



UNIVERSIDAD POLITÉCNICA SALESIANA

SEDE QUITO

CARRERA DE INGENIERÍA DE SISTEMAS

**ANÁLISIS Y DESARROLLO DE UN SISTEMA INFORMÁTICO, PARA LA WEB,
QUE AUTOMATICHE LA GESTIÓN SOBRE EL USO DE LAS ZONAS DE
PARQUEO EN EL CAMPUS SUR DE LA UNIVERSIDAD POLITÉCNICA
SALESIANA**

Trabajo de titulación previo a la obtención del
Título de Ingenieras de Sistemas

AUTORAS: Tatiana Lisbeth Masapanta Tulmo

Micaela Maribel Minga Medina

TUTORA: Lina Patricia Zapata Molina

Quito - Ecuador

2022

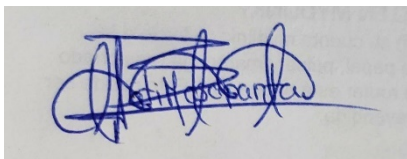
**CERTIFICADO DE RESPONSABILIDAD DE AUTORIA DEL TRABAJO DE
TITULACIÓN**

Nosotras, Tatiana Lisbeth Masapanta Tulmo, con documento/s de identificación N° 1726434267 y Micaela Maribel Minga Medina N° 1723840805; manifestamos que:

Somos las autoras y responsables del presenta trabajo; y, autorizamos a que sin fines de lucro la Universidad Politécnica Salesiana pueda usar, difundir, reproducir o publicar de manera total o parcial el presente trabajo de titulación.

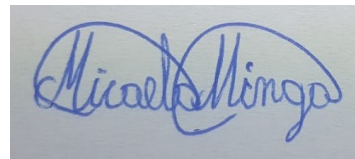
Quito, 04 de marzo de 2022

Atentamente,



Tatiana Lisbeth Masapanta Tulmo

1726434267



Micaela Maribel Minga Medina

1723840805

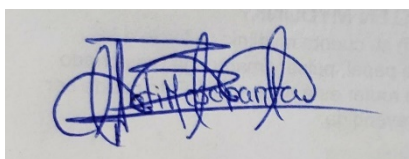
**CERTIFICADO DE CESIÓN DE DERECHOS DE AUTOR DEL TRABAJO DE
TITULACIÓN A LA UNIVERSIDAD POLITÉCNICA SALESIANA**

Nosotras, Tatiana Lisbeth Masapanta Tulmo, con documento/s de identificación N° 1726434267 y Micaela Maribel Minga Medina N° 1723840805, expresamos nuestra voluntad y por medio del presente documento cedemos a la Universidad Politécnica Salesiana la titularidad sobre los derechos patrimoniales en virtud de que somos autores del Proyecto Técnico: “Análisis y desarrollo de un sistema informático, para la web, que automatice la gestión sobre el uso de las zonas de parqueo en el campus sur de la Universidad Politécnica Salesiana”, el cual 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 concordancia con lo manifestado, suscribimos este documento en el momento que hacemos la entrega del trabajo final en formato digital a la Biblioteca de la Universidad Politécnica Salesiana.

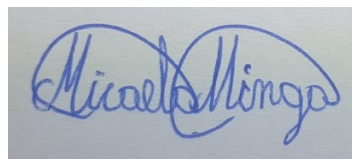
Quito, 04 de marzo de 2022

Atentamente,



Tatiana Lisbeth Masapanta Tulmo

1726434267



Micaela Maribel Minga Medina

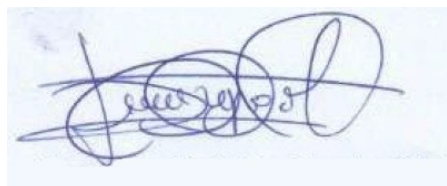
1723840805

CERTIFICADO DE DIRECCIÓN DEL TRABAJO DE TITULACIÓN

Yo, Lina Patricia Zapata Molina con documento de identidad N° 0501877278, docente de la Universidad Politécnica Salesiana, declaro que bajo mi tutoría fue desarrollado el trabajo de titulación: ANÁLISIS Y DESARROLLO DE UN SISTEMA INFORMÁTICO, PARA LA WEB, QUE AUTOMATICE LA GESTIÓN SOBRE EL USO DE LAS ZONAS DE PARQUEO EN EL CAMPUS SUR DE LA UNIVERSIDAD POLITÉCNICA SALESIANA realizado por Tatiana Lisbeth Masapanta Tulmo con documento de identificación N° 1726434267 y Micaela Maribel Minga Medina con documento de identificación N° 1723840805, obteniendo como resultado final el trabajo de titulación bajo la opción proyecto técnico que cumple con todos los requisitos determinados por la Universidad Politécnica Salesiana.

Quito, 04 de marzo de 2022

Atentamente,

A handwritten signature in blue ink, appearing to read 'Lina Patricia Zapata Molina', is written over a light blue rectangular background.

Ing. Lina Patricia Zapata Molina, PhD

0501877278

DEDICATORIA

A Dios por cuidar y guiar mi camino dándome la resiliencia suficiente ante todas las situaciones. A mis padres Alberto y Elsa quienes que son su amor y esfuerzo me ayudaron a seguir siempre adelante y me han permitido cumplir un objetivo más, son y serán mi motor fundamental para alcanzar todos mis sueños. Mis hermanos Jimmy y Noemi que con su apoyo incondicional y palabras de aliento me impulsaron a seguir adelante. A mi abuelito Francisco que estuvo en todo mi camino preocupándose por mí y dándome su amor, palabras de aliento y apoyo incondicional. A toda mi familia por sus oraciones y buenas vibras para cumplir mis objetivos. A mi amiga Micaela que siempre tuve su apoyo incondicional ante situaciones buenas y más aún en las difíciles. Finalmente, a mis amigos que supieron brindarme su mano.

Tatiana Lisbeth Masapanta Tulmo

El presente trabajo lo dedico principalmente a Dios por darme la vida, ser mi apoyo espiritual y brindarme la fortaleza para poder seguir adelante y superar las dificultades que se presentaron a lo largo de mi vida universitaria y así poder cumplir una meta más.

A mis padres por su apoyo incondicional a lo largo de mi vida, por todo su esfuerzo y sacrificio, por siempre estar para mí y darme palabras de aliento y consejos cuando lo necesitaba y quienes, con su amor, educación y confianza me han permitido llegar a cumplir una mete más en vida, gracias por todo, los amo infinitamente.

A mi hermana mi mejor amiga, mi confidente, mi ejemplo a seguir, quien ha estado siempre presenta para escuchar todas mis historias, dudas, dificultades que se presentaron, te agradezco por todas esas palabras de aliento que me permitieron seguir adelante y por estar siempre dispuesta ayudarme y permanecer a mi lado cuando lo necesito.

Micaela Mariel Minga Medina

AGRADECIMIENTO

Nos gustaría agradecer en primer lugar a Dios por bendecirnos con la fortaleza, sabiduría y por guiar nuestros pasos para cumplir un objetivo más en nuestras vidas. A nuestros padres por su apoyo incondicional, quienes estuvieron a nuestro lado dándonos consejos y la fortaleza ante todas las situaciones que se nos presentaron en el camino que fueron tanto buenas como malas. A nuestros hermanos por sus palabras de aliento que nos impulsaron a seguir adelante ante cualquier dificultad. A la ingeniera Lina Zapata por darnos la oportunidad de desarrollar el tema de tesis, y por la guía brindada a lo largo de todo este proyecto. Al ingeniero Julio Proaño por compartirnos su conocimiento acerca del tema de web services, para poder realizar la implementación dentro del proyecto. A nuestros familiares por estar al pendiente de nosotras. Al área de Dirección Técnica de Administración e Inventarios de la Universidad Politécnica Salesiana por su colaboración en las reuniones requeridas para validar los avances del proyecto. A la Universidad Politécnica Salesiana y a la carrera de Ingeniera de Sistemas por brindarnos el conocimiento y herramientas para ser mejores personas y grandes profesionales.

Tatiana Lisbeth Masapanta Tulmo

Micaela Maribel Minga Medina

ÍNDICE GENERAL

INTRODUCCIÓN.....	1
CAPÍTULO I.....	5
MARCO TEÓRICO.....	5
1.1. UNIVERSIDAD POLITÉCNICA SALESIANA.....	5
1.2. Gestión de parqueaderos.....	6
1.3. Herramientas.....	6
1.3.1. Netbeans.....	6
1.3.2. Postman	6
1.4. Servidor WEB.....	7
1.4.1. Xampp.....	7
1.4.1.1. Apache.....	8
1.4.1.2. phpMyAdmin.....	8
1.5. Lenguaje de programación.....	8
1.6. CodeIgniter.....	9
1.6.1. Modelo Vista Controlador MVC.....	9
1.6.1.1. Modelo.	10
1.6.1.2. Vista.....	10
1.6.1.3. Controlador.....	10
1.7. Web service.....	10
1.7.1 APIs REST.....	11
1.8. Metodología XP.....	11
1.8.1. Planeación.....	12
1.8.2. Diseño.....	12
1.8.3. Codificación.....	12
1.8.4. Pruebas.....	12
CAPÍTULO II.....	13
ANÁLISIS Y DISEÑO.....	13
2.1. Análisis de requerimientos.....	13
2.1.1. Alcance.....	13
2.1.2. Personal involucrado.....	13
2.1.3. Descripción del sistema.....	15
2.1.3.1. Administrador.....	15
2.1.3.2. Servicios.....	15

2.1.3.3.	Administrativo, docente y estudiante.....	16
2.1.4.	Requerimientos.....	16
2.1.4.1.	Requerimientos funcionales.	16
2.1.4.2.	Requerimientos no funcionales.	17
2.2.	Historias de usuarios.....	17
2.3.	Diseño.....	19
2.3.1.	Diagrama caso de uso.....	19
2.3.2.	Diagrama de actividades.....	21
2.3.3.	Diagrama de secuencia.....	29
2.3.4.	Diagrama de clases.....	38
2.3.5.	Diagrama entidad relación.....	39
2.3.6.	Mockup de interfaces.....	40
2.3.6.1.	Página principal.....	40
2.3.6.2.	Administrador.....	42
2.3.6.3.	Administrativo, docente y estudiante.....	49
CAPÍTULO III.....		52
CONSTRUCCIÓN Y PRUEBAS.....		52
3.1.	CONSTRUCCIÓN.....	52
3.1.1.	Base de datos.....	52
3.1.2.	Implementación del Framework.....	53
3.1.2.1.	Codeigniter.....	53
3.1.3.	Modelo-Vista-Controlador.....	53
3.1.3.1.	Modelo.....	53
3.1.3.2.	Vista.....	54
3.1.3.3.	Controlador.....	55
3.1.4.	Web Service.....	56
3.1.4.1.	Postman.....	58
3.1.5.	Importación y exportación del Excel.....	59
3.1.6.	Implementación del correo.....	59
3.1.7.	Encriptación de contraseñas.....	60
3.1.8.	Interfaz de usuario.....	60
3.1.8.1.	Administrador.....	60
3.1.8.2.	Administrativo, docente y estudiante.....	65
3.2.	PRUEBAS.....	66
3.2.1.	Estrés.....	66
3.2.2.	Caja Negra.....	68

3.2.3. Usabilidad.....	76
CONCLUSIONES.....	82
RECOMENDACIONES.....	83
GLOSARIO.....	84
LISTA DE REFERENCIAS	85

ÍNDICE DE TABLAS

Tabla 1. Personal involucrado.....	14
Tabla 2. Historia de Usuario 1.....	17
Tabla 3. Historia de Usuario 2.....	18
Tabla 4. Historia de Usuario 3.....	18
Tabla 5. Historia de Usuario 4.....	18
Tabla 6. Historia de Usuario 5.....	18
Tabla 7. Historia de Usuario 6.....	19
Tabla 8. Historia de Usuario 7.....	19
Tabla 9. Historia de Usuario 8.....	19
Tabla 10. Pruebas Estrés Jmeter.....	68
Tabla 11. Prueba de caja negra 1.....	68
Tabla 12. Prueba de caja negra 2.....	70
Tabla 13. Prueba caja negra 3.....	71
Tabla 14. Prueba caja negra 4.....	72
Tabla 15. Prueba de caja negra 5.....	74

ÍNDICE DE FIGURAS

Figura 1. Diagrama caso de uso inicio de sesión.	20
Figura 2. Diagrama caso de uso para el sistema.	20
Figura 3. Diagrama de uso proceso de solicitud.	21
Figura 4. Diagrama de actividades inicio de sesión.	22
Figura 5. Diagrama de actividades ingreso de persona.....	23
Figura 6. Diagrama de actividades para cargar un archivo.....	24
Figura 7. Diagrama de actividades para exportar un archivo.....	25
Figura 8. Diagrama de actividades para el cambio de estado de zona.....	26
Figura 9. Diagrama de actividades para la configuración.....	27
Figura 10. Diagrama de actividades para activación de solicitud.....	28
Figura 11. Diagrama de actividades para estudiante nuevo y antiguo.....	29
Figura 12. Diagrama de secuencia inicio de sesión.	30
Figura 13. Diagrama de secuencia ingreso de persona.	31
Figura 14. Diagrama de secuencia para cargar un archivo.	32
Figura 15. Diagrama de secuencia para exportar un archivo.....	33
Figura 16. Diagrama de secuencia para el cambio de estado de zona.	34
Figura 17. Diagrama de secuencia para la configuración.	35
Figura 18. Diagrama de secuencia para activación de solicitud.	36
Figura 19. Diagrama de secuencia para estudiante nuevo.	37
Figura 20. Diagrama de secuencia para estudiante antiguo.	38
Figura 21. Diagrama de clases.	39
Figura 22. Base de Datos	40
Figura 23. Interfaz: página principal.....	41

Figura 24. Interfaz: inicio de sesión.....	41
Figura 25. Interfaz: Ingreso de personas.....	42
Figura 26. Interfaz: Búsqueda de personas por rol.....	43
Figura 27. Interfaz: Carga de archivo.....	44
Figura 28. Interfaz: Generación del excel.....	45
Figura 29. Interfaz: activación de zona.....	46
Figura 30. Interfaz: activación de solicitudes.....	47
Figura 31. Interfaz: Búsqueda de personas por cédula.....	48
Figura 32. Interfaz: cambio de contraseñas.....	49
Figura 33. Interfaz: envío de solicitudes para personas nuevas.....	50
Figura 34. Interfaz: envío de solicitudes para personas antiguas.....	51
Figura 35. Conexión con a la Base de datos.....	52
Figura 36. Framework Codeigniter.....	53
Figura 37. Framework Codeigniter.....	54
Figura 38. Código de la vista.....	54
Figura 39. Interfaz generada por el código de la vista.....	55
Figura 40. Código del controlador.....	56
Figura 41. Carpeta de las librerías.....	56
Figura 42. Código REST_Controller.....	57
Figura 43. Código del intermediario.....	58
Figura 44. Herramienta Postman.....	58
Figura 45. Carpeta libraries.....	59
Figura 46. Código para envío de correo.....	59
Figura 47. Código para la encriptación de contraseñas.....	60

Figura 48. Código para la descriptación de contraseñas	60
Figura 49. Pantalla Persona.....	61
Figura 50. Pantalla Vehículo.....	61
Figura 51. Pantalla Parqueadero.....	62
Figura 52. Pantalla Solicitud.....	63
Figura 53. Pantalla Autorizado.....	63
Figura 54. Pantalla Adicional.....	64
Figura 55. Pantalla Histórico.....	65
Figura 56. Pantalla Envío de solicitudes.....	66
Figura 57. Resultado en árbol 1	67
Figura 58. Reporte resumen 1	67
Figura 59. Usar el sistema le resultó	76
Figura 60. ¿Le resultó amigable la interfaz?	77
Figura 61. ¿Puede completar el trabajo usando el sistema?.....	77
Figura 62. ¿Puede completar el trabajo rápidamente usando el sistema?.....	78
Figura 63. ¿La información proporcionada es clara?.....	78
Figura 64. ¿Considera que gráficamente el sitio está equilibrado?.....	79
Figura 65. ¿Cree que los iconos son fáciles de entender?.....	79
Figura 66. Le fue fácil encontrar la ventana para la tarea a realizar	80
Figura 67. Los mensajes de alerta le parecieron claros.....	80
Figura 68. Utilizaría frecuentemente el sistema.....	81

RESUMEN

En la presente investigación se desarrolló un Sistema Web que permite gestionar la administración del parqueadero de la Universidad Politécnica Salesiana Campus Sur. En los últimos años la UPS-CS dispone de un control manual para la gestión de parqueaderos, lo que puede ocasionar un problema al momento de llevar el control de la información de alumnos, personal docente y administrativo que hacen uso del parqueadero. Por lo tanto, se desarrolló un sistema que ayudo a automatizar el proceso, mejorando el acceso al parqueadero, el sistema tiene las funciones de ingresar, modificar, visualizar y eliminar los datos de las personas que fueron autorizadas para el uso del parqueadero, el proceso se realiza de manera individual o mediante un archivo de Excel que contenga varios registros. Se puede visualizar las zonas disponibles y la información de las personas autorizadas. El sistema se desarrolló implementando un web services y un framework denominado codeigniter. El sistema se desarrolló bajo los requerimientos establecidos por las personas del departamento encargado para hacer uso del mismo, de tal forma que, se tiene un sistema intuitivo y fácil de usar.

ABSTRACT

This thesis will develop a Web System to manage the administration of the parking lot of the Universidad Politécnica Salesiana Campus Sur. In recent years, the UPS-CS has had a manual control system for parking lot management, which can cause a problem when it comes to keeping track of information on both students and engineers authorized to use the parking lot. Therefore, a system will be developed to automate the process, improving access to the parking lot, the system will have the functions to enter, modify, view and delete authorized persons to the parking lot, such entry may be manually or through an Excel file, you can view the available areas and information of authorized persons, the system will be developed through a web service and a framework called codeigniter. The system has been developed under the requirements established by the people of the department in charge of using it, in such a way that the system is intuitive and easy to use.

INTRODUCCIÓN

PROBLEMA

El automóvil es un medio de transporte cuyo uso se ha incrementado en la actualidad, lo que deriva en el aumento del uso de los parqueaderos públicos y privados. Uno de los problemas que tiene la administración manual de los parqueaderos, es el control de la disponibilidad y asignación de los mismos. Según He y Wang (2011) el no conocer los espacios disponibles genera un problema al momento de parquear, o a pesar de saber que hay un puesto se puede generar congestión si varios vehículos buscan ocupar dicho espacio.

La Universidad Politécnica Salesiana Campus Sur (UPS-CS) dispone de un control manual para la gestión de parqueaderos. A los autos favorecidos se les coloca un sticker, el cual, es revisado por los guardias antes de ingresar a la UPS-CS. Hasta la actualidad la administración del parqueadero no tiene un seguimiento adecuado de las personas que ingresan a la UPS-CS. Esto puede llegar a afectar en cuanto a la seguridad, más aún si existe concurrencia al momento de organizar eventos.

Por lo tanto, se ha decidido crear un sistema web para el área administrativa que permita gestionar el uso del parqueadero de la UPS-CS. El desarrollo de una aplicación web podrá dar una solución a los problemas mencionados, ya que permitirá al área administrativa tener un mejor control de las personas que ingresan al parqueadero además de ver los espacios disponibles dentro del mismo.

JUSTIFICACIÓN

El desarrollo de la aplicación web es importante para la automatización del proceso y mejorar el control de acceso de las personas, ayudando a la seguridad, ya que, se obtendrá una base de datos donde se almacene la información para el proceso. Se dejará abierto el registro de entrada y salida de los vehículos, el cual, puede ser considerado para un futuro trabajo. El proyecto permitirá a la administración tener un mejor control y seguimiento de los parqueaderos, lo que ayudará a la eficiencia de la gestión que se lleva a cabo en la UPS-CS.

El sistema será diseñado para las personas que tengan conocimientos acerca del proceso de autorización para el uso de parqueaderos de la UPS-CS, por lo tanto, el departamento de dirección técnica de inventarios de UPS-CS será el beneficiario del sistema y se realizara reuniones con las personas involucradas.

El sistema permitirá la gestión de información de docentes, estudiantes y administrativos previamente aprobados para el uso de parqueaderos y el registro de asignación de sticker de autorización para el ingreso de sus vehículos. Adicionalmente, el sistema permitirá gestionar la disponibilidad de las zonas de parqueo dentro de la UPS-CS, es decir, el usuario administrador podrá consultar y visualizar el estado (ocupado/libre/no habilitado) de las diferentes zonas de parqueo, y también podrá modificar el estado de disponibilidad.

Se implementará web service con el objetivo de que otras aplicaciones puedan interactuar con el sistema, por otro lado, el software permitirá la visualización de reportes, y contará con servicios para la gestión de personas, vehículos, autorizados, entre otros. El software desarrollado quedará implementado en el Data Center de la UPS-CS.

OBJETIVOS

General

Desarrollar un sistema informático para la web, que permita gestionar el uso de las zonas de parqueo de la Universidad Politécnica Salesiana Campus Sur.

Específicos

- Llevar un registro histórico del número de sticker vehicular asignado a docentes y estudiantes.
- Desarrollar un sistema informático en un entorno web para la administración de las zonas de parqueo.
- Implementar un web service para poder integrar con otras aplicaciones.

METODOLOGÍA

El presente proyecto busca el desarrollo de un sistema web que ayude a gestionar las zonas de parqueo de la UPS-CS, por lo tanto, para este proyecto se tomará como base la metodología ágil XP, lo que ayudará a promover el trabajo en equipo. La misma tiene un enfoque a proyectos pequeños, por lo que es ligera, eficiente, y es utilizada para proyectos que tiene alto índice de incertidumbre en los requerimientos. Para este proyecto se hará uso de las siguientes fases de la metodología ágil XP (Expósito, 2008).

1. Planeación: dentro de esta fase se realizará las historias de usuarios y establecer los requerimientos.
2. Diseño: dentro de esta fase se diseñará el sistema haciendo uso de los diagramas UML
3. Desarrollo: dentro de esta fase se desarrollará los diferentes servicios que conforman el sistema web.
4. Pruebas: dentro de esta fase se realizarán las pruebas necesarias para el buen funcionamiento del sistema.

CAPÍTULO I

MARCO TEÓRICO

El presente capítulo abarcará un breve acercamiento al contexto de la institución para la cual será desarrollada el sistema, metodologías, herramientas y lenguajes de programación que van a ser usados para el desarrollo de software orientado a la web.

1.1. UNIVERSIDAD POLITÉCNICA SALESIANA

La Universidad Politécnica Salesiana cuenta con sedes en las ciudades de Quito, Cuenca y Guayaquil. En la ciudad de Quito tiene dos campus, un campus se encuentra ubicado al norte de la ciudad, en la Av. Isabel la Católica N23-52 Madrid, y el segundo campus se encuentra ubicado al sur, en la Av. Moran Valverde y Av. Rumichaca Ñan. La Universidad fue fundada el 5 de agosto de 1994 en la ciudad de Cuenca, siendo una institución autónoma para la educación superior particular católica. La Universidad Politécnica Salesiana se caracteriza por sus estudios universitarios con formación cristiana de carácter católico, y la formación profesional integral, humana, moral y ética (Universidad Politécnica Salesiana, s.f.).

La Universidad Politécnica Salesiana tiene una sede principal que se encuentra en la ciudad de Cuenca, al inicio contaba con un número reducido de estudiantes. Con el paso del tiempo se ha ido incrementando el número de estudiantes y de la misma forma las instalaciones de la Universidad. Actualmente, la universidad acoge alrededor de 6.000 estudiantes, las instalaciones de la Universidad se encuentran ubicadas a dos cuadras del Aeropuerto Mariscal Lamar y a tres cuadras del Terminal Terrestre (Universidad Politécnica Salesiana, s.f.).

Dentro de la sede Quito el campus Sur la Universidad cuenta con una cancha de fútbol con césped natural, una cancha de volley, un espacio grande de césped natural donde pueden realizar cualquier tipo de deporte. Cuenta con áreas de laboratorios adecuadas para cada carrera, tiene una biblioteca, una capilla, un área para ceremonias o conferencias grandes y también cuenta con varias zonas de parqueo tanto para estudiantes como para docentes. La

Universidad tiene beneficios con los estudiantes al momento de realizar la matrícula, ayudando de esa forma en diferir los pagos de la matrícula (Universidad Politécnica Salesiana, s.f.).

1.2. GESTIÓN DE PARQUEADEROS

Encontrar una zona de parqueo disponible puede resultar complicado. La alta demanda de estos espacios en diferentes zonas de parqueo puede presentar varios problemas, uno de ellos es el aumento del tráfico, por lo que, los automóviles gastan combustible tratando de buscar un parqueadero disponible, lo que, genera un aumento de contaminación ambiental, adicional puede afectar la salud de los conductores por el estrés que llegue a tener al tratar de buscar una zona disponible de parqueadero (Ávalos-Silva et al., 2018).

1.3. HERRAMIENTAS

1.3.1. *Netbeans*

Un entorno de desarrollo integrado (IDE) que puede ser utilizado para el desarrollo de software es NetBeans, el cual permite crear aplicaciones en lenguajes de programación como java, php, entre otros. El IDE tiene una ventaja que es open source, es decir, no tiene costo el acceder a sus funcionalidades (Apache Netbeans, s.f.).

Netbeans es multiplataforma, se puede instalar en los diferentes sistemas operativos que acepten java, también es multilenguaje, ya que, permite seleccionar entre inglés y español. Una de las características que tiene Netbeans es su editor de texto que tiene una colaboración para la sintaxis del código, lo que ayuda a tener un mejor control del código, también posee un depurador para poder ejecutar el código paso a paso validando de esa manera si existe algún error (Boudreau et al., 2002).

1.3.2. *Postman*

Postman es un software que permite probar peticiones utilizando los diferentes métodos como post, put, get, entre otros. Esto permite probar la API para saber si la petición regresa lo que se espera, testeando de esa forma el HTTP requests (Postman, s.f.).

1.4. SERVIDOR WEB

Un servidor es un ordenador el cual ayuda a la transmisión de datos, de tal forma que, un servidor web puede almacenar una página Web. Una persona puede realizar peticiones al servidor desde cualquier parte del mundo, y de la misma forma puede usar distintos navegadores para realizar dicha petición, ya que, el servidor buscará los archivos o información que el usuario solicitó. Para poder navegar por un sitio Web es necesario tener la dirección o URL exacta. Una URL está formada por un protocolo, el nombre del dominio y la ruta del servidor, haciendo de esa forma una única URL (Ramos & Ramos, 2014).

1.4.1. Xampp

Xampp es un software libre que incluye algunos programas como el servidor web Apache, un sistema gestor de base de datos MySQL, el lenguaje de programación php con el cual se puede crear páginas web y el lenguaje de programación Perl el cual puede ser usado para programar web o de red, en el administrador del sistema. Xampp puede ser instalado en los sistemas operativos Windows, Linux o Mac. Algunas características de Xampp es que permite hacer pruebas localmente es decir dentro de nuestro ordenador ya que los proyectos pueden ser instalados de manera local dentro de la misma (Carrión et al., 2019).

1.4.1.1. Apache. Apache es un servidor web HTTP, gratuito y de código abierto, es el más utilizado en países americanos, pero también a nivel mundial. Apache puede alojar sitios web ya sea estáticos como dinámicos, este servidor puede ser utilizado en diferentes sistemas operativos. Apache y PHP son piezas fundamentales para el desarrollo, por también acepta otros lenguajes de programación como Perl, Python, aceptando de esa forma una base de datos de autenticación (Cobo et al., 2005).

1.4.1.2. phpMyAdmin. PhpMyAdmin desarrollado en PHP es un software gratuito el cual permite que a través de la web el usuario pueda utilizar diferentes operaciones de MySQL y MariaDB. Algunas de las operaciones es la gestión de base de datos y tablas, es decir la creación, modificación y eliminación de las mismas, también permite insertar, modificar, visualizar y eliminar los campos de las tablas, permite sacar respaldos de la base de datos y poder exportar la misma, en caso de desear abrir una base de datos se la puede importar (phpMyAdmin, s.f.).

Esta herramienta posee una interfaz intuitiva lo que facilita su uso y se encuentra en proceso de traducción para 72 idiomas. Existe documentación la cual se puede consultar, un libro que se puede encontrar en los idiomas de inglés y español. En caso de tener algún problema tiene la opción de dirigirse a diferentes canales que se podrá solventar las dudas (phpMyAdmin, s.f.).

1.5. LENGUAJE DE PROGRAMACIÓN

A inicios el lenguaje PHP tenía un objetivo específico y era la creación o desarrollo de scripts en Perl donde los usuarios podían crear sus propias páginas personales, de tal manera, que en un inicio PHP significaba Personal Home Page. Después de un tiempo se desarrollaron formularios que eran privados, pero con el paso del tiempo dichos formularios fueron de código abierto, dando la opción que el usuario pueda utilizarlos (Cobo et al., 2005).

Php es un lenguaje de programación más utilizado para el desarrollo de páginas Web por lado del servidor, contiene varias librerías y cuenta con la documentación de todas ellas. En PHP se puede generar varios scripts y de la misma forma se puede trabajar con una gran diversidad de bases de datos como MySQL, PostgreSQL, SQL Server, entre otros. Existen proveedores que ofrecen PHP a precios muy accesibles, sin contar que PHP es un lenguaje muy sencillo de aprender y entender, ya que sigue una estructura open source (Spona, 2010).

PHP tiene una cierta similitud en sintaxis con los lenguajes como C, Perl o Java, claro que este lenguaje es multiplataforma, ya que, puede trabajar en varios servidores e interactuar con una gran variedad de base de datos. Estas características han ayudado al lenguaje a crecer y destacarse, teniendo la ventaja de ser reconocido por los usuarios debido a su facilidad de conectividad. PHP y la conectividad con el sistema gestor MySQL es la más utilizada para la realización de páginas dinámicas y no solo a nivel personal sino también para empresas u organizaciones (Cobo et al., 2005).

1.6. CODEIGNITER

El framework codeigniter es usado por gran parte de desarrolladores que deseen crear sitios web dinámicos utilizando el lenguaje PHP, dicho framework es de código abierto, permitiendo a los desarrolladores crear sitios web desde cero. El propósito del framework es facilitar la creación de código, teniendo una estructura organizada y ligera (CodeIgniter, s.f.).

1.6.1. Modelo Vista Controlador MVC

El objetivo principal del Modelo-Vista-Controlador es reducir el desarrollo del código, ya que, su arquitectura está basada en capas para aplicaciones web. En el modelo se divide en aplicación, interfaz y lógica, estos servicios son desarrollados de manera independiente (Ávila, 2019).

1.6.1.1. Modelo. El modelo es la parte donde se trabaja con los datos y se realizan las diferentes operaciones para obtener los resultados que se espera mostrar. Los datos pueden estar almacenados en un sistema gestor de base de datos, por lo cual, se debe realizar la conexión con el mismo para poder actualizar, visualizar, modificar o eliminar los datos que se encuentren dentro de las tablas (Alonso-Aranda, 2019).

1.6.1.2. Vista. La vista se encarga de presentar la parte visual, es decir toda la interfaz con la que el usuario tendrá que interactuar, esto incluye pantallas, ventanas, formularios y dentro de la mismas se tiene código HTML, pero a pesar de eso no se puede tener acceso directo, ya que, requiere los datos de un modelo para tener una salida (Hernández, 2018).

1.6.1.3. Controlador. El controlador es la conexión entre la vista y el modelo por medio del cual se pueden comunicar. Dentro del controlador no se realizan acciones que deben ser ejecutadas por la vista o por el modelo como es el trabajo con los datos o la definición de la interfaz. En el controlador se escribe el código que permita dar respuestas a las peticiones que sean realizadas desde la aplicación por parte del usuario (Alonso-Aranda, 2019).

1.7. WEB SERVICE

Un Web service es una alternativa que surgió para mejorar las oportunidades de interoperabilidad, mediante el uso de protocolos y estándares, ayudando de esa manera la comunicación entre diferentes computadoras y permitiendo el intercambio de información. Un web service puede ser multiplataforma, donde la configuración del cliente y el servidor no debe ser la misma, por otro lado, se tiene una web service distribuido donde se pueden conectar a través de Internet (Driss et al., 2020).

El mantenimiento de un Web service dependerá de las operaciones que realicen, la facilidad del cambio, análisis y pruebas. La empresa Microsoft lo clasifica según su funcionalidad, la primera es orientada a datos, teniendo un enfoque a los datos que son frecuentemente manipulados, por lo tanto, la organización necesita un adecuado tiempo y

recursos para obtener resultados eficientes, también puede por colaboración, donde los usuarios de la organización están involucrados, por otro lado, puede ser por análisis, donde una organización recibe informes de varias empresas, de tal forma que, necesita impulsar la usabilidad de la organización y por último se tiene por alertas donde el usuario necesite que se envíe notificaciones de aviso y puedan actuar de manera rápida (Pérez et al., 2005).

1.7.1 APIs REST

En la actualidad las empresas han tratado de implementar la comunicación de sus sistemas con aplicaciones de terceros, este proceso se puede realizar mediante APIs. Las APIs web facilitan la creación del código permitiendo pasar ciertas funcionalidades complejas a simples utilizando protocolos. Las APIs ayudan a la comunicación de los servicios, sin tener la necesidad de saber cómo fueron implementados (Mirabella et al., s.f.).

REST no es un protocolo ni un estándar, por lo tanto, una APIs REST ayudan a tener mejor control por medio del CRUD, los recursos pueden ser accedidos por protocolos HTTP y estos pueden ser GET, POST, PUT, DELETE. Para que una API pueda ser considerada RESTful debe tener una arquitectura cliente-servidor (RedHat, 2020).

1.8. METODOLOGÍA XP

XP es una metodología ágil para el desarrollo de software, sus interacciones son cortas donde el equipo de trabajo es reducido y realizan actividades específicas para ser presentadas al final. La metodología XP tiene una filosofía de trabajo a lo que se refiere programación, dentro de este servicio lo realizan en parejas y con pruebas constantes, por otro lado, se acepta cambios a pesar de que el proyecto ya se esté desarrollando. Adicionalmente la metodología XP ayuda a que la creatividad vaya creciendo en los desarrolladores y también acepta errores como una parte del trabajo (Ambler, 2002).

1.8.1. Planeación

En la fase de planeación se revisan las historias de usuario, se debe estimar tiempo de desarrollo y la dificultad para cada tarea, con lo cual se puede proceder a priorizar cada una (Fernández, 2013).

1.8.2. Diseño

En la fase de diseño se trabaja con el código para el desarrollo del software, el mismo tiene como objetivo generar código sencillo para cumplir con el funcionamiento eficiente del sistema (Orientación, 2020).

1.8.3. Codificación

En la fase de codificación se realiza la programación del software la cual se debe en parejas, teniendo como objetivo que el código pueda ser entendido por diferentes programadores, además puede permitir que las personas que estén programando puedan añadir o modificar partes del código (Sinnaps, s.f.).

1.8.4. Pruebas

El código se va integrando en el sistema, de tal forma que, las pruebas deben ser continuas, debido a que el proyecto o sistemas son a corto plazo. Por otro lado, el cliente puede realizar pruebas y de esa forma generar un punto de vista validando el sistema y versiones que se vayan generando (Sinnaps, s.f.).

CAPÍTULO II

ANÁLISIS Y DISEÑO

1.9. ANÁLISIS DE REQUERIMIENTOS

El análisis de los requerimientos es un paso previo para empezar con el desarrollo del software. En este apartado se presentará el alcance del software, el personal involucrado, los requerimientos funcionales o no funcionales, una descripción del sistema informático.

1.9.1. Alcance

El software tiene como propósito crear un ambiente donde los docentes, administrativos y estudiantes puedan tener acceso al sistema de solicitudes de parqueaderos y el Administrador pueda gestionar los parqueaderos de la UPS-CS.

El sistema permitirá al Administrador ingresar toda la información de las personas que soliciten usar el parqueadero, de la misma forma, se podrá hacer con los vehículos. Esto facilita tener un control de las personas que tengan acceso al parqueadero previamente ingresadas, permitiendo de esa forma, tener la información más rápida ya sea de la persona o vehículo. Se podrá realizar la gestión de los parqueaderos a través del ingreso de zonas y espacio de parqueo, donde el administrador podrá visualizar y modificar el estado, ayudando a tener un mejor control de las zonas de parqueo de la UPS-CS. El sistema desarrollado no incluye la modificación de parqueaderos en tiempo real, tampoco el ingreso automatizado de autos, es decir la lectura de placas.

1.9.2. Personal involucrado

El sistema toma en cuenta cuatro roles para el manejo del mismo, dos roles para el desarrollo del sistema y un rol para la supervisión del desarrollo, el detalle se puede verificar en la tabla 4.

Tabla 1*Personal involucrado*

Nombre - Rol	Detalle	Características
Administrador	Personas que harán uso del sistema.	<ul style="list-style-type: none"> ● Podrá ingresar, visualizar, modificar y eliminar usuarios y vehículos. ● Gestión de parqueaderos. ● Exportación e importación de archivos Excel.
Administrativo	Personas que harán uso del sistema.	<ul style="list-style-type: none"> ● Podrá ingresar una solicitud para pedir una zona de parqueo.
Docente	Personas que harán uso del sistema.	<ul style="list-style-type: none"> ● Podrá ingresar una solicitud para pedir una zona de parqueo.
Estudiante	Personas que harán uso del sistema.	<ul style="list-style-type: none"> ● Podrá ingresar una solicitud para pedir una zona de parqueo.
Lina Zapata	Ingeniería a cargo del proyecto.	<ul style="list-style-type: none"> ● Supervisar el desarrollo.
Tatiana Masapanta	Persona a cargo del desarrollo.	<ul style="list-style-type: none"> ● Programador Backend y Frontend. ● Diseño de interfaces. ● Diseño de Base de datos ● Diseño de diagramas UML. ● Construcción de APIS.
Micaela Minga	Persona a cargo del desarrollo.	<ul style="list-style-type: none"> ● Programador Backend y Frontend. ● Diseño de interfaces.

		<ul style="list-style-type: none"> ● Diseño de Base de datos ● Diseño de diagramas UML. ● Construcción de APIS.
--	--	--

Nota. Detalle de las personas involucradas para el desarrollo y uso del sistema. Elaborado por: Tatiana Masapanta & Micaela Minga.

1.9.3. Descripción del sistema

El sistema está diseñado para cuatro usuarios que son administrador, docentes, administrativos y estudiantes; por lo tanto, se desarrolló diferentes interfaces para cada usuario, en las cuales podrán realizar las acciones correspondientes a cada uno, las mismas serán accesibles a través del inicio de sesión en el sistema utilizando las credenciales que estén registradas dentro de la base de datos.

1.9.3.1. Administrador. El administrador dispone de ocho servicios que se detallan más adelante, los mismo que serán de ayuda para la gestión de parqueaderos.

1.9.3.2. Servicios. En el sistema se tiene los siguientes servicios:

- **Persona:** en este servicio el administrador podrá ingresar, modificar, buscar ya sea mediante el rol o cédula, también podrá eliminar los datos de la persona, si lo ve necesario.
- **Vehículo:** en este servicio el administrador podrá ingresar, modificar, buscar ya sea mediante la placa o cédula, también podrá eliminar los datos del vehículo, si lo ve necesario.
- **Parqueadero:** en este servicio podrá ingresar los datos de las zonas y lugares de parqueo, también permitirá visualizar, eliminar y cambiar los estados de las zonas y lugares de parqueo.
- **Solicitud:** en este servicio se podrá activar los sistemas de solicitudes para docentes, administrativos y estudiantes, así como cambiar la fecha de cierre si la solicitud está

activa o cambiar la fecha de inicio y fin si está pendiente la activación. También se podrá descargar el Excel con la lista de solicitudes pendientes.

- Autorizado: se podrá cargar el archivo de las personas que han sido autorizadas para usar el parqueadero, también agregar un nuevo autorizado y visualizar si una persona está autorizada para utilizar el parqueadero en el periodo actual.
- Otro: en este servicio se puede ingresar, modificar carrera, horario y periodo, también se podrá eliminar una carrera.
- Histórico: en este servicio se podrá ver datos históricos de los semestres anteriores.
- Configuración: en este servicio se podrá realizar el cambio de clave y correo del administrador.

1.9.3.3. Administrativo, docente y estudiante. El personal administrativo, docentes y estudiantes tienen acceso al envío de las solicitudes para el parqueadero cuando el administrador las habilite. Al momento de enviar la solicitud se debe iniciar sesión en el sistema, ingresar el número de cédula, si no está registrado, deberá llenar sus datos personales y los del vehículo, por otro lado, si ya se registró antes en el sistema solo deberá llenar los datos del vehículo y enviar el formulario.

1.9.4. *Requerimientos*

1.9.4.1. Requerimientos funcionales. El sistema tiene los siguientes requerimientos funcionales:

- El administrador podrá visualizar, agregar modificar y eliminar la información de las personas y vehículos.
- El administrador podrá visualizar y agregar carreras y periodos académicos.
- El administrador podrá gestionar todas las zonas de parqueo.
- Los usuarios deberán iniciar sesión para acceder a los servicios correspondientes.
- El sistema deberá tener diferentes servicios para las diferentes acciones.

- El sistema debe dejar escoger las fechas de inicio y fin para la activación de solicitudes, así como establecer la contraseña.
- Se controlará las fechas de inicio y fin de las solicitudes.
- El sistema deberá tener un servicio donde permita subir la información de la persona y vehículo que desee una zona de parqueo.
- El sistema deberá contar con envío de correos al generar una solicitud.
- El sistema deberá tener un servicio donde permita visualizar datos históricos.
- El sistema podrá realizar el cambio de contraseña y correo del administrador.

1.9.4.2. Requerimientos no funcionales. El sistema tiene los siguientes requerimientos no funcionales:

- El sistema se desarrollará en php bajo el framework de codeigniter.
- El sistema gestor de base de datos a utilizar será MySQL.
- Las contraseñas deberán ser encriptadas dentro de la base de datos.
- El sistema será desarrollado bajo un editor de texto denominado NetBeans.

1.10. HISTORIAS DE USUARIOS

En las siguientes tablas de la 2 a la 9 se muestra las historias de usuario con los requerimientos establecidos por algunos de los usuarios que participaron en una reunión virtual. Las tablas incluyen campos como el usuario, la prioridad, el riesgo y una descripción de cada requerimiento, esto se lo realiza para los cuatro roles que son docente, administrador, estudiante y administrativo.

Tabla 2

Historia de Usuario 1.

Número:	1	Usuario:	Administrador
Nombre Historia:	Servicio persona		
Prioridad en negocio:	Alta	Riesgo de desarrollo:	Media
Descripción:	El administrador tiene la opción de ingresar una persona al sistema, modificar y visualizar los datos de una persona ya registrada, adicional podrá eliminar a la persona.		

Elaborado por: Tatiana Masapanta & Micaela Minga.

Tabla 3*Historia de Usuario 2.*

Número:	2	Usuario:	Administrador
Nombre Historia:	Servicio vehículo		
Prioridad en negocio:	Alta	Riesgo de desarrollo:	Media
Descripción:	El administrador tiene la opción de ingresar un vehículo al sistema, modificar y visualizar los datos de una persona ya registrada, adicional podrá eliminar al vehículo, estas acciones se podrán realizar siempre y cuando la persona este registrada.		

*Elaborado por: Tatiana Masapanta & Micaela Minga.***Tabla 4***Historia de Usuario 3.*

Número:	3	Usuario:	Administrador
Nombre Historia:	Servicio parqueadero		
Prioridad en negocio:	Alta	Riesgo de desarrollo:	Media
Descripción:	El administrador tiene la opción de ingresar una zona y el número de parqueaderos que tendrá la misma. Puede visualizar los estados de las zonas y parqueaderos, los cuales podrán ser modificados, en caso de ser necesario pueden eliminarlos.		

*Elaborado por: Tatiana Masapanta & Micaela Minga.***Tabla 5***Historia de Usuario 4.*

Número:	4	Usuario:	Administrador
Nombre Historia:	Servicio de solicitud		
Prioridad en negocio:	Alta	Riesgo de desarrollo:	Media
Descripción:	El administrador puede activar la recepción de solicitudes de parqueadero por parte de estudiante, docentes y administrativos. Adicional podrá descargar un archivo de Excel de las solicitudes que han sido enviadas.		

*Elaborado por: Tatiana Masapanta & Micaela Minga.***Tabla 6***Historia de Usuario 5.*

Número:	5	Usuario:	Administrador
Nombre Historia:	Servicio autorizado		
Prioridad en negocio:	Alta	Riesgo de desarrollo:	Media
Descripción:	El administrador podrá cargar el archivo de Excel con las personas autorizadas, también podrá ingresar manualmente a una persona, de la misma forma, tendrá la opción de visualizar o eliminar a la persona.		

Elaborado por: Tatiana Masapanta & Micaela Minga.

Tabla 7*Historia de Usuario 6.*

Número:	6	Usuario:	Administrador
Nombre Historia:	Servicio adicional		
Prioridad en negocio:	Alta	Riesgo de desarrollo:	Media
Descripción:	El administrador podrá ingresar, modificar y visualizar horario, periodo. La carrera cuenta con las opciones anteriormente mencionadas y adicional tiene la opción de eliminar.		

Nota. Código de la configuración para el envío de correos. Elaborado por: Tatiana Masapanta

& Micaela Minga.

Tabla 8*Historia de Usuario 7.*

Número:	7	Usuario:	Administrador
Nombre Historia:	Servicios históricos		
Prioridad en negocio:	Baja	Riesgo de desarrollo:	Baja
Descripción:	El administrador podrá visualizar los reportes de datos históricos.		

Elaborado por: Tatiana Masapanta & Micaela Minga.

Tabla 9*Historia de Usuario 8.*

Número:	8	Usuario:	Administrativo - Estudiante - Docente
Nombre Historia:	Servicio envío de solicitudes		
Prioridad en negocio:	Alta	Riesgo de desarrollo:	Media
Descripción:	Los usuarios podrán llenar el formulario para el envío de una nueva solicitud de parqueadero.		

Nota. Código de la configuración para el envío de correos. Elaborado por: Tatiana Masapanta

& Micaela Minga.

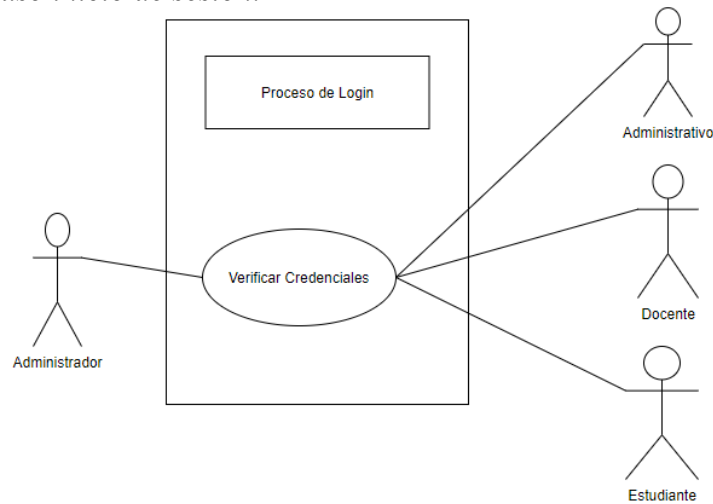
1.11. DISEÑO

El sistema va a ser diseñado en base a los diagramas UML, con el fin de proporcionar seguridad al construir el mismo, los cuales, serán detallados a continuación.

1.11.1. Diagrama caso de uso

En la figura 1 se puede observar cómo los usuarios pueden iniciar sesión en el sistema y realizar las acciones que proporciona el mismo.

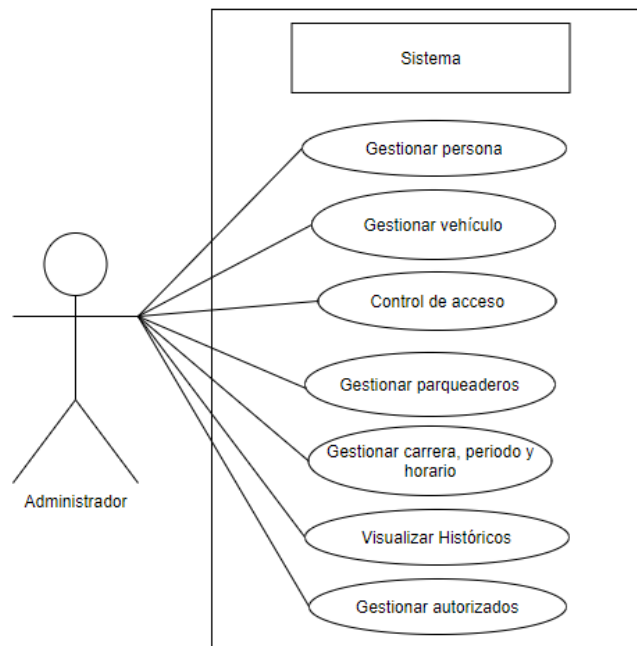
Figura 1
Diagrama caso de uso inicio de sesión.



Nota. Diagrama de caso de uso inicio de sesión para todos los roles. Elaborado por: Tatiana Masapanta & Micaela Minga.

En la figura 2 se puede observar las acciones que puede realizar el administrador luego de iniciar sesión como la gestión de personas, vehículos, parqueaderos, autorizados entre otras.

Figura 2
Diagrama caso de uso para el sistema.

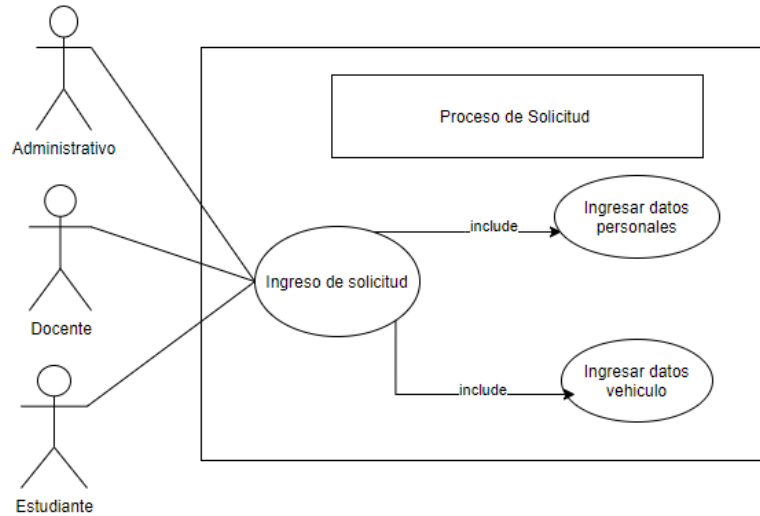


Nota. Diagrama para las actividades que puede realizar el Administrador. Elaborado por: Tatiana Masapanta & Micaela Minga.

En la figura 3 se observa la acción que puede realizar los docentes, estudiantes y administrativos para el envío de solicitud de parqueadero.

Figura 3

Diagrama de uso proceso de solicitud.

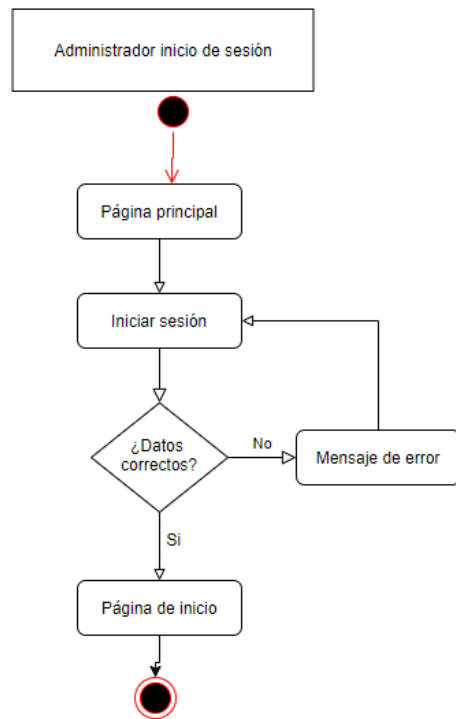


Nota. Diagrama de proceso de solicitud para los roles de administrativos, docentes y estudiantes. Elaborado por: Tatiana Masapanta & Micaela Minga.

1.11.2. Diagrama de actividades

En la figura 4 se puede observar las actividades que deben realizar los usuarios, para iniciar sesión en el sistema y tener acceso a los servicios que corresponde a cada uno.

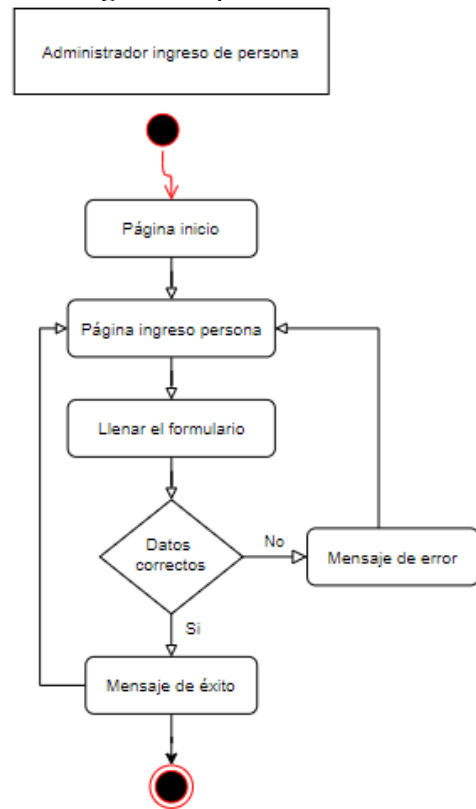
Figura 4
Diagrama de actividades para inicio de sesión.



Nota. Diagrama de actividades para el inicio de sesión para todos los roles. Elaborado por: Tatiana Masapanta & Micaela Minga.

En la figura 5 se muestra las actividades que debe seguir el administrador para el ingreso de una persona. Estas actividades son similares para el ingreso de vehículo, autorizado, parqueadero, zona, carrera, horario y periodo, lo que cambia es la información que se debe llenar para cada uno.

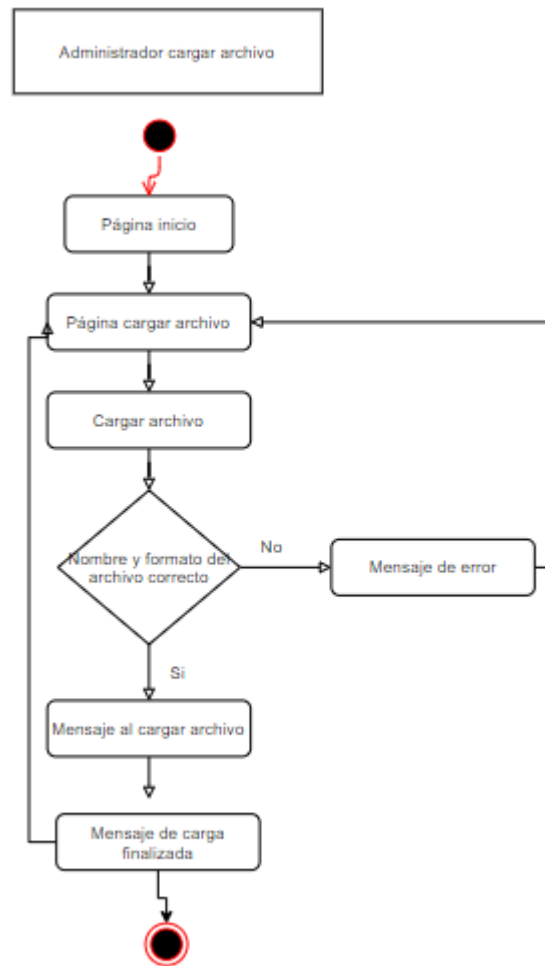
Figura 5
Diagrama de actividades para el ingreso de persona



Nota. Diagrama de actividades para que el administrador pueda ingresar una persona.
Elaborado por: Tatiana Masapanta & Micaela Minga.

En la figura 6 se indica las actividades que debe seguir el administrador para cargar un archivo de Excel que contenga las personas autorizadas para el uso del parqueadero en ese periodo académico.

Figura 6
Diagrama de actividades para cargar un archivo.

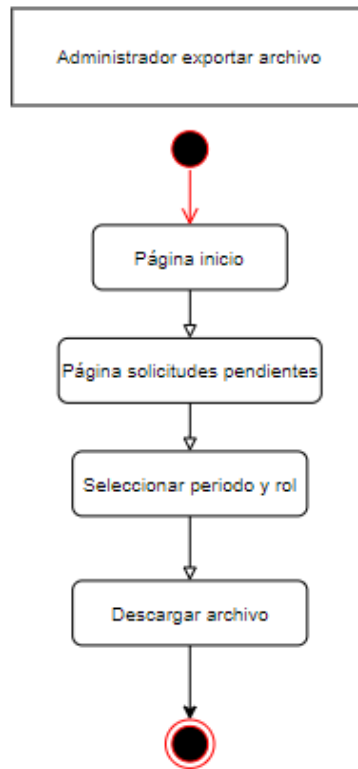


Nota. Diagrama de actividades para cargar un archivo de Excel por parte del administrador.

Elaborado por: Tatiana Masapanta & Micaela Minga.

En la figura 7 se indica las actividades que debe seguir el administrador para exportar un archivo de Excel que contenga la información de las personas que solicitaron el uso de parqueaderos.

Figura 7
Diagrama de actividades para exportar un archivo



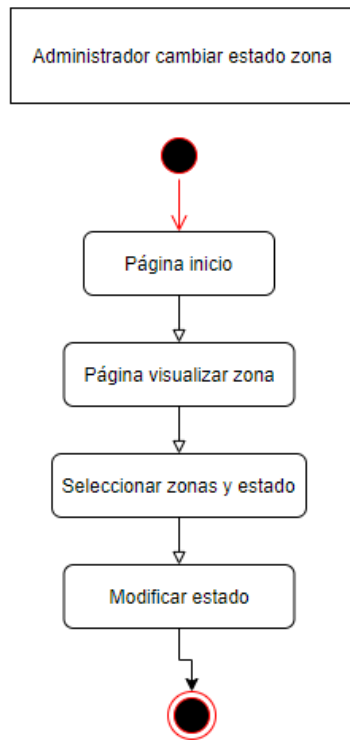
Nota. Diagrama de actividades para exportar un archivo de Excel por parte del administrador.

Elaborado por: Tatiana Masapanta & Micaela Minga.

En la figura 8 se indica las actividades que debe seguir el administrador para cambiar el estado de una zona o parqueadero.

Figura 8

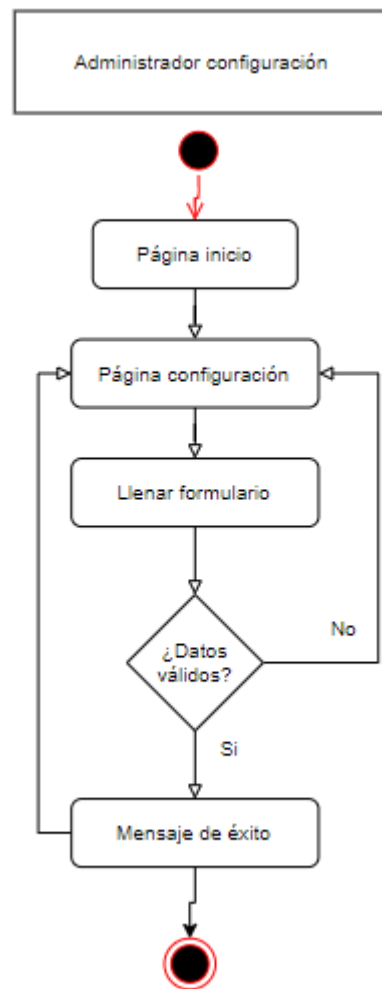
Diagrama de actividades para el cambio de estado de zona.



Nota. Diagrama de actividades para el cambio de estado de zona en un parqueadero por parte del administrador. Elaborado por: Tatiana Masapanta & Micaela Minga.

En la figura 9 se indica las actividades que debe seguir el administrador para cambiar la contraseña o correo electrónico del usuario administrador.

Figura 9
Diagrama de actividades para la configuración.

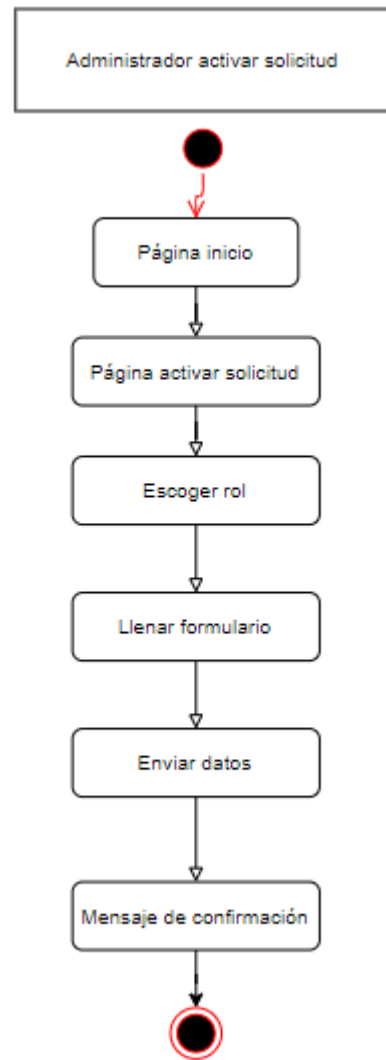


Nota. Diagrama de actividades para la configuración por parte del administrador. Elaborado por: Tatiana Masapanta & Micaela Minga.

En la figura 10 se indica las actividades que debe seguir el administrador para activar el servicio de solicitud.

Figura 10

Diagrama de actividades para activación de solicitud.



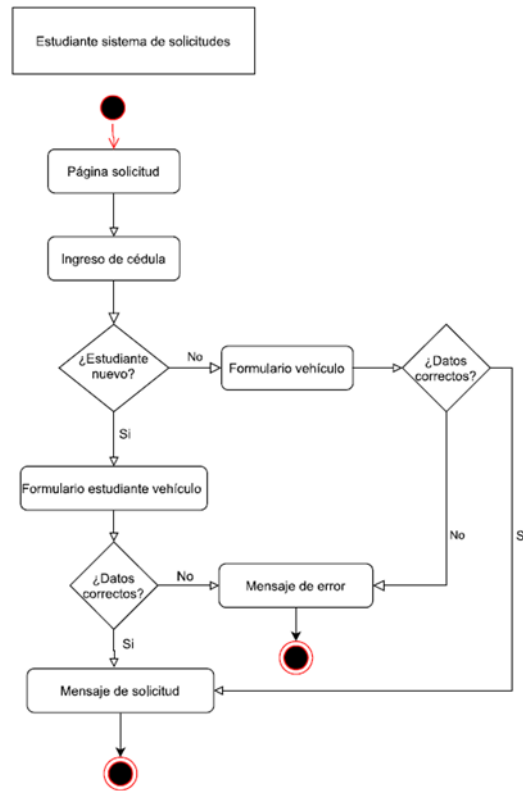
Nota. Diagrama de actividades para activación de solicitud por parte de la administración.

Elaborado por: Tatiana Masapanta & Micaela Minga.

En la figura 11 se indica las actividades que debe seguir el estudiante que esté o no registrado en el sistema de solicitudes para un parqueadero. Las actividades son similares para docentes y administrativos.

Figura 11

Diagrama de actividades para estudiante nuevo y antiguo.



Nota. Diagrama de actividades para ingreso de información de estudiante nuevo y antiguo.

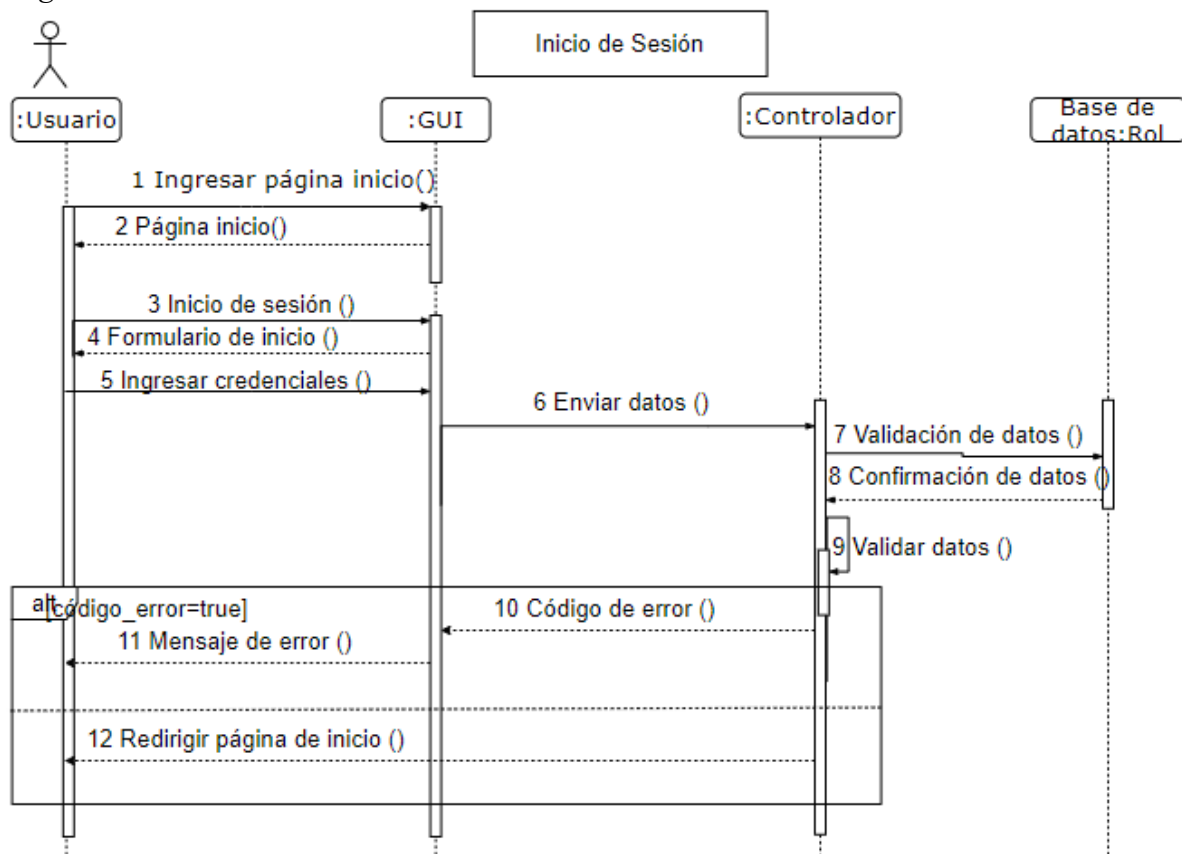
Elaborado por: Tatiana Masapanta & Micaela Minga.

1.11.3. Diagrama de secuencia

En la figura 12 se observa cómo los usuarios pueden iniciar sesión en el sistema y realizar las acciones que proporciona el mismo. Para ello debe acceder a la página principal y dirigirse al inicio de sesión, llenar el usuario y contraseña, los cuales serán validados en la base de datos, después de un acceso exitoso el usuario podrá hacer uso de los servicios.

Figura 12

Diagrama de secuencia inicio de sesión.

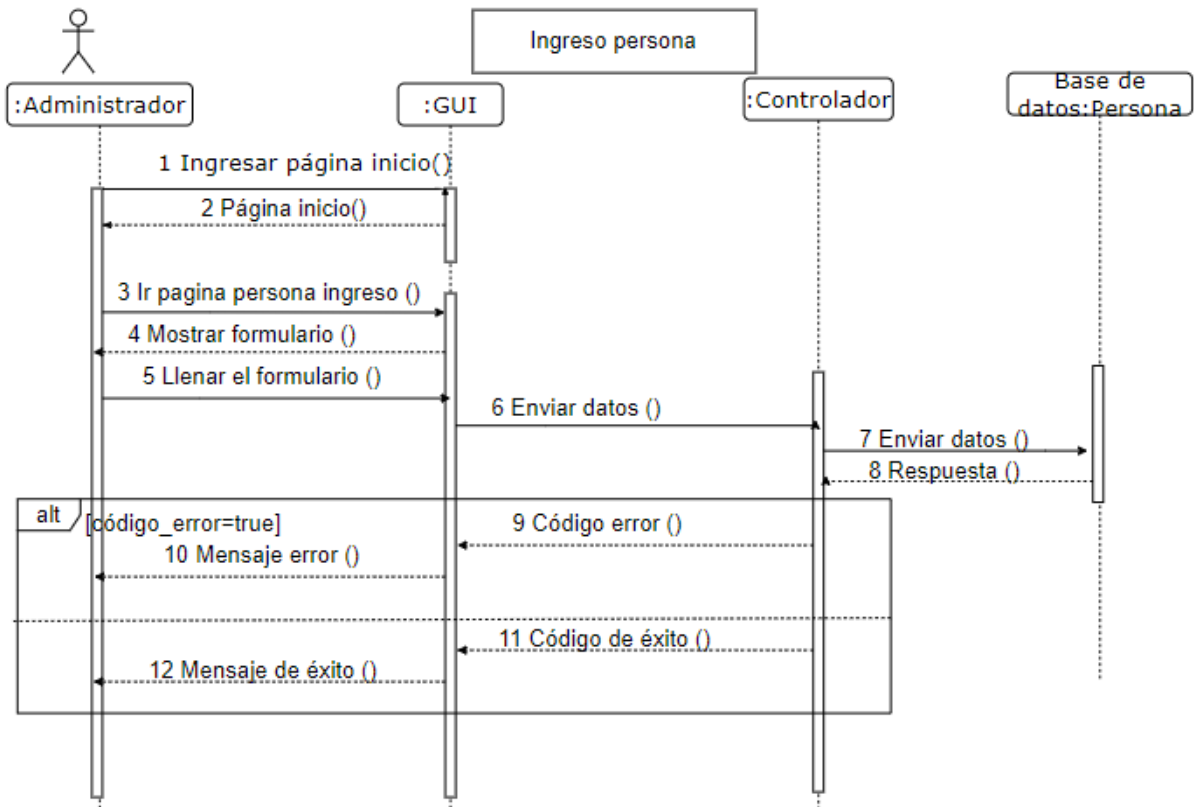


Nota. Diagrama de caso de uso inicio de sesión para todos los roles. Elaborado por: Tatiana Masapanta & Micaela Minga.

En la figura 13 se muestra la secuencia de pasos que debe seguir el administrador para el ingreso de una persona. Luego del iniciar sesión, el administrador debe dirigirse a la página de ingreso persona, llenar los datos, enviarlos al controlador para que los mismos sean enviados a la base de datos, en la GUI se mostrara un mensaje de éxito o error según sea el caso. Estas actividades son similares para el ingreso de vehículo, autorizado, parqueadero, zona, carrera, horario y periodo.

Figura 13

Diagrama de secuencia ingreso de persona.

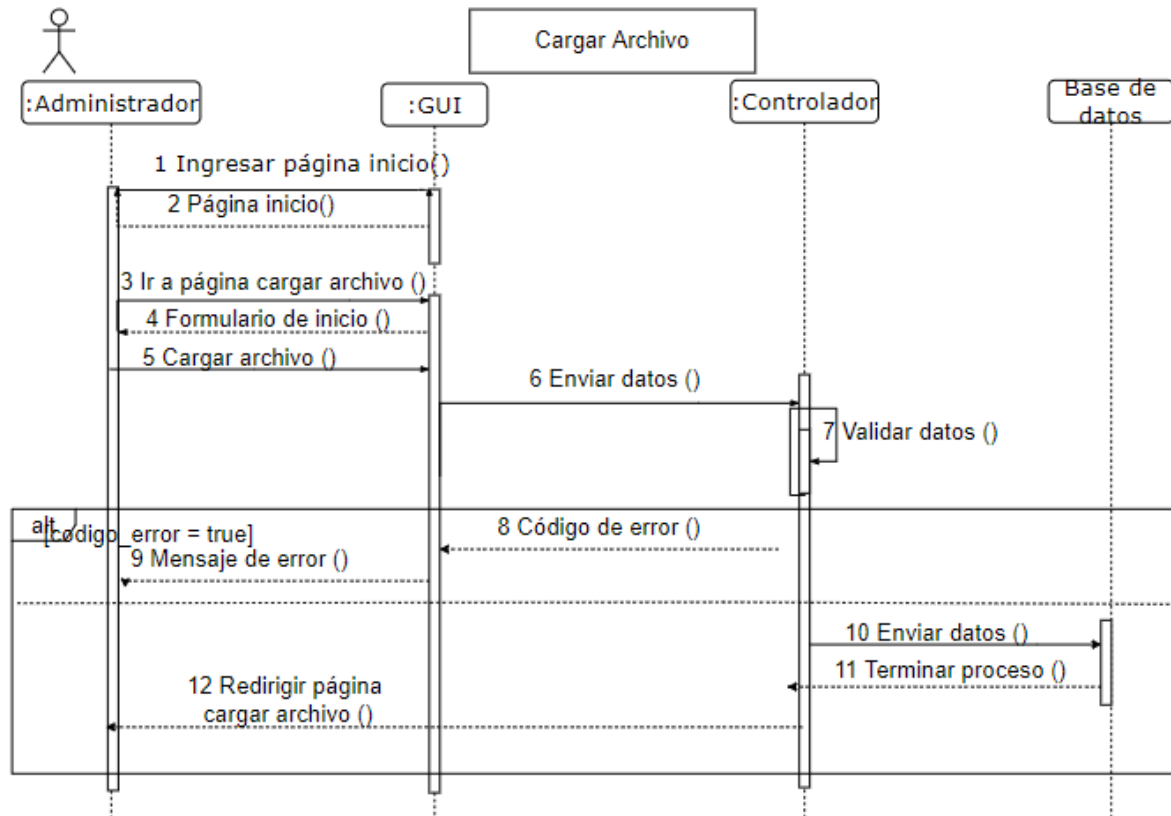


Nota. Diagrama de secuencia para que el administrador pueda ingresar una persona. Elaborado por: Tatiana Masapanta & Micaela Minga.

En la figura 14 se indica la secuencia de pasos que debe seguir el administrador para cargar un archivo de Excel que contenga las personas autorizadas para el uso del parqueadero en ese periodo. Luego del iniciar sesión, el administrador debe dirigirse a la página de cargar archivo, seleccionar el archivo, mismo que es enviado al controlador que valida que el nombre del archivo sea correcto, si no lo es, se muestra un mensaje de error en la GUI, si es correcto se envía la información del archivo a la base de datos, en la GUI se mostrara un mensaje de éxito. Las tablas involucradas para este proceso serán persona, vehículo, sticker y autorizado.

Figura 14

Diagrama de secuencia para cargar un archivo.



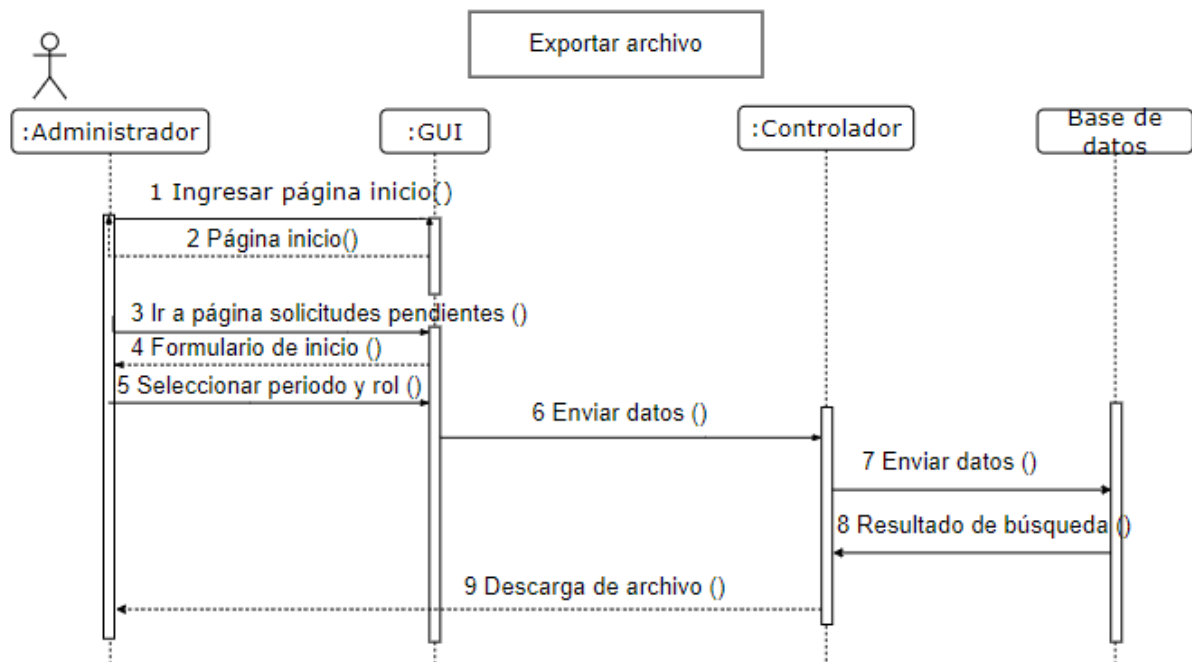
Nota. Diagrama de secuencia para cargar un archivo de Excel por parte del administrador.

Elaborado por: Tatiana Masapanta & Micaela Minga.

En la figura 15 se indica la secuencia de pasos que debe seguir el administrador para exportar un archivo de Excel que contenga la información de las personas que solicitaron el uso de parqueaderos. Luego del iniciar sesión, el administrador debe dirigirse a la página solicitudes pendientes, seleccionar el periodo académico y el rol del cual desea generar el archivo, los cuales son enviados al controlador que realiza la petición a la base de datos, en la GUI se descarga el archivo. Las tablas involucradas para este proceso serán persona, vehículo, horario y autorizado.

Figura 15

Diagrama de secuencia para exportar un archivo.



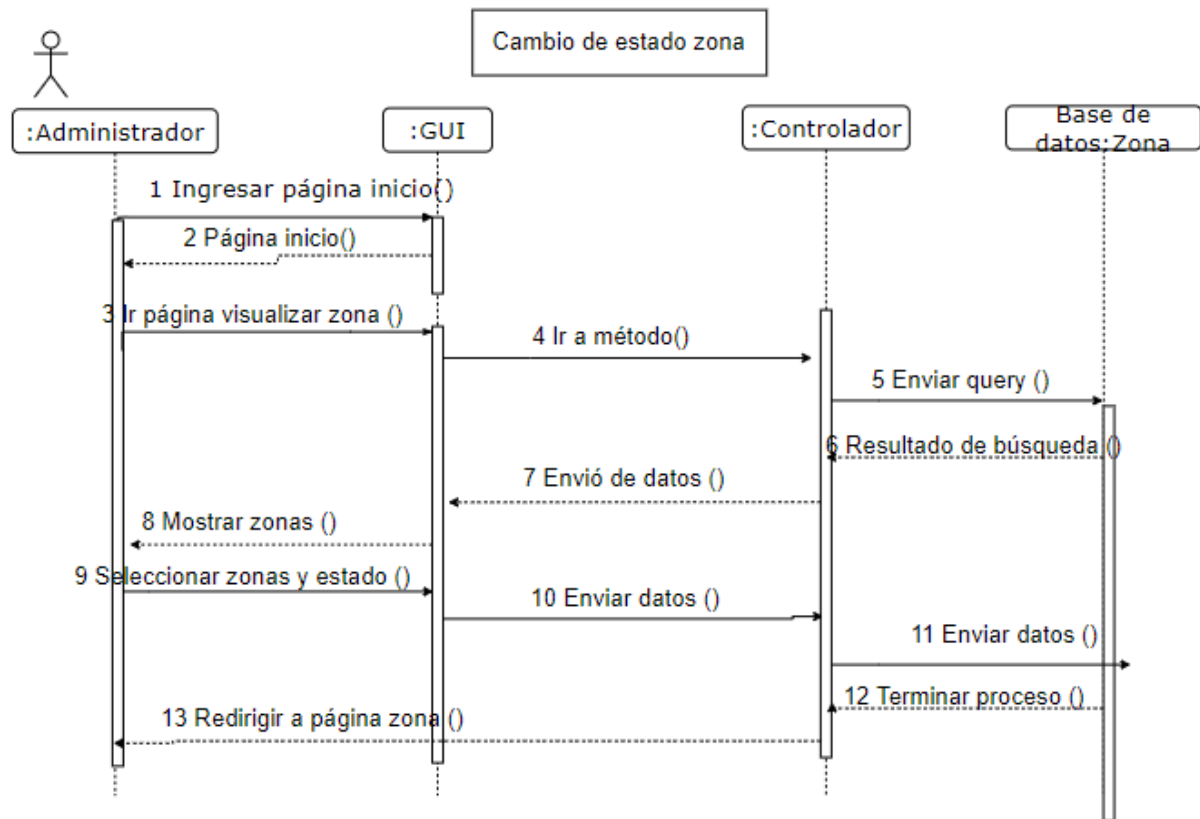
Nota. Diagrama de secuencia para exportar un archivo de Excel por parte del administrador.

Elaborado por: Tatiana Masapanta & Micaela Minga.

En la figura 16 se indica la secuencia de pasos que debe seguir el administrador para cambiar el estado de una zona o parqueadero. Luego del iniciar sesión, el administrador debe dirigirse a la página de visualizar zona, el controlador realiza la búsqueda de las zonas y se las muestra en la GUI. El administrador podrá seleccionar las zonas y el estado, los mismo que serán enviados al controlador, para luego ser enviados a la base de datos, y de esa forma se pueda realizar los cambios y regresar hacia la página para que el usuario pueda visualizar las zonas.

Figura 16

Diagrama de secuencia para el cambio de estado de zona.

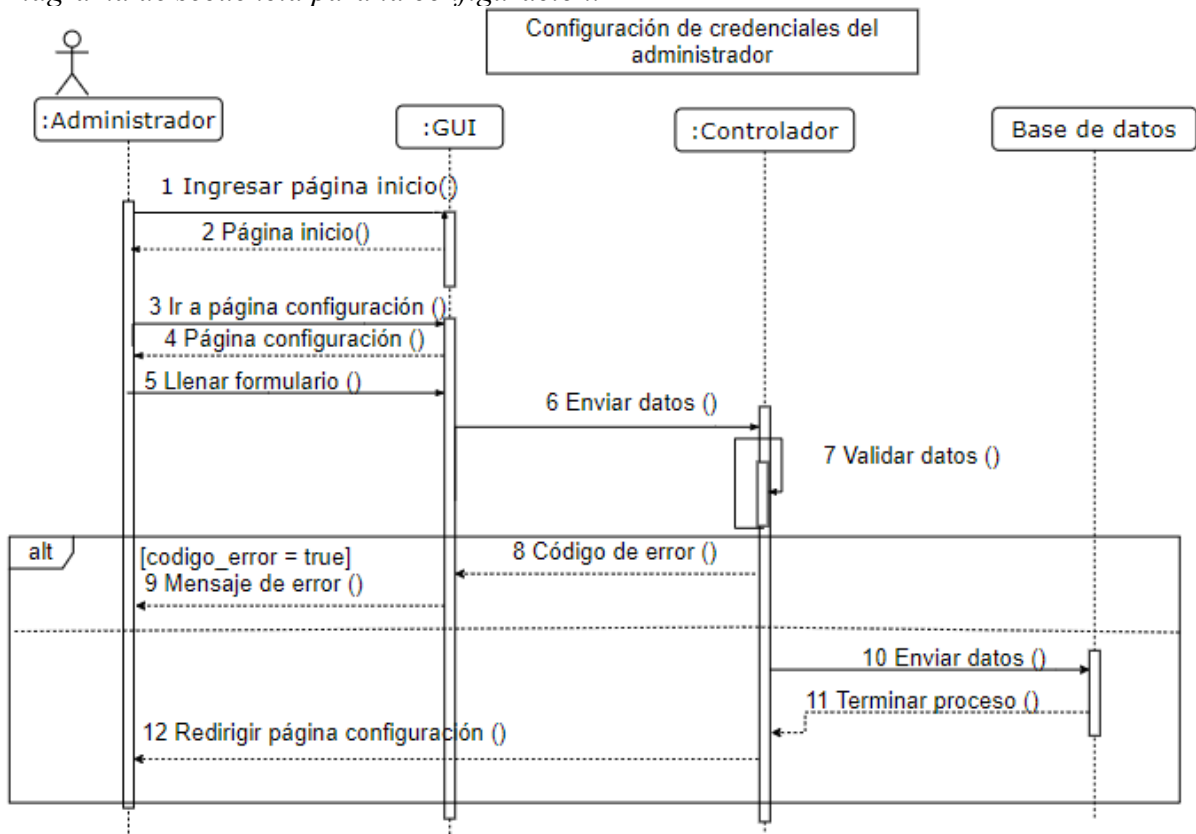


Nota. Diagrama de secuencia para el cambio de estado de zona en un parqueadero por parte del administrador. Elaborado por: Tatiana Masapanta & Micaela Minga.

En la figura 17 se indica la secuencia de pasos que debe seguir el administrador para cambiar la contraseña o correo electrónico. Luego del iniciar sesión, el administrador debe dirigirse a la página de configuración, llenar el formulario, se envía los datos al controlador que valida que los mismos sean correctos, si no lo son se muestra un mensaje de error en la GUI, y si es correcto se envía la información a la base de datos para modificarlos, en la GUI se mostrara un mensaje de éxito.

Figura 17

Diagrama de secuencia para la configuración.

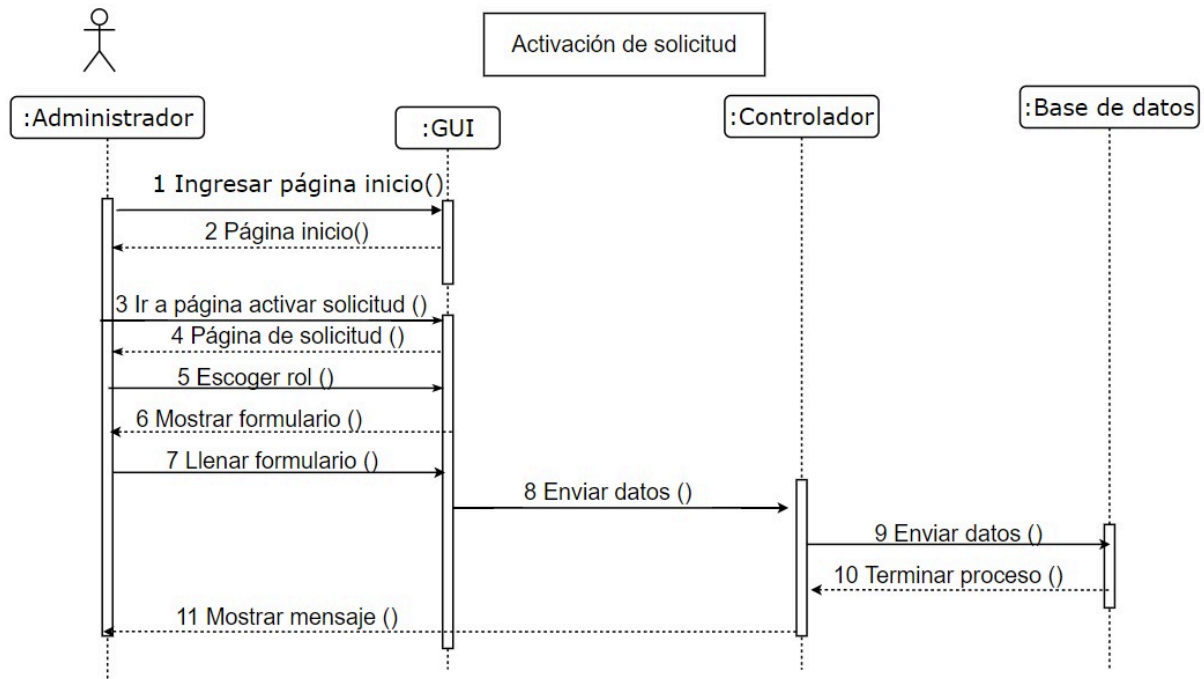


Nota. Diagrama de secuencia para la configuración por parte del administrador. Elaborado por: Tatiana Masapanta & Micaela Minga.

En la figura 18 se indica la secuencia de pasos que debe seguir el administrador para activar el servicio de solicitud. Luego del iniciar sesión, el administrador debe dirigirse a la página de activar solicitud, seleccionar el rol, será enviado a la página para llenar el formulario de activación de solicitud, debe llenar los datos, se envía los datos al controlador, para luego enviar la información a la base de datos, en la GUI se mostrará un mensaje.

Figura 18

Diagrama de secuencia para activación de solicitud.



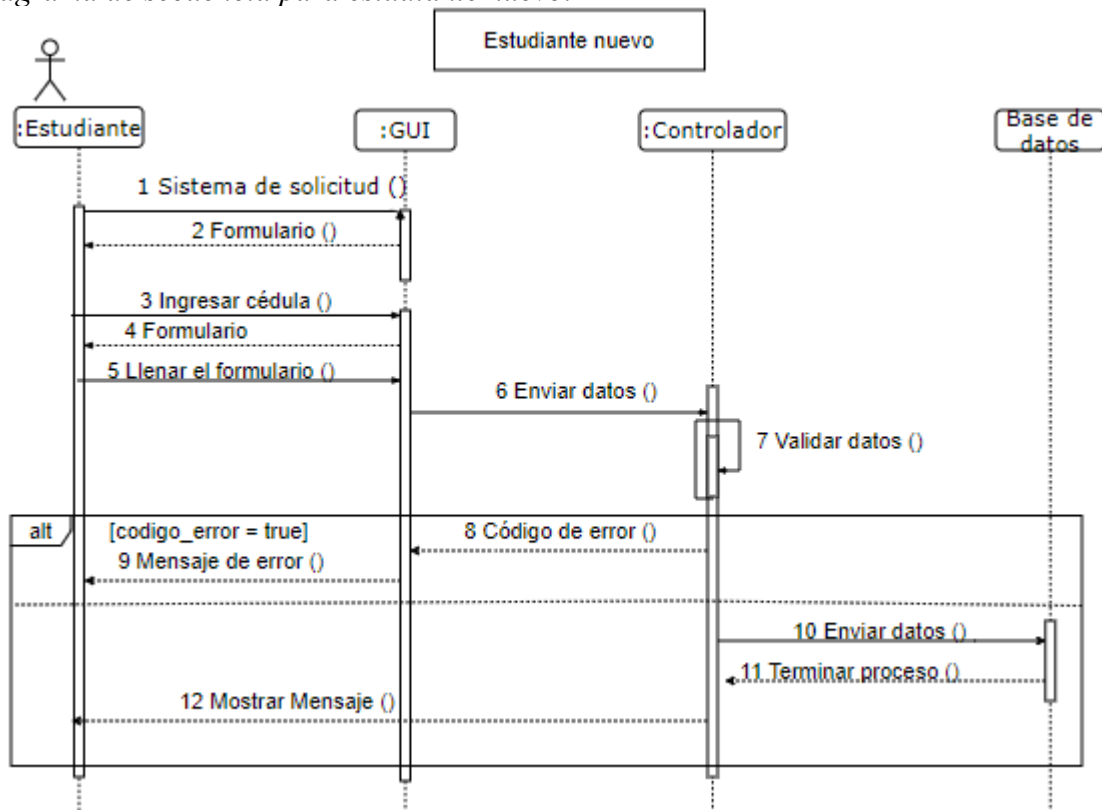
Nota. Diagrama de secuencia para activación de solicitud por parte de la administración.

Elaborado por: Tatiana Masapanta & Micaela Minga.

En la figura 19 se indica la secuencia de pasos que debe seguir el estudiante que no esté registrado en el sistema de solicitudes para un parqueadero. Luego del iniciar sesión, el estudiante debe ingresar el número de cédula, luego se le redirige al formulario, llenar todos los datos del formulario, se envían al controlador donde se valida los datos, si no son correctos se muestra un mensaje de error en la GUI, si es correcto se envía la información a la base de datos, en la GUI se mostrará un mensaje de éxito. La secuencia es similar para docentes y administrativos.

Figura 19

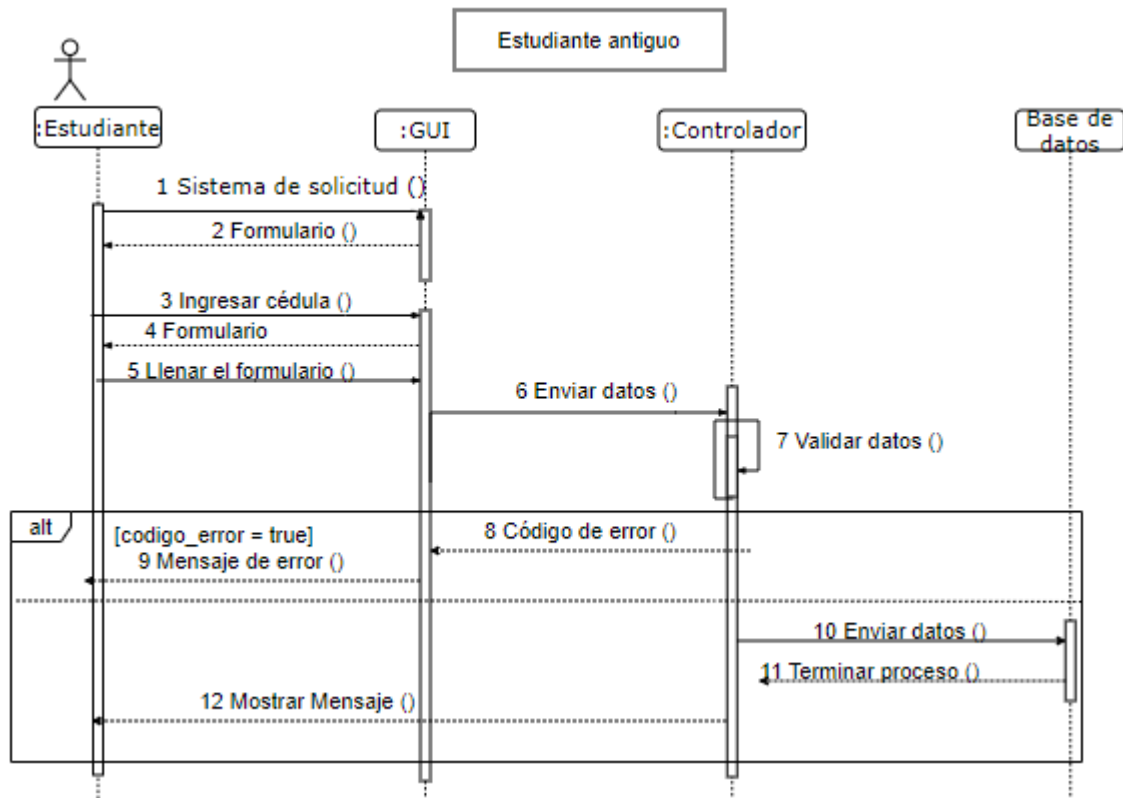
Diagrama de secuencia para estudiante nuevo.



Nota. Diagrama de secuencia para ingreso de información de estudiante nuevo. Elaborado por: Tatiana Masapanta & Micaela Minga.

En la figura 20 se indica la secuencia de pasos que debe seguir el estudiante que esté registrado en el sistema de solicitudes para un parqueadero. Luego del iniciar sesión, el estudiante debe ingresar el número de cédula, luego se le redirige al formulario, llenar todos los datos del formulario, se envían al controlador donde se valida los datos, si no son correctos se muestra un mensaje de error en la GUI, si es correcto se envía la información a la base de datos, en la GUI se mostrará un mensaje de éxito. La secuencia es similar para docentes y administrativos.

Figura 20
Diagrama de secuencia para estudiante antiguo.

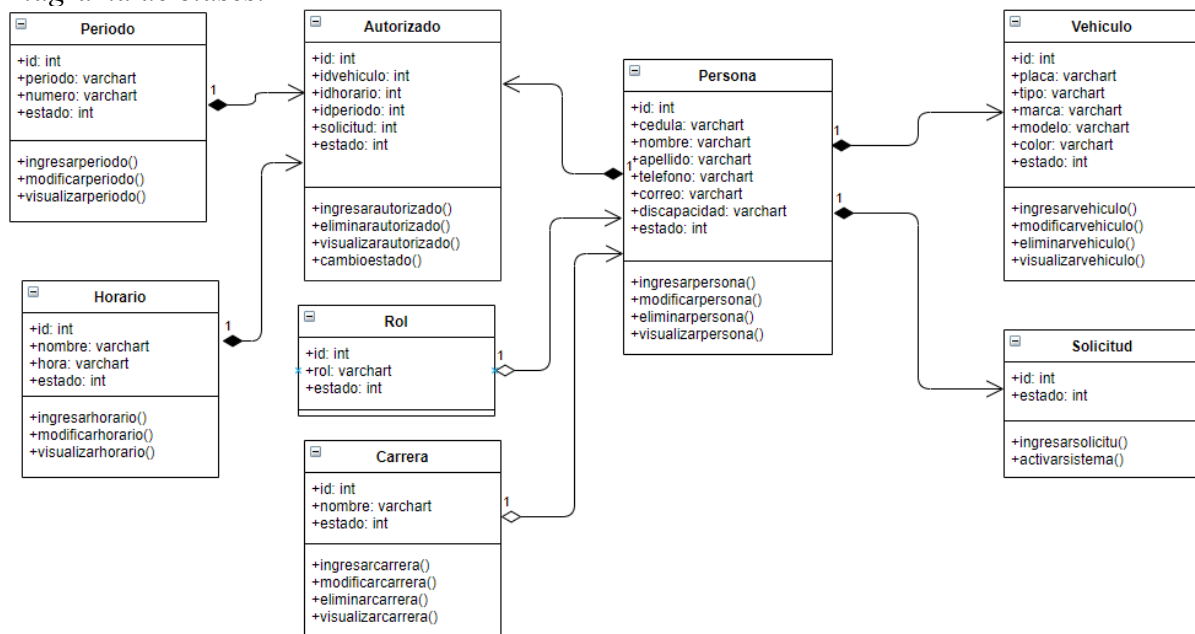


Nota. Diagrama de secuencia para ingreso de información de estudiante antiguo. Elaborado por: Tatiana Masapanta & Micaela Minga.

1.11.4. Diagrama de clases

En la figura 21 se indica la relación que existe entre las clases y la dependencia que existe entre ella como por ejemplo no puede existir un vehículo si no existe una persona, por otro lado, no es necesario tener una persona para que exista un rol.

Figura 21
Diagrama de clases.



Nota. Diagrama de clases del sistema. Elaborado por: Tatiana Masapanta & Micaela Minga.

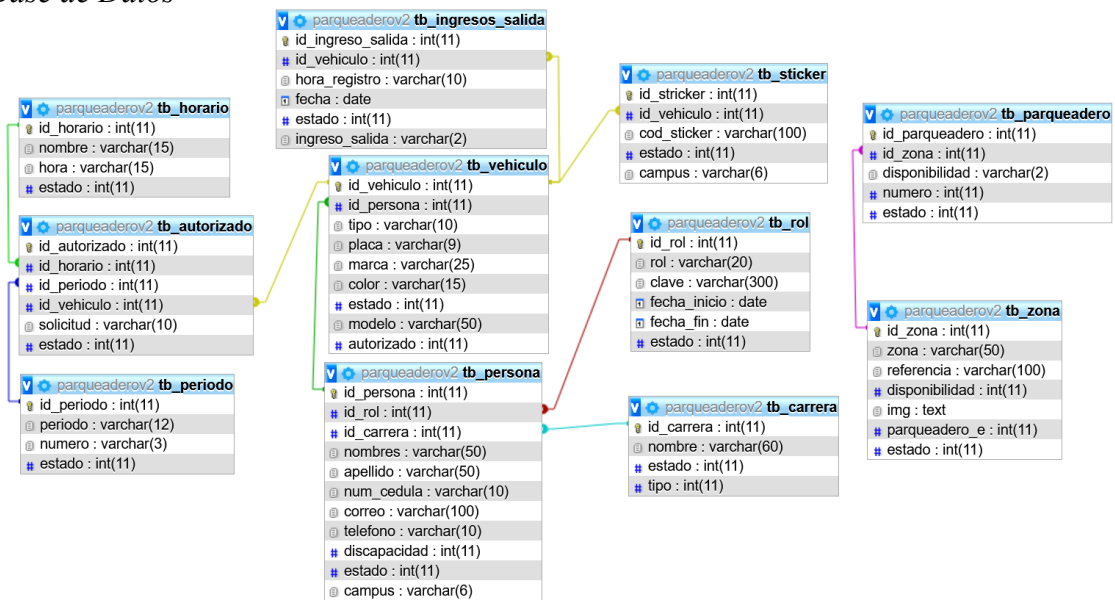
1.11.5. Diagrama entidad relación

El diagrama representa la información que se podría almacenar de los usuarios, La base está representada con algunas tablas, una de ellas es la de vehículo, en el cual se podrá ingresar toda la información referente al vehículo.

La tabla persona almacena la información de los estudiantes, docentes, administrativos que soliciten el uso del parqueadero. La tabla autorizada va a contener la información de los vehículos autorizados con su respectivo estado.

La tabla parqueadero estará relacionada con la tabla zona y en conjunto tendrán la información de las áreas donde se puede ver la disponibilidad de las mismas y cambiar los estados. Lo antes mencionado se puede visualizar en la figura 22.

Figura 22
Base de Datos



Nota. Base de datos. Elaborado por: los autores

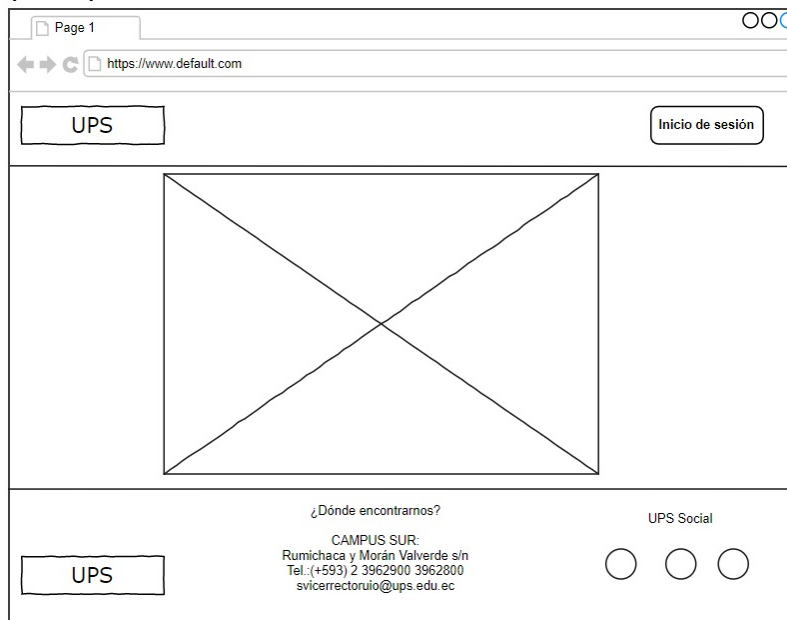
1.11.6. Mockup de interfaces

El diseño del sistema será realizado para dos grupos, el primer grupo está conformado por el Administrador y el segundo grupo por administrativos, docentes y estudiantes.

1.11.6.1. Página principal

Al ingresar al sistema se mostrará a los usuarios la página de inicio como la que se muestra en la figura 23 donde podrán ir al inicio de sesión para acceder a las funciones que les corresponda según el rol.

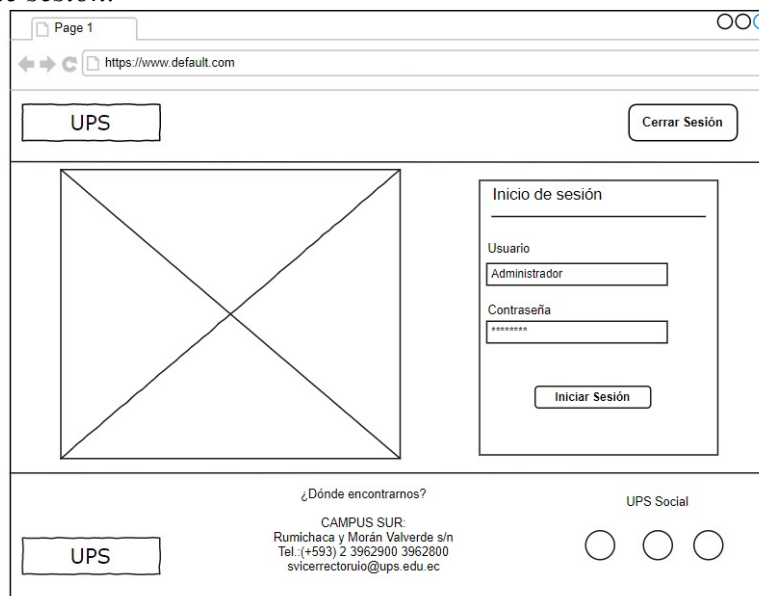
Figura 23
Interfaz: página principal.



Nota. Diseño de la página principal del sistema. Elaborado por: Tatiana Masapanta & Micaela Minga.

En la página de inicio de sesión los usuarios ingresan sus credenciales para acceder a los servicios correspondientes.

Figura 24
Interfaz: inicio de sesión.



Nota. Diseño de la página de inicio de sesión. Elaborado por: Tatiana Masapanta & Micaela Minga.

1.11.6.2. Administrador

En esta pantalla se pedirá al administrador llenar el formulario con la información de la persona para poder ser ingresada al sistema.

Figura 25

Interfaz: Ingreso de personas.

Page 1
https://www.default.com

UPS Cerrar Sesión

PERSONA VEHÍCULO PARQUEADERO SOLICITUD AUTORIZADO OTRO HISTÓRICO CONFIGURACIÓN

Inicio/Persona

Persona
Ingreso
Visualizar
Rol
Cédula

Información

Nombre Apellido

Correo Institucional Cédula

Teléfono Rol

Carrera

Discapacidad

Guardar

¿Dónde encontramos?

UPS

CAMPUS SUR:
Rumichaca y Morán Valverde s/n
Tel.:(+593) 2 3962900 3962800
svicerrectorio@ups.edu.ec

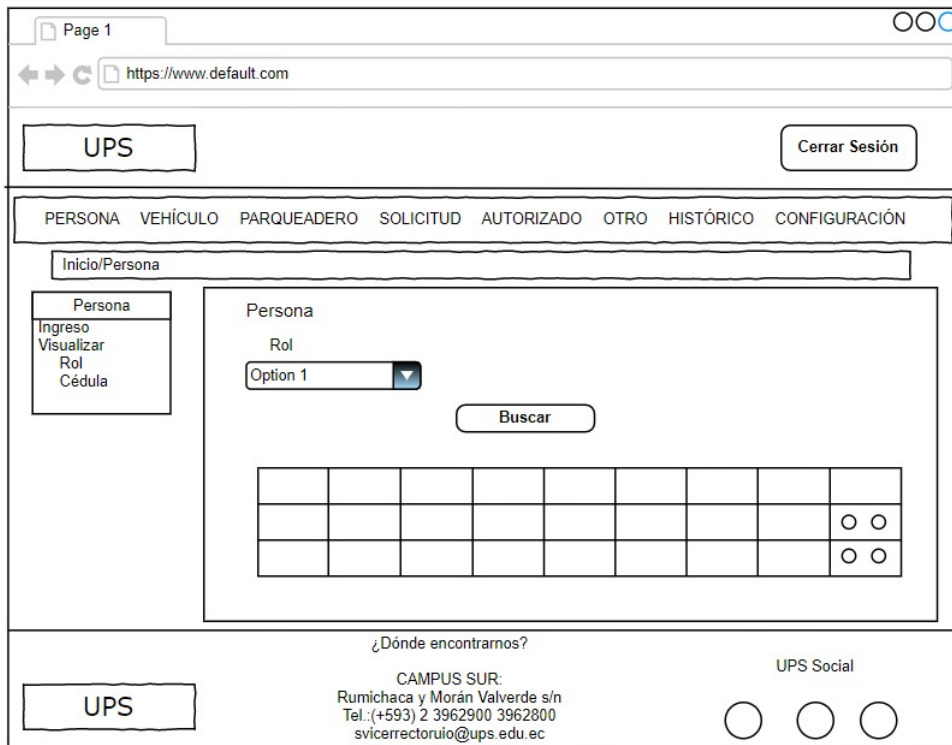
UPS Social

Nota. Ingreso de personas. Elaborado por: Tatiana Masapanta & Micaela Minga.

En esta pantalla se podrá visualizar los datos de las personas en una tabla, para eso debe realizar una búsqueda ya sea por rol o por el número de cédula.

Figura 26

Interfaz: Búsqueda de personas por rol.

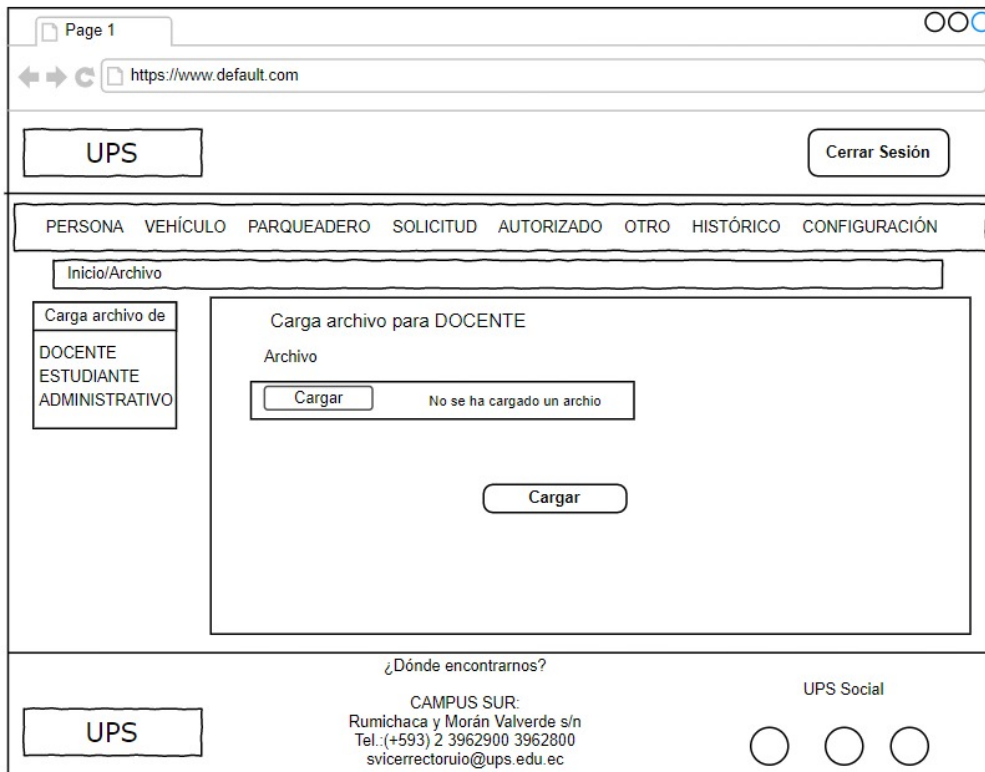


Nota. Búsqueda de personas por rol. Elaborado por: Tatiana Masapanta & Micaela Minga.

En esta interfaz se podrá realizar la carga de un archivo Excel, se debe tener en cuenta que al lado izquierdo están los roles, se debe escoger el rol dependiendo el archivo a cargar.

Figura 27

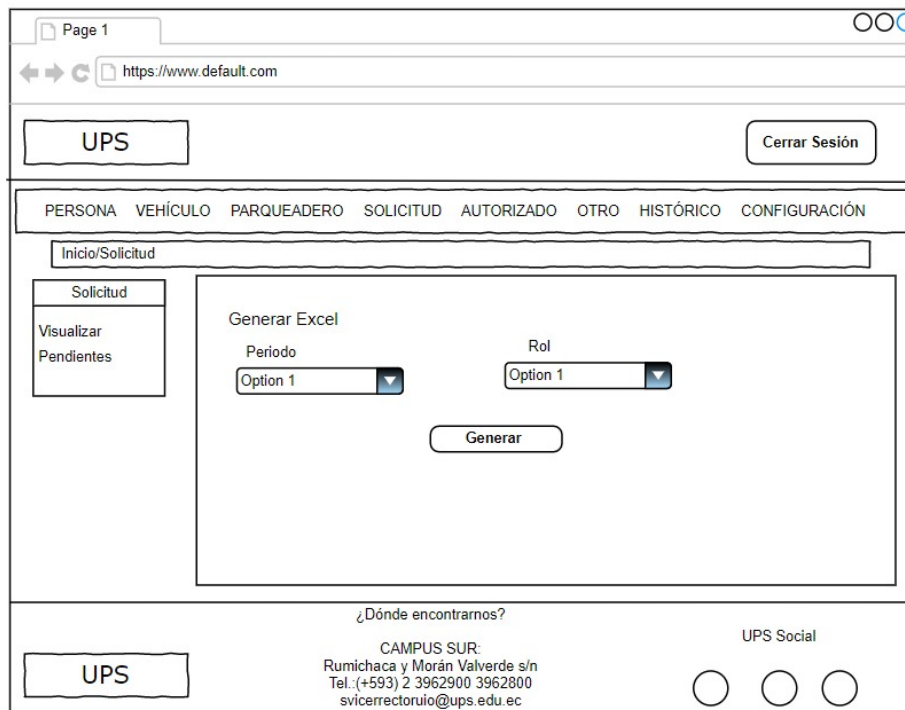
Interfaz: Carga de archivo.



Nota. Carga de archivo Excel. Elaborado por: Tatiana Masapanta & Micaela Minga.

En esta interfaz se puede descargar un archivo de Excel de todas las personas que hayan llenado el formulario de solicitud, dicha acción va a depender del periodo y rol que se elija.

Figura 28
Interfaz: Generación del Excel.

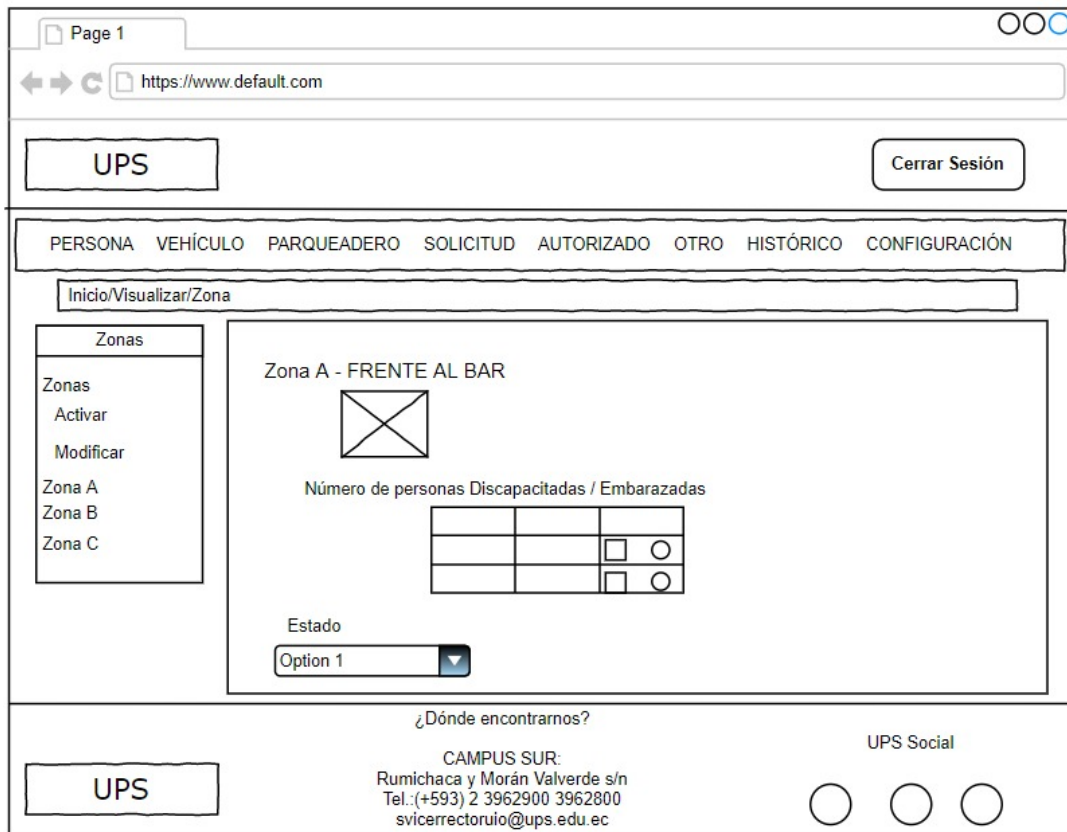


Nota. Generación de Excel. Elaborado por: Tatiana Masapanta & Micaela Minga.

En la interfaz se podrá visualizar una zona de parqueo con el número de parqueaderos para personas discapacitada/embarazadas, y una tabla donde se muestra el número de zonas de parqueo y el estado en el que se encuentra las mismas.

Figura 29

Interfaz: activación de zona.



Nota. Activación de zona. Elaborado por: Tatiana Masapanta & Micaela Minga.

En la interfaz se podrá realizar la activación de solicitudes para los estudiantes, administrativos y docentes, la misma que será conforme al último periodo y se podrá elegir la fecha de activación y cierre, y establecer la clave temporal para que las personas tengan acceso.

Figura 30

Interfaz: activación de solicitudes.

Page 1

https://www.default.com

UPS Cerrar Sesión

PERSONA VEHÍCULO PARQUEADERO SOLICITUD AUTORIZADO OTRO HISTÓRICO CONFIGURACIÓN

Inicio/Solicitud

Solicitud

Activar

Activar sistema de solicitud para DOCENTE

Fecha inicio Fecha fin

Date Date

Periodo numero Periodo

Clave temporal Verificar de clave temporal

Guardar

¿Dónde encontramos?

CAMPUS SUR:
Rumichaca y Morán Valverde s/n
Tel.:(+593) 2 3962900 3962800
svicerrectorio@ups.edu.ec

UPS Social

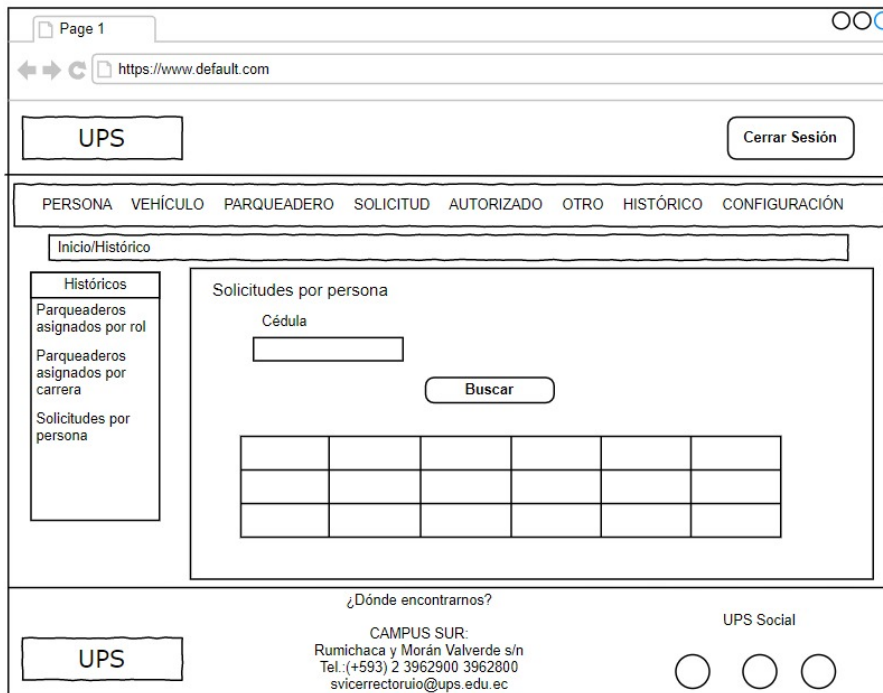
UPS

Nota. Interfaz para activación de solicitudes. Elaborado por: Tatiana Masapanta & Micaela Minga.

En esta pantalla se podrá visualizar la información histórica en forma de tabla, dependiendo el histórico se puede requerir realizar una búsqueda para que se despliegue la información.

Figura 31

Interfaz: Búsqueda de personas por cédula.



Nota. Búsqueda de personas por rol. Elaborado por: Tatiana Masapanta & Micaela Minga.

La interfaz representa cuando el administrador desee realizar un cambio de contraseñas, para lo cual, se encontrará con tres campos.

Figura 32

Interfaz: cambio de contraseñas.

Page 1

https://www.default.com

UPS Cerrar Sesión

PERSONA VEHÍCULO PARQUEADERO SOLICITUD AUTORIZADO OTRO HISTÓRICO CONFIGURACIÓN

Inicio/Configuración

Configuración

Cambio de contraseña

Correo del Administrador

Cambiar contraseña administrador

Contraseña anterior

Contraseña nueva

Verificar la contraseña nueva

Buscar

¿Dónde encontramos?

CAMPUS SUR:
Rumichaca y Morán Valverde s/n
Tel.:(+593) 2 3962900 3962800
svicerrectorio@ups.edu.ec

UPS Social

UPS

Nota. Interfaz para cambio de contraseña. Elaborado por: Tatiana Masapanta & Micaela Minga.

1.11.6.3. Administrativo, docente y estudiante

En la interfaz se podrá realizar el ingreso de solicitud, esta pantalla es para las personas que van a generar por primera vez la solicitud en el sistema.

Figura 33

Interfaz: envío de solicitudes para personas nuevas.

The image shows a web browser window with the URL <https://www.default.com>. The page has a header with a 'UPS' logo on the left and a 'Cerrar Sesión' button on the right. Below the header is a navigation bar with 'Inicio/Estudiante'. The main content area is titled 'Ingresar información' and includes a warning: 'Revise que su cédula este correcta, caso contrario regrese a la página de inicio'. The form contains several fields: 'Número de cédula', 'Nombre', 'Apellido', 'Correo Institucional', 'Teléfono', 'Carrera' (with a dropdown menu), 'Horario' (with a dropdown menu), 'Tiene alguna discapacidad' (with a dropdown menu), 'Tipo' (with a dropdown menu), 'Placa', 'Marca', 'Modelo', and 'Color'. A 'Guardar' button is located at the bottom of the form. The footer contains the text '¿Dónde encontramos?' and 'UPS Social' with three circular icons. Contact information for 'CAMPUS SUR' is provided: Rumichaca y Morán Valverde s/n, Tel.: (+593) 2 3962900 3962800, and svicerrectorio@ups.edu.ec.

Nota. Envío de solicitudes para personas nuevas. Elaborado por: Tatiana Masapanta & Micaela Minga.

En la interfaz se podrá realizar el ingreso de solicitud, esta pantalla es para las personas que ya han generado una solicitud.

Figura 34

Interfaz: envío de solicitudes para personas antiguas.

The image shows a web browser window with the URL <https://www.default.com>. The page has a header with a 'UPS' logo on the left and a 'Cerrar Sesión' button on the right. Below the header is a navigation bar with 'Inicio/Estudiante'. The main content area is titled 'Ingresar información' and contains several form fields: 'Número de cédula' (text input), 'Nombre' (text input), 'Apellido' (text input), 'Horario' (dropdown menu with 'Option 1' selected), 'Información del Vehículo' section with 'Tipo' (dropdown menu with 'Option 1' selected), 'Placa' (text input), 'Marca' (text input), 'Modelo' (text input), and 'Color' (text input). A 'Guardar' button is located at the bottom of the form. At the bottom of the page, there is a footer with the text '¿Dónde encontramos?' and 'UPS Social' with three circular icons. On the left side of the footer, there is a 'UPS' logo and contact information for 'CAMPUS SUR: Rumichaca y Morán Valverde s/n, Tel.: (+593) 2 3962900 3962800, svicerrectorio@ups.edu.ec'.

Nota. Envío de solicitudes para personas antiguas. Elaborado por: Tatiana Masapanta & Micaela Minga.

CAPÍTULO III

CONSTRUCCIÓN Y PRUEBAS

El capítulo va a tratar los temas acerca de las herramientas que se utilizó para el desarrollo de software y su implementación. Para la construcción y pruebas se basó en la metodología ágil XP.

1.12. CONSTRUCCIÓN

1.12.1. Base de datos

En este apartado se indicará la configuración que se realizó en la base de datos. Una vez creada la base de datos en phpMyAdmin, se procede hacer la conexión dentro del framework codeigniter, donde consta una carpeta llamada config y dentro de la misma existe un documento con el nombre de database.php, en el cual se debe poner correctamente las credenciales para la conexión a la base de datos.

La figura 35 representa el código donde se va a realizar la conexión a la base de datos con la que se va a trabajar.

Figura 35

Conexión con a la Base de datos.

```
$db['default'] = array(
    'dsn' => '',
    'hostname' => 'localhost',
    'username' => 'root',
    'password' => '',
    'database' => 'parqueaderov2',
    'dbdriver' => 'mysqli',
    'dbprefix' => '',
    'pconnect' => FALSE,
    //'db_debug' => FALSE,
    'db_debug' => (ENVIRONMENT !== 'production'),
    'cache_on' => FALSE,
    'cachedir' => '',
    'char_set' => 'utf8',
    'dbcollat' => 'utf8_general_ci',
    'swap_pre' => '',
    'encrypt' => FALSE,
    'compress' => FALSE,
    'stricton' => FALSE,
    'failover' => array(),
    'save_queries' => TRUE
);
```

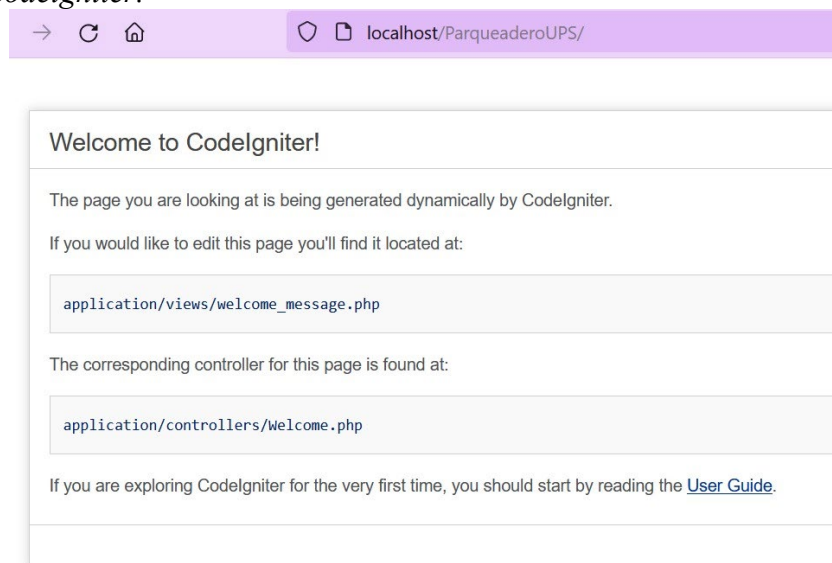
Nota. Conexión con a la Base de datos. Elaborado por: Tatiana Masapanta & Micaela Minga.

1.12.2. Implementación del Framework

1.12.2.1. Codeigniter. Una vez descargado el framework se debe descomprimir y ubicar la carpeta dentro de la ruta establecida por Xampp. Dirigirse al editor y abrir la carpeta que contiene el framework para poder utilizarlo. Para acceder a través del navegador se debe poner localhost/NombredCarpeta.

La figura 36 representa la página principal que tiene por defecto el framework codeigniter.

Figura 36
Framework Codeigniter.



Nota. Framework Codeigniter. Elaborado por: Tatiana Masapanta & Micaela Minga.

1.12.3. Modelo-Vista-Controlador

1.12.3.1. Modelo. Dentro del modelo se coloca el código que se va a utilizar para realizar las diferentes consultas a la base de datos.

La figura 37 representa el código que se usa para realizar una consulta en la tabla persona dentro del modelo, de una forma similar se puede realizar diferentes consultas.

Figura 37
Framework Codeigniter.

```

<?php

defined('BASEPATH') OR exit('No direct script access allowed');

class Persona_model extends CI_Model {
    public function __construct() {
        $this->load->database();
    }

    public function save($data) {
        $this->db->insert('tb_persona', $data);
    }

    public function getPersona($id) {
        $this->db->select('*');
        $this->db->from('tb_persona');
        $this->db->where("id_persona='$id'");
        $resultados= $this->db->get();
        return $resultados->result();
    }
}

```

Nota. Framework Codeigniter. Elaborado por: Tatiana Masapanta & Micaela Minga.

1.12.3.2. Vista. Se desarrolla el código para las diferentes interfaces que serán visualizadas por los usuarios.

La figura 38 representa el código que se usa para las interfaces dentro de la vista.

Figura 38
Código de la vista

```

<div id="all">
    <div id="content">
        <div class="container">
            <div class="row">
                <div class="col-lg-12">
                    <!-- Breadcrumbs -->
                    <nav aria-label="breadcrumb">
                        <ol class="breadcrumb">
                            <li class="breadcrumb-item"><a href="<?php echo base_url() ?>inicio">Inicio</a></li>
                            <li aria-current="page" class="breadcrumb-item active">Persona</li>
                        </ol>
                    </nav>
                </div>
                <div class="col-lg-3">
                    <!--
                    *** PAGES MENU ***
                    -->
                    <div class="card sidebar-menu mb-4">
                        <div class="card-header">
                            <h3 class="h4 card-title">Persona</h3>
                        </div>
                        <div class="card-body">
                            <ul class="nav nav-pills flex-column">
                                <li><a href="<?php echo base_url() ?>persona" class="nav-link">Ingreso</a></li>
                                <li class="nav-link" style="color:#2b909d;">Visualizar</li>
                                <li class="nav nav-pills flex-column">
                                    <li><a href="<?php echo base_url() ?>ver_persona" class="nav-link">Rol</a></li>
                                    <li><a href="<?php echo base_url() ?>ver_i_persona" class="nav-link">Cédula</a></li>
                                </li>
                            </ul>
                        </div>
                    </div>
                    <!-- *** PAGES MENU END *** -->
                </div>
            </div>
            <div class="col-lg-9">
                <div id="contact" class="box">
                    <h2>Información</h2>
                    <form action="<?php echo base_url() ?>persona/save" method="post" >
                        <div class="row">
                            <div class="col-md-6">
                                <div class="form-group">

```

Nota. Código de la vista para la ventana de ingresar persona. Elaborado por: Tatiana Masapanta & Micaela Minga.

La figura 39 representa la ventana que se usa para el ingreso de una persona.

Figura 39

Interfaz generada por el código de la vista.

UNIVERSIDAD POLITÉCNICA SALESIANA ECUADOR

Cerrar Sesión

PERSONA ▾ VEHÍCULO ▾ PARQUEADERO ▾ SOLICITUD ▾ AUTORIZADO ▾ OTRO ▾ HISTÓRICO CONFIGURACIÓN ▾

Inicio / Persona

Persona

Ingreso

Visualizar

Rol

Cédula

Información

Nombre

Apellido

Correo Institucional @ups.edu.ec

Número de Cédula

Teléfono

Rol ADMINISTRATIVO ▾

Campus

Departamento

Nota. Interfaz para el ingreso de la persona. Elaborado por: Tatiana Masapanta & Micaela Minga.

1.12.3.3. Controlador. Dentro del controlador se desarrolla el código que va a realizar el llamado a las vistas y consultas al modelo mediante web service.

La figura 40 representa el código que se usa para llamar a las vistas y consultas dentro del controlador.

Figura 40
Código del controlador

```
k?php
defined('BASEPATH') OR exit('No direct script access allowed');

class Persona extends CI_Controller {

    function __construct() {
        parent::__construct();
        $this->load->helper('intermediario');
    }

    public function index() {

        if ($this->session->usuario == 'ADMINISTRADOR') {

            $a = array();

            $url = 'http://localhost/cambio2/api/persona/buscar';
            $metodo = 'GET';
            $resultado = conectar($url, $a, $metodo);

            $this->load->view('paginas/Header/header');
            $this->load->view('paginas/Ingreso/persona', $resultado);
            $this->load->view('paginas/Header/footer');
        } else {
            redirect(base_url());
        }
    }

    public function save() {
        $nombre = trim($this->input->post('nombre'));
        $apellido = trim($this->input->post('apellido'));
        $email = trim($this->input->post('email'));
        $cedula = trim($this->input->post('cedula'));
        $telefono = trim($this->input->post('telefono'));
        $rol = trim($this->input->post('rol'));
        $campus = trim($this->input->post('campus'));
    }
}
```

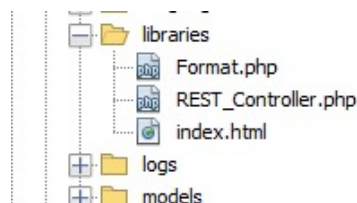
Nota. Código del controlador para hacer el llamado a la vista persona y una función para guardar los datos. Elaborado por: Tatiana Masapanta & Micaela Minga.

1.12.4. Web Service

Para la implementación web services utilizando el framework codeigniter se necesita usar dos librerías adicionales denominadas Format.php y REST_Controller.php, las cuales deben ser colocadas dentro de la carpeta libraries.

La figura 41 representa las librerías que fueron utilizadas para el desarrollo del proyecto.

Figura 41
Carpeta de las librerías



Nota. Librerías utilizadas para el web service. Elaborado por: Tatiana Masapanta & Micaela Minga.

Se debe crear un controlador con la extensión REST_Controller, dentro del cual se debe crear las funciones usando los métodos get, post, put o delete, y hacer el llamado a las funciones de los modelos para enviar la información obtenida mediante un response y de esa forma, obtener los datos en formato json.

La figura 42 representa el controlador con la extensión REST_Controller haciendo el llamado a la función buscar.

Figura 42
Código REST_Controller

```
<?php
defined('BASEPATH') OR exit('No direct script access allowed');
require APPPATH . 'libraries/REST_Controller.php';
require APPPATH . 'libraries/Format.php';

class Persona extends REST_Controller {

    function __construct() {
        parent::__construct();
        $this->load->model('Persona_model');
    }

    public function buscar_get() {
        $data1 = array('data1' => $this->Persona_model->getRol());
        $data2 = array('data2' => $this->Persona_model->getCarreras());
        $data3 = array('data3' => $this->Persona_model->getDepartamentos());

        $resultado = array_merge($data1, $data2,$data3);

        $this->response($resultado, REST_Controller::HTTP_OK);
    }
}
```

Nota. Fragmento del código de un controlador para web service. Elaborado por: Tatiana Masapanta & Micaela Minga.

Para conectar el controlador del aplicativo con el del web service, se necesita de un intermediario que realice la petición para luego decodificar el archivo json, y de esa manera, enviar la respuesta al controlador del aplicativo.

La figura 43 representa el intermediario que realiza la petición para decodificar el archivo json.

Figura 43
Código del intermediario

```
<?php

function conectar($url, $a, $metodo) {
    $curl = curl_init();

    curl_setopt_array($curl, array(
        CURLOPT_URL => $url,
        CURLOPT_RETURNTRANSFER => true,
        CURLOPT_ENCODING => '',
        CURLOPT_MAXREDIRS => 10,
        CURLOPT_TIMEOUT => 0,
        CURLOPT_FOLLOWLOCATION => true,
        CURLOPT_HTTP_VERSION => CURL_HTTP_VERSION_1_1,
        CURLOPT_CUSTOMREQUEST => $metodo,
        CURLOPT_POSTFIELDS => $a,
    ));

    $response = curl_exec($curl);
    return $data = json_decode($response, true);

    curl_close($curl);
}

?>
```

Nota. Código del intermediario para llamar al servidor y decodificar el formato json, que se da como respuesta. Elaborado por: Tatiana Masapanta & Micaela Minga.

1.12.4.1. Postman. Para probar el correcto funcionamiento del web service se utilizó postman, con el fin de verificar el resultado que devuelve las peticiones.

La figura 44 representa la petición que realiza dentro de la herramienta postman

Figura 44
Herramienta Postman

The screenshot shows a Postman interface for a POST request to `http://localhost/cambio2/api/vehiculo/ver`. The request body is set to `form-data` with the following parameters:

KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/>	placa	PER-1234
<input checked="" type="checkbox"/>	combo	1
Key	Value	Description

The response is shown in JSON format:

```
{
  "data": [
    {
      "id_persona": "1",
      "num_cedula": "1743263748",
      "nombres": "JUAN DANIEL",
      "apellido": "GUERRA",
      "id_vehiculo": "6",
      "placa": "PER-1234",
      "tipo": "CARRO",
      "marca": "CHEVROLET",
      "color": "PLOMO",
      "autorizado": "2",
      "modelo": "SPARK"
    }
  ]
}
```

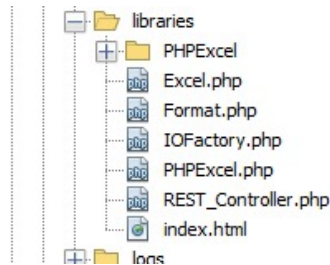
Nota. Prueba del web service utilizando postman. Elaborado por: Tatiana Masapanta & Micaela Minga.

1.12.5. *Importación y exportación del Excel*

Para la importación y exportación de un archivo Excel se necesita usar librerías adicionales como PHPExcel, las cuales deben ser colocadas dentro de la carpeta libraries.

La figura 45 representa las librerías usadas para la importación y exportación del Excel.

Figura 45
Carpeta libraries



Nota. Librerías utilizadas para la importación y exportación de un archivo Excel. Elaborado por: Tatiana Masapanta & Micaela Minga.

1.12.6. *Implementación del correo*

Para la implementación del correo se debe realizar hacer una modificación en los archivos php.ini y el sendmail.ini que se encuentra en la carpeta de Xampp. Los requisitos para él enviaron de correos usando Gmail, es crear una cuenta y generar una contraseña de aplicación. Por último, se debe realizar la implementación de la librería email y colocar las credenciales dentro del archivo correspondiente.

La figura 46 representa el código utilizado para la implementación del correo.

Figura 46
Código para envío de correo

```
$config = Array(  
    'protocol' => 'smtp',  
    'smtp_host' => 'ssl://smtp.googlemail.com',  
    'smtp_port' => 465,  
    'smtp_user' => '*****@gmail.com',  
    'smtp_pass' => '*****',  
    'mailtype' => 'html',  
    'charset' => 'utf-8'  
);  
$CI =& get_instance();  
$CI->load->library('email');
```

Nota. Código de la configuración para el envío de correos. Elaborado por: Tatiana Masapanta & Micaela Minga.

1.12.7. *Encriptación de contraseñas*

El sistema cuenta con la encriptación de contraseñas, la cual se realizó mediante el método con el que cuenta el framework codeigniter, de la misma forma, se puede tener la descriptación de contraseñas.

La figura 47 representa el código para la encriptación de contraseñas.

Figura 47

Código para la encriptación de contraseñas

```
$encrypted_string = $this->encryption->encrypt($clave);
```

Nota. Código para la encriptación para las diferentes contraseñas. Elaborado por: Tatiana Masapanta & Micaela Minga.

La figura 48 representa el código para la descriptación de contraseñas.

Figura 48

Código para la descriptación de contraseñas

```
$clave_base = $this->encryption->decrypt($resultado[0]['clave']);
```

Nota. Código para la descriptación para las diferentes contraseñas. Elaborado por: Tatiana Masapanta & Micaela Minga.

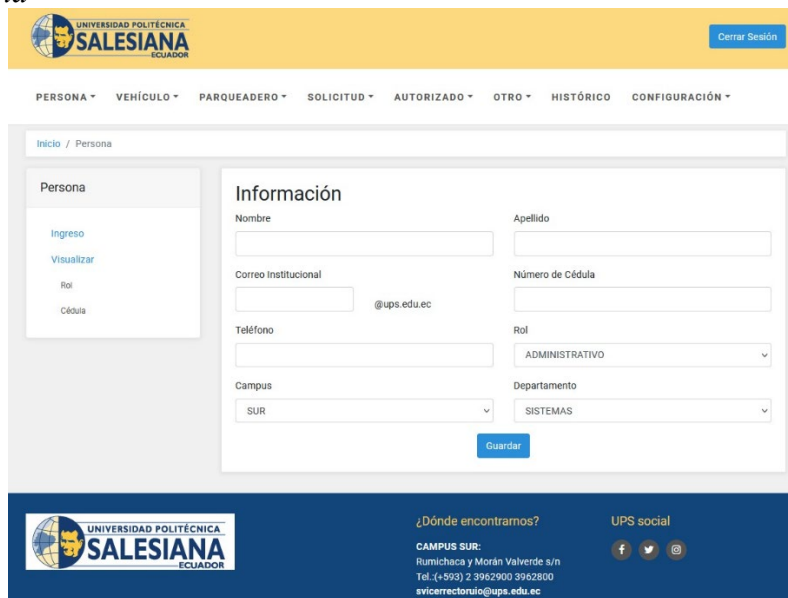
1.12.8. *Interfaz de usuario*

En base a las historias de usuarios generadas previamente se desarrolló el sistema.

1.12.8.1. Administrador

La figura 49 representa la pantalla para el ingreso de una persona por parte del administrador.

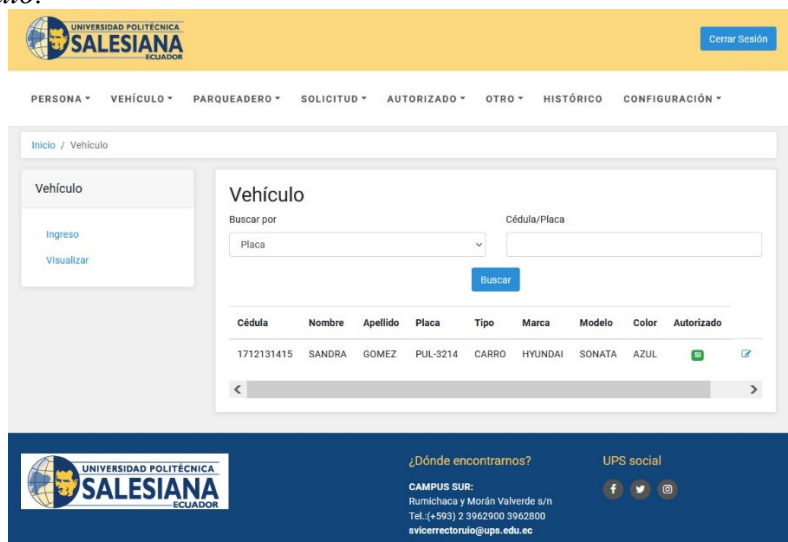
Figura 49
Pantalla Persona



Nota. Pantalla para el ingreso de la persona. Elaborado por: Tatiana Masapanta & Micaela Minga.

La figura 50 representa la pantalla para el ingreso del vehículo por parte del administrador.

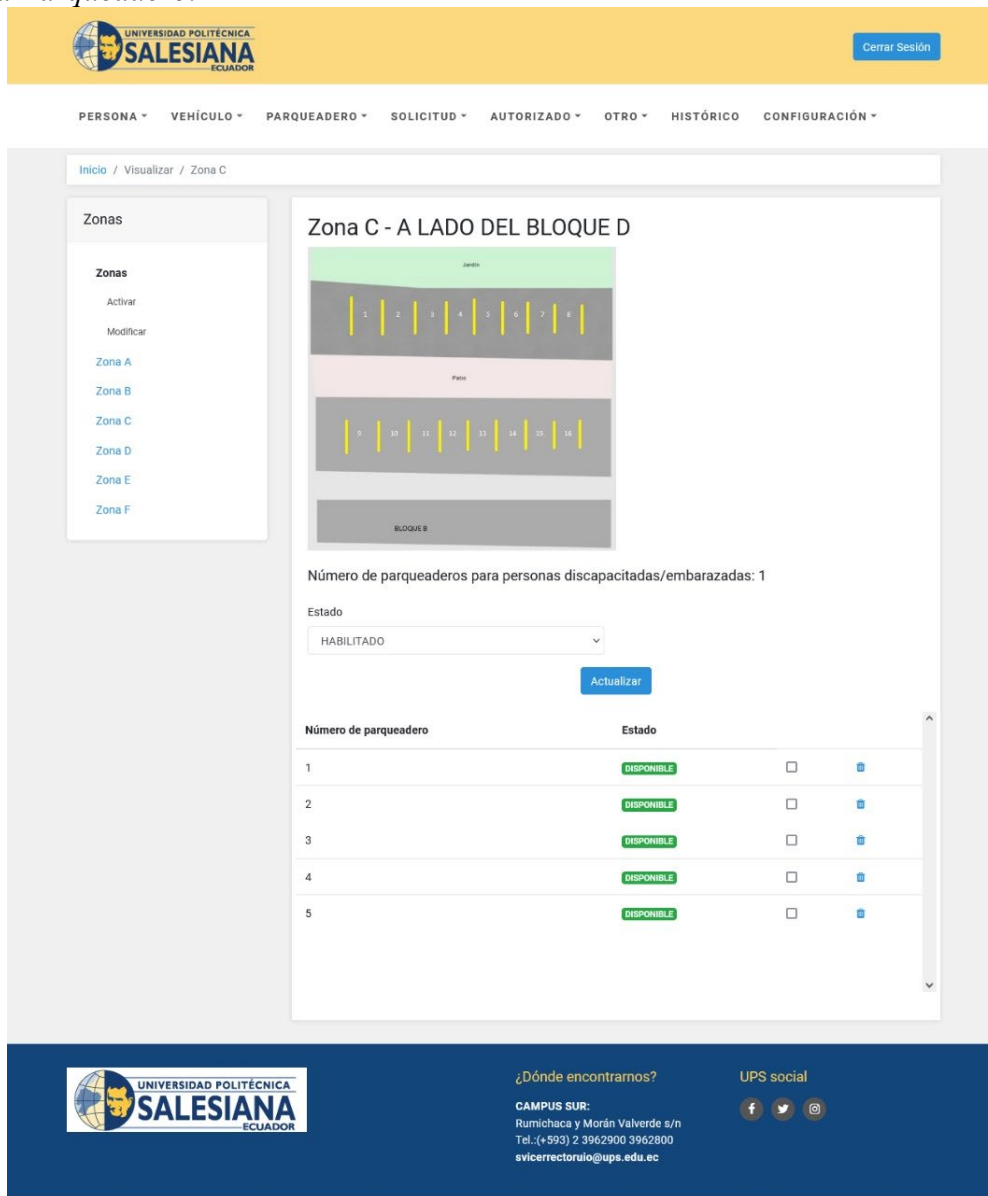
Figura 50
Pantalla Vehículo.



Nota. Pantalla de visualización de vehículo al momento de realizar la búsqueda por placa o cedula. Elaborado por: Tatiana Masapanta & Micaela Minga.

La figura 51 representa la pantalla para a modificación del parqueadero por parte del administrador.

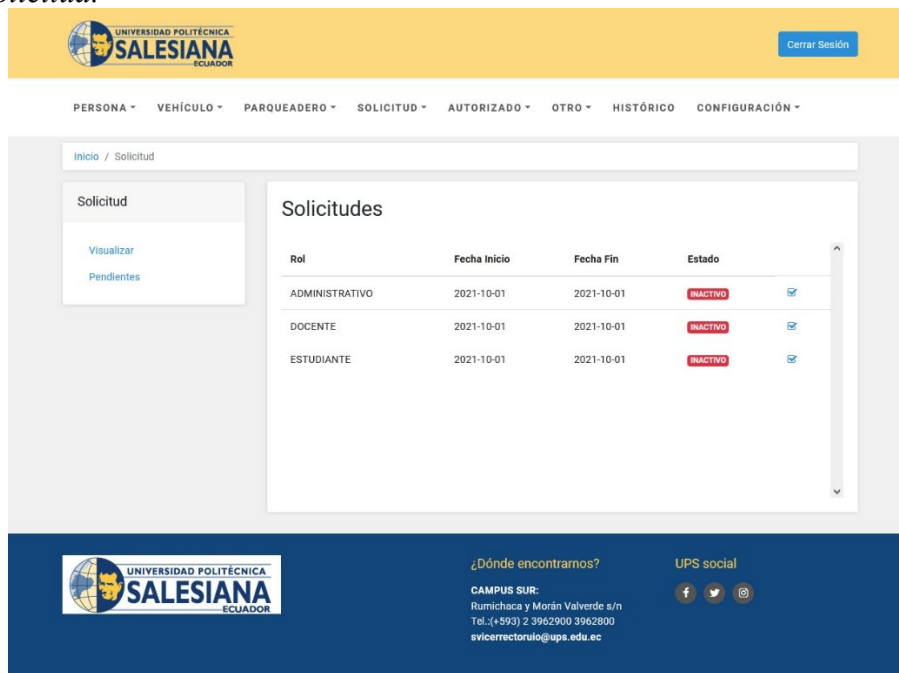
Figura 51
Pantalla Parqueadero.



Nota. Pantalla de parqueadero donde se puede verificar la disponibilidad del mismo. Elaborado por: Tatiana Masapanta & Micaela Minga.

La figura 52 representa la pantalla para la activación de solicitudes por parte del administrador.

Figura 52
Pantalla Solicitud.



Nota. Pantalla de apertura de solicitudes, con su respectivo estado. Elaborado por: Tatiana Masapanta & Micaela Minga.

La figura 53 representa la pantalla para cargar un archivo Excel de las personas autorizadas por parte del administrador.

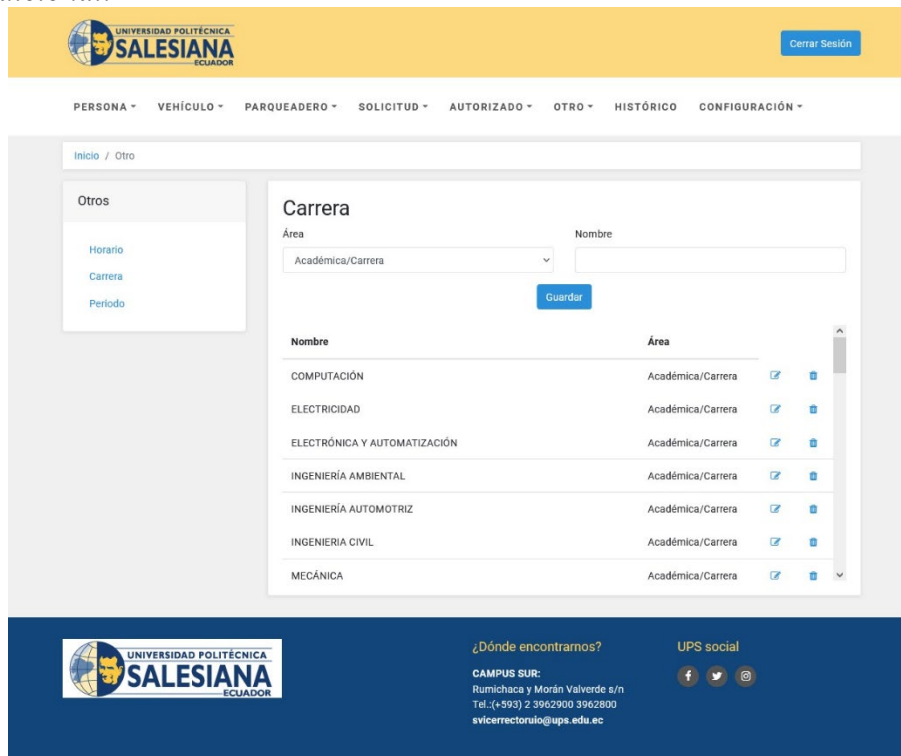
Figura 53
Pantalla Autorizado.



Nota. Pantalla donde se carga un archivo Excel de las personas autorizadas. Elaborado por: Tatiana Masapanta & Micaela Minga.

La figura 54 representa la pantalla para el ingreso de carreras por parte del administrador.

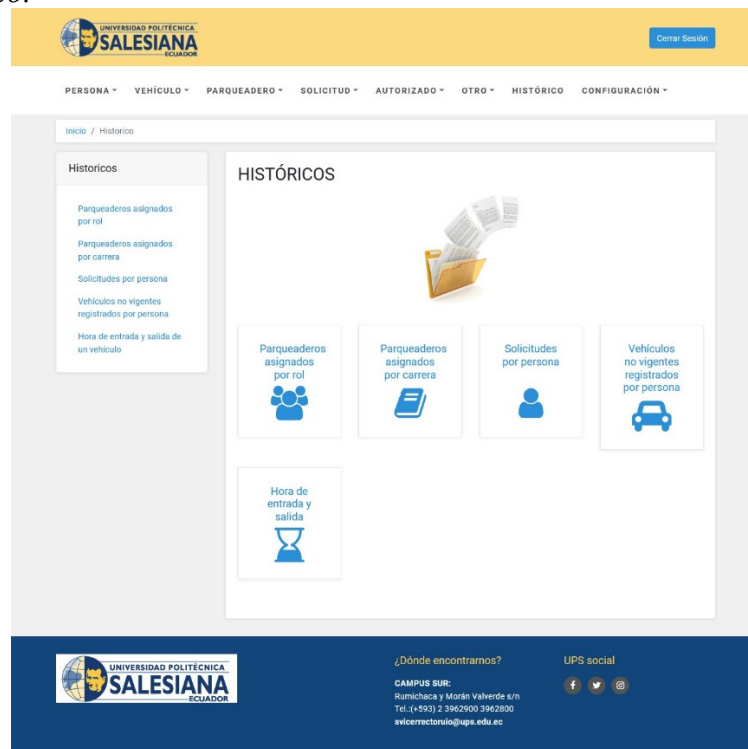
Figura 54
Pantalla Adicional.



Nota. Pantalla de visualización de las carreras existentes. Elaborado por: Tatiana Masapanta & Micaela Minga.

La figura 55 representa la pantalla para visualizar los datos históricos por parte del administrador.

Figura 55
Pantalla Histórico.



Nota. Pantalla de todos los históricos existentes para el sistema. Elaborado por: Tatiana Masapanta & Micaela Minga.

1.12.8.2. Administrativo, docente y estudiante

La figura 56 representa la pantalla para el envío de solicitudes por parte del administrativo, docente y estudiante.

Figura 56
Pantalla Envío de solicitudes.

The screenshot shows a web form titled 'Ingresar Información' for a student. The form is divided into two main sections: 'Ingresar Información' and 'Información del vehículo'. The 'Ingresar Información' section contains fields for 'Número de Cédula' (ID number), 'Rol' (Role), 'Nombre' (Name), 'Apellido' (Surname), 'Correo Institucional' (Institutional Email), 'Teléfono' (Phone), 'Carrera' (Career), 'Campus', 'Tiene alguna discapacidad' (Do you have any disability), and 'Horario' (Schedule). The 'Información del vehículo' section contains fields for 'Tipo' (Type), 'Placa' (License Plate), 'Marca' (Brand), 'Modelo' (Model), and 'Color' (Color). A 'Guardar' button is located at the bottom right of the form. The page header includes the Universidad Politécnica Salesiana logo and a 'Cerrar Sesión' button. The footer contains contact information for 'CAMPUS SUR' and social media links for 'UPS social'.

Nota. Pantalla para el envío de una solicitud donde el usuario podrá ingresar la información
Elaborado por: Tatiana Masapanta & Micaela Minga.

1.13. PRUEBAS

1.13.1. Estrés

En esta sección se realizó pruebas de estrés, utilizando la aplicación de Jmeter, para lo cual, se realizó la simulación con 50, 100, 500 y 1.000 usuarios en 1 segundo, esto se lo hizo para los diferentes roles seleccionados algunas páginas de la aplicación.

En la figura 57 se ve el resultado de las peticiones y el estado de algunas de ellas. El resultado obtenido con 50 usuarios fue un tiempo promedio de 1.663 milisegundos, con un porcentaje de error del 0%, como se observa en la figura 58.

Figura 57
Resultado en árbol 1

Muestra #	Tiempo de c...	Nombre del ...	Etiqueta	Tiempo de M...	Estado	Bytes	Sent Bytes	Latency	Connect Tim...
1	16:42:59.715	Grupo de Hil...	Ingresar Pers...	1401	✓	514	379	1396	5
2	16:42:59.676	Grupo de Hil...	Ingresar Pers...	1440	✓	514	379	1435	2
3	16:43:00.155	Grupo de Hil...	Ingresar Pers...	963	✓	514	379	963	2
4	16:42:59.556	Grupo de Hil...	Ingresar Pers...	1584	✓	514	379	1584	31
5	16:42:59.859	Grupo de Hil...	Ingresar Pers...	1290	✓	514	379	1290	3
6	16:42:59.484	Grupo de Hil...	Ingresar Pers...	1675	✓	514	379	1675	104
7	16:42:59.484	Grupo de Hil...	Ingresar Pers...	1679	✓	514	379	1679	92
8	16:42:59.839	Grupo de Hil...	Ingresar Pers...	1328	✓	514	379	1328	3
9	16:42:59.580	Grupo de Hil...	Ingresar Pers...	1589	✓	514	379	1589	8
10	16:42:59.484	Grupo de Hil...	Ingresar Pers...	1686	✓	514	379	1686	94
11	16:42:59.598	Grupo de Hil...	Ingresar Pers...	1584	✓	514	379	1584	2
12	16:42:59.736	Grupo de Hil...	Ingresar Pers...	1474	✓	514	379	1474	3
13	16:42:59.497	Grupo de Hil...	Ingresar Pers...	1775	✓	514	379	1775	82

Nota. Resultado de árbol por parte del administrador simulando el ingreso de 50 usuarios por 1 segundo. Elaborado por: Tatiana Masapanta & Micaela Minga.

Figura 58
Reporte resumen 1

Etiqueta	# Muestras	Media	Min	Máx	Desv. Estándar	% Error	Rendimiento	Kb/sec	Sent KB/sec	Media de By...
Ingresar Per...	50	1766	963	2706	383,09	0,00%	17,6/sec	8,85	6,52	514,0
Modificar P...	50	1332	1041	1639	137,95	0,00%	19,1/sec	14,24	7,09	763,0
Ver rol	50	4194	1484	5214	890,15	0,00%	8,1/sec	12069,57	1,81	1523399,4
Login	50	931	291	1521	304,51	0,00%	8,5/sec	4,17	1,89	500,0
Ingresar Veh...	50	993	288	1776	282,19	0,00%	9,5/sec	6,57	2,83	705,0
Ingresar Aut...	50	764	276	1498	286,73	0,00%	10,2/sec	5,23	3,03	526,0
Total	300	1663	276	5214	1260,44	0,00%	26,1/sec	6481,57	7,74	254401,2

Nota. Resultado del reporte por parte del administrador simulando el ingreso de 50 usuarios por 1 segundo. Elaborado por: Tatiana Masapanta & Micaela Minga.

A continuación de detallar en una tabla todos los resultados obtenidos en las pruebas realizadas en la herramienta Jmeter. En este apartado están los resultados obtenidos para los cuatro roles que son administrador, docente, estudiante, administrativo

Tabla 10*Pruebas Estrés Jmeter*

Rol	Número de usuarios	Tiempo de ejecución (milisegundos)	Error (%)
Administrador	100	2.067	0
Administrador	500	8.186	0
Administrador	1.000	7.830	42,78
Docente, estudiante, administrativo	50	756	0
Docente, estudiante, administrativo	100	3.703	0
Docente, estudiante, administrativo	500	6.282	0
Docente, estudiante, administrativo	1.000	7.435	40,35

Nota. Prueba de caja negra para el ingreso de personas. Elaborado por: Tatiana Masapanta & Micaela Minga.

1.13.2. Caja Negra

En la tabla 11 se puede observar el resultado de la prueba de la caja negra para el ingreso de personas detallado algunos parámetros como: datos de entrada, resultado esperado, resultado obtenido, código, entre otros.

Tabla 11*Prueba de caja negra 1*

Prueba: 1	Ingreso de persona
Propósito:	Ingresar la información de la persona que solicite el parqueadero.
Prerrequisito:	<ul style="list-style-type: none"> • Llenar todos los campos. • Llenar los campos acordes a los datos solicitados.
Datos de entrada:	<ul style="list-style-type: none"> • Ingreso de los datos de la persona.
Pasos:	<ul style="list-style-type: none"> • Llenar formulario. • Guardar
Resultado esperado:	<ul style="list-style-type: none"> • Si el ingreso de datos es correcto la información se guarda, caso contrario salta un mensaje de error.
Resultado obtenido:	<ul style="list-style-type: none"> • Se guardó con éxito.
Código:	<pre>\$a1 = array('num_cedula' => \$cedula); \$url = 'http://localhost/cambio2/api/persona/buscarP'; \$metodo = 'POST';</pre>

	<pre> \$resultado = conectar(\$url, \$a1, \$metodo); if (empty(\$resultado)) { \$a = array('nombres' => \$nombre, 'apellido' => \$apellido, 'correo' => \$email, 'num_cedula' => \$cedula, 'telefono' => \$telefono, 'id_rol' => \$rol, 'id_carrera' => \$carrera, 'discapacidad' => \$dis, 'campus' => \$campus); \$url = 'http://localhost/cambio2/api/persona/save'; \$metodo = 'POST'; \$resultado = conectar(\$url, \$a, \$metodo); \$url = 'http://localhost/cambio2/api/persona/buscarP'; \$metodo = 'POST'; \$resultado = conectar(\$url, \$a1, \$metodo); if (empty(\$resultado)) { \$this->session->set_flashdata('no'); redirect(base_url() . 'persona'); } else { \$this->session->set_flashdata('si'); redirect(base_url() . 'persona'); } } else { \$this->session->set_flashdata('error'); redirect(base_url() . 'persona'); } </pre>
--	---

Nota. Prueba de caja negra para el ingreso de personas. Elaborado por: Tatiana Masapanta & Micaela Minga.

En la tabla 12 se puede observar el resultado de la prueba de la caja negra para la activación de solicitud detallado algunos parámetros como: datos de entrada, resultado esperado, resultado obtenido, código, entre otros.

Tabla 12
Prueba de caja negra 2

Prueba: 2	Activación solicitud
Propósito:	Validar las fechas de apertura de la solicitud, de tal manera que, la fecha de cierre no sea menor a la fecha de apertura.
Prerrequisito:	<ul style="list-style-type: none"> • Llenar el formulario.
Datos de entrada:	<ul style="list-style-type: none"> • Ingreso de fecha • Ingreso de contraseña
Pasos:	<ul style="list-style-type: none"> • Llenar formulario. • Guardar
Resultado esperado:	<ul style="list-style-type: none"> • Si el ingreso de datos es correcto la información se guarda, caso contrario salta un mensaje de error.
Resultado obtenido:	<ul style="list-style-type: none"> • Usted activo la página para recibir solicitudes de ADMINISTRATIVO para el periodo 59 (2021 - 2021) <p>Fecha inicio: 2021-11-15</p> <p>Fecha fin: 2021-11-18</p>
Código:	<pre>\$(document).ready(function (e) { \$(function () { \$("#limite").change(function () { \$("#message").empty(); var fecha inicial = document.getElementById("inicio").value; var fecha final = document.getElementById("limite").value; if (Date.parse(fechar final) < Date.parse(fechar inicial)) { \$("#results").attr('src', 'noimage.png'); \$("#message").html("<div class='alert alert- danger'>Por favor seleccione una fecha mayor a la de inicio</div>"); \$("#limite") document.getElementById("limite").value = ""; } }); }); });</pre>

Nota. Prueba de caja negra para validación de fechas. Elaborado por: Tatiana Masapanta & Micaela Minga.

En la tabla 13 se puede observar el resultado de la prueba de la caja negra para el ingreso de autorizado detallado algunos parámetros como: datos de entrada, resultado esperado, resultado obtenido, código, entre otros.

Tabla 13
Prueba caja negra 3

Prueba: 3	Ingreso Autorizado
Propósito:	Ingresar una persona autorizada mediante la búsqueda de cédula.
Prerrequisito:	<ul style="list-style-type: none"> • Llenar el formulario.
Datos de entrada:	<ul style="list-style-type: none"> • Ingreso de cédula.
Pasos:	<ul style="list-style-type: none"> • Llenar el campo de cédula. • Buscar
Resultado esperado:	<ul style="list-style-type: none"> • Si la cédula existe y no tiene una autorización activa le re direccionará para llenar el formulario, por otro lado, si la persona existe y tiene una autorización mandara un mensaje.
Resultado obtenido:	<ul style="list-style-type: none"> • Redireccionamiento a la página para llenar el formulario.
Código:	<pre> \$a = array('id' => \$cedula); \$url = 'http://localhost/cambio2/api/autorizado/buscar'; \$metodo = 'POST'; \$resultado = conectar(\$url, \$a, \$metodo); \$url = 'http://localhost/cambio2/api/autorizado/buscarAutorizado'; \$metodo = 'POST'; \$resultado1 = conectar(\$url, \$a, \$metodo); if (empty(\$resultado['data'])) { \$this->session->set_flashdata('buscar_persona'); redirect(base_url() . 'autorizacion'); } else if (!empty(\$resultado1['data'])) { \$this->session->set_flashdata('buscar_authorized'); redirect(base_url() . 'autorizacion'); } else if (!empty(\$resultado1['data1'])) { \$this->session->set_flashdata('buscar_authorizedPendiente'); redirect(base_url() . 'autorizacion'); } else { \$this->load->view('paginas/Header/header'); \$this->load->view('paginas/Autorizacion/autorizacion', \$resultado); \$this->load->view('paginas/Header/footer'); </pre>

	}
--	---

Nota. Prueba de caja negra para ingresar una persona autorizada. Elaborado por: Tatiana Masapanta & Micaela Minga.

En la tabla 14 se puede observar el resultado de la prueba de la caja negra para la carga del Excel detallado algunos parámetros como: datos de entrada, resultado esperado, resultado obtenido, código, entre otros.

Tabla 14
Prueba caja negra 4

Prueba: 4	Carga del Excel
Propósito:	Cargar un documento de Excel dependiendo el rol.
Prerrequisito:	<ul style="list-style-type: none"> Tener el Excel.
Datos de entrada:	<ul style="list-style-type: none"> Excel.
Pasos:	<ul style="list-style-type: none"> Seleccionar el Excel a subir Cargar
Resultado esperado:	<ul style="list-style-type: none"> Si el archivo tiene al rol que pertenece se procede a cargar el archivo, caso contrario se mostrara un mensaje de error.
Resultado obtenido:	<ul style="list-style-type: none"> La información se ha cargado.
Código:	<pre> \$nombre = \$_FILES["file"]["name"]; \$info = new SplFileInfo(\$nombre); \$s = \$info->getExtension(); if (!empty(\$nombre)) { if ((strcmp(\$s, 'xls') == 0) (strcmp(\$s, 'xlsx') == 0)) { if (strncasecmp(\$nombre, 'Docente', 7) == 0) { \$a = array(); \$url = 'http://localhost/cambio2/api/autorizado/updateD'; \$metodo = 'POST'; \$data = conectar(\$url, \$a, \$metodo); \$url = 'http://localhost/cambio2/api/autorizado/buscarPeriodoActual'; \$periodo = conectar(\$url, \$a, \$metodo); \$id_periodo = \$periodo[0]['numero'] . ' (' . \$periodo[0]['periodo'] . ')'; if (!empty(\$_FILES["file"]["name"])) { \$path = \$_FILES["file"]["tmp_name"]; } } } </pre>

```

        $subject = PHPExcel_IOFactory::load($path);
        foreach ($subject->getWorksheetIterator() as
$worksheet) {
            $highestRow = $worksheet->getHighestRow();
            $highestColumn = $worksheet-
>getHighestColumn();
            for ($row = 2; $row <= $highestRow; $row++) {
                $cedula = $worksheet-
>getCellByColumnAndRow(0, $row)->getValue();
                $nombre = $worksheet-
>getCellByColumnAndRow(1, $row)->getValue();
                $apellido = $worksheet-
>getCellByColumnAndRow(2, $row)->getValue();
                $correo = $worksheet-
>getCellByColumnAndRow(3, $row)->getValue();
                $placa = $worksheet-
>getCellByColumnAndRow(4, $row)->getValue();
                $campus = $worksheet-
>getCellByColumnAndRow(5, $row)->getValue();
                $cod_sticker = $worksheet-
>getCellByColumnAndRow(6, $row)->getValue();
                $a = array(
                    'num_cedula' => strtoupper($cedula),
                    'nombres' => strtoupper($nombre),
                    'apellido' => strtoupper($apellido),
                    'placa' => strtoupper($placa),
                    'cod_sticker' => strtoupper($cod_sticker),
                    'campus' => $campus
                );

                $url =
'http://localhost/cambio2/api/autorizado/excelD';

                $metodo = 'POST';
                $data = conectar($url, $a, $metodo);
                $url =
'http://localhost/cambio2/api/autorizado/buscarIngresoAutorizado';
                $data = conectar($url, $a, $metodo);

                if (!empty($data)) {
                    enviar_correo_autorizado('2', $correo,
$id_periodo, $placa, "");
                }
            }
        }
        echo 'La información se ha cargado';
    }
} else {
    echo 'El archivo tiene un nombre diferente al solicitado
';
}

```


	<pre> } } else { echo 'Solo puede cargar archivos de Excel'; } } else { echo 'Seleccione un archivo'; } </pre>
--	--

Nota. Prueba de caja negra para importar el Excel. Elaborado por: Tatiana Masapanta & Micaela Minga

En la tabla 15 se puede observar el resultado de la prueba de la caja negra para la solicitud de estudiante nuevo o antiguo detallado algunos parámetros como: datos de entrada, resultado esperado, resultado obtenido, código, entre otros.

Tabla 15

Prueba de caja negra 5

Prueba: 5	Solicitud de estudiante nuevo o antiguo
Propósito:	Buscar a un estudiante y dependiendo si es nuevo o antiguo se le indique los campos a llenar.
Prerrequisito:	<ul style="list-style-type: none"> • Llenar el formulario, dependiendo del estudiante.
Datos de entrada:	<ul style="list-style-type: none"> • Ingreso de cédula.
Pasos:	<ul style="list-style-type: none"> • Llenar el campo de cédula. • Buscar.
Resultado esperado:	<ul style="list-style-type: none"> • Si el estudiante es nuevo ingresar toda la información, es decir de la persona y vehículo, caso contrario llenar la información del vehículo.
Resultado obtenido:	<ul style="list-style-type: none"> • Redireccionamiento a la página para llenar el formulario.
Código:	<pre> \$a = array('id' => \$cedula); //persona cedula \$url = 'http://localhost/cambio2/api/estudiante/buscar'; \$metodo = 'POST'; \$resultado = conectar(\$url, \$a, \$metodo); //persona periodo \$url = 'http://localhost/cambio2/api/estudiante/buscarp'; \$resultado3 = conectar(\$url, \$a, \$metodo); \$a = array(); //carrera \$url = 'http://localhost/cambio2/api/estudiante/carrera'; \$metodo = 'GET'; </pre>

	<pre> \$resultado2 = conectar(\$url, \$a, \$metodo); if (empty(\$resultado)) { if (\$this->session->datos == 'prueba') { \$this->session->set_userdata("mensaje", 'prueba'); \$data = array('id' => \$cedula); \$data4 = array('data4' => \$data); \$resultado = array_merge(\$data4, \$resultado2); \$this->load->view('paginas/Header/header_est'); \$this->load->view('paginas/Estudiante/nuevo', \$resultado); \$this->load->view('paginas/Header/footer'); } else { redirect(base_url() . 'repcionE'); } } else { if (\$resultado[0]['id_rol'] == 3) { if (empty(\$resultado3)) { if (\$this->session->datos == 'prueba') { \$this->session->set_userdata("mensaje", 'prueba'); \$data = array('data' => \$resultado); \$resultado = array_merge(\$resultado2, \$data); \$this->load- >view('paginas/Header/header_est'); \$this->load->view('paginas/Estudiante/antiguo', \$resultado); \$this->load->view('paginas/Header/footer'); } else { redirect(base_url() . 'repcionE'); } } else { \$data = array('data' => \$resultado); \$this->load->view('paginas/Header/header_est'); \$this->load->view('paginas/Estudiante/mensaje2', \$resultado); \$this->load->view('paginas/Header/footer'); } } else { \$this->load->view('paginas/Header/header_est'); \$this->load->view('paginas/Estudiante/mensaje3'); \$this->load->view('paginas/Header/footer'); } } </pre>
--	---

	}
--	---

Nota. Prueba de caja negra para llenar la solicitud dependiendo si el estudiante es nuevo o antiguo. Elaborado por: Tatiana Masapanta & Micaela Minga

1.13.3. Usabilidad

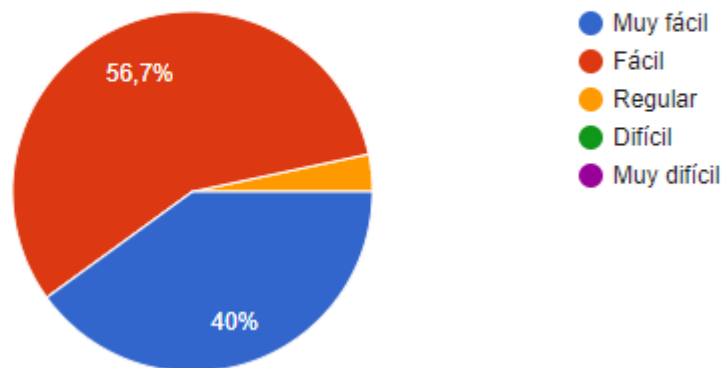
En este apartado se procede a realizar una encuesta que consta de 10 preguntas, las cuales ayudará ver el punto de vista de los usuarios con el sistema, la encuesta se realizará a 30 personas.

Figura 59

Usar el sistema le resultó

Usar el sistema le resultó

30 respuestas



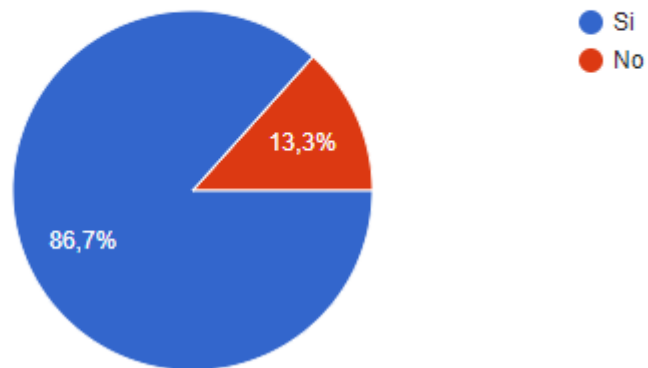
Nota. El color rojo representa que a 17 usuarios les pareció fácil de usar el sistema, a 12 les parece muy fácil y a 1 persona le parece regular usar el sistema, Elaborado por: Tatiana Masapanta & Micaela Minga.

Figura 60

¿Le resultó amigable la interfaz?

¿Le resultó amigable la interfaz?

30 respuestas



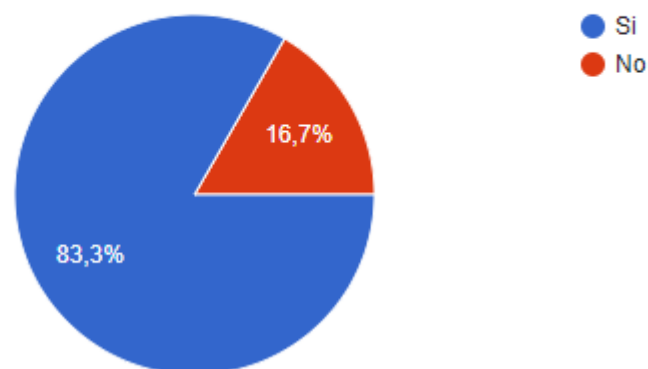
Nota. El color azul representa que a 26 usuarios les resulto amigable la interfaz y a 4 usuarios no les pareció amigable. Elaborado por: Tatiana Masapanta & Micaela Minga.

Figura 61

¿Puede completar el trabajo usando el sistema?

¿Puede completar el trabajo usando el sistema?

30 respuestas



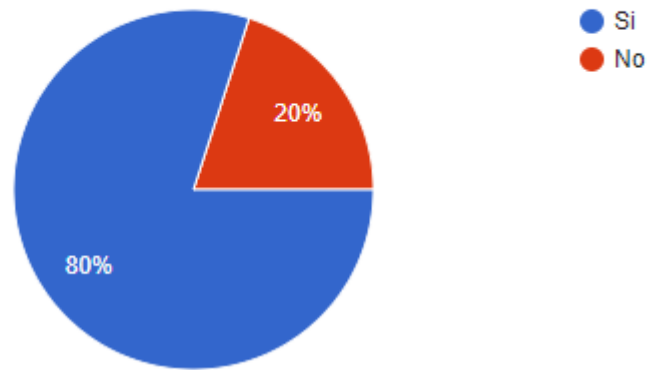
Nota. El color azul representa que 25 usuarios lograron completar el trabajo usando el sistema, mientras que 5 usuarios no lograron completar el trabajo. Elaborado por: Tatiana Masapanta & Micaela Minga.

Figura 592

¿Puede completar el trabajo rápidamente usando el sistema?

¿Puede completar el trabajo rápidamente usando el sistema?

30 respuestas



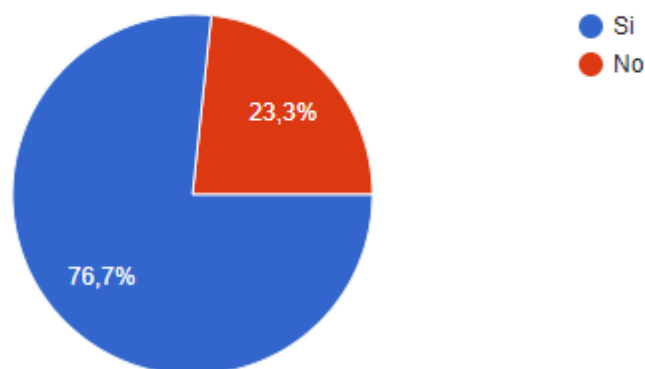
Nota. El color azul representa que 24 usuarios pudieron completar de forma rápida el trabajo, mientras que 6 personas no lograron realizar la tarea de forma rápida. Elaborado por: Tatiana Masapanta & Micaela Minga.

Figura 63

¿La información proporcionada es clara?

¿La información proporcionada es clara?

30 respuestas



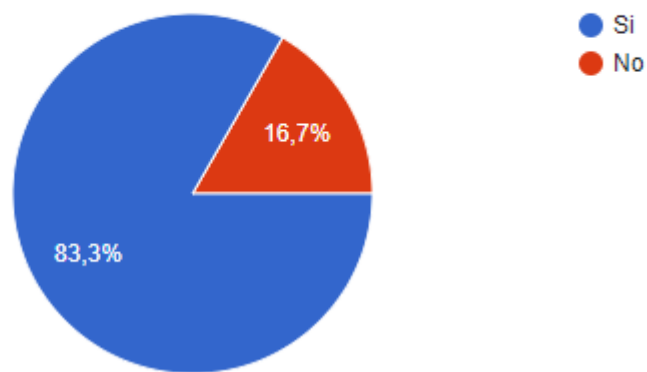
Nota. El color azul representa que 23 usuarios les resulto que el sistema cuenta co información clara, mientras que, a 7 usuarios no le resulto clara la información. Elaborado por: Tatiana Masapanta & Micaela Minga.

Figura 64

¿Considera que gráficamente el sitio está equilibrado?

¿Considera que gráficamente el sitio está equilibrado?

30 respuestas



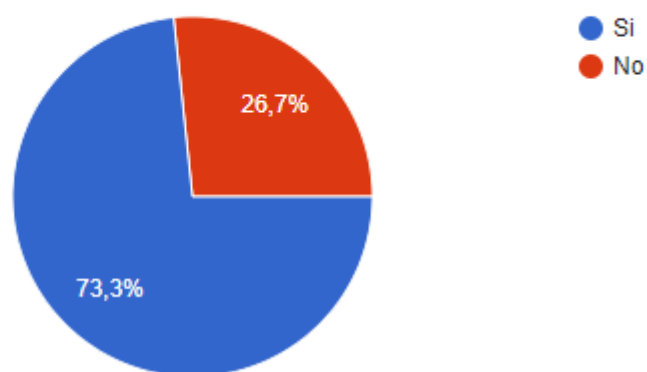
Nota. El color azul representa que 25 usuarios consideran que el sistema cuenta con las suficientes imágenes, mientras que, 5 usuarios no consideran que el sistema cuenta con un equilibrio de imágenes. Elaborado por: Tatiana Masapanta & Micaela Minga.

Figura 65

¿Cree que los iconos son fáciles de entender?

¿Cree que los iconos son fácil de entender?

30 respuestas



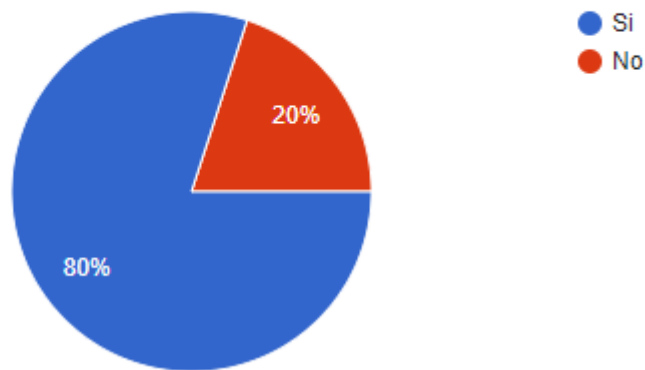
Nota. El color azul representa que 22 usuarios consideran que el sistema cuenta con iconos entendibles, mientras que, 8 usuarios no consideran que el sistema tenga iconos fáciles de entender. Elaborado por: Tatiana Masapanta & Micaela Minga.

Figura 66

Le fue fácil encontrar la ventana para la tarea a realizar

Le fue fácil encontrar la venta para la tarea a realizar

30 respuestas



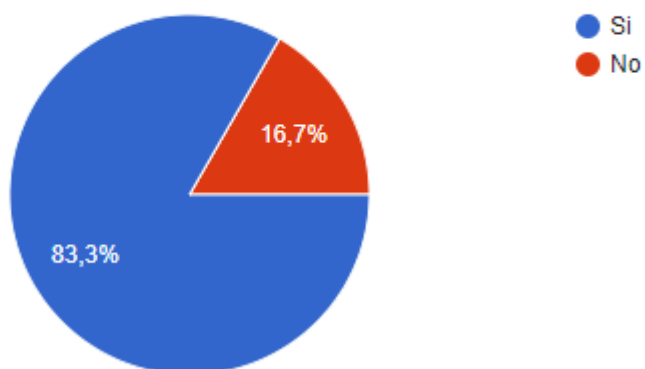
Nota. El color azul representa que 24 usuarios encontraron rápidamente la ventana para realizar la tarea, mientras que, 6 usuarios no consideran lo mismo. Elaborado por: Tatiana Masapanta & Micaela Minga.

Figura 67

Los mensajes de alerta le parecieron claros

Los mensajes de alerta le parecieron claros

30 respuestas



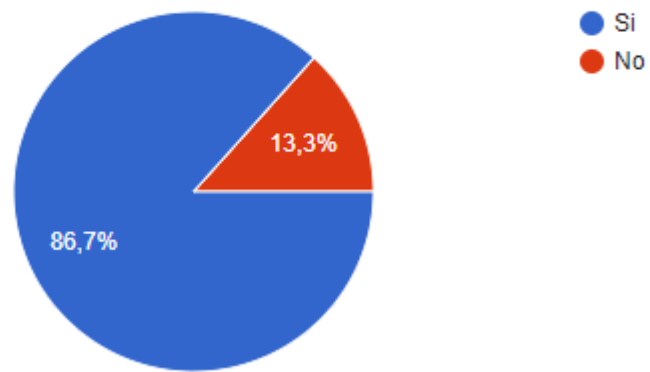
Nota. El color azul representa que 25 usuarios consideran que el sistema cuenta mensajes de alerta entendibles, mientras que, 5 usuarios no consideran lo mismo. Elaborado por: Tatiana Masapanta & Micaela Minga.

Figura 68

Utilizaría frecuentemente el sistema

Utilizaría frecuentemente el sistema

30 respuestas



Nota. El color azul representa que 26 usuarios volverían a usar el sistema, mientras que, 4 usuarios no volverías a usar el sistema. Elaborado por: Tatiana Masapanta & Micaela Minga.

CONCLUSIONES

El sistema permite tener el control de todas las personas que han realizado el proceso de solicitud o las que han sido ingresadas al sistema por parte de los administradores, para poder tener acceso a una zona de parqueo una vez que hayan sido autorizados a usarlo, esto permitirá tener un mejor control del parqueadero UPS-CS en el periodo que se encuentre en ese tiempo.

El sistema cuenta con un servicio, el cual, permite tener el control de una zona de parqueo de la UPS-CS, haciendo referencia a una zona en específico o un espacio de parqueadero individual, ayudando de esa forma a que las personas encargadas del proceso puedan tener acceso a la administración del parqueadero de una manera más rápida.

La base de datos es una parte fundamental a la cual otros aplicativos pueden conectarse para poder utilizar la información almacenada o realizar cambios en algunas tablas, para evitar que estos aplicativos realizan cambios directamente en la base de datos no se les permite conectarse directamente a la misma, por lo tanto, se optó por implementar un web service que permita acceder a los datos de la base por medio de un intermediario.

Las pruebas que se realizó arrojaron que el sistema soporta varias peticiones por parte de los usuarios, sin que este llegara a colapsar, por otro lado, para gran parte de usuarios el sistema tiene una interfaz amigable e intuitiva, donde se puede realizar todo el proceso de solicitud de parqueadero de una manera rápida y exitosa.

El sistema permite tener un mejor control de todas las personas que han realizado el proceso de solicitud o las que han sido ingresadas al sistema por parte de los administradores, ya que la información de las mismas quedará almacenada en la base de datos del sistema y de esa manera podrán ser consultadas cuando se lo requiera.

RECOMENDACIONES

Para que los docentes, estudiantes, administrativos y el administrador puede tener acceso al sistema fuera de la UPS-CS, se recomienda subir el sistema a un servidor teniendo en cuenta todos los protocolos de seguridad que requiere un software y de esa forma las personas mediante un link puedan tener acceso al mismo, desde cualquier parte del mundo.

Se recomienda la implementación de cámaras en los parqueaderos de la UPS-CS, de tal forma que, se pueda obtener el cambio en tiempo real de los estados de todas las zonas de parqueo y el resultado podrá ser visualizado en el sistema en la sección de parqueaderos.

Se recomienda ejecutar el sistema de solicitudes en el navegador Firefox ya que el sistema cuenta con alertas o mensajes que pueden ser visualizadas de mejor manera dentro del mismo, considerando que existe navegadores que pueden omitir las alertas debido a la seguridad y bloqueos que tienen.

En caso de que existan nuevos requerimientos, se recomienda sacar una copia de seguridad tanto del código del aplicativo web como de la base de datos, ya que, si se llega a presentar algún tipo de problema puedan regresar a una versión anterior.

GLOSARIO

JSON: JavaScript Object Notation.

UPS-CS: Universidad Politécnica Salesiana Campus Sur.

XP: Xtreme Programming.

API: Application Programming Interfaces

LISTA DE REFERENCIAS

Artículos Académicos

- Ávalos-Silva, H., Gómez, E., Ordóñez-Camacho, D., & Taipe, O. (2018). Implementation of a system for the administration, configuration and monitoring of parking areas. *International Conference on Information Systems and Computer Science (INCISCOS)*, 356-360. doi:10.1109/INCISCOS.2018.00058
- Driss, M., Aljehani, A., Boulila, W., Ghandorh, H., & Al-Sarem, M. (2020). Servicing Your Requirements: An FCA and. *IEEE Access*, 8, 59326-59339. doi:10.1109/ACCESS.2020.2982592
- Expósito, E. (2008). Metodologías de desarrollo de software. ¿Cuál es el camino? *Revista de Arquitectura e Ingeniería*, 2(3). Retrieved 02 20, 2021, from <https://www.redalyc.org/pdf/1939/193915935003.pdf>
- Mirabella, G., Martin-Lopez, A., Segura, S., Valencia-Cabrera, L., & Ruiz-Cortés, A. (s.f.). *Inferencia Automática de Dependencias*. Retrieved 08 26, 2021, from <https://personal.us.es/amarlop/wp-content/uploads/2021/10/Inferencia-Automatica-de-Dependencias-Inter-Parametro-en-APIs-REST.pdf>
- Wang, H., & He, W. (2011). A Reservation-based Smart Parking System. *IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPs)*, 690-695. doi:10.1109/INFCOMW.2011.5928901

Libros

- Ambler, S. (2002). *Agile Modeling: Effective Practices for eXtreme Programming and the Unified Process*. John Wiley & Sons. Retrieved 09 11, 2021, from https://books.google.es/books?hl=es&lr=&id=uh_jSk2FSa0C&oi=fnd&pg=PR5&dq=

Extreme+programming+&ots=sU6-

CH8upk&sig=nwG8awafLJUwSppaUl6IJuViEt8#v=onepage&q=xp&f=false

Boudreau, T., Glick, J., Greene, S., Spurlin, V., & Woehr, J. (2002). *NetBeans: the definitive guide: developing, , and deploying Java code*. O'Reilly Media, Inc. Retrieved from https://books.google.com.ec/books?id=9I5lr0uE4GAC&printsec=frontcover&dq=NetBeans:+the+definitive+guide:+developing,+debugging,+and+deploying+Java+code&hl=es&sa=X&redir_esc=y#v=onepage&q=NetBeans%3A%20the%20definitive%20guide%3A%20developing%2C%20debuggin

Carrión, R., Noriega, A., & Del Castillo, D. (2019). *Usando XAMPP con Bootstrap y WordPress*. RamAstur. Retrieved from https://books.google.com.ec/books?id=pP-uDwAAQBAJ&printsec=frontcover&hl=es&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false

Cobo, Á., Gómez, P., Pérez, D., & Rocha, R. (2005). *PHP y MySQL Tecnologías para el desarrollo de aplicaciones web*. Ediciones Díaz de Santos. Retrieved from https://books.google.es/books?hl=es&lr=&id=zMK3GOMOpQ4C&oi=fnd&pg=PR17&dq=definicion+servidor+web&ots=Fhhr_ZAiun&sig=F3QVfw4crvebbQdwXcA7jUeYpWU#v=onepage&q=definicion%20servidor%20web&f=true

Fernández, J. (2013). *Introducción a las metodologías ágiles Otras formas de analizar y desarrollar*. Universitat Oberta de Catalunya. Retrieved 08 26, 2021, from [https://www.exabyteinformatica.com/uoc/Informatica/Tecnicas_avanzadas_de_ingenieria_de_software/Tecnicas_avanzadas_de_ingenieria_de_software_\(Modulo_3\).pdf](https://www.exabyteinformatica.com/uoc/Informatica/Tecnicas_avanzadas_de_ingenieria_de_software/Tecnicas_avanzadas_de_ingenieria_de_software_(Modulo_3).pdf)

Hernández, C. (2018). *Patrón arquitectónico MVC (Modelo Vista Controlador)*. Repositorio Digital Konrad Lorenz. Retrieved 09 08, 2021, from <http://repositorio.konradlorenz.edu.co/handle/001/138>

Pérez, M., Mendoza, L., & Grimán, A. (2005). Modelo para estimación de la calidad. *Academia*, 989-1000. Retrieved 09 23, 2021, from https://d1wqtxts1xzle7.cloudfront.net/40656517/calidad_43-with-cover-page-v2.pdf?Expires=1635381913&Signature=NwQD52f0QpK4FwmWAonm3W2NOxTaHwU6AfLELxb30n8irwlmSSpkka4R5~Z8QAFoe6ORFa9V21AwpB3lyfFah1d6sxLF9qdNSLWwR8N6Gm3RBYvOhFnE1twV94DOw2jYk8zRhjs5-jZvGzzKC

Ramos, A., & Ramos, J. (2014). *Aplicaciones Web*. Ediciones Paraninfo,SA. Retrieved from https://books.google.es/books?hl=es&lr=&id=43G6AwAAQBAJ&oi=fnd&pg=PA1&dq=servidor+web+definiciones&ots=Dh57mZq5EN&sig=uluf_RIg2Kh_4Ioi5d3oBR LjAZ0#v=onepage&q=servidor%20web%20definiciones&f=true

Sinnaps. (s.f.). *METODOLOGÍA XP O PROGRAMACIÓN EXTREMA*. Retrieved 09 14, 2021, from Sinnaps: <https://www.sinnaps.com/blog-gestion-proyectos/metodologia-xp>

Spona, H. (2010). *Programacion de base de datos con MySQL y PHP*. Marcombo. Retrieved from https://books.google.es/books?hl=es&lr=&id=y11L7pQfdRsC&oi=fnd&pg=PA1&dq=php+&ots=5zdXmCHSPR&sig=ahEiIgWPVkoh_uVKeM2DHtYohGs#v=onepage&q=php&f=true

Páginas Web

Apache Netbeans. (s.f.). *Apache Netbeans 12.5*. Retrieved 09 18, 2021, from Apache Netbeans: <https://netbeans.apache.org/>

Ávila, C. (2019). *Modelo Vista Controlador*. Repositorio Digital Konrad Lorenz. Retrieved from <https://repositorio.konradlorenz.edu.co/handle/001/1528>

CodeIgniter. (s.f.). *Welcome to CodeIgniter*. Retrieved 08 27, 2021, from CodeIgniter: <https://codeigniter.com/userguide3/general/welcome.html>

Orientación. (2020, 08 13). *¿Qué es XP y cómo usarlo en el desarrollo de un proyecto?*

Retrieved 07 26, 2021, from Orientación:

<https://orientacion.universia.edu.pe/infodetail/orientacion/consejos/que-es-xp-y-como-usarlo-en-el-desarrollo-de-un-proyecto-6157.html>

RedHat. (08 de mayo de 2020). *¿Qué es una API de REST?* Recuperado el 02 de 09 de 2021,

de RedHat: <https://www.redhat.com/es/topics/api/what-is-a-rest-api>

phpMyAdmin. (s.f.). *About*. Retrieved 08 26, 2021, from phpMyAdmin Bringing MySQL to

the web: <https://www.phpmyadmin.net/>

Postman. (s.f.). *What is Postman*. Retrieved 09 21, 2021, from Postman:

<https://www.postman.com/product/what-is-postman/>

Universidad Politécnica Salesiana. (s.f.). *Reseña Histórica*. Retrieved 07 12, 2021, from

<https://www.ups.edu.ec/resena-historica>

Tesis

Alonso-Aranda, C. (2019). *MODELO-VISTA-CONTROLADOR. LENGUAJE UML*.

Universidad de Jaén. Jaén: Universidad de Jaén. Retrieved 09 17, 2021, from

<https://hdl.handle.net/10953.1/11437>