UNIVERSIDAD POLITÉCNICA SALESIANA
SEDE QUITO

CARRERA:
INGENIERÍA DE SISTEMAS

Trabajo de titulación previo a la obtención del título de:
Ingenieras de Sistemas

TEMA:
APLICACIÓN WEB PARA COMPARTIR VEHÍCULO

AUTORAS:
BENAVIDES ANRANGO EVELYN JOHANA
TIXILEMA AZOGUEZ SENAIDA MAYTE

TUTOR:
PROAÑO ORELLANA JULIO RICARDO

Quito, septiembre el 2020

CESIÓN DE DERECHOS DE AUTOR

Nosotras, Evelyn Johana Benavides Anrango con documento de identificación N°.1718748997 y Senaida Mayte Tixilema Azoguez con documento de identificación N°.0202385142, manifestamos nuestra voluntad y cedemos a la Universidad Politécnica Salesiana la titularidad sobre los derechos patrimoniales en virtud de que somos autoras del trabajo de titulación con el tema:

APLICACIÓN WEB PARA COMPARTIR VEHICULO, mismo que ha sido desarrollado para optar por el título de INGENIERAS DE SISTEMAS en la Universidad Politécnica Salesiana, quedando la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente.

En aplicación a lo determinado en la Ley de Propiedad Intelectual, en nuestra condición de autoras nos reservamos los derechos morales de la obra antes citada. En concordancia, suscribimos este documento en el momento que hacemos entrega del trabajo final en digital a la Biblioteca de la Universidad Politécnica Salesiana.

EVELYN JOHANA
BENAVIDES ANRANGO
CI: 171874899-7

SENAIDA MAYTE
TIXILEMA AZOGUEZ
CI: 020238514-2

Quito, septiembre del 2020

DECLARATORIA DE COAUTORÍA DEL DOCENTE TUTOR

Yo declaro que bajo mi dirección y asesoría fue desarrollado el Proyecto Técnico, **APLICACIÓN WEB PARA COMPARTIR VEHICULO** realizado por Evelyn Johana Benavides Anrango y Senaida Mayte Tixilema Azoguez, obteniendo un producto que cumple con todos los requisitos estipulados por la Universidad Politécnica Salesiana, para ser considerados como trabajo final de titulación.

Quito, septiembre del 2020

JULIO RICARDO PROAÑO ORELLANA
CI: 0103909412

DEDICATORIA

Este proyecto se lo dedico en primer lugar a DIOS por iluminar y bendecir mi camino en cada etapa de mi vida, por no soltarme en ningún momento, por permitirme cumplir este sueño y darme la fortaleza para no rendirme jamás.

A mi madre Susana Anrango, por ser un ejemplo para seguir, por su amor incondicional, sus consejos que me guían siempre en la vida y su compañía en todo momento. Gracias por creer en mí y apoyarme en todo sentido, sin su apoyo no sería posible alcanzar esta meta.

A mi padre Luis Benavides por su amor y sacrificio, por desear y anhelar lo mejor para mí y por estar pendiente de mí en todo momento.

A mi hermano Christian Benavides por su amistad y amor, por estar en muchos momentos a mi lado, por motivarme y enseñarme a nunca rendirme y a ser mejor cada día.

Y a todas esas personas que han estado directa o indirectamente en el desarrollo de este proyecto, ya que este logro también es gracias a ellos que, aunque parecía interminable, con su ayuda personal, con sus llamadas palabras de aliento me han acompañado y demostrado su apoyo y cariño hacia mí.

Evelyn Johana Benavides Anrango

DEDICATORIA

El presente proyecto de titulación está dedicado en primera instancia para Dios, ya que gracias a él tuve la fortaleza de nunca rendirme a pesar de las adversidades presentadas en el camino. Para mis amados padres José Aurelio Tixilema y María Blanca Azoguez, por su infinito amor, por sus consejos en momentos de flaqueza, por su apoyo incondicional en cada etapa de mi vida, pero sobre todo por su apoyo económico para poder estudiar y que este sueño sea una realidad. Para mi amado hijo Johan Elian Galarraga Tixilema, que con su llegada cambio por completo mi vida, por ser mi motor de ser alguien mejor todos los días como profesional y como mujer de ética, que, con cada sonrisa, con cada mirada y como tanta dulzura me inculcabas a no rendirme a pesar de lo duro que fue el camino. ara mis queridos hermanos José Luis, Adriana, Adolfo y Vanessa Tixilema Azoguez por su amor y apoyo en cada etapa de mi vida.

Senaida Mayte Tixilema Azoguez

AGRADECIMIENTO

Nuestros sinceros agradecimientos a la Universidad Politécnica Salesiana que contribuyó en nuestra formación como profesional, a cada uno de nuestros profesores por sus conocimientos impartidos con mucho esmero para formar profesionales con ética y valores. A nuestros compañeros que formaron parte de nuestra formación académica por los momentos compartidos para lograr un sueño en común, a nuestro tutor el Ingeniero Julio Ricardo Proaño Orellana mil gracias por su apoyo desmedido y por el acompañamiento en cada etapa del proyecto.

Evelyn Johana Benavides Anrango

Senaida Mayte Tixilema Azoguez

ÍNDICE

INTRODUCCIÓN 1

Antecedentes 1

Planteamiento del problema 3

Justificación 4

Objetivos 4

Objetivo General 4

Objetivos Específicos 4

Metodología 4

Capítulo 1 6

Marco Teórico 6

1. Fundamentos de programación web 6

1.1. Arquitectura cliente-servidor 6

1.1.1. Cliente 7

1.1.2. Servidor 7

1.1.3. Características de la arquitectura cliente-servidor 8

1.2. Tecnología “Front-End” 8

1.3. Tecnología “Back-End” 8

1.4. Servidor web 9

1.4.1. Funcionamiento de un servidor web 9

1.4.1.1. Protocolo HTTP (Hypertext Transfer Protocol) 9

1.4.2. DNS (Domain Name System) 10

1.5. Arquitectura MVC (Modelo – Vista – Controlador) 11

1.5.1. Modelo 11

1.5.2. Vista 11

1.5.3. Controlador 12

1.5.4. Ciclo de vida de MVC 12

1.6. Fundamentos de bases de datos relacionales 13

1.6.1. Modelo de datos 13

1.6.2. Diseño de bases de datos relacionales 15

1.6.3. Búsquedas en bases de datos relacionales 17

1.7. Fundamentos de Scrum 18

1.7.1. Product Owner (Dueño del Producto) 21

1.7.2. Artefactos de Scrum 22

1.7.3. Eventos 22

1.8. Geolocalización 24
1.9. Economía colaborativa 24
1.9.1. Ejemplos de Economía Colaborativa. 25
Capítulo 2 26
Análisis del Sistema 26
2.1. Alcance del proyecto 26
2.2. Análisis de requerimientos 27
2.3. Análisis de Factibilidad 29
2.3.1. Factibilidad Técnica. 29
2.3.2. Factibilidad Operacional. 31
2.4. Diagrama del proceso 34
2.5. Implementación de la Metodología. 35
2.6. Definición de los roles. 35
2.7. Cronograma del Proyecto (Hitos). 36
2.8. Plan de trabajo (Sprints) 38
2.9. Resultados esperados. 41
Capítulo 3 42
Diseño del Sistema 42
3.1. Especificación de los casos de uso. 42
3.1.1. Caso de uso relacionado con el usuario conductor 43
3.1.2. Caso de uso relacionado con el usuario cliente (pasajero) 44
3.2. Diseño de diagrama de clases. 45
3.2.1. Diseño de diagrama de clases 46
3.3. Diseño del diagrama de secuencias. 47
3.3.1. Diagrama de secuencia para el usuario Conductor 49
3.3.2. Diagrama de secuencia para el usuario Cliente 50
3.4. Diseño de la base de datos. 51
3.4.1. Modelo Entidad -Relación 52
3.4.2. Modelo Relacional 53
3.5. Arquitectura utilizada para la implementación. 54
3.5.1. Diagrama de la arquitectura en tres capas de la aplicación web para compartir vehículo. 55
Capítulo 4 56
Desarrollo del Sistema 56

4. Herramientas de desarrollo 56
4.1. PHP 56
4.1.1. Funcionamiento 57
4.2. Framework Laravel. 58
4.2.1. Estructura de directorios. 59
4.3. Sistema gestor de base de datos. 61
4.3.1. MySQL. 61
4.4. API de Google para geolocalización. 62
4.4.1. API key de Google. 62
4.4.2. API de JavaScript de Maps 63
4.5. Estructura utilizada para el desarrollo. 64
4.5.1. Configuraciones Iniciales 65
4.6. Conexión con la base de datos. 66
4.6.5. Desarrollo del módulo del usuario conductor. 75
4.6.5.1. Especificación de modelos. 75
4.6.5.2. Controladores principales. 76
4.6.5.3. Rutas utilizadas para el acceso. 79
4.6.5.4. Diseño de interfaces. 79
4.7. Desarrollo del módulo de usuario cliente. 84
4.7.1. Especificación de modelos. 84
4.7.2. Controladores principales. 85
4.7.3. Rutas utilizadas para el acceso. 86
4.7.4. Diseño de interfaces. 86
Capítulo 5 89
Pruebas del Sistema 89
5.1. Pruebas de rendimiento 89
5.2. Pruebas de funcionalidad 92
5.3. Pruebas de usabilidad 95
CONCLUSIONES 101
RECOMENDACIONES 102
LISTA DE REFERENCIAS 103

ÍNDICE DE FIGURAS

Figura 1. Estructura básica cliente-servidor 7
Figura 2. Diagrama de Procesos de la aplicación web 34
Figura 3. Segmentación del sistema 38
Figura 4.Modulo de Diseño de la base de datos 38
Figura 5.Módulo de Seguridad 39
Figura 6.Modulo usuario conductor 40
Figura 7.Modulo usuario cliente 40
Figura 8. Caso de uso usuario conductor 43
Figura 9. Caso de uso usuario cliente 44
Figura 10. Diagrama de Clase Módulo Seguridad 46
Figura 11. Diagrama de Clase de la Aplicación Web 47
Figura 12. Diagrama de secuencias Usuario Conductor 49
Figura 13. Diagrama de secuencias Usuario Cliente 50
Figura 14. Diagrama Entidad-Relación 52
Figura 15. Diagrama Relacional 53
Figura 16.Diagrama de la arquitectura de tres capas 55
Figura 17. Esquema de funcionalidad de PHP 57
Figura 18. Estructura de directorios de Laravel 59
Figura 19. Pagina para agregar el API KEY al proyecto: 63
Figura 20.API de Google Maps JavaScript 64
Figura 21.Arquitectura de la aplicación web 65
Figura 22. Configuración del fichero. env 67
Figura 23. Migraciones realizadas para crear las tablas que el SGBD requiere 67
Figura 24. Migraciones realizadas con éxito 68
Figura 25. Modelos del módulo seguridad 69
Figura 26. Creación del modelo Usuario 70
Figura 27. Modelo Usuario para el módulo de seguridad 70
Figura 28. Controladores que utiliza el módulo de seguridad 71
Figura 29. Creación del controlador UsuarioController 71
Figura 30. Controlador UsuarioController 72
Figura 31. Rutas de acceso al módulo de seguridad 73
Figura 32. Ventana para iniciar sesión 74
Figura 33. Ventana de registro de los usuarios 74
Figura 34. Modelos del módulo conductor 75
Figura 35. Creación del modelo Vehículo 76
Figura 36. Modelo Vehículo para el módulo del usuario conductor 76
Figura 37. Controladores que utiliza el módulo del usuario conductor 77
Figura 38. Creación del controlador VehículoController 77
Figura 39. Controlador UsuarioController 77
Figura 40. Rutas de acceso al módulo del usuario conductor 79
Figura 41.Ventana principal del usuario conductor 80
Figura 42.Ventana con la presentación de la información del vehículo 80
Figura 43. Ventana para registrar el vehículo 80
Figura 44.Formulario para registrar y visualizar las rutas 81
Figura 45.Formulario para registrar viaje 82
Figura 46.Pantalla de Solicitudes recibidas 82
Figura 47.Ventana donde el usuario puede aceptar o rechazar la solicitud 83
Figura 48.Ventana con los detalles de los viajes realizados 83

Figura 49.Ventana para realizar la calificación al usuario 84
Figura 50. Modelos del módulo usuario..... 84
Figura 51. Controladores que utiliza el módulo del usuario conductor 85
Figura 52. Rutas de acceso al módulo del usuario pasajero..... 86
Figura 53.Ventana Principal del Usuario cliente 86
Figura 54. Ventana con las rutas asociadas al punto ingresado 87
Figura 55. Solicitud de la ruta 87
Figura 56. Ventana de viajes realizados..... 88
Figura 57.Ventana para registrar la calificación 88
Figura 58. Pruebas de Rendimiento del usuario conductor..... 90
Figura 59. Pruebas de Rendimiento del usuario cliente 90
Figura 60. Visitas al servidor del usuario conductor..... 91
Figura 61. Visitas al servidor del usuario cliente 91
Figura 62. Tiempos de respuesta del usuario conductor 92
Figura 63. Tiempos de respuesta del usuario cliente 92
Figura 64. Uso de la aplicación web 95
Figura 65. Complejidad de la aplicación web 96
Figura 66. Diseño de la aplicación web 96
Figura 67. Manejo de la aplicación web 97
Figura 68. Manejo de la aplicación web 97
Figura 69. Uso de la aplicación web 98
Figura 70. Uso de la aplicación web 98
Figura 71. Diseño de la aplicación web 99
Figura 72. Diseño de la aplicación web 99
Figura 73. Diseño de la aplicación web 100

ÍNDICE DE TABLAS

Tabla 1. Ejemplo de Economía Colaborativa..... 25
Tabla 2. Hardware Mínimo requerido..... 30
Tabla 3. Procesos y recurso Humanos del proyecto..... 32
Tabla 4.Definición de rol de los involucrados del proyecto 35
Tabla 5. Cronograma de Hitos 37
Tabla 6.Directorios de un proyecto de Laravel..... 60
Tabla 7. Configuraciones iniciales de los componentes 65
Tabla 8.Método principales del controlador 72
Tabla 9.Métodos principales de los controladores para el usuario conductor. 78
Tabla 10.Modelos y Controladores del Módulo usuario cliente 85
Tabla 11.Prueba funcional modulo usuario conductor..... 93
Tabla 12.Prueba funcional Modulo Usuario Cliente/Pasajero..... 94

RESUMEN

El presente documento detalla el proceso de desarrollo de una aplicación web para compartir vehículo. El propósito de crear esta aplicación es que las personas que cuentan con un vehículo como medio de transporte puedan brindar su servicio mediante el uso de economía colaborativa y de esta manera poner en contacto a las personas que deseen un transporte seguro con los conductores ofertantes, facilitando este proceso mediante el uso del prototipo creado.

La aplicación web brinda al usuario final dos potentes alternativas, según su necesidad puede registrarse como "conductor" y tener la posibilidad de registrar su vehículo con todas sus características, crear su propia ruta, ubicar puntos referenciales dentro de Google Maps y de esta manera crear un viaje que será visible para el usuario que lo solicite. Por otro lado, el usuario "pasajero" tendrá la posibilidad de buscar un viaje acorde a su destino basándose en las rutas publicadas, enviar una solicitud para hacer uso del servicio y contactarse con el conductor del vehículo.

El resultado final es un prototipo de fácil manejo que cumple con el proceso descrito anteriormente, brindando así una alternativa diferente de servicio de transporte, tanto para las personas ofertantes (conductor) como para las personas que harán uso del servicio (pasajero).

ABSTRACT

This document details the process of developing a vehicle-sharing web application. The purpose of creating this application is that people who have a vehicle as a means of transport can provide their service through the use of collaborative economy and thus contact people who want a safe transport with the providers, facilitating this process by using the prototype created.

The web application provides the end-user two powerful alternatives, according to their need he or she can register as a "driver" and have the possibility to register their vehicle with all its features, create their own route, locate landmarks within Google Maps and in this way create a trip that will be visible to the user who requests it. On the other hand, the "passenger" user will have the possibility to search for a trip according to his or her destination based on the published routes, send a request to make use of the service and contact the driver of the vehicle.

The end result is an easy-to-use prototype that complies with the process described above, thus providing a different alternative of transportation service, both for the providers (driver) and for the people who will make use of the service (passenger).

INTRODUCCIÓN

Antecedentes

Desde que inició la era de los automóviles en el año de 1890. Los países como Estados Unidos, Japón y China son los principales países líderes en el mercado automotriz. Así, en el año 1929 existían alrededor de 32.028.500 automóviles en todo el mundo. De los cuales Estados Unidos producía el 90% de ellos. Luego, en el año 2006 Japón lideró el mercado hasta el año 2009. Actualmente se encuentra en tercer lugar con 9.2 millones de unidades. Mientras que China en el año 2015 duplicó la producción de Estados Unidos con 24.5 millones de unidades. (hoylosAngeles, 2016). De acuerdo con los estudios realizados por Enrique Kogan, en el año 2016 existían más de 1,2 Billones de automóviles en el planeta tierra, esto representa un aumento del 70.5% de los 700 millones de automóviles que se encontraban en las vías hace 11 años, el crecimiento de la población, la distancia en la que se encuentran los sectores comerciales, educativas, centros de salud, son de los factores para que muchas personas opten por adquirir vehículos personales para movilizarse,esto es uno de los causantes de la contaminación ambiental en todo el mundo. (ecologistasenaccion, 2019). Por otro lado, Según los resultados que obtuvieron, de los estudios que realizaron ANETA (Automóvil Club del Ecuador),” el 52% de emisiones de CO_2 proviene del exceso de tráfico”. Diego Enríquez, director de Cambio Climático de la Secretaría de Ambiente de Quito afirma que, “el transporte que más CO_2 genera son los vehículos livianos” (ANETA, 2019).

Movilidad en la ciudad de Quito

Julio Puga, director de la Agencia Metropolitana de Tránsito (AMT), menciona que en la actualidad circulan en Quito alrededor de 650.000 vehículos, esto provoca que “La ciudad ocupe el vigésimo sexto lugar en un ranking de las ciudades en el mundo, con problemas de congestión vehicular, por esta razón se maneja las medidas, del pico y placa, con el finde sacar

alrededor de 20 y el 30% de vehículos, en las horas pico". (Carvajal, 2019). En base a la problemática que tiene la ciudad, se plantea el desarrollo de una aplicación web para compartir vehículo. Esta aplicación tiene como propósito poner en contacto a las personas para que puedan compartir vehículo. En particular, este proyecto se enfoca en los estudiantes, docentes, administrativos, de la Universidad Politécnica Salesiana Campus sur. A continuación, se detallan aspectos de la propuesta.

Nuestra propuesta

Con el uso de la aplicación para compartir vehículo, se espera que facilite la comunicación entre las personas, que quieren compartir su vehículo. Dónde el usuario conductor como el usuario cliente obtendrán los siguientes beneficios:

- La aplicación permitirá poner en contacto al cliente con el usuario conductor.
- El usuario conductor tendrá la posibilidad de compartir gastos del viaje.
- El usuario cliente tendrá la facultad de realizar viajes a un costo accesible.
- Los usuarios cliente podrá reservar su viaje.

En la actualidad existen muchas aplicaciones que permiten la gestión de viajes en vehículo compartido. El objetivo de este tipo de aplicaciones es economizar los gastos que implica realizar un viaje. Además, al trasladarse varias personas en un solo vehículo, se pueda disminuir la congestión vehicular que existe en la ciudad.

Trabajos relacionados

En la actualidad existen diversas aplicaciones en Internet que permiten realizar viajes en vehículos compartidos. A continuación, se mencionan las más populares.

BlaBlaCar

Está vigente en España desde el año 2010, actualmente cuenta con más de 85 millones de usuarios en 22 países de los cuales 5 millones se encuentran en España, el objetivo de la

aplicación es poner en contacto a personas que desean realizar un viaje con una trayectoria común (blog.blablacar, 2019).

Conduzco.es

La aplicación de origen alemana permite localizar a personas que están planeando viajar por alguna ruta en común. El objetivo de los viajes compartidos es ponerse de acuerdo con el propietario del vehículo para compartir gastos como una forma de ayudar a reducir la contaminación del medio ambiente (elblogalternativo, 2011).

Amovens.com

Es la aplicación web que permite a los usuarios llegar a cualquier parte de España mediante el alquiler de Vehículos y viajes compartidos (amovens, 2009).

Viajamosjuntos.com

Es una página web gratuita y sencilla de manejar utiliza en España y Argentina, creado en el año 2004, para que las personas se puedan ponerse en contacto y compartir trayectos en coche (Riera, 2004).

Planteamiento del problema

En la actualidad las personas de la Universidad Politécnica Salesiana Campus Sur no tienen una forma de comunicarse para coordinar viajes.

De la misma forma, no existe un sistema de transporte universitario para los estudiantes de la UPS Sede Quito Campus Sur. Sin embargo, hay docentes, estudiantes y personal administrativo que se trasladan al lugar con vehículos personales.

En muchas ocasiones los vehículos tienen un solo ocupante, lo cual es ineficiente ya que se podría compartir el vehículo, con más personas, reduciendo así la cantidad de vehículos que circulan dentro del campus universitario con lo cual se podría:

- Optimizar el espacio físico
- Reducir la contaminación ambiental

- Reducir el tráfico

De modo que, la problemática principal es la carencia de una aplicación web donde los estudiantes, docente y administrativo puedan acceder a buscar el servicio de un vehículo compartido, el mismo que puede ser ofertado por los mismos estudiantes, docentes y personal administrativo de UPS con las mejores opciones dependiendo de las rutas.

Justificación

El presente trabajo, surge de la necesidad que tenemos como estudiantes, con el propósito de contar con un espacio a través del cual se pueda conocer si alguna persona está compartiendo un vehículo y ponerse en contacto. Por otra parte, la aplicación web, al estar planteada como un prototipo inicial, el trabajo tiene la utilidad de ser tomado como un nuevo trabajo para realizar las mejoras que sean necesarias para repotenciar y poner al servicio de la sociedad en general.

Objetivos

Objetivo General

Desarrollar una Aplicación Web que permita poner en contacto personas para compartir vehículo.

Objetivos Específicos

Registrar usuarios propietarios de vehículos, usuarios clientes, rutas, horarios y características de los vehículos en una base de datos relacional.

Consultar vehículos que coincidan con rutas y horarios establecidos por los clientes.

Registrar comentarios y calificaciones de propietarios y clientes.

Evaluar el funcionamiento de la aplicación.

Metodología

Para el desarrollo y control de los procesos de la aplicación web para compartir vehículo se hace uso de la metodología ágil scrum y la arquitectura MVC (Modelo – Vista – Controlador).

Con el fin de tener una mejor organización de los procesos a realizarse durante el proceso de desarrollo y cumplir con los tiempos establecidos inicialmente para el desarrollo del proyecto se propone utilizar la metodología SCRUM debido a que se puede adaptar algunas modificaciones mínimas en cuanto a los requerimientos solicitados por el cliente, ya que en muchas ocasiones los clientes omiten ciertos detalles al momento de definir los requerimientos, y no tienen el panorama claro, esto hace que no se pueda definir claramente el alcance que tendrá el sistema. además, esta metodología permite que los miembros de equipo de desarrollo trabajen con más rapidez, con el fin de que el propietario del producto pueda visualizar los primeros resultados, para que tenga una visión general de cómo quedará el producto final.

Para poder realizar el desarrollo del sistema se propone usar la arquitectura MVC, ya que el sistema a desarrollarse necesita una arquitectura de este tipo para poder realizar la comunicación entre los componentes, con este fin se utilizará el concepto de MVC. El modelo; en donde se almacenarán los datos, la vista; con la que podrá interactuar el usuario y el controlador que será el que establezca la comunicación entre la vista y el modelo.

Capítulo 1

Marco Teórico

1. Fundamentos de programación web

Internet se ha convertido en los últimos años en parte esencial de la vida para la sociedad en general y junto con él las aplicaciones web han encontrado nuevos mercados que les han permitido expandirse alrededor de todo el mundo.

Una aplicación web consta de al menos dos partes esenciales, un ordenador que solicita ser atendido y envía peticiones es denominado "cliente", por otro lado, quien está a la espera de solicitudes y las resuelve toma el nombre de "servidor" (Luján, 2002).

De la misma forma que existen servidores dedicados al manejo de archivos y de bases de datos, una aplicación web interactúa con un servidor de aplicaciones para procesar información o realizar consultas y de esta idea básica de interacción entre clientes y servidores nace la arquitectura "cliente/servidor". Esta arquitectura otorga nuevas características y ventajas a las aplicaciones web permitiéndolas ser distribuidas, escalables, portables, concurrentes y accesibles desde cualquier lugar.

Por consiguiente, en este capítulo se va a tratar los conceptos básicos que se debe conocer, para iniciar con el desarrollo de una página web, por lo tanto, es importante conocer como aporta cada uno de los siguientes temas durante el proceso de desarrollo; servidor web, la arquitectura MVC (Modelo – Vista – Controlador), el protocolo de transferencias HTTP, dirección IP, servidor de dominios y tecnologías "Front-End y Back -End".

1.1. Arquitectura cliente-servidor

La arquitectura "cliente-servidor" pretende lograr el procesamiento de la información de un modo distribuido donde los usuarios finales pueden estar dispersos geográficamente por áreas extensas. Para (Luján, 2002) "cliente-servidor" representa una arquitectura de red en la que cada ordenador perteneciente a esta red puede actuar como un cliente y como un servidor, donde

las tareas según (Marini, 2012) son distribuidas entre diferentes proveedores o servicios denominados servidores y los demandantes toman el nombre de clientes.(Márquez Avendaño & Zulaica Rugarcía, 2004) definen las partes de esta arquitectura de la siguiente manera:

1.1.1. *Cliente*

Conocido como “front-end”, el cliente es un ordenador con una estructura básica, tanto en hardware como en software que le permite realizar funciones y despliegue de datos utilizando una interfaz gráfica de usuario (GUI).

1.1.2. *Servidor*

También nombrado “back-end”, los servidores atienden múltiples clientes con peticiones de sus recursos, además es donde se valida, da formato, procesa y alojan distintos tipos de datos. La interacción entre “cliente y servidor” se presenta en la figura 1.

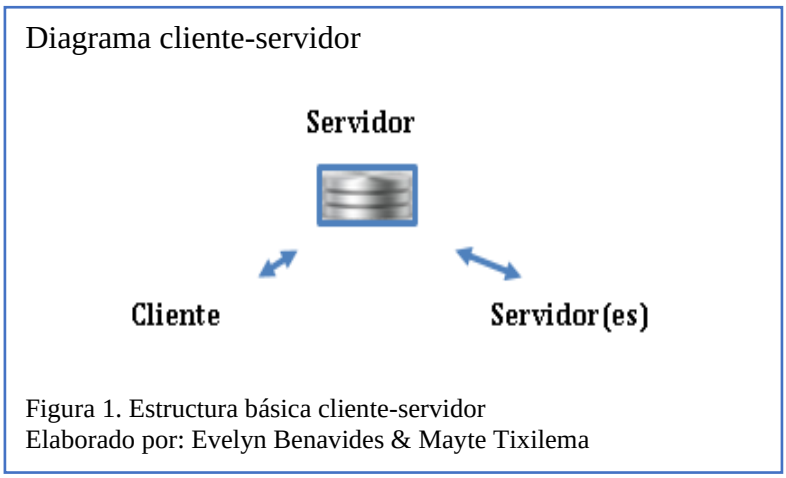


Figura 1. Estructura básica cliente-servidor
Elaborado por: Evelyn Benavides & Mayte Tixilema

Utilizando la arquitectura cliente-servidor se pueden crear aplicaciones distribuidas que (Luján, 2002) presenta con las siguientes ventajas:

- Al ser una red de ordenadores las aplicaciones distribuidas pueden ser procesadas en múltiples procesadores.
- Facilita la integración entre múltiples plataformas y entre diferentes tipos de tecnologías.
- Se logra escalabilidad horizontal (añadir o suprimir ‘clientes’) y escalabilidad vertical (migración de servidores).

1.1.3. *Características de la arquitectura cliente-servidor*

De acuerdo con (Villada Romero, 2014) la arquitectura cliente-servidor tiene las siguientes características:

- La arquitectura "cliente-servidor" se desarrolla en un ambiente heterogéneo, la tecnología de hardware y software del cliente y del servidor no son siempre las mismas.
- Los procesos del cliente y del servidor se pueden ejecutar en un mismo nodo.
- La comunicación entre cliente y servidor se limita a mensajes de petición y respuesta.
- Esta arquitectura cuenta con una capa intermedia o "middleware" entre el cliente y servidor, que provee una interfaz común en la que los clientes pueden hacer las peticiones a los servidores permitiendo una comunicación transparente.

1.2. Tecnología "Front-End"

Se encuentra en el lado del cliente, y es la tecnología que permite la comunicación de las aplicaciones web con el usuario. Para la implementación de esta tecnología, los lenguajes que se utilizan principalmente son: "HTML (Hyper Text Markup Language)", las hojas de estilo "CSS (Cascading Style Sheets), JavaScript y Bootstrap". Mediante el uso del código HTML (que se enfoca en dar un código de etiquetas sencillas de forma estructurada) y su interacción con JavaScript y CSS, se logra un resultado visual intuitivo y agradable para el usuario (Luján, 2002).

1.3. Tecnología "Back-End"

(Nieto, 2016) menciona que esta tecnología se centra en las tareas que se realizan en el servidor mediante el procesamiento de datos enviados desde el "front-end", a través de un controlador y la conexión con un motor de base de datos. El "Back End" se encarga de la parte lógica (parte no visible para el cliente) e interactúa directamente con el desarrollador, dando funcionalidad a los componentes de la interfaz gráfica de la aplicación web. Por otra parte, existen diferentes

lenguajes de programación como por ejemplo Java, PHP, etc. que permiten la comunicación con la base de datos (Brañes Vilchez, R. E. 2019).

1.4. Servidor web

El servidor web está creado con el propósito de mantenerse a la espera permanente de peticiones, acepta solicitudes desde un navegador web y esta a su vez crea una conexión utilizando el protocolo HTTP, procesa los datos para que el cliente solicitante pueda ver como respuesta una página HTML.Según (Luján, 2002) un servidor web aloja páginas estáticas y dinámicas, recursos multimedia y scripts (que se ejecutan cuando el cliente hace una solicitud y muchas veces acceden al motor de datos).

1.4.1. *Funcionamiento de un servidor web*

(Luján, 2002) explica el proceso que realiza el servidor web durante su funcionamiento, de la siguiente manera:

1. El cliente web recibe la información de la dirección de la página (URL) que ha sido enviada por el usuario.
2. El cliente realiza una petición que permitirá la comunicación con el servidor web.
3. El cliente envía una solicitud con la página que envío el usuario.
4. El servidor web devuelve una respuesta con la página solicitada en el caso de que esta exista, de lo contrario devuelve un código de error.
5. El cliente realiza la interpretación de la página HTML. Si el cliente encuentra instrucciones que no puede procesar, las solicita al servidor web.
6. Finaliza la comunicación entre cliente y servidor web.
7. El usuario puede visualizar la página solicitada.

1.4.1.1. *Protocolo HTTP (Hypertext Transfer Protocol)*

De acuerdo con (Luján, 2002) los protocolos son parte fundamental para el desarrollo de las aplicaciones web, y esto permite que la interacción entre el usuario y el ordenador sea posible.

Existe una gran familia de protocolos denominada “TCP/IP”. El protocolo HTTP no almacena ninguna información de conexiones realizadas debido a que no tiene estado y permite que la aplicación web creada y alojada en un servidor web logre interactuar con el usuario y otros sistemas. La finalidad del protocolo HTTP es alcanzar la escalabilidad y mejorar el rendimiento**.**

1.4.1.2. *Ventajas*

- Al ser un protocolo ampliable, HTTP puede enviar diferentes tipos de datos mediante una red y ser visualizados en cualquier plataforma con un navegador web.
- El protocolo HTTP comprime los campos de su encabezado utilizando el algoritmo “HPACK” para representar correcta y adecuadamente los datos, de esta manera se minimizan posibles amenazas (Peon & Ruellan, 2015).
- Los datos enviados por el protocolo HTTP son transparentes al navegar por servidor al estar apoyado en la capa de red y transporte (Mozilla Developer, 2019).

1.4.2. *DNS (Domain Name System)*

(Silva Toledo, 2018) explica que el servidor web devuelve una dirección IP, que el navegador no la puede traducir por sí solo, quien realiza este proceso es el servidor DNS, este sistema se encarga de realizar las traducciones de una dirección IP a un nombre de dominio (y viceversa). Convirtiendo una dirección IP numérica a texto entendible para el usuario. El dominio es el equivalente de una dirección física a la que un navegador web acude para la búsqueda de información específica. Generalmente el nombre de dominio está formado por el nombre de la aplicación web y la extensión del dominio, tal como .com, .edu, .ec, etc. (Perez, 2019).

El DNS utiliza el puerto 53 para intercambiar datos del protocolo “UDP y TCP”, resuelve jerárquicamente la dirección IP y entrega un nombre de dominio, además permite realizar actualizaciones de IP de manera segura y dinámica, garantizando la entrega de mensajes a los usuarios (Lopéz Padilla, 2013).

1.5. Arquitectura MVC (Modelo – Vista – Controlador)

De acuerdo con (Díaz González & Fernández Romero, 2012) MVC es un patrón de diseño arquitectónico que permite la implementación de una aplicación de manera sencilla y en el menor tiempo posible mediante la separación de sus partes llamadas modelo, vista y controlador, esto permite una mayor organización, actualización y mantenimiento del software, puede ser aplicable a cualquier framework para desarrollo de aplicaciones web, un ejemplo es Laravel, que está escrita bajo el código PHP.El Modelo, Vista, Controlador (MVC) hace referencia a la manera como estructuramos una aplicación web, por ejemplo, tenemos la parte de la ventana que puede ser definida como "vista", para el funcionamiento de esta ventana se necesita codificar al que se le denomina "controlador", por otro lado, cuando el usuario ingresa a una aplicación web (ventana) y registra alguna información necesita ser almacenada en algún espacio, el cual se define como "modelo".

1.5.1. *Modelo*

(Berzal, Cortijo, & Cubero, 2005) definen al modelo como el componente que permite encapsular los datos que forman parte de una aplicación, estos datos se almacenan independientemente de cómo se vayan a mostrar al usuario. Trabaja con las bases de datos lógicas relacionales entre ellas MYSQL, que es un lenguaje de consultas estructurados, que define los datos para luego poder utilizarlos. Para (Díaz González & Fernández Romero, 2012) el modelo maneja todos los datos de la aplicación y controla todo su proceso, sin necesidad de tener referencia de los controladores o vistas que forman parte de la arquitectura MVC.

1.5.2. *Vista*

Según (Díaz González & Fernández Romero, 2012) la vista es la capa externa o interfaz gráfica de usuario que se encarga únicamente de generar una representación visual de los datos que han

sido procesados previamente en el modelo. El modelo interactúa la mayor parte del proceso con el controlador, aunque hay excepciones en las que la vista hace una referencia directa hacia el modelo.

1.5.3. *Controlador*

(Berzal, Cortijo, & Cubero, 2005) mencionan que el controlador es la parte intermedia que se comunica directamente con el modelo o con la vista para realizar las operaciones que solicita el usuario. Cuando existen cambios en la vista o en la información que se encuentra en el modelo, el controlador realiza acciones para comunicarse con los mismos. El controlador forma parte del servidor para recibir peticiones HTTP mediante los métodos "GET, PUT, POST y DELETE". Por tanto, cada vez que un usuario quiera realizar una acción, envía una petición llamada GET, para actualizar envía una petición HTTP llamada PUT, para guardar los datos envía una petición HTTP llamada POST, para eliminar enviamos la petición llamada DELETE (Luján, 2002).

1.5.4. *Ciclo de vida de MVC*

Diagrama del Modelo Vista Controlador

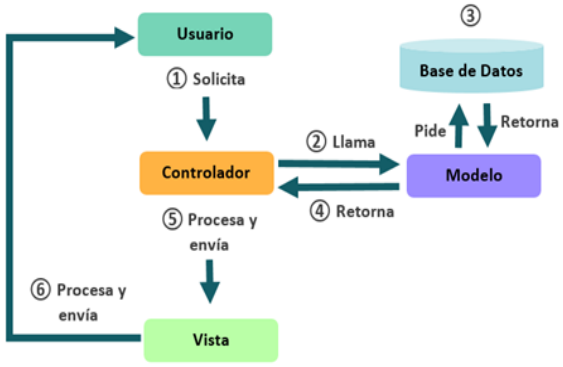


Figura 1. Ciclo de vida de la arquitectura MVC
Fuente: (Gómez, 2015)

El siguiente diagrama detalla el ciclo de vida que cumple la arquitectura MVC mientras realiza la interacción con el usuario.

El usuario realiza una solicitud a la aplicación (1), la misma que se envía al controlador quien captura esta solicitud y establece quién va a realizar la tarea. El controlador realiza una llamada al modelo (2), para responder la solicitud del controlador y realizar las operaciones necesarias, el modelo realiza interacciones con la base de datos (3). El modelo envía esta información al controlador (4), el mismo que realiza el proceso interno y envía una respuesta a la vista (5). Finalmente, la vista transforma los datos recibidos en información que el usuario pueda entender. Este ciclo puede repetirse, de acuerdo con las necesidades del usuario (Rivera López, 2008).

1.6. Fundamentos de bases de datos relacionales

Según (Piñeiro Gómez, 2014) con la evolución de los sistemas computacionales y aplicaciones web surge la necesidad de utilizar mecanismos que permitan organizar, almacenar y recuperar la información; para cubrir esta necesidad nacen los sistemas denominados SGBD (Sistema Gestor de Base de Datos). Un SGBD es un software que permite la comunicación entre el usuario y las aplicaciones web que se estén utilizando.

De acuerdo con (Silberschatz, Korth, & Sudarshan, 2002) este sistema está desarrollado para almacenar datos en grandes cantidades y recuperar información interrelacionada. Estos sistemas deben garantizar la integridad de los datos si la aplicación web llegara a fallar, ya que la información es el activo principal de toda organización. En el presente apartado se va a tratar sobre los modelos de datos como parte fundamental de un buen diseño de la base de datos y una óptima búsqueda de datos ya que de este modo se evitan posibles fallos en la etapa de producción del sistema.

1.6.1. *Modelo de datos*

Para (Coronel, Morris, & Rob, 2011) modelar los datos, es un paso importante en el diseño de la base de datos, ya que sirve como un canal entre los instrumentos reales y la base de datos que

está alojada en un ordenador. (Carrillo, Ruiz, Rodríguez, & Capote, 2005) Afirman que los modelos de datos son una colección de instrumentos conceptuales que se emplean con el fin de especificar los datos, identificar la relación que existe entre sí, la semántica que está asociada y las condiciones de seguridad de los datos. Se clasifican en modelo entidad- relación y modelo relacional, a continuación, se detalla cada uno de ellos.

1.6.1.1. *Modelo Entidad-Relación*

El modelo entidad- relación para (Silberschatz, Korth, & Sudarshan, 2002) es un modelo de alto nivel, con el que se puede abstraer los datos desde la percepción real. Este modelo fue desarrollado para dar mayor facilidad durante la especificación de los esquemas y representación de la parte conceptual de una base de datos. Este modelo se conforma de tres elementos: entidades, atributos y relaciones representadas en la figura 3.

Diagrama Modelo Entidad Relación

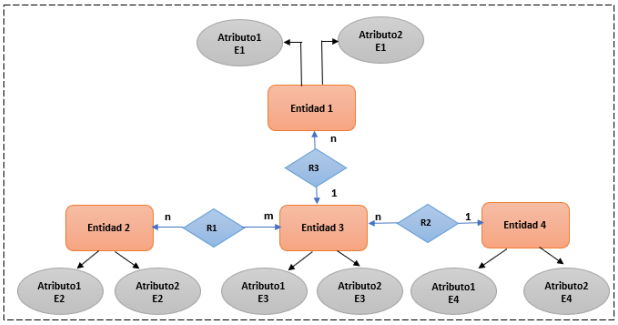


Figura 2. Representación del modelo entidad-relación
Fuente: Evelyn Benavides & Mayte Tixilema

1.6.1.2. *Modelo relacional*

El modelo relacional dicho por (Coronel, Morris, & Rob, 2011) representa a la base de datos como un conjunto de tablas relacionadas entre sí, permitiendo a los usuarios realizar distintas acciones en la base de datos como: consultar, insertar, borrar y actualizar las tuplas (también conocidas como “filas”) y campos denominados como “columnas”. Según (Silberschatz, Korth,

& Sudarshan, 2002) el modelo relacional se utiliza para organizar y gestionar datos. Está diseñando para que la información se almacene en tablas, brindando mayor comprensión para el usuario con poco conocimiento sobre la gestión de datos. Este modelo está compuesto por los siguientes elementos: tablas e identificadores, donde cada fila de la tabla contiene un valor único denominado “clave primaria” y columnas que contienen todos los atributos por los que una entidad está formada. Además, si una tabla se relaciona con otra, dentro de los atributos de una tabla debe existir el identificador de la tabla a la que se está relacionando, denominado “clave secundaria”. En la siguiente figura se puede observar una representación del modelo relacional.

Diagrama relacional

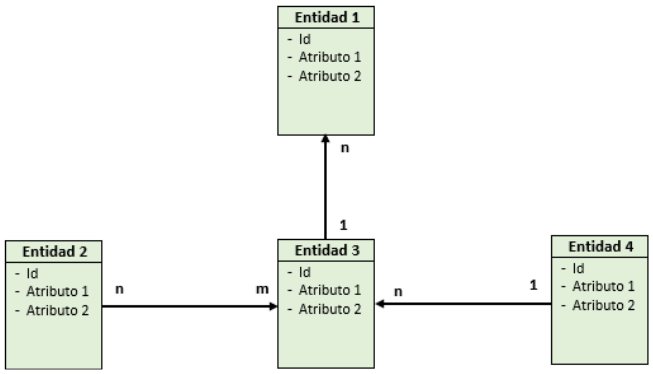


Figura 3. Representación del modelo relacional
Fuente: Evelyn Benavides & Mayte Tixilema

1.6.2. *Diseño de bases de datos relacionales*

El proceso de diseño es muy importante y se tiene que ajustar al modelo SGBD, ya que a partir de este se realizará la implementación de la base de datos. Un buen diseño de base de datos según (Microsoft, 2016) permitirá que el espacio de almacenamiento sea óptimo para que los datos siempre estén actualizados y sean íntegros con el fin de evitar su pérdida y agilizar su procesamiento; esto ayudará a que los objetivos que requiera la base de datos se cumplan.

A continuación, se detalla el proceso que (Silberschatz, Korth, & Sudarshan, 2002) propone seguir para un buen diseño.

1. **Análisis de requisitos**

   Se refiere al proceso de obtener y analizar la información que se requiere para que el sistema cumpla con el objetivo. Es necesario documentar todo lo que se recoge en esta fase, ya que los requisitos pueden cambiar mientras se realiza el proceso de diseño.

2. **Diseño conceptual**

   Es la creación del esquema conceptual de la base de datos, con el fin de visualizar gráficamente la representación de los requisitos tomados en la fase de análisis de requisitos y de esta manera poder estructurar la información.

3. **Elección del SGBD**

   Al tener clara la estructura de la información que se va a manejar en la base de datos, podemos elegir el SGBD que mejor se adapte al proyecto que vamos a iniciar y que soporte las necesidades del negocio.

4. **Diseño lógico**

   Este esquema representa la manera en la que se estructurará la base de datos, para su elaboración es importante considerar el SGBD que se seleccionó en la fase anterior ya que este modelo se ajusta al tipo de SGBD elegido.

5. **Diseño físico**

   En esta fase se utiliza el diseño lógico realizado previamente y se crea la base de datos utilizando el lenguaje de definición de datos del SGBD, generando tablas, filas y columnas, a partir de las entidades, instancias y atributos elaborados en el diseño lógico.

1.6.3. *Búsquedas en bases de datos relacionales*

(Silberschatz, Korth, & Sudarshan, 2002) definen la base de datos como un espacio para almacenar y compartir datos a los usuarios que lo requieran mediante el lenguaje de consulta estructurada SQL. Las mismas que están compuestas por tres clausulas denominadas "select, from y where".

- La cláusula "select" es utilizada para listar los atributos que se desea visualizar de una entidad.
- La cláusula "from" es el espacio donde se coloca la lista de las entidades que se relacionan entre sí para obtener como resultado un conjunto de atributos de dichas entidades escritas.
- La cláusula "where" es el espacio donde se engloba todos los atributos de cada entidad que tiene relación entre sí, en otras palabras, se coloca la condición dando uso a los atributos de "clave primaria" de cada entidad.

1.6.3.1. *Sentencias para modificar los datos de la base de datos*

Para (Coronel, Morris, & Rob, 2011) las sentencias que los usuarios pueden utilizar para modificar los registros de una base de datos cumplen con tres acciones: añadir, eliminar y actualizar, utilizando el lenguaje estándar SQL.

Añadir

La sintaxis "INSERT INTO", se utiliza para registrar nuevos datos en una tabla mediante la siguiente sintaxis:

```
INSERT INTO nombre de la tabla (p nombres de las columnas de la tabla)
VALUES (valores de los atributos de la tabla);
```

Eliminar

Utilizar la sentencia “DELETE” para borrar los datos de una tabla. Para ello se coloca la tabla del que se quiere eliminar los registros y sí necesita eliminar un solo dato de una tabla es importante utilizar la sentencia “WHERE”, de no usarlo de borrará todos los registros.

DELETE FROM nombre de la tabla **WHERE** nombre de la columna=valor;

Actualizar

La sintaxis UPDATE, se utiliza con el fin de actualizar los datos que ya se encuentran almacenados en una tabla. También se puede actualizar los campos de manera personalizada usando la sentencia WHERE.

UPDATE nombre de la tabla **SET** nombre de la columna = valor **WHERE** condición del campo que va a ser actualizado.

1.7. Fundamentos de Scrum

Según (Sutherland, 2016) Scrum nace en los años 80, con la necesidad de crear productos que tengan un impacto alto en todo el mundo como lo tenían en ese momento las empresas Honda, Canon y Fuji-Xerox. El análisis de este estudio permitió identificar que el éxito estaba en trabajar con iteraciones (bloques de entrega temporal), entregas constantes e integración del equipo de trabajo. Para lograr el éxito con esta metodología es importante dar inicio el proyecto con la elaboración de la documentación de los procesos de Scrum.

Para (The Blockhead, 2016) Scrum es un marco de trabajo ágil para el desarrollo de proyectos con niveles de complejidad alto. Inicialmente Scrum estaba enfocado en la implementación del desarrollo de proyectos informáticos, a medida que ha evolucionado, hoy en día se lo puede aplicar en cualquier ámbito laboral debido a la flexibilidad de la metodología. Se dice que es

ágil debido a que es un marco de trabajo bastante amplio en el que se pueden emplear diversas técnicas y procesos.

De acuerdo con (Sutherland, 2016) las estrategias planteadas para utilizarlo son según el entorno de trabajo donde se lo va a emplear, pero existen tres pilares que siempre deben estar presentes, tales como: transparencia, inspección constante y adaptación.

Transparencia

Es importante definir un estándar común durante cada proceso realizado ya que esto garantiza la transparencia del proceso y una mejor claridad para todos los responsables del proyecto mediante el entendimiento del proceso para poder alcanzar el objetivo inicial.

Inspección

Es el espacio donde se verifica el avance del proyecto, con el fin de llegar al objetivo inicial planteado por los integrantes del equipo de desarrollo. La inspección de los documentos de Scrum, se deben manejar de manera constante para detectar variaciones del proyecto**.**

Adaptación

Es la inspección constante, lo que garantiza encontrar procesos que no estén aportando con los resultados esperados, con esto se puede realizar los cambios a tiempo con la finalidad de que el sistema sea adaptable. En el presente apartado se va a tratar sobre el ciclo de vida de Scrum, roles, artefactos y la planificación de los eventos.

Ciclo de vida de Scrum

Proceso del ciclo de vida de Scrum

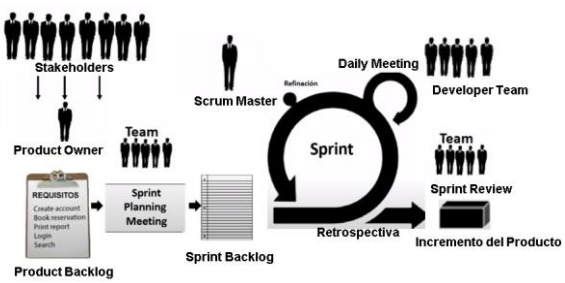


Figura 4.Ciclo de vida de Scrum
Fuente: (Benavides, 2013)

(Colla, 2012) afirma que el ciclo de vida de la metodología Scrum trabaja cómo se puede apreciar en la figura 5, Scrum inicia sus actividades a partir de que los stakeholders o clientes tienen la necesidad de automatizar sus procesos mediante un software, el mismo que se pone en contacto con un usuario (Product Owner) que es el que especifica los requisitos y este a su vez es enviado para que el equipo de desarrollo lo planifiquen y generen una lista de tareas y comiencen con el desarrollo de la funcionalidad. Se desarrolla de forma iterativa hasta lograr un incremento del producto terminado, continúa con la ejecución de todos los requisitos hasta alcanzar el final del proyecto. Para lograr el éxito del proyecto, Scrum cuenta con el apoyo de los siguientes roles: Product Owner, Scrum Master, Developer Team (Equipo de desarrollo). De acuerdo con (The Blockhead, 2016) para que los roles encajen dentro de Scrum, el Product Owner, es el usuario con el que los clientes toman contacto, para que garantice los requerimientos solicitados, el Srum Master es el responsable de velar que se cumpla de manera estricta todo el marco de referencia especificado (Colla, 2012) y el equipo de desarrollo son los responsables de elaborar el proyecto.

(Laínez Fuentes, 2015) explica que dentro del ciclo de vida también intervienen los artefactos las mismas que son: Product Backlog (lista de los requerimientos o historial del usuario que es diseñado bajo la responsabilidad del Product Owner) y Sprint Backlog (lista de las tareas en detalle, para alcanzar el objetivo.

Según (Dimes, 2015) mientras Scrum esté en ejecución, también interviene los eventos, siendo el Sprint el corazón de Scrum ya que es el que interactúa con el tiempo, el mismo que tiene una duración de una a 4 semanas; el Sprint Planning Meeting que toma como base los requerimientos de los usuarios; el Daily Meeting que son reuniones diarias de los miembros de equipo de desarrollo, con el fin de analizar las tareas pendientes; el Sprint Review es la revisión de los resultados una vez finalizado el Sprint y finalmente la Retrospectiva en el que se realiza una reunión para analizar los altos y bajos que hayan experimentado durante el Sprint.

Roles de Scrum

Para (Dimes, 2015) Scrum está conformado por cuatro roles principales, los mismos que se detallan a continuación:

Stakeholders

Son los clientes que tienen interés directo con el proyecto, los mismos que pueden ser personas naturales o empresas, que desean automatizar sus procesos, por lo que deciden buscar una empresa que se dedique a desarrollar software y establecer contacto con el Product Owner, con el fin de detallar los requerimientos del producto.

1.7.1. *Product Owner (Dueño del Producto)*

Se encarga de realizar la lista de requerimientos inicial, juntamente con el cliente para identificar las tareas más urgentes con el fin de crear una lista de sprints. Además, él es el único con autorización para manipular la lista y es el que decide si se aumentan los tiempos a algún sprint si esta tuviera mayor dificultad.

Scrum Master (Líder del equipo)

También denominado como líder del equipo, es el intermediario entre el Product Owner y los miembros del equipo de desarrollo, es importante mencionar que el Scrum Master no controla el equipo de desarrollo, al contrario, a través de él se eliminarán los obstáculos que dificulten alcanzar el objetivo inicial.

Development Team (Equipo de desarrollo)

Se encargan de construir el producto, que el cliente ha solicitado, este grupo debe contar con todas las habilidades para no tener retrasos y finalizar el sprint totalmente desplegable, con el fin de realizar ajustes antes de la entrega del producto.

1.7.2. *Artefactos de Scrum*

Según (Laínez Fuentes, 2015) los artefactos brindan mayor transparencia de la información de todo el proceso y son importantes para todos los miembros del equipo, ya que esto permite que todos puedan tener claro su rol y sus funciones a través de la inspección y adaptación. Entre los artefactos tenemos:

Proct BackLog (La pila del producto)

Es el documento que contiene la lista con los requerimientos solicitados por parte del dueño del producto, la misma que evoluciona de acuerdo a su desarrollo. También permite identificar las necesidades que tiene el producto con el fin de alcanzar su máxima utilidad y entendimiento.

Sprint BackLog (La pila del Sprint)

Este documento es generado por los miembros del equipo de desarrollo, luego de pasar un filtro denominado "producto backlog", posteriormente son establecidos como sprints independientes con el fin de priorizar los sprints que son más urgentes.

Para (Ticona Yanqui, 2018.) los artefactos deben ser transparentes, ya que a partir de ellos se tomarán decisiones futuras, por esta razón deben estar correctamente definido ya que de esta manera se garantizará la optimización del valor y el control de riegos.

1.7.3. *Eventos*

Según (Gaitán González, 2017) Scrum contiene eventos con el que se puede minimizar las reuniones informales y están sujetos a un determinado tiempo, de modo que todos tiene un principio y fin con una duración máxima de cuatro semanas. Cuando un Sprint se comienza a ejecutar no puede ser modificado y debe ser entregado en la fecha establecida. Del mismo modo, los demás eventos pueden dar por finalizado si ha cumplido con el objetivo. A continuación, se detallan los eventos que define ~~s~~crum:

Sprint

Para (Gaitán González, 2017) Un Sprint es una especie de contenedor de todos los demás eventos de Scrum que tiene un inicio y un fin y se lo conoce como el "corazón de Scrum", debe culminar en un lapso de tiempo de un mes. Durante la ejecución del sprint, no se realizarán cambios, ya que se debe cumplir con el bloque del tiempo definido para el sprint, por lo que no se pueden incluir tareas que pongan en riegos con el tempo definido, a menos que el dueño del producto lo haya solicitado.

Sprint Planning

Es una reunión que debe tardar máximo 8 horas con todos los involucrados del proyecto, principalmente el dueño del producto (Product Owner), Scrum Master y el equipo de desarrollo, con el fin de planificar cada uno de los requerimientos solicitados y armar el Sprint según las prioridades del producto.

Daily Scrum

Reuniones diarias organizadas por el Scrum Master y los miembros de equipo de desarrollo, con el propósito de ponerse al día, con tareas que se cumplieron el día anterior y que tareas las van a realizar este día, también pueden informar sobre los problemas encontrados, que están haciendo que se tarde en cumplir el objetivo. Dicha reunión debe tardar máximo de 15 minutos.

Sprint Review

Es el espacio donde se realiza una reunión con el cliente propietario del producto, con la finalidad de que los desarrolladores puedan solventar sus dudas y además que el cliente pueda constatar qué tareas ya están en función para realizar algunos ajustes si fuera el caso.

Sprint Retrospective

Reunión planificada por el Scrum Master para los desarrolladores con el objetivo de discutir cuales fueron las dificultades que enfrentaron y hacerlas como fortaleza para la elaboración del próximo Producto.

1.8. Geolocalización

Según (Beltrán, 2016) con la llegada del Internet, también se han abierto otros campos como la geolocalización, con los que muchos usuarios pueden emprender un nuevo servicio en bien de la sociedad en general. En la actualidad, la mayoría de los medios online y aplicaciones web dan uso de esta tecnología. La geolocalización tiene por objetivo ubicar geográficamente un determinado objeto o lugar que el usuario este buscando, esto a su vez es devuelto mediante representación de mapas. Además, permite que los usuarios puedan ponerse en contacto con el mundo real usando la tecnología.

Para (O'Reilly, 2006) estas son las ventajas que tiene el usuario al contar con un servicio de geolocalización.

- La geolocalización, facilita a sus usuarios obtener las rutas más cortas para ir a nuestro destino, o a su vez para saber dónde se encuentra el objeto comprado vía online.
- Con el geoposicionamiento, los usuarios están dando permiso a una app cada momento, con lo que facilita adquirir la información sobre los lugares, o rutas.
- La información alojada en la red es una fuente muy valiosa para las organizaciones que se dedican a crear aplicaciones web, para con ello mejorar las expectativas de los usuarios.
- Las búsquedas están basadas en la ubicación de los usuarios.

1.9. Economía colaborativa

De acuerdo con (Marcuello Servós, Monreal Garrido, & Díaz Fonseca , 2016) se define economía colaborativa a las actividades que permiten que haya una compensación (puede ser dinero u otros servicios) al realizar un intercambio en el consumo de bienes o servicios a través de la utilización de plataformas tecnológicas que faciliten el desarrollo del mismo; esta tendencia ha permitido que los costos se reduzcan y que su utilización incremente.

Para (Alfonso Sánchez, 2016) la economía colaborativa se basa en las siguientes actividades:

- **Consumo colaborativo:** Se retribuye económicamente por acceder a estos bienes o servicios, no por adquirirlos.
- **Producción colaborativa:** En donde grupos colaboran en la producción o distribución de este bien o servicio.
- **Aprendizaje colaborativo:** Permite el acceso a plataformas de aprendizaje en donde se comparten conocimientos.
- **Finanzas colaborativas:** Prestan servicios para inversión o préstamos siguiendo la estructura de las entidades financieras físicas.

En este modelo de negocio, las plataformas tecnológicas son fundamentales en el desarrollo de la economía colaborativa, ya que dan facilidad al usuario al realizar este proceso de intercambio; hay q tomar en cuenta que las plataformas pueden cobrar una compensación extra mínima por la prestación de un bien o servicio (Alfonso Sánchez, 2016).

1.9.1. *Ejemplos de Economía Colaborativa.*

La economía colaborativa hace que las personas puedan compartir los gastos que se genera en cualquier actividad que se esté aplicando este método.

Tabla 1. Ejemplo de Economía colaborativa

Tipo de Economía Colaborativa	Ejemplo de lugares donde se aplican
Transporte colaborativo	BlaBla Car o Uber
Vivienda colaborativa	Airbnb o HomeAway
Financiamiento colaborativo	Verkami.
Comercio colaborativo	Ebay, Wallapop o Chicfy.

Nota: Contiene todos los ejemplos donde se hace uso de la economía colaborativa

Capítulo 2

Análisis del Sistema

Tomando como base la problemática se plantea realizar un previo análisis del sistema que se va a desarrollar con el fin de recabar la información necesaria para lograr una aplicación web de fácil entendimiento y manejo para el usuario final.

El análisis del sistema pretende delimitar el alcance de nuestra aplicación, realizar el análisis de factibilidad, demostrar el flujo del sistema mediante diagramas de procesos, plantear el cronograma de hitos en base a los requerimientos, implementar la metodología ágil SCRUM para la gestión del proyecto y analizar los resultados que esperamos obtener de nuestra aplicación web.

2.1. Alcance del proyecto

La aplicación web para compartir vehículo es un prototipo que permitirá poner en contacto a un usuario conductor y cliente (pasajero) para que puedan compartir su vehículo mediante el uso de la economía colaborativa. La petición del viaje se lo realizará mediante una solicitud de parte del cliente el día anterior del que va a usar el servicio, de esta manera se podrán poner en contacto vía telefónica o mail para establecer los detalles del servicio que se brindará.

Al ser un prototipo funcional, la aplicación se encontrará disponible en un servidor web local, y no será en tiempo real. Para realizar las pruebas del sistema luego de su desarrollo con el fin de verificar la consistencia de los datos e integridad del sistema, se tomará como caso de prueba a los estudiantes, docentes y personal administrativo de la Universidad Politécnica Salesiana sede Quito, Campus Sur, debido a temas de seguridad y ubicación geográfica.

La aplicación web contará con el método tradicional de registro que será el requerimiento base para acceder a la aplicación en el que se definirá el perfil de conductor o cliente y se validará su ingreso. Se han establecido dos tipos de perfiles de usuarios con el fin de que cada usuario

cumpla con una tarea específica dentro de la aplicación web: usuario conductor y usuario cliente (pasajero).

Usuario Conductor

Es el usuario que pondrá a disposición el vehículo para realizar los viajes. Tendrá la opción de registrarse, ingresar el vehículo, crear viajes, establecer una calificación y comentarios al usuario cliente, definir puntos (de partida y llegada) que marcarán el viaje, definir puntos intermedios en su ruta, que tienen que ser lugares conocidos (parques, centros comerciales, hospitales, etc.) que servirán al usuario cliente como referencia para usar el servicio.

Usuario Cliente (Pasajero)

Es el usuario que usará el servicio brindado por el usuario conductor. Tendrá la posibilidad de registrarse, buscar viajes cercanos a su ruta (referencias que estableció el conductor previamente) mediante el registro del punto de partida y llegada, establecer una calificación y comentarios al usuario conductor, buscar los conductores que se encuentren disponibles en estos puntos y enviar una solicitud para ponerse en contacto con el conductor.

2.2. Análisis de requerimientos

El análisis de requerimientos permite entender el problema del que partimos, establecer soluciones para el mismo y realizar un análisis para que se cumplan los resultados, una vez que se haya construido la aplicación web. Este proceso permite que el sistema tenga un buen rendimiento, sea escalable, ofrezca seguridad y fiabilidad a los usuarios.

El conjunto de requerimientos establecidos a continuación definirá las funciones que la aplicación web debe cumplir para solventar las necesidades del cliente, los mismos que se clasifican de acuerdo al perfil de cada usuario.

Usuario Conductor

- Realizar el registro del usuario conductor mediante el ingreso de sus datos personales y la selección de su perfil.

- Permitir el ingreso del usuario conductor a la aplicación mediante el ingreso de su correo electrónico y contraseña.
- Registrar un vehículo con sus características y capacidad (número de asientos).
- Modificar los datos personales desde su perfil.
- Realizar el ingreso del punto de partida y punto de llegada que tendrá su ruta.
- Registrar puntos intermedios en su ruta, que sirvan de referencia al cliente.
- Aceptar o rechazar las solicitudes enviadas por el usuario cliente.
- Calificar a los clientes que ya han usado el servicio.
- Ver la calificación que tienen el usuario cliente que envía la solicitud.
- Ver un promedio de su calificación.
- Reducir el número de asientos de manera automática, cuando un cliente lo haya utilizado.

Usuario Cliente

- Realizar el registro del usuario cliente mediante el ingreso de sus datos personales y la selección de su perfil.
- Permitir el ingreso del usuario cliente a la aplicación mediante el ingreso de su correo electrónico y contraseña.
- Modificar los datos personales desde su perfil.
- Realizar el ingreso del punto de partida y punto de llegada, para conocer los puntos más cercanos.
- Ver viajes disponibles de acuerdo con el punto de partida.
- Buscar conductores disponibles para el punto seleccionado.
- Ver disponibilidad de los asientos del vehículo.
- Enviar solicitud al conductor para ponerse en contacto.
- Calificar al conductor después de utilizar el servicio.

2.3. Análisis de Factibilidad

Con el fin de logar los resultados planificados y un mayor porcentaje de éxito en este proyecto, se ha realizado el análisis de factibilidad que comprueba la existencia del conocimiento y las herramientas específicas necesarias para el desarrollo de la aplicación web. A continuación, se presenta el análisis técnico y operativo realizado a partir de los requerimientos solicitados.

2.3.1. *Factibilidad Técnica.*

El sistema se ejecutará en un servidor local y con una arquitectura cliente/servidor, realizará procesamiento de datos y utilizará geolocalización para el manejo de rutas óptimas y puntos de referencia. Para solventar estos requerimientos se realiza un análisis en cuanto a Hardware y Software.

Hardware. Antes de seleccionar un equipo de cómputo se debe analizar la función que cumplirá. Con el objetivo de trabajar sin mayor inconveniente, los ordenadores para ser considerados óptimos para desarrollo de software y aplicaciones web deben cumplir las siguientes mínimas características.

- Memoria: RAM 8 Gigabyte
- Tarjeta de video: 4 Gigabyte
- Procesador i5 de 5to generación.
- Disco Duro: 1Terabyte

Tomando en consideración los requerimientos mínimos que debe tener un equipo de cómputo, se identificó una PC con el rendimiento adecuado, para dar uso en el desarrollo de la aplicación web. El computador designado a actuar como servidor (y cliente) dispone las siguientes características:

- Memoria RAM: 8 Gigabyte.
- Procesador: Intel Core i7 7ma generación.

- Sistema Operativo: Windows 10.
- Disco Duro: capacidad de 1Terabyte.

Software. Para iniciar con el desarrollo de la aplicación, se realizó un análisis para identificar las herramientas que se van a utilizar de acuerdo a las necesidades. El lenguaje de programación PHP, debido a que es bastante flexible y cuenta con abundante documentación e información lo suficiente como para resolver cualquier problema sé que presente durante el desarrollo. Se utiliza el Framework de Laravel, ya que nos permite trabajar con el lenguaje de programación PHP para la funcionalidad del sistema, y para el diseño de la interfaz gráfica para el usuario se utiliza la plantilla Blade proporcionada por el Framewok de Laravel y la integración de las librerías de Bootstrap y JavaScript. Para el almacenamiento de la información se usa el gestor de bases de datos MYSQL debido a que se puede adaptar con gran facilidad con el lenguaje PHP.

La aplicación web va a estar sujeta a consultas en las bases de datos y geolocalización (para lo cual se utiliza el API de Google Maps), que debe ser desarrollada en corto tiempo y con pocos recursos. Para ello contamos con el siguiente conjunto de herramientas compatibles entre sí y que brinden el entorno adecuado para el desarrollo del proyecto. Para instalar estas herramientas se debe contar con los siguientes requerimientos mínimos en Hardware.

Tabla 2. Hardware Mínimo requerido

Software		Hardware mínimo requerido	Detalle
XAMPP	Apache 2.4.41	Memoria RAM:256 Megabyte.	Instalar XAMPP es la mejor opción ya que al ser de código abierto, cuenta con las herramientas base para el
	PHP 7.3.11		

	MYSQL 5.5	Disco Duro:8 Gigabyte. Procesador: Indefinido	desarrollo de la aplicación web.
PHP 5.5.9	Laravel 6.5.2	Memoria RAM:250 Megabyte. Disco Duro:40 Gigabyte.	Para contar con el Framework de Laravel se debe tener instalado PHP en la versión 5.5.9 o superior.

Nota: contiene la información del Hardware mínimo requerido para el desarrollo del software.

El estudio de la factibilidad técnica nos permitió identificar si contamos con los recursos idóneos para el desarrollo de la aplicación web. En cuanto a disponibilidad del hardware se tomó en consideración la capacidad del equipo tanto a nivel almacenamiento como en velocidad, ya que la PC seleccionada nos va a servir de apoyo para alojar todas las herramientas que se requieren. Por otra parte, las herramientas de software seleccionadas se evaluaron por la cantidad de fuentes de investigación que disponen y al nivel de conocimiento del manejo de cada una de ellas, con el fin de facilitar el proceso de la construcción del producto.

2.3.2. *Factibilidad Operacional.*

Tomando como base el análisis de los requerimientos para la aplicación web en el que se identificó que cada usuario debe cumplir una función única dentro del sistema, se considera que el usuario final deberá contar con conocimientos de nivel intermedio en cuanto al manejo de los sistemas computacionales. Ya que al finalizar el proyecto se busca que los usuarios puedan operarlo sin mayor contratiempo y con mayor facilidad. Por lo tanto, el usuario que oferte el servicio debe ser capaz de crear viajes y debe definir su propia ruta dentro del mapa. El usuario

quien hace uso del servicio debe ser capaz de identificar los viajes disponibles en base a los puntos de referencia ingresada.

Para cumplir con el objetivo este proyecto requiere de conocimientos en varias áreas y distintas herramientas, en la Tabla 3 se presenta los recursos asignados a cada proceso del desarrollo y una descripción de este.

Tabla 3. Procesos y recurso Humanos del proyecto

Recursos humanos	Cantidad	Módulo a desarrollar	Procesos	Descripción
Programador Junior	Se cuenta con 2 desarrolladores Junior, los mismos que cuentan con conocimientos de nivel intermedio sobre el manejo de las herramientas para el desarrollo de la aplicación. Así como programación	Módulo Usuario Conductor.	Crear el menú con todas las tareas que el usuario conductor debe realizar dentro del sistema, así como: registrar el vehículo, modificar el perfil, crear viajes, recibir solicitudes por el servicio, calificar al cliente, visualizar la calificación que tiene el usuario cliente.	Diseño de las pantallas y funcionalidad que sean necesarias para que el usuario conductor pueda navegar y ofrecer el servicio.
			Crear el menú con las tareas que el usuario cliente debe	Diseño de las pantallas y

	en PHP usando el Framework de Laravel, y conocimientos sobre diseño utilizando las librerías de Bootstrap y JavaScript.	Módulo Usuario Cliente.	cumplir una vez dentro de la aplicación web, así como: modificar sus datos, buscar viajes, visualizar la disponibilidad del vehículo, enviar solicitud por el servicio, calificar al usuario conductor.	funcionalidad para que el usuario cliente pueda ingresar a la aplicación y solicitar el servicio.
Consultor Externo	Se cuenta con un consultor externo con conocimientos en Geolocalización.	Se complementa en el desarrollo del Módulo Usuario Conductor.	Implementación de ruta óptima con geolocalización. Puntos de referencia con geolocalización.	Implementación funcional de rutas y puntos referenciales utilizando herramientas de geolocalización.

Nota: contiene la información de los recursos humanos con los conocimientos que se requiere para el desarrollo

El análisis de esta factibilidad nos ha permitido identificar los recursos humanos que se disponen para realizar el desarrollo de la aplicación web, los mismos que deben lograr que el sistema sea completamente funcional y escalable. Para logar con el objetivo se evaluó el nivel de conocimiento que tiene cada desarrollador en cuanto al manejo de las herramientas.

Mediante este análisis lo que se busca es evaluar si la aplicación web creada es correctamente funcional, de fácil entendimiento para el usuario final.

2.4. Diagrama del proceso

Diagrama de proceso de la aplicación web para compartir vehiculo

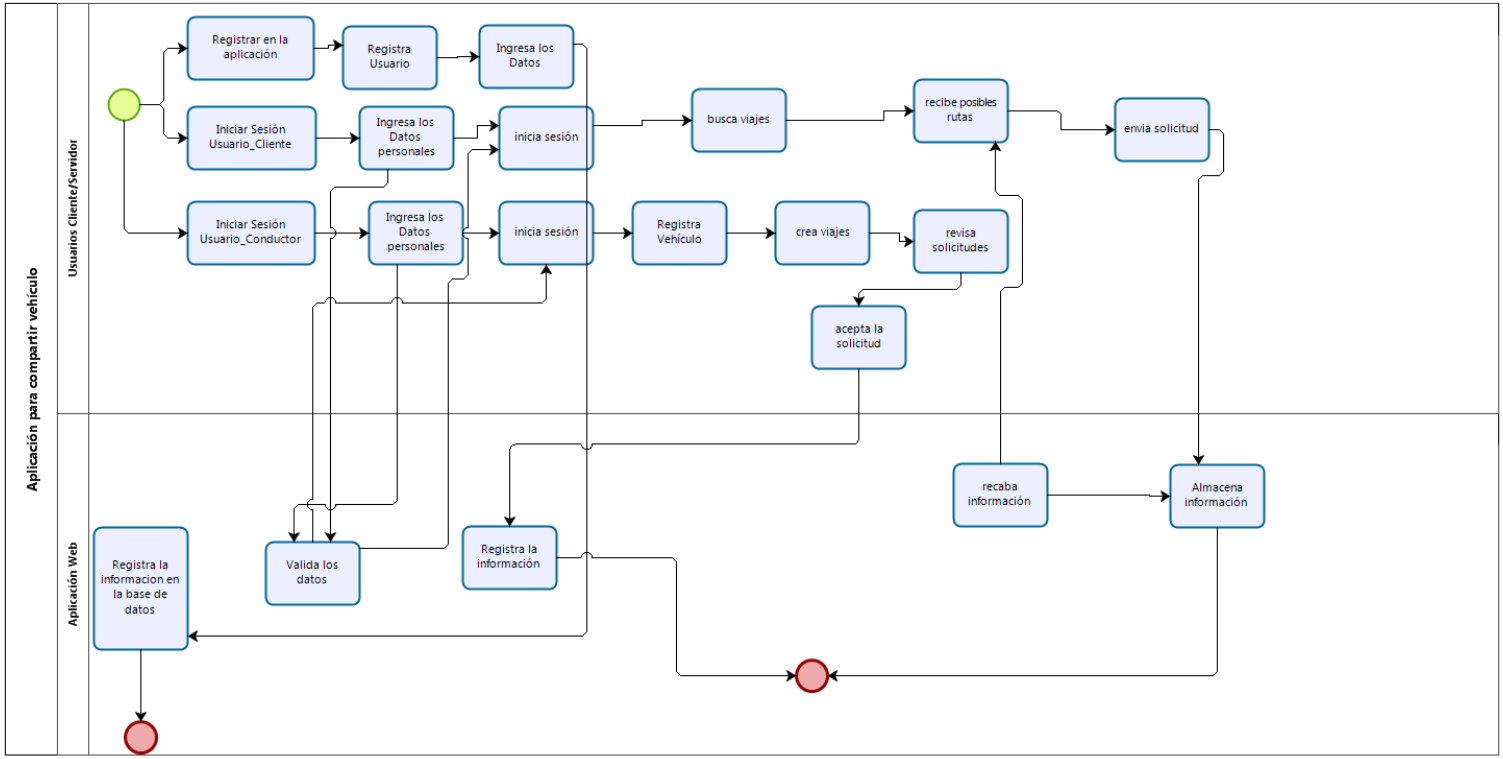


Figura 2. Diagrama de Procesos de la aplicación web
Fuente: Evelyn Benavides & Mayte Tixilema

2.5. Implementación de la Metodología.

Para administrar el proceso de desarrollo de la aplicación web se utilizó la metodología ágil SCRUM, se optó por dar uso de esta metodología debido a que se puede adaptar algunas modificaciones mínimas en cuanto a los requerimientos solicitados por el cliente, ya que en muchas ocasiones los clientes omiten ciertos detalles al momento de definir los requerimientos, ya que muchas veces no tienen el panorama claro y esto hace que no se pueda definir claramente el alcance que tendrá el sistema. Además, esta metodología permite que los miembros de equipo de desarrollo trabajen con más rapidez, con el fin de que el propietario del producto pueda visualizar los primeros resultados, para que tenga una visión general de cómo quedará el producto final. Rescatando todas las ventajas que se identificó se puede decir que la Metodología es bastante flexible y permite con gran facilidad adaptar a las necesidades que el cliente tiene sobre el proyecto.

2.6. Definición de los roles.

Dentro del proyecto denominado aplicación web para compartir vehículo, existe un grupo de personas que son los principales interesados que el proyecto tenga éxito, por lo tanto, se realizó un análisis del rol que cumplirá cada involucrado durante el proceso de desarrollo del producto. En la Tabla 4 se detalla el rol que desempeñará y el cargo que tiene cada participante dentro del proyecto.

Tabla 4. Definición de rol de los involucrados del proyecto

Nombres	Rol	Organización y Cargo	Detalle
Evelyn Benavides	Miembro de equipo de desarrollo.	UPS/Estudiante	Son los encargados de levantar todos los requerimientos que se necesita para el desarrollo del

Mayte Tixilema	Miembro de equipo de desarrollo.	UPS/Estudiante	producto. De igual manera son las que desarrollan el sistema. Al igual que la documentación del proceso de la creación de la aplicación web.
Ing. Julio Proaño	Scrum Master, Principal interesado.	UPS/Tutor	Es el que se encarga de revisar todos los entregables con el fin de probar.

Nota: Detalle del cargo que ocupa los involucrados durante la etapa de desarrollo.

Como en todo proyecto de desarrollo que hacen uso de la metodología ágil Scrum, definir la responsabilidad a cada miembro del equipo ayuda a que el proyecto avance, ya que cada uno debe presentar su trabajo en el tiempo establecido, de ese modo se evitará el retraso de la entrega.

2.7. Cronograma del Proyecto (Hitos).

Mediante el correcto planteamiento del cronograma de hitos se puede identificar todos los eventos que está ocurriendo en el desarrollo del proyecto, ya que de esta manera se puede tener mayor control y trabajar con transparencia. Además, permite la gestión optima de los recursos humanos que se dispone y tener plenamente identificados cuales deben ser los entregables al finalizar el proyecto. A continuación, se plantea el cronograma de hitos propuesto para el desarrollo de la ampliación web, el mismo que cuenta con fechas tentativas las mismas que estas sujetas a cambios.

Tabla 5. Cronograma de Hitos

CRONOGRAMA DE HITOS			
#	ENTREGABLE/SITUACIÓN	FECHA	OBSERVACIÓN
1	Análisis de requerimientos	4/Noviembre/2019 – 15/Noviembre/2019	Reunión presencial para establecer los requerimientos iniciales del sistema
2	Etapa de diseño del sistema. Diagramas que cubran los requerimientos establecidos en la etapa de análisis de requerimientos.	18/Noviembre/2019 – 30/Noviembre/2019	Diagramas de diseño del sistema
3	Desarrollo de la aplicación web	2/Diciembre/2019- 4/Mayo/2020	La entrega de avances se realizará de acuerdo con el cronograma de desarrollo, elaborado mediante el diagrama de Gant.
4	Entrega del prototipo funcional	19/Junio/2020	Demostración de la funcionalidad de la aplicación
5	Pruebas del sistema	25/Junio/2020	
6	Primer borrador del proyecto técnico.	10/Julio /2020	Formato digital

Nota: Cronograma con la información de las fechas de todos los entregables

2.8. Plan de trabajo (Sprints)

Previa una reunión con el Scrum Master y los miembros del equipo de desarrollo, se analiza cómo se va a estructurar el sistema, para lo cual se toma como base el análisis de requerimientos con el que se puede definir que el sistema se estructure bajo la segmentación de 4 módulos. En la figura 7 se detalla la segmentación de la aplicación web por módulos.

Segmentación del sistema por módulos

Modo de	Nombre de tarea	Duración	Comienzo	Fin	Predecesoras	Nombres de los recursos
	Aplicación para compartir vehiculo	**111 días**	**lun 2/12/19**	**lun 4/5/20**		
	Diseño de Bases de Datos	**5 días**	**lun 2/12/19**	**vie 6/12/19**		
	Modulo de Seguridad	**28 días**	**lun 9/12/19**	**mié 15/1/20**	**2**	
	Modulo Usuario Conductor	**42 días**	**jue 16/1/20**	**vie 13/3/20**	**6**	
	Modulo Usuario Cliente	**36 días**	**lun 16/3/20**	**lun 4/5/20**	**14**	

Figura 3. Segmentación del sistema
Fuente: Evelyn Benavides & Mayte Tixilema

Sprint 1. Mediante una reunión de inicio de Sprint con el Scrum Master y los miembros del equipo de desarrollo, se analiza todas las posibles variables con el fin de definir que variables serán denominados nombre de las entidades y cuáles serán los atributos. Para diseñar el modelo Entidad Relación y el Modelo Relacional a fin de generar un script.

Desglose del diseño de bases de datos

Modo de	Nombre de tarea	Duración	Comienzo	Fin	Predecesoras
	Aplicación para compartir vehiculo	**111 días**	**lun 2/12/19**	**lun 4/5/20**	
	Diseño de Bases de Datos	**5 días**	**lun 2/12/19**	**vie 6/12/19**	
	Diseño de Bases de Datos Modelo Entidad Relación	2 días	lun 2/12/19	mar 3/12/19	
	Diseño de la base de datos Modelo Relacional	2 días	mié 4/12/19	jue 5/12/19	3
	Generar el script de la base de datos.	1 día	vie 6/12/19	vie 6/12/19	4

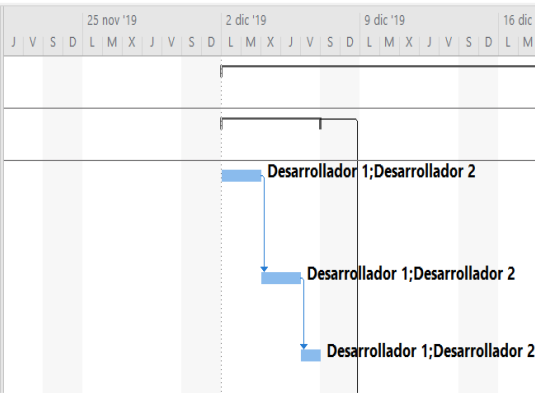
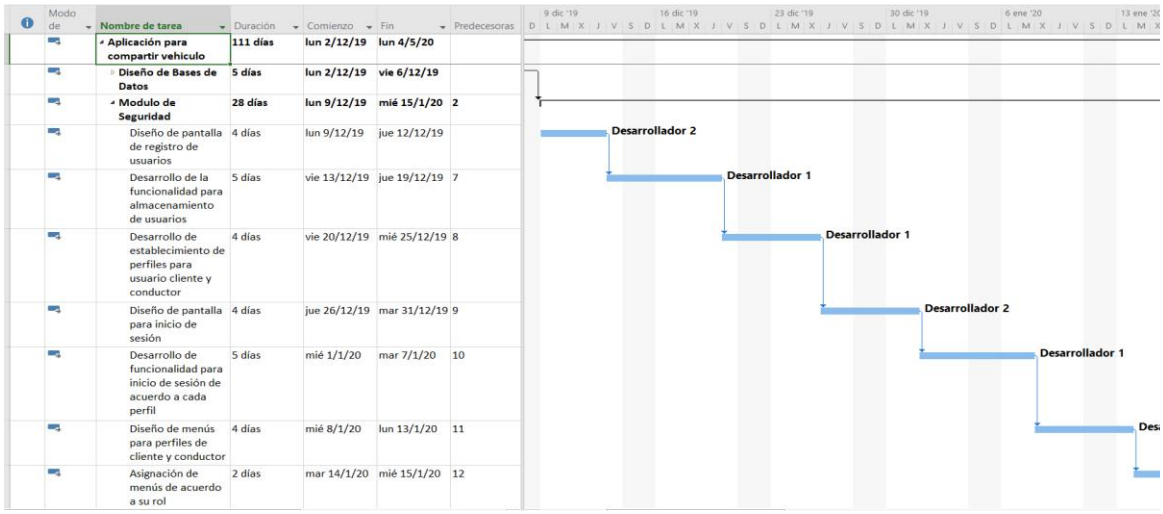


Figura 4.Modulo de Diseño de la base de datos
Fuente: Evelyn Benavides & Mayte Tixilema

Sprint 2. En la reunión que se llevó acabo para definir el Sprint 1, se planificó iniciar con el desarrollo del módulo de Seguridad. Ya que son las pantallas base para definir que tereas debe cumplir cada usuario una vez dentro de la aplicación web. En la figura 9 se desglosa todas las tareas, el número de días que llevará en implantar cada uno de ellos y se define las fechas de inicio y fin de cada actividad.

Desglose del modulo de seguridad



Modo de	Nombre de tarea	Duración	Comienzo	Fin	Predecesoras
	Aplicación para compartir vehiculo	**111 días**	**lun 2/12/19**	**lun 4/5/20**	
	Diseño de Bases de Datos	**5 días**	**lun 2/12/19**	**vie 6/12/19**	
	Modulo de Seguridad	**28 días**	**lun 9/12/19**	**mié 15/1/20**	**2**
	Diseño de pantalla de registro de usuarios	4 días	lun 9/12/19	jue 12/12/19	
	Desarrollo de la funcionalidad para almacenamiento de usuarios	5 días	vie 13/12/19	jue 19/12/19	7
	Desarrollo de establecimiento de perfiles para usuario cliente y conductor	4 días	vie 20/12/19	mié 25/12/19	8
	Diseño de pantalla para inicio de sesión	4 días	jue 26/12/19	mar 31/12/19	9
	Desarrollo de funcionalidad para inicio de sesión de acuerdo a cada perfil	5 días	mié 1/1/20	mar 7/1/20	10
	Diseño de menús para perfiles de cliente y conductor	4 días	mié 8/1/20	lun 13/1/20	11
	Asignación de menús de acuerdo a su rol	2 días	mar 14/1/20	mié 15/1/20	12

Figura 5.Módulo de Seguridad
Fuente: Evelyn Benavides & Mayte Tixilema

Sprint 3

En la reunión planificado para la gestión del Sprint 2, se define que el siguiente modulo a desarrollarse es el módulo usuario conductor, debido a que la implementación de la geolocalización dentro del producto tiene un alto nivel de complejidad. Por lo tanto, en la figura 10 de desglosa todas las tareas y el número de días que llevara cada actividad.

Desglose del módulo del usuario coductor

Modo de	Nombre de tarea	Duración	Comienzo	Fin	Pred
	Modulo Usuario Conductor	**42 días**	**jue 16/1/20**	**vie 13/3/20**	**6**
	Diseño de las pantallas para las tareas del usuario conductor	5 días	jue 16/1/20	mié 22/1/20	
	Desarrollo de la funcionalidad de modificación de información personal del usuario	4 días	jue 23/1/20	mar 28/1/20	15
	Desarrollo de la funcionalidad de registro de vehículo.	3 días	mié 29/1/20	vie 31/1/20	16
	Lista de información del vehículo registrado	2 días	lun 3/2/20	mar 4/2/20	17
	Desarrollo de la funcionalidad de creación de viajes.	5 días	mié 5/2/20	mar 11/2/20	18
	Implementación del API de geolocalización para la elaboración de rutas	10 días	mié 12/2/20	mar 25/2/20	19
	Desarrollo de la funcionalidad de la pestaña de solicitudes del servicio del usuario cliente.	5 días	mié 26/2/20	mar 3/3/20	20
	Desarrollo de la funcionalidad para que el usuario califique al usuario cliente.	6 días	mié 4/3/20	mié 11/3/20	21
	Visualización del promedio de la calificación del usuario conductor en la pantalla principal.	2 días	jue 12/3/20	vie 13/3/20	22

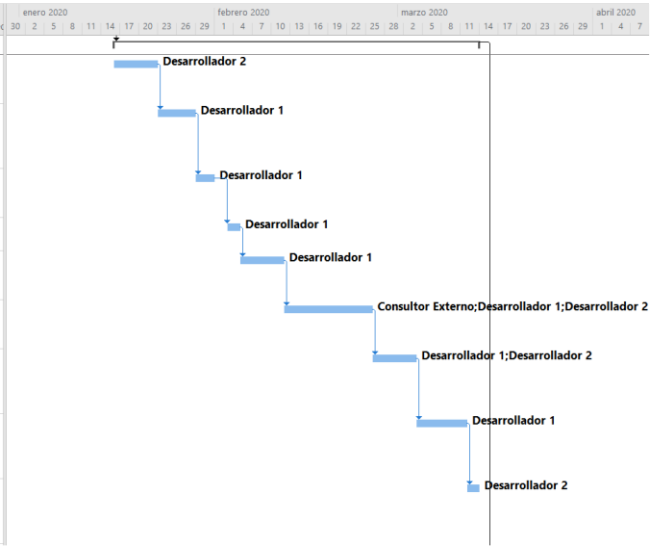


Figura 6.Modulo usuario conductor
Fuente: Evelyn Benavides & Mayte Tixilema

Sprint 4

En la reunión planificada para analizar los avances del Sprint 3, se define el Módulo usuario cliente y la función que debe cumplir el usuario dentro de la aplicación. Por lo tanto, en la figura 11 se detalla todas las tareas y el número de horas a invertir para su desarrollo al igual que la fecha de inicio y fin de cada actividad.

Desglose del módulo del usuario cliente

Modo de	Nombre de tarea	Duración	Comienzo	Fin	Pred
	Modulo Usuario Cliente	**36 días**	**lun 16/3/20**	**lun 4/5/20**	**14**
	Diseño de las pantallas para las tareas del usuario cliente	5 días	lun 16/3/20	vie 20/3/20	
	Desarrollo de la funcionalidad de modificación de información personal del usuario	4 días	lun 23/3/20	jue 26/3/20	25
	Desarrollo de la funcionalidad de la pestaña buscar viajes.	10 días	vie 27/3/20	jue 9/4/20	26
	Desarrollo de la funcionalidad de la pestaña para que el usuario pueda visualizar la disponibilidad de asientos del vehículo.	6 días	vie 10/4/20	vie 17/4/20	27
	Desarrollo de la funcionalidad de la pestaña envió de solicitudes por el servicio.	5 días	lun 20/4/20	vie 24/4/20	28
	Desarrollo de la funcionalidad de la pestaña para que el usuario califique al usuario conductor.	6 días	lun 27/4/20	lun 4/5/20	29

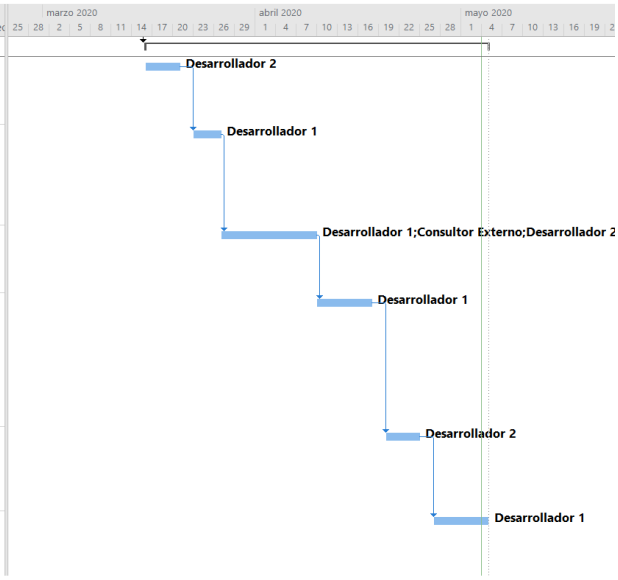


Figura 7.Modulo usuario cliente
Fuente: Evelyn Benavides & Mayte Tixilema

2.9. Resultados esperados.

Al finalizar los sprints el producto final se espera que sea capaz de poner en contacto a los usuarios clientes con los usuarios que ofrecen el servicio en vehículo compartido. De igual forma la aplicación web debe permitir buscar y crear rutas.

Capítulo 3

Diseño del Sistema

La finalidad del diseño del sistema es la organización de los procesos y lograr un mejor entendimiento para su desarrollo.

Para el diseño de la aplicación web, se ha optado por dar uso del modelado de la arquitectura de tres capas denominados como: presentación, aplicación o modelo de negocio y persistencia o datos, a fin de obtener una estructura de fácil entendimiento y completamente escalable que pueda crecer en versiones futuras. Con el diseño del sistema se busca identificar todas las tareas y procesos con los que bebe contar la aplicación web con el fin de encontrar el camino correcto al momento de desarrollar el sistema.

El presente capítulo contiene la especificación de; los casos de uso para los usuarios que utilizarán el sistema, diagrama de clases, el diagrama de secuencias, diseño de los modelos de bases de datos (Modelo Entidad-Relación, Modelo Relacional), detalle del modelo de la arquitectura utilizada y el diseño de las ventanas que permitirán la interacción de los usuarios con el sistema.

3.1. Especificación de los casos de uso.

Con el diseño de los diagramas de casos de uso se pretende identificar el comportamiento y la comunicación de la aplicación con los usuarios mediante la evaluación de cada una de sus interacciones. Para identificar las acciones y funciones que se realizarán se procede a realizar el diagrama dando uso a los actores (Usuario Conductor y Usuario Cliente) y los casos de uso los cuales contienen las funciones que brindará la aplicación al usuario final. A continuación, en las figuras 1 y 2 se puede observar los diagramas diseñados para los usuarios de la aplicación web para compartir vehículo entre conductor y cliente (pasajero).

3.1.1. Caso de uso relacionado con el usuario conductor

Caso del usuario conductor

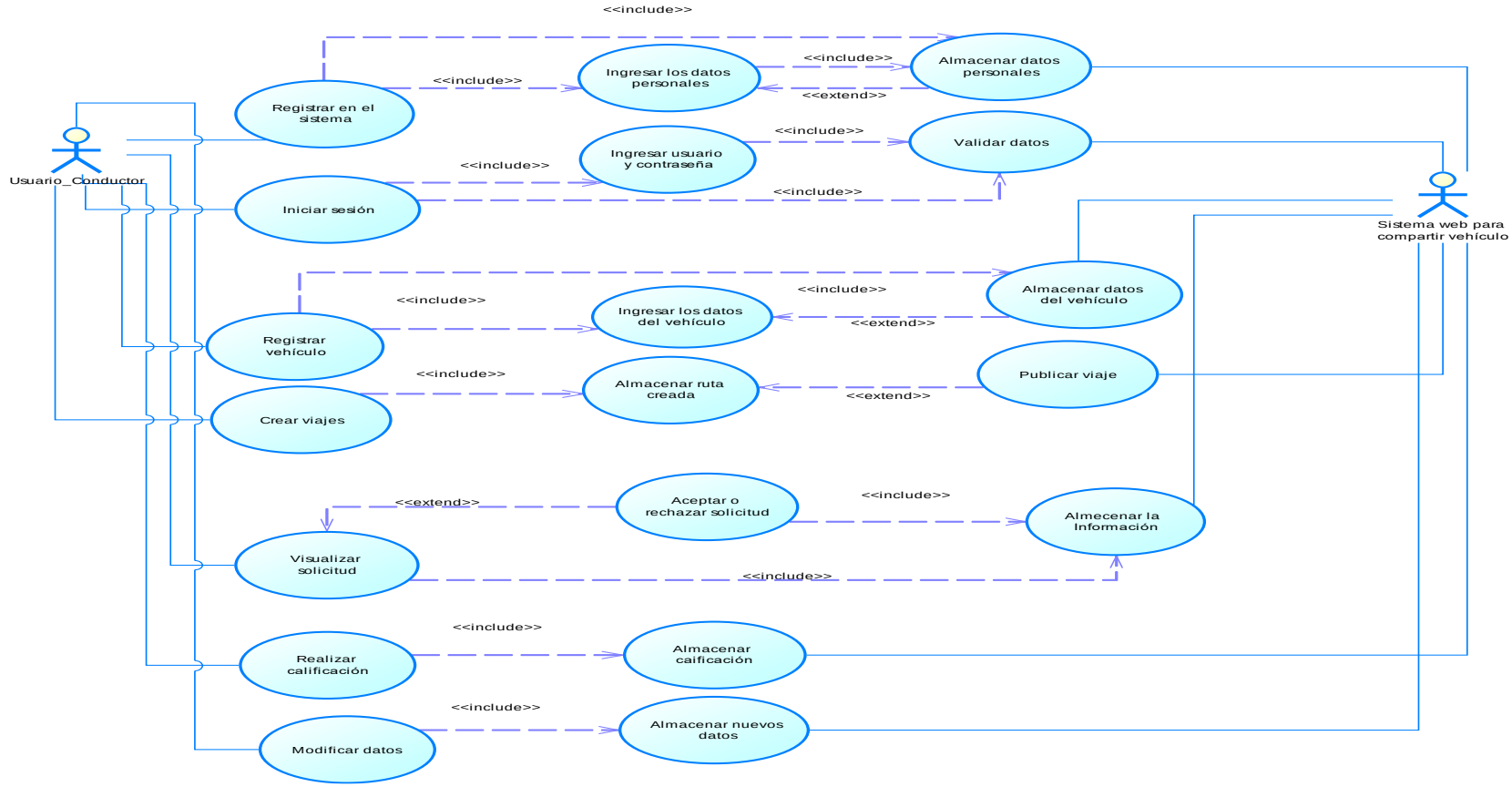


Figura 8. Caso de uso usuario conductor
Fuente: Evelyn Benavides & Mayte Tixilema

3.1.2. Caso de uso relacionado con el usuario cliente (pasajero)

Diagrama de caso de uso para el usuario pasajero

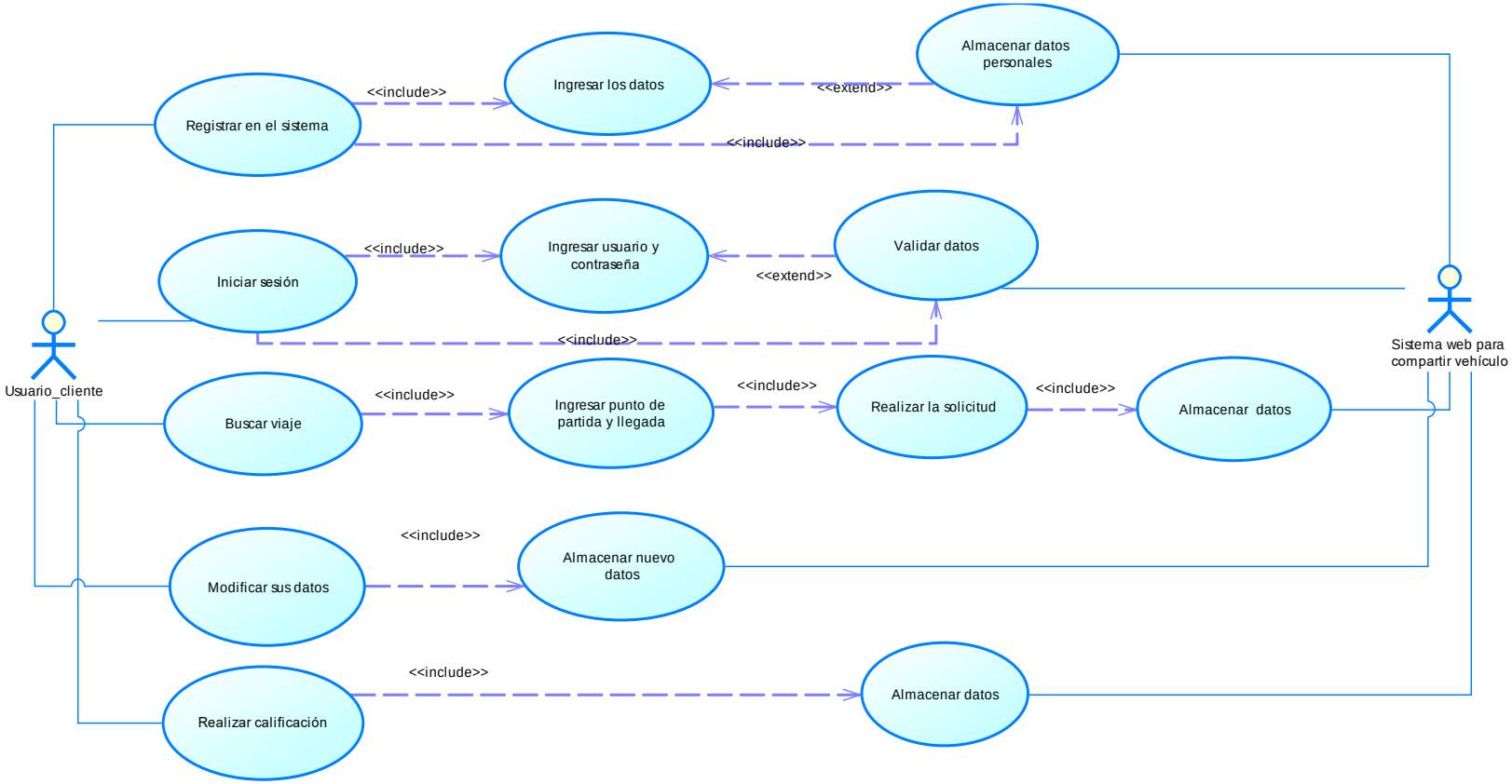


Figura 9. Caso de uso usuario cliente
Fuente: Evelyn Benavides & Mayte Tixilema

Mediante la especificación de cada una de las actividades plasmadas en los casos de uso para los usuario conductor y cliente (pasajero), lo que se espera es visualizar qué procesos necesitan cada uno de los usuarios mediante una precondición para que el sistema funcione correctamente. El detalle de actividades para cada usuario se lo puede visualizar en el Anexo 1.

3.2. Diseño de diagrama de clases.

El diseño del diagrama de clases permite formar una estructura clara del sistema mediante el modelamiento de las clases, atributos y relaciones para que de esta manera se logre un mayor entendimiento e identificación de los posibles objetos que van a formar parte de la infraestructura interna del sistema.

Dentro de cada diagrama de clases entran en acción diferentes interacciones entre las clases y los objetos que forman el mismo, esto ayudará a tener claras las necesidades del sistema, ya que se tendrá de forma visual los esquemas de la aplicación.

A continuación, se muestra el diagrama de clases modelado para el diseño de la aplicación web, en donde la figura 14 representa la estructura que se usará para el manejo de inicio de sesión y la figura 15 la estructura a la que tendrán acceso el conductor y usuario (pasajero).

3.2.1. *Diseño de diagrama de clases*

Diagrama de clase (módulo de seguridad)

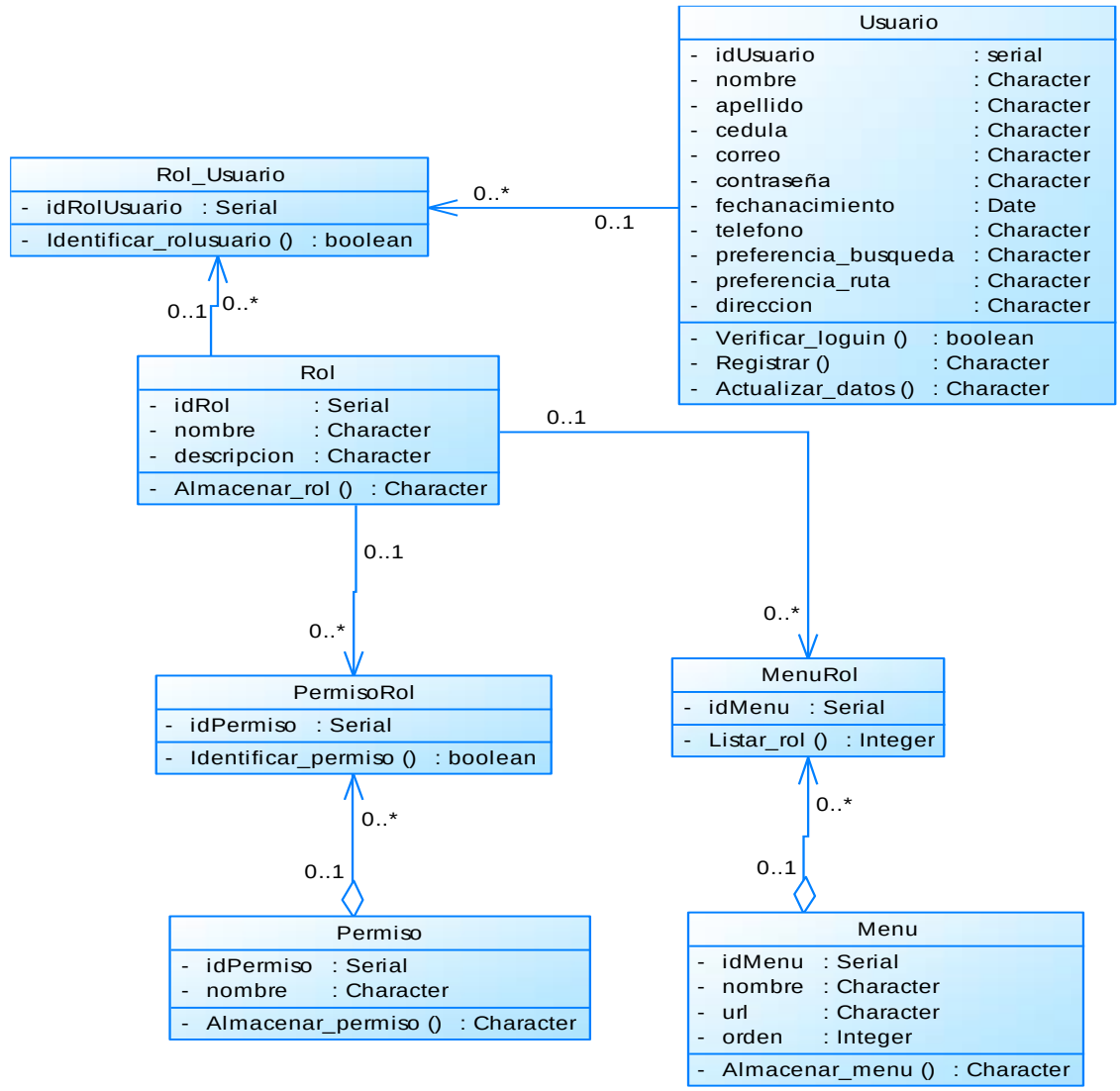


Figura 10. Diagrama de Clase Módulo Seguridad
Fuente: Evelyn Benavides & Mayte Tixilema

Diagrama de clase (estructura usuario y conductor)

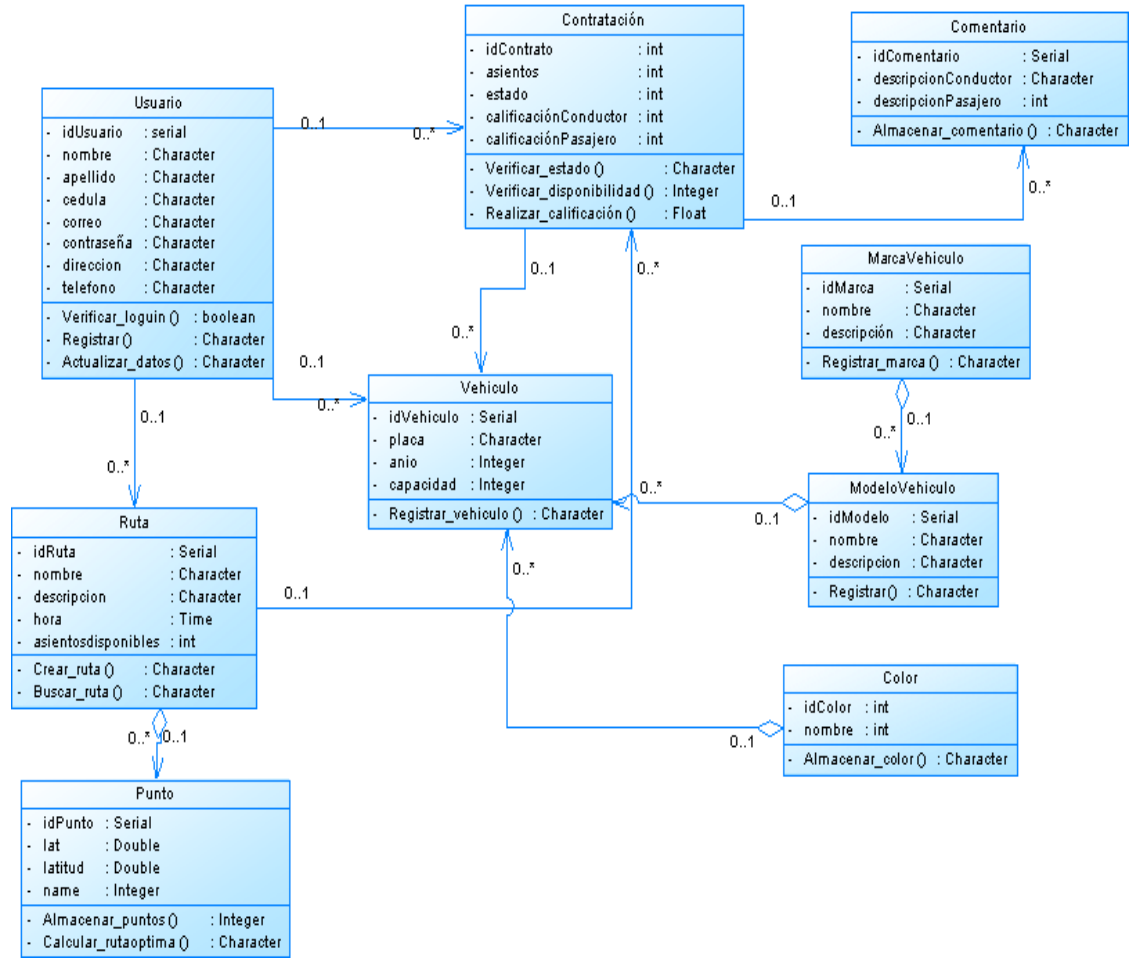


Figura 11. Diagrama de Clase de la Aplicación Web
Fuente: Evelyn Benavides & Mayte Tixilema

La especificación del diseño de los diagramas de clases que conforma la aplicación web se encuentra en el **Anexo 2** el mismo que se realizó a partir del diagrama diseñado anteriormente, en donde se especifican cada una de sus clases con sus respectivos atributos, operación de la clase (métodos) y relaciones.

3.3. Diseño del diagrama de secuencias.

Con el diseño de los diagramas de secuencia se busca identificar el comportamiento de la aplicación web mediante el intercambio de mensajes con todos los objetos que interactúan con un determinado actor, dado que con este tipo de diagrama se puede identificar el tiempo que invierte una actividad u objeto en pasar de una a otra, nos permite visualizar el tiempo promedio

empleado y tener claramente identificados las acciones que debe realizar el usuario final. Para realizar el diseño de los diagramas de secuencias, se tomó como base los diagramas de casos de uso con el fin de identificar los procesos de cada actividad que el usuario realiza dentro del sistema y ambas partes del diagrama de clases a fin de identificar todos los objetos del sistema. Con esto se desea tener bien identificadas las tareas que el usuario debe realizar para que el sistema ejecute los procesos que han sido automatizados.

3.3.1. Diagrama de secuencia para el usuario Conductor

Diagrama de secuencias del usuario conductor

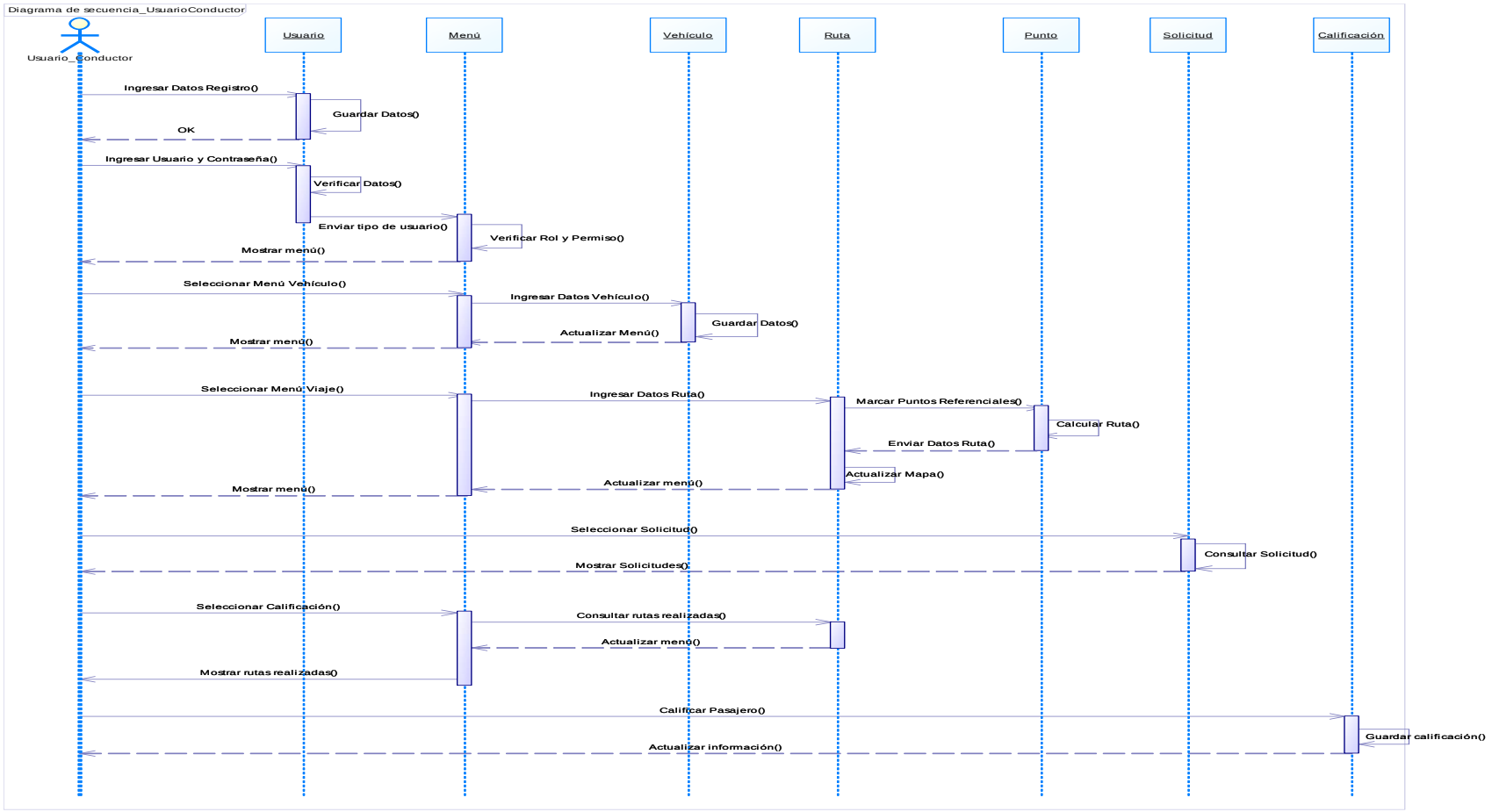


Figura 12. Diagrama de secuencias Usuario Conductor
Fuente: Evelyn Benavides & Mayte Tixilema

3.3.2. Diagrama de secuencia para el usuario Cliente

Diagrama de secuencias del usuario cliente

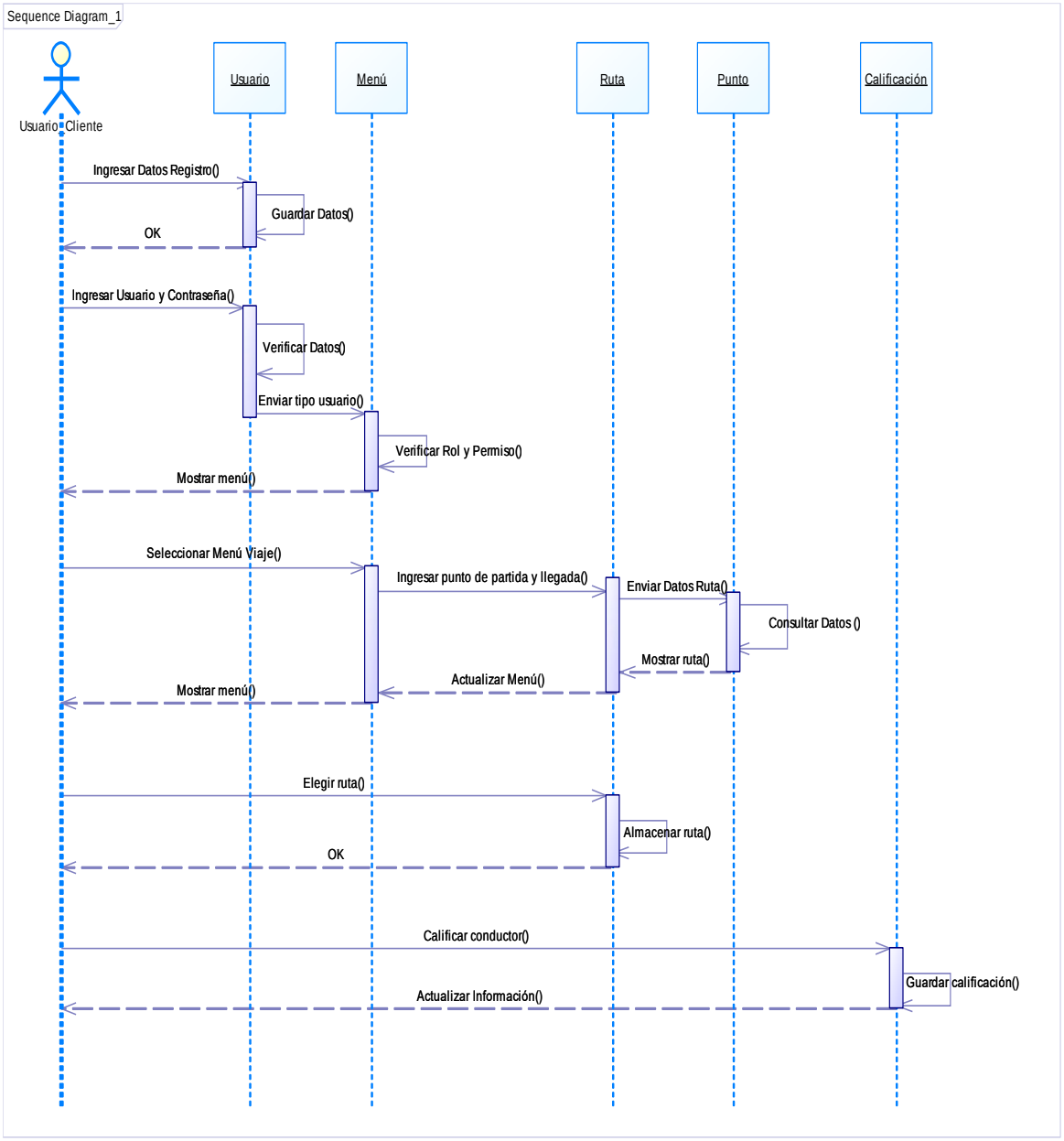


Figura 13. Diagrama de secuencias Usuario Cliente
Fuente: Evelyn Benavides & Mayte Tixilema

El diagrama de secuencias para el usuario conductor cuenta con 7 objetos (rol de la clase) como se puede observar en la figura 16 y el diagrama de secuencias para el usuario cliente cuenta con 5 objetos (figura 17), con los que se puede describir el comportamiento de cada uno de ellos dentro del sistema. La especificación de cada objeto que conforman los diagramas se los encontrará en el **Anexo 3.**

3.4. Diseño de la base de datos.

El diseño de la base de datos es fundamental, ya que a partir de él se establecerá la estructura de almacenamiento en la base de datos. La base de datos es el espacio donde se almacenará toda la información que el usuario proporcione cuando esté dando uso del sistema, es el corazón de cualquier sistema informático debido a que una aplicación web o sistema informático extrae todos los datos a partir del sistema gestor de base de datos al que se encuentre conectado. Por lo tanto, contar con una base de datos que internamente se encuentre funcionando correctamente ayuda a que los usuarios puedan encontrar la información que están buscando de manera ágil y con menos probabilidad de perdida de información. Mientras más información tenga la base, el sistema tiene más probabilidades de tener éxito y brindar facilidad, rapidez e información actualizada. Para el diseño de una buena base de datos se debe tomar en cuenta la parte de la abstracción lógica y conceptual de la información mediante la representación en diagramas de Modelo Entidad Relación y el Modelo Relacional, así como también un análisis sobre los objetos y atributos identificados en los diagramas de clases.

3.4.1. Modelo Entidad -Relación

Diagrama modelo entidad- relación

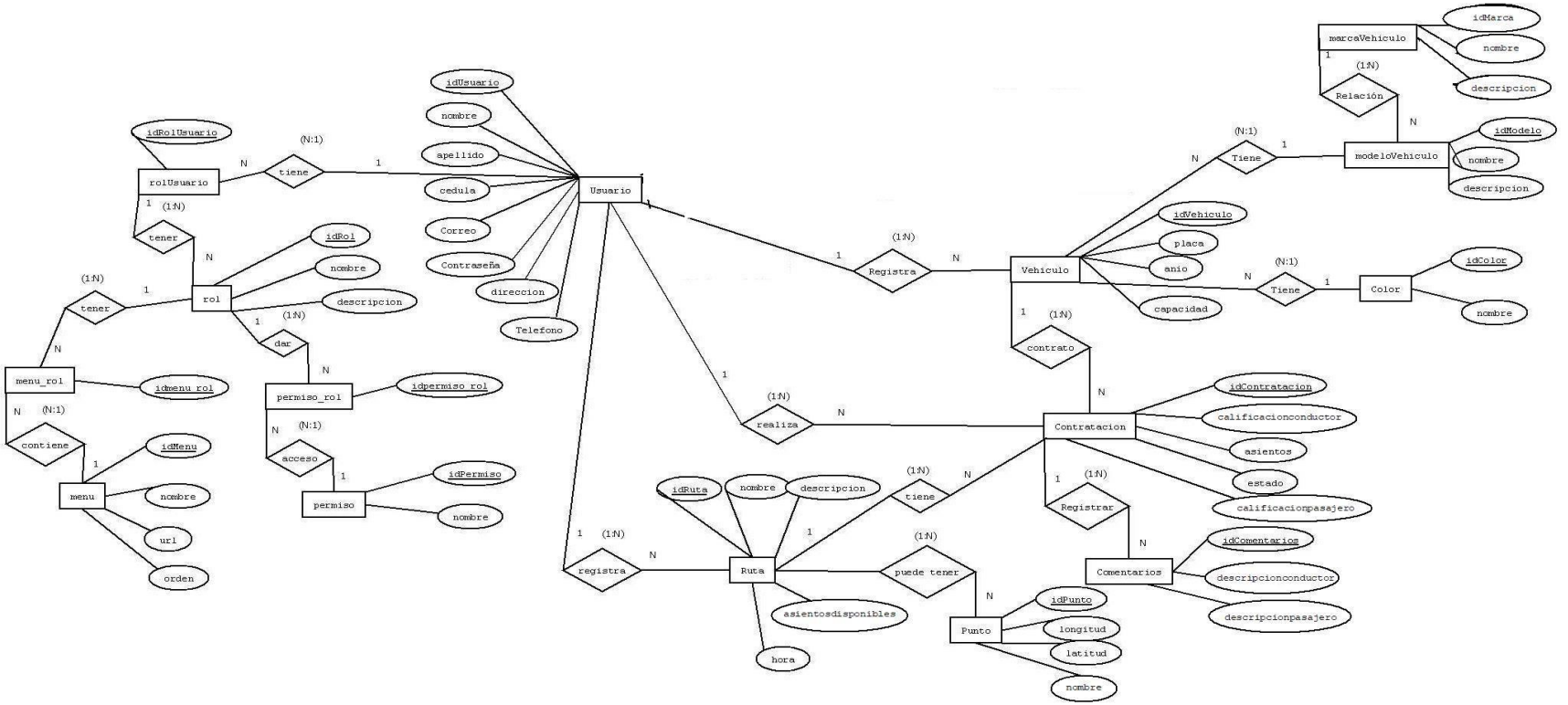


Figura 14. Diagrama Entidad-Relación
Fuente: Evelyn Benavides & Mayte Tixilema

3.4.2. Modelo Relacional

Diagrama modelo relacional

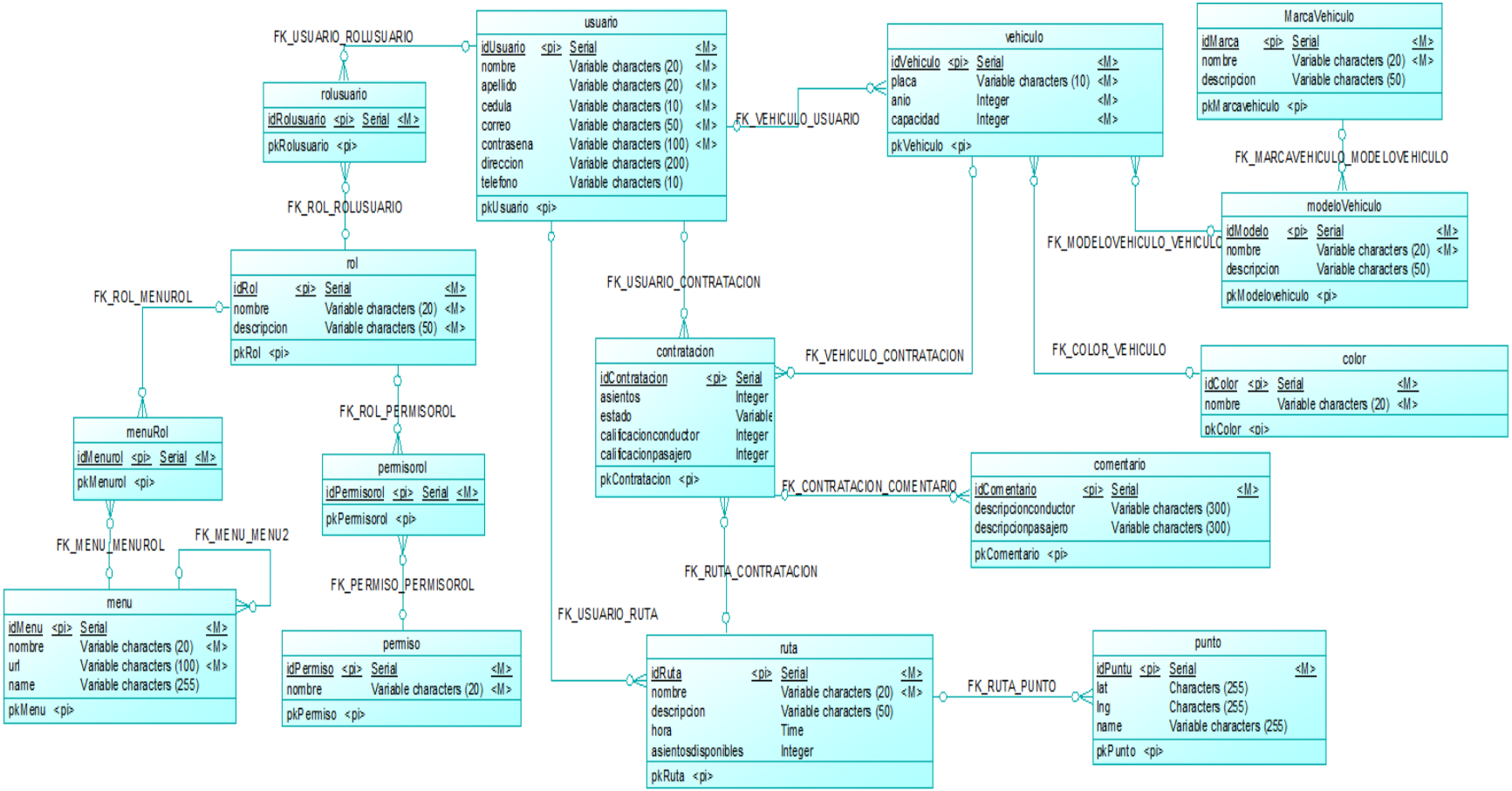


Figura 15. Diagrama Relacional
Fuente: Evelyn Benavides & Mayte Tixilema

Como se puede observar en la figura 18 se encuentra realizado el diagrama del modelo Entidad-Relación de la aplicación web en donde se encuentran las clases y atributos que pertenecen a la misma. En la figura 19 se puede observar el diagrama relacional del que se partirá para realizar el diseño de la base de datos.

En el **Anexo 4** se encuentran las especificaciones tanto del modelo relacional, como del modelo entidad relación, en donde se encuentran definidos sus clases y atributos.

3.5. Arquitectura utilizada para la implementación.

Para realizar la implementación de la aplicación web se utilizó la arquitectura de tres capas que se detallan a continuación: presentación, aplicación, datos.

En función del trabajo a destacar por parte de la aplicación web para el usuario final en cada capa se procede a diseñar un diagrama para mayor entendimiento en cuanto a la estructura del sistema.

Arquitectura Modelo Vista Controlador

Para realizar la implementación de la aplicación web se utilizó la arquitectura MVC, también conocido como la arquitectura en de tres capas que se detallan a continuación:

Capa de aplicación (M)

Dentro de esta capa se identifica el proceso de negocio que un usuario debe llevar acabo. Implica toda aquella tarea que son automatizadas para que el sistema cumpla con el objetivo propuesto, así como: los permisos, las restricciones.

Capa de presentación (V)

En esta sección se identifica como se van a diseñar la interfaz gráfica para el usuario, a fin de lograr ventanas de fácil manejo y comprensión.

Capa de datos (C)

Esta capa realiza el proceso de manejo de datos, a través de los datos que se encuentran almacenados en una base de datos.

3.5.1. Diagrama de la arquitectura en tres capas de la aplicación web para compartir vehículo.

Diagrama de la arquitectura en tres capas de la aplicación web para compartir vehículo

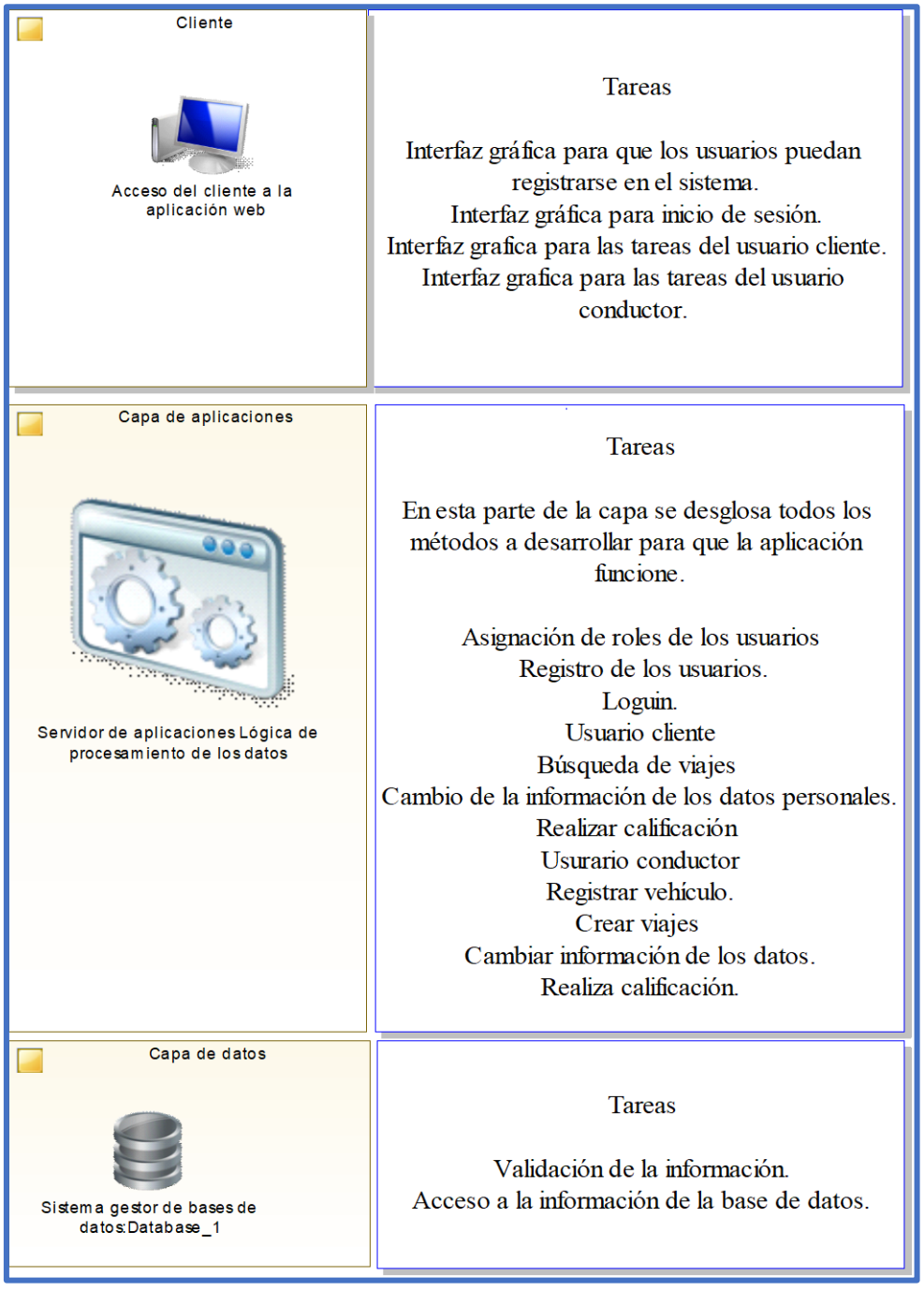


Figura 16.Diagrama de la arquitectura de tres capas
Fuente: Evelyn Benavides & Mayte Tixilema

Capítulo 4

Desarrollo del Sistema

Con el fin de obtener un desarrollo iterativo, adaptable y con varias fuentes bibliográficas se decidió utilizar el lenguaje de programación PHP con su Framework Laravel esta herramienta es flexible, robusta y permite que el desarrollo sea ligero y estructurado, tiene un sistema de plantillas (Blade), maneja la arquitectura MVC, permite organizar el código por paquetes y es compatible con cualquier sistema gestor de base de datos.

Para el almacenamiento de la información se utilizó como sistema gestor de bases de datos un software de código abierto (MySQL) que nos permite darle a la aplicación web todas las funcionalidades que necesita para cumplir con el objetivo propuesto. MySQL es el gestor predeterminado de la herramienta XAMPP y está disponible para todo tipo de sistemas operativos.

En este apartado se va a tratar sobre las herramientas que se usó para su desarrollo, la estructura utilizada, la conexión realizada con la base de datos, el estándar de codificación manejado, el desarrollo del módulo de seguridad a fin de gestionar todos los permisos que tendrán los usuarios (conductor y cliente) y desarrollo del módulo del usuario conductor y cliente.

4. Herramientas de desarrollo

4.1. PHP

PHP (Hypertext Preprocessor) según (Huebla, 2018) es un lenguaje de programación y se encuentra entre los más usados dentro del área de desarrollo. Con el que se puede diseñar excelentes pantallas para que el usuario final las pueda manejar con gran facilidad. Es de libre acceso y se ejecuta en el lado del servidor para lograr que el usuario cliente solo pueda visualizar el código HTML.

La velocidad, estabilidad y flexibilidad son las razones por las que se decidió utilizar este lenguaje de programación para crear la aplicación web. Permite integrar la aplicación web sin mayor dificultad con otras herramientas, así como con los servicios de Google Maps. Además, dispone de un sin número de fuentes de información y cuenta con un soporte en línea para que los desarrolladores pueden solventar cualquier problema presentado durante el desarrollo. Crear la aplicación web bajo el código PHP nos garantiza que el sistema pueda crecer y se pueda realizar el mantenimiento con gran facilidad, debido a que este lenguaje nos permite trabajar con módulos bien diseñados y estructurados con el que se logra obtener un código puro sin redundancia que sea reutilizable para múltiples funciones. Para comprender el proceso de cómo funciona internamente se toma como referencia el esquema de funcionalidad del Lenguaje PHP.

4.1.1. Funcionamiento

Diagrama de la funcionalidad del lenguaje PHP

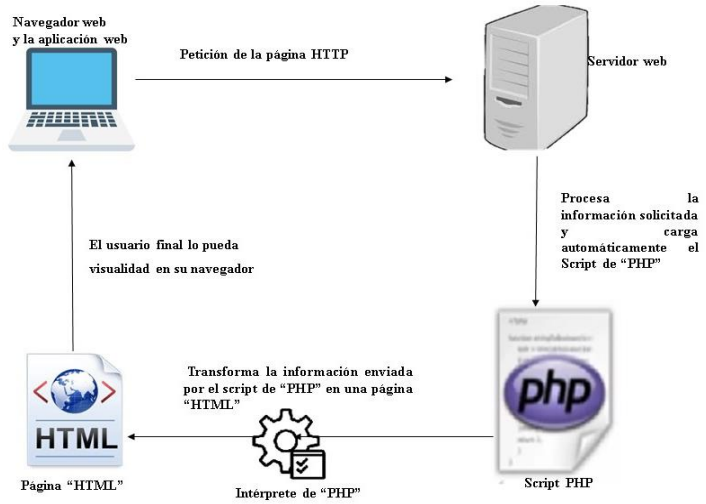


Figura 17. Esquema de funcionalidad de PHP
Fuente: (Robles, 2019)

Tomando como referencia el esquema planteado por el autor (Robles, 2019) en la figura 21, se toma como punto de partida el ordenador en el que ha inicializado el navegador web junto a la aplicación web que el usuario está visitando. En este caso se plantea como ejemplo, que la página visitada cuente con un formulario que el usuario debe llenar para navegar por el sistema,

en cada interacción con el usuario el navegador web se comunica internamente envido una petición "**HTTP**" hacia el servidor web, el mismo que procesa la información solicitada y carga automáticamente el Script de "**PHP",** este a su vez recibe los datos desde "**POST**" o "**GET"**, los procesa y devuelve. El intérprete de "**PHP"**, por otra parte, se activa y transforma la información enviada por el script de "**PHP"** en una página "**HTML"** la cual es enviada como respuesta para que el usuario final lo pueda visualidad en su navegador.

4.2. Framework Laravel.

Según el autor (Malan, 2016) Laravel es una herramienta de código abierto que fue creada bajo el lenguaje de programación PHP con el fin de lograr desarrollar las aplicaciones web de manera ágil y utilizando la arquitectura MVC (Modelo, Vista, Controlador) en donde el Framework puede responder de manera directa la petición HTTP realizada hacia el servidor web. Además, cuenta con una paquetería que permite la administración de las bases de datos y la gestión de los usuarios, permitiéndonos trabajar bajo el código modular y escalable.

Para los autores (Espinoza & Vera, 2017) Laravel es una herramienta que brinda muchas facilidades para los desarrolladores debido a que permite trabajar con paquetes, los mismos que se pueden encontrar en su repositorio oficial.

Al realizar un análisis sobre el framework a utilizarse se destaca el sistema de gestión de contenidos CMS (Content Management System), el mismo que contiene varios entornos de desarrollo de los que se optó por utilizar el Framework de PHP Laravel debido a que es bastante flexible y robusto por lo que permite diseñar la aplicación web de manera personalizada y culminar el diseño de forma ágil con menos tiempo de inversión. Otra de las razones por las que se decidió dar uso de este framework es la capacidad que tiene en cuanto a la reducción y reutilización del código mediante el uso de la plantilla "Blade". La seguridad es la parte más importante de una aplicación web, por lo que desarrollar en este Framework nos garantiza la integridad de los datos debido a que Laravel incluye un sistema de autenticación basado en

tokens (o valor único) y trabaja con un token CSRF que evita que otras personas envíen peticiones no deseadas a nuestra aplicación También se garantiza que el sistema desarrollado pueda crecer por lo que facilita añadir más funcionalidades. Por otro lado, trabajar con el Framework de Laravel nos proporciona un conjunto de directorios el mismo que facilita la gestión de código al momento de desarrollar el sistema.

4.2.1. *Estructura de directorios.*

Al iniciar un nuevo proyecto en Laravel se crea una estructura ordenada de directorios, la misma que facilita su manejo al momento de desarrollar la aplicación web. Para la creación de un nuevo proyecto se da uso de la herramienta composer, para lo cual se ejecuta la siguiente línea de comandos **"cgomposer create-project laravel/laravel nombre_del_proyecto".** Una vez ejecutado el comando se crea un directorio con el nombre del proyecto, dentro del cual se crea un conjunto de directorios como se detalla en la siguiente estructura de directorios diseñada por el autor (Velasque, 2013) .

Diagrama de directorios del framework de laravel

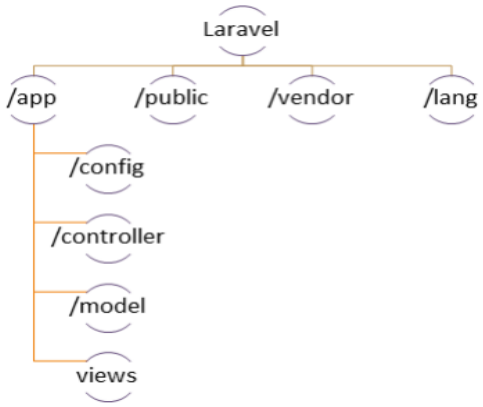


Figura 18. Estructura de directorios de Laravel
Fuente: (Velasque, 2013)

Tomando de base la figura 22 se describe a continuación el contenido de la estructura de los directorios más sobresalientes según el autor (Arroyo, 2019) que tiene un proyecto creado en Laravel.

Tabla 6. Directorios de un proyecto de Laravel

Nombre	Descripción
Laravel	Directorio principal que contiene en su interior todos los demás directorios que hacen parte del proyecto, es el directorio cuyo nombre es el que se le asignó al momento de crear el proyecto por primera vez utilizando la herramienta composer.
/app	Este directorio contiene la mayor parte del código para que la aplicación web funcione. Además, contiene los siguientes subdirectorios: Console, Exceptions, Helpers, Models, Providers.
/public	Directorio que da paso a los usuarios para visualizar su contenido en el navegador web, al hacer una búsqueda de la página que desea visitar, esta envía una petición, la misma que debe pasar por este directorio para que se ejecute todo el proceso.
/vendor	Contiene las librerías y las dependencias de las que está compuesta el framework.
/resources /Lang	Almacena todos los archivos de origen PHP cuyo contenido es un conjunto de Arrays. Al pasar por este directorio los textos que contiene los Arrays se traducen para que la aplicación lo pueda interpretar.
/app/config	El directorio contiene todos los archivos de configuraciones que se realiza durante el desarrollo de una aplicación.
/app/Http/Controllers/	Permiten la comunicación con las rutas, modelos, clases con el fin de dar la información que el usuario está buscando dentro de la aplicación.

/app/model	Contiene los modelos creados para el proyecto. Para crear los modelos se debe ejecutar en siguiente comando “artisan make: model nombre del modelo”.
/app/resources/views/	Contiene las posibles vistas creadas para cumplir el objetivo de una aplicación, ya que las vistas no son más que un conjunto de documentos con la extensión “.**php**” pero que en su interior está escrito por código "**HTML** “para las salidas de la aplicación.

Nota: Detalle de los directorios más sobresalientes de Laravel

4.3. Sistema gestor de base de datos.

Dicho por los autores (Capote, Ruiz, Rodriguez, Carrillo, & Miranda, 2009) el sistema gestor de bases de datos es una herramienta que funciona similar a una interfaz gráfica para el usuario que permite interactuar a las aplicaciones con el sistema operativo del ordenador. Contiene una serie de programas que permiten crear, actualizar, eliminar y realizar el mantenimiento de una base de datos, garantizando de esta manera la disponibilidad y la integridad de los datos.

4.3.1. *MySQL.*

Para el autor (Rouse, 2018) MySQL es conocido como un sistema gestor de bases de datos (**SGBD**) de código abierto que se caracteriza por su velocidad en cuanto al procesamiento de la información, es bastante robusto y estable. MySQL posee un lenguaje de consultas estructurado y representa la mejor herramienta para almacenar los datos de nuestra aplicación web. El rendimiento, la flexibilidad, la escalabilidad y la seguridad son las principales razones por que se da uso de este Sistema Gestor de Bases de Datos. Además, es uno de los motores de bases de datos con las que el Framework de Laravel se comunica a la perfección sin el riego de tener perdidas de información. También se tomó en consideración el rendimiento en cuanto a la velocidad de las transacciones y número de peticiones que los usuarios lo realicen. Usar MySQL nos garantiza la confiabilidad de los datos debido a que cuenta con un mecanismo

fuerte de seguridad, con la que se puede autenticar a los usuarios y asignar ciertos privilegios. Además, cuenta con un soporte de cifrado y descifrado de la data con el que se puede asegurar la integridad de la misma contra los usuarios no autorizados.

4.4. API de Google para geolocalización.

Según el autor (Olcina, 2017) las APIs de Google facilita a los desarrolladores poner en comunicación la aplicación web que están desarrollando con otra aplicación externa y así compartir servicios entre sí, el mismo que se puede lograr mediante el acceso al código que está compuesto por una serie de funciones y procedimientos con lo que se evita la programación desde cero. Una de las razones por las que el API de Google es bastante utilizado por los desarrolladores es la facilidad con la que pueden integrar un determinado sistema con otras aplicaciones.

Para el autor (Cuajimalpa, 2018) Google Maps es un servicio que brinda Google, con el que se puede obtener la ubicación de un objeto a fin de verificar la ruta sugerida por el mapa. Para integrar este servicio con la aplicación web que se está desarrollando se ha dado uso de los siguientes APIs (API de JavaScript de Maps, API key de google, API Directions).

4.4.1. *API key de Google.*

(Martín J. , 2018) explica que para agregar las funcionalidades de los mapas de Google es necesario añadir el código directamente desde el repositorio de Google Maps y se debe contar con la presencia de un API KEY de Google Maps con el fin de lograr la manipulación completa del mapa.El API KEY es el único identificador que logra autenticar todas las peticiones asociadas con el proyecto que se está desarrollando. Según (Google, 2020) para generar el API KEY y agregarlo a la página que se está desarrollando se debe ingresar al área de consola **Developers** y crearlo para lo cual se debe ingresar en el siguiente enlace:

https://console.developers.google.com/projectselector2/apis/dashboard, en donde se creará o elegirá un proyecto al que se le desea agregar el API (figura 23).

Figura 19. Pagina para agregar el API KEY al proyecto:
Fuente: (Google, 2020)

Una vez dentro de la página se debe seguir los pasos que se detallan a continuación.

1. Entrar a la consola de Google Maps
2. Crear el proyecto para el que se le va a agregar el API.
3. Seleccionar el API y todos los Servicios.
4. Ingresar en la página de credenciales y crear una nueva para el API KEY.
5. En la ventana de la API KEY se puede visualizar la clave que se acaba de crear.
6. La clave recién creada se puede ver en la página de Credenciales.

4.4.2. *API de JavaScript de Maps.*

Según el autor (Padial, 2015) el API de JavaScript de Maps es soportado por todos los navegadores web que permiten la configuración de las imágenes con el fin de personalizar el contenido del mapa para ser visualizado en la página web. Para la localización de un objeto se puede obtener mediante las coordenadas GPS, con lo que se puede acceder a la información de latitud y longitud, el mismo que puede ser enviado hacia el servidor web para ser utilizado según el programador lo amerite. Para cargar el API se debe ejecutar el siguiente script, que debe adjuntarse al código HTML que se esté utilizando.

```
<script async defer src=https://maps.googleapis.com/maps/api/js?
Key=YOUR_API_KEY&callback=initMap”> </script>
```

Script para implementar el API_KEY
Fuente: (Padial, 2015)

En la figura 24 se puede identificar como se ve la pagina donde se encuentra el API de JavaScript, para hacer uso de sus servicios dentro de la aplicación web que se está desarrollando.

Página principal de maps javascript

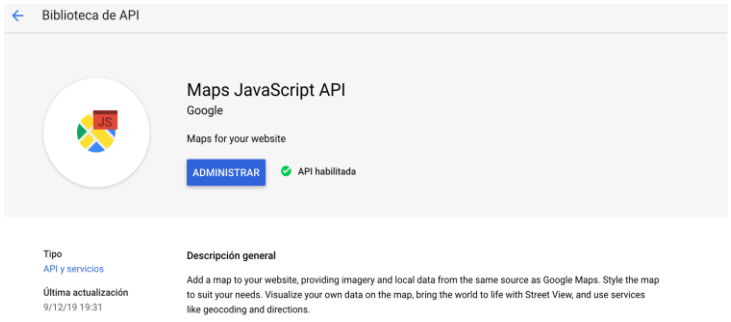


Figura 20. API de Google Maps JavaScript
Fuente: (Google, 2020)

4.5. Estructura utilizada para el desarrollo.

El objetivo por el que se plantea el siguiente diagrama (Figura 25) es con el propósito de detallar la forma como se encuentra estructurado el proyecto desde su repositorio raíz en el servidor web (Apache) y también detallar como los usuarios hacen que el sistema cobre vida al interactuar con el mismo. Se toma como base el Framework de Laravel donde se encuentra alojada la aplicación web, que está compuesto por un conjunto de plantillas, paquetes y con una conexión hacia la herramienta externa para hacer uso de sus servicios (API de Google). Adminlte es la plantilla principal con la que los usuarios finales interactúan directamente para realizar cualquier petición hacia el servidor web, el motor de plantilla Blade trabaja internamente con el propósito de ayudar al desarrollador a optimizar el código y reutilizar el mismo. El uso de la paquetería es un entorno propio de Laravel con el fin de asegurar que se está implementando la arquitectura MVC, en donde los modelos son los encargados de hacer

que el sistema gestor de base de datos se comunique con la aplicación a fin de que los usuarios logren visualizar la información solicitada.

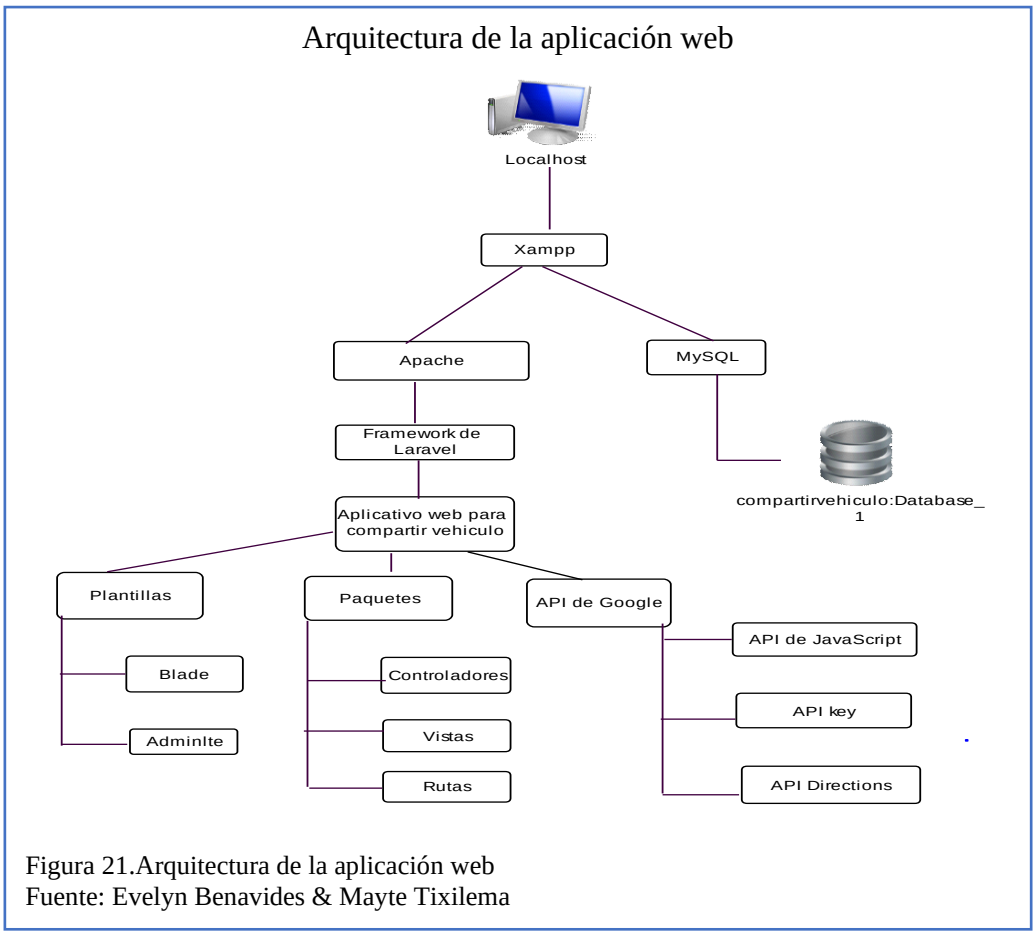


Figura 21.Arquitectura de la aplicación web
Fuente: Evelyn Benavides & Mayte Tixilema

Tomando como base el diagrama de la arquitectura de la aplicación web, se procede a detallar las configuraciones iniciales.

4.5.1. Configuraciones Iniciales

Tabla 7. Configuraciones iniciales de los componentes

Componentes	Configuraciones
Ordenador PC/Laptop	Es el contenedor de la herramienta base para el desarrollo “XAMPP”, en donde se ha levantado el servidor de aplicaciones de manera local (localhost).
Arquitectura de Software	Se utilizó la arquitectura MVC (Modelo Vista Controlador), mediante el uso del Framework Laravel.

XAMPP	Esta herramienta contiene el servidor web “Apache” y el sistema gestor de bases de datos “MySQL”.
Visual Studio Code	Para iniciar con el desarrollo se optó por usar el digitalizador de código, debido a que cuenta con las librerías necesarias para trabajar con el Framework de Laravel.
Laravel	Para iniciar con el desarrollo, se utilizó este Framework en donde se tiene que instalar (https://laravel.com/docs/7.x) la herramienta “composer” el mismo que nos ayuda para el control y gestión de las dependencias que Laravel usa para el correcto funcionamiento.
MySQL	Se usa el phpmyadmin, el mismo que provee el “PHP” que viene incrustado dentro de la herramienta XAMPP.
Api de Google	Para hacer uso de los servicios API de Google, se creó una cuenta de facturación en Google iCloud, el mismo nos permite utilizar todas las funciones de las APIs.

Nota: Configuración de los componentes de software que se va a utilizar para el desarrollo

4.6. Conexión con la base de datos.

Para la configuración de la base de datos dentro de Laravel, en el directorio **database** se va a encontrar el fichero. env, en donde se encuentra el nombre de la aplicación y el código para la configuración de la base de datos. A continuación, se detalla cómo se debe configurar la misma.

"DB_CONNECTION” = Nombre del gestor de bases de datos que se va a utilizar

"DB_HOST” =Se coloca el puerto donde se encuentra alojada la base de datos o el puerto 127.0.0.1 (localhost)

"DB_PORT” = Se coloca el número del puerto de el motor de base de datos que se está utilizando.

"DB_DATABASE” = Nombre de la base de datos en donde se almacenará la información

"DB_USERNAME" = Nombre del usuario que por lo general es root

"DB_PASSWORD" = Contraseña del motor de base de datos si así lo requiere.

En la figura 26 se puede visualizar como se configuro la base de datos diseñada para la aplicación web.

Fichero para la configuración de las bases de datos

```
APP_NAME=Laravel
APP_ENV=local
APP_KEY=base64:8+C+FaKta8WC1+hoktCi2HXRkduRdnRJhS0a9MIqI5w=
APP_DEBUG=true
APP_URL=http://localhost

LOG_CHANNEL=stack

DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=compartirVehiculo
DB_USERNAME=root
DB_PASSWORD=
```

Figura 22. Configuración del fichero. env
Fuente: Evelyn Benavides & Mayte Tixilema

El Framework de Laravel proporciona la herramienta de migraciones que facilitará la creación de las tablas que el motor de bases de datos requiere para cumplir con el objetivo propuesto. En la figura 27 se puede visualizar las migraciones realizadas para la aplicación web. El directorio de las migraciones se las puede visualizar luego a haber ejecutado las migraciones creadas.

Directorio de las migraciones

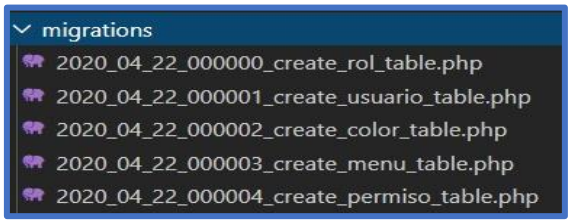


Figura 23. Migraciones realizadas para crear las tablas que el SGBD requiere
Fuente: Evelyn Benavides & Mayte Tixilema

En este caso Laravel tiene disponibles las migraciones para crear la tabla usuario y la tabla para restablecer la contraseña. Para realizar las migraciones creadas se debe ejecutar en la consola el siguiente comando **"php artisan migrate",** si todo está correcto no se debe indicar ningún error y recibir un mensaje que las migraciones se ha ejecutado con éxito como se detalla en la figura 28.

Migraciones finalizadas con éxito

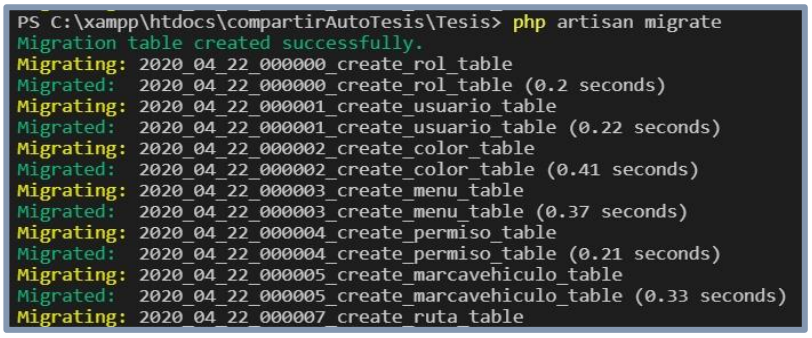


Figura 24. Migraciones realizadas con éxito
Fuente: Evelyn Benavides & Mayte Tixilema

4.6.1. ***Diccionario de datos.*** Con el fin de organizar los datos y tener un repositorio organizado se ha realizado el diccionario de datos, tomando como base el diseño de la base de datos relacional, en donde se van a detallar cada una de las tablas y atributos de las que está conformada la base de datos. El diccionario de datos se lo puede observar en el **Anexo 5.**

4.6.2. ***Estándar de codificación*.** Debido a que se ha utilizado el lenguaje de programación PHP, juntamente con el Framework Laravel, se establece una estandarización del código fuente, con el objetivo de evitar código basura y facilitar el mantenimiento de la aplicación web, el mismo se lo puede encontrar en el **Anexo 6.**

4.6.3. ***Desarrollo de los módulos.*** Para el desarrollo de los módulos de seguridad, usuario conductor y usuario cliente o pasajero se da uso de las herramientas que se detallaron anteriormente, con el fin de lograr diseñar las ventanas necesarias y dar toda la funcionalidad que el sistema requiere para que los usuarios puedan interactuar con la aplicación web. Durante el desarrollo de los módulos se da uso los paquetes "**Modelos**, **Controladores**, **Rutas**". donde los modelos son las encargadas de realizar todas las peticiones relacionadas con la base de datos. Mientras que el controlador maneja toda la parte funcional del sistema, el miso que consta de todos los métodos, funciones que se han utilizado para el desarrollo. Vista es la parte que el usuario puede ver del sistema para lo cual se da uso de las rutas con el propósito de formar con conjunto de todas las vistas creadas en un archivo denominado "web.php".

4.6.4. *Desarrollo del módulo de seguridad.*

El desarrollo del módulo de seguridad se lo realiza con el objetivo de autenticar a los usuarios que ingresan a la aplicación web, mediante la validación del correo electrónico y la contraseña con las que se registraron en el sistema. Para lograr con el objetivo se procede a autenticar a todos los usuarios, para lo cual se les asignan los diferentes permisos dependiendo del rol con la que se registre en el sistema. En este apartado se van a especificar los modelos creados para la interacción con la base de datos, los controladores principales, las rutas utilizadas para el acceso y las ventanas diseñadas para el usuario interactúe con el sistema.

4.6.4.1. *Especificación de modelos.*

Los modelos creados dentro de este módulo nos van a permitir acceder a las tablas de la base de datos (usuario, menú, permiso, rol, etc.) e interactuar con la misma, para poder realizar las consultas necesarias y darle funcionalidad al módulo. Como se puede observar en la figura 29 se han creado las carpetas Admin y Seguridad para tener una mejor organización de los paquetes y los modelos Menú, Menurrol, Permiso, Rol y Usuario, que son los que se han utilizado en este módulo.

Ficheros del módulo de seguridad

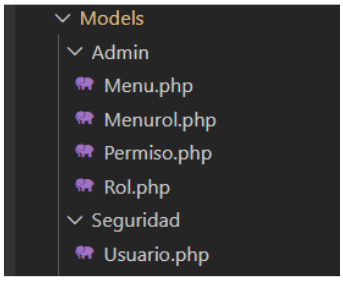


Figura 25. Modelos del módulo seguridad
Fuente: Evelyn Benavides & Mayte Tixilema

Para crear los modelos dentro de Laravel se utiliza el comando “php artisan make: model nombre_modelo”, en este caso se han creado los modelos especificando la ruta en la que se van a almacenar, como se puede ver en la figura 30, la creación del modelo Usuario.

Creación del modelo usuario

```
PS C:\xampp\htdocs\compartirAutoTesis\Tesis> php artisan make:model Models/Seguridad/Usuario
Model created successfully.
```

Figura 26. Creación del modelo Usuario
Fuente: Evelyn Benavides & Mayte Tixilema

La estructura dentro de los modelos, tomando en cuenta que Laravel provee librerías para realizar la autenticación, es la que se muestra en la figura 31 con el ejemplo del modelo Usuario, en donde se observa el directorio principal (Ruta en donde se encuentra el modelo), paquetes importados (Paquetes necesarios para poder realizar los métodos), propiedades de la tabla usuario (nombre de la tabla, nombre de los campos, especificación de la clave primaria y timestamps) y métodos a los que tendrán acceso los controladores.

Modelo para módulo de seguridad

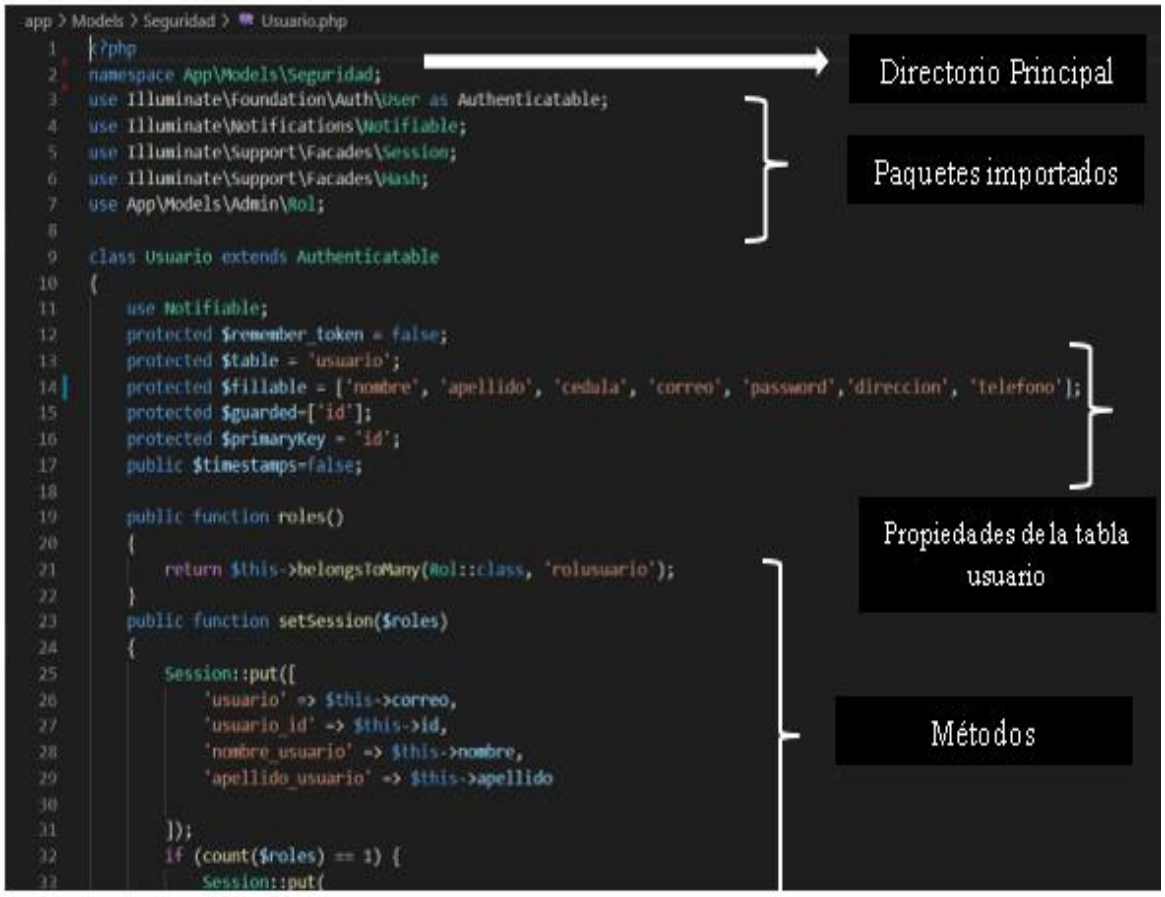


Figura 27. Modelo Usuario para el módulo de seguridad
Fuente: Evelyn Benavides & Mayte Tixilema

4.6.4.2. *Controladores principales*

Los controladores que se han utilizado en este módulo nos permitirán comunicarnos con los modelos y acceder a las vistas mediante las rutas. Como se puede observar en la figura 32, se han creado las carpetas Admin y Seguridad para almacenar los controladores que se utilizarán en este módulo.

Controladores del módulo de seguridad

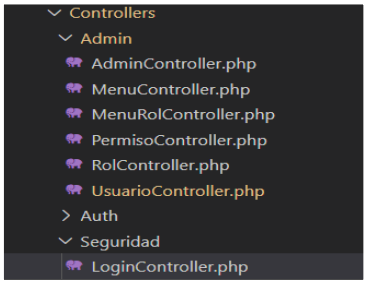


Figura 28. Controladores que utiliza el módulo de seguridad
Fuente: Evelyn Benavides & Mayte Tixilema

Para crear los controladores dentro de Laravel se utiliza el comando "php artisan make:controller nombre_controlador", como se puede ver en la figura 33 la creación del controlador UsuarioController.

Creación del controlador

Figura 29. Creación del controlador UsuarioController
Fuente: Evelyn Benavides & Mayte Tixilema

El controlador como se muestra en la figura 34 tomando como ejemplo el controlador UsuarioController, está formado por el namespace (directorio en donde se encuentra el controlador), paquetes importados (paquetes que se utilizarán para poder realizar los métodos) y métodos.

Controlador de usuario

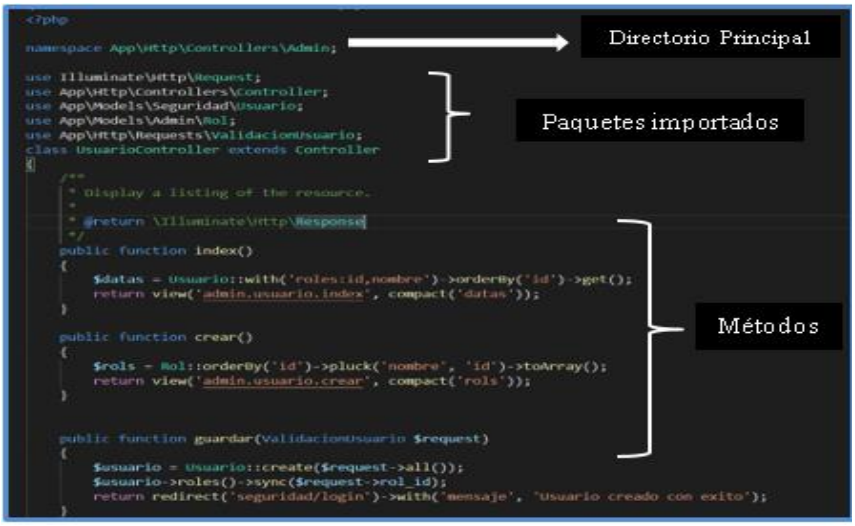


Figura 30. Controlador UsuarioController
Fuente: Evelyn Benavides & Mayte Tixilema

Los métodos principales que se han realizado para que el módulo pueda funcionar son los que se muestran en la siguiente tabla:

Tabla 8.Método principales del controlador

Controlador	Métodos	Descripción
UsuarioController.php	index()	Muestra una lista de usuarios según su rol
	guardar(ValidacionUsuario $request)	Crea un usuario según su rol
	editar(Request $request, $id)	Actualiza la información del usuario
MenuController	index()	Muestra una lista de menús disponibles
	guardar(ValidacionMenu $request)	Crea un menú y lo almacena en la tabla menú
RolController	index()	Muestra una lista de roles disponibles

	guardar(ValidacionMenu $request)	Crea un rol y lo almacena en la tabla menú
LoginController	_construct()	Constructor que muestra el middleware disponible
	username()	Establece que el username que se utilizará en este sistema será el correo para el inicio de sesión
	authenticated(Request $request, $user)	Método que permitirá la autenticación del usuario según su rol.

Nota: Métodos principales realizados para el correcto funcionamiento de los controladores

4.6.4.3. Rutas utilizadas para el acceso

Las rutas establecidas en el archivo web.php establecen las rutas de acceso que tendrá la aplicación. En la figura 35 se muestran las rutas para acceder al módulo de seguridad.

Rutas principales para el acceso

```
Route::get('/', 'InicioController@index')->name('inicio');

  /*RUTAS PARA AUTENTICACIÓN*/
Route::get('seguridad/login', 'Seguridad\LoginController@index')->name('login');
Route::post('seguridad/login', 'Seguridad\LoginController@login')->name('login_post');
Route::get('seguridad/logout', 'Seguridad\LoginController@logout')->name('logout');
Route::post('ajax-sesion', 'AjaxController@setSession')->name('ajax')->middleware('auth');

/*RUTAS DE USUARIO*/
Route::get('usuario', 'Admin\UsuarioController@index')->name('usuario');
Route::get('usuario/crear', 'Admin\UsuarioController@crear')->name('crear_usuario');
Route::post('usuario', 'Admin\UsuarioController@guardar')->name('guardar_usuario');
Route::get('usuario/{id}/editar', 'Admin\UsuarioController@editar')->name('editar_usuario');
```

Figura 31. Rutas de acceso al módulo de seguridad
Fuente: Evelyn Benavides & Mayte Tixilema

4.6.4.4. *Diseño de interfaces.*

La ventana que se puede apreciar en la figura 36, es el espacio en el que los usuarios pueden ingresar el correo y la contraseña para su registro en el sistema con el fin de navegar por la aplicación web. El usuario podrá acceder a su cuenta y podrá ejecutar las tareas dependiendo

del rol que hayan seleccionado. Ya que el sistema de autenticación para los usuarios internamente valida el tipo de rol seleccionado para en base a ello, otorgar todos los permisos correspondientes.

Inicio de sesión

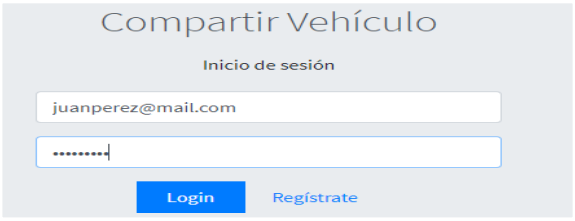


Figura 32. Ventana para iniciar sesión
Fuente: Evelyn Benavides & Mayte Tixilema

La figura 37 muestra el formulario en el que los usuarios que desean hacer uso de la aplicación web se registran llenando todos los campos solicitados. El usuario debe seleccionar el tipo de Rol (Conductor o cliente/pasajero) ya que dependiendo de su elección el sistema internamente se ejecuta y da todos los permisos para que pueda navegar por el sistema.

Ventana para registro de usuarios

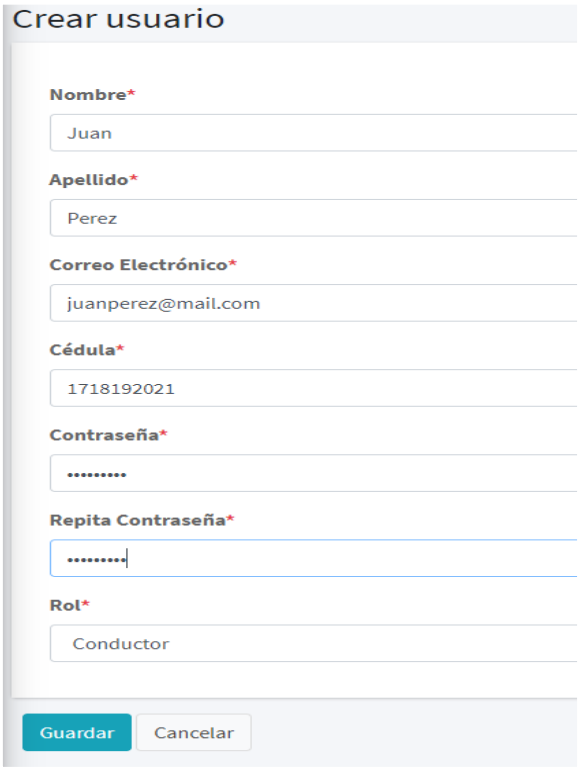


Figura 33. Ventana de registro de los usuarios
Fuente: Evelyn Benavides & Mayte Tixilema

Los usuarios independientemente del rol que hayan seleccionado al momento de registrarse en el sistema contarán con la opción de actualizar sus datos en el menú de tareas. Para ello deben llenar todos los campos solicitados por el formulario como se detalla en la figura 37.

4.6.5. *Desarrollo del módulo del usuario conductor*.

Este módulo está desarrollado con el objetivo de que los usuarios registrados con el rol de conductor puedan ofertar sus servicios de una manera segura. Para lo cual el presente módulo será capaz de que los usuarios registrados como rol conductor tenga los privilegios de registrar su vehículo, publicar viajes indicando la ruta por donde se va a trasladar de un punto a otro. En este apartado se va a detallar la especificación de los modelos utilizados, los controladores principales, las rutas utilizadas para el acceso y el diseño de las interfaces para que el usuario conductor pueda interactuar con el sistema.

4.6.5.1. *Especificación de modelos.*

Los modelos creados dentro de este módulo nos van a permitir acceder a las tablas de la base de datos (usuario, vehículo, modelo Vehículo, marca Vehículo, color, contratación, comentario, ruta y punto) e interactuar con la misma, para poder realizar las consultas necesarias y darle funcionalidad al módulo. Como se puede observar en la figura 38 los modelos que se han utilizado son Color, Comentario, Contratación Marca vehículo, Modelo vehículo, Punto, Ruta, Vehículo.

Modelos del usuario conductor

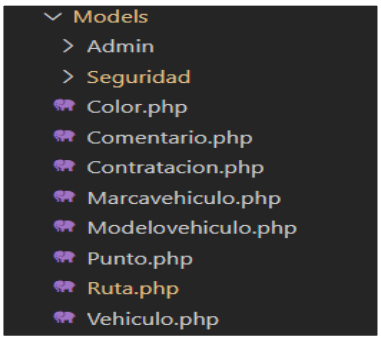


Figura 34. Modelos del módulo conductor
Fuente: Evelyn Benavides & Mayte Tixilema

Para crear los modelos se utiliza el comando “php artisan make:model nombre_modelo”, en este caso se han creado los modelos especificando la ruta en la que se van a almacenar, como se puede ver en la figura 39, la creación del modelo Vehículo.

Creación del modelo vehículo

```
PS C:\xampp\htdocs\compartirAutoTesis\Tesis> php artisan make:model Models/Vehiculo
Model created successfully.
```

Figura 35. Creación del modelo Vehículo
Fuente: Evelyn Benavides & Mayte Tixilema

La estructura dentro de los modelos es la que se muestra en la figura 40 con el ejemplo del modelo Vehículo, en donde se observa el directorio principal (Ruta en donde se encuentra el modelo), paquetes importados (Paquetes necesarios para poder realizar los métodos) y las propiedades de la tabla vehículo (nombre de la tabla, nombre de los campos, especificación de la clave primaria y timestamps).

Modelo vehículo

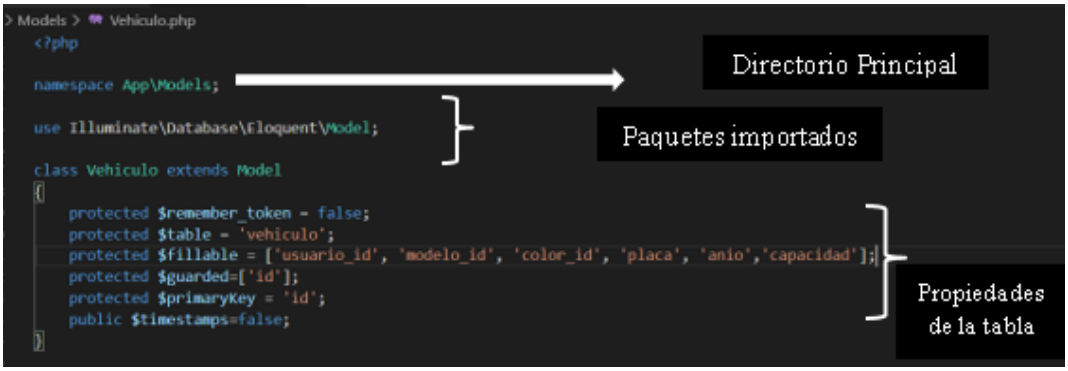


Figura 36. Modelo Vehículo para el módulo del usuario conductor
Fuente: Evelyn Benavides & Mayte Tixilema

4.6.5.2. *Controladores principales.*

Los controladores que se han utilizado en este módulo nos permitirán comunicarnos con los modelos y acceder a las vistas mediante las rutas. Como se puede observar en la figura 41, se presentan los controladores que se utilizarán en este módulo.

Ficheros de los controladores

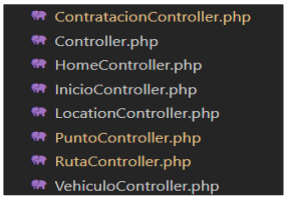


Figura 37. Controladores que utiliza el módulo del usuario conductor
Fuente: Evelyn Benavides & Mayte Tixilema

Para crear los controladores se utiliza el comando "php artisan make:conntroller nombre_controlador", como se puede ver en la figura 42 la creación del controlador VehículoController.

Creación del controlador

Figura 38. Creación del controlador VehículoController
Fuente: Evelyn Benavides & Mayte Tixilema

El controlador como se muestra en la figura 43 tomando como ejemplo el controlador VehiculoController, está formado por el namespace (directorio en donde se encuentra el controlador), paquetes importados (paquetes que se utilizarán para poder realizar los métodos) y métodos (En los métodos se utiliza la función can(nombre_permiso) que permite que el usuario con el rol de conductor tenga acceso a ese permiso).

Controladores para el usuario controller

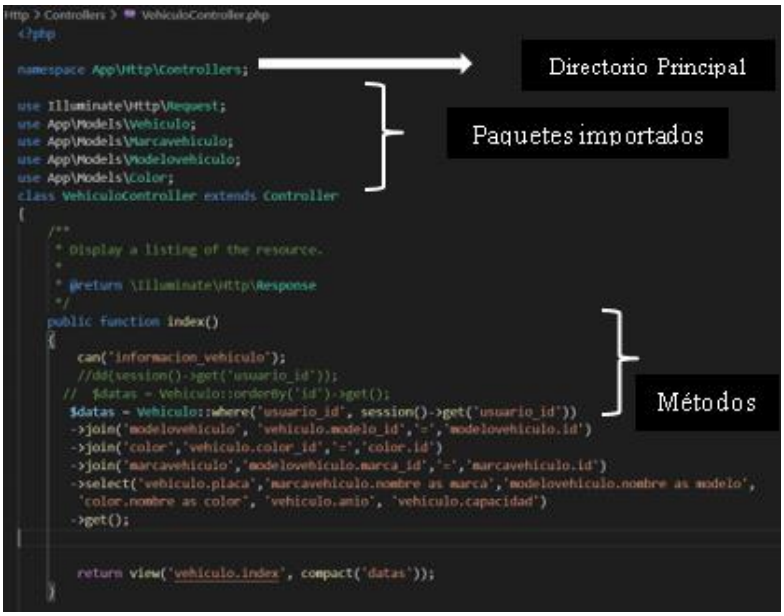


Figura 39. Controlador UsuarioController
Fuente: Evelyn Benavides & Mayte Tixilema

Los métodos principales que se han realizado para que el módulo pueda funcionar son los que se muestran en la siguiente tabla:

Tabla 9. Métodos principales de los controladores para el usuario conductor.

Controlador	Métodos	Descripción
VehiculoController.php	index()	Muestra la información del vehículo asociado al conductor.
	guardar(Request $request)	Crea un vehículo con las características específicas
	getModelo(Request $request)	Devuelve una lista de modelos, que se mostrarán en la vista de la creación de vehículos.
RutaController	index()	Muestra una lista de rutas asociadas al usuario que se encuentra en la sesión.
	guardar(Request $request)	Crea una ruta y la almacena en la tabla ruta, con el id del usuario que se encuentra en la sesión
LocationController	index()	Muestra una lista de puntos según una ruta que se mostrarán en el mapa.
	store(LocationRequest $request)	Guarda los puntos asociados a una ruta
ContratacionController	index()	Muestra una lista de rutas que el usuario realizó

	actualizar(Request $request, $id)	Establece una calificación y un comentario al usuario pasajero
	actualizarSolicitud(Request $request, $id)	Permite aceptar las solicitudes enviadas al conductor, para solicitar una ruta.
	cancelarSolicitud(Request $request, $id)	Permite cancelar las solicitudes enviadas al conductor.

Nota: Detalle de los métodos principales elaborados para el usuario conductor

4.6.5.3. *Rutas utilizadas para el acceso.*

Las rutas establecidas en el archivo web.php establecen las rutas de acceso que tendrá la aplicación. En la figura 44 se muestran las rutas que se han utilizado para acceder al módulo del usuario conductor.

Rutas para el acceso al módulo usuario conductor

```
Route::get('vehiculo', 'VehiculoController@index')->name('vehiculo')->middleware('auth');
Route::get('vehiculo/crear', 'VehiculoController@crear')->name('crear_vehiculo')->middleware('auth');
Route::get('vehiculo/modelo', 'VehiculoController@getModelo');
Route::post('vehiculo', 'VehiculoController@guardar')->name('guardar_vehiculo')->middleware('auth');
Route::post('vehiculo/{vehiculo}', 'VehiculoController@ver')->name('ver_vehiculo')->middleware('auth');

/*Rutas*/
Route::get('ruta/crear', 'RutaController@crear')->name('crear_ruta')->middleware('auth');
Route::post('ruta', 'RutaController@guardar')->name('guardar_ruta')->middleware('auth');
Route::get('ruta/crear', 'RutaController@index')->name('vehiculo')->middleware('auth');


/*Contratación*/
Route::get('contratacion', 'ContratacionController@index')->name('contratacion')->middleware('auth');
Route::put('contratacion/{id}', 'ContratacionController@actualizar')->name('actualizar_contratacion');
Route::put('solicitud/{id}', 'ContratacionController@actualizarSolicitud')->name('actualizar_solicitud');
Route::put('solicitudCancelada/{id}', 'ContratacionController@cancelarSolicitud')->name('cancelar_solicitud');
```

Figura 40. Rutas de acceso al módulo del usuario conductor
Fuente: Elaboración propia

4.6.5.4. *Diseño de interfaces*.

Es la ventana principal en la que el usuario conductor va a encontrar un conjunto de tareas que podrá realizar dentro de la aplicación web, contiene un menú dinámico y submenús con todas las tareas como se puede visualizar en la figura 45.

Interfaz gráfica del usuario conductor

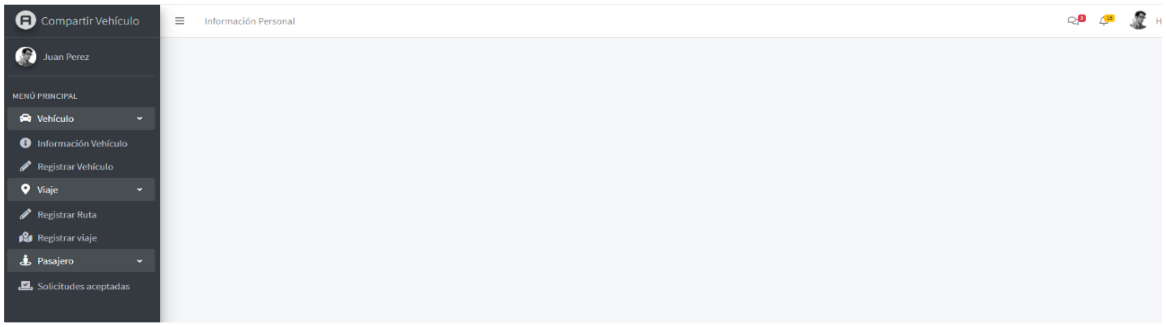


Figura 41.Ventana principal del usuario conductor
Fuente: Evelyn Benavides & Mayte Tixilema

El menú vehículo contiene los submenús Información vehículo y Registrar vehículo. En la opción la Información del Vehículo se puede apreciar los datos del vehículo con el fin de que el usuario pueda visualizar la información de su vehículo como se detalla en la figura 46.

Interfaz con la información del vehículo

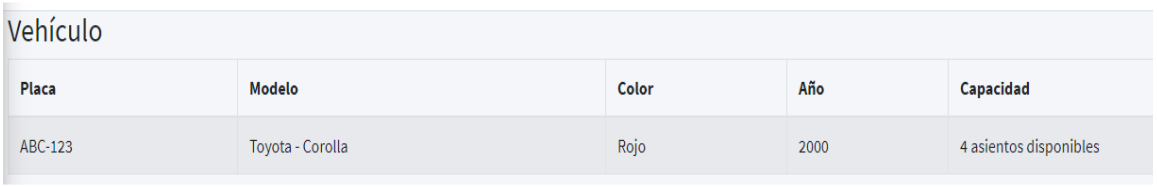


Figura 42.Ventana con la presentación de la información del vehículo
Fuente: Evelyn Benavides & Mayte Tixilema

En el submenú Registrar vehículo, el usuario conductor cuenta con un formulario de registro del vehículo, con el objetivo de que pueda ofertar los servicios para los usuarios clientes (pasajeros), como se puede apreciar en la figura 47.

Interfaz para que el usuario registre su vehículo

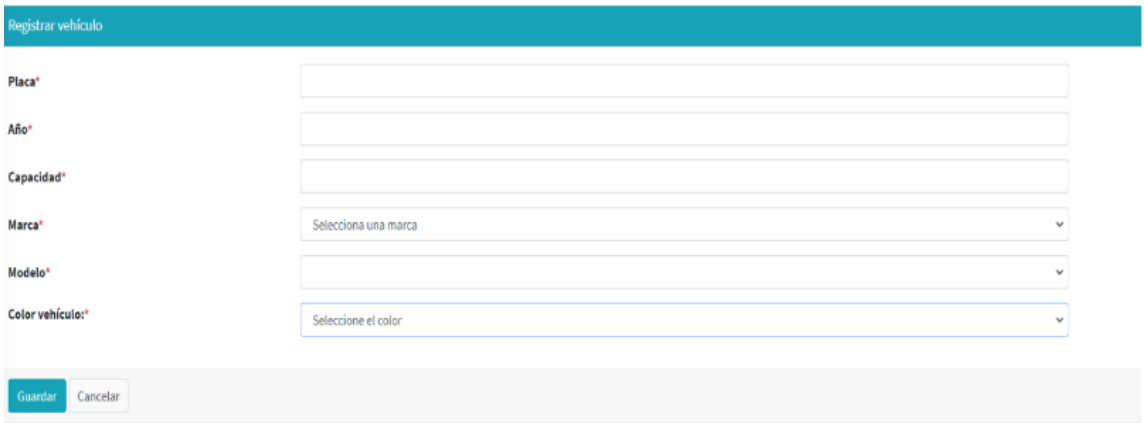


Figura 43. Ventana para registrar el vehículo
Fuente: Evelyn Benavides & Mayte Tixilema

En el menú Viajes el usuario conductor cuenta con los submenús Registrar ruta y Registrar viajes. En la parte de registrar la ruta, el usuario conductor podrá registrar su ruta de manera personalizada como se detalla en la figura 48 el sector por donde se va a desplazar de un punto a otro con el propósito de que los usuarios clientes (pasajeros) puedan buscarlo y hacer uso del servicio. Además, tendrá la posibilidad de visualizar todas las rutas registradas.

Interfaz para que el usuario ´pueda registra las rutas

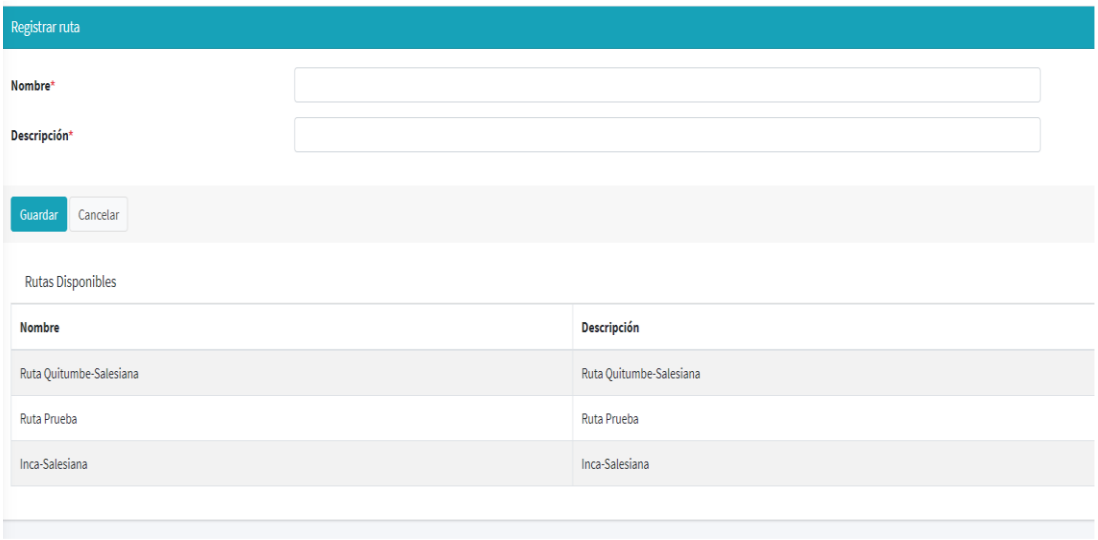


Figura 44.Formulario para registrar y visualizar las rutas
Fuente: Evelyn Benavides & Mayte Tixilema

En la opción Registrar viaje, el usuario conductor se encontrará con el formulario que permitirá crear los puntos pertenecientes a una ruta que han sido creadas previamente. Primero se selecciona la ruta y con la ayuda de un marcador que se encuentra en el mapa, se seleccionan los puntos por los que pasará la ruta, como se puede visualizar en la figura 49.

Interfaz donde el usuario registra el viaje

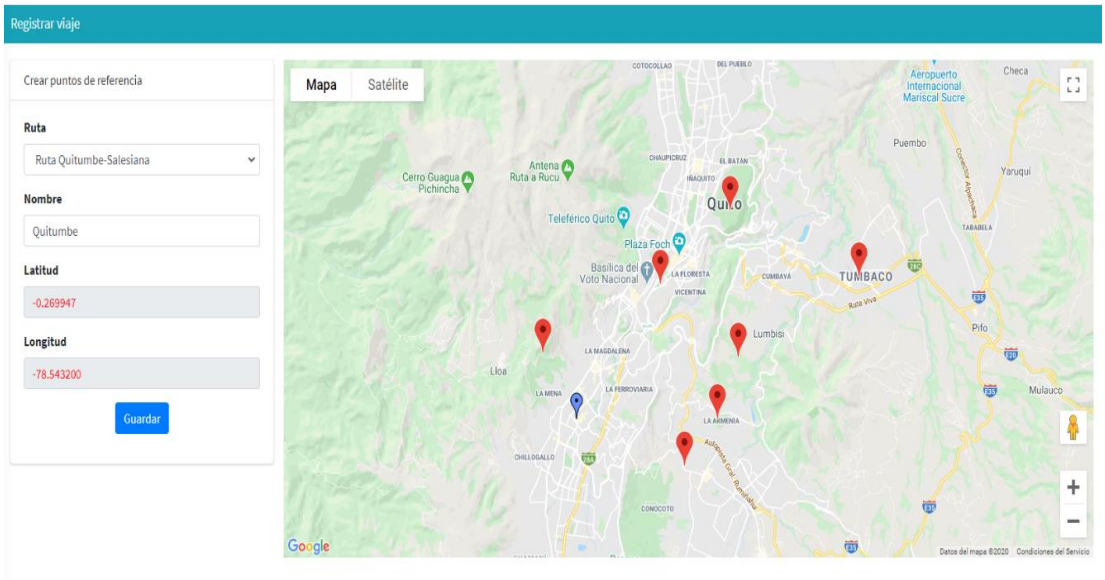


Figura 45.Formulario para registrar viaje
Fuente: Evelyn Benavides & Mayte Tixilema

Después de que el usuario establezca sus rutas, estas se mostrarán a los usuarios que necesiten usar el servicio. Si el usuario envía una solicitud se mostrará una notificación como se puede observar en la figura 50.

Notificación de solicitudes

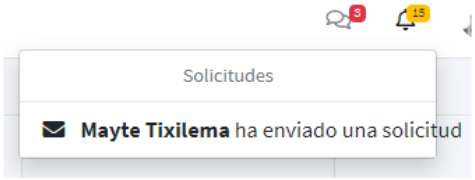


Figura 46.Pantalla de Solicitudes recibidas
Fuente: Evelyn Benavides & Mayte Tixilema

El usuario conductor tendrá disponibles las solicitudes hasta que el decida aceptarlas o rechazarlas, como se puede visualizar en la figura 51. Además, el conductor puede visualizar la información para tomar contacto si llegara a aceptar la solicitud.

Interfaz con la información del usuario para realizar el contacto

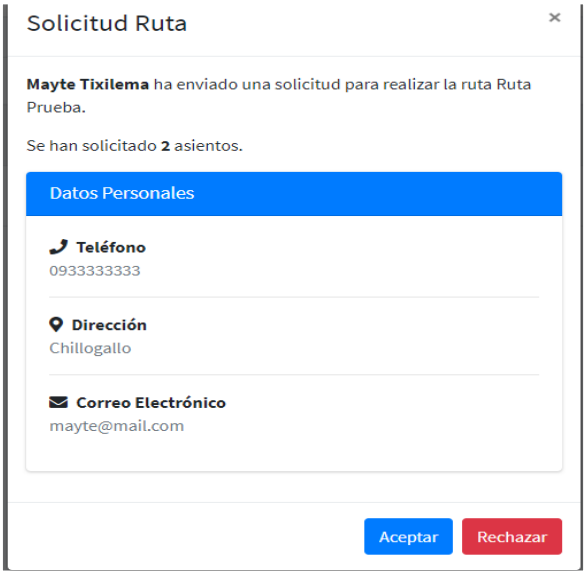


Figura 47.Ventana donde el usuario puede aceptar o rechazar la solicitud
Fuente: Evelyn Benavides & Mayte Tixilema

En la ventana Viajes realizados, el usuario conductor puede visualizar el resgistro de todos los viajes realizados, en donde se encuentra detallado el usuario a quien se prestaron los servicios, la placa del vehículo en el que se movilizaron, la ruta tomada, el número de asientos ocupados y la calificación del usuario cliente(pasajero) como se detalla en la figura 52.

Interfaz donde el usuario puede visualizar su historial de viajes

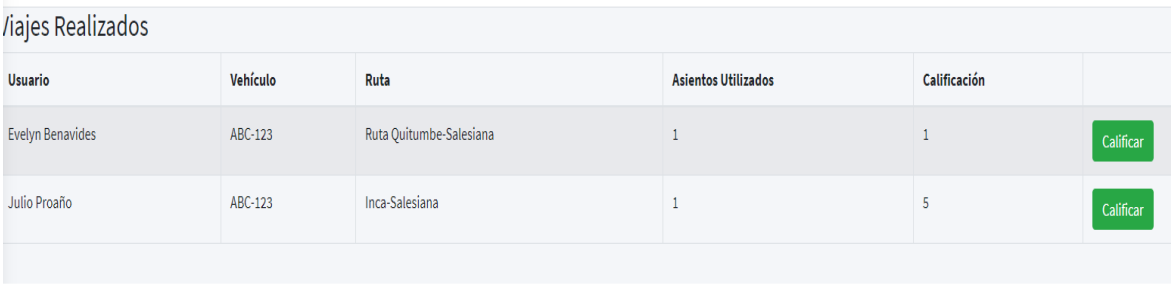


Figura 48.Ventana con los detalles de los viajes realizados
Fuente: Evelyn Benavides & Mayte Tixilema

El usuario conductor tiene la opción de establecer una calificación y comentario a los usuarios pasajeros a los que se les prestó el servicio, como se puede observar en la figura 53.

Interfaz donde el usuario podrá calificar

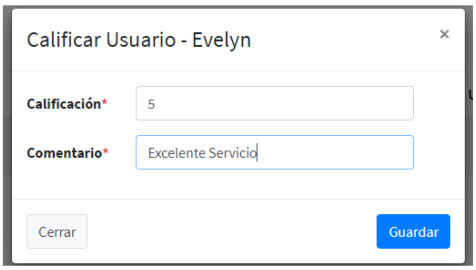


Figura 49.Ventana para realizar la calificación al usuario
Fuente: Evelyn Benavides & Mayte Tixilema

4.7. Desarrollo del módulo de usuario cliente.

El módulo está diseñado con el fin de que los usuarios que se registren con el rol de usuario cliente y tomen contacto con el usuario conductor para hacer uso del servicio en vehículo compartido. Por lo tanto, la aplicación web debe permitir buscar los viajes y verificar si el vehículo está disponible con el fin de enviar una notificación en el que se solicita el viaje. Para lo cual se realiza la especificación de los modelos utilizados, el detalle de los controladores principales, las rutas utilizadas para el acceso y las ventanas diseñadas para que el usuario cliente pueda interactuar con el sistema.

4.7.1. *Especificación de modelos.*

Los modelos creados dentro de este módulo nos van a permitir acceder a las tablas de la base de datos (usuario, contratación, comentario, ruta y punto) e interactuar con la misma, para poder realizar las consultas necesarias y darle funcionalidad al módulo. Como se puede observar en la figura 54, los modelos que se han utilizado son Comentario, Contratación Punto y Ruta.

Modelos del usuario cliente

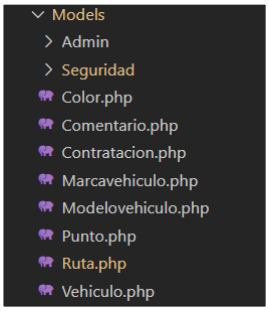


Figura 50. Modelos del módulo usuario
Fuente: Evelyn Benavides & Mayte Tixilema

Los modelos para realizar este módulo ya fueron creados previamente en el módulo del usuario conductor, así como también la estructura de cada modelo que se utilizará a continuación.

4.7.2. *Controladores principales*.

Los controladores que se han utilizado en este módulo nos permitirán comunicarnos con los modelos y acceder a las vistas mediante las rutas. Como se puede observar en la figura 55, se presentan los controladores que se utilizarán en este módulo.

Controladores

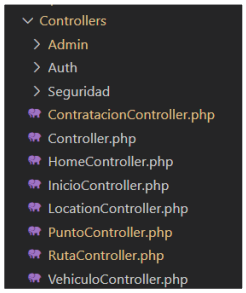


Figura 51. Controladores que utiliza el módulo del usuario conductor
Fuente: Evelyn Benavides & Mayte Tixilema

Los controladores a utilizarse son los mismos que para el módulo del usuario conductor, la diferencia es que en este módulo los métodos cambian, como se detalla a continuación:

Tabla 10. Modelos y Controladores del Módulo usuario cliente

Controlador	Métodos	Descripción
RutaController	buscarRuta(Request $request)	Realiza una búsqueda de rutas asociadas a la referencia ingresada por el usuario
	guardarContratacion (Request $request, $id)	Crea un contrato entre el usuario pasajero y el conductor, para realizar la solicitud de la ruta.
LocationController	index()	Muestra una lista de puntos según una ruta que se mostrarán en el mapa.

ContratacionController	indexPasajero()	Muestra una lista de rutas que el usuario pasajero realizó.
	actualizarPasajero (Request $request, $id)	Establece una calificación y un comentario al usuario conductor

Nota: Detalle de los controladores desarrollado para el correcto funcionamiento del módulo cliente

4.7.3. *Rutas utilizadas para el acceso.*

Las rutas establecidas en el archivo web.php establecen las rutas de acceso que tendrá la aplicación. En la figura 56, se muestran las rutas que se han utilizado para acceder al módulo del usuario pasajero.

Rutas de acceso

```
/*Rutas*/
Route::get('ruta/buscar', 'RutaController@buscarRuta')->name('buscar_ruta')->middleware('auth');
Route::put('ruta/buscar/{id}', 'RutaController@guardarContratacion')->name('guardar_contratacion')->middleware('auth');

/*Contratación*/
Route::put('contratacion/pasajero/{id}', 'ContratacionController@actualizarPasajero')->name('actualizar_contratacionpasajero');
Route::get('contratacion/pasajero', 'ContratacionController@indexpasajero')->name('contratacion_pasajero')->middleware('auth');
```

Figura 52. Rutas de acceso al módulo del usuario pasajero
Fuente: Evelyn Benavides & Mayte Tixilema

4.7.4. *Diseño de interfaces.*

En la ventana principal del usuario cliente (pasajero), se pueden visualizar todos los procesos que este usuario debe ejecutar dentro de la aplicación web, que se encuentran dentro de un menú principal (Ruta, Buscar ruta, Rutas realizadas) como se puede visualizar en la figura 57.

Ventana principal del usuario cliente

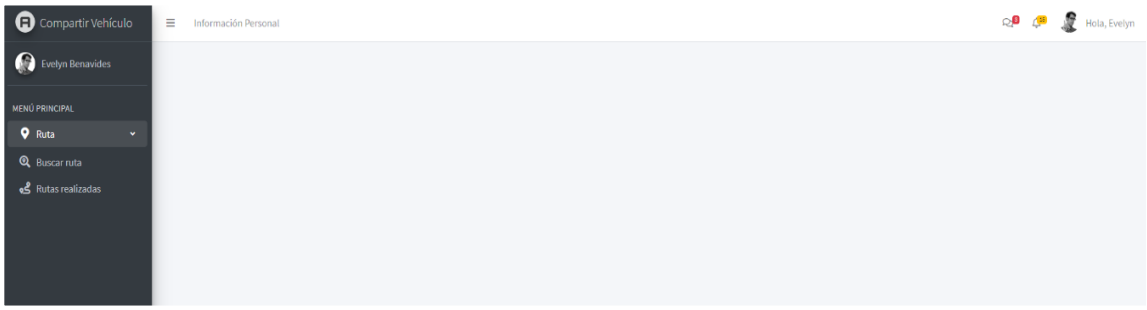


Figura 53.Ventana Principal del Usuario cliente
Fuente: Evelyn Benavides & Mayte Tixilema

En la figura 58, se detalla las rutas asociadas al punto ingresado por el usuario cliente (pasajero), en el cual el usuario puede ingresar el punto de referencia para realizar el viaje y buscar las rutas asociadas al mismo.

Ventana con las rutas asociadas al punto ingresado

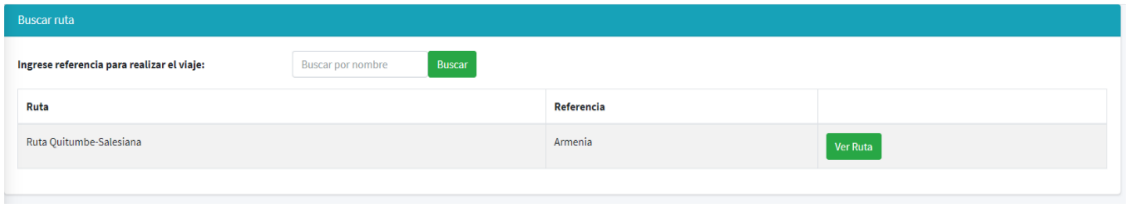


Figura 54. Ventana con las rutas asociadas al punto ingresado
Fuente: Evelyn Benavides & Mayte Tixilema

En la figura 59, se aprecia la ventana en la que el usuario cliente(pasajero) puede visualizar las rutas disponibles en base al punto de referencia colocada y por consiguiente enviar la solicitud al usuario conductor para realizar el viaje en vehículo compartido.

Solicitud de la ruta

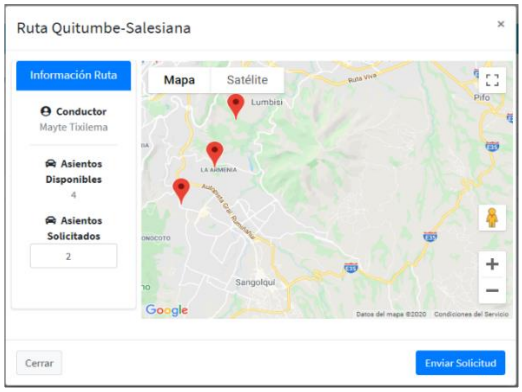


Figura 55. Solicitud de la ruta
Fuente: Evelyn Benavides & Mayte Tixilema

El usuario cliente/pasajero puede visualizar todos los viajes que, a realizados, la placa del vehículo en el que se movilizó, la ruta realizada, el número de asientos solicitados y la calificación que se ha establecido al usuario conductor como se puede apreciar en la figura 60.

Ventana de viajes realizados

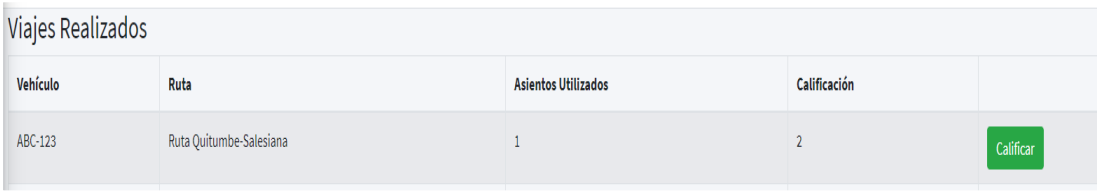


Figura 56. Ventana de viajes realizados
Fuente: Evelyn Benavides & Mayte Tixilema

En la figura 61, se puede apreciar la ventana en la que el usuario cliente(pasajero) puede realizar la calificación al usuario conductor quien ha prestado los servicios.

Ventana para registrar la calificación

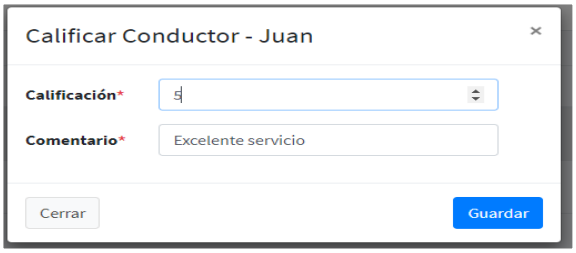


Figura 57.Ventana para registrar la calificación
Fuente: Evelyn Benavides & Mayte Tixilema

El desarrollo de la aplicación web para compartir vehículo, refleja la funcionalidad del sistema para la utilización de los usuarios finales, mediante el proceso de análisis y diseño que se realizó previamente con la utilización de la metodología SCRUM y el establecimiento de objetivos iniciales. El prototipo final se encuentra disponible para realizar todas las pruebas necesarias para comprobar su funcionamiento.

Capítulo 5

Pruebas del Sistema

El objetivo de realizar pruebas a la aplicación web es asegurar el cumplimiento de los requerimientos planteados inicialmente y cubrir las necesidades del cliente, las pruebas evaluarán tanto la parte visual (interfaces con las que interactúa el usuario) como el funcionamiento interno que permite la comunicación entre el sistema y usuario final. En esta etapa se procede a realizar las siguientes pruebas: **Pruebas de rendimiento** con el fin de identificar el tiempo de respuesta de cada proceso interno del sistema, **Pruebas de funcionalidad** con el propósito de asegurarnos que el sistema cumpla con todas las funciones especificadas para cada usuario, **Pruebas de usabilidad** se realiza con el propósito de medir que tan fácil le resulta al usuario final relacionarse con el sistema al ejecutar un proceso.

El desarrollo de los diferentes tipos de pruebas realizadas nos permite ver el estado en que se encuentra el sistema actualmente, identificar y corregir errores no detectados durante la etapa de desarrollo y garantizar el funcionamiento del sistema web.

5.1. Pruebas de rendimiento

Las pruebas de rendimiento se realizan para medir la velocidad, fiabilidad, escalabilidad y estabilidad de la aplicación web con el fin de verificar que tan rápido responde el sistema bajo cualquier condición de trabajo e identificar el tiempo que invierte el sistema para ejecutar una determinada tarea realizada por el usuario. Para realizar estas pruebas se optó por utilizar la herramienta JMeter ya que permite evaluar el rendimiento de la aplicación mediante métricas al someter a máxima carga al servidor de aplicaciones.

Para realizar este tipo de pruebas se creó un ambiente en el que se separó en grupos de usuarios conductores y clientes, asignando las páginas posibles como una petición a las que cada uno tendrá acceso. Se creó una simulación de 69 usuarios por petición en un tiempo de 5 segundos.

Resultados obtenidos de las pruebas de rendimiento - perfil conductor

Etiqueta	# Muestras	Media	Mín	Máx	Desv. Estándar	% Error	Rendimiento	Kb/sec	Sent KB/sec
HTTP Index	50	5013	269	11005	2989,37	0,00%	3,1/sec	29,26	0,36
HTTP login	50	9378	540	12046	3398,33	0,00%	1,9/sec	8,77	0,25
HTTP registro de usuario	50	9659	1911	12073	1938,29	0,00%	1,4/sec	18,70	0,18
HTTP crear vehiculo	50	16718	15342	21637	1668,33	0,00%	1,0/sec	6,13	0,27
HTTP información de vehículo	50	15581	15115	18164	525,78	0,00%	1,1/sec	6,28	0,27
HTTP crear ruta	50	17118	15134	18019	725,18	0,00%	1,1/sec	6,42	0,28
HTTP crear viajes	50	8785	7429	9157	287,32	0,00%	1,3/sec	16,67	0,16
HTTP viajes realizados	50	15979	14339	18085	1037,05	0,00%	1,1/sec	6,31	0,28
Total	400	12279	269	21637	4703,53	0,00%	3,7/sec	29,66	0,71

Figura 58. Pruebas de Rendimiento del usuario conductor
Fuente: Evelyn Benavides & Mayte Tixilema

Resultados obtenidos de las pruebas de rendimiento - perfil cliente

Etiqueta	# Muestras	Media	Mín	Máx	Desv. Estándar	% Error	Rendimiento	Kb/sec	Sent KB/sec
HTTP buscar ruta	50	21303	17813	22504	1146,36	0,00%	56,5/min	5,55	0,24
HTTP Index	50	6038	253	11532	3366,01	0,00%	3,0/sec	28,33	0,35
HTTP login	50	9200	519	11647	2925,48	0,00%	1,9/sec	8,86	0,25
HTTP registro de usuario	50	9560	2312	11646	1656,69	0,00%	1,4/sec	17,54	0,17
HTTP viajes realizados	50	15951	14705	22424	2107,12	0,00%	59,8/min	5,88	0,26
Total	250	12411	253	22504	5983,99	0,00%	3,5/sec	27,16	0,63

Figura 59. Pruebas de Rendimiento del usuario cliente
Fuente: Evelyn Benavides & Mayte Tixilema

Al realizar la evaluación al grupo de usuarios conductores, como se puede observar en la figura 62 el número de muestras es 400, es decir que por cada petición HTTP se enviaron 50 peticiones, no tiene porcentaje de error y su rendimiento tiene un promedio de 3.7 segundos. Para el grupo de usuarios pasajeros, se puede observar en la figura 63 el número de muestras es 250, es decir que por cada petición HTTP se enviaron 50 peticiones, no tiene porcentaje de error y su rendimiento tiene un promedio de 3.5 segundos. Estos resultados quieren decir que el tiempo de carga para cada grupo de usuarios es aceptable y la aplicación no tiene problemas de rendimiento, es decir es estable y no presenta interrupciones al tener varias peticiones.

A continuación, se puede observar de forma gráfica el resultado obtenido para los grupos de usuarios evaluados. En las figuras 64 y 65, se encuentran las visitas al servidor por segundo en donde se pueden observar que las dos gráficas tienen el mismo comportamiento, y los picos representan que el servidor se sometió a un proceso grande, pero es estable. En las figuras 66 y 67 representan los tiempos de respuesta a lo largo del proceso, como se puede observar, cada petición está representada por un color y se puede observar en que segundo entró cada petición y no presentó interrupciones.

Visitas al servidor por segundo-usuario conductor

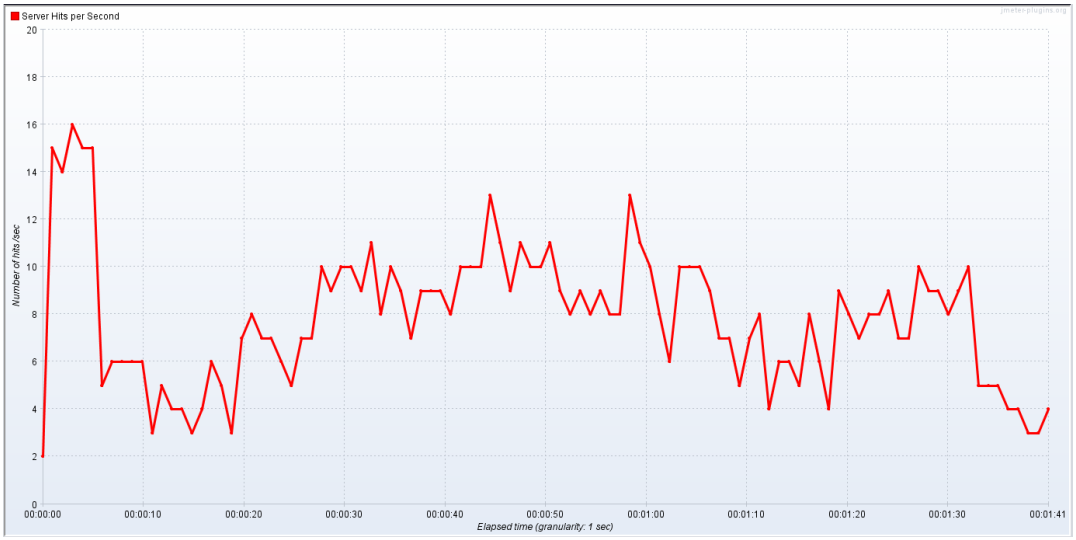


Figura 60. Visitas al servidor del usuario conductor
Fuente: Evelyn Benavides & Mayte Tixilema

Visitas al servidor por segundo-usuario cliente

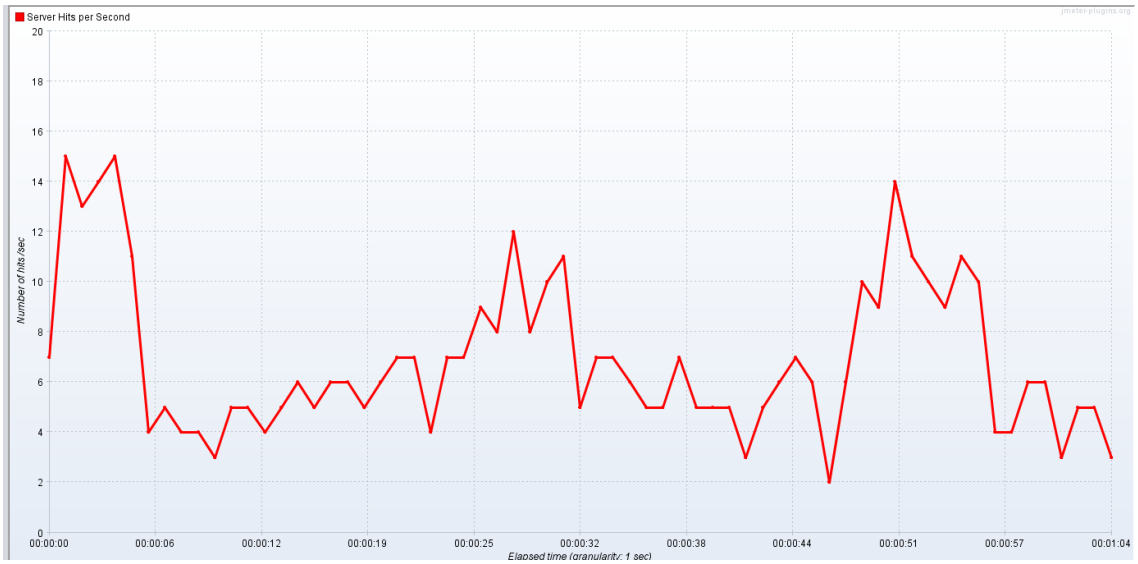


Figura 61. Visitas al servidor del usuario cliente
Fuente: Evelyn Benavides & Mayte Tixilema

Tiempos de respuesta-usuario conductor

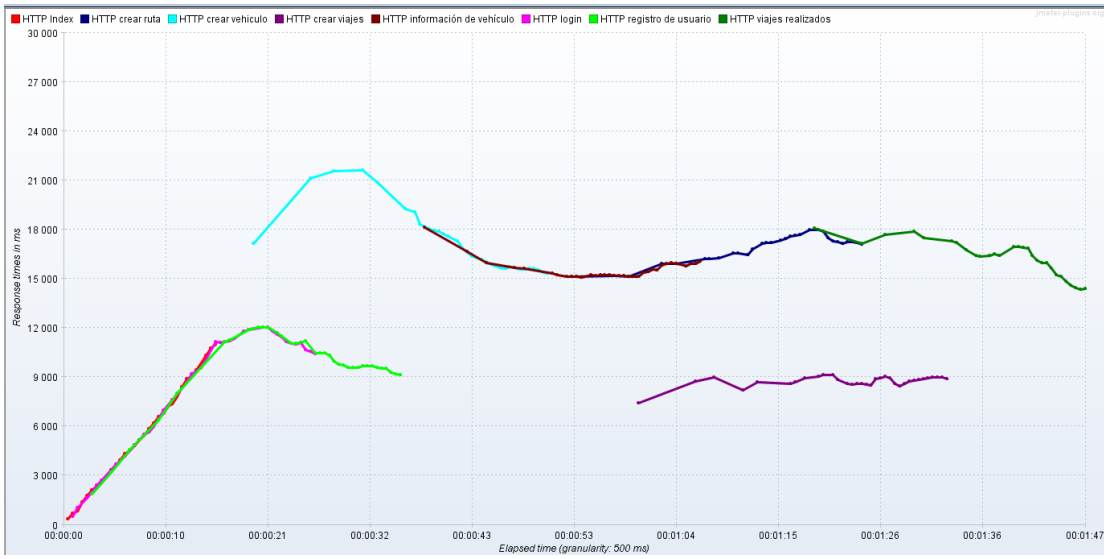


Figura 62. Tiempos de respuesta del usuario conductor
Fuente: Evelyn Benavides & Mayte Tixilema

Tiempos de respuesta-usuario cliente

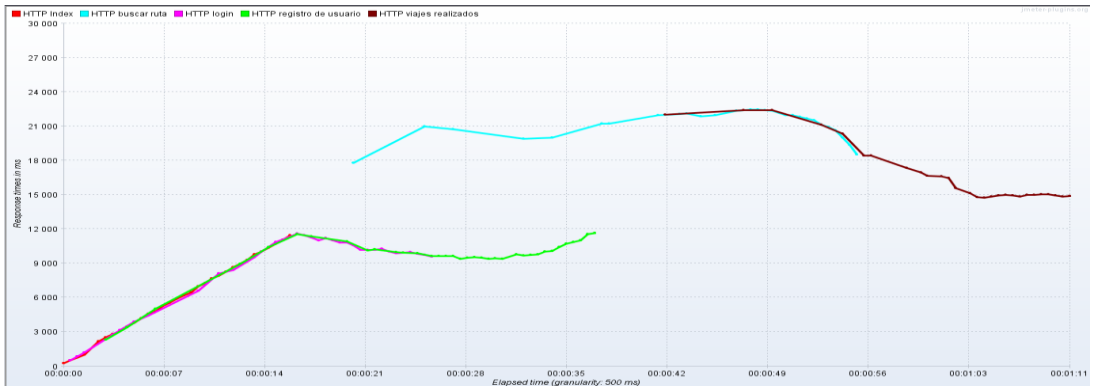


Figura 63. Tiempos de respuesta del usuario cliente
Fuente: Evelyn Benavides & Mayte Tixilema

5.2. Pruebas de funcionalidad

Para ejecutar este tipo de pruebas sobre la aplicación web se ha tomado como referencia el análisis de los requerimientos planteados por módulos antes de su desarrollo, en el que se especifica la funcionalidad del sistema dependiendo del rol del usuario (conductor o pasajero), de este modo se pretende validar si el sistema es funcional para el usuario final. Para la evaluación se tomó en consideración un grupo de 15 usuarios de los cuales 10

realizaron la evaluación como usuarios clientes(pasajeros) y 5 como usuarios conductores. Los resultados de la calificación, será un porcentaje (entre 0 y 100%) para cada tarea establecidas en la tabla 24 y tabla 25 y a partir de este resultado se considerarán como pruebas satisfactorias las tareas que obtengan un porcentaje entre 80 y 100%.

Módulo Conductor

Tabla 11. Prueba funcional modulo usuario conductor

Tareas para ejecutarse por el usuario cliente/pasajero	Calificación obtenida (%)	Satisfacción de la prueba		Fecha de la prueba	Observación de las tareas
		SI	NO		
Registrar usuario con el perfil de conductor.	100%	*		29/06/2020	Ninguna
Ingresar al sistema y visualizar el menú del usuario conductor	100%	*		29/06/2020	Ninguna
Registrar el vehículo perteneciente al conductor.	80%	*		29/06/2020	Ninguna
Registrar y visualizar las rutas estableciendo una hora y asientos disponibles.	80%	*		29/06/2020	Ninguna
Crear un viaje mediante la selección de ruta y puntos pertenecientes a la misma.	90%	*		29/06/2020	Explicar cómo se tiene que marcar los puntos en el mapa.
Visualizar solicitudes enviadas por clientes (pasajeros) que deseen usar el servicio, y comentarios realizados por usuarios que	100%	*		29/06/2020	Ninguna

utilizaron el servicio					
Confirmar o rechazar solicitudes, en donde se podrá poner en contacto con el cliente (pasajero) y ver su calificación.	80%	*		29/06/2020	Ninguna
Visualizar viajes realizados.	100%	*		29/06/2020	Ninguna
Establecer una calificación y comentario al usuario cliente.	80%	*		29/06/2020	Ninguna

Nota: Resumen de la encuesta realizada con el fin de verificar si el módulo conductor es totalmente funcional

Módulo Cliente/Pasajero

Tabla 12. Prueba funcional Modulo Usuario Cliente/Pasajero

Tareas para ejecutarse por el usuario cliente/pasajero	Satisfacción de la prueba		Calificación obtenida	Fecha de la prueba	Observación de las tareas
	SI	NO			
Registrar usuario con el perfil de pasajero.	*		100%	29/06/2020	Ninguna
Ingresar al sistema y visualizar el menú del usuario pasajero	*		100%	29/06/2020	Ninguna
Buscar rutas asociadas al punto de referencia ingresado por el usuario.	*		80%	29/06/2020	Ninguna
Visualizar en el mapa las rutas disponibles en base al punto de referencia colocada y solicitar el viaje.	*		90%	29/06/2020	Ninguna
Visualizar los viajes realizados.	*		90%	29/06/2020	Ninguna
Establecer una calificación y comentario al usuario conductor.	*		80%	29/06/2020	Ninguna

Nota: Resumen de la encuesta realizada con el fin de verificar si el módulo pasajero es totalmente funcional

Para realizar la evaluación se ha elaborado un cuestionario que se encuentra en el Anexo 7. Para obtener el resultado final se ha realizado un promedio en el que se ha obtenido un resultado de 88.9% dentro del módulo del usuario cliente (pasajero) y un 90% en el módulo del usuario conductor, con lo que podemos decir que la aplicación se encuentra funcional.

5.3. Pruebas de usabilidad

Con el fin de mejor la experiencia del usuario final se procede a realizar las pruebas de usabilidad de la aplicación web. Ya que de esta manera se puede comprobar el nivel de aceptabilidad y la facilidad con la que los usuarios pueden interactuar con el sistema. Para lo cual se toma como caso de estudio a 15 usuarios de los cuales 9 usuario son clientes/pasajeros y 6 usuario conductor a lo que se les va a realizar una encuesta.

Pasajero

Uso de la aplicación web -usuario conductor

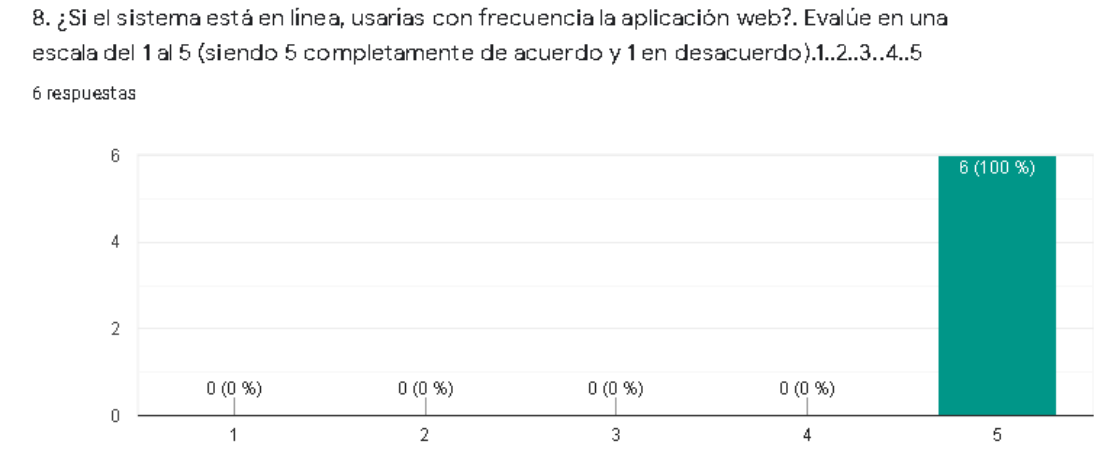


Figura 64. Uso de la aplicación web
Fuente: Evelyn Benavides & Mayte Tixilema

De los 6 usuarios con el Rol conductor encuestados, el 100% de la población harían uso de la aplicación web si esta estuviese expuesta en línea.

Complejidad de la aplicación web -usuario conductor

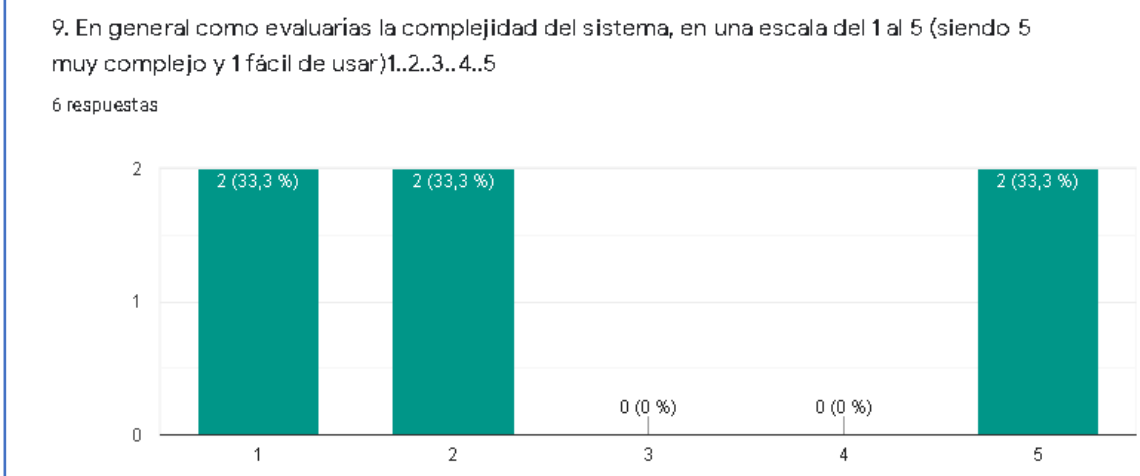


Figura 65. Complejidad de la aplicación web
Fuente: Evelyn Benavides & Mayte Tixilema

Con respecto a la complejidad de la aplicación web el 66,6% (4) de los 6 usuarios encuetados, resaltan que es fácil de usar, mientras que el 33,3% (2) asumen que es muy complejo de utilizarlo.

Diseño de la aplicación web -usuario conductor

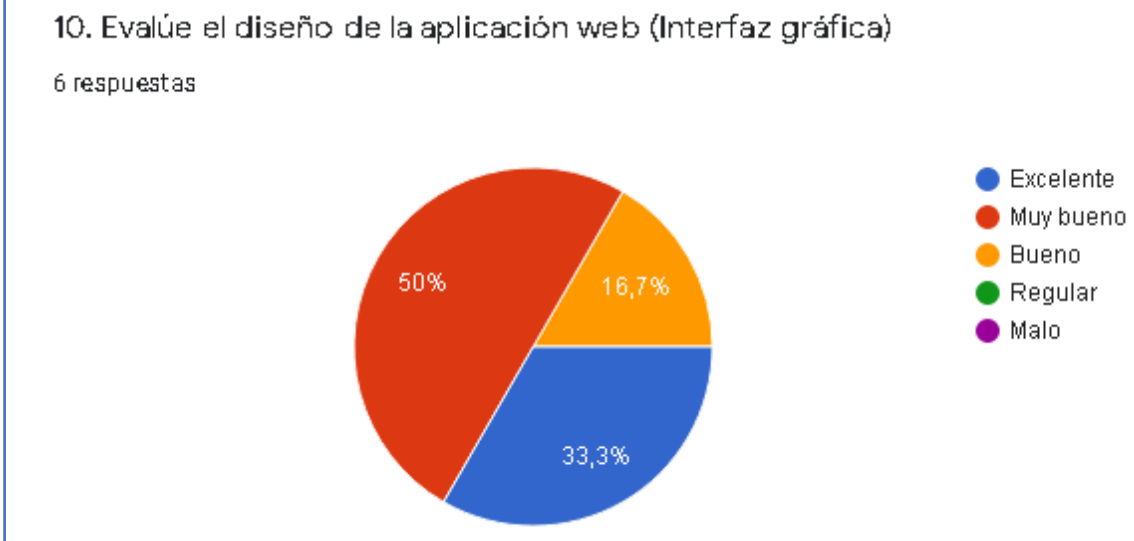


Figura 66. Diseño de la aplicación web
Fuente: Evelyn Benavides & Mayte Tixilema

Según los porcentajes visualizados en la gráfica de los 6 usuario encuestados al 50% (3) le parece que el diseño de la aplicación es Muy bueno, al 16,7% (1) le parece que el diseño es Bueno, mientras que el 33,3% (2) asumen que el diseño de la aplicación web es Excelente.

Manejo de la aplicación web -usuario conductor

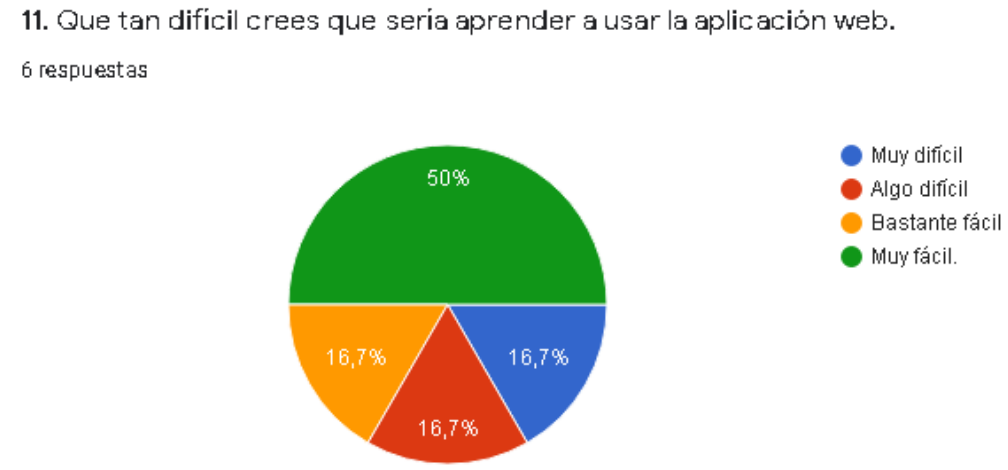


Figura 67. Manejo de la aplicación web
Fuente: Evelyn Benavides & Mayte Tixilema

En cuanto a la usabilidad del sistema, de los 6 usuarios Conductores encuestados el 66,7% asumen que el sistema es fácil de utilizar, esto quiere decir que tenemos aceptación en cuanto al aprendizaje de la aplicación y el resto (33,3%) consideran que es difícil su utilización.

Recomendación de la aplicación web -usuario conductor

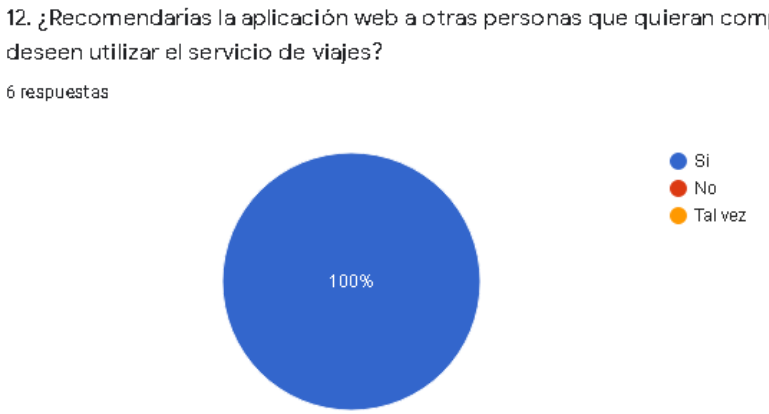


Figura 68. Manejo de la aplicación web
Fuente: Evelyn Benavides & Mayte Tixilema

Del total de 9 usuarios clientes/pasajero encuestados el 33.3% (4), mencionan que si la aplicación web estuviese en producción harían uso del mismo de manera regular, mientras que el 66,7%(5) aseguran que harían uso del servicio frecuentemente.

Del total de la población encuestada el 100% (6), aseguran que si recomendasen el uso de la aplicación web a terceros para que hagan uso del servicio.

Cliente

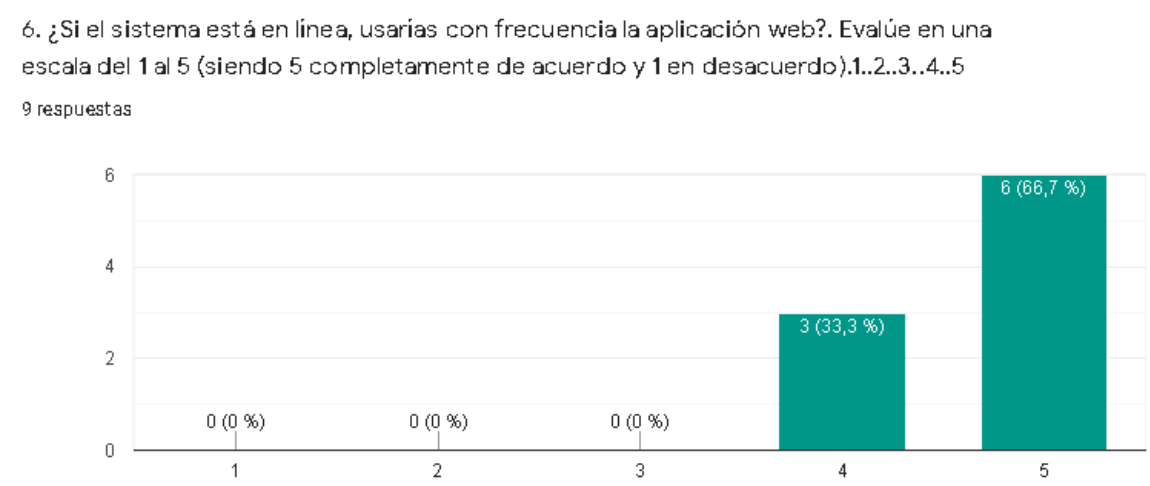


Figura 69. Uso de la aplicación web
Fuente: Evelyn Benavides & Mayte Tixilema

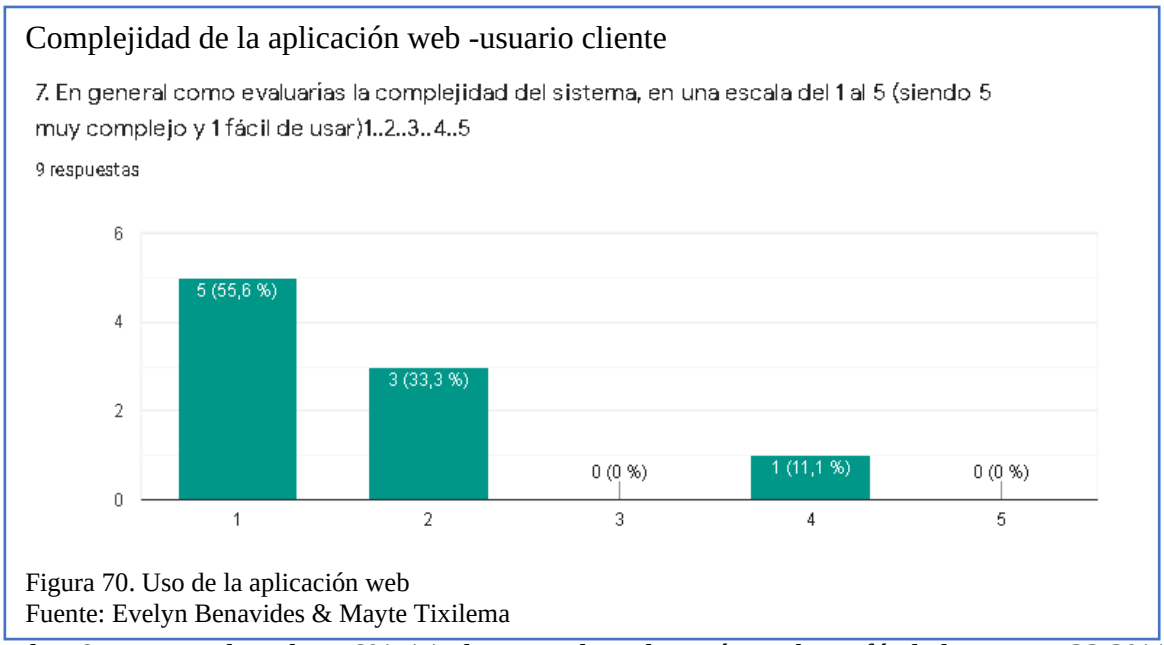


Figura 70. Uso de la aplicación web
Fuente: Evelyn Benavides & Mayte Tixilema

De los 9 encuestados el 55.6% (5) dice que la aplicación web es fácil de usar y 33.3%(3) asumen que la complejidad es regular mientras que para el 11.1%(1) la aplicación web les parece que es demasiado complejo.

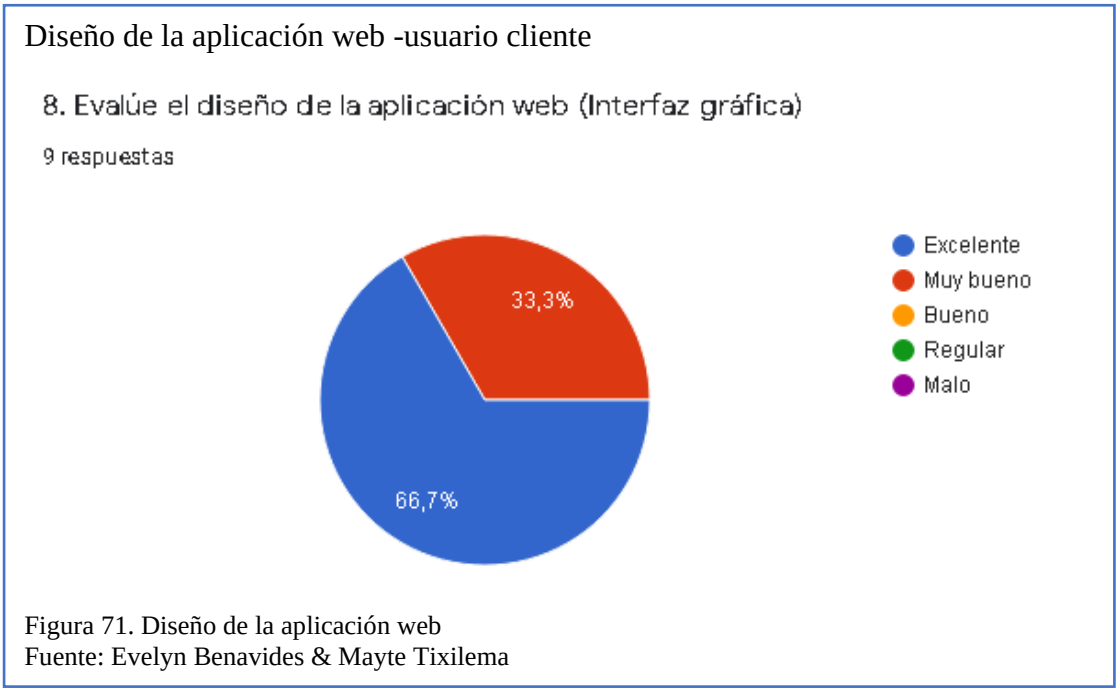


Figura 71. Diseño de la aplicación web
Fuente: Evelyn Benavides & Mayte Tixilema

El 66.7% (6) de la población encuestada dice que el diseño de la aplicación web es Excelente, mientras que para el 33.3% (3) el diseño es Muy bueno.

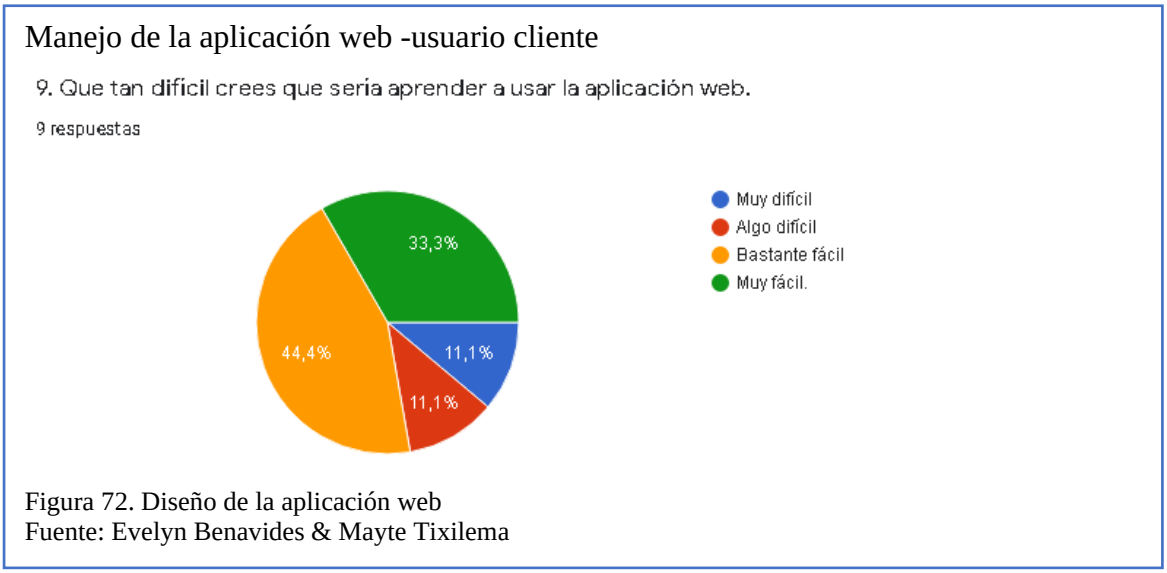


Figura 72. Diseño de la aplicación web
Fuente: Evelyn Benavides & Mayte Tixilema

Como se detalla en la Figura 77 el 44,4%(4) de la población encuestada cree que es bastante fácil de aprender a manejar la aplicación web y el 33,3%(3) dice que la aplicación es muy fácil.

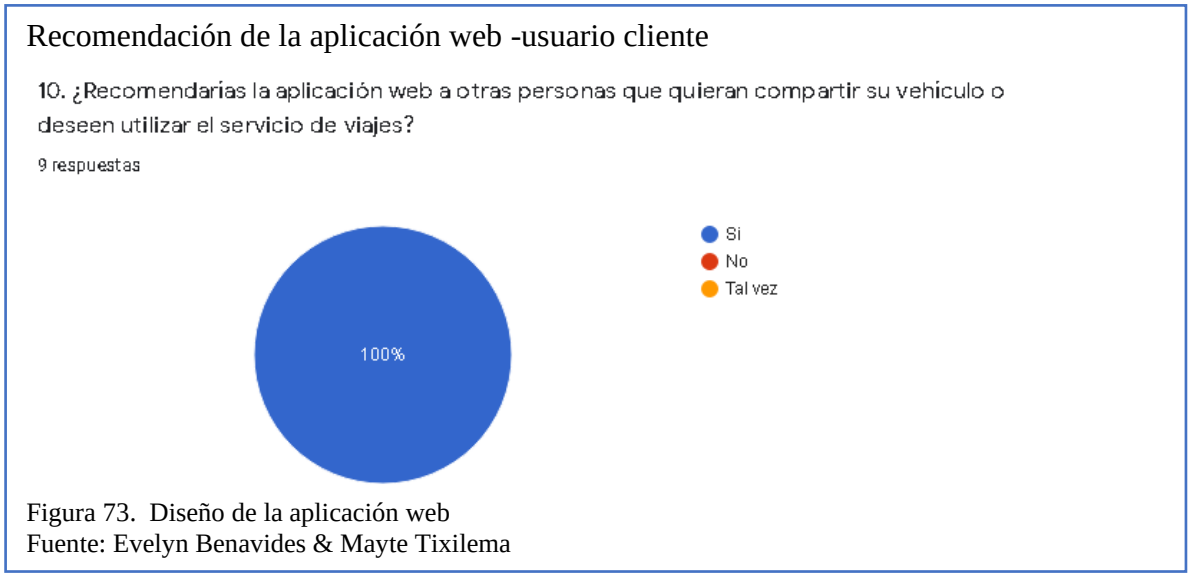


Figura 73. Diseño de la aplicación web
Fuente: Evelyn Benavides & Mayte Tixilema

El 100% (9) de la población encuestada todos recomendaría el uso de la aplicación web para que otras personas hagan uso del servicio.

El resultado de la encuesta realizada nos muestra que la aplicación web es usable debido a que los resultados son en la mayoría favorables.

CONCLUSIONES

- Con el objetivo de lograr mayor escalabilidad, velocidad y seguridad en la aplicación web, se dio uso del Framework de PHP Laravel, el mismo que nos permitió realizar una aplicación estable, lo que garantiza que el sistema puede crecer en versiones futuras.
- La etapa de análisis dentro del desarrollo del proyecto ha sido fundamental ya que el establecer los requisitos necesarios y realizar un análisis adecuado nos permitió evitar retrasos grandes en el proyecto y errores que no se podrían resolver.
- El resultado final del proyecto técnico es una aplicación web que cumple con el planteamiento de requisitos iniciales, en el que un conductor puede realizar un registro de viajes y ofertar sus servicios que estarán disponibles para los clientes que encontrarán una alternativa diferente para trasladarse de un lugar a otro.
- El uso de la metodología SCRUM nos permitió realizar el desarrollo de la aplicación web de manera organizada, mediante el planteamiento de requisitos y planificación de tareas que nos ayudaron realizar entregas del producto final en fechas aproximadas de las planteadas inicialmente.
- El análisis de resultados nos indica que el desarrollo del prototipo de la aplicación web cumplió con los objetivos iniciales, mediante los resultados podemos observar que tiene un rendimiento de procesos óptimo, es funcional y usable para el usuario final.

RECOMENDACIONES

- La utilización de una metodología ágil nos da varias ventajas, pero no siempre se puede aplicar totalmente, por esa razón es necesario que dependiendo del proyecto y las circunstancias se tomen decisiones con respecto a los elementos que se deben utilizar.
- El cálculo del tiempo para el desarrollo de cada sprint puede no ser exacto, por lo que se recomienda dar uso de tiempos aproximados establecidos para cada tarea, lo que evitará retrasos en el proyecto, ya que al iniciar con el desarrollo no se cuenta con los conocimientos necesarios para cumplir con la tarea en el tiempo establecido.
- Para una mejor visualización de la interfaz gráfica es recomendable usar los navegadores Google Chrome y Firefox ya que van a permitir que los componentes se adapten de mejor manera para que el usuario tenga una mejor experiencia.
- Para la utilización del API de Google es importante revisar la documentación que proporciona Google, ya que esto nos ahorrará tiempo durante su implementación

LISTA DE REFERENCIAS

Artículos Académicos

Dimes, T. (2015). Conceptos Básicos de Scrum: Desarrollo de software Agile y manejo de proyectos Agile. Babelcube Inc.

Marcuello Servós, C., Monreal Garrido, M., & Díaz Fonseca , M. (2016). Economía social y economía colaborativa: Encaje y potencialidades. 27-35. Obtenido de https://zaguan.unizar.es/record/69619/files/texto_completo.pdf

Marini, I. E. (2012). El Modelo Cliente/Servidor. Obtenido de https://www.linuxito.com/docs/el-modelo-cliente-servidor.pdf

Bibliografía

Beltrán, G. (2016). Geolocalización online: la importancia del dónde. Barcelona: UOC.

Berzal, F., Cortijo, F. J., & Cubero, J. C. (2005). Desarrollo profesional de aplicaciones web con ASP:NET. iKor Consulting.

Capote, P., Ruiz, M., Rodriguez, M., Carrillo, A., & Miranda, V. (2009). Introduccion a las base de datos: el modelo relacional. España : Thomson: Editors Spain.

Carrillo, S., Ruiz, N., Rodríguez, J., & Capote, O. &. (2005). Introducción a las bases de datos: El modelo relacional. Paraninfo.

Laínez Fuentes, J. R. (2015). Desarrollo de Software ÁGIL: Extreme Programming y Scrum. IT Campus Academy.

Luján, S. (2002). Programacion de aplicaciones web:historia,principios basicos y clientes web. España: Club Universitario.

Márquez Avendaño, B. M., & Zulaica Rugarcía, J. M. (2004). Implementación de un reconocedor de voz gratuito a el sistema de ayuda a invidentes Dos-Vox en español. Puebla, México: Universidad de las Américas Puebla

Nieto, R. J. (2016). Desarrollo de una aplicación web, con Front-end y Back-end, para compraventa de segunda mano. Gandía, España: Universidad Politécnica de Valencia.

Piñeiro Gómez, J. M. (2014). Diseño de bases de datos relacionales. España: Paraninfo,S.A.

Silberschatz, A., Korth, H., & Sudarshan, S. (2002). Fundamentos de bases de datos. España: Concepción Fernández Madrid.

Silva Toledo, F. E. (2018). Implementación de servidor web Y DNS para la empresa GIGOWIRELESS para proveerel servicio de internet a la ciudad de Quito. Quito, Ecuador: Escuela Politécnica Nacional.

Sutherland, J. (2016). Scrum: El arte de hacer el doble de trabajo en la mitad de tiempo. . Océano.

The Blockhead. (2016). Scrum - ¡Guía definitiva de prácticas ágiles esenciales de Scrum! Babelcube inc.

Villada Romero, J. L. (2014). Instalación y configuración del software de servidor Web. IFCT0509. Málaga, España: IC Editorial.

Imágenes

Benavides, S. (15 de 05 de 2013). https://comunidad.iebschool.com/. Obtenido de https://comunidad.iebschool.com/scrumenaccion/2013/05/15/empezando-con-scrum/

Google. (2020). Google. Obtenido de https://developers.google.com/maps/documentation/javascript/get-api-key?hl=es

Sitios Web

ANETA. (2019, 02 01). Retrieved from http://www.aneta.org.ec/wordpress/en-quito-los-vehiculos-son-los-que-mas-contaminan-el-aire/

Alfonso Sánchez, R. (2016). Economía colaborativa: un nuevo mercado para la economía socal. CIRIEC-España, 231-258. Obtenido de https://www.redalyc.org/pdf/174/17449696008.pdf

amovens. (2009). Obtenido de amovens.com: https://amovens.com/
blog.blablacar. (2019). blog.blablacar.es. Obtenido de https://blog.blablacar.es/blablalife/novedades/blablacar-3-millones-usuarios-espana
Carvajal, A. M. (21 de 02 de 2019). Obtenido de https://www.elcomercio.com/actualidad/congestion-vehicular-ranking-movilidad-amt.html
Coronel, C., Morris, S., & Rob, P. (2011). Base de datos: diseño, implementación y administración. Mexico: Raúl D.Zendejas Espejel.
developers. (07 de 05 de 2020). Obtenido de https://developers.google.com/maps/documentation/javascript/tutorial
Díaz González, Y., & Fernández Romero, Y. (2012). Patrón Modelo-Vista-Controlador. Revista digital de las tecnologías de la información y las comunicaciones, 11(1), 47-57. Obtenido de http://revistatelematica.cujae.edu.cu/index.php/tele/article/view/15/10
ecologistasenaccion. (2019). ecologistasenaccion.org. Obtenido de https://www.ecologistasenaccion.org/quienes-somos/programa-ambiental-de-ecologistas-en-accion/
elblogalternativo. (22 de 06 de 2011). elblogalternativo.com. Obtenido de https://www.elblogalternativo.com/2011/06/22/conduzco-es-plataforma-de-coche-compartido-en-espana-y-europa-entrevistamos-a-angel-gonzalez-responsable-de-comunicacion/
Gómez, R. (2015). Modelo Vista Controlador. Obtenido de http://rodrigogr.com/blog/modelo-vista-controlador/
hoylosAngeles. (23 de 09 de 2016). Hoy los Angeles. Obtenido de https://www.hoylosangeles.com/vidayestilo/autos/hoyla-aut-cuantos-autos-hay-en-el-mundo-y-cuantos-se-fabrican-anualmente-20160923-story.html
Lopéz Padilla, A. (2013). Guía de seguridad en servicios DNS. INTECO (Instituto Nacional de Tecnologías de la Comunicación), 6-12. Obtenido de https://www.incibe-cert.es/sites/default/files/contenidos/guias/doc/guia_de_seguridad_en_servicios_dns.pdf
Martín, J. (2018). ingresosviaweb. Obtenido de https://www.ingresosviaweb.com/generar-api-key-de-google-maps/
Microsoft. (2016). Microsoft Office. Conceptos básicos del diseño de una base de datos. Obtenido de https://support.office.com/es-es/article/conceptos-b%C3%A1sicos-del-dise%C3%B1o-de-una-base-de-datos-eb2159cf-1e30-401a-8084-bd4f9c9ca1f5
Mozilla Developer. (18 de 03 de 2019). MDN web docs. Obtenido de https://developer.mozilla.org/es/docs/Web/HTTP/Overview
Olcina, A. (2017). 20aplicaciones. Obtenido de https://riunet.upv.es/bitstream/handle/10251/91213/OLCINA%20-%20Desarrollo%20de%20aplicaciones%20web%20con%20el%20API%20de%20Google%20Cloud.pdf?sequence=1&isAllowed=y
O'Reilly, T. (30 de 09 de 2006). Qué es Web 2.0. Patrones del diseño y modelos del negocio para la siguiente generación del software. Boletín de la Sociedad de la Información. Obtenido de https://f72a491e-a-ef8ef19a-s-sites.googlegroups.com/a/espacio3i.com/cibercultura_publico/modulo-1---el-contexto-de-la-cibercultura/2-historiadeinternet/Qu%C3%A9%20es%20Web%202.pdf?attachauth=ANoY7cq2SV-dsO1UTSGLn7QG5x7SU7kmstrtc1EJ7AuUMBbOne0kiuLU9D-jkOd
Padial, J. (15 de 02 de 2015). cybmeta. Obtenido de https://cybmeta.com/introduccion-a-geolocation-api-de-javascript

Peon, R., & Ruellan, H. (2015). HPACK: Header Compression for HTTP/2. Obtenido de https://www.hjp.at/doc/rfc/rfc7541.html
Perez, M. (30 de 11 de 2019). superadmin. Obtenido de https://superadmin.es/blog/que-es/direccion-web-url/
Riera. (2004). viajamosjuntos.com. Obtenido de https://www.viajamosjuntos.com/nosotros.jsp
Rivera López, A. (16 de 01 de 2008). Sistema asistente para la generación de horarios de cursos . Obtenido de Colección de Tesis Digitales: http://catarina.udlap.mx/u_dl_a/tales/documentos/lis/rivera_l_a/
Rouse, M. (08 de 05 de 2018). searchdatacenter. Obtenido de https://searchdatacenter.techtarget.com/es/definicion/MySQL
Ticona Yanqui, F. E. (2018.). http://repositorio.uancv.edu.pe/. Obtenido de http://repositorio.uancv.edu.pe/bitstream/handle/UANCV/1894/PROYECTO%20DE%20INVESTIGACI%c3%93N.pdf?sequence=3&isAllowed=y
Velasque, R. (2013). codehero. Obtenido de http://codehero.co/laravel-4-desde-cero-instalacion-configuracion/.
Sandoval, M., & Heriberto, H. (2018). Desarrollo de un sistema de geolocalización web de rutas de transporte para los estudiantes de la EPN (Bachelor's thesis, Quito, 2018.).
Cuajimalpa. (2018). neoattack. Retrieved from https://neoattack.com/neowiki/google-maps/
Robles, V. (2019). profesores. Retrieved from https://www.youtube.com/watch?v=laTPz49TgZE

Tesis / trabajos de titulación

Arroyo, W. (2019). dspace. Obtenido de http://dspace.espoch.edu.ec/bitstream/123456789/12243/1/18T00788.pdf
Colla, P. (2012). 41jaiio.sadio.org.ar. Obtenido de http://41jaiio.sadio.org.ar/sites/default/files/086_ASSE_2012.pdf
Espinoza, L., & Vera, C. (2017). repositorio. Obtenido de http://repositorio.ug.edu.ec/bitstream/redug/19964/1/B-CISC-PTG.1284%20Ayala%20Espinoza%20Leslye%20Yanina.pdf
Gaitán González, L. M. (2017). Uso de la metodología SCRUM en el desarrollo de aplicaciones web . Guatemala: Doctoral dissertation.
Huebla, B. (2018). dspace.espoch.edu.e. Obtenido de http://dspace.espoch.edu.ec/bitstream/123456789/9115/1/18T00744.pdf
Malan, F. (2016). dspace. Obtenido de http://dspace.espoch.edu.ec/bitstream/123456789/6235/1/18T00652.pdf