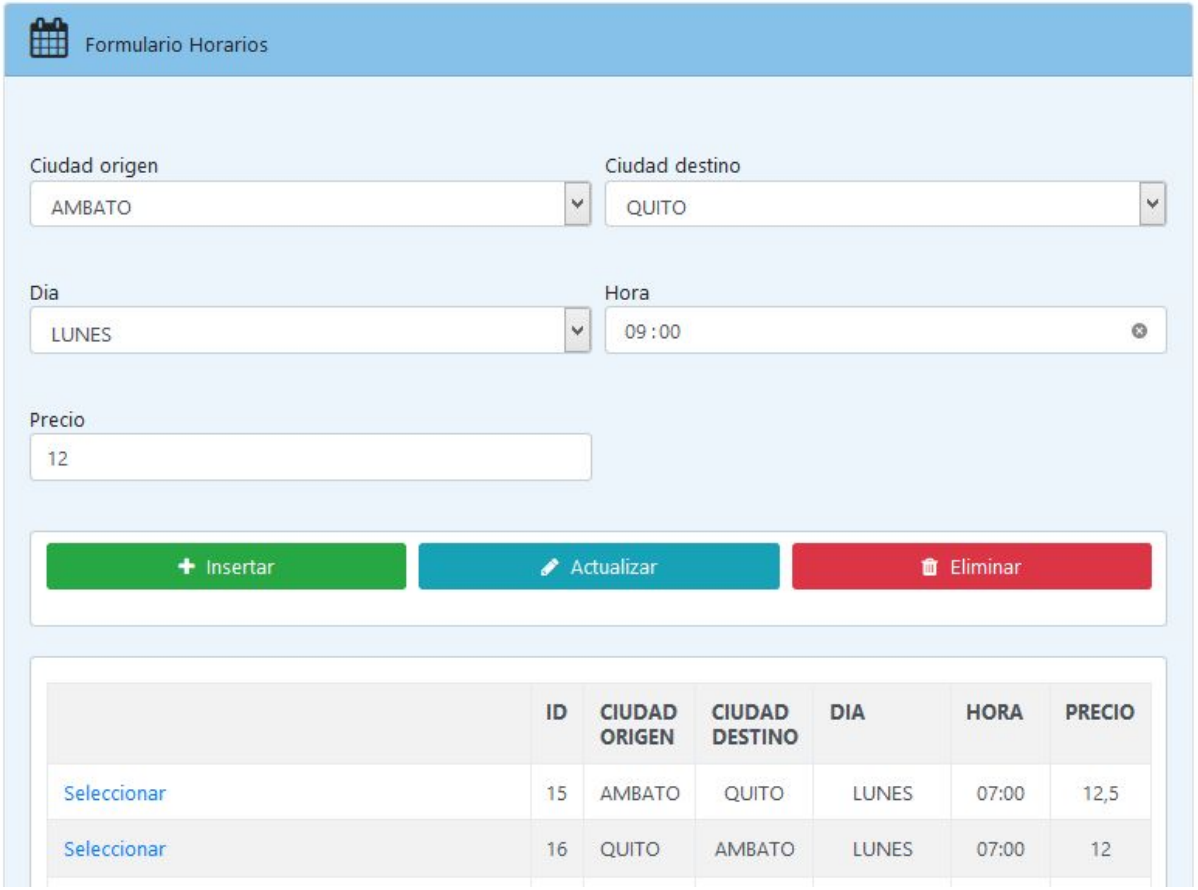


7. Horarios.

Para gestionar los horarios hacemos clic en horarios en el menú principal.

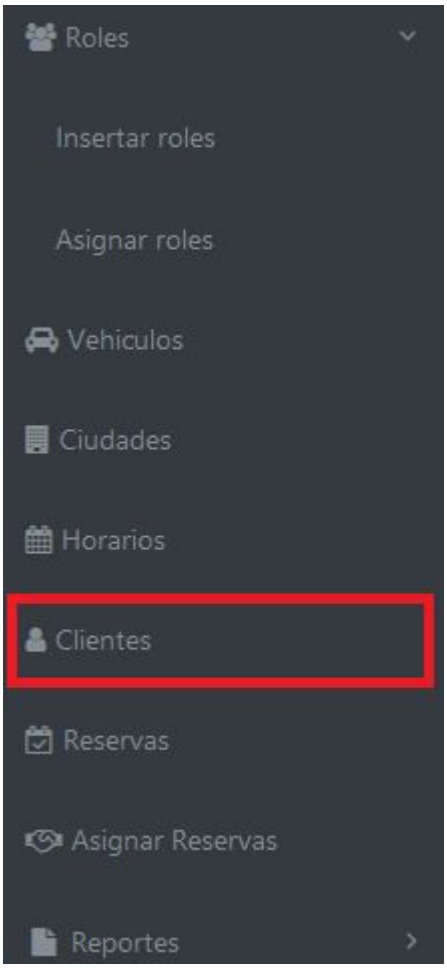


En la pantalla principal debemos seleccionar del listado la ciudad origen y la ciudad destino, seleccionar el día y escribir la hora y al final el precio, e insertar.
Para actualizar o eliminar se debe primero seleccionar el horario

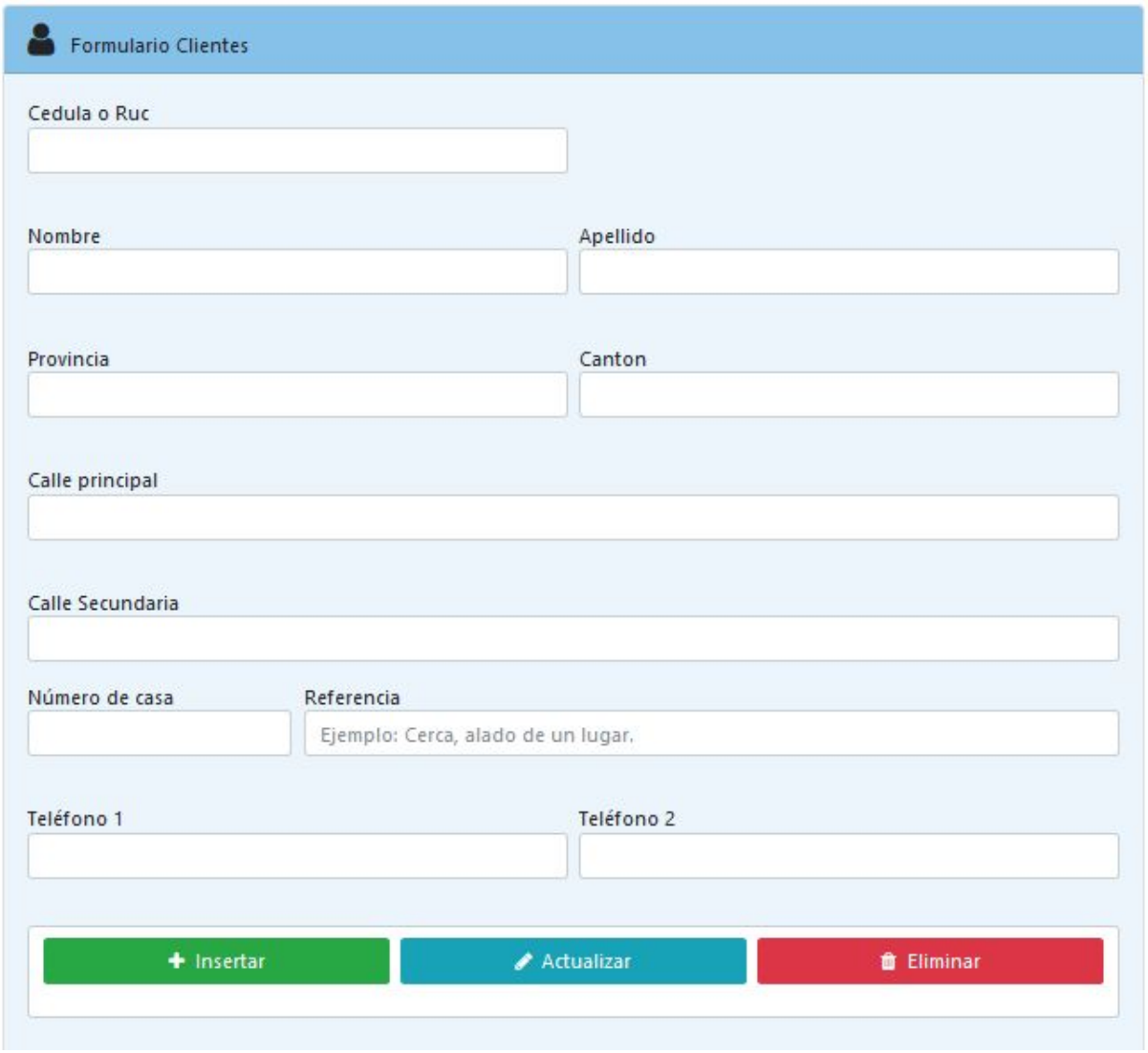


8. Clientes.

Para ingresar a clientes hacemos clic en el menú principal clientes.

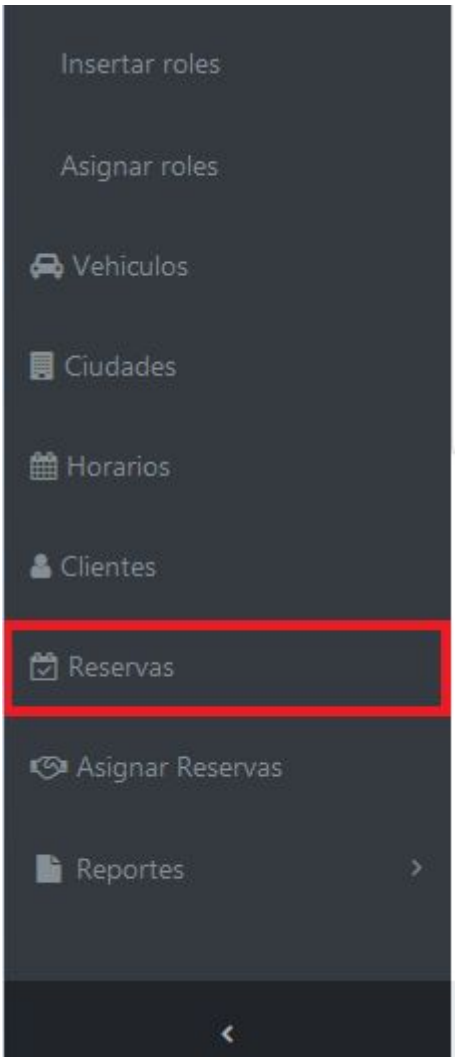


Completamos todos los datos necesarios y hacemos clic en insertar, para actualizar o eliminar primero debemos seleccionar del listado de abajo.


Formulario Clientes
Cedula o Ruc
Nombre
Apellido
Provincia
Canton
Calle principal
Calle Secundaria
Número de casa
Referencia
Ejemplo: Cerca, alado de un lugar.
Teléfono 1
Teléfono 2
Insertar
Actualizar
Eliminar

Clientes
ID CEDULA NOMBRE APELLIDO PROVINCIA CANTON CALLE_PRINCIPAL CALLE_SECUNDARIA N_CASA REFERENCIA
Seleccionar 1 1804677951 JAIME LOPEZ TUNGURAHUA AMBATO SN1 SN2 120 TERMINAL
Seleccionar 2 1802190981 CARLOS CRUZ PICHINCHA QUITO RUMIÑAHUI 10 DE AGOSTO 10-22 PARQUE CAROLINA
Seleccionar 7 1804677952 ANIBAL MOREJON TUNGURAHUA AMBATO CEVALLOS QUITO 345 CENTRO
Seleccionar 9 1804677953 PEDRO VALLE TUNGURAHUA AMBATO CEVALLOS MARTINEZ 3454 CENTRO
Seleccionar 10 1802190981 FRANCISCO ACOSTA TUNGURAHUA AMBATO AVATAHUALPA RUMINAUI SN REDONDELPOLICIA
Seleccionar 23 1802190981 SOFIA RAMOS COTOPAXI SALCEDO ANTONIO CLAVIJO JUAN LEON 1234 entrada a salcedo

9. Reservas.



Se debe primero seleccionar a un cliente y luego el horario en el que realizará el viaje, este formulario consta de dos partes, la primera con la información de origen y la otra en la que se debe poner la información de destino, o esto depende al momento de elegir al cliente, si los datos se usarán para el origen o destino.

De igual manera para actualizar o dar de baja una reserva se debe primero seleccionar del listado.

Formulario Reservas

Cliente

MOREJON ANIBAL

Horario

AMBATO - QUITO MIERCOLES 07:00

Numero de asientos

1

Numero de equipaje

2

Calle principal Origen

CEVALLOS

Calle Secundaria Origen

QUITO

Número de casa

345

Referencia

CENTRO

Teléfono Auxiliar

0984785126

Observación

SER PUNTUAL

Calle principal Destino

SN

Calle Secundaria Destino

SN

Fecha

2018-4-1

Referencia Destino

CENTRO

Latitud Destino

-0.23071

Longitud Destino

-78.49628

Estado

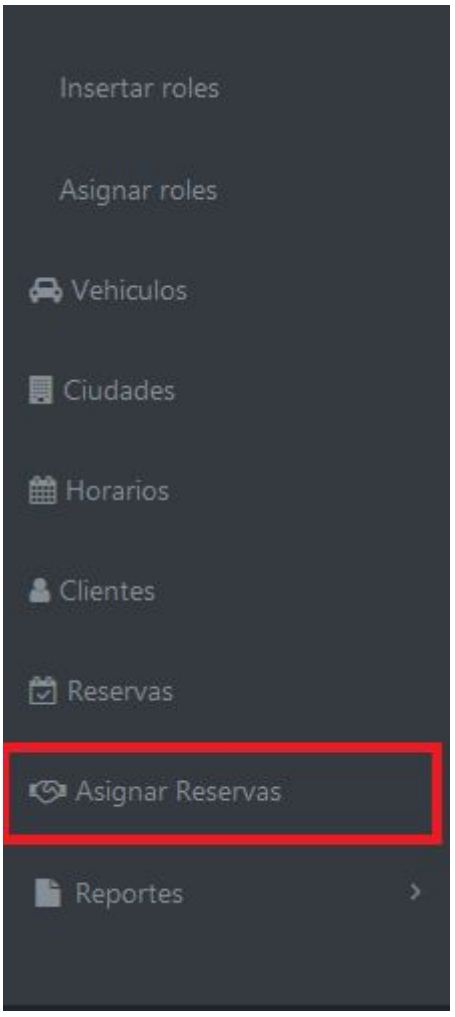
LLEGÓ AL DESTINO

+ Insertar | Actualizar | − Dar de baja

Reservas

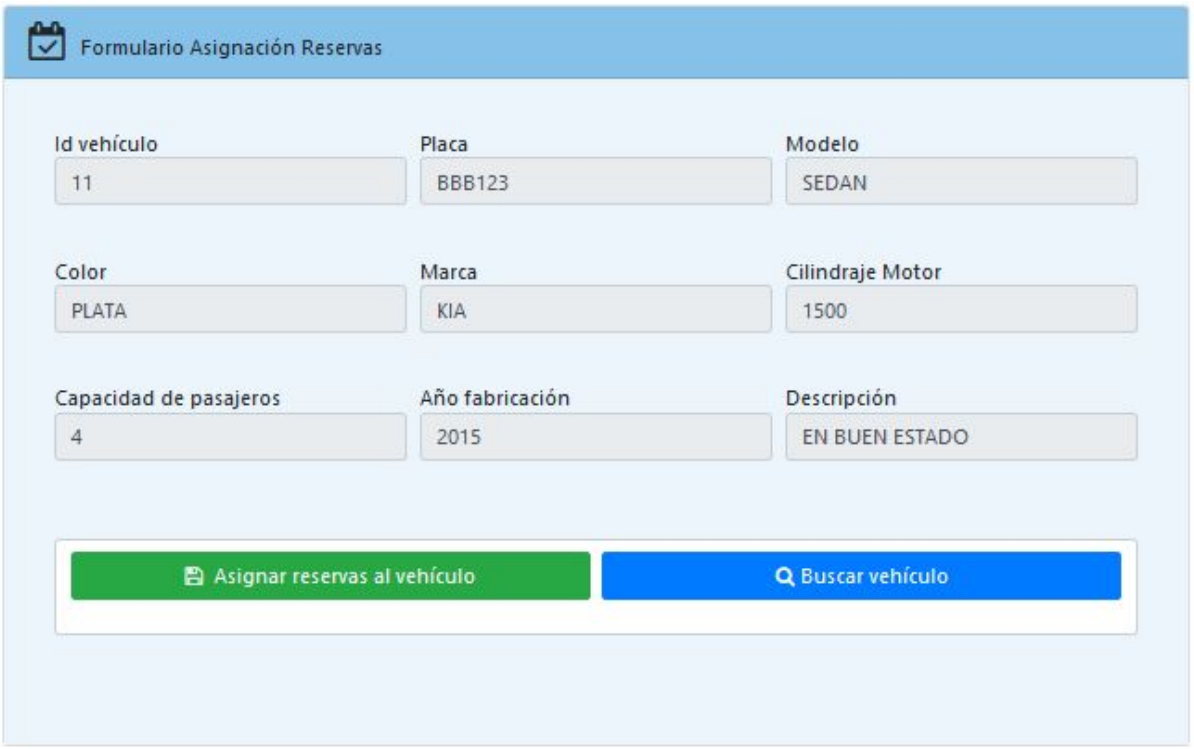
	ID	HORARIO	CLIENTE	ASIENTOS	EQIPAJE	CALLE_PRINCIPAL_ORIGEN	CALLE_SECUNDARIA_ORIGEN	NUMERO_CASA	REFERENCIA_ORIGEN
Seleccionar	1	AMBATO - QUITO LUNES 07:00	LOPEZ JAIME	2	1	CEVALLOS	ELOY ALFARO	40-15	MERCADO MODELO
Seleccionar	3	QUITO - AMBATO LUNES 07:00	LOPEZ JAIME	4	3	SN1	SN2	120	TERMINAL
Seleccionar	4	QUITO - AMBATO LUNES 07:00	CRUZ CARLOS	4	3	RUMIÑAHUI	10 DE AGOSTO	10-22	PARQUE CAROLINA
Seleccionar	6	QUITO - AMBATO LUNES 07:00	PALACIOS CARMEN	4	3	RUMIÑAHUI	10 DE AGOSTO	10-22	PARQUE CAROLINA
Seleccionar	7	AMBATO - QUITO LUNES 07:00	MOREJON ANIBAL	4	1	CEVALLOS	QUITO	345	CENTRO
Seleccionar	10	AMBATO - QUITO MIERCOLES 07:00	MOREJON ANIBAL	1	2	CEVALLOS	QUITO	345	CENTRO

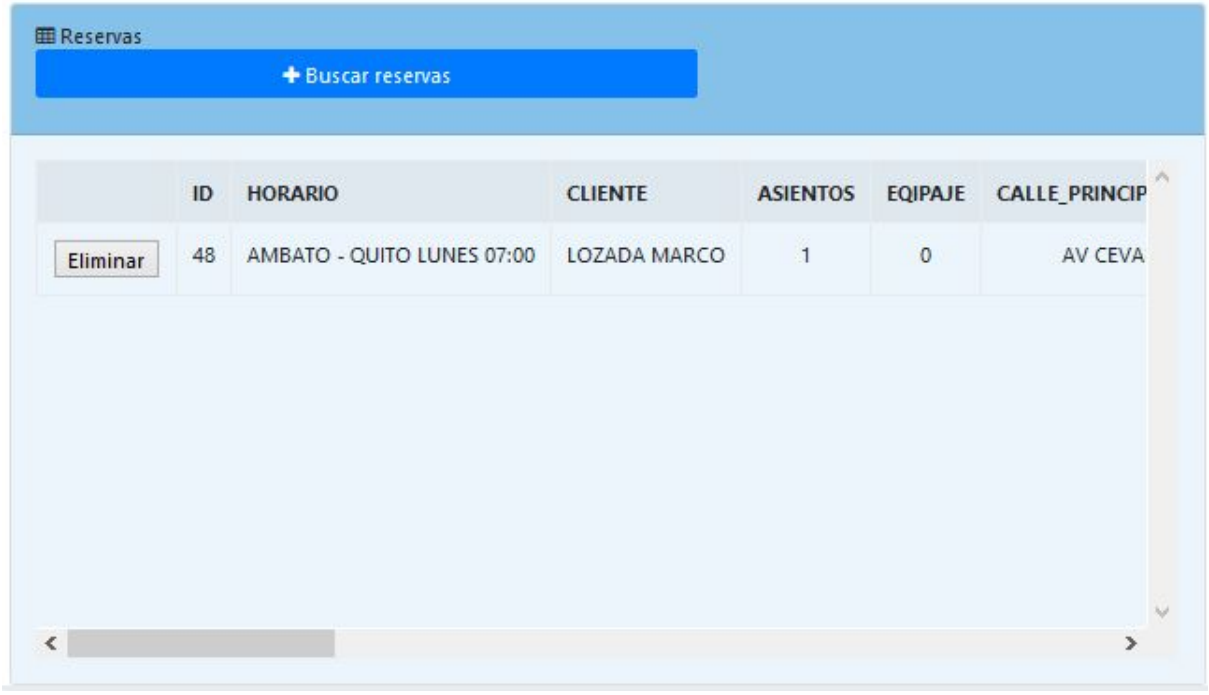
10. Asignar reservas.



En esta parte es para cuando sucede algo en el proceso de asignación, ya que el momento que se hace una reserva el sistema asigna la reserva a un vehículo automáticamente

Para hacer una asignación se debe primero buscar un vehículo, buscar la reserva y finalmente asignar la o las reservas al vehículo.



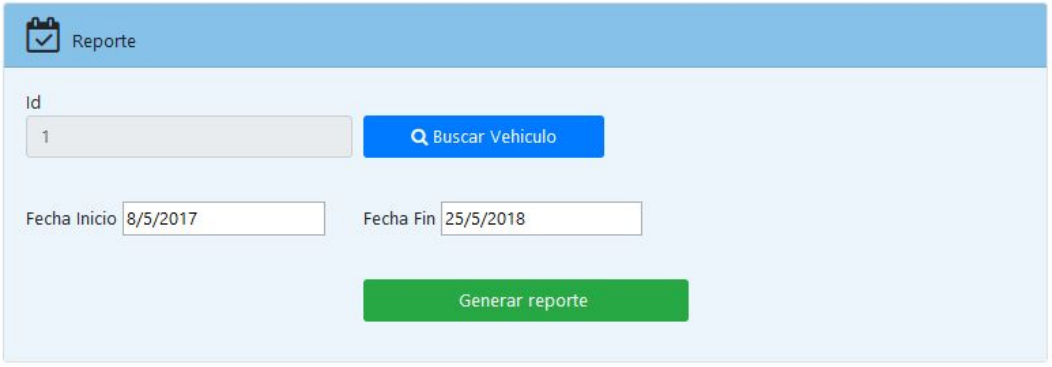


11. Reportes.

Para ingresar a la parte de reportes hacemos clic en el menú principal en reportes. Tenemos tres tipos de reportes, como se muestran a continuación.

11.1. Reporte Reserva por vehículo.

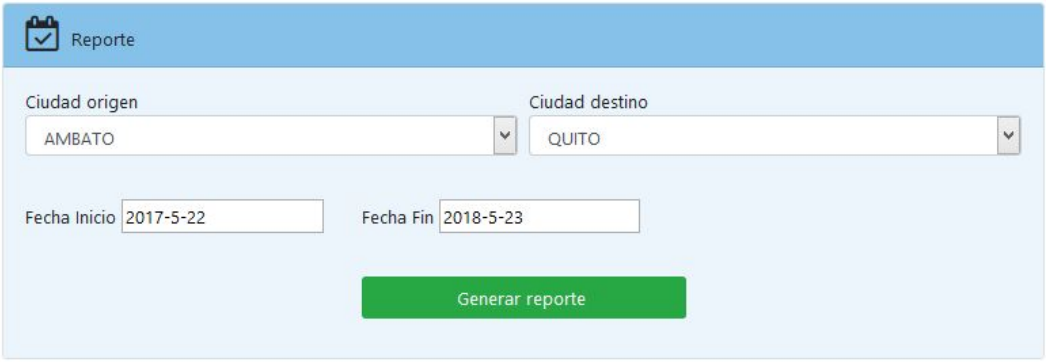
El primer reporte muestra los viajes realizados por un determinado vehículo en un rango de fechas determinado.
Debemos buscar el vehículo, y seleccionar la fecha inicio y fecha fin del reporte, y hacer clic en Generar reporte.



El reporte se visualizará de la siguiente manera.
En donde tenemos la información del vehículo y detallado los viajes realizados, y el precio que se debe cancelar.

11.2. Reporte clientes más frecuentes.

El segundo reporte muestra a los clientes más frecuentes que viajaron de una ciudad a otra en un determinado tiempo.
Se debe seleccionar la ciudad de origen y destino y clic en Generar reporte.



El reporte se muestra de la siguiente manera, los parámetros de búsqueda en la cabecera y el detalle del cliente, el origen y destino y el número de viajes realizados.

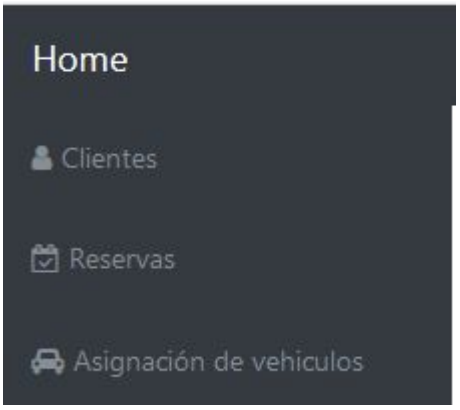
CLIENTE	ORIGEN	DESTINO	NUMERO VIAJES
RAMOS SOFIA	AMBATO	QUITO	4
SEVILLA DANIEL	AMBATO	QUITO	2
MOREJON ANIBAL	AMBATO	QUITO	2
ACOSTA FRANCISCO	AMBATO	QUITO	2
CASTRO HIRMA	AMBATO	QUITO	1
CRUZ CARLOS	AMBATO	QUITO	1
LOPEZ JAIME	AMBATO	QUITO	1
LOZADA MARCO	AMBATO	QUITO	1

12. Menú de asistente.

El menú del asistente consta de las opciones como se muestran a continuación.

13. Menú de cliente.

El menú del cliente consta de las siguientes opciones.



En la parte de clientes puede agregar clientes bajo su usuario.

Reservas

	ID	ESTADO	HORARIO	CLIENTE	ASIENTOS	EQIPAJE	CALLE_PRINCIPAL_C
Seleccionar	48	PENDIENTE	AMBATO - QUITO LUNES 07:00	LOZADA MARCO	1	0	AV CEVALLOS

Formulario Asignación Reservas

Id vehículo
11

Placa
BBB123

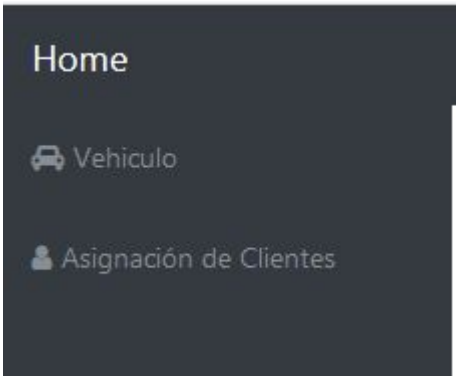
Modelo
SEDAN

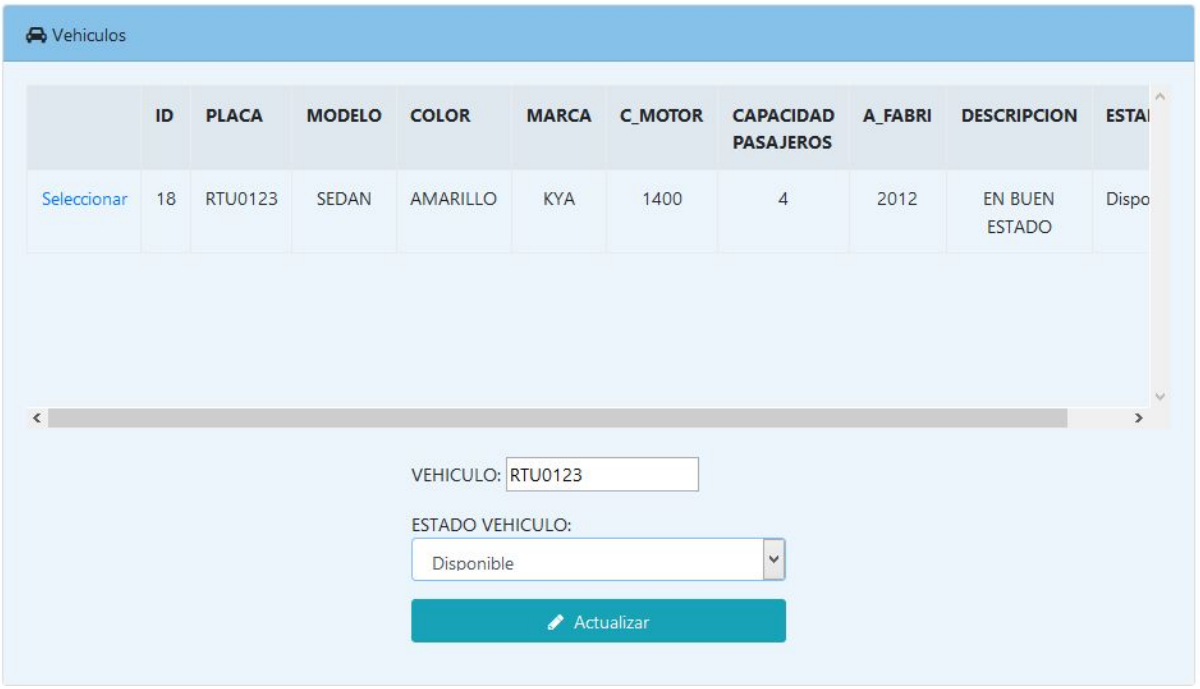
Color
PLATA

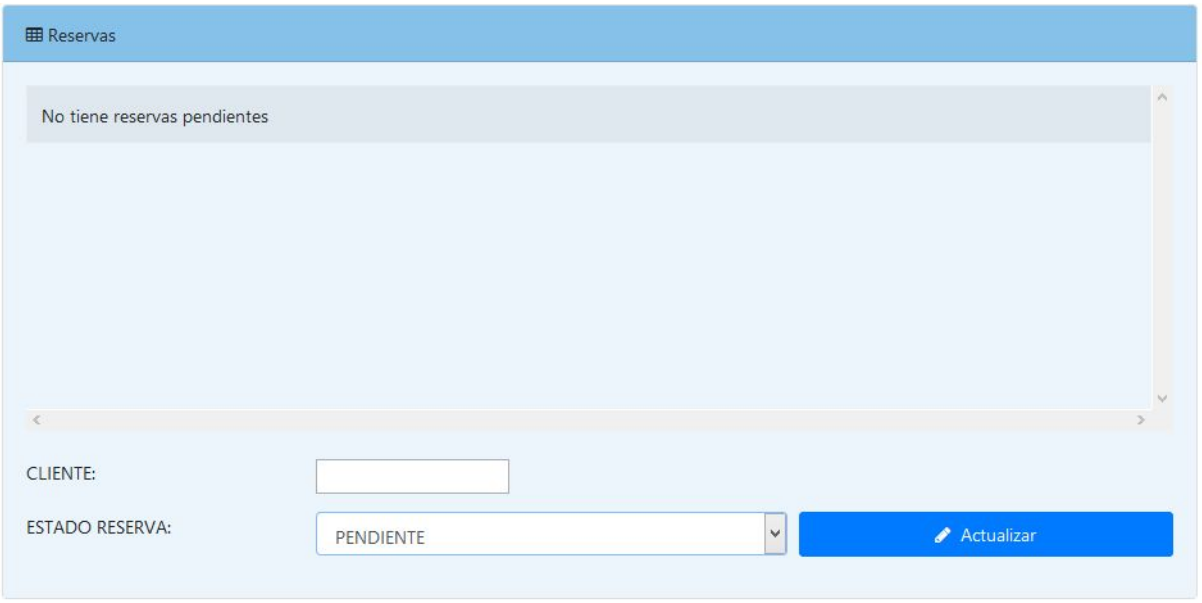
Marca
KIA

Descripción
EN BUEN ESTADO

14. Menú chofer.




Vehiculos
ID
PLACA
MODELO
COLOR
MARCA
C_MOTOR
CAPACIDAD PASAJEROS
A_FABRI
DESCRIPCION
ESTA
Seleccionar
18
RTU0123
SEDAN
AMARILLO
KYA
1400
4
2012
EN BUEN ESTADO
Dispo
VEHICULO: RTU0123
ESTADO VEHICULO:
Disponible
Actualizar


Reservas
No tiene reservas pendientes
CLIENTE:
ESTADO RESERVA:
PENDIENTE
Actualizar

Ubicación en el mapa