PROYECTO DE GRADO

Presentado ante la ilustre UNIVERSIDAD DE LOS ANDES como requisito final para obtener el Título de INGENIERO DE SISTEMAS

Desarrollo de una aplicación Web que permita organizar viajes interurbanos en Venezuela bajo el modelo *Carpooling*

Por

Br. Lenin Roa
Tutor: Prof. Isabel Besembel

Noviembre 2017

UNIVERSIDAD DE LOS ANDES
MERIDA VENEZUELA

Desarrollo de una aplicación Web que permita organizar viajes interurbanos en Venezuela bajo el modelo *Carpooling*

Br. Lenin Roa

Proyecto de Grado — Sistemas Computacionales, 67 páginas
Escuela de Ingeniería de Sistemas, Universidad de Los Andes, 2017.

Resumen: Los medios de transporte públicos juegan un papel muy importante en cualquier sociedad. Actualmente, hacer uso del transporte interurbano en Venezuela resulta problemático para la mayoría de las personas debido a distintas razones. La baja concurrencia de viajes causada por variados factores no permite que la demanda del servicio sea cubierta en su totalidad. El presente trabajo, tiene como objetivo principal el desarrollo de una aplicación web que permite poner en contacto a personas que viajan a una misma ciudad destino, desde una misma ciudad origen, para que ambas partes puedan beneficiarse haciendo uso del modelo *Carpooling*. El *Carpooling* es la práctica que permite a distintas personas compartir un vehículo cuando viajan desde y hacia un mismo lugar. La aplicación web permite a los usuarios registrarse y autenticarse para posteriormente publicar y buscar viajes en fechas y con características específicas, realizar preguntas a los usuarios conductores, así como también calificarlos. Dicha aplicación fue elaborada haciendo uso de Laravel, un potente *framework* de PHP y siguiendo el método Blue Watch. En la búsqueda de cumplir el objetivo general se cumplieron, a su vez, los objetivos específicos satisfactoriamente. Desde el análisis previo al desarrollo del sistema hasta la elaboración completa de la aplicación web y la realización pruebas funcionales.

Palabras claves: Aplicación Web, Base de Datos, MySQL, Uso compartido de vehículos, Blue Watch.

Dedicatoria

A mis padres, Mercedes Dávila y Lenin Roa. Por todo el apoyo brindado a lo largo de mi formación académica. Por ser excelentes padres y guías. Por todo el esfuerzo, sacrificio y amor brindado; pensando siempre en pro de mí beneficio sin nunca pedir nada a cambio.

A Daniela Altuve, por creer en mi siempre y ser mi principal fuente de motivación e inspiración.

A ustedes tres, mis más sinceras gracias.

Índice

Capítulo 1 6
1.1 Planteamiento del problema 6
1.2 Antecedentes 7
1.3 Objetivos del trabajo 9
1.3.1 Objetivo general 9
1.3.1 Objetivos específicos 9
1.4 Metodología 10
1.5 Resumen capítulo 1 11
Capítulo 2 12
2.1 Marco teórico 12
2.1.1 Carpooling 12
2.1.2 Funcionamiento del *Carpooling* 13
2.1.3 Ventajas 13
2.1.4 Desventajas 14
2.1.5 *Carpooling* en Venezuela 14
2.1.6 Aplicaciones *ride matching* 15
2.1.7 Aplicación web 15
2.1.8 Base de datos 16
2.1.9 Sistema Manejador de Base de Datos (DBMS) 16
2.1.10 HTML 17
2.1.11 CSS 17
2.1.12 PHP 18
2.1.13 Laravel 18
2.1.14 UML 19
2.2 Resumen capítulo 2 20
Capítulo 3 21
3.1 Análisis de contexto y desarrollo inicial de requisitos 21
3.2 Modelo del Sistema de Negocio 21
3.2.1 Diagrama de jerarquía de sistemas 22

3.2.2 Diagrama de Objetivos....22
3.3 Ingeniería de requisitos....23
3.3.1 Listado de requisitos....23
3.3.2 Descripción de las reglas del negocio....29
3.3.3 Descripción de actores y sus roles....29
3.3.4 Matriz de actores vs. procesos....30
3.4 Resumen capítulo 3....32
Capítulo 4....33
4.1 Diseño arquitectónico....33
4.1.1 Listado descriptivo de las metas de diseño....33
4.1.2 Estructura de la aplicación....34
4.1.3 Diagrama de clases....35
4.1.4 Diagramas de caso uso....36
4.1.5 Planificación de las iteraciones....37
4.2 Resumen capítulo 4....38
Capítulo 5....40
5.1 Primera Iteración....40
5.1.1 Refinamiento de los requisitos....40
5.2 Segunda Iteración....46
5.2.1 Refinamiento de los requisitos....46
5.3 Tercera Iteración....49
5.3.1 Refinamiento de los requisitos....49
5.4 Implementación....52
5.5 Resumen capítulo 5....61
Capítulo 6....62
6.1 Conclusiones....62
6.2 Resultados....64
6.3 Recomendaciones....65

Capítulo 1

En este primer capítulo se explica la problemática que justificó el desarrollo de este trabajo de investigación. Se expone el objetivo general y los objetivos específicos que se buscaron alcanzar. Se introduce al lector la metodología que se utilizó (Blue WATCH).

1.1 Planteamiento del problema

Para la sociedad venezolana, el desplazamiento interurbano en lugar de ser una actividad cotidiana es una actividad que se realiza cada vez con mayor dificultad debido a numerosos factores. Se estima que gran cantidad de la flota de autobuses usados para este tipo de transporte está paralizada por falta de repuestos que van desde la necesidad de un caucho, una batería, o incluso hasta el cambio completo del motor.

De igual manera, es muy común que los autobuses fallen mientras prestan el servicio, obligando a los usuarios a esperar un tiempo prolongado mientras se repara dicha falla. En ocasiones es necesario que otro autobús socorra a los pasajeros transbordándolos a este segundo para poder terminar el recorrido.

Como consecuencia de esto, el número de viajes diarios que se realizan de una ciudad a otra ha disminuido notoriamente (Migdalis, 2014), imposibilitando el desplazamiento de la totalidad de las personas que buscan trasladarse a cierto destino en un día específico. Asimismo, el constante aumento del costo de los repuestos, servicios de mantenimiento y reparación de automóviles en nuestro país deriva en el aumento del pasaje de los viajes tanto urbanos como interurbanos.

Adicionalmente, los pasajeros deben realizar largas colas desde la noche anterior para poder garantizar un asiento en un viaje. Esto perjudica tanto a las personas que laboran para compañías que ofrecen el servicio como a la sociedad venezolana en general.

En este proyecto se realizó una aplicación web que ofrece al venezolano común la posibilidad de organizar y encontrar viajes dentro del territorio nacional, todo esto bajo el concepto de *carpooling* o uso compartido de vehículos.

El uso de la aplicación permite a cualquier persona que viaja con su vehículo agendar su viaje en la aplicación y generar ingresos económicos, mientras que los pasajeros tienen la posibilidad de ahorrar tiempo y dinero. Esto podría ofrecerles la capacidad de buscar viajes ajustados a sus preferencias y necesidades, reservando su cupo desde cualquier lugar y en cualquier momento.

1.2 Antecedentes

El transporte público es un sistema integral de medios de transporte de uso generalizado, capaz de dar solución a las necesidades de desplazamiento de las personas. El transporte público terrestre puede ser urbano e interurbano. Cuando se hace mención al transporte urbano, se hace referencia a aquel que se presta dentro de una misma ciudad, mientras que el interurbano es el que transita fuera del perímetro urbano, es decir, de una ciudad a otra.

El uso compartido de vehículos también conocido por los términos en inglés *carpool, carsharing* o *carpooling* es la práctica que consiste en compartir un automóvil con otras personas tanto para viajes periódicos como para trayectos puntuales (Carpool, s.f).

El uso compartido de vehículos se hizo prominente en Estados Unidos durante la segunda guerra mundial como una táctica de racionamiento. Así como también lo hizo a principio de los años setenta debido a la crisis del crudo y nuevamente a finales de la misma década a causa de la crisis de energía (Marc & Andrew, 2010).

Nelson D. Chan y Susan A. Shaheen (Febrero, 2011) exponen el crecimiento del uso compartido de vehículos en Norteamérica a través de la historia. En el artículo que ambos escribieron también afirma que la tecnología jugará un papel muy importante en el futuro de este modelo, a través del intercambio de datos entre las empresas que permiten el enlace entre pasajeros y conductores.

Frédéric Mazzella (2004) funda BlaBlaCar en Francia, una red social que hace posible que las personas que quieran desplazarse al mismo lugar puedan hacerlo haciendo uso de un mismo vehículo. De esta manera, los pasajeros pueden ahorrar dinero compartiendo los gastos del viaje.

Stefan Weber, Matthias Siedler y Michael Reinicke (2001), desarrollan la plataforma web Mitfahrgelegenheit.de que más tarde sería conocida como Carpooling.com. Dicha plataforma tenía la intención de permitir a personas de recursos económicos limitados moverse de un lugar a otro permitiéndoles ahorrar dinero y combustible, así como también reducir las emisiones de gas y cuidar el medio ambiente.

En el año 2007, a través del boca a boca, Mitfahrgelegenheit se había convertido en el sitio web dedicado al uso compartido de vehículos más usado en Alemania. Posteriormente, decidieron extenderse a través de Europa y para el año 2010 ya el servicio estaba siendo utilizado por más de 2 millones de usuarios distribuidos en Italia, Francia, España, Suecia, Austria y Reino Unido.

En Abril del 2015, Carpooling.com fue vendido a su competidor BlaBlaCar por un monto desconocido (Dillet, 2015). Recientemente, internet y los dispositivos electrónicos como teléfonos celulares han ayudado al crecimiento de esta práctica, facilitando a los usuarios compartir sus vehículos en viajes cuyo recorrido sea el mismo. Para inicios del 2015, las plataformas que ofrecían este servicio sumaban entre ellas un total de 16 millones de usuarios a través de Europa (Chan & Susan A, 2011).

Otro ejemplo de una aplicación móvil, tal vez un poco similar, que ha logrado captar a millones de usuarios a través de Europa y el mundo es Uber (Mazas, 2015). Uber es un servicio de choferes que pretende simplificar el alquiler de coches con conductor. Éste funciona de forma parecida a los taxis convencionales, pero en este caso el conductor puede ser cualquier persona que haya sido previamente autorizado por la compañía tras una serie de exigencias.

El crecimiento de Uber ha sido tal que ha generado un conflicto con las empresas de taxis tradicionales, principalmente en España cuyos servicios en la ciudad de Barcelona iniciaron en abril del 2014 y en Madrid solo unos meses después, en septiembre de 2014. Incluso, se han llegado a registrar casos de amenazas y violencia física de los taxistas contra conductores y pasajeros de Uber.

El 21 de octubre de 2014, se realizó una manifestación en las calles de Barcelona en la que acudieron aproximadamente unos 300 taxistas. La protesta culminó con un metafórico funeral del sector taxista frente al departamento de Tranports de la Generalitat. Al día siguiente, se encontraron varios vehículos violentados y quemados cuyos dueños hacían uso de la popular aplicación (Victor & Javier, 2014).

Uber detiene sus servicios en España el 31 de diciembre del 2014, luego de que un juez ordenara su cese de actividades bajo la acusación de que la aplicación permitía a particulares sin licencia

dedicarse al transporte de viajeros. Actualmente, Uber busca trabajar dentro del marco legal obligando a los conductores a poseer licencia.

De igual manera, los conflictos de la aplicación se extienden, aunque de menor intensidad, a lo largo de América Latina en países como Colombia, Costa Rica, México y Uruguay. La mayoría de estos conflictos se deben a que Uber no cuenta con el debido permiso por parte de los países.

1.3 Objetivos del trabajo

El presente proyecto de grado busca solventar algunas de las dificultades que presenta el venezolano promedio al intentar viajar de una ciudad a otra haciendo uso del transporte público convencional. A continuación, se describe el objetivo general y los objetivos específicos que se buscan alcanzar en el desarrollo de este trabajo.

1.3.1 Objetivo general

Desarrollar una aplicación web que permita a los venezolanos agendar y encontrar viajes interurbanos en territorio nacional compartiendo vehículo con personas que viajan del mismo lugar y al mismo destino.

1.3.2 Objetivos específicos

- Analizar el sistema a desarrollar junto con todos sus componentes.
- Analizar el modelo de negocio para gestionar los riesgos, tanto sociales como propios de la aplicación, a través de una lista documentada de riesgos junto con un plan de gestión y de respuesta.
- Definir la arquitectura de software para el diseño de la aplicación.
- Diseñar e implementar la estructura de la base de datos.

- Desarrollar el *Front-End* y *Back-End* de la aplicación que permita una interacción amigable con el usuario.
- Implementar la aplicación web en un servidor de prueba.
- Realizar pruebas de integración, compatibilidad, escalabilidad y usabilidad.

1.4 Metodología

En el desarrollo de esta tesis se utilizó el método denominado Blue WATCH. Este marco de trabajo ofrece la ventaja de que está dirigido a pequeños grupos de desarrollo de software y se distingue a sí mismo como un método balanceado que guarda un equilibrio entre los métodos ágiles y disciplinados (Montilva, 2010).

El Modelo de Productos, el Modelo de Procesos y el Modelo de Actores son los tres principales modelos que componen el marco metodológico de trabajo de desarrollo Blue WATCH. Dichos modelos se consideran parcialmente flexibles puesto que son patrones adaptables y extensibles.

En el Modelo de Productos se describe un conjunto de productos intermedios y finales que serán desarrollados a lo largo del camino. Este modelo es iterativo, es decir, sus procesos se ejecutan en forma repetitiva hasta alcanzar un resultado deseado. También es considerado un modelo versionado e incremental. Cada ciclo del incremento produce una pieza de código ejecutable en un tiempo estimado entre 1 y 3 semanas.

El modelo de procesos describe detalladamente los procesos técnicos, de soporte y de gerencia que el equipo de desarrollo deberá utilizar. Finalmente, el tercer modelo denominado Modelo de Actores establece como estará organizado el equipo de desarrollo y el respectivo rol de cada uno de sus integrantes. Blue WATCH plantea una estructura para organizar grupos de trabajo e identifica los roles que son necesarios para ejecutar los procesos descritos en el Modelo de Procesos.

En el diseño de la aplicación se utilizó el lenguaje UML (*Unified Modeling Language*) en su versión 2.0 para modelar el comportamiento de la aplicación, ya que este lenguaje permite expresar de forma gráfica, explícita y versátil los métodos y procesos en el sistema, así como también la interacción entre sus componentes. Adicionalmente, el maquetado de la aplicación se realizó con HTML5 (*HyperText Markup Language*), CSS3 (*Cascading Style Sheets*) y haciendo uso del *framework* Bootstrap.

Puesto que el uso de dispositivos móviles como teléfonos inteligentes y tabletas se encuentra en constante crecimiento, el diseño de la aplicación web es completamente responsivo, permitiendo que todo tipo usuario hacedor de cualquier tipo de dispositivo con cualquier resolución pueda acceder a la aplicación web sin que la presentación estética de la página ni su funcionalidad se vea afectada.

La funcionalidad está programada en JavaScript en conjunto con PHP en su última versión más estable (7.0.2), y a través de éste segundo, se interactuará con la base de datos que a su vez estará diseñada con el manejador de bases de datos MySQL.

Para la arquitectura del software se utilizó el patrón MVC (modelo-vista-controlador), el cual propone la construcción de tres componentes distintos que son el modelo, la vista y el controlador, que nos permitirán separar los datos y la lógica de negocio de una aplicación de interfaz de usuario y el módulo encargado de gestionar los eventos y las comunicaciones.

El desarrollo de la aplicación se planificó para tres versiones o ciclos, en donde cada versión consta a su vez de tres iteraciones. Cada versión añade funcionalidades a la versión anterior y en cada iteración se refinan los requisitos y se corrigen los *bugs* en caso de existir alguno.

1.5 Resumen capítulo 1

La situación de Venezuela, en cuanto a transporte interurbano se refiere, ha venido presentando a lo largo del tiempo un déficit en su funcionamiento. La ausencia de unidades de transporte imposibilita al sector, cumplir la demanda de la población.

Los objetivos planteados en esta tesis buscan disminuir esta problemática ofreciendo una alternativa a la población venezolana para poder desplazarse en el país de una manera práctica.

Blue WATCH es el método que se utilizó en el desarrollo de la aplicación web, para modelar el comportamiento de la aplicación se utilizó UML 2.0. El lenguaje de programación utilizado fue PHP, haciendo uso de Laravel (un *framework* utilizado en el desarrollo de aplicaciones web) y como manejador de base de datos MySQL.

Capítulo 2

Para la compresión de cualquier trabajo de investigación es necesario que el lector tenga claro los conceptos que en este se manejan. El objetivo de este capítulo es poner a disposición del lector un conjunto de conceptos que son utilizados durante el resto del documento. Las definiciones técnicas presentadas a lo largo de este capítulo facilitaran el entendimiento y la comprensión lectora de esta tesis.

2.1 Marco teórico

A continuación, se definen algunos conceptos, ideas e informaciones previas fundamentales que permitan abordar el presente proyecto de grado con mayores conocimientos técnicos que faciliten la comprensión del mismo.

2.1.1 Carpooling

Carpooling o como es denominado en español "uso compartido de vehículos", es una técnica que permite a personas que buscan trasladarse de un mismo lugar a un mismo destino compartir un vehículo, del cual uno de los pasajeros es el dueño. Generalmente los pasajeros recompensan al dueño del vehículo económicamente o pagando los gastos del combustible y/o algún otro gasto ocasionado durante el viaje.

Existen excepciones donde el destino no es compartido, especialmente en largos recorridos, en este caso, el pasajero se dirige a un destino distinto aunque se encuentra en la ruta que recorrerá el conductor.

El *Carpooling* es una de las alternativas más comunes y económicas al transporte público, particularmente en lugares donde éste no es ofrecido como un buen servicio.

Esta técnica se ve más beneficiada a medida que aumentan los usuarios, tendiendo a experimentar una economía de escala, mientras más usuarios activos posea, más altas son las probabilidades de encontrar un viaje adecuado a ciertas necesidades (Victoria Transport Policy Institute, 2015).

Según (Beroldo, 1990), el éxito de esta técnica depende de varios factores, entre los más resaltables menciona que: la práctica de éste permita al conductor generar ingresos, ahorrar dinero a los pasajeros, que las ubicaciones de recolección de pasajeros sean cerca de autopistas, residencias o paradas de transporte público y que los pasajeros posean un destino común.

2.1.2 Funcionamiento del *Carpooling*

Los conductores hacen público su recorrido mediante un medio de comunicación, generalmente internet a través de páginas de mercadeo o una página/aplicación web dedicada exclusivamente a este tipo de servicio con los detalles pertinentes, lugar de salida, hora, fecha, destino y costo.

Los interesados se valen del medio para contactar al conductor en orden de solventar dudas y acordar detalles que no hayan sido especificados como por ejemplo espacio para el equipaje, lugar y número de paradas para descansar, medio de pago, etc.

Los pasajeros y el conductor deben hacer acto de presencia en el lugar a la hora y a la fecha acordada, los pasajeros pagan al conductor el monto pertinente e inician el recorrido hasta su destino.

Dependiendo del medio utilizado para contratar el servicio, los involucrados en el recorrido pueden calificarse unos a otros de manera positiva o negativa generando una reputación que sirva como referencia a la comunidad que hace uso del servicio.

2.1.3 Ventajas

El uso compartido de vehículos es muy popular alrededor del mundo puesto que ofrece múltiples ventajas. Sus beneficios más destacables son los siguientes:

- La práctica del *Carpooling* permite tanto a los conductores como a los pasajeros beneficiarse económicamente. Dependiendo de la modalidad que se aplique el conductor podría generar ingresos extras y los pasajeros podrían gastar menos dinero de lo que gastarían usando el transporte común.

- Colabora con el medio ambiente, menor uso de vehículos es traducido en menor emisión de gases que contaminan el aire, disminuyendo la contaminación visual y auditiva de igual manera.
- A través de esta práctica se puede lograr reducir el tráfico reduciendo el número de vehículos en la carretera.
- El *Carpooling* conecta personas, dándoles la oportunidad de establecer una amistad e incluso beneficiarse mutuamente de alguna manera.
- Ofrece flexibilidad, permitiendo a los pasajeros encontrar conductores que ofrezcan un itinerario adecuado a sus necesidades.
- Facilita el desplazamiento a personas que no posean vehículo propio.
- Permite el traslado a áreas que no están cubiertas por el sistema de transporte público.

2.1.4 Desventajas

El problema más común que enfrenta la práctica del servicio es la inquietud que presentan tanto los pasajeros como conductores al momento de compartir un vehículo privado con personas que no conocen de antemano.

La magnitud de este problema se puede decir que es relativa en cuanto a la cultura y a los índices de inseguridad del lugar donde se ofrezca el servicio.

Una solución a esta problemática, utilizada por los sitios y aplicaciones web, es integrar un sistema de reputación que permita calificar a los usuarios haciendo notar quienes son usuarios problemáticos y quienes son confiables y seguros.

Otro factor a tomar en cuenta, es la posibilidad de que la práctica de este modelo de transporte sea visto como una amenaza por las empresas de transporte interurbano convencional.

2.1.5 *Carpooling* en Venezuela

Según (Gutiérrez, 2016) en la década de los noventa, era muy común que estudiantes que estudiaban en la capital del país y hacian uso de su vehiculo para trasladarse desde y hacia otra ciudad, compartian su vehículo con otros estudiantes con el objetivo de cubrir los costos del combustible.

Actualmente en Venezuela no existe ninguna aplicación web que ofrezca el servicio de emparejamiento (*ride matching*) que permita hacer uso de esta modalidad de transporte. Así como tampoco ningún organismo que promueva públicamente esta práctica.

Aunque no es inusual encontrar personas que de manera informal prestan un tipo de servicio parecido, realizan trayectos puntuales llevando varios pasajeros y piden una retribución económica a cambio. Muy común en lugares donde el servicio de transporte público no es prestado o no se presta de manera continua.

2.1.6 Aplicaciones *ride matching*

Una aplicación *ride matching* es una aplicación cuya función es emparejar conductores con pasajeros interesados en trasladarse de un mismo lugar de origen a un destino común. La aplicación permite a los conductores agregar un recorrido junto a su itinerario a la base de datos a través de un formulario.

Los pasajeros o usuarios interesados en trasladarse a un lugar pueden navegar entre los distintos viajes que se encuentran disponibles y seleccionar el que más se adecue a sus necesidades. Adicionalmente, pueden hacer uso de un formulario que les permita seleccionar el origen, el destino, la fecha, la hora y algún otro dato. La aplicación haciendo uso de un algoritmo de emparejamiento y de los datos ingresados por el usuario debe mostrar los traslados disponibles que cumplan o se asemejen a las especificaciones de este.

Comúnmente estas aplicaciones cuentan con un sistema de pago incluido y un sistema de calificación de usuario, así como también, conexión a redes sociales generando mayor confianza entre los usuarios.

2.1.7 Aplicación web

Una aplicación web es un tipo de aplicación cliente/servidor en el que el usuario, haciendo uso de un navegador web realiza peticiones a una aplicación remota a través de internet (o incluso una intranet) y que recibe una respuesta que es mostrada en el mismo navegador (Mora, 2002).

Los sitios web no ofrecen ninguna funcionalidad que permita interactuar con la página. Su objetivo principal se limita a ofrecer información de forma estática que podría ser o no relevante sus visitantes.

Las aplicaciones web se diferencian de los sitios web ya que estos primeros muestran información de manera dinámica haciendo uso de un lenguaje del lado del servidor como lo es PHP, Python, Ruby o cualquier otro. De igual manera pueden interactuar con bases de datos para guardar información relevante y mostrar datos almacenados en estas. Realizar procesamientos de datos en el lado del servidor a través de complejos algoritmos, tomar decisiones con base a los datos introducidos por el usuario, interactuar con otras aplicaciones web, etc.

2.1.8 Base de datos

Una base de datos es el conjunto de datos informativos organizados en un mismo contexto para su uso y vinculación (Bembibre, 2009).

Las propiedades más resaltantes de una base de datos son:

- Representan un aspecto del mundo real y los cambios que ocurren en este son reflejados de igual manera en la base de datos.
- Las base de datos son una colección de datos, pero más alla de esto, deben ser datos logicamente coherentes con algun significado inherente y no solo datos aleatorios.
- Una base de datos es diseñada y construida con algún objetivo específico que permita a un grupo de usuarios y/o aplicaciones hacer uso de los datos que esta contenga.

Las bases de datos pueden generarse y mantenerse manualmente o a través de un computador, estas segundas llamadas computarizadas, donde la introducción de datos generalmente es a través de uno o varios formularios electrónicos (Elmasri & Navathe, 2007).

2.1.9 Sistema Manejador de Base de Datos (DBMS)

Un sistema manejador de base de datos es un conjunto de programas que permiten al usuario la creación y el mantenimiento de las bases de datos. El DBMS ofrece herramientas que facilitan el proceso de definir, construir, manipular y compartir las bases de datos entre múltiples usuarios y aplicaciones.

Adicionalmente, los DBMS ofrecen funciones de protección contra el acceso no autorizado de usuarios y contra errores ocasionados por funcionamiento defectuoso del hardware (Elmasri & Shamkant B., 2007)

2.1.10 HTML

HTML (*HyperText Markup Language*) es un lenguaje de marcado de hipertexto usado para la creación de páginas web. La palabra "*Hypertext"* hace referencia a los hipervínculos que contiene una página HTML y "*Markup Language*" hace referencia a las etiquetas que son usadas para definir los elementos que constituyen la estructura de la página (Christensson, 2015).

HTML fue propuesto en 1980 como un nuevo sistema de hipertexto que permitía compartir documentos, su creador fue el físico Tim Berners-Lee.

El 22 de septiembre de 1996, el IETF (*Internet Engineering Task Force*) consigue publicar el estándar HTML 2.0 y, a pesar de su nombre, este es el primer estándar oficial de HTML.

Posteriormente, en enero del año 1997 se publicó la versión 3.2 de HTML, pero esta vez lo hacia otro organismo de estandarización, el W3C (*World Wide Web Consortium*).

En el año 1998 se publicó el HTML 4.0 con una gran cantidad de mejoras que incluían a las hojas de estilo (CSS) y que permitía la inclusión de "*scripts*", así como también mejoras en las tablas y en los formularios. Un año después, en diciembre de 1999 se hizo pública la versión 4.01 la cual es una revisión y una actualización de la versión 4.0.

HTML5 es la última versión del lenguaje y fue publicada como estándar en octubre del 2014. Esta última versión contiene significantes cambios como la adición de nuevas etiquetas que permiten definir la estructura de la página web, elementos que habilitan la reproducción de audio y video, nuevos tipos de ingreso de texto a través de formularios, barras de progreso, canvas, entre otros.

2.1.11 CSS

CSS (*Cascading Style Sheets*) es una tecnología que permite la manipulación de los elementos HTML contenidos en una página web. Las hojas de estilo definen como lucirán los elementos HTML en el navegador del usuario. A través de estas hojas es posible modificar el color, el tamaño, el margen, la

tipografía y otras características que tendrá un determinado elemento. El término "*Cascading*" indica que distintas hojas de estilos son aplicables simultáneamente a una página (Mora, 2002).

La especificación CSS3 (*Cascading Style Sheets, level 3*) es actualmente la última actualización a las hojas de estilo de cascada. Esta nueva especificación está divida en módulos, ha dividido la versión anterior (CSS2) en pequeños módulos y se han agregado otros nuevos. Algunos de los más importantes son: selectores, modelo de caja, fondo y bordes, animaciones, interfaz de usuario, efectos de texto y transformaciones 2D y 3D.

2.1.12 PHP

PHP (*Hypertext Preprocessor*) es un lenguaje de programación y *scripting* del lado del servidor usado en la creación de sitios web dinámicos, diseñado por Rasmus Lerdorf en 1994.

Cuando un usuario hace una petición a una página web que contiene código PHP, el código es procesado por el módulo PHP que se encuentra en el servidor y genera el código correspondiente en HTML que será mostrado en el navegador del usuario (Balkhi).

A través de PHP se puede interactuar con la base de datos bien sea para consultar o alojar información en ella. De igual manera permite realizar válidaciones para asegurarnos de no introducir datos con formato erroneo o código malicioso provisto por usuarios malintencionados a través de formularios, ofreciendo mayor seguridad y robustez a la aplicación.

2.1.13 Laravel

Laravel es un potente *framework* de código abierto (*open source*) que permite el desarrollo de aplicaciones web con PHP de una manera rápida, elegante, robusta y segura.

Este *framework* proporciona un gran conjunto de herramientas que permite desarrollar aplicaciones bajo la arquitectura MVC (modelo-vista-controlador), permitiendo escribir código más claro y obligando al desarrollador a elaborar una aplicación más limpia, fácil de depurar, manipular y escalar.

Entre las características más populares ofrecidas por Laravel tenemos las siguientes:

- Sistema de ruteo: a través de este sistema se pueden definir las rutas que manejará la aplicación web. El sistema de ruteo permite decidir qué hacer cuando la aplicación recibe una petición HTTP (*Hypertext Transfer Protocol*) en una url específica.

- Motor de plantillas: el motor de plantilla ofrecido por Laravel es llamado *Blade*. Este motor permite escribir código PHP a través de una sintaxis más limpia que el método tradicional. Asimismo, *Blade* permite dividir una vista general en distintas vistas parciales que pueden ser reutilizadas a lo largo del proyecto.
- Migraciones: Las migraciones permiten tener un registro de las modificaciones que se han hecho o se harán en la base de datos, haciendo y deshaciendo cambios a nuestra voluntad.
- *Eloquent* ORM: un mapeo de objeto relacional llamado *Eloquent* ORM es una interfaz que provee el *framework* Laravel para interactuar con la base de datos como si las tablas se trataran de clases, y los registros como instancias de esas clases.
- *Database seeding*: esta característica de Laravel proporciona una manera de ingresar datos a la base de datos que podrían ser utilizados por defecto en la realización de pruebas.
- Unidad de pruebas: la unidad de pruebas de Laravel permite escribir pruebas que serán realizadas sobre la aplicación, facilitando la detección y depuración de errores.
- Controladores: los controladores nos ofrecen una alternativa de separar la lógica detrás del servicio de peticiones HTTP.

2.1.14 UML

UML (Lenguaje Unificado de Modelado) es un lenguaje de modelado gráfico que permite especificar, visualizar y documentar modelos de sistemas de software incluyendo su estructura y diseño (*Object Management Group*, 2005).

UML 2.5 ofrece trece tipos de diagramas los cuales están divididos en tres categorías:

- Diagramas de estructura: esta categoría de diagramas incluye a los diagramas de clases, diagramas de objeto, diagramas de componentes, diagramas de estructura compuesta, diagramas de paquete y diagramas de despliegue.
- Diagramas de comportamiento: incluye a los diagramas de clases, diagramas de actividades y los diagramas de máquinas de estado.
- Diagramas de interacción: en esta categoría se incluyen los diagramas de secuencia, diagramas de comunicación, diagramas de temporización y resumen de la interacción.

2.2 Resumen capítulo 2

El *Carpooling* es un paradigma que busca facilitar el traslado de personas que viajan hacia un destino en común. Presenta un conjunto de ventajas que hacen de este una buena alternativa sobre el transporte público convencional.

Las aplicaciones web juegan un papel importante en la práctica del *Carpooling,* facilitando el emparejamiento entre conductores y pasajeros.

HTML, CSS, y Laravel son las herramientas tecnológicas que se usaron para el desarrollo de la aplicación web descrita en esta tesis.

PHP, través del *framework* Laravel se usó el desarrollo de la funcionalidad de la aplicación. Laravel es un *framework* que facilita la elaboración de aplicaciones disminuyendo el tiempo de desarrollo, aumentando la seguridad y ofreciendo una fácil interacción con la información almacenada en la base de datos.

Capítulo 3

En este capítulo se expone cómo está compuesto y cómo funciona el sistema desarrollado; que reglas rigen su comportamiento y sobre cuales requisitos fue construido. La forma en como están clasificados los potenciales usuarios de la aplicación y el alcance que estos poseen en el entorno de la misma.

3.1 Análisis de contexto y desarrollo inicial de requisitos

El modelado del negocio es la primera de las siete fases del método Blue Watch. Esta fase permite generar un conjunto de descripciones y diagramas (mayormente en UML) como producto de un proceso de modelado que ofrecerán un conocimiento global y detallado del comportamiento de la aplicación.

3.2 Modelo del Sistema de Negocio

El modelado del negocio es la primera de las siete fases del método Blue Watch. Esta fase permite generar un conjunto de descripciones y diagramas (mayormente en UML) como producto de un proceso de modelado que nos ofrecerán un conocimiento global y detallado del comportamiento de la aplicación.

En este modelo se definirá la jerarquía de sistemas, los objetivos del sistema de negocio, los diagramas de procesos, diagramas de actividades, el conjunto de reglas del negocio, entre otros.

La aplicación web cuenta con dos interesados principales, el autor y la tutora de la tesis.

Los requisitos fueron fijados directamente por el autor con el objetivo de crear una aplicación web que genere impacto en la sociedad venezolana y que a su vez ofrezca una buena "experiencia de usuario" y el tiempo límite de desarrollo, el cual es de un semestre y extensible hasta dos, está fijado por EISULA.

3.2.1 Diagrama de jerarquía de sistemas

La aplicación web puede ser descompuesta en distintos subsistemas. El diagrama presentado en la figura 1 permite definir el supra sistema e identificar los subsistemas relacionados a la aplicación.

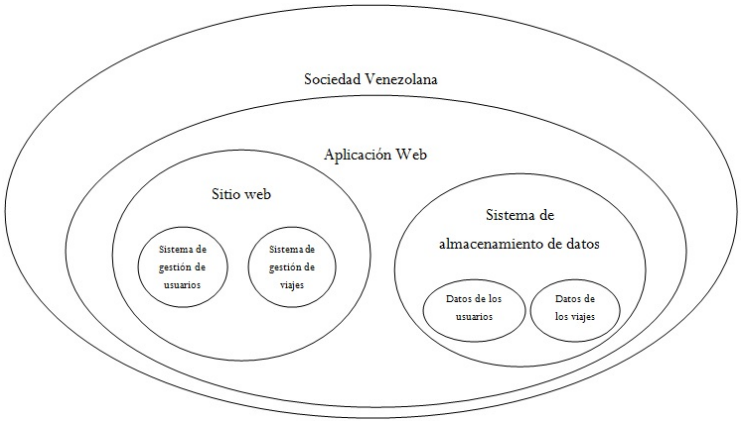

Figura 1. Diagrama de jerarquía de Sistemas

3.2.2 Diagrama de Objetivos

La figura 2 muestra el diagrama de objetivos que permite definir la visión, la misión, los objetivos generales y específicos que persigue el desarrollo de esta tesis.

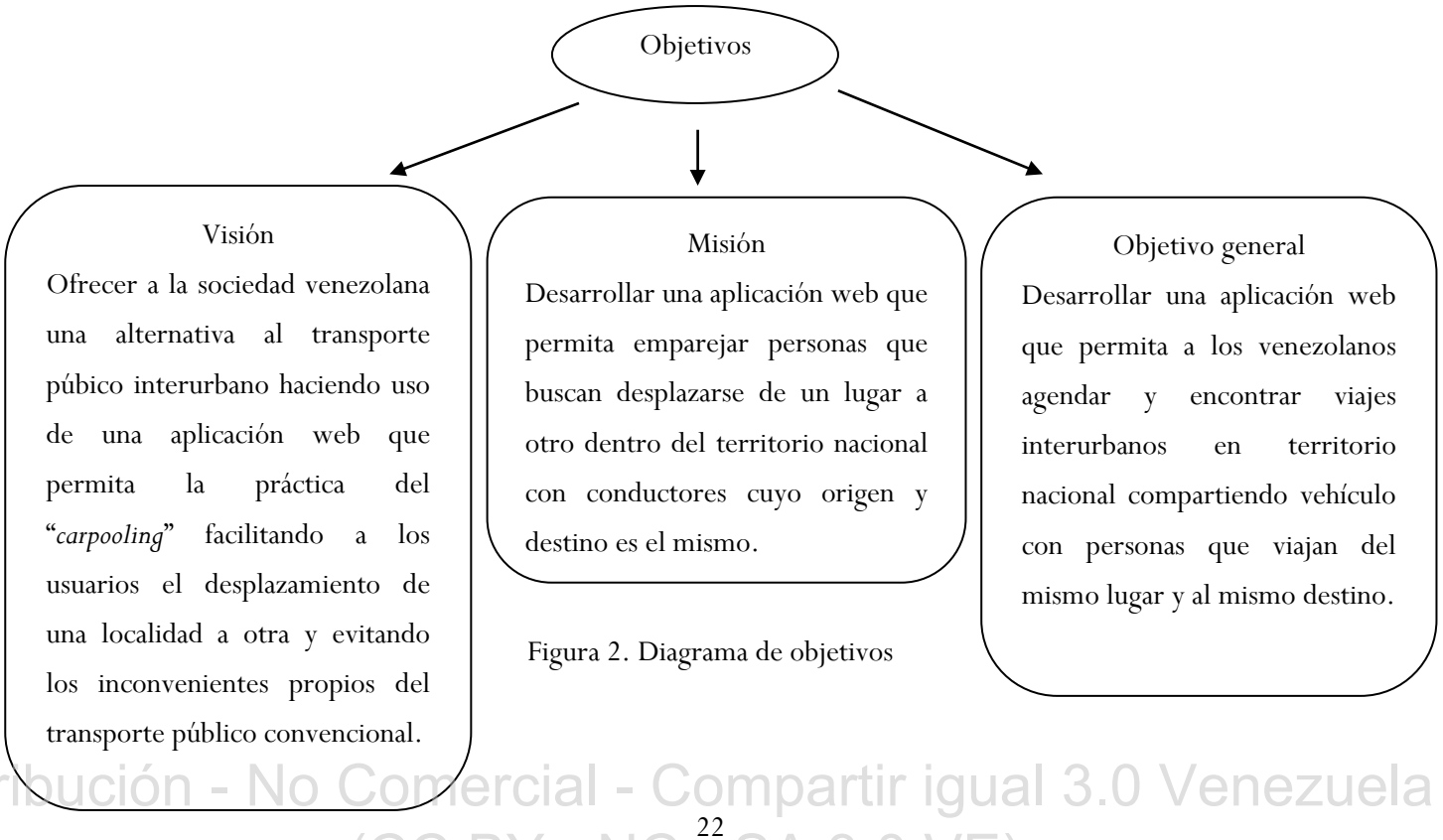

Figura 2. Diagrama de objetivos

3.3 Ingeniería de requisitos

Esta es la segunda fase del método Blue Watch. A continuación, se especifican los requisitos del sistema haciendo uso de plantilla Volere y diagramas de caso de uso tal como lo indica el método Blue Watch.

3.3.1 Listado de requisitos

Tabla 1. Requisito 1.

# de requerimiento	01	Tipo de requerimiento	Funcional	CU / Evento relacionado	
Descripción	Permitir a un usuario no autenticado crearse una cuenta en el sistema				
Justificación	Con el objetivo de que posteriormente, el usuario, pueda ingresar al sistema y hacer uso de sus características				
Origen (Interesado)	Lenin Roa				
Criterio de aceptación/validación	El usuario debe poder iniciar sesión en el sistema con la cuenta creada				
Grado de satisfacción	5		Grado de insatisfacción	5	
Prioridad	5		Requisito en conflicto		
Última modificación	Creado 20/08/2015 – Lenin Roa				

Tabla 2. Requisito 2.

# de requerimiento	02	Tipo de requerimiento	Funcional	CU / Evento relacionado	
Descripción	Un usuario que posee una cuenta debe ser capaz de autenticarse usando su correo y contraseña				
Justificación	Identificarse ante al sistema con el objeto de identificar todas las acciones y solicitudes del usuario				
Origen (Interesado)	Lenin Roa				
Criterio de aceptación/validación	El usuario debe ser capaz de solicitar reservaciones de plazas en viajes, publicar viajes, realizar preguntas, editar su perfil, etc.				
Grado de satisfacción	5		Grado de insatisfacción	5	
Prioridad	5		Requisito en conflicto	01	
Última modificación	Creado 20/08/2015 – Lenin Roa				

Tabla 3. Requisito 3.

# de requerimiento	03	Tipo de requerimiento	Funcional	CU / Evento relacionado	
Descripción	El usuario debe ser capaz de buscar los viajes disponibles para un lugar y una fecha especifica				
Justificación	El usuario debe poder navegar entre las distintas opciones disponibles				
Origen (Interesado)	Lenin Roa				
Criterio de aceptación/validación	El sistema debe mostrar al usuario el listado de todos los viajes disponibles que concuerden con los datos ingresados o generar una alertar en caso de que no existan coincidencias				
Grado de satisfacción	5		Grado de insatisfacción	5	
Prioridad	5		Requisito en conflicto		
Última modificación	Creado 20/08/2015 – Lenin Roa				

Tabla 4. Requisito 4.

# de requerimiento	04	Tipo de requerimiento	Funcional	CU / Evento relacionado	
Descripción	El usuario debe ser capaz de enviar una solicitud de reserva de plaza en un viaje deseado al usuario conductor del viaje				
Justificación	De esta manera se permitirá al usuario conductor del viaje gestionar las solicitudes y decidir quiénes serán los pasajeros en su viaje				
Origen (Interesado)	Lenin Roa				
Criterio de aceptación/validación	El sistema debe enviar una notificación al conductor del viaje el cual podrá aceptar o rechazar dicha solicitud				
Grado de satisfacción	5		Grado de insatisfacción	5	
Prioridad	5		Requisito en conflicto	02	
Última modificación	Creado 20/08/2015 – Lenin Roa				

Tabla 5. Requisito 5.

# de requerimiento	05	Tipo de requerimiento	Funcional	CU / Evento relacionado	
Descripción	Permitir a los usuarios realizar preguntas en la publicación de un viaje				
Justificación	Este requerimiento se hace con la finalidad de que todas las dudas que posea un usuario con respecto a un viaje sean despejadas				
Origen (Interesado)	Lenin Roa				
Criterio de aceptación/validación	El usuario debe poder realizar una pregunta en un viaje determinado llenando un formulario				
Grado de satisfacción	5		Grado de insatisfacción	5	
Prioridad	5		Requisito en conflicto	02	
Última modificación	Creado 20/08/2015 – Lenin Roa				

Tabla 6. Requisito 6.

# de requerimiento	06	Tipo de requerimiento	Funcional	CU / Evento relacionado	
Descripción	Permitir a los usuarios asociar vehículos a su cuenta				
Justificación	De esta forma se evita al usuario la necesidad de introducir la información sobre el vehículo cada vez que publica un viaje. También le puede permitir gestionar que vehículo utilizar en un determinado viaje en caso de que posea más de uno asociado a su cuenta.				
Origen (Interesado)	Lenin Roa				
Criterio de aceptación/validación	El vehículo debe asociarse a la cuenta del usuario y al momento de que el usuario desee publicar un viaje este debe mostrarse como una opción de medio de transporte que se utilizará en el viaje				
Grado de satisfacción	5		Grado de insatisfacción	5	
Prioridad	5		Requisito en conflicto	02	
Última modificación	Creado 20/08/2015 – Lenin Roa				

Tabla 7. Requisito 7.

# de requerimiento	07	Tipo de requerimiento	Funcional	CU / Evento relacionado	
Descripción	Permitir a los usuarios publicar un viaje				
Justificación	Este requerimiento es necesario para que existan viajes disponibles en el sistema. Permitiendo a usuarios interesados en ser pasajeros enviar solicitudes de plaza				
Origen (Interesado)	Lenin Roa				
Criterio de aceptación/validación	El viaje y toda la información referente a este debe mostrarse cuando se realice una búsqueda que coincida con sus especificaciones				
Grado de satisfacción	5		Grado de insatisfacción	5	
Prioridad	5		Requisito en conflicto	06	
Última modificación	Creado 20/08/2015 – Lenin Roa				

Tabla 8. Requisito 8.

# de requerimiento	08	Tipo de requerimiento	Funcional	CU / Evento relacionado	
Descripción	Permitir a los usuarios conductores de un viaje responder las preguntas realizadas por otros usuarios				
Justificación	Despejar cualquier duda que tengan los potenciales pasajeros sobre el viaje				
Origen (Interesado)	Lenin Roa				
Criterio de aceptación/validación	El usuario debe poder realizar una pregunta en un viaje determinado llenando un formulario				
Grado de satisfacción	5		Grado de insatisfacción	4	
Prioridad	4		Requisito en conflicto	07	
Última modificación	Creado 20/08/2015 – Lenin Roa				

Tabla 9. Requisito 9.

# de requerimiento	09	Tipo de requerimiento	Funcional	CU / Evento relacionado	
Descripción	Un usuario debe ser capaz de calificar positiva o negativamente a otro usuario siempre y cuando el primero haya sido pasajero en un viaje conducido por el segundo				
Justificación	Ofrecer un sistema de reputación a la aplicación que permita generar mayor confianza entre los usuarios. Permitiendo de esta manera futuros potenciales pasajeros de un conductor conocer la reputación de este y decidir si viajar o no con él.				
Origen (Interesado)	Lenin Roa				
Criterio de aceptación/validación	La calificación debe asignarse al usuario calificado				
Grado de satisfacción	5		Grado de insatisfacción	4	
Prioridad	4		Requisito en conflicto	04	
Última modificación	Creado 20/08/2015 – Lenin Roa				

Tabla 10. Requisito 10.

# de requerimiento	10	Tipo de requerimiento	Funcional	CU / Evento relacionado	
Descripción	Un usuario debe ser capaz de cancelar un viaje que haya publicado anteriormente				
Justificación	En caso de que al conductor de viaje se le presente un inconveniente mayor y no pueda realizar el viaje, este pueda cancelarlo para que no aparezca en el listado de viajes				
Origen (Interesado)	Lenin Roa				
Criterio de aceptación/validación	El estatus del viaje debe ser cambiado en la base de datos y notificarse a los pasajeros de que el viaje ha sido cancelado				
Grado de satisfacción	5		Grado de insatisfacción	4	
Prioridad	3		Requisito en conflicto	07	
Última modificación	Creado 20/08/2015 – Lenin Roa				

Tabla 11. Requisito 11.

# de requerimiento	11	Tipo de requerimiento	Funcional	CU / Evento relacionado	
Descripción	El sistema debe notificar a los pasajeros de un viaje si este es cancelado en algún momento por el conductor del viaje				
Justificación	Hacerles saber a los pasajeros de un viaje que el este ha sido cancelado con el objetivo de evitarles mayores inconvenientes				
Origen (Interesado)	Lenin Roa				
Criterio de aceptación/validación	Una notificación debe ser enviada a cada uno de los pasajeros del viaje cancelado				
Grado de satisfacción	5		Grado de insatisfacción	4	
Prioridad	3		Requisito en conflicto		
Última modificación	Creado 20/08/2015 – Lenin Roa				

3.3.2 Descripción de las reglas del negocio

A continuación, haciendo uso de las reglas del negocio se describe detalladamente las normas, operaciones, restricciones y políticas del negocio:

1. Una dirección de correo electrónico puede estar asociada a una y solo una cuenta.
2. Un usuario debe haber registrado al menos un vehículo antes de publicar un viaje.
3. Un usuario puede publicar tantos viajes como desee siempre y cuando no haya colisión entre las fechas y hora de los mismos.
4. El usuario debe estar autenticado en el sistema para poder publicar viajes o solicitar una reserva de plaza en un viaje.
5. Los pasajeros de un viaje podrán calificar al conductor siempre y cuando el viaje haya ocurrido
6. El conductor de un viaje puede cancelar el viaje en cualquier momento que desee.
7. Cuando un viaje haya sido cancelado por el usuario conductor, el sistema debe enviar una notificación a los pasajeros del viaje.
8. Si un conductor cancela un viaje planeado donde haya al menos una plaza reservada, el conductor recibirá una penalización.
9. Toda cuenta de usuario que reciba tres amonestaciones será dada de baja del sistema imposibilitando la creación de otra cuenta bajo el mismo correo.
10. No se permite a un usuario conductor de un viaje modificar la fecha ni hora del mismo. Para ello deberá cancelar el viaje y agendar uno nuevo.
11. Los usuarios autenticados deben poder realizar preguntas en la publicación de un viaje con el fin de disipar cualquier duda sobre el mismo.
12. Si un usuario autenticado permanece inactivo en el sistema su sesión deberá finalizar automáticamente tras diez (10) minutos de inactividad.
13. Un usuario moderador debe ser capaz de eliminar publicaciones de viajes de terceros.
14. Cuando un moderador cancela un viaje, este deberá indicar explícitamente por qué lo ha hecho.

3.3.3 Descripción de actores y sus roles

El sistema contará con tres tipos de actores que interactuarán constantemente con este. Estos actores son: usuario no autenticado, usuario autenticado y el moderador. La tablas 12, 13 y 14 describen, correspondientemente, a estos tipos de usuarios.

Tabla 12. Usuario no autenticado

Nombre	Usuario no autenticado
Todo aquel usuario que no se haya identificado ante el sistema iniciando sesión. Este actor debe ser capaz de registrarse en el sistema (si no lo ha hecho antes), iniciar una sesión, buscar y ver información detallada sobre los viajes disponibles	

Tabla 13. Usuario autenticado

Nombre	Usuario autenticado
Es todo aquel usuario que ha ingresado sus datos de autenticación correctamente y se encuentra en una sesión activa. Este usuario debe ser capaz de cerrar la sesión, editar su perfil, buscar y ver información sobre los viajes disponibles, publicar viajes y gestionar las solicitudes de reserva de viajes propios, enviar solicitudes de reserva de plaza en viajes de terceros, calificar a otro usuarios (siempre y cuando haya "viajado" con ellos) y realizar preguntas públicas a otros usuarios conductores de un viaje.	

Tabla 14. Usuario moderador

Nombre	Moderador
El moderador es un usuario autenticado con privilegios que le permite cancelar viajes de terceros cuya información expuesta no cumpla con las políticas de uso del sistema o no se considere apropiada.	

3.3.4 Matriz de actores vs. procesos

A continuación, se listan los procesos involucrados en la matriz de procesos con sus respectivos identificadores.

P1: Registrar cuenta

P2: Iniciar sesión

P3: Finalizar sesión

P4: Buscar/ver viajes

P5: Enviar solicitud de reserva de asiento

P6: Publicar viajes

P7: Gestionar solicitudes de reserva de asiento/plaza

P8: Calificar usuario

P9: Editar perfil

P10: Registrar automóvil

P11: Realizar preguntas en la publicación de un viaje

P12: Darse de baja del sistema

P13: Cancelar viajes propios

P14: Cancelar viajes de terceros

P15: Eliminar preguntas de terceros

P16: Autenticarse con Facebook.

La tabla 15 (parte 1 y 2) indica que procesos pueden ser ejecutados por los distintos actores. Un actor es una entidad que interactúa con el sistema jugando un rol específico, mientras que el usuario, en contexto, es una persona física. Existen distintos tipos de usuarios, y una persona puede representar uno u otro dependiendo de la situación, tal como un actor puede representar distintos tipos de roles.

Tabla 15. Matriz de actores vs procesos (Parte 1).

	P1	P2	P3	P4	P5	P6	P7	P8
Usuario no autenticado	X	X		X				
Usuario autenticado		X	X	X	X	X	X	X
Moderador		X	X	X				

Tabla 15. Matriz de actores vs procesos (Parte 2).

	P9	P10	P11	P12	P13	P14	P15	P16
Usuario no autenticado								X
Usuario autenticado	X	X	X	X	X			
Moderador						X	X	

La tabla 15 (parte 1 y 2), muestra qué procesos listados anteriormente están asociados a qué tipo de usuario. Es decir, cada usuario puede iniciar el proceso al cual está asociado.

En la tabla se puede observar que los procesos 1, 2, 4 y 16 están asociados al tipo de usuario "usuario no autenticado". Los procesos del 2 al 13 al "usuario autenticado" y los procesos 14 y 15 al usuario tipo "moderador".

3.4 Resumen capítulo 3

El análisis de contexto, el modelo del sistema de negocio y la Ingeniería de Requisitos son fundamentales en el proceso de desarrollo cuando se hace uso de la metodología WATCH. Estas etapas permiten definir el alcance y el funcionamiento de la aplicación.

En este capítulo se utilizó la plantilla Volere para definir los requisitos de la aplicación web; se describieron los tipos de actores que interactúan con la aplicación (usuario no autenticado, usuario autenticado y moderador) y se realizó una matriz de actores-procesos donde se señalan los procesos que pueden ser ejecutados por cada uno de los roles.

Capítulo 4

El capítulo 4 presenta el diseño arquitectónico y la estructura de la aplicación. A través de un conjunto de diagramas UML (de actividad, caso de uso y de clases) se expone cómo funciona la aplicación a un nivel técnico y la interacción que tienen los usuarios con la misma, tal y como fue definido anteriormente en el capítulo anterior. Adicionalmente se muestra las iteraciones planificadas y los procesos que constituyen a cada una de estas.

4.1 Diseño arquitectónico

La tercera fase del método Blue Watch tiene como objetivo elaborar un diseño de la aplicación que sea apropiado a los requisitos especificados donde se establezcan los subsistemas junto con sus componentes y las conexiones entre estos.

4.1.1 Listado descriptivo de las metas de diseño

- Seguridad y datos: la arquitectura de la aplicación debe ser segura. Una arquitectura segura debe ser capaz de separar los datos, de la lógica de negocio y la interfaz de usuario. El acceso a los datos debe ser a través de piezas de software cuya única finalidad sea únicamente esta, la lectura y escritura de la información almacenada en la base de datos. Asimismo, estas piezas de software deben ofrecer protección contra los ataques comunes realizados por usuarios malintencionados.
- Escalabilidad: la arquitectura del sistema debe estar diseñada de tal manera que permita realizar cambios en su capacidad de una manera simple y fácil sin la necesidad de rehacer la totalidad del sistema.
- Robustez: un sistema con una arquitectura robusta debe ofrecer módulos que permitan dirigir el comportamiento del sistema cuando ocurra un evento no contemplado. La robustez debe permitir identificar un error a través de mensajes apropiados y permitir finalizar cualquier proceso sin comprometer los datos.
- Reutilización de software: el diseño arquitectónico debe suprimir en lo posible la escritura repetitiva de código ofreciendo un sistema que permita reutilizar piezas de

software creadas con anterioridad. Esto permitirá a su vez desarrollar la aplicación de una manera más rápida y limpia.

- Orientado a pruebas: la arquitectura del sistema debe ser orientada a pruebas, es decir, debe permitir realizar pruebas con lotes de datos manipulados previamente sin la necesidad de comprometer datos originales ni el sistema en sí.

4.1.2 Estructura de la aplicación

La aplicación está estructurada con el patrón de diseño de software MVC (modelo-vista-controlador). Esta arquitectura permite separar la información de la interfaz de usuario y la forma de cómo esta accede a los datos.

El desarrollo de la aplicación se realizó haciendo uso del *framework* Laravel. Este *framework* permite hacer uso del patrón de diseño MVC. Asimismo, ofrece un conjunto de herramientas que se acoplan perfectamente con esta arquitectura y nos permite desarrollar la aplicación de una manera más rápida y segura.

La arquitectura MVC propone la construcción de tres módulos. El modelo, la vista y el controlador.

El modelo es el módulo encargado de la interacción con los datos. Realiza consultas a la base de datos y retorna la información como respuesta a las peticiones realizadas por el controlador.

El controlador recibe las peticiones realizadas por el usuario a través de un módulo de ruteo (disponible en Laravel) y procesa esas peticiones, el controlador decide qué acciones realizar. Este puede procesar únicamente la petición y enviar una vista al usuario o interactuar con la base de datos a través del modelo para luego enviar una respuesta al usuario. Además, el controlador a su vez puede interactuar con otros módulos más pequeños que no están contemplados en la estructura MVC convencional.

La vista (o vistas) representan lo que usuario ve y con lo que el usuario interactúa. La vista es el código HTML que presenta al usuario la información final que es enviada por el sistema como una respuesta a su petición.

La figura 3 representa gráficamente la estructura de la aplicación y la interacción entre los distintos módulos que la componen.

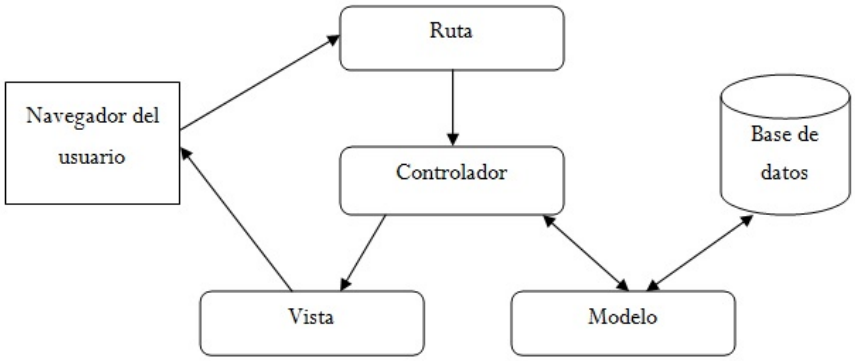

Figura 3. Estructura de la aplicación

4.1.3 Diagrama de clases

El diagrama de clases mostrado a continuación en la figura 4 permite describir la estructura del sistema a través de las clases que lo componen y sus atributos.

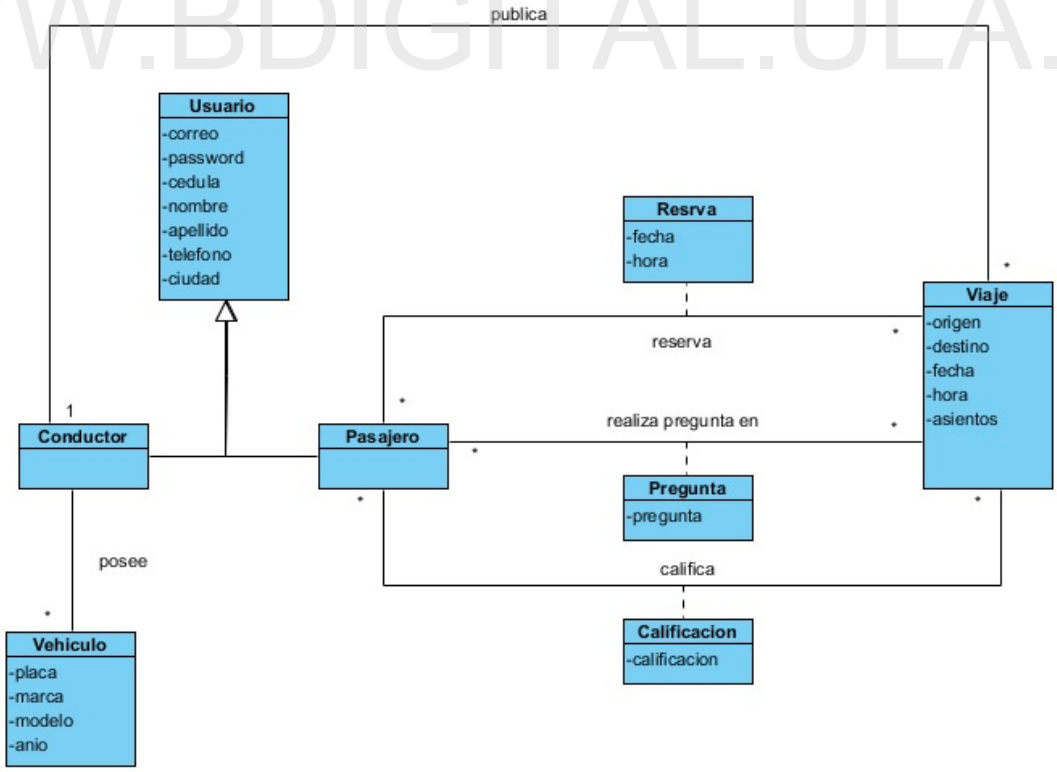

Figura 4. Diagrama de clases en UML

4.1.4 Diagramas de caso uso

Los diagramas de caso de uso presentados describen el comportamiento que tendrá cada tipo de usuario con el sistema.

La figura 5 muestra las acciones que puede ejecutar un usuario anónimo o no autenticado, es decir, que no ha iniciado sesión en la aplicación. La figura 6 muestra el conjunto de acciones que pueden realizar los usuarios que previamente han pasado exitosamente por el proceso de autenticación. El diagrama de la figura 7 muestra las acciones que pueden ser realizadas únicamente por un usuario autenticado pero con el rol de Moderador.

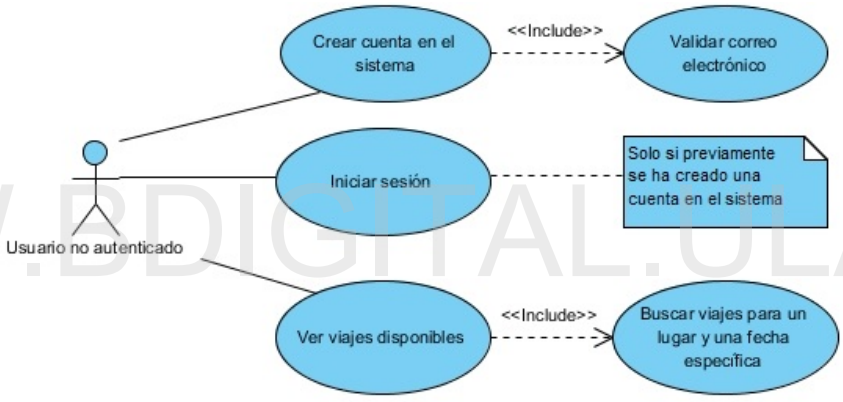

Figura 5. Diagrama de caso de uso de usuario no autenticado

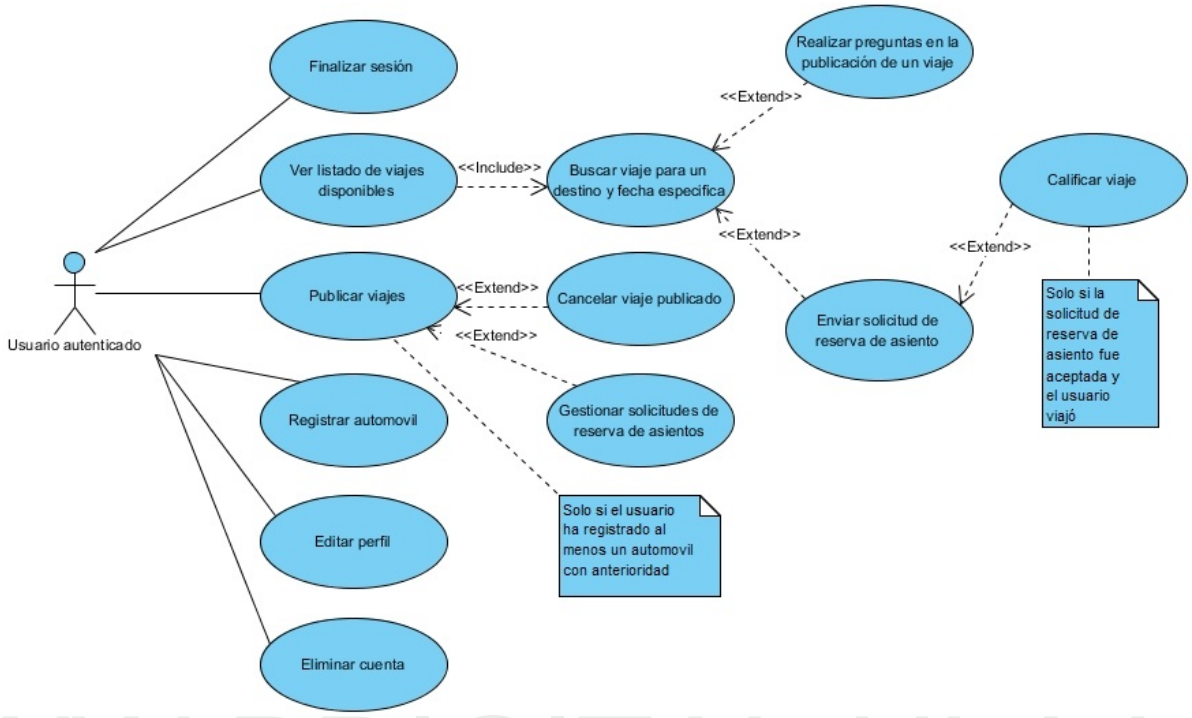

Figura 6. Diagrama de caso de uso de usuario autenticado

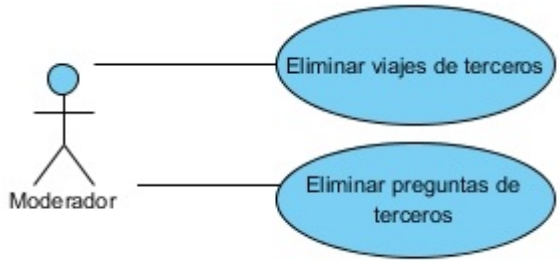

Figura 7. Diagrama de caso de uso de usuario moderador

4.1.5 Planificación de las iteraciones

El estudio detallado de los requisitos permite determinar los ciclos tentativos y a su vez necesarios para llevar a cabo el desarrollo completo de la aplicación web. Tras un análisis profundo de

los requerimientos, se planifican tres ciclos de versiones que, a su vez, contienen varios ciclos de incremento.

Los ciclos de versiones se especifican a continuación:

- Primer ciclo: en este primer ciclo de versión se desarrollaron los procesos P1, P2, P3 P4, P6 y P11 especificados en la Tabla 15, donde se incluyen los casos de uso relacionados al "usuario no autenticado" mostrados en la figura 5.
  Duración estimada: 7-8 semanas.
- Segundo ciclo: el segundo ciclo de versión contempla lograr el desarrollo de las acciones asociadas a los procesos P5, P7, P13 y P9.
  Duración estimada: 4-6 semanas.
- Tercer ciclo: el tercer y último ciclo se enfoca en el desarrollo de las acciones del actor "moderador" en la figura 7 y en los procesos P8, P11, P12, P14 y P15.
  Duración estimada: 3-4 semanas.

Cada ciclo consta de tres iteraciones, la primera de desarrollo, la segunda de refinamiento y la tercera es correctiva en caso de existir algún error. La duración estimada no solamente contempla el desarrollo funcional si no también el desarrollo de la interfaz gráfica, pruebas, verificaciones y validaciones.

4.2 Resumen capítulo 4

En este capítulo se definió la arquitectura de la aplicación web. El patrón arquitectónico MVC se utilizó para la elaboración de la aplicación web. Este patrón permite separar la interfaz gráfica con la que interactúa el usuario, de la lógica de la aplicación y el acceso a los datos.

El diseño arquitectónico utilizado facilita el mantenimiento del código y la escalabilidad de la aplicación. Asimismo, esta arquitectura permite crear múltiples representaciones para el mismo conjunto de datos con relativa facilidad, esto, a través de la reutilización de componentes. Por ejemplo, más de una vista puede hacer uso de un mismo controlador para acceder y manipular los datos.

La separación de responsabilidades permite, a su vez, realizar cambios rápidamente y facilita la realización de pruebas unitarias. Esta arquitectura permite a los desarrolladores identificar fácilmente en que capa debe realizarse que cambio para lograr el resultado deseado.

De igual manera, en este capítulo, se elaboraron el diagrama de clases y los diagramas de caso uso. Este conjunto de diagramas son una abstracción del comportamiento de la aplicación y de cómo los usuarios interactúan con la misma.

Asimismo, se realizó la planificación de las iteraciones. Se definieron que procesos y en cuanto tiempo se iba a desarrollar cada una de estas.

Capítulo 5

En este capítulo se definen el conjunto de iteraciones que se siguieron durante el desarrollo de la aplicación. Para cada una de estas, se muestran los diagramas de actividad relacionados a cada uno de los requisitos pertenecientes a dicha iteración. De igual manera, a través de un conjunto de imágenes acompañadas de una breve descripción, se muestra la implementación, el resultado del desarrollo de la aplicación. El orden en que fueron desarrollados los requerimientos tiene que ver con el hecho de que cada requerimiento, de cierta manera, tiene como prerrequisito los requerimientos anteriores a este. Por ejemplo, para que un usuario pueda ejecutar la acción descrita en el requisito 02 (iniciar sesión), el usuario tuvo que haber creado previamente una cuenta, y esto es únicamente posible si el requerimiento 01 se encuentra desarrollado.

5.1 Primera Iteración

En esta primera iteración se desea cumplir con los requerimientos 01, 02, 03, 04, 06 y 11.

5.1.1 Refinamiento de los requisitos

Para lograr el completo refinamiento de los requisitos se decidió utilizar diagramas de actividades UML ya que estos permiten obtener una descripción detallada del funcionamiento de cada uno de los requisitos listados en la actual iteración.

5.1.1.1 Diagramas de actividades

La figura 8 muestra el diagrama de actividad relacionado con el proceso de registro para un usuario que no posee una cuenta en la aplicación.

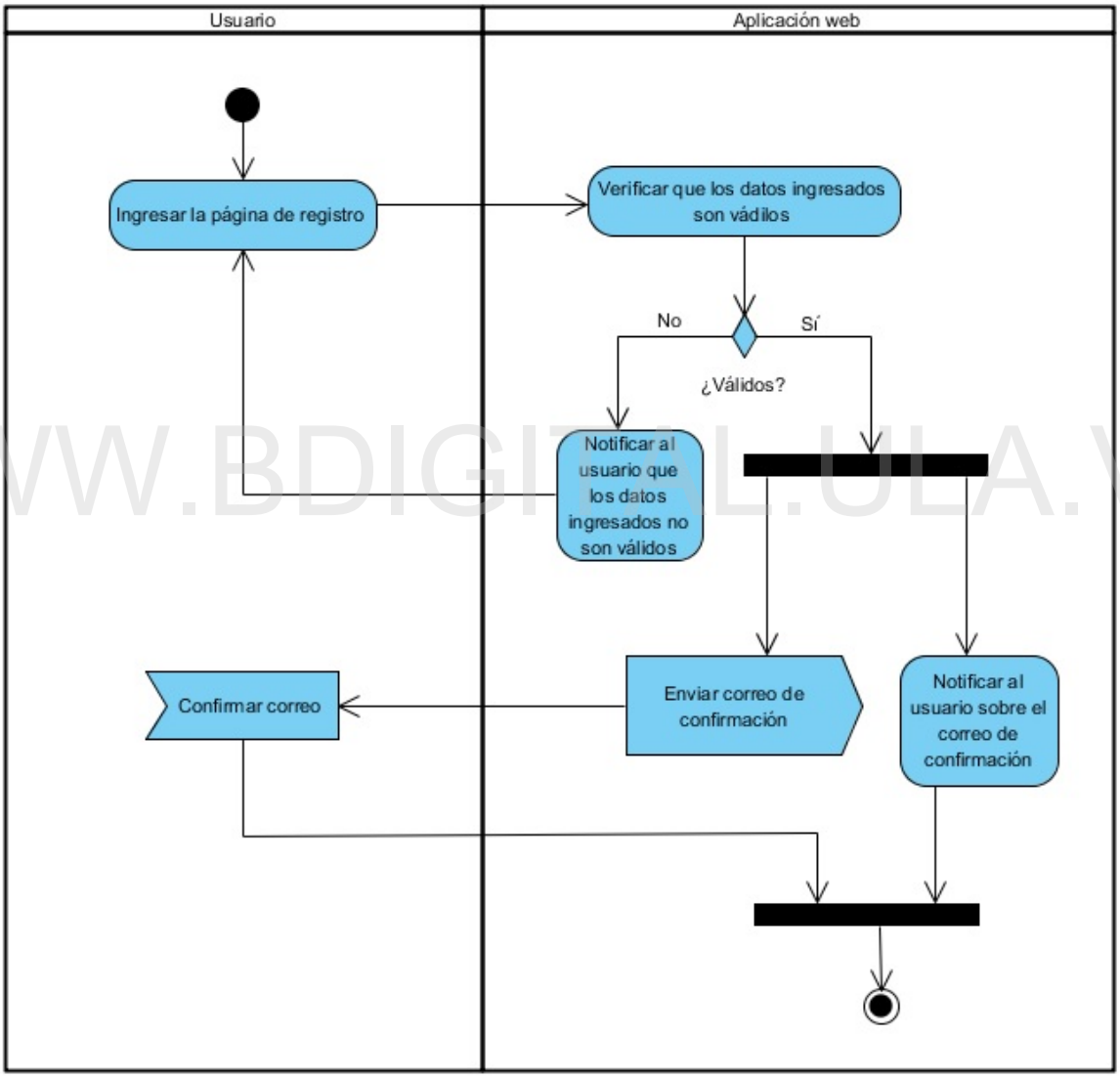

Figura 8. Diagrama de actividad UML– Crear una cuenta en el sistema

El diagrama de la figura 9 muestra el flujo de actividad que se desencadena cuando un usuario registrado se autentica o intenta autenticarse en la aplicación.

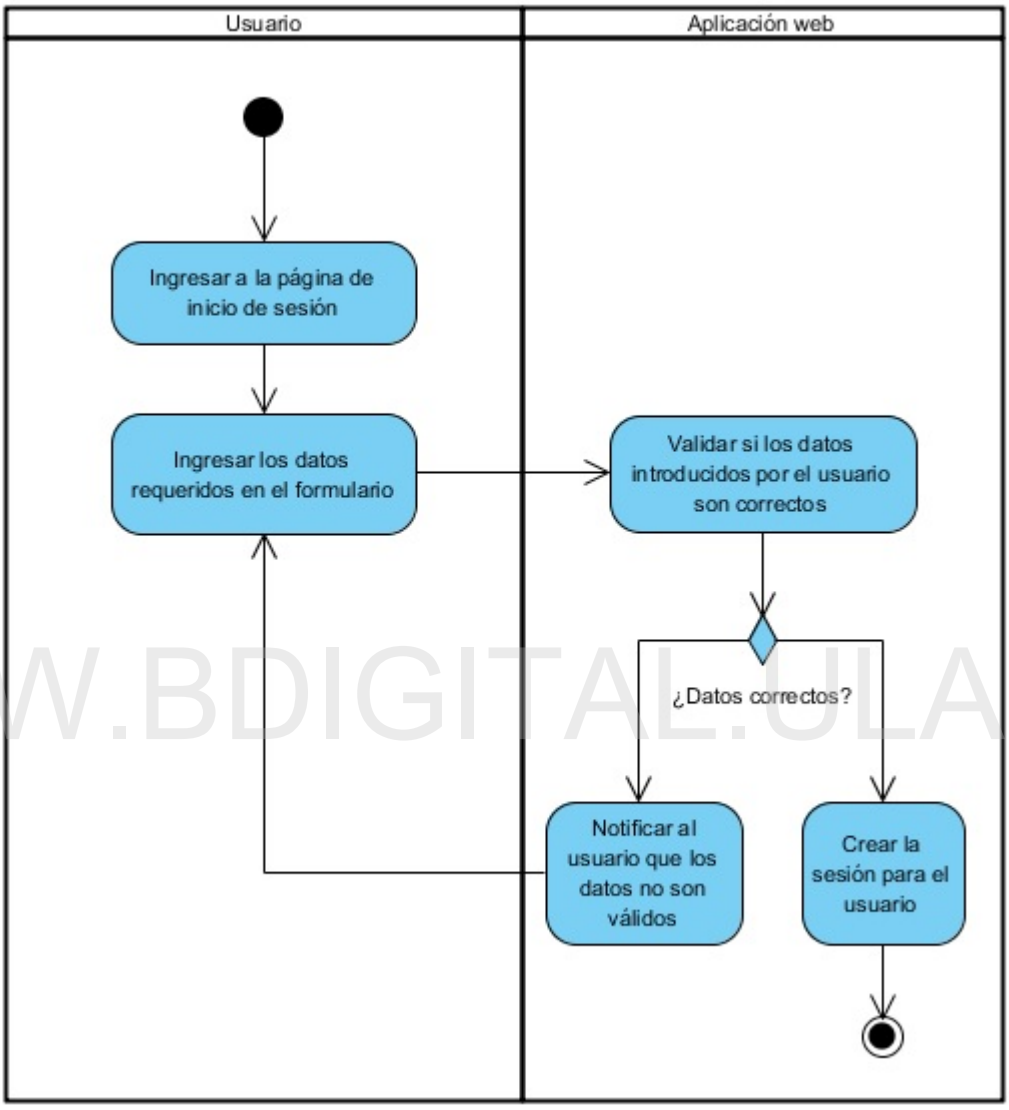

Figura 9. Diagrama de actividad UML– Iniciar sesión en el sistema

El diagrama de la figura 10 muestra las acciones relacionadas con el proceso de publicación de un viaje.

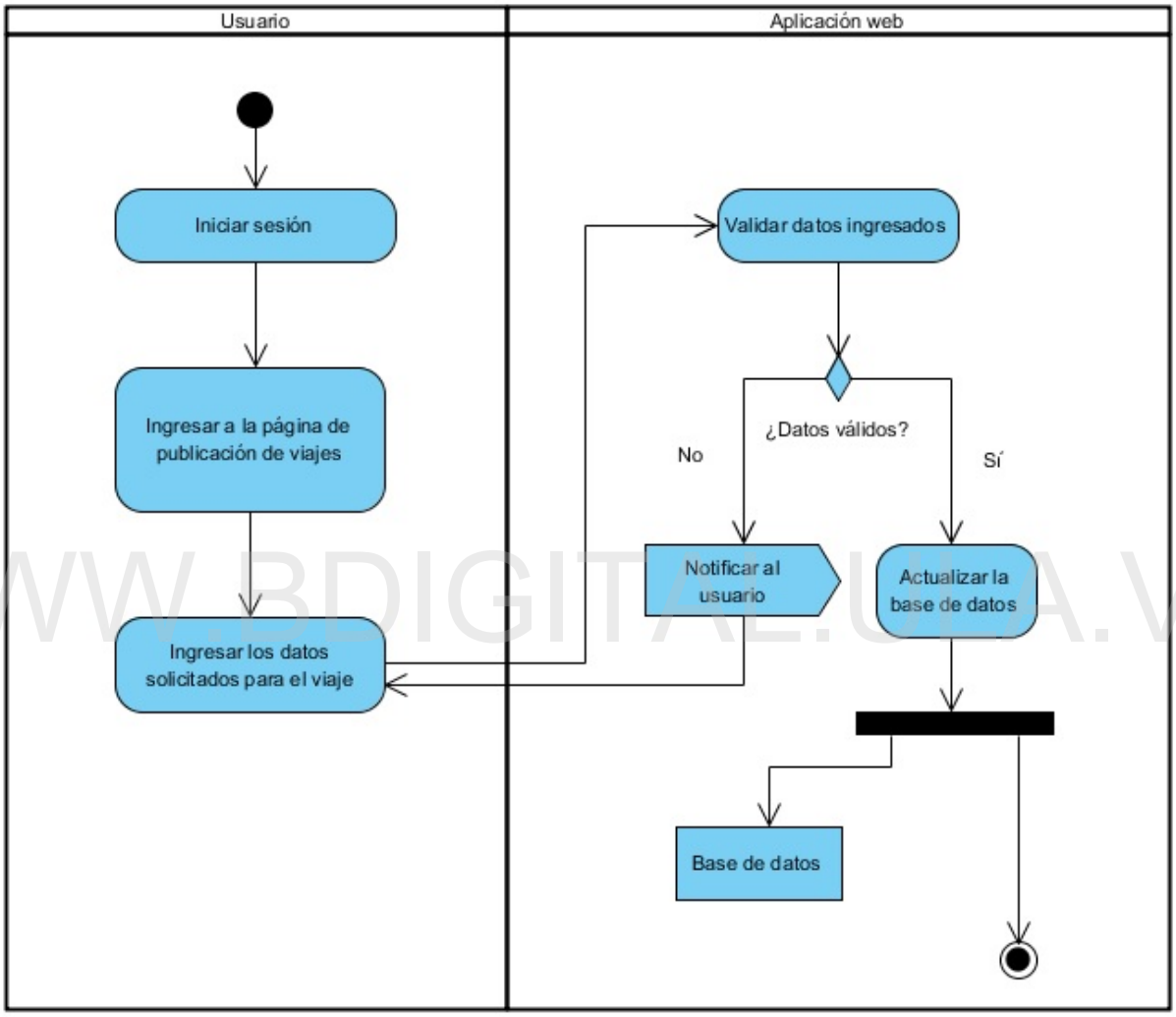

Figura 10. Diagrama de actividad – Publicar un viaje

La figura 11 representa el diagrama de actividad que describe el proceso de búsqueda de viajes realizada por un usuario de la aplicación web. Este proceso puede ser desencadenado por un usuario autenticado o no autenticado.

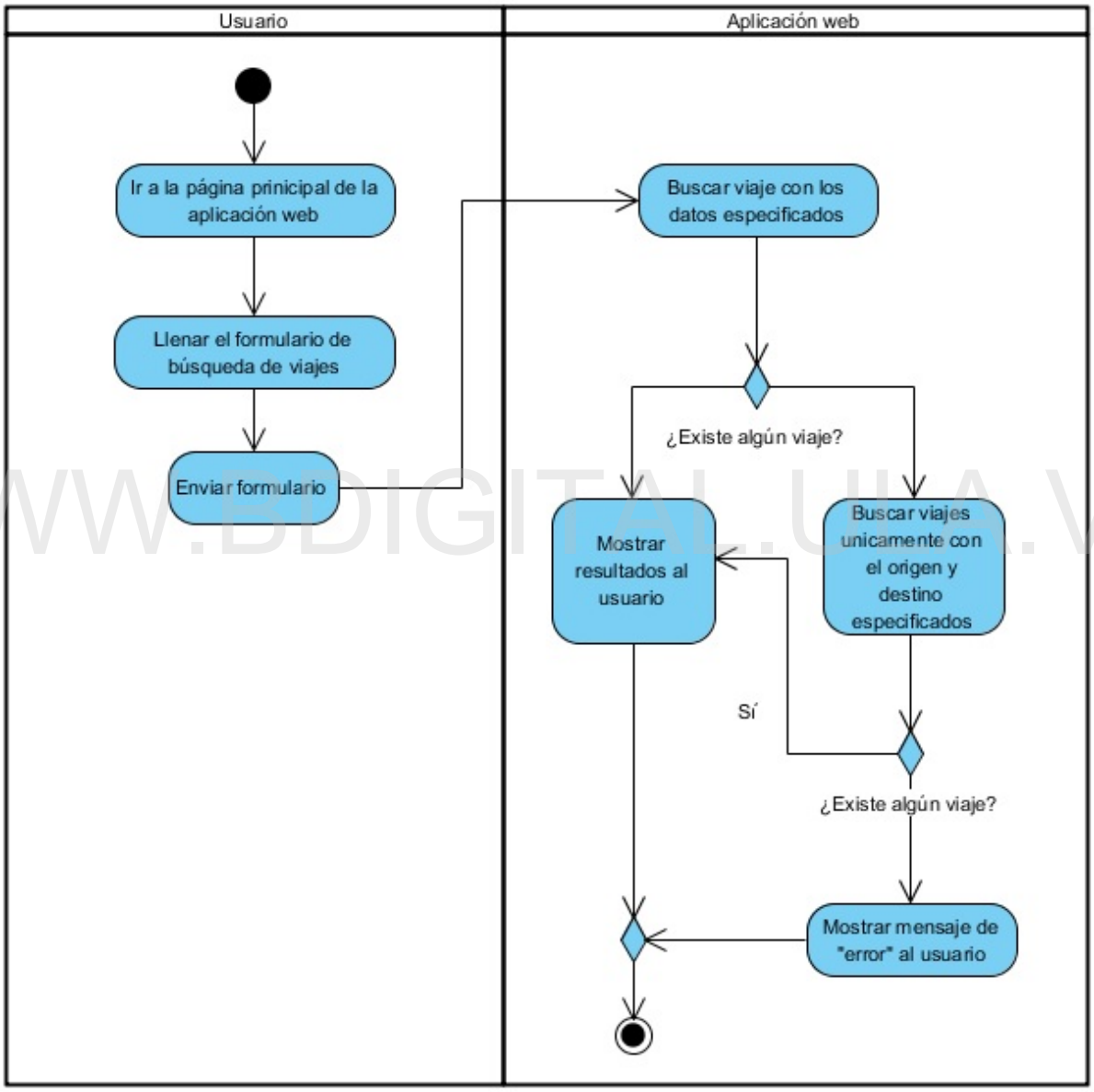

Figura 11. Diagrama de actividad – Buscar un viaje

La figura 12 representa el diagrama de actividad relacionado con el proceso de finalizar sesión. Acción que únicamente puede ser realizado por un usuario que previamente tiene una cuenta en el sistema y ha iniciado sesión.

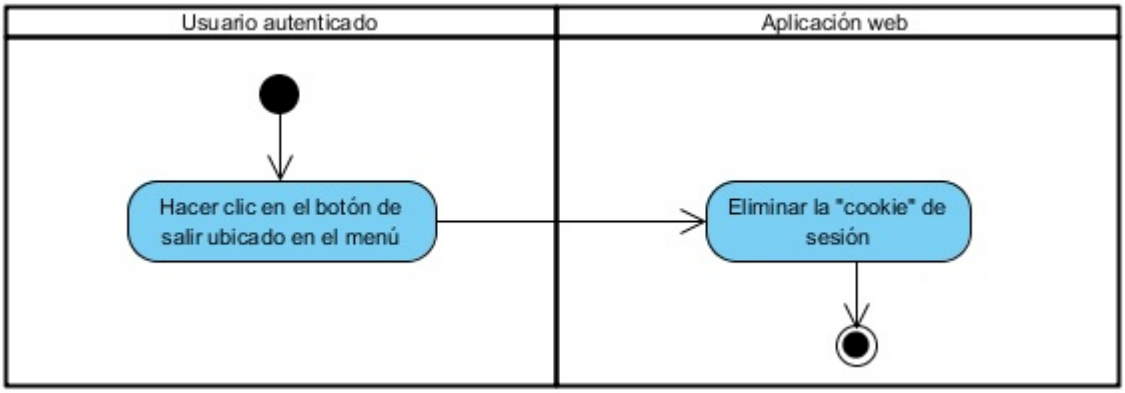

Figura 12. Diagrama de actividad – Finalizar sesión

El diagrama de actividad de la figura 13 describe el proceso de realizar una pregunta en la página de un viaje publicado. Este proceso lo desencadena un usuario autenticado.

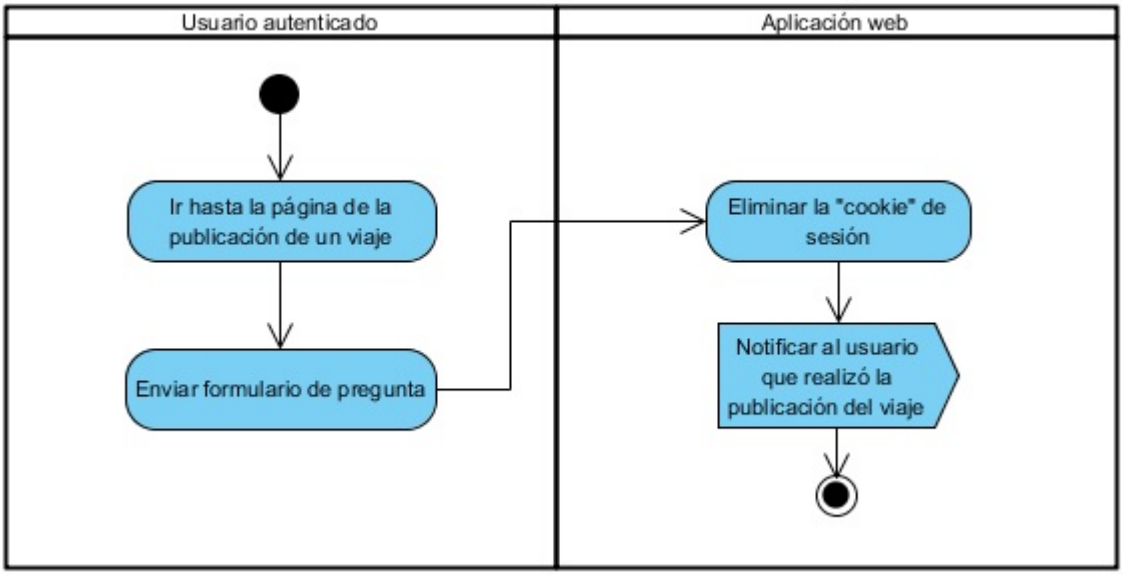

Figura 13. Diagrama de actividad – Realizar pregunta

5.2 Segunda Iteración

La segunda iteración tiene por objetivo cumplir los requerimientos 05, 07, 09 y 13 en una duración de 05 semanas.

5.2.1 Refinamiento de los requisitos

Esta iteración usa de igual manera los diagramas de actividades para refinar los requisitos.

5.2.1.1 Diagramas de actividades

La figura 14 muestra, a través de un diagrama de actividad, el proceso que ocurre cuando un usuario envía una solicitud de reserva de asiento.

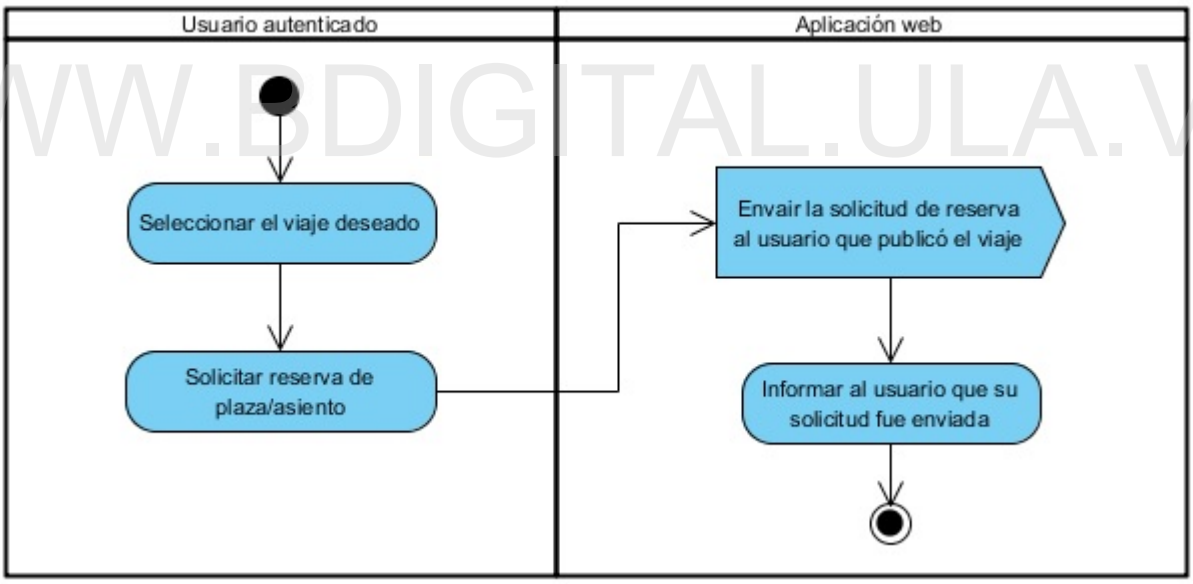

Figura 14. Diagrama de actividad – Enviar solicitud de reserva de asiento

La figura 15 describe el proceso de gestión de solicitud de reserva de asiento. Como prerrequisito a este proceso el usuario que desencadena el proceso tiene que haber recibido previamente una solicitud sobre la cual va a operar (aceptar o rechazar dicha solicitud).

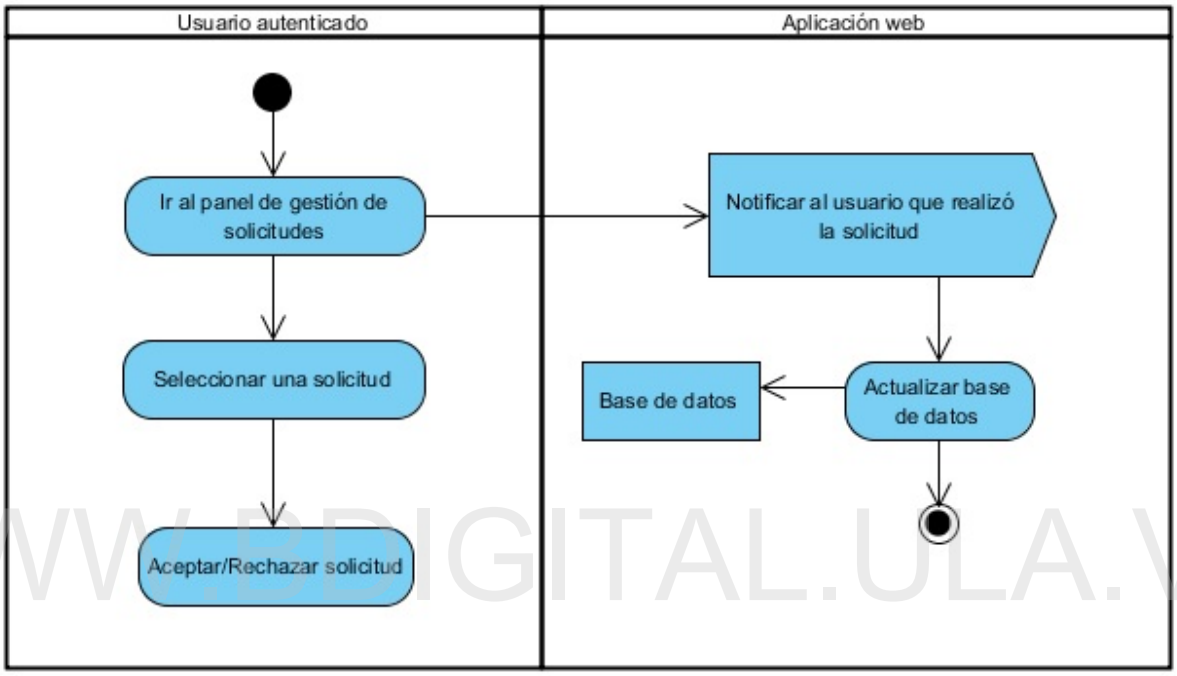

Figura 15. Diagrama de actividad – Gestionar solicitudes de reserva de asientos

El diagrama de actividad mostrado en la figura 16 muestra el conjunto de acciones que se realizan cuando el proceso de cancelación de un viaje es desencadenado. Este proceso lo puede realizar únicamente un usuario previamente autenticado con una publicación de viaje activa.

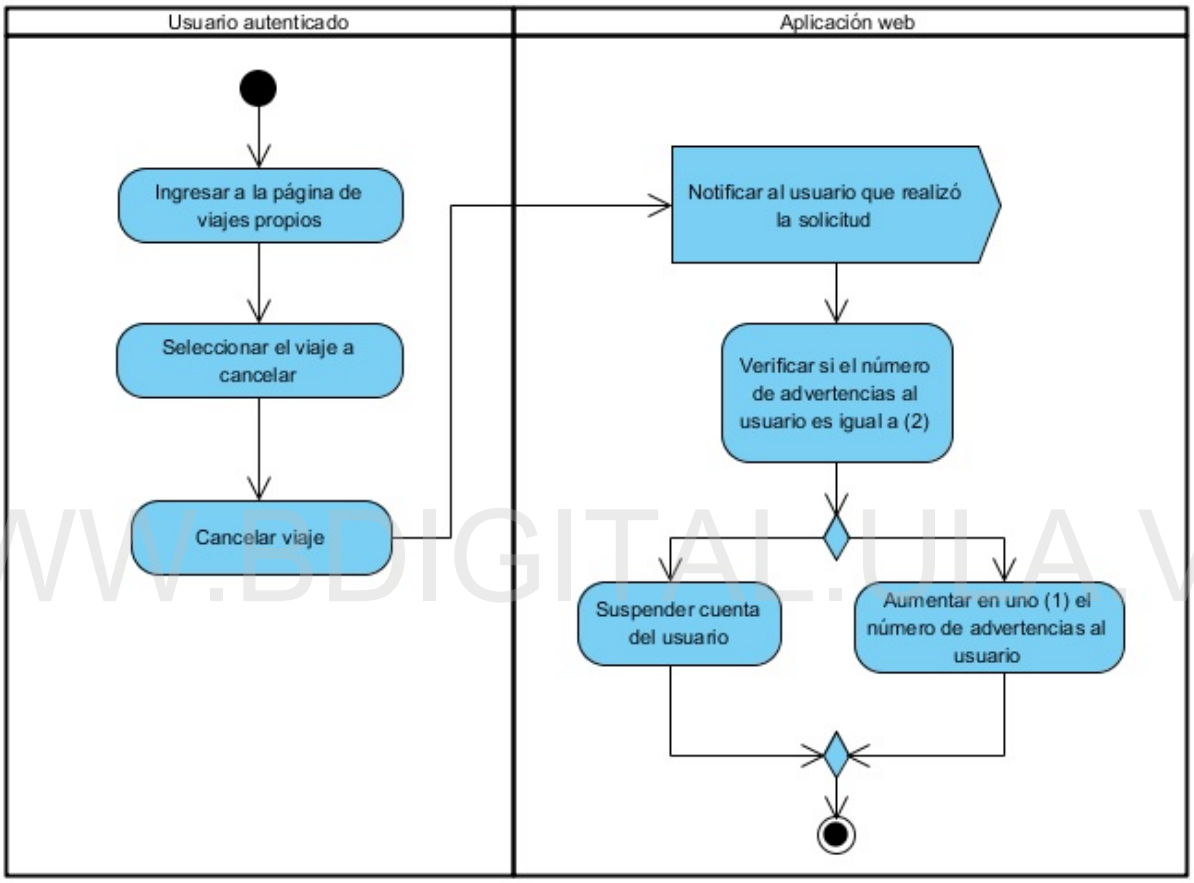

Figura 16. Diagrama de actividad – Cancelar un viaje publicado

La figura 17 representa el proceso de edición de perfil. Este proceso permite al usuario modificar la información personal relacionada a su cuenta.

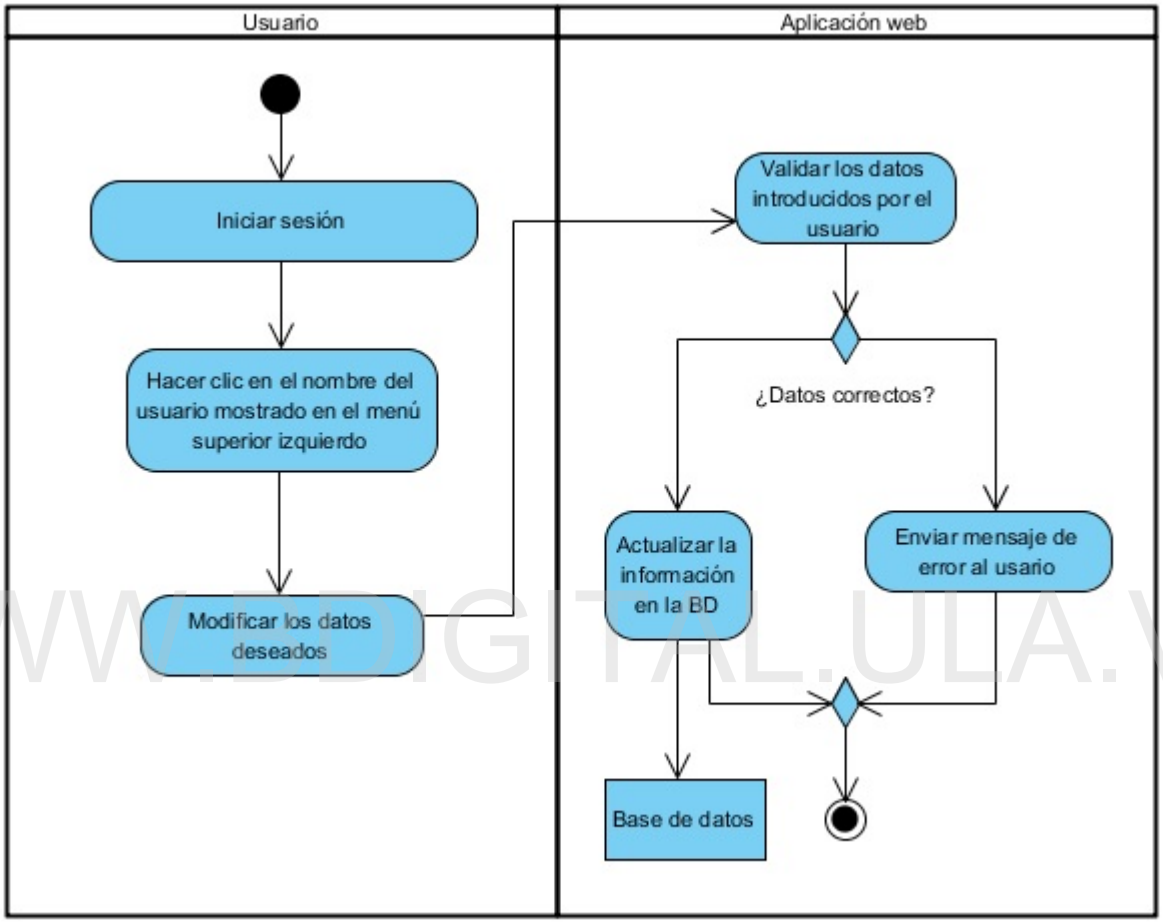

Figura 17. Diagrama de actividad – Editar perfil

5.3 Tercera Iteración

La segunda iteración tiene por objetivo cumplir los requerimientos 12, 13, 14 y 15 en una duración de 08 semanas.

5.3.1 Refinamiento de los requisitos

Esta iteración usa de igual manera los diagramas de actividades para refinar los requisitos.

5.3.1.1 Diagramas de actividades

El diagrama de actividad mostrado a continuación, en la figura 18, describe el proceso de calificación de usuario.

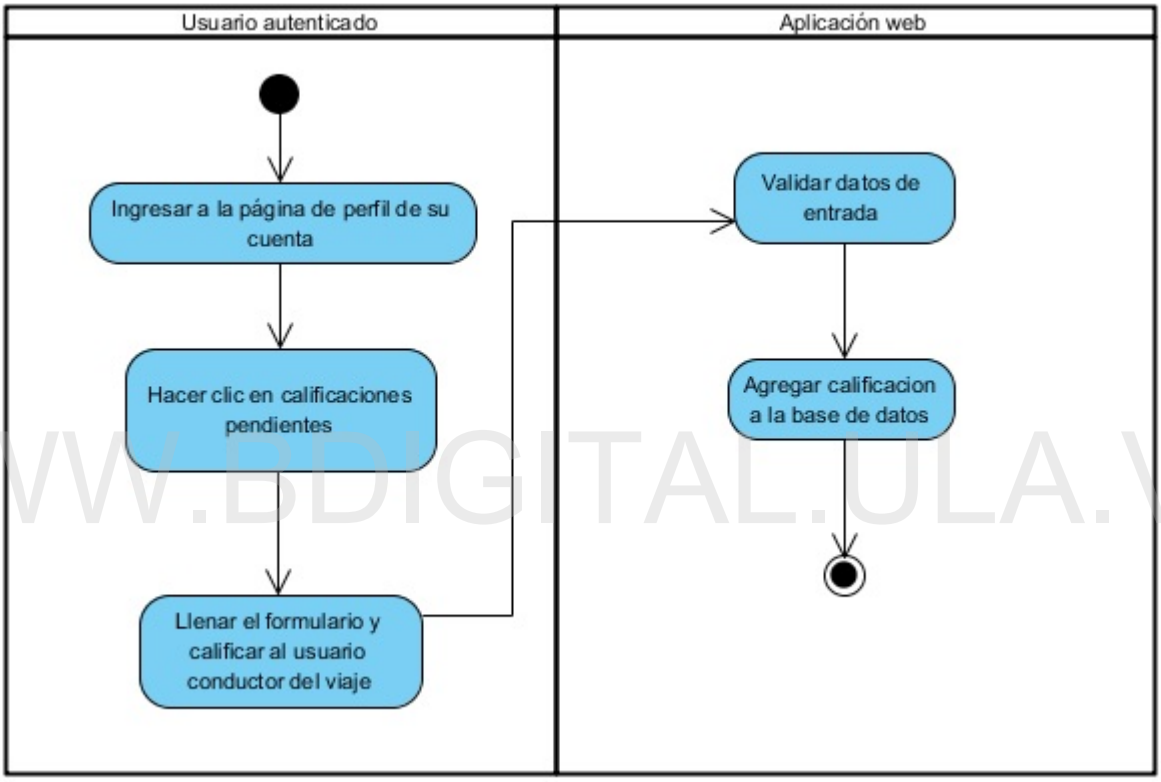

Figura 18. Diagrama de actividad – Calificar a un usuario

La figura 19 describe gráficamente, a través de un diagrama de actividad el proceso de darse de baja, es decir, cuando un usuario quiere eliminar su cuenta del sistema.

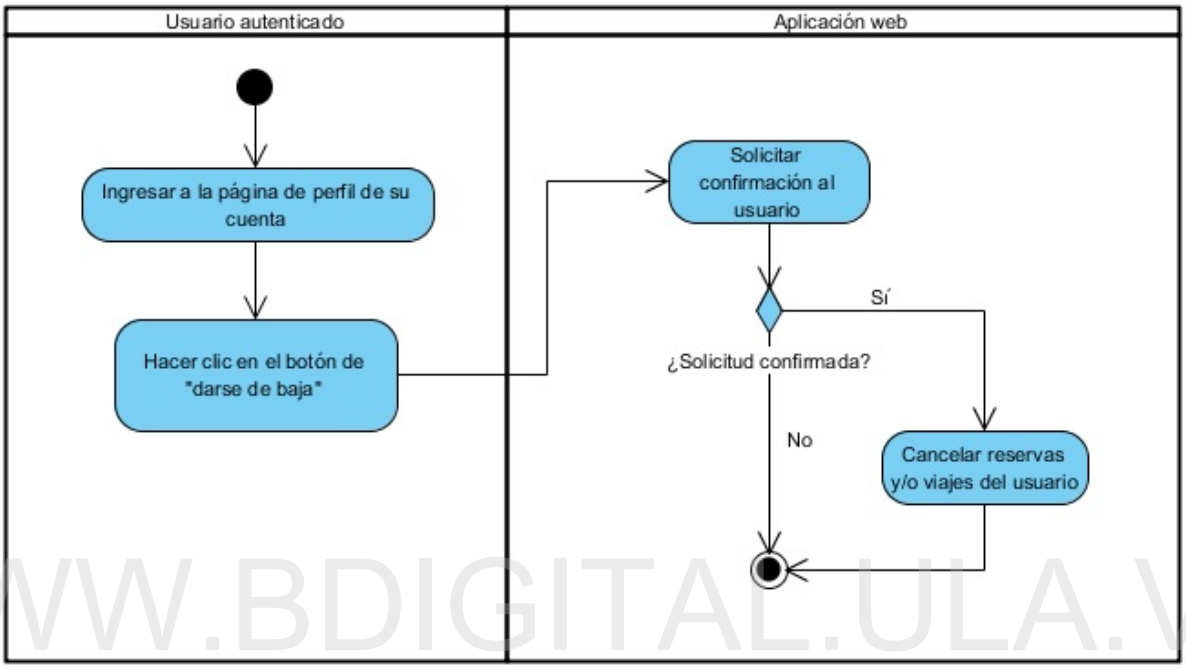

Figura 19. Diagrama de actividad – Darse de baja

La figura 20 el diagrama de actividad de cancelación de viaje de terceros, muestra el conjunto de acciones relacionadas al proceso. Este proceso puede ser desencadenado únicamente por un usuario moderador.

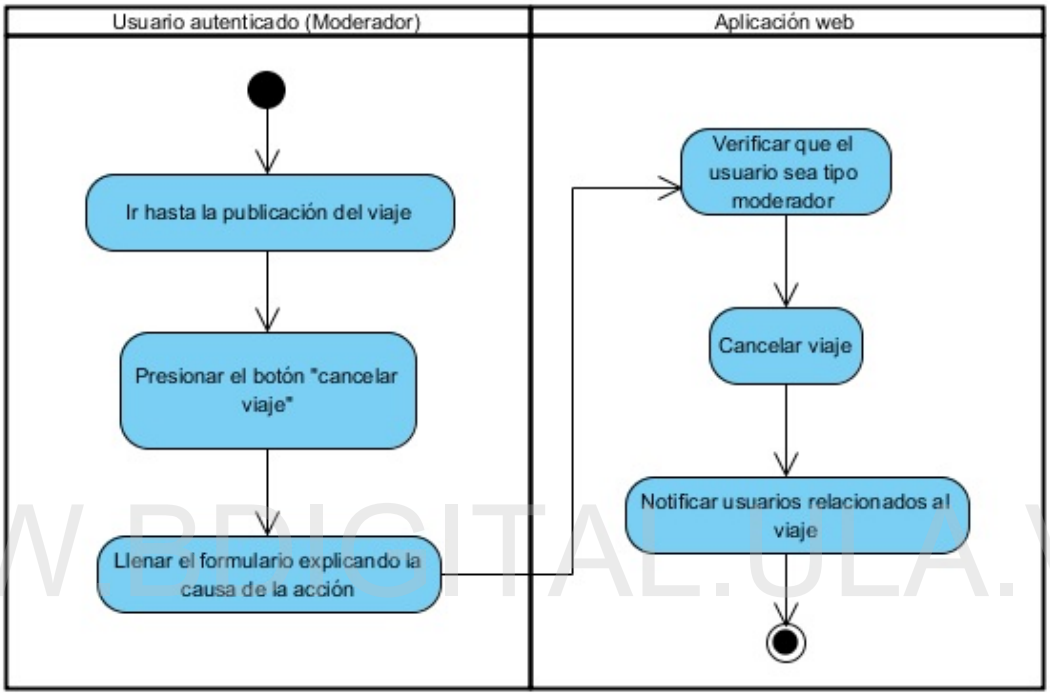

Figura 20. Diagrama de actividad – Cancelar viaje de terceros

5.4 Implementación

A continuación, se muestra el resultado de la implementación de los distintos requerimientos a través de un conjunto de capturas de pantalla de la aplicación, acompañadas de una breve descripción, sin dejar de lado lo técnico.

Registro de usuario: esta pantalla muestra el formulario a través del cual una persona puede registrar una cuenta en el sistema para posteriormente autenticarse y poder hacer uso del resto de las funcionalidades que ofrece la aplicación.

El formulario cuenta con validaciones realizadas del lado del cliente con HTML5 y del lado del servidor. Estas validaciones obligan al usuario a insertar solo datos válidos, así como también, evita que

se inserten datos duplicados, por ejemplo, dos usuarios distintos utilizando el mismo correo o un mismo número de cédula.

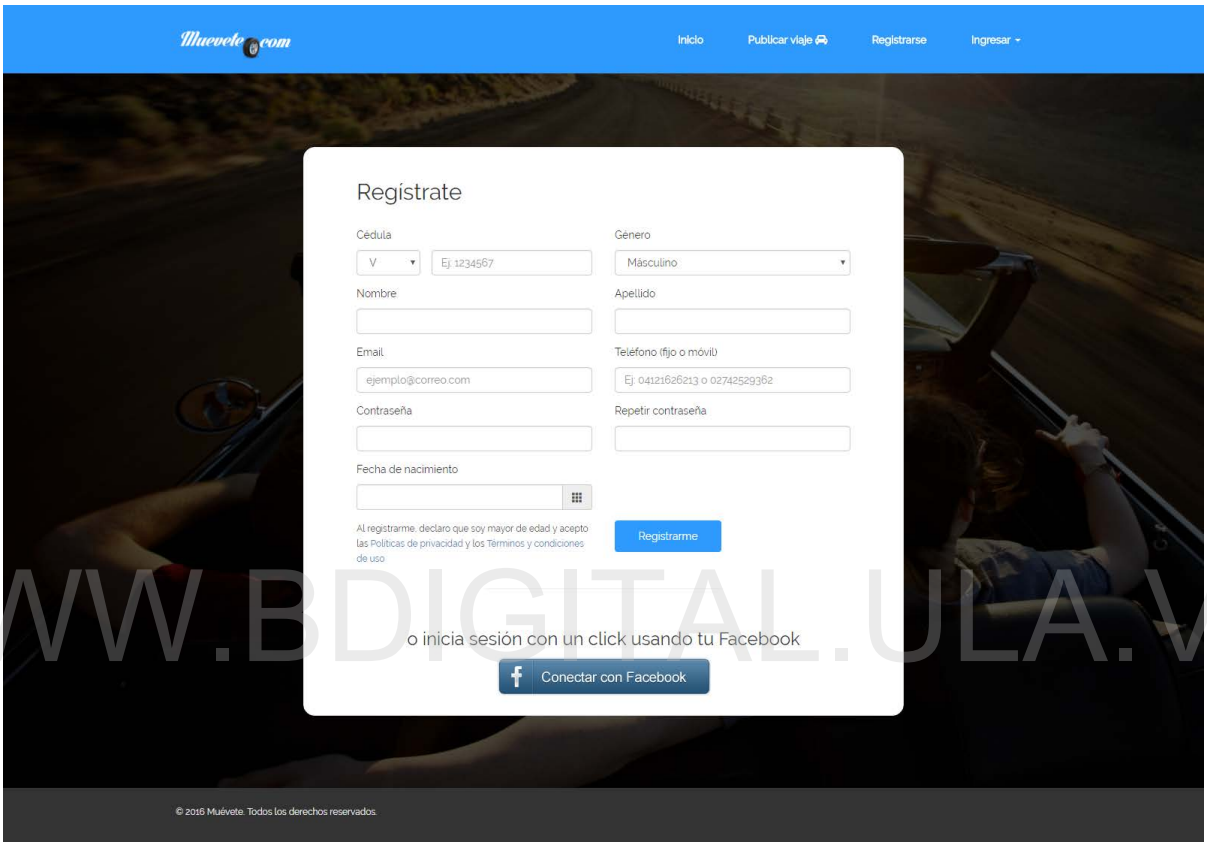

Figura 21. Pantalla de registro de usuario

Publicación del viaje: la vista de publicación del viaje permite a un usuario autenticado publicar un viaje haciendo uso del formulario. El formulario de publicación de un viaje se encuentra dividido en tres secciones. En la primera sección se definen datos como el lugar de partida y de llegada, datos del vehículo y la hora y fecha en que iniciará el recorrido.

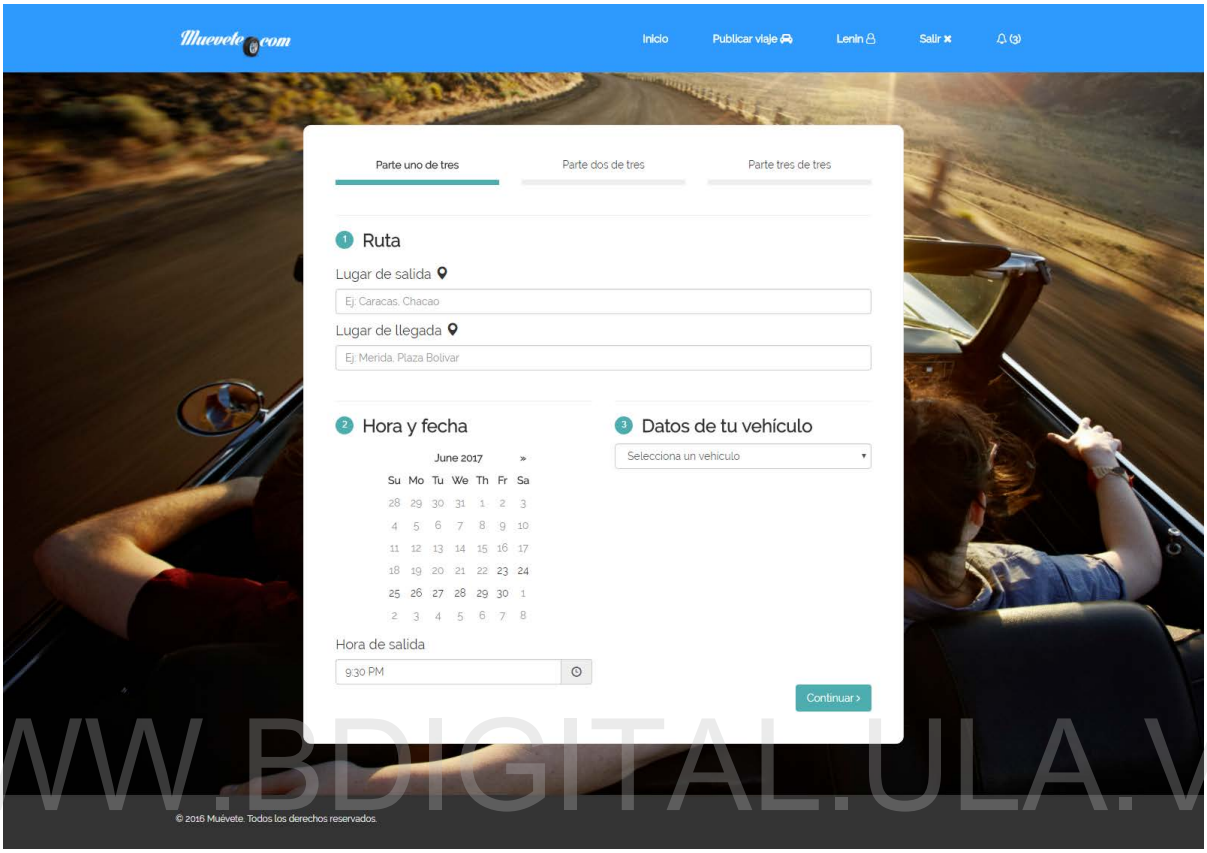

Figura 22. Pantalla de publicación de viaje, parte 1

La segunda sección permite seleccionar el número de asientos que ofertará en el viaje, el precio que debe pagar un usuario para ser pasajero, el método de pago e información adicional que considere pertinente.

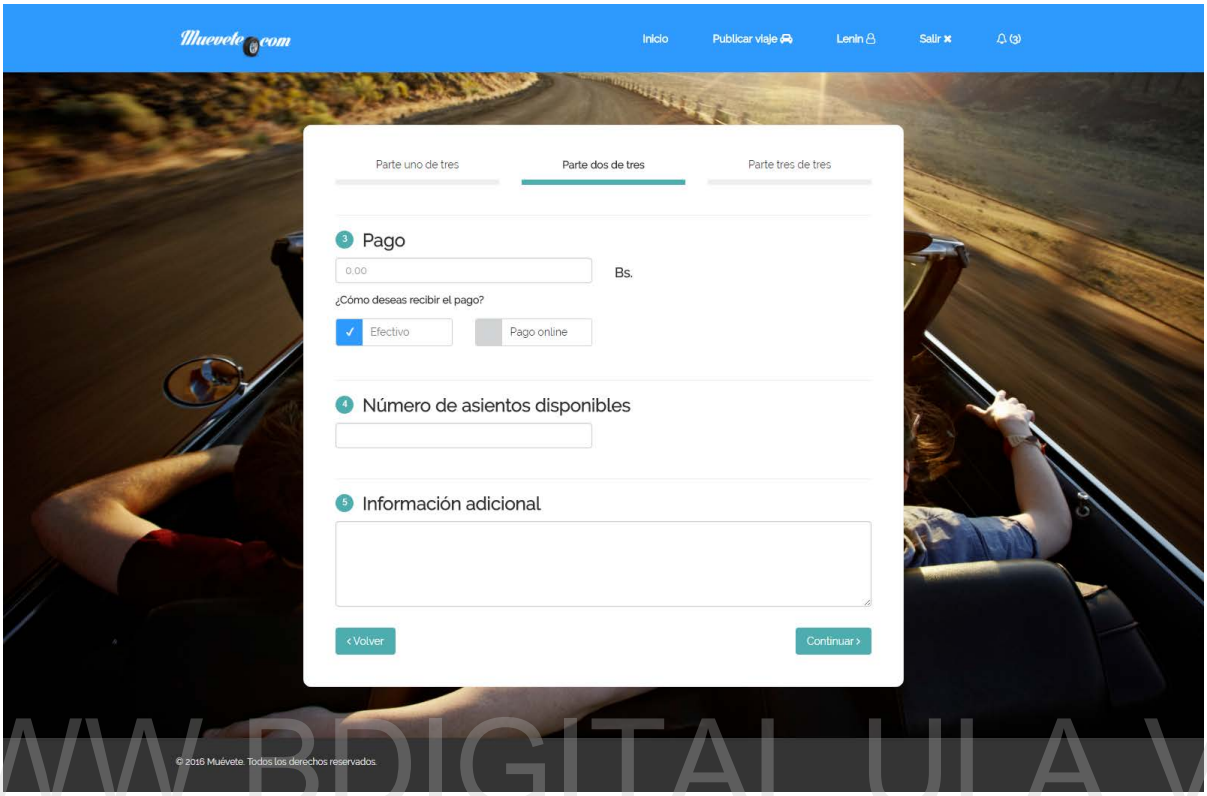

Figura 23. Pantalla de publicación de viaje, parte 2

La tercera y última sección permite al conductor seleccionar las preferencias de viaje, es decir, si viaja con aire acondicionado, con música, si permite llevar mascotas, entre otros.

Este formulario de igual manera cuenta con validaciones del lado del cliente y del lado del servidor.

Los campos de "lugar de salida" y "lugar de llegada" cuentan con un sistema de predicción haciendo uso de la API de Google Maps. Cuando se envía el formulario al servidor, a través de la API se obtienen las coordenadas en latitud y longitud de ambos lugares y se almacenan en la base de datos para posteriormente utilizarlos en la página de detalle del viaje.

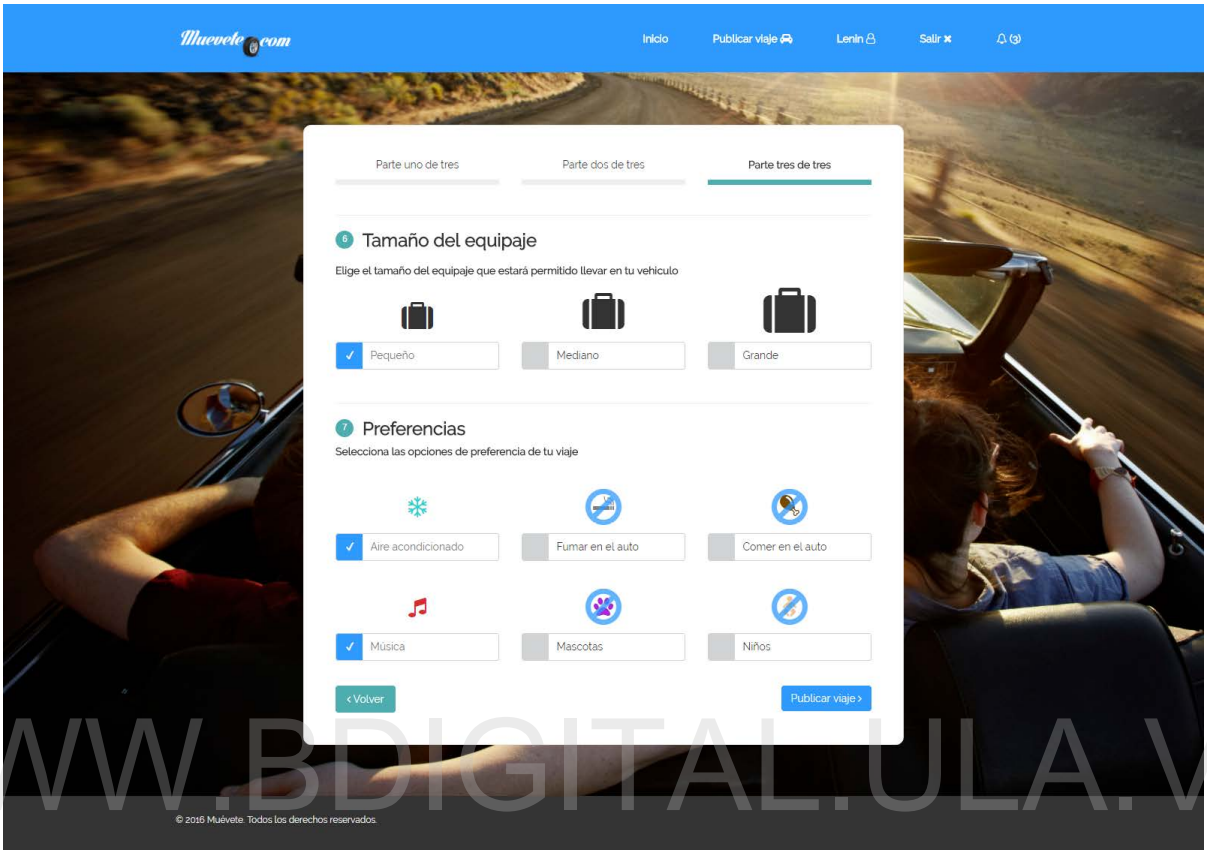

Figura 24. Pantalla de publicación de viaje, parte 3

Búsqueda de viaje: a través de la página inicial de la aplicación se puede observar un formulario lineal a través del cual se pueden consultar los viajes que coincidan con el lugar de salida, lugar de llegada y la fecha especificada.

En caso de que no exista algún viaje para la fecha indicada, la aplicación web devolverá al usuario los viajes que coincidan solamente con el lugar de salida y de llegada, es decir, sin tomar en cuenta la fecha. Esto permitirá al usuario visualizar las distintas alternativas con las que puede contar.

De no existir ningún tipo de coincidencia se mostrará al usuario un mensaje de que no se consiguieron viajes y que intente en otro momento o usando distintas localidades.

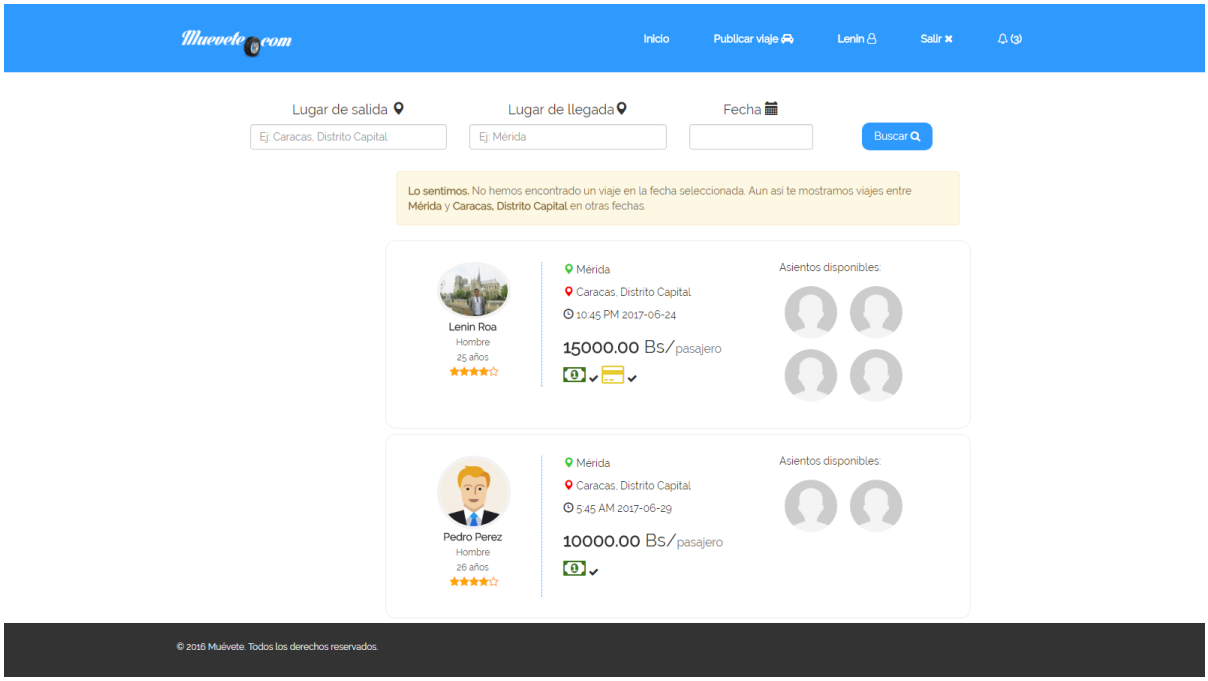

Figura 25. Pantalla de búsqueda de viaje

Detalles del viaje, preguntas y respuestas: la página de detalle de un viaje es la página que se muestra cuando se ingresa a un viaje. A partir de esta pantalla el usuario tiene la posibilidad de ver toda la información detallada del viaje, incluyendo los pasajeros actuales, ver preguntas de terceros, y, en caso del usuario estar autenticado, este podrá enviar preguntas y solicitudes de reserva de asiento.

En esta página se hace uso de la coordenada en latitud y longitud de las ciudades desde y hasta la cual viaja el usuario, guardadas cuando se realiza el envío del formulario de publicación de un viaje. Haciendo uso de la API de Google se calcula la distancia entre ambos puntos y el tiempo estimado que tomará realizar el recorrido. De igual manera se muestra una imagen del mapa de Venezuela y la ruta automáticamente calculada haciendo uso de la API.

Cuando una pregunta o solicitud de reserva es enviada, el sistema crea una notificación almacenando un mensaje predeterminado y guardando la hora en que dicha acción fue realiza. Esta notificación es enviada al conductor del viaje para que este esté al tanto y pueda responder a dicha acción. Asimismo, cuando el conductor responde una solicitud o una pregunta, una notificación es enviada a la persona que realizó la pregunta o la solicitud de reserva de asiento.

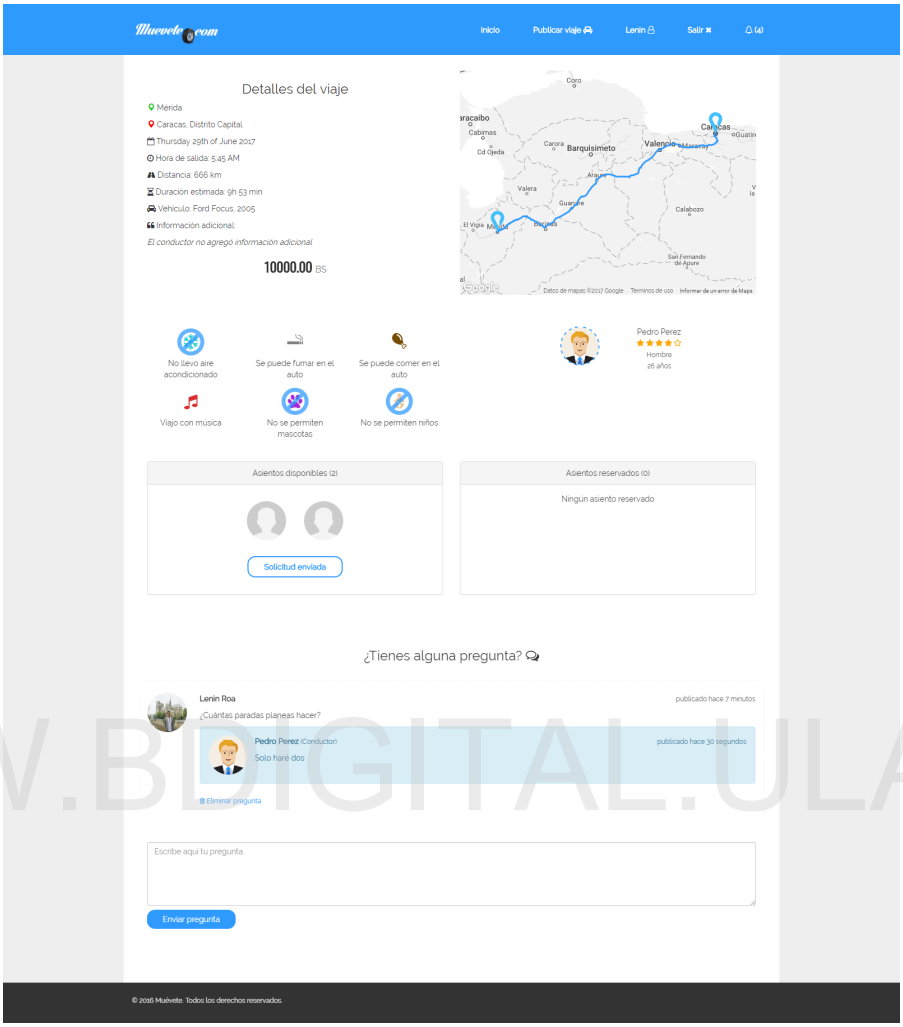

Figura 26. Página de detalles de un viaje

Sistema de reputación: los usuarios pueden calificar a otros usuarios que participaron como conductores en un viaje en el que ellos fueron pasajeros. A través de la opción de "calificaciones pendientes/realizadas" se listan todas las calificaciones realizadas o por realizar. Tras calificar al usuario con una puntuación entre 1 y 5 estrellas, además de un comentario, la calificación es almacenada en la base de datos y se promedia con el resto de las calificaciones existentes.

Antes de enviar la solicitud de reserva de asiento en un viaje, el usuario puede basarse en la reputación que tiene el usuario conductor para tomar la decisión de viajar con él o con otro usuario que posea una mejor reputación. La reputación es visible en la página de publicación del viaje bajo el nombre del conductor.

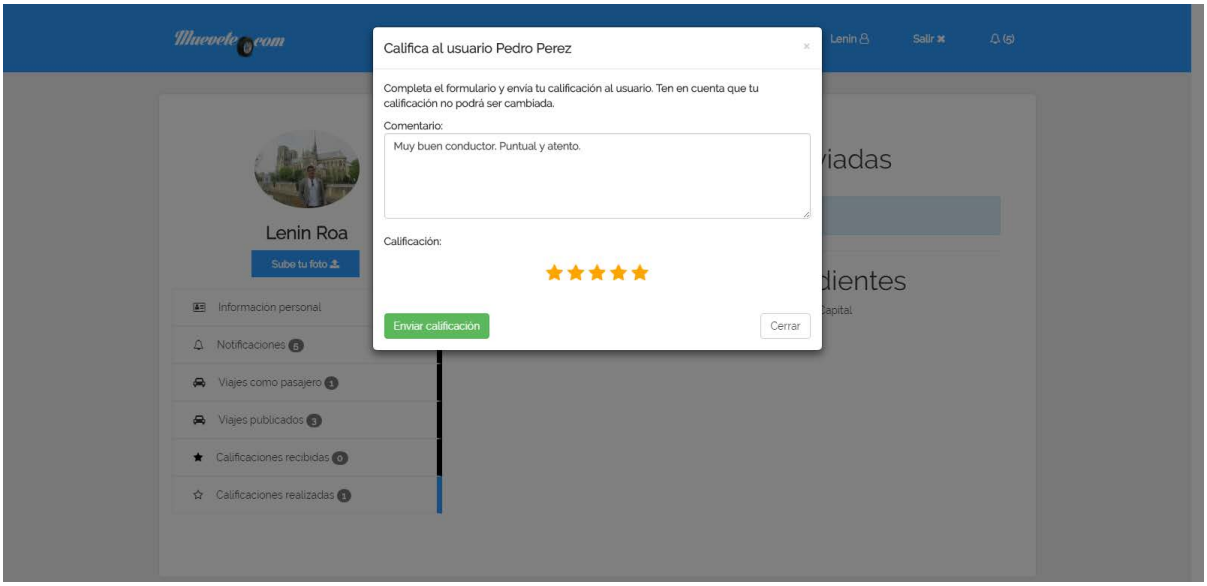

Figura 27. Calificación a usuario

Autenticación con Facebook: la autenticación de usuarios a través de Facebook es posible haciendo uso de la librería *Socialite* que a su vez facilita la comunicación con la API de Facebook. Esto permite al usuario iniciar sesión en Facebook y quedar automáticamente autenticado en la aplicación web.

Cuando el usuario decide autenticarse haciendo uso de la red social, la aplicación web lo re direccionará hasta Facebook. En caso de estar previamente autenticado en la red social, esta le solicitará permiso de proveer algunos de sus datos personales a la aplicación web. En caso de no estar autenticado, Facebook le pedirá que lo haga y luego solicitará el permiso para proveer sus datos.

La aplicación web verificará en la base de datos si ya existe un usuario con el email devuelto por Facebook. En caso de ser cierto, la aplicación web cederá el acceso al usuario, en caso contrario, se registrará un nuevo usuario que será únicamente autenticable a través de este método hasta que este acceda a la configuración de su cuenta y cree una contraseña para iniciar sesión posteriormente. La imagen de perfil en Facebook, así como también, otros datos devueltos por la red social (nombre, apellido, año de nacimiento) serán usados para completar el registro de la cuenta en la aplicación web.

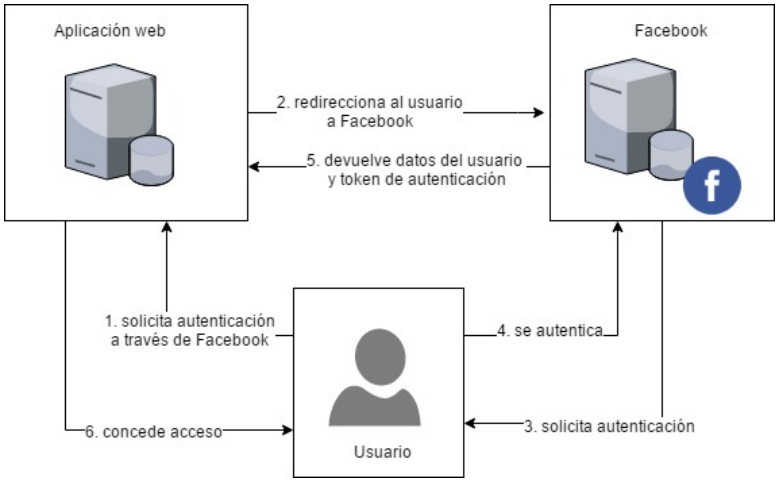

Figura 28. Autentificación con Facebook

Implementación de Google Maps API: tal como se mencionó anteriormente la API de Google Maps es utilizada para obtener las coordenadas de latitud y longitud de las localidades de origen y destino de los viajes.

Con cada pulsación de tecla en el formulario de búsqueda de viajes o en el formulario de publicación se envía una consulta AJAX a Google para que este realice una búsqueda en su base de datos y retorne los datos de interés, en este caso, el listado de las localidades existentes en una región determinada (en este caso Venezuela) junto con sus coordenadas.

Cada vez que un usuario visita la página de publicación de un viaje. La aplicación web realiza otra petición AJAX, esta vez con las coordenadas de la ciudad origen y la ciudad destino y un conjunto de datos para personalizar la apariencia del mapa que muestra la ruta entre ambas ciudades.

5.5 Resumen capítulo 5

En el presente capítulo se definieron las iteraciones, se realizaron los diagramas de actividades de las funcionalidades que iban a ser desarrolladas en cada una de las iteraciones y se mostró el resultado tras haber implementado cada uno de los requisitos.

En la primera iteración se desarrollaron los procesos 1, 2, 3, 4, 6 y 11, estos relacionados principalmente con las acciones básicas en la aplicación web, tal como, registrarse en el sistema, autenticarse, publicar un viaje, etc.

En la segunda iteración los requisitos cuyo comportamiento normal de la aplicación, tienen como prerrequisito, los procesos anteriores. Los procesos 5, 7, 9, 10 y 13 son los procesos desarrollados en la segunda iteración, permitiendo al usuario realizar acciones como cancelar un viaje, registrar un vehículo, enviar una solicitud de reserva de asiento y realizar una pregunta en la publicación de un viaje.

Finalmente en la tercera, los procesos restantes 8, 12, 14, 15 y 16, que dan un valor agregado a la aplicación, permitiendo al usuario autenticarse a través de redes sociales, facilitando el proceso de autenticación y añadiendo mayor confianza a los usuarios, darse de baja del sistema y eliminar preguntas de terceros siendo usuario moderador.

De la figura 21 a la figura 28 se puede apreciar la interfaz de usuario de la aplicación. Cada figura está acompañada de una descripción que resume brevemente la funcionalidad con la que esta está relacionada.

La unión de todas estas funcionalidades interrelacionadas de distintas maneras define la aplicación web en su totalidad. Una aplicación que permite a las personas que habitan en Venezuela ponerse en contacto con otros usuarios para compartir un viaje, cuyo origen y destino de este, es el mismo.

Capítulo 6

En este último capítulo se exponen las conclusiones, los resultados y las recomendaciones tras la realización del proyecto. Estas tres secciones dan por concluido el proyecto de grado donde se realiza un análisis general de lo que fue la realización del proyecto, se mencionan los resultados obtenidos y se exponen un conjunto de recomendaciones para futuros investigadores que deseen continuar con el trabajo añadiendo mejoras y nuevas funcionalidades a la aplicación.

6.1 Conclusiones

El desarrollo de aplicaciones web para buscar soluciones a problemas cotidianos es algo que se ha vuelto muy común en los últimos años. Es un proceso complejo en el que se debe tomar en cuenta gran cantidad de factores. Definir y tener claro el objetivo principal de la aplicación, así como también a qué tipo de usuarios estará dirigida y como estos interactuaran con la aplicación debe ser un requisito clave antes de iniciar con el proceso de desarrollo.

Existen diversas metodologías para llevar a cabo el desarrollo de una aplicación web. Los tres patrones básicos en las metodologías de desarrollo de software son *Waterfall*, *Prototyping* y *Spiral.*

El método Blue Watch es un método de tipo *Spiral* inspirado en la metáfora del reloj de pulsera donde el ciclo de incrementos aumenta el ciclo de versión y este a su vez el ciclo de la aplicación.

Tal como se mencionó anteriormente, es indispensable tener claro los requisitos y objetivos que se desean alcanzar. La elaboración de los diagramas de actividad y de "casos de uso" debe ser explícitos y concisos sin dejar espacio a ambigüedades que puedan perjudicar posteriormente al desarrollador.

El diagrama de clases debe ser planeado cuidadosamente puesto que una incorrecta elaboración seguida de una implementación codificada puede causar grandes pérdidas de tiempo y recursos.

El desarrollo de una aplicación web independientemente de su nivel de complejidad no es algo que deba ser tomado a la ligera. Llevar a cabo la creación de una aplicación web demanda gran cantidad de conocimientos, creatividad y tiempo.

Para lograr llevar a cabo todos los objetivos se debe realizar una planificación y una investigación exhaustiva de cómo alcanzar ese resultado, analizando las ventajas y desventajas de la

forma en que se planea lograr el objetivo y posibles problemas que podrían presentarse al momento de desarrollar las funcionalidades.

La planificación de las iteraciones debe ser trazada con cautela, cuidando que los requisitos de cada iteración sean metas alcanzables y realistas. Cuando un requisito está mal elaborado, bien sea porque puede ser descompuesto a su vez en otros requisitos o es irrealizable, puede privar al desarrollador de avanzar a través de las iteraciones u otros requisitos y haciéndolo perder tiempo valeroso.

PHP es un poderoso lenguaje de programación orientado a soluciones web. Laravel, el *framework* que está construido sobre este lenguaje permite un desarrollo rápido, ahorrando tiempo y manteniendo el código organizado.

Laravel posee una amplia comunidad de desarrolladores y una amplia documentación oficial, por lo tanto, es fácil conseguir soluciones a problemas comunes que se presentan a los programadores.

Eloquent, el ORM de Laravel, permite una fácil interacción con la base de datos a través de un esquema orientado a objetos, es decir, permite al desarrollador tratar los registros de la base de datos como si de objetos se tratasen, facilitando el procesamiento de los datos y su manipulación.

Cuando se planea desarrollar una aplicación web que sea responsiva y adaptable a múltiples dispositivos independientemente de su resolución es importante seguir la filosofía *mobile first* creada por Luke Wrobleski. A simple vista podemos observar que la mayoría de los usuarios que acceden a internet, en la actualidad, lo hacen a través de un dispositivo móvil. Por lo tanto, es importante, que la interfaz de usuario luzca bien, agradable al ojo del usuario, fácil de manejar y que concluya en una buena experiencia de usuario.

Además de esto, para el desarrollador es más fácil iniciar el desarrollo de la interfaz de usuario pensando en dispositivos con resoluciones pequeñas (teléfonos celulares y tabletas) y luego trasladar esta interfaz a dispositivos con grandes resoluciones (laptops y desktops). Esto debido a que filosóficamente desarrollar siguiendo este esquema permite lograr un "mejoramiento progresivo", es decir, el desarrollador pone su mayor esfuerzo en la presentación móvil, enfocándose en lo que es estrictamente necesario y mejorar su presentación, agregando funcionalidades extras a medida que el rango de visualización lo permita. Hacerlo en sentido contrario genera como resultado una "degradación agraciada", es decir, el desarrollador empieza implementando una interfaz y funcionalidades que deben ser reducidas gradualmente para adaptarse a una resolución cada vez menor.

6.2 Resultados

Todos los objetivos, a diferencia de uno solo, lograron cumplirse, cubriendo así, la mayoría de las expectativas que se tenían sobre el proyecto.

Como resultado general se obtuvo una aplicación web completamente funcional que permite a los usuarios registrarse, publicar viajes y contactar con otros conductores que viajan de una ciudad a otra en territorio venezolano y podrían estar dispuestos a intercambiar un asiento en su vehículo a cambio de una retribución económica.

Los usuarios conductores pueden personalizar las características de su viaje a dando respuesta a las comunes interrogantes que podrían tener los pasajeros, por ejemplo, si el vehículo cuenta con aire acondicionado o si está permitido viajar con mascotas y/o niños. De igual manera, la publicación del viaje puede ir acompañada de un mensaje personalizado con información que el conductor considere relevante. Adicionalmente, la aplicación web a su vez cuenta con un sistema de preguntas y respuestas. Esto es, los usuarios conductores del viaje pueden responder a preguntas realizadas por personas que están interesadas en viajar como pasajeros.

Se implementó satisfactoriamente la API de Google Maps que permite calcular la distancia y tiempo aproximado que le tomaría al conductor manejar de una ciudad origen a una ciudad destino. Trazar en la imagen de un mapa de Venezuela la ruta a seguir, así como también, utilizar el predictor de texto en los formularios de búsqueda para predecir los nombres de las ciudades.

La implementación de un sistema de pago online como MercadoPago decidió no implementarse pues contradecía totalmente la filosofía de lo que es el correcto *user experience*. MercadoPago exige al usuario tener una cuenta registrada en su portal web y agregar dinero a ésta para poder hacer uso del mismo. Esto podría haber ocasionado muchos problemas a un posible usuario, puesto que si una persona no era usuario previo de MercadoPago se iba a ver obligado a crear una cuenta en este portal para poder realizar pagos a través de la aplicación web.

También logró implementarse un sistema de reputación que permite a los pasajeros de un viaje calificar al conductor, así como también la autenticación a través de la red social Facebook. Estas dos características decidieron desarrollarse pensando en que podrían generar mayor confianza en los usuarios de la aplicación.

El cálculo del tiempo necesario para llevar a cabo el desarrollo de la aplicación web fue desacertado. Este tomó casi el doble de lo que se había estipulado debido a que no se tomaron en consideración diversos factores como:

- No se consideró el tiempo necesario para estudiar a fondo el *framework* usado en el desarrollo de la aplicación. La curva de aprendizaje fue mayor de lo esperada.
- No se consideró el retraso que podía generar ciertos *bugs* que debían ser corregidos tras su aparición. Algunos de estos *bugs* retrasaron el desarrollo puesto que para poder implementar un nuevo *feature* o característica, estos debían ser corregidos.
- No se consideró el tiempo que tomó la configuración del servidor para realizar pruebas.
- No se consideró el tiempo de factores externos que por una razón u otra pudiesen causar un retraso en el desarrollo del proyecto. Ejemplo: fallas de conexión a internet.

A pesar de esto, la aplicación web se desarrolló con éxito. Se realizó una correcta implementación siguiendo el patrón de diseño MVC (Modelo-Vista-Controlador).

Se realizaron pruebas haciendo uso de PHPUnit, un *framework* incluido en Laravel que permite escribir pruebas unitarias. Estas pruebas ayudaron en el seguimiento y corrección de errores y se realizaron hasta que los módulos funcionales desarrollados aprobaran la totalidad de las pruebas.

6.3 Recomendaciones

El desarrollo de esta tesis fue un trayecto largo que permitió la implementación de gran cantidad de funcionalidades, aun así, tanto la aplicación web como el proyecto en general da cabida a mejoras que permitirían hacer de este una solución de mayor calidad para los venezolanos.

Algunas de las recomendaciones y/o sugerencias que se pueden realizar son las siguientes:

- Implementar una API Rest que pueda ser utilizada para ofrecer los datos contenidos en la base de datos a cualquier cliente.
- Desarrollar una aplicación móvil que se conecte a una API Rest previamente creada y que ofrezca la misma funcionalidad de la aplicación web.
- Modificar la implementación del API de Google Maps para permitir al conductor del viaje seleccionar entre las distintas rutas que permiten el desplazamiento desde el punto de origen al punto destino.

Adicionalmente, se aconseja encarecidamente a los futuros desarrolladores continuar con las buenas prácticas de desarrollo de software, mantener un código limpio, comentado y documentado. Realizar pruebas de funcionalidad y estrés.

Bibliografía

Balkhi, S. (s.f.). *What is: PHP*. Recuperado el 12 de Julio de 2016, de WPBeginner: http://www.wpbeginner.com

Beroldo, S. (1990). Casual Carpooling in the San Francisco Bay Area. *Transportation Quaterly*, 133-150.

Carpool. (s.f.). Recuperado el 5 de Febrero de 2016, de Wikipedia: http://en.wikipedia.org/wiki/Carpool

Chan, N. D., & Susan A, S. (2011). Ridesharing in North America: Past, Present, and Future. *Transport Reviews*, 93-112.

Christensson, P. (23 de Mayo de 2015). *HTML Definition*. Recuperado el 10 de Julio de 2016, de Tech Terms: http://techterms.com

Dillet, R. (15 de Abril de 2015). Recuperado el 5 de Febrero de 2016, de TechCrunch: https://techcrunch.com/2015/04/15/blablacar-acquires-its-biggest-competitor-carpooling-com-to-dominate-european-market/

Elmasri, R., & Shamkant B., N. (2007). *Fundamentos de Sistemas de Bases de Datos.* Madrid: Pearson Education.

Gutiérrez, J. (18 de Febrero de 2016). *Alternativas de locomoción ante nuevo precio de la gasolina*. Recuperado el 15 de Agosto de 2016, de El Cambur: http:http://www.elcambur.com.ve/ciudadania-2/alternativas-de-locomocion-ante-nuevo-precio-de-la-gasolina

Montilva, J. (Octubre de 2010). Recuperado el 02 de Marzo de 2016, de Blog de Luis Castellanos: https://luiscastellanos.files.wordpress.com/2014/02/mc3a9todos-balanceados-para-ds-blue-watch-jonas-montilva.pdf

Marc, O., & Andrew, A. (Agosto de 2010). *MIT SIPB Script Services for Athena.* Recuperado el 17 de Febrero de 2016, de Dynamic Ridesharing: Carpooling Meets the Information Age: http://ridesharechoices.scripts.mit.edu/home/wp-content/papers/APA_TPD_Webinar_Aug2010.pdf

Mazas, C. R. (01 de Agosto de 2015). *Con tan sólo cinco años de vida, Uber supera el valor de 50 mil millones de dólares*. Recuperado el 25 de Enero de 2016, de Xombit: http://xombit.com/2015/08/uber-valor-mas-de-50-mil-millones-dolares

Migdalis, C. (15 de Junio de 2014). *El Universal*. Recuperado el 28 de Enero de 2016, de http://www.eluniversal.com/noticias/caracas/eliminan-rutas-interurbanas-por-falta-unidades_154237

Mora, S. L. (2002). *Programación de aplicaciones web: historia, principios básicos y clientes web*. Alicante: Editorial Club Universitario.

Object Management Group. (Julio de 2005). *What is UML*. Recuperado el 8 de Agosto de 2016, de Unified Modeling Language: http://www.uml.org/what-is-uml.htm

Mondelo, V., & Oms, J. (21 de Octubre de 2014). *El Mundo Catalunya*. Recuperado el 01 de Febrero de 2016, de http://www.elmundo.es/cataluna/2014/10/21/544633e8268e3ed87e8b4577.html

Victoria Transport Policy Institute. (21 de Diciembre de 2015). *Ridesharing Carpooling and Vanpooling*. Recuperado el 13 de Agosto de 2016, de Victoria Transport Policy Institute: http://www.vtpi.org/tdm/tdm34.htm