UNIVERSIDAD TÉCNICA DE AMBATO

FACULTAD DE INGENIERÍA EN SISTEMAS ELECTRÓNICA E INDUSTRIAL

CARRERA DE INGENIERÍA EN SISTEMAS COMPUTACIONALES E INFORMÁTICOS

Tema:

Aplicación Web para la distribución de espacios disponibles de parqueo en la Universidad Técnica de Ambato campus Huachi Chico.

Trabajo de Graduación Modalidad: TEMI, Trabajo Estructurado de Manera Independiente, presentado previo la obtención del título de Ingeniero en Sistemas Computacionales e Informáticos

Sublinea de investigación: Aplicación Web.

AUTOR: Wilson German Pérez Nata
TUTOR: Ing. Mg. Franklin Mayorga

Ambato - Ecuador
Noviembre/2014

APROBACIÓN DEL TUTOR

En mi calidad de tutora del trabajo de investigación sobre el tema " **Aplicación Web para la distribución de espacios disponibles de parqueo en la Universidad Técnica de Ambato campus Huachi Chico** ", del señor Wilson German Pérez Nata, estudiante de la Carrera de Ingeniería en Sistemas Computacionales e Informáticos de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial, de la Universidad Técnica de Ambato, considero que el informe investigativo reúne los requisitos suficientes para que continúe con los trámites y consiguiente aprobación de conformidad con el art. 16 del Capítulo II, del reglamento de Graduación para obtener el título terminal de tercer nivel de la Universidad Técnica de Ambato.

Ambato, Noviembre 7 de 2014

TUTOR

Ing. Franklin Mayorga M., Mg.

AUTORÍA

El presente trabajo de investigación titulado: "**Aplicación Web para la distribución de espacios disponibles de parqueo en la Universidad Técnica de Ambato campus Huachi Chico**", es absolutamente original, auténtico y personal, en tal virtud, el contenido, efectos legales y académicos que se desprenden del mismo son de exclusiva responsabilidad del autor.

Ambato, Noviembre 7 de 2014

Sr. Wilson German Pérez Nata

CC: 180303994621

APROBACIÓN DE LA COMISIÓN CALIFICADORA

La Comisión Calificadora del presente trabajo conformada por los señores docentes Ing. Vicente Morales L. e Ing. Carlos Núñez M., revisó y aprobó el Informe Final del trabajo de graduación titulado "**Aplicación Web para la distribución de espacios disponibles de parqueo en la Universidad Técnica de Ambato campus Huachi Chico**", presentado por el señor Wilson German Pérez Nata de acuerdo al Art. 17 del Reglamento de Graduación para obtener el título Terminal de tercer nivel de la Universidad Técnica de Ambato.

Ing. Vicente Morales L., Mg.

PRESIDENTE DEL TRIBUNAL

Ing. Vicente Morales L., Mg.

DOCENTE CALIFICADOR

Ing. Carlos Núñez M., Mg.

DOCENTE CALIFICADOR

DEDICATORIA

Este trabajo se lo dedico, a Dios por haberme dado la constancia y la perseverancia para terminar esta carrera profesional.

A mi madre María Nata por su apoyo incondicional durante esta etapa académica y en las decisiones tomadas en el transcurso de mi vida, por darme ejemplo de superación y lucha constante, y sobre todo por el amor madre

A mi hermano Medardo Pérez por su apoyo durante esta etapa académica y enseñarme que no hay limitaciones cuando uno quiere alcanzar sus metas, gracias hermano porque siempre estuviste ahí aun cuando no era tu obligación.

A mi padre +Humberto Pérez que en paz descanse, por enseñarme a que el esfuerzo y la dedicación es lo que necesitaba para alcanzar mis metas y objetivos;

También a Nelly, Luis, Mario, Abigail Pérez hermanos que siempre me brindaron el apoyo cuando necesite de ellos este logro es el esfuerzo de todos gracias por todo.

A los que nunca dudaron que lo lograría, este es triunfo de ustedes.

Wilson German Pérez Nata

AGRADECIMIENTO

Doy gracias a Dios por siempre haber sido mi guía y mi amigo a lo largo de todos mis estudios, y poder culminar esta etapa de formación profesional en mi vida.

A mi familia por haber confiado en mí en todo momento, quienes con el transcurso de los años se han convertido en mi guía y mi refugio.

A mis amigos por su apoyo y amistad en todos estos años de estudio y de esfuerzo.

También quiero expresar un sincero agradecimiento a la Facultad de Ingeniería en Sistemas, Electrónica e Industrial de la Universidad Técnica de Ambato, por haberme dado la oportunidad de formarme profesionalmente con los saberes brindados a mi persona.

Wilson German Pérez Nata

PÁGINAS PRELIMINARES

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

ÍNDICE DE FOTOS

RESUMEN EJECUTIVO

La Universidad Técnica de Ambato fue creada el 14 de abril de 1969 según aprobación del Congreso Nacional. Nació con el lema "Educarse es aprender a ser libres" bajo el pensamiento y la égida del Doctor Carlos Toro Navas quien presidió la conformación del Primer Consejo Universitario, luego de realizada la primera Asamblea Universitaria un 10 de mayo de 1969. Vicerrector fue designado el economista Víctor Cabrera Guzmán.

Esta Organización Gubernamental, se encarga de forjar profesionales de éxito en las diferentes ramas competitivas, a través de sus distintas áreas académicas comprometidas con el desarrollo la provincia y país.

La institución no cuenta con un método de distribución de espacios disponibles de parqueadero, por lo cual se propone el desarrollo de una aplicación web que ayudara a dar un mejor servicio al usuario.

La aplicación web permitirá al usuario que utiliza un medio de trasporte, al ingresar a la institución tenga información sobre un lugar de libre donde puede estacionar su vehículo, mediante indicadores electrónicos utilizados para la obtención y presentación de la ubicación de sitios de estacionamientos.

Considerando, el diseño se presenta una aplicación web intuitiva para el usuario mediante la utilización herramienta informáticas y nuevas metodologías de creación de software MVC, utilización de JavaScript, Ajax, Css, dando como resultado una aplicación rápida en respuesta y diseño amigable.

SUMMARY

The Universidad Técnica de Ambato was founded on April 14, 1969 as approved by the National Congress. Born with the motto "Educate to learn to be free" under the aegis of thought and Doctor Carlos Toro Navas who presided over the creation of the first University Council, after completion of the first University Assembly a May 10, 1969 was appointed Vice President economist Victor Guzman Cabrera.

This Governmental Organization is responsible for shaping professional success in different competitive branches, through its various academic areas committed to developing the province and country.

The institution does not have a method of distribution of parking spaces available, so developing a web application that will help provide better service to the user is proposed.

The web application allows the user using a means of transportation, to enter the institution with information on a free site where you can park your vehicle, with electronic indicators used for the preparation and presentation of the location of parking places.

Whereas the design an intuitive web application is presented to the user by using new methodologies and computer software creation MVC using JavaScript, Ajax, CSS, results in a quick and friendly response application design tool.

INTRODUCCIÓN

El informe final del proyecto denominado: "**APLICACIÓN WEB PARA LA DISTRIBUCIÓN DE ESPACIOS DISPONIBLES DE PARQUEO EN LA UNIVERSIDAD TÉCNICA DE AMBATO CAMPUS HUACHI CHICO**", se lo ha dividido en capítulos para una mayor comprensión, los cuales se detallan a continuación:

CAPÍTULO I denominado "**EL PROBLEMA**", identifica el problema a investigar, además se plantea la justificación y los objetivos.

CAPÍTULO II denominado "**MARCO TEÓRICO**", presentan los antecedentes investigativos, la fundamentación teórica.

CAPÍTULO III denominado "**METODOLOGÍA**", determina la metodología de investigación a utilizar, el enfoque, la modalidad de la investigación utilizada, el tipo de investigación realizada.

CAPÍTULO IV denominado "**DESARROLLO DE LA PROPUESTA**", presenta el desarrollo de la propuesta ante el problema planteado.

CAPÍTULO V denominado "**CONCLUSIONES Y RECOMENDACIONES**", presenta las conclusiones y recomendaciones del trabajo desarrollado.

ANEXOS contiene Ficha de observación, Manuales y fotos.

CAPÍTULO I

1. EL PROBLEMA

1.1. **Tema**

Aplicación Web para la distribución de espacios disponibles de parqueo de la Universidad Técnica de Ambato campus Huachi Chico

1.2. **Planteamiento del problema**

Las empresas en los últimos años, han implantado herramientas informáticas con el fin de solucionar múltiples problemas, aprovechando las ventajas que estas herramientas ofrecen como la velocidad, seguridad de la información, sistematización, control de los servicios, entre otros beneficios que da las herramientas informáticas.

Es necesario argumentar el papel que juega la tecnología y el avance tecnológico en la humanidad, así como analizar su incidencia en distintos ámbitos sociales, educativos y empresariales ya que se ha podido observar en numerosas ocasiones, el desarrollo de nuevas técnicas tecnológicas que facilita la administración empresas e instituciones.

Además la tecnología se encuentra prácticamente disponible para cualquier institución ya que proporciona un conjunto de herramientas de hardware y software para dar soporte a la actividad individual y organizacional en el marco de una concepción global.

En el Ecuador la fuerte expansión de las tecnologías en empresas también ha alcanzado el campo de automatización de procesos y actividades que el ser

humano las realizaba, generado allí grandes cambios potenciales ya que como podemos observar existen instituciones que utilizan algún medio tecnológico que les permita controlar la utilización de los parqueaderos sea este de manera total o parcial ayudando en la administración del mismo.

En Tungurahua el uso de la tecnología para el control de procesos ha ido creciendo, automatizando los procesos en las instituciones públicas, particulares y aprovechando el uso de las tecnologías existentes y desarrollando software para que faciliten la administración, reduciendo el tiempo de búsqueda de datos y minimiza la duplicidad de la misma, en el campo de control de parqueaderos podemos observar algunos formas de control como por ejemplo:

Mall de los Andes y el del Hotel Emperador de La ciudad de Ambato donde es evidente el control de uso de parqueaderos, con la carencia de un módulo de asignación de espacios que facilite el uso del mismo.

En la Universidad Técnica de Ambato campus Huachi Chico, existe el problema de asignación de espacios disponibles de parqueo, ya que al no contar con un sistema o método tecnológico que facilite la asignación de sitio disponibles en el mismo, provoca grandes inconvenientes a la comunidad universitaria, causando pérdida de tiempo al elegir un lugar donde ubicar el vehículo, tanto del personal estudiantil, docente y administrativo que utiliza un medio de trasporte, debido a que cuenta con la tecnología para el control de ingreso de vehículos a sus instalaciones, mas no para la distribución de puntos utilizables de parqueo.

En la actualidad la Universidad Técnica de Ambato campus Huachi Chico, al no contar con señalización de áreas de parqueadero autorizadas para la misma, provoca que se utilice lugares que no son parte del estacionamiento ocasionando daños en bines de la institución.

La inexistencia de método tecnológico que ayude a la distribución de espacios de parqueo en la Universidad Técnica de Ambato campus Huachi Chico, es el principio de la afluencia de vehículos en la institución, ocasionada por no contar

con una media tecnológica de distribución de sitios valederos provocando una deplorable estética a la institución.

El diseño parcial de la aplicación de control parqueadero utilizado por la institución, provoca que no se pueda realizar el proceso de distribución de los vehículos que ingresan a la institución, causando molestia y pérdida de tiempo al usuario al momento de seleccionar un sitio de estacionamiento.

El poco interés para automatizar el proceso de asignación de sitios de estacionamientos, es la procedencia que la información de esta actividad se nula dificultando verificar la existencia de espacios disponibles que agilite dicho proceso entregándonos información verídica a la comunidad universitaria de la UTA.

1.3. **Delimitación del problema**

Área académica: Software.

Línea de investigación: Desarrollo de software.

Sublinea: Aplicación Web.

Delimitación Espacial

La presente investigación se realizará en la Universidad Técnica de Ambato campus Huachi Chico"

Delimitación Temporal

La presente investigación se realizará en el período de seis meses a partir de la aprobación del proyecto por Honorable Consejo Directivo de la Facultad de Ingeniería en Sistemas Electrónica e Industrial.

1.4.**Justificación**

El presente proyecto se justifica porque la Universidad técnica de Ambato campus Huachi Chico, al ser una institución de prestigio, tiene la necesidad y está interesado en un método de distribución de espacios de estacionamientos que mejore el servicio y utilización del mismo.

Un método de distribución de sitios de estacionamientos es necesario porque facilitara la búsqueda de zonas de estacionamientos, permitiéndonos localizar de una manera confiable y rápida la ubicación de lugares en la institución, llevándonos a estar a la par o superar a las instituciones de primer nivel, satisfaciendo de esta manera las necesidades de los usuarios de la institución.

El impacto será alto ya que se implantara una manera de realizar la distribución de espacios de estacionamiento en la institución, creando un nuevo método que resuelva esta necesidad.

Además es factible realizarlo porque tendrá el apoyo de facultad de Ingeniería en Sistemas, Electrónica e Industrial, contando además con la colaboración del personal que lleva a cabo el proceso de control del parqueadero en la Universidad Técnica de Ambato campus Huachi Chico.

1.5. **Objetivos**

1.5.1. **Objetivo general**

Desarrollar una Aplicación Web para la distribución de espacios disponibles de parqueo en la Universidad Técnica de Ambato campus Huachi Chico.

1.5.2. **Objetivos específicos**

Analizar el proceso manual de control de espacios disponibles de parqueaderos en la Universidad Técnica de Ambato campus Huachi Chico.

Determinar los métodos de asignación existentes de espacios de parqueo en la Universidad Técnica de Ambato campus Huachi Chico.

Analizar las características de la aplicación web para el control de espacios disponibles de parqueo en la Universidad Técnica de Ambato campus Huachi Chico.

Diseñar una aplicación web que mejore la asignación de espacios de parqueo en la Universidad Técnica de Ambato campus Huachi Chico.

Desarrollar la aplicación web para la asignación de espacios disponibles de parqueo en la Universidad Técnica de Ambato campus Huachi Chico.

Realizar las pruebas del funcionamiento la aplicación web de asignación de espacios disponibles de parqueo en la Universidad Técnica de Ambato campus Huachi Chico.

CAPÍTULO II

2. MARCO TEORICO

2.1. **Marco Teórico**

Luego de haber revisado los proyectos y tesis en la Biblioteca de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial de la Universidad Técnica de Ambato no se ha encontrado temas de Aplicación Web para el control de asignación de parqueo, pero se encontraron tesis referentes a sistemas web como por ejemplo:

"Sistema web integrado para la gestión de cobranza de valores en el Gobierno Provincial de Tungurahua, este proyecto se utiliza para que facilite y optimice el proceso de recaudación y almacenamiento de información, a través de él se podrá controlar de mejor manera los valores recaudados y comprobantes emitidos, la recuperación de la información será rápida y eficiente, eliminando la duplicidad de datos, existirá seguridad en la información almacenada, se realizará un control y seguimiento de los pagos que cada vehículo matriculado en la provincia tenga que obligatoriamente realizar a partir de su registro en la provincia [1]".

"Servicio Web para realizar transferencias de fondos en línea entre Instituciones financieras del país a través del Banco Central del Ecuador para la Cooperativa de Ahorro y Crédito Chibuleo Ltda. De la Ciudad de Ambato, brindan accesibilidad y compatibilidad con prácticamente todas las plataformas que hoy en día son usadas para el desarrollo de aplicaciones, además de asegurar información vital para la empresa mediante el protocolo HTTPS y la encriptación de datos. Por esta razón el desarrollo de un Servicio Web para la Cooperativa de Ahorro y Crédito Chibuleo Ltda [2]"

En la Universidad Técnica de Ambato el control de espacios disponibles no cuenta con un proceso que ayude a la distribución del mismo, ya que solo cuenta la con control de ingreso y salida de vehículos, mediante la utilización de tiquetes y tarjetas magnéticas adquiridas para el uso del parqueadero durante todo un semestre, para la salida se la realiza de manera similar mediante el comprobantes, estos son los adquiridos a la entrada del mismo, cobrando un valor por el servicio de brindado, donde tiene que intervenir el personal humano para el cobro del mismo, además no se controla la total máximo permitido, dando como resultado que se exceda en la cantidad de vehículos, de la misma manera se lo realiza por las tarjetas magnéticas sin rubro alguno.

La Universidad Técnica de Ambato, solo cuenta con una señalización de espacios de parqueadero, en cada una de las facultades ubicados en diferentes lugares de la institución y un sistema tarifado de utilización del mismo.

2.1.1. **Sistema De Control**

"Norman S. Nise define como un sistema de control como un conjunto de componentes que pueden regular su propia conducta o la de otro sistema con el fin de lograr un funcionamiento predeterminado" [3]

"Para Brotons lo define como Un sistema de control es un tipo de sistema que se caracteriza por la presencia de una serie de elementos que permiten influir en el funcionamiento del sistema. La finalidad de un sistema de control es conseguir, mediante la manipulación de las variables de control, un dominio sobre las variables de salida, de modo que estas alcancen unos valores prefijados" [4]

2.1.2. **Control de Procesos**

"Ing. José Roberto, define control de proceso como se controlan variables inherentes al mismo, para incrementar la eficiencia y reducir las variables del producto final" [5].

"Roberto Pastrana, El sistema de control permitirá una operación del proceso más fiable y sencilla, al encargarse de obtener unas condiciones de operación estables, y corregir toda desviación que se pudiera producir en ellas respecto a los valores de ajuste" [6].

2.1.3. **Tipos de Controles**

"Lic. Adm. Sabino Ayala Villegas, "El control puede definirse como la evaluación de la acción, para detectar posibles desvíos respecto de lo planeado, desvíos que serán corregidos mediante la utilización de un sistema determinado cuando excedan los límites admitidos" [7].

2.1.3.1. **Controles computacionales**

"Sistemas de información computarizados son los tienen un soporte informático, es decir se desarrollan en un entorno usuario - computadora, utilizando hardware y software computacional, redes de telecomunicaciones, técnicas de administración de bases de datos computarizadas y otras formas de tecnología de información" [8].

2.1.4. **Control Automático**

"Sistema de control basado en la aplicación del concepto de realimentación (medición tomada desde el proceso que entrega información del estado actual de las variables que se desea controlar) cuya característica especial es la de mantener al controlador central informado del estado de las variables para generar acciones correctivas cuando así sea necesario. Puede estar constituido por múltiples lazos de control" [9].

2.1.5. **Modelos de Asignación de espacios**

“Salazar López declare que El método de la esquina Noroeste es un algoritmo heurístico capaz de solucionar problemas de transporte o distribución mediante la consecución de una solución básica inicial que satisfaga todas las restricciones existentes sin que esto implique que se alcance el costo óptimo total.

Este método tiene como ventaja frente a sus similares la rapidez de su ejecución, y es utilizado con mayor frecuencia en ejercicios donde el número de fuentes y destinos sea muy elevado. Su nombre se debe al génesis del algoritmo, el cual inicia en la ruta, celda o esquina Noroeste. Es común encontrar gran variedad de métodos que se basen en la misma metodología de la esquina Noroeste, dada que podemos encontrar de igual manera el método e la esquina Noreste, Sureste o Suroeste” [10].

“Salazar López el método del costo mínimo o de los mínimos costos es un algoritmo desarrollado con el objetivo de resolver problemas de transporte o distribución, arrojando mejores resultados que métodos como el de la esquina noroeste, dado que se enfoca en las rutas que presentan menores costos. El diagrama de flujo de este algoritmo es mucho más sencillo que los anteriores dado que se trata simplemente de la asignación de la mayor cantidad de unidades posibles (sujeta a las restricciones de oferta y/o demanda) a la celda menos costosa de toda la matriz hasta finalizar el método” [10]

2.1.6. **Administración optimizada de estacionamiento**

Siemens Mesoamérica “detalla como el requerimiento de administración que involucra la guía inteligente para el conductor, el monitoreo dirigido y la señalización de espacios libres de estacionamiento (monitoreo de espacios individuales, guía de vehículos)” [11].

Para el investigador es la suspensión del movimiento del vehículo y su colocación en lugares y posiciones determinadas, generalmente con el motor detenido, durante un período dado dependiendo de la disponibilidad del espacio de parqueo de un área o lugar.

2.1.7. **Control de asignación de parqueadero**

Para el investigador el control de asignación es la forma como se controla el ingreso y asignación de espacios disponibles de parqueo para las distintas necesidades de los usuarios que utilizan los espacios de parqueo dependiendo de la disponibilidad del mismo o necesidad del usuario.

2.1.8. **Apache**

"El Proyecto Apache HTTP Server es un esfuerzo para desarrollar y mantener un servidor HTTP de código abierto para sistemas operativos modernos, incluyendo UNIX y Windows NT. El objetivo de este proyecto es proporcionar un servidor seguro, eficiente y extensible que proporciona los servicios de HTTP en sincronización con los estándares HTTP actuales" [12].

2.1.9. **Aplicación Web**

"Una aplicación web es una aplicación informática distribuida cuya interfaz de usuario es accesible desde un cliente web, normalmente un navegador web" [13]

2.1.10. **Servidor web**

"Un servidor web es un programa que se ejecuta continuamente en un computador, manteniéndose a la espera de peticiones de ejecución que le hará un cliente o un usuario de Internet. El servidor web se encarga de contestar a estas peticiones de forma adecuada, entregando como resultado una página web o información de todo tipo de acuerdo a los comandos solicitados" [12].

2.1.11. **Navegador**

"Un navegador web o explorador web es una aplicación software libre que permite al usuario recuperar y visualizar documentos de hipertexto, comúnmente descritos en HTML, desde servidores web de todo el mundo a través de Internet" [12].

2.1.12. **POO**

“El término de Programación Orientada a Objetos indica más una forma de diseño y una metodología de desarrollo de software que un lenguaje de programación, ya que en realidad se puede aplicar el Diseño Orientado a Objetos a cualquier tipo de lenguaje de programación que en programador o diseñador lo decidan realizar” [14].

2.1.13. **PostgreSQL**

“Entre los sistemas de bases de datos existentes hoy en día, PostgreSQL juega un papel muy importante ya que es un sistema que tiene muchas cualidades que lo hacen ser una muy buena alternativa para instalar sistemas en empresas, universidades y una gran cantidad de otras aplicaciones por la versatilidad que tiene y lo más importante es que no es un software propietario y fácil de manejar” [15].

2.1.14. **Estándar de IEEE 830**

“Ayuda a los clientes a describir claramente lo que se desea obtener mediante un determinado software: El cliente debe participar activamente en la especificación de requisitos, ya que éste tiene una visión mucho más detallada de los procesos que se llevan a cabo. Asimismo, el cliente se siente partícipe del propio desarrollo.

Ayudar a los desarrolladores a entender qué quiere exactamente el cliente: En muchas ocasiones el cliente no sabe exactamente qué es lo que quiere. La ERS permite al cliente definir todos los requisitos que desea y al mismo tiempo los desarrolladores tienen una base fija en la que trabajar. Si no se realiza una buena especificación de requisitos, los costes de desarrollo pueden incrementarse considerablemente, ya que se deben hacer cambios durante la creación de la aplicación.

Servir de base para desarrollos de estándares de ERS particulares para cada organización: Cada entidad puede desarrollar sus propios estándares para definir sus necesidades” [16].

2.2. **Propuesta de solución**

La presente investigación propone una Aplicación Web para la distribución de espacios disponibles de parqueo de la Universidad Técnica de Ambato campus Huachi Chico.

CAPÍTULO III

3. METODOLOGIA

3.1.**Modalidad de la investigación**

La investigación es aplicada ya que permitirá dar solución a la problemática de distribución de espacios disponibles de estacionamientos, con la ayuda de los educadores de planta o cátedra de la Facultad de Ingeniería en Sistemas, electrónica e Industrial o un técnico en el tema sobre el cual se está realizando el tema de investigación.

Modalidad Bibliográfica o Documentada: Se ha considerado esta modalidad porque se ha tomado información de Internet, Libros virtuales, Tesis, Artículos publicados en la Web etc.

Modalidad De Campo: Se ha considerado esta modalidad ya que el investigador irá a recoger la información primaria directamente de los involucrados a través de encuestas y entrevistas acerca de proceso manual de control de espacios de parqueo en relación con su distribución de espacios disponibles de parqueo.

3.2.**Población y muestra**

La población o muestra no son requeridas ya que no se ha planteado una hipótesis para la aplicación web.

3.3. **Operacionalización de objetivos**

Objetivo	**Actividades**
Analizar el proceso manual de control de espacios disponibles de parqueaderos en la Universidad Técnica de Ambato campus Huachi Chico.	Observación del proceso manual de control de espacios de parqueo en la institución ficha de observación. Describir el proceso de control tarifado. Determinar el proceso manual.
Determinar los métodos de asignación existentes de espacios de parqueo en la Universidad Técnica de Ambato campus Huachi Chico.	Observación de los métodos actuales de asignación de parqueo ficha de observación. Determinación de métodos de asignación utilizada por la institución.
Analizar las características de la aplicación web para el control de espacios disponibles de parqueo en la Universidad Técnica de Ambato campus Huachi Chico.	Levantamiento de requerimiento. Selección de Lenguaje de programación.
Diseñar una aplicación web que mejore la asignación de espacios de parqueo en la Universidad Técnica de Ambato campus Huachi Chico.	Diseño de la base de datos. Diseño de las interfaces de los distintos usuarios. Maquetación de la página web.
Desarrollar la aplicación web para la asignación de espacios disponibles de parqueo en la Universidad Técnica de Ambato campus Huachi Chico.	Creación de la base de datos Configuración de servidor web Codificación de la página web Instalación de los dispositivos electrónicos.
Realizar las pruebas del funcionamiento la aplicación web de asignación de espacios disponibles de parqueo en la Universidad Técnica de Ambato campus Huachi Chico.	Pruebas de caja negra Pruebas de caja blanca

Tabla 3.1: Operacionalización de variables u objetivos
Elaborado por: Wilson Pérez – Investigador

3.4.**Recolección de información**

Se realizara la recolección de la información directamente con el personal encargado del parqueadero de la Universidad técnica de Ambato campus Huachi Chico, quienes controlan el ingreso y salida de parqueadero de la institución a través de contacto directo.

La recolección de la información secundaria se la realizara estudios realizados anteriormente como tesis de grado que se han realizado con anterioridad además mediante información disponible en el internet ya sea libros virtuales como paginas o artículos que proporciones información validad sobre el tema de investigación.

3.5. **Procesamiento y análisis de datos**

Para la recolección, procesamiento y análisis de la información se aplicará los siguientes procedimientos:

- Elaboración del instrumento de recolección de datos
- Aplicación de instrumento de recolección de información.
- Análisis de la información.
- Resultados.

3.6. **Desarrollo del Proyecto.**

Estándar de IEEE 830

1 Introducción
1.1 Propósito
1.2 Ámbito del Sistema
1.3 Definiciones, Acrónimos y Abreviaturas
1.4 Referencias
1.5 Visión general del documento
2 Descripción General
2.1 Perspectiva del Producto
- 2.1.1. Indicar si es un producto independiente o parte de un sistema mayor
- 2.1.2. Interfaces de sistema
- 2.1.3. Interfaces de usuario
- 2.1.3.1. Características lógicas del interfaz
- 2.1.3.2. Cuestiones de optimización del interfaz de usuario
- 2.1.4. Interfaces hardware
- 2.1.5. Interfaces software
- 2.1.5.1. Descripción del producto software utilizado
- 2.1.5.2. Propósito del interfaz
- 2.1.5.3. Definición del interfaz: contenido y formato
- 2.1.6. Interfaces de comunicaciones
- 2.1.7. Limitaciones de memoria
- 2.1.8. Operaciones
- 2.1.8.1. Modos de operación de los distintos grupos de usuarios
- 2.1.8.2. Periodos de operaciones interactivas y automáticas
- 2.1.8.3. Funciones respaldo del procesamiento de datos
- 2.1.8.4. Operaciones de backup y recuperación
- 2.1.9. Requerimientos para adaptarse a la ubicación
- 2.1.9.1. Indicar cualquier dato o secuencia de inicialización específico de cualquier lugar, modo de operación.
- 2.1.9.2. Características que deben ser modificadas para una instalación en particular.

2.2 Funciones del Producto
2.3 Características de los usuarios
2.4 Restricciones
2.5 Suposiciones y Dependencias
2.6 Requisitos Futuros
3 Requisitos Específicos
3.1 Interfaces Externas
3.2 Funciones
3.3 Requisitos de Rendimiento
3.4 Restricciones de Diseño
3.5 Atributos del Sistema
3.6 Otros Requisitos

CAPÍTULO IV

4. DESARROLLO DE LA PROPUESTA

4.1. **Analizar el proceso manual de control de espacios disponibles de parqueaderos.**

El proceso de control de ingreso y salida a la institución es el siguiente.

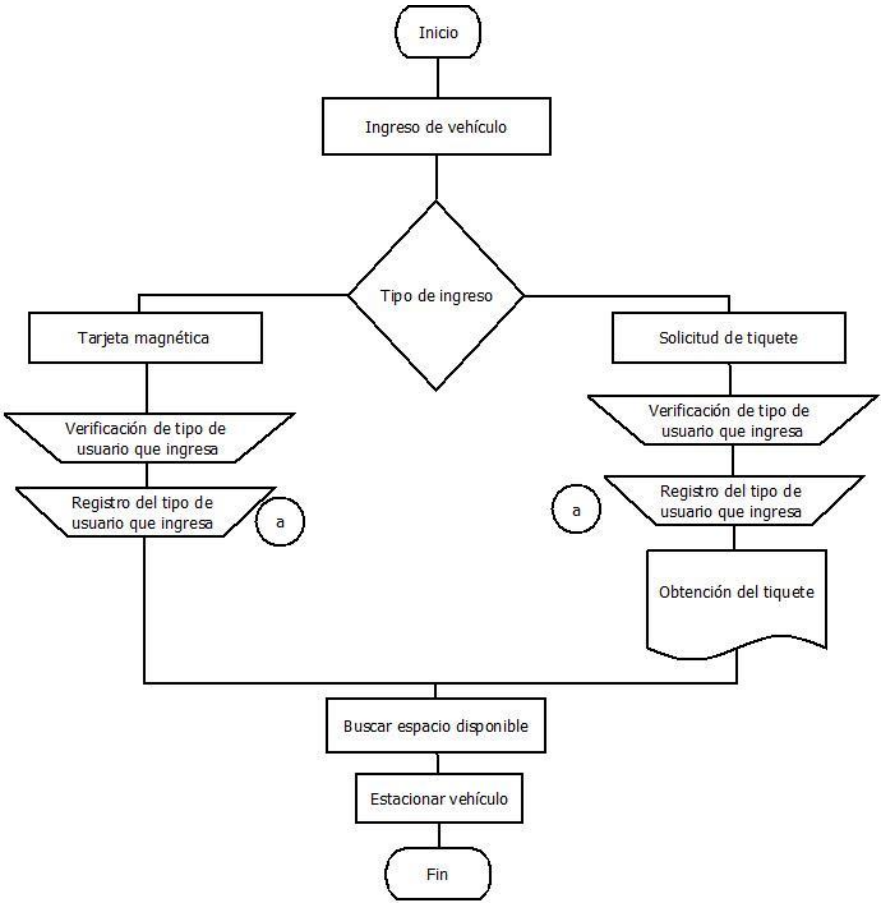

Figura 4.1: Proceso de ingreso al parqueadero

Elaborado por: Wilson Pérez – Investigador

Mediante la observación y acercamiento con el personal que controla el ingreso al parqueadero de la universidad, el control es realizado por personal de seguridad de la institución quien entrega un tiquete y verifican mediante adhesivos ubicados en los vehículos, el tipo de usuario que ingresa a la institución.

Los propietarios de los vehículos que solicitan un lugar en el parqueadero de la universidad campus Huachi Chico, obtienen la información por el personal de institución, pero la información requerida de la ubicación es proporcionada sin la fiabilidad de que exista o no sitio de estacionamiento disponibles en la misma.

FICHA DE OBSERVACIÓN 1 ÁREA: Universidad Técnica De Ambato campus Huachi Chico. Observador: Investigador	
Objetivo: Analizar el proceso manual de control de espacios disponibles de parqueaderos Observaciones	
Como es el proceso manual de control de lugares de estacionamiento.	Como, información guardada, resultados obtenidos. Según lo observado en el proceso manual es realizado de la siguiente manera: Este proceso es evidente en el paso de registro de tipo de usuarios, donde hay un subproceso que se utiliza para cumplir con lo anterior descrito, llevando acabo las siguientes actividades. a) Verificación del tipo de usuario. b) Identificación de la dependencia a la que pertenece. c) Registro y distribución interna. Para la actividad a) existe una persona autorizada que está ubicada en la entrada principal de la institución donde mediante un identificador (adhesivo), que es el permite el reconocimiento del mismo y que se encuentra en

	parabrisas de vehículo, donde podemos verificar el tipo de usuario. Si el adhesivo no se encuentra en un auto se determina que el usuario es un visitante que no está relacionado con alguna actividad con la institución. Para la actividad b) al igual que en el caso anterior podemos determinar la dependencia a la que pertenece, con información que está el medio de control establecido por la institución, y así determinar si es parte de la institución. Si no es parte de la institución no cuenta con el identificador, por lo tanto se lo registra con una pregunta del lugar al cual se dirige y registrando en un documento informal (hojas, cuaderno). Para la actividad c) el procedimiento es el mismo donde se guarda la información de a la dependencia como referencia de la ubicación o el lugar de estacionamiento. Esto es para sacar resultados de cuantos vehículos ingresan a diario, la cantidad que ingresas a las distintas secciones de parqueadero.
Observaciones: Se puede determinar que existe un proceso manual informal que utilizan los encargados del parqueadero de la institución.	

Tabla 4.1: Ficha de observación proceso manual de asignación

Elaborado por: Wilson Pérez – Investigador

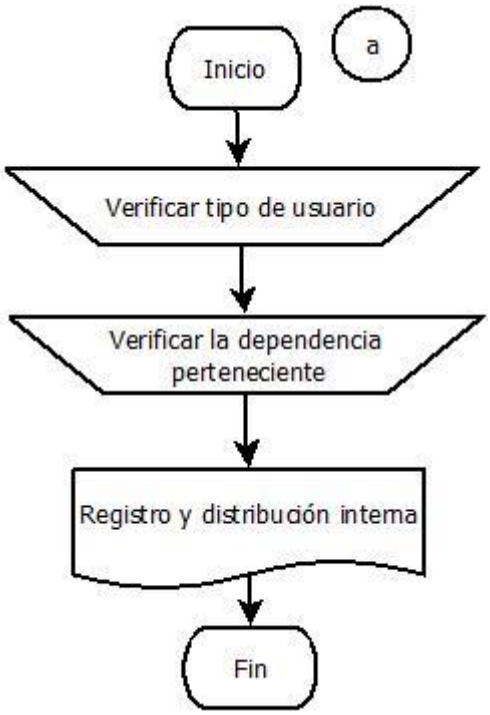

Figura 4.2: Proceso manual distribución de espacios

Elaborado por: Wilson Pérez – Investigador

4.2. **Determinar los métodos de asignación existentes de espacios de parqueo**.

Utilizando como técnica la observación y como instrumento una guía de observación, se ha recolectado los datos necesarios, que permiten conocer los métodos actuales de asignación de espacios de parqueo en la Universidad técnica de Ambato campus Huachi chico.

FICHA DE OBSERVACIÓN 2 ÁREA: Universidad Técnica De Ambato campus Huachi Chico Observador: Investigador	
Objetivo: Determinar los métodos de asignación existentes de espacios de parqueo Observación	
Métodos existentes	Según lo observado por el investigador, se puede determinar que existe un método, el cual es un proceso manual de distribución

	informal donde los encargados del parqueadero de la institución, utilizan, por medio de una actividad de registro al ingreso del vehículo.
Ventajas	El método no tiene ventajas, ya que al ser un método informal que utilizan los administradores del parqueadero de la institución, solo como un registró.
Desventajas	Provoca que los vehículos que ingresan tengan dificultades y pérdida de tiempo al momento de elegir un espacio dentro del mismo, para el desarrollo de sus labores o actividades que desempeñe en la institución. La información proporcionada sobre la existencia de lugar de estacionamiento es nula, ya que al carecer de indicadores o métodos de asignación, resulta complicado llegar a las dependencias o facultades de la institución. El método de asignación existe pero no está definido formalmente en la institución. El proceso no está definido correctamente debido a las limitaciones de limitaciones de conocimiento del personal que administrador del mismo.
Problemas ocasionados	Sien los problemas más comunes los siguientes : ➢ Pérdida de tiempo ➢ Exceso de vehículos en la institución. ➢ Molestia en los usuarios. ➢ Retraso en las actividades de los usuarios al no localizar con facilidad un lugar de estacionamiento. ➢ Utilización de zonas no señaladas para uso de estacionamiento.

Observaciones: Existe un método informal pero no un método para la distribución de espacios en la cual se utilice la tecnología para el control del propio.

Tabla 4.2: Ficha de observación métodos de asignación

Elaborado por: Wilson Pérez – Investigador

4.3. **Análisis y diseño de la aplicación**

El levantamiento de requerimientos se los realizo con la observación directa y mediante un documento de análisis, previo realizado por la institución que se constara en los anexos del presente trabajo.

El diseño de la aplicación se la realiza mediante el análisis de la recopilación de información obtenida por el investigador, para la realización de la aplicación web, la cuan ayudara a la distribución de sitios de estacionamiento de la siguiente forma.

El vehículo ingresara al parqueadero de la institución, siempre y cuando exista espacios disponibles de estacionamiento, esto se realiza mediante la presentación de información al usuario que controla la aplicación web donde se puede observar los espacios disponibles totales así con por secciones.

Una vez que vehículo ingresa a la institución, el conductor observara señalizaciones mediante matrices led, que señalan que sección están ocupadas, así como indicadores de donde existen espacios disponibles.

.

4.4. **Estándar de IEEE 830**

Es un estándar que ayuda para determinar los requisitos de una aplicación o software mediante un formato establecido por la IEEE que facilitara el desarrollo del mismo ya que mediante esta podemos realizar el diseño de nuestra aplicación.

4.5. **Introducción**

4.5.1. **Propósito**

El siguiente tiene como objetivo definir las especificaciones funcionales de la aplicación así como el alcance del mismo, para el desarrollo de una "Aplicación Web para la distribución de espacios disponibles de parqueo en la Universidad Técnica de Ambato campus Huachi Chico". El mismo que será utilizado por el personal administrativo, docente, estudiantil y sociedad que ingresa al parqueadero.

4.5.2. **Ámbito del Sistema**

La aplicación tendrá como nombre UtaPark, el cual permite realizar la distribución de espacios disponibles de parqueadero mediante indicadores electrónicos señalándonos secciones con espacios utilizables.

La aplicación facilitara la administración y control del parqueadero de la universidad por parte del personal que administra el mismo, ayudando a tener información de la cantidad de espacios ocupados, evita que el número de vehículos ingresados superan el número de espacios existentes en el mismo permitiendo de esta manera evitar la congestión e inconvenientes por el uso de espacios que no son aginados como parqueaderos.

4.5.3. **Definiciones, Acrónimos y Abreviaturas**

Nombre	Descripción
APW	Aplicación Web
ERS	Especificación de Requisitos Software
SCAFFOLDING	Scaffolding es un método para construir aplicaciones basadas en bases de datos, esta técnica está soportada por algunos frameworks del tipo MVC.
MVC	Modelo vista controlador.
CAMELCASE	Es un estilo de escritura que se aplica a palabras compuestas.

Tabla 4.3 Definiciones, Acrónimos y Abreviaturas

Elaborado por: Wilson Pérez - Investigador

4.5.4. **Referencias**

Título del Documento	Referencia
Standard IEEE 830 – 1998	IEEE

Tabla 4.4 Referencias

Elaborado por: Wilson Pérez - Investigador

4.5.5. **Visión general del documento**

Este documento consta de tres secciones. En la primera sección se realiza una introducción al mismo y se proporciona una visión general de la especificación de recursos del sistema.

En la segunda sección del documento se realiza una descripción general del sistema, con el fin de conocer las principales funciones que éste debe realizar, los datos asociados y los factores, restricciones, supuestos y dependencias que afectan al desarrollo, sin entrar en excesivos detalles.

Por último, la tercera sección del documento es aquella en la que se definen detalladamente los requisitos que debe satisfacer el sistema.

4.6. **Descripción General**

4.6.1. **Perspectiva del Producto**

El APW será un producto diseñado para trabajar en entornos WEB, lo que permitirá su utilización de forma rápida y eficaz, además será integrara conjuntamente con el sistema de control de parqueadero de la universidad para lograr una mejor respuesta.

4.6.1.1. **Indicar si es un producto independiente o parte de un sistema mayor**

Nuestra aplicación será independiente ya que no contamos con el acceso al sistema de control de parqueaderos de la universidad por que dicho sistema es propietario y el acceso al mismo es restringido y no se podrá incorporar al mismo.

4.6.1.2. **Interfaces de sistema**

La interfaz del sistema es lo que todas las partes que tendrán en común entro los diferentes usuarios para lo cual utilizaremos técnicas que facilite el diseño de la interfaz de nuestro sistema entre los que tenemos:

Como por ejemplo lo que será propio del sistema como pantalla de inicio, área de configuración del sistema donde se encontrara el nombre de la institución o logo del sistema como también el menú con las opciones de información del usuario conectado así como la configuración del mismo y además la opción de salir del sistema además una opción de acerca de.

Otras opciones que deber realizar el sistema es la generación de códigos de barras, así como también la generación de tikets de ingreso al parqueadero, la utilización de tarjetas magnéticas y leer código de barras.

4.6.1.3. **Interfaces de usuario**

La interfaz con el usuario consistirá en un conjunto de ventanas con botones, listas y campos de textos. Ésta deberá ser construida específicamente para el sistema propuesto y, será visualizada desde un navegador de internet

Interfaz de usuario se refiere a los elementos y características específicas de una aplicación web, incluyendo los aspectos funcionales y visuales. El diseño de interfaz de usuario de su aplicación web puede hacer la diferencia entre una experiencia inteligente, agradable o que frustra a los usuarios.

Se detallara las funciones de cada una de las interfaces de los usuarios del sistema:

Lo que tendrá en común todas las interfaces de los usuarios será ingreso y salida del sistema así como un menú donde estará la configuración del usuario autentificado él acerca de del sistema.

Admin

Esta interfaz además de las lo que tienen en común las demás interfaces podrá realizar la administración de todos los usuarios del sistema, así como crear más grupos y la respectiva configuración de lo que cada grupo podrá hacer en el sistema.

Administrador

La interfaz tendrá la administración de los distintos usuarios que utilizaran el parqueadero que son como losa Administrativos, Docentes, Estudiantes entre otros así como la también la administración de los precios de por la utilización del mismo.

Otra de las características será que contara con el registro de las secciones que contendrá el parqueadero, y los procesos correspondientes al proceso de ver espacios disponibles de parqueaderos de la institución.

Usuarios

Verificar la disponibilidad de espacios existentes en el parqueadero, cobros de los espacios utilizados y la visualización de espacios disponibles por sección y otros.

Anónimos

Estos solo podrán observar la página de información de los clientes.

4.6.1.3.1. Características lógicas del interfaz

No se ha definido.

4.6.1.3.2. Cuestiones de optimización del interfaz de usuario

No se ha definido.

4.6.1.4. **Interfaces hardware**

Será necesario disponer de equipos de cómputos en perfecto estado con las siguientes características:

Especificaciones para el área de control de ingreso y salida de los cetros de control de parqueaderos.

- Adaptadores de red.
- Procesador de 1.66GHz o superior.
- Memoria mínima de 1 GB.
- Mouse.
- Teclado.
- Lector de tarjetas magnéticas
- Lector de código de barras

Especificaciones para cada espacio de parqueadero.

- Sensor electromagnético

4.6.1.5. **Interfaces software**

Las interfaces software se refiere a las características y software adicional de algún requerimiento o dispositivo adicional que sea necesario para el funcionamiento del sistema

4.6.1.5.1. **Descripción del producto software utilizado**

Cuadro de comparación para selección de la herramienta de programación que facilite el desarrollo del mismo de maenara que cumpla con la propuesta de solución.

Lenguajes de programación	
Php	Ventajas:
	➢ Orientado a objetos. ➢ Multi plataforma ➢ Garantiza una alta velocidad ➢ Es gratuito. ➢ Orientado a desarrollar aplicaciones web dinámicas. ➢ Capacidad de conexión con la mayoría de motores de base de datos como MySQL, portgestSQL etc. ➢ Información actualizada y disponible en la web. ➢ Integración de módulos (extensiones). ➢ Soportado por varios servidores web ➢ MVC
	Desventajas:
	➢ Puedes descargar el código fuente con programas. ➢ Le seguridad depende del nivel de programación.
Java	Ventajas:
	➢ Orientado a objetos. ➢ Multi plataforma ➢ Orientado a desarrollar aplicaciones web dinámicas. ➢ Capacidad de conexión con la mayoría de motores de base de datos como MySQL, portgestSQL etc. ➢ Soporte actualizado y disponible en la web. ➢ MVC
	Desventajas:
	➢ La velocidad de aplicación no son muy rápidas ➢ Algunas herramientas de desarrollo son de paga
C#	Ventajas:
	➢ Orientado a objetos ➢ Orientado a desarrollar aplicaciones web dinámicas

	➢ Soporte actualizado y disponible en la web. ➢ MVC
	Desventajas:
	➢ Pagado ➢ Limitaciones de conexión con la mayoría de motores de base de datos ➢ No es multi plataforma

Tabla 4.5: Cuadro de comparación de lenguajes de programación

Elaborado por: Wilson Pérez - Investigador

Se escogió php como lenguaje de desarrollo por las características que ofrece sobre las otras opciones.

- Sistema operativo libre o propietario
- Desarrollo MVC cakephp
- Navegadores Firefox, Explorer, Chrome
- Servidor Web
- Motor de base de datos Postgres
- Driver de tarjeta magnética

4.6.1.5.2. **Propósito del interfa**z

Sistema operativo libre o propietario

- Sera donde realizara la instalación de nuestro aplicación y además serán nuestros clientes en los distintos navegadores ya que será una aplicación cliente servidor.

Framework de desarrollo MVC cakephp

- Este será nuestra plataforma de desarrollo de nuestra aplicación que facilitara la creación de nuestra aplicación mediante tecnología MVC que es Modelo vista controlador.

Navegadores Firefox, Explorer, Chrome

- Sera donde la aplicación tendrá Front-end que se presentara a los distintos usuarios que utilizaran nuestro sistema.

Servidor Web

- Permitirá alojar nuestra aplicación web que soportara el despliegue de etiquetas html y php y que soporten extensiones para trabajar con postgesql .

Motor de base de datos Postgres

- Será donde se guardara la información de los proceso de nuestra aplicación.

Driver de tarjeta magnética

- Permitirá la conexión entre lector y nuestro sistema operativo y la aplicación desarrollada.

4.6.1.6. **Definición del interfaz: contenido y formato**

Framework de desarrollo MVC cakephp

Los formatos para el desarrollo son mediante las convenciones propias que maneja el framework:

Convenciones de los nombres de archivos y clases

Los nombres de archivo llevan el símbolo underscore "_", mientras que los nombres de las clases usan CamelCase.

Convenciones de Modelo y de la Base de datos

Los nombres de las clases de modelos están en singular y en formato CamelCase.

Por ejemplo PersonaGrande

Los nombres de las tablas correspondientes a modelos de CakePHP están en plural y usando guion bajo cuando deseamos separar palabras.

La tabla de la base de datos deberá de tener un identificador único que es su clave primaria con un id como capo único para cada tabla, y que sea un serial de tipo serial.

Por ejemplo, personas_grandes

Los nombres de los campos con dos o más palabras se definen con guiones bajos:

Por ejemplo, fecha_de_nacimiento.

Las claves foráneas no son requeridas que se las una necesariamente con un join, ya que se las puede unir mediante el framework mediante las siguientes tipos de relaciones entre modelos.

Como se describe en la siguiente tabla:

Relación	**Tipo de asociaciones**	**Ejemplos**
Uno a uno	hasOne	Un usuario tiene un perfil.
Uno a muchos	hasMany	Un usuario puede tener múltiples recetas.
Muchos a uno	belongsTo	Muchas recetas pertenecen a un usuario.
Muchos a muchos	hasAndBelongsToMany	Recetas tienen, y pertenecen a, muchos ingredientes.

Tabla 4.6 Tipos de asociaciones.

Elaborado por: Wilson Pérez – Investigador

El nombre por defecto de las claves foráneas en relaciones hasMany, belongsTo o hasOne, es el nombre de la tabla relacionada (en singular) seguido de _id.

Por ejemplo, user_id.

El nombre de las tablas de unión entre modelos, usadas en relaciones, debería estar formado por el nombre de las tablas que une puestos en orden alfabético.

CakePHP no soporta claves primarias compuestas. Si desea manipular directamente los datos de una tabla de unión, se deben hacer llamadas directas a query o adicionar una clave primaria para que actuara como un modelo normal.

En vez de utilizar una clave autoincremental como clave primaria (campo de nombre id), se puede utilizar char(36). De este modo CakePHP utilizará un uuid (String::uuid) único de 36 caracteres siempre que grabes un nuevo registro utilizando el método Model::save.

Convenciones de Controladores

Los nombres de las clases de los controladores son en plural, con formato CamelCase, y Terminan en Controller.

El primer método que se escribe para un controlador debe de ser el método index(). Cuando la petición especifica un controlador pero no una acción, el comportamiento por defecto de CakePHP es ejecutar el método index() de dicho controlador.

También se puede cambiar la visibilidad de los métodos de los controladores en CakePHP anteponiendo al nombre del método guiones bajos. Si un método de un controlador comienza por un guión bajo, el método no será accesible directamente desde la web, sino que estará disponible sólo para uso interno.

Convenciones de Vistas

Los archivos de plantillas de Vistas (Views) deben ser nombradas después de las funciones de los controladores con guión bajo "_".

El patrón básico es:

/app/views/controller/index.ctp.

Consideraciones de URL para nombres de controladores:

Los controladores con un nombre simple pueden ser fácilmente mapeados a una URL en minúsculas.

Por ejemplo, GroupsController (que se define en el archivo "groups_controller.php") y accedido desde http://example.com/groups.

Por otro lado múltiples combinaciones de palabras pueden ser transformadas automáticamente en un mismo nombre de controlador:

- /adminGroups
- / AdminGroups
- / Admin_Groups
- / admin_groups

Todas resuelven la acción index de controlador AdminGroups. Sin embargo, la convención es que las URL's sean en minúsculas y separadas con guion bajo.

Resultado de convenciones:

Al seguir las convenciones de CakePHP, se adquiere funcionalidad sin mucho mantenimiento de la configuración.

Ejemplo de las convenciones

Tabla de Base de Datos: user		
Tipo	**Formato**	**Creado en**
Modelo	User.php	/app/Models/
Controlador	UsersController.php	/app/Controllers/
Vista	index.ctp	/app/Views/Users/

Tabla 4.7 Ejemplo de convenciones.

Elaborado por: Wilson Pérez – Investigador

Usando estas convenciones, CakePHP entiende que la peticion http://example.com/users/ apunta a la llamada de funcion index() en el controlador, UsersController, donde el modelo Users que está disponible automáticamente y apunta a la tabla users en la base de datos, y se renderiza en el archivo.

4.6.1.7. **Interfaces de comunicaciones**

La aplicación se conectara de manera cliente servido mediante un protocolo de comunicación como TCP-IP lo que es para la conexión y HTTP para el acceso al servicio del servidor por los clientes.

4.6.1.8. **Limitaciones de memoria**

Las limitaciones de momería son mínimas por el avance tecnológico en los últimos años.

4.6.1.9. **Operaciones**

4.6.1.9.1. **Modos de operación de los distintos grupos de usuarios**

Los tipos de usuarios con cuatro para los cuales se establece dos tipos de modos de operaciones:

El usuario admin tendrá también la potestad para la verificación, corrección y eliminación de cualquier información almacenada de manera involuntaria, que afecte la consistencia de los datos en la aplicación.

Los usuarios Administradores realizaran las operaciones de administración y configuración de como los usuarios interactúan con la aplicación así como también la creación de nuevos usuarios de tipo interactivo y gráfico.

Los usuarios visualizaran información de un proceso de una manera que le ayudara a tomar decisiones sobre el proceso para el cual está desarrollado la aplicación.

.

Los usuarios anónimos solo podrán visualizar la información pública que la aplicación presentara para todos los usuario en común

4.6.1.9.2. **Funciones respaldo del procesamiento de datos**

No se establece operaciones de backup en el sistema porque esta será realizada por los administradores del sistema de manera manual y la recuperación será realizada por el mismo con los procedimientos establecidos en las políticas de administración de datos de la propia entidad.

4.6.1.10. **Requerimientos para adaptarse a la ubicación**

Para esto se establece que es necesario un lugar que tenga un acceso a energía eléctrica, acceso a la red interna de la institución o entidad un equipo de cómputo en condiciones óptimas, un UPS que garantice la energía necesaria para el cierre de la aplicación de manera adecuada para evitar pérdidas de información.

4.6.1.11. **Indicar cualquier dato o secuencia de inicialización específico de cualquier lugar, modo de operación**.

Se debe establecer para la instalación de la aplicación web es que se debe de crear la base de datos en el respectivo motor de base de datos, además de la creación de los grupos de usuarios mencionados como son:

- Admin
- Administrador
- Usuario
- Anónimo

Donde el índice o clave primaria de cada uno de ellos se las realice de manera secuencial de 1 al 4, esto se debe a que el manejo en la aplicación se tiene un algoritmo que permite la navegación en la aplicación.

Además se debe de llenar las tablas de nuestra base que permite dar el acceso por elementos o vistas a la aplicación web con son los propietarios que se llenaran con los grupos anterior mente mencionados, como también establecer que propietario tiene acceso a dicho elemento de nuestra aplicación web.

4.6.1.12. **Características que deben ser modificadas para una instalación en particular**.

Para todas las instalación de nuestra aplicación web de debe de tener configurado lo que es una servidor web apache y php además de la librería que permita utilizar el motor de base de datos así como también el pdo de la misma ya que esta es la que permite la interacción MVC.

También se debe de tomar en cuenta en los sistemas operativos no propietarios la verificación de los permisos del mismo y que pertenezcan al grupo del servidor web apache.

4.6.2. **Funciones del Producto**

Las funciones de los distintos usuarios que realizaran en la aplicación web se establecen por el nivel de responsabilidad de cada uno, tomando en cuenta la actividad que realiza en el proceso que nuestra aplicación automatizara.

Funciones admin

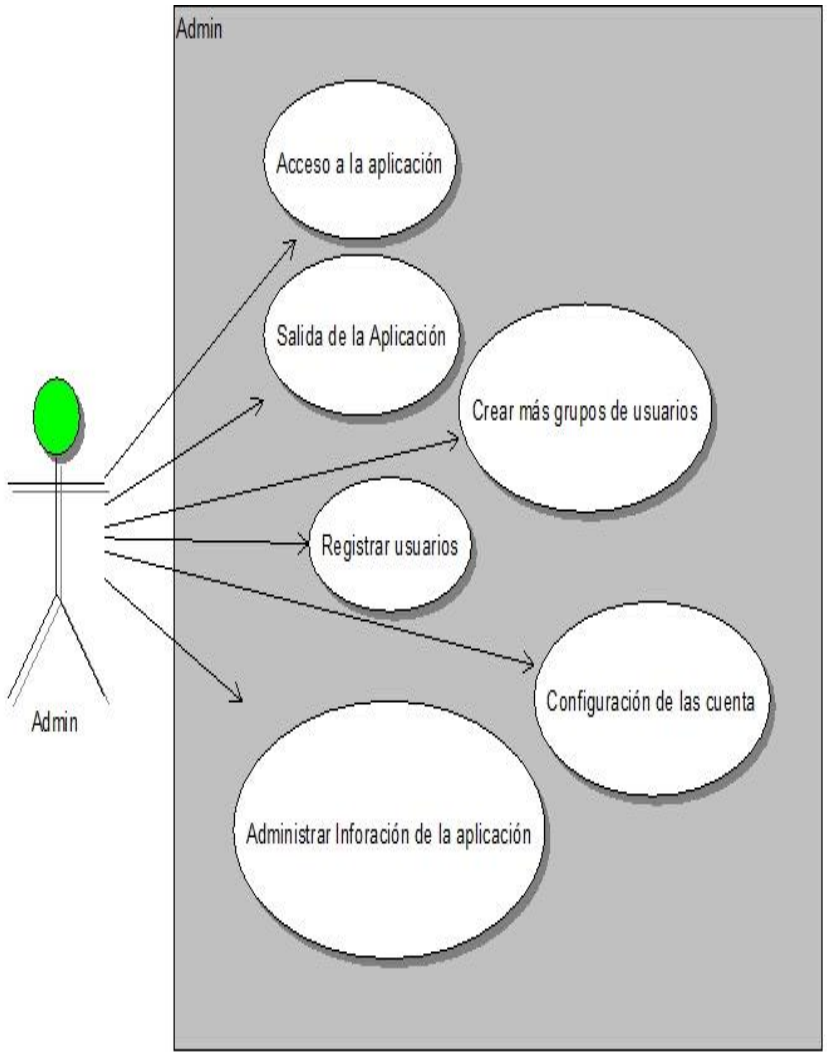

Figura 4.4.3: Diagrama de casos de uso Admin

Elaborado por: Wilson Pérez – Investigador

`

Funciones Administrador

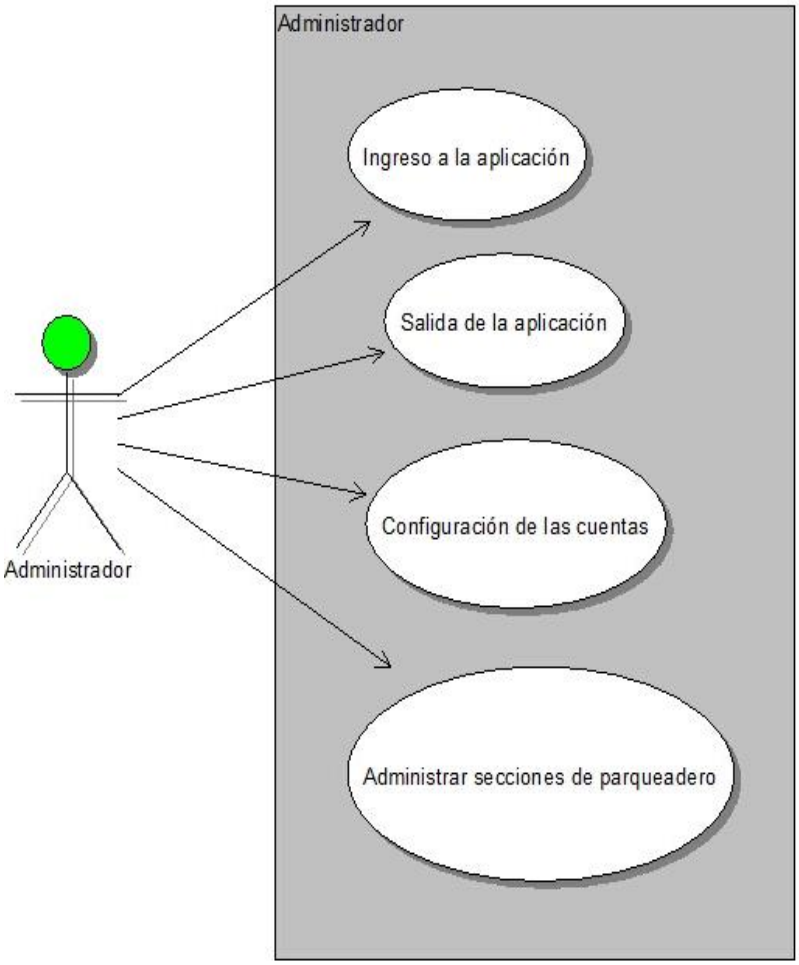

Figura 4.4: Diagrama de casos de uso Administrador

Elaborado por: Wilson Pérez – Investigador

Funciones Usuario

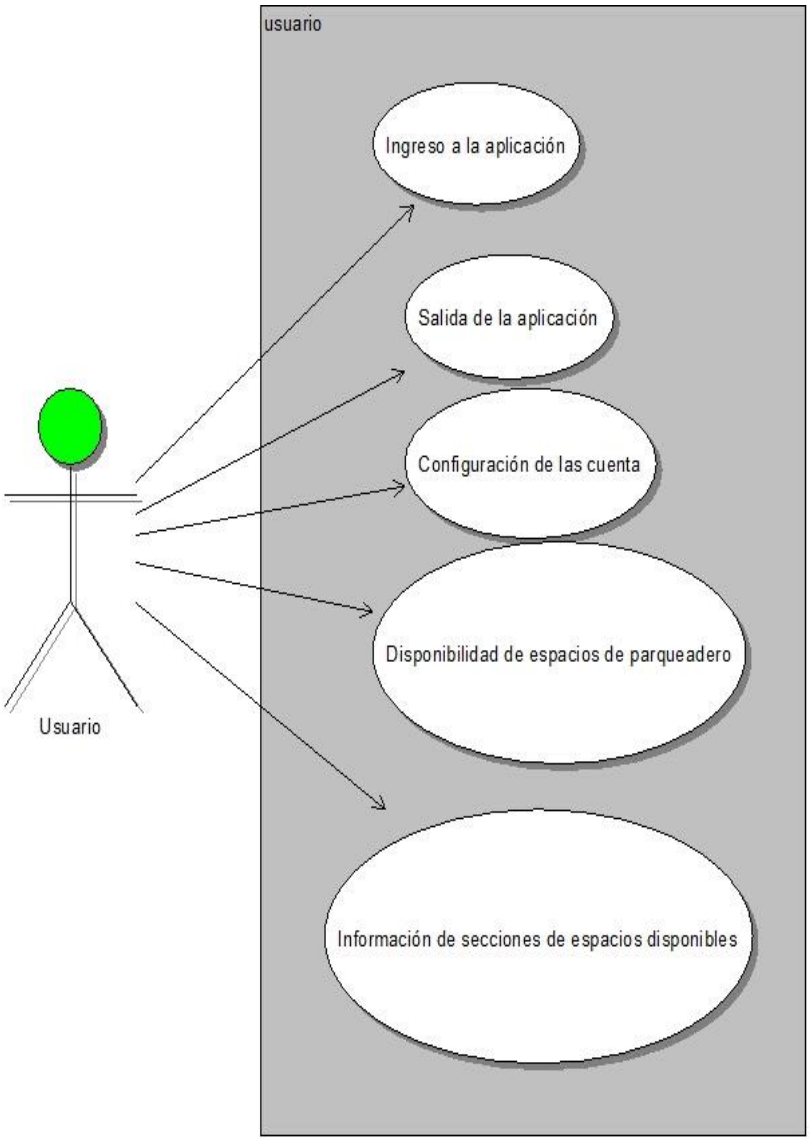

Figura 4.5: Diagrama de casos de uso Administrador

Elaborado por: Wilson Pérez – Investigador

Funciones Anonimo

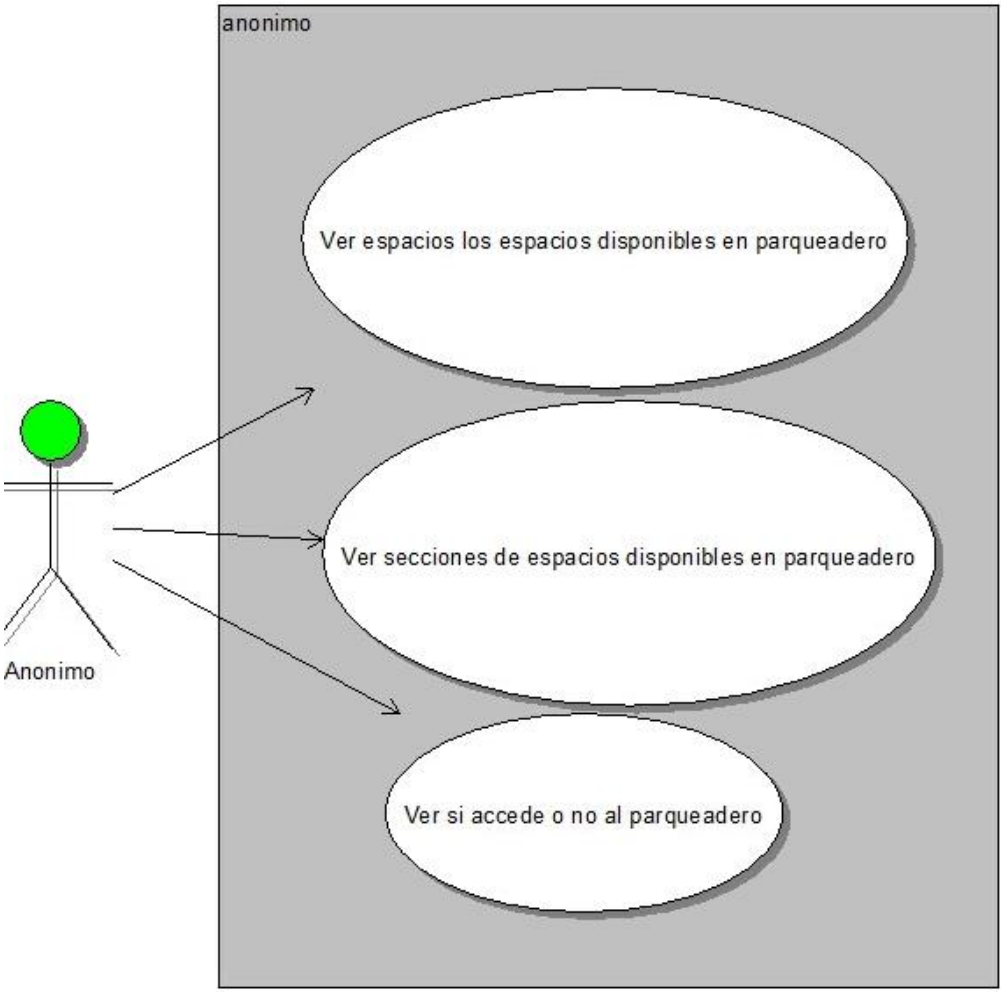

Figura 4.6: Diagrama de casos de uso usuario anónimo

Elaborado por: Wilson Pérez – Investigador

Descripción de los casos de uso

Nombre:	Ingreso Aplicación	
Actor:	Admin, Administrador, Usuario	
Descripción	Detalle de como ingresar a nuestra aplicación web.	
Flujo Principal	Evento Actor	Evento Sistema
	1. Ingreso de datos necesarios para su autentificación.	1. Muestra la pantalla de acceso al sistema de la aplicación web.
	2. Envió de datos mediante una petición web.	2. Mostrar Mensajes de validaciones de datos.
		3. Verificación de la información enviada
		4. Redireccionamiento (aplicación o nueva petición)
Precondición:	Los actores deben estar registrados como usuarios previamente, y activados para poder acceder a la aplicación.	
Poscondición:	Se presentara la interfaz al actor con un filtrado de lo que puede hacer.	
Presunción:	La base de datos de nuestra aplicación webestá disponible.	

Tabla 4.8: Descripción de caso salida aplicación

Elaborado por: Wilson Pérez – Investigador

Nombre:	Salida Aplicación	
Actor:	Admin, Administrador, Usuario	
Descripción	Detalle de cómo salir de nuestra aplicación web.	
Flujo Principal	Evento Actor	Evento Sistema
	1. Búsqueda de la opción salir.	1. Muestra la pantalla la opción de salir.
	2. Cerrar página web.	2. Redireccionamiento página de login.
Precondición:	Los actores deben tener una sección activa.	
Poscondición:	Se presentara la interfaz principal de nuestra APW.	
Presunción:	La base de datos de nuestra aplicación web está disponible.	

Tabla 4.9: Descripción de caso salida aplicación

Elaborado por: Wilson Pérez – Investigador

Nombre:	Configuración de cuenta	
Actor:	Admin, Administrador, Usuario	
Descripción	Información del actor que utiliza la APW.	
Flujo Principal	Evento Actor	Evento Sistema
	1. Búsqueda de la opción Configurar cuenta.	1. Muestra la pantalla un menú donde está la opción de la configuración de la cuenta.
	2. Editar campos a modificarse de su información.	2. Cargar la página de la configuración de la opción de la cuenta.
	3. Enviar datos mediante una petición web	3. Mostrar Mensajes de validaciones de datos.
		4. Verificación de la información enviada
		5. Redireccionamiento página de configuración cuenta.
Alternativa 1	1. Buscar opción de inicio	
	2. Envía petición web	3. Redireccionamiento página de principal de la aplicación web
Alternativa 2	1. Buscar opción de actualizar foto.	
	2. Envía petición web	2. Abrir explorador del sistema de búsqueda de archivos
		3. Cargar Archivo.
		4. Actualizar información de la fotografía de los actor
Precondición:	Los actores deben tener una sección activa.	
Poscondición:	Se presentara la información actualizada de los datos del actor que inicio la sesión.	
Presunción:	La base de datos de nuestra aplicación web está disponible.	

Tabla 4.10: Descripción de caso configuración cuenta

Elaborado por: Wilson Pérez – Investigador

Nombre:	Crear grupo de usuarios	
Actor:	Admin	
Descripción	Realizar la creación de los grupos de usuarios que intervienen en el proceso de nuestra aplicación.	
Flujo Principal	Evento Actor	Evento Sistema
	1. Búsqueda de la creación de grupos.	1. Muestra un menú donde está la opción de la administración de grupos.
	2. Seleccionamos la opción a realizar con los grupos.	2. Cargar la página de administración de grupos.
	3. Enviar datos mediante una petición web	3. Mostrar Mensajes de validaciones de datos.
		4. Verificación de la información enviada
		5. Redireccionamiento página de administración de grupos.
Alternativa 1	1. Buscar opción de inicio	
	2. Envía petición web	2. Redireccionamiento página de principal de la aplicación web
Alternativa 2	1. Buscar opción de actualizar foto.	
	2. Envía petición web	2. Abrir explorador del sistema de búsqueda de archivos
		3. Cargar Archivo.
		4. Actualizar información de la fotografía de los actor
Precondición:	El actor debe tener una sección activa.	
Poscondición:	Se presentara la información actualizada de los datos de los grupos creados.	
Presunción:	La base de datos de nuestra aplicación web está disponible.	

Tabla 4.11: Descripción de caso crear grupo de usuarios

Elaborado por: Wilson Pérez – Investigador

Nombre:	Agregar usuarios	
Actor:	Admin	
Descripción	Realizar la creación de los de usuarios que intervienen en el proceso de nuestra aplicación.	
Flujo Principal	Evento Actor	Evento Sistema
	1. Búsqueda de la creación de usuarios.	1. Muestra un menú donde está la opción de la administración de usuarios.
	2. Seleccionamos la opción a realizar con los usuarios.	2. Cargar la página de administración de grupos.
	3. Enviar datos mediante una petición web.	3. Mostrar Mensajes de validaciones de datos.
		4. Verificación de la información enviada.
		5. Redireccionamiento página de administración de usuarios.
Alternativa 1	1. Buscar opción de inicio.	
	2. Envía petición web.	2. Redireccionamiento página de principal de la aplicación web.
Alternativa 2	1. Buscar opción de actualizar foto.	
	2. Envía petición web.	2. Abrir explorador del sistema de búsqueda de archivos.
		3. Cargar Archivo.
		4. Actualizar información de la fotografía del actor.
Precondición:	El actor debe tener una sección activa.	
Poscondición:	Se presentara la información actualizada y agregara datos de los usuarios.	
Presunción:	La base de datos de nuestra aplicación web está disponible.	

Tabla 4.12: Descripción de caso agregar usuario

Elaborado por: Wilson Pérez – Investigador

Nombre:	Administrar Información de la aplicación	
Actor:	Admin, Administrador	
Descripción	Permitirá el control de toda nuestra aplicación, donde administrara la información del proceso de la aplicación en su totalidad.	
Flujo Principal	Evento Actor	Evento Sistema
	1. Búsqueda la operación a realizar.	1. Muestra un menú donde está la opción de la administración.
	2. Seleccionamos la opción a realizar.	2. Cargar las interfaces de las operaciones a realizar.
	3. Seleccionar la actividad a administrar.	3. Mostrar Interfaz de las Opciones que se puede realizar en el proceso.
	4. Enviar datos mediante una petición web.	4. Verificación de la información enviada.
		5. Actualización de la información en los objetos de visualización en la interfaz.
Alternativa 1	1. Buscar opción de inicio.	
	2. Envía petición web.	2. Redireccionamiento página de principal de la aplicación web.
Precondición:	El actor debe tener una sección activa.	
Poscondición:	Se presentara la información actualizada en la o las interfaces correspondientes que se utilizan para los distintos procesos que nuestra aplicación.	
Presunción:	La base de datos de nuestra aplicación web está disponible.	

Tabla 4.13: Administrar Información de la aplicación

Elaborado por: Wilson Pérez – Investigador

Nombre:	Administrar secciones de parqueadero.	
Actor:	Administrador	
Descripción	Permitirá el control de todas las secciones de los parqueaderos de la institución.	
Flujo Principal	Evento Actor	Evento Sistema
	1. Búsqueda de la operación a realizar.	1. Muestra un menú donde está la opción de las secciones de parque.
	2. Seleccionamos Administrar secciones de parqueadero.	2. Cargar las interfaces de las operaciones a realizar.
	3. Seleccionar la actividad a administrar.	3. Mostrar Opciones que se puede realizar con las secciones de parqueadero.
	4. Enviar datos mediante una petición web	5. Verificación de la información enviada.
		6. Actualización de la información en los objetos de visualización en la interfaz.
Alternativa 1	1. Buscar opción de inicio.	
	2. Envía petición web.	1. Redireccionamiento página de principal de la aplicación web.
Precondición:	El actor debe tener una sección activa.	
Poscondición:	Se presentara la interfaz para verificación de la acción realizada.	
Presunción:	La base de datos de nuestra aplicación web está disponible.	

Tabla 4.14: Administración de tarifas de cobro

Elaborado por: Wilson Pérez – Investigador

Nombre:	Disponibilidad de espacios de parqueadero.	
Actor:	Usuario	
Descripción	Permitirá la visualización de todos los espacios disponibles de cada uno de las secciones y espacios ocupados como disponibles de parqueadero.	
Flujo Principal	Evento Actor	Evento Sistema
	1. Búsqueda de la operación a realizar.	1. Muestra un menú donde está la opción de tarjetas magnéticas.
	2. Seleccionamos ver espacios disponibles de parqueadero.	2. Cargar las interfaces de la operación a realizar.
	3. Seleccionar la sección de parqueadero.	3. Mostrar todos los espacios disponibles y ocupados en la sección seleccionada.
		4. Actualización de la información en los objetos de visualización en la interfaz.
Alternativa 1	1. Buscar opción de inicio.	
	2. Envía petición web.	2. Redireccionamiento página de principal de la aplicación web.
Precondición:	El actor debe tener una sección activa.	
Poscondición:	Se presentara la interfaz actualizada para la verificación de la acción realizada.	
Presunción:	La base de datos de nuestra aplicación web está disponible.	

Tabla 4.15: Disponibilidad de espacios de parqueadero.

Elaborado por: Wilson Pérez – Investigador

Nombre:	Informaciones de secciones de espacios disponibles.	
Actor:	Usuario	
Descripción	Permitirá ver que secciones tienen aún espacios disponibles.	
Flujo Principal	Evento Actor	Evento Sistema
	1. Búsqueda de la operación a realizar.	1. Muestra la operación a realizarse.
	2. Enviar una petición web.	2. Mostar la interfaz con las secciones que tienen espacios disponibles.
		3. Actualización de la información en los objetos de visualización en la interfaz.
Alternativa 1	1. Buscar opción de inicio.	
	2. Envía petición web.	2. Redireccionamiento página de principal de la aplicación web.
Precondición:	El actor debe tener una sección activa.	
Poscondición:	Se presentara la interfaz actualizada para la verificación de la acción realizada.	
Presunción:	La base de datos de nuestra aplicación web está disponible.	

Tabla 4.16: Información de secciones de espacios disponibles.

Elaborado por: Wilson Pérez – Investigador

Nombre:	Ver espacios disponibles de parqueadero.	
Actor:	Anonimo	
Descripción	Permitirá mostrar que secciones tienen aún espacios disponibles.	
Flujo Principal	Evento Actor	Evento Sistema
		1. Mostar la interfaz con información de espacios disponibles por secciones.
Precondición:	El actor debe tener una sección activa.	
Poscondición:	Presentará la interfaz actualizada para la verificación de la acción realizada.	
Presunción:	La base de datos de nuestra aplicación web está disponible.	

Tabla 4.17: Ver espacios disponibles de parqueadero.

Elaborado por: Wilson Pérez – Investigador

Nombre:	Ver secciones disponibles de parqueadero.	
Actor:	Anonimo	
Descripción	Permitirá todas las secciones tienen espacios disponibles.	
Flujo Principal	Evento Actor	Evento Sistema
		1. Mostar la interfaz con información de secciones con espacios disponibles por secciones.
Precondición:	El actor debe tener una sección activa.	
Poscondición:	Presentará la interfaz actualizada para la verificación de la acción realizada.	
Presunción:	La base de datos de nuestra aplicación web está disponible.	

Tabla 4.18: Ver espacios disponibles de parqueadero.

Elaborado por: Wilson Pérez – Investigador

Nombre:	Ver si accede o no al parqueadero.	
Actor:	Anonimo	
Descripción	Permitirá visualizar si puede o no ingresar al parqueadero de la institución.	
Flujo Principal	Evento Actor	Evento Sistema
		1. Mostar la si el parqueadero está lleno o si existe espacios disponibles.
Precondición:	El actor debe tener una sección activa.	
Poscondición:	Presentará la interfaz actualizada para la verificación de la acción realizada.	
Presunción:	La base de datos de nuestra aplicación web está disponible.	

Tabla 4.19: Ver si accede o no al parqueadero.

Elaborado por: Wilson Pérez – Investigador

Diagrama de secuencias MVC

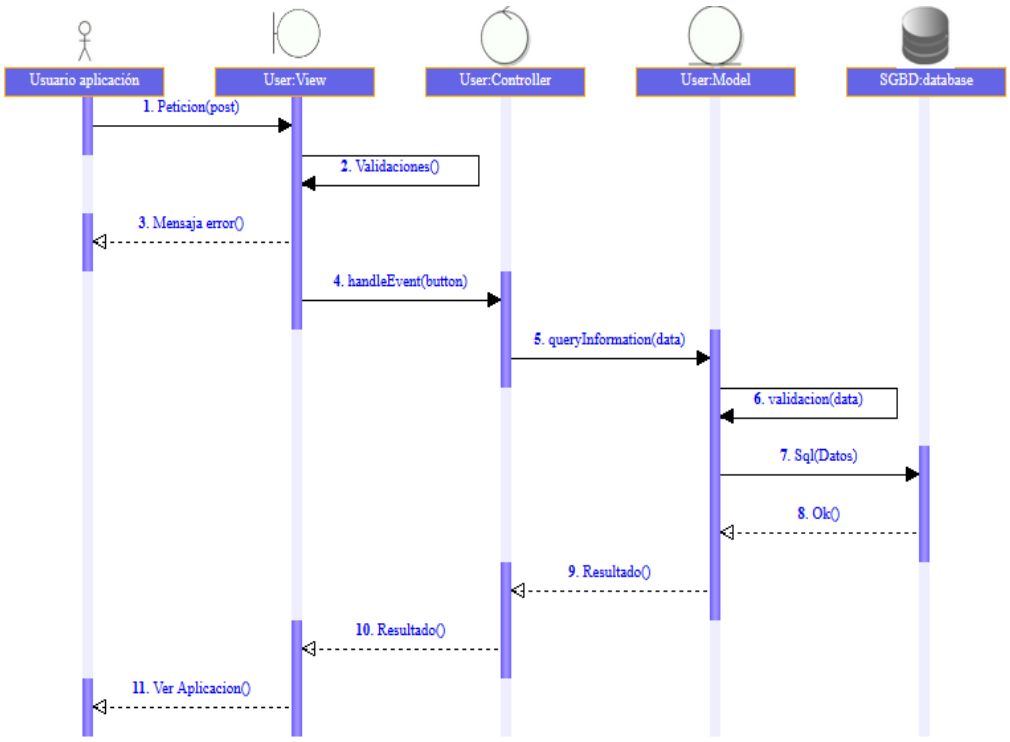

Figura 4.7: Diagrama de secuencias MVC ingreso a la aplicación.

Elaborado por: Wilson Pérez – Investigador

Descripción
1. Usuario aplicación ase una llamada a User:View->Peticion(post)
2. User:View ase una llamada a User:View->Validaciones()
3. Usuario aplicación recibe de User:View->Mensaja error()
4. User:View ase una llamada a User:Controller->handleEvent(button)
5. User:Controller ase una llamada a User:Model->queryInformation(data)
6. User:Model ase una llamada a User:Model->validacion(data)
7. User:Model ase una llamada a SGBD:database->Sql(Datos)
8. User:Model recibe de SGBD:database->Ok()
9. User:Controller recibe de User:Model->Resultado()
10. User:View recibe de User:Controller->Resultado()
11. Usuario aplicación recibe de User:View->Ver Aplicacion()

Tabla 4.20: Descripción MVC ingreso a la aplicación.

Elaborado por: Wilson Pérez – Investigador

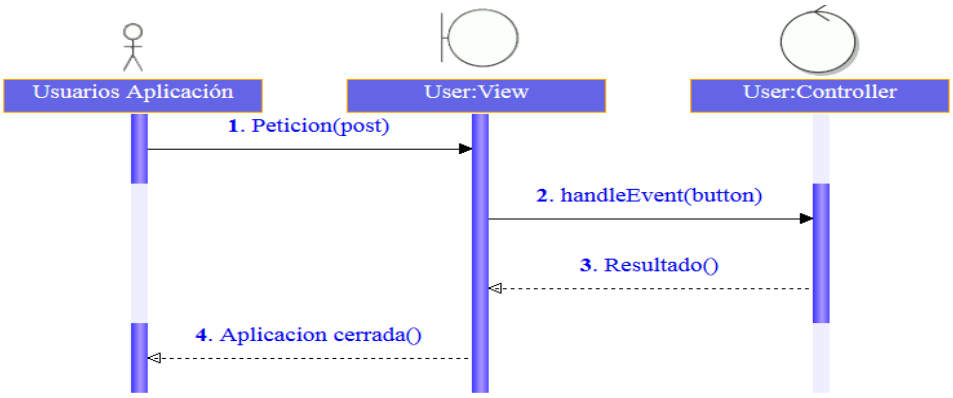

Figura 4.8: Diagrama de secuencias MVC salida a la aplicación.

Elaborado por: Wilson Pérez – Investigador

Descripción
1. Usuario aplicación ase una llamada a User:View->Peticion(post)
2. User:View ase una llamada a User:Controller->handleEvent(button)
3. User:View recibe de User:Controller->Resultado()
4. Usuario aplicación recibe de User:View->Aplicacion cerrada()

Tabla 4.21: Descripción MVC salida a la aplicación.

Elaborado por: Wilson Pérez – Investigador

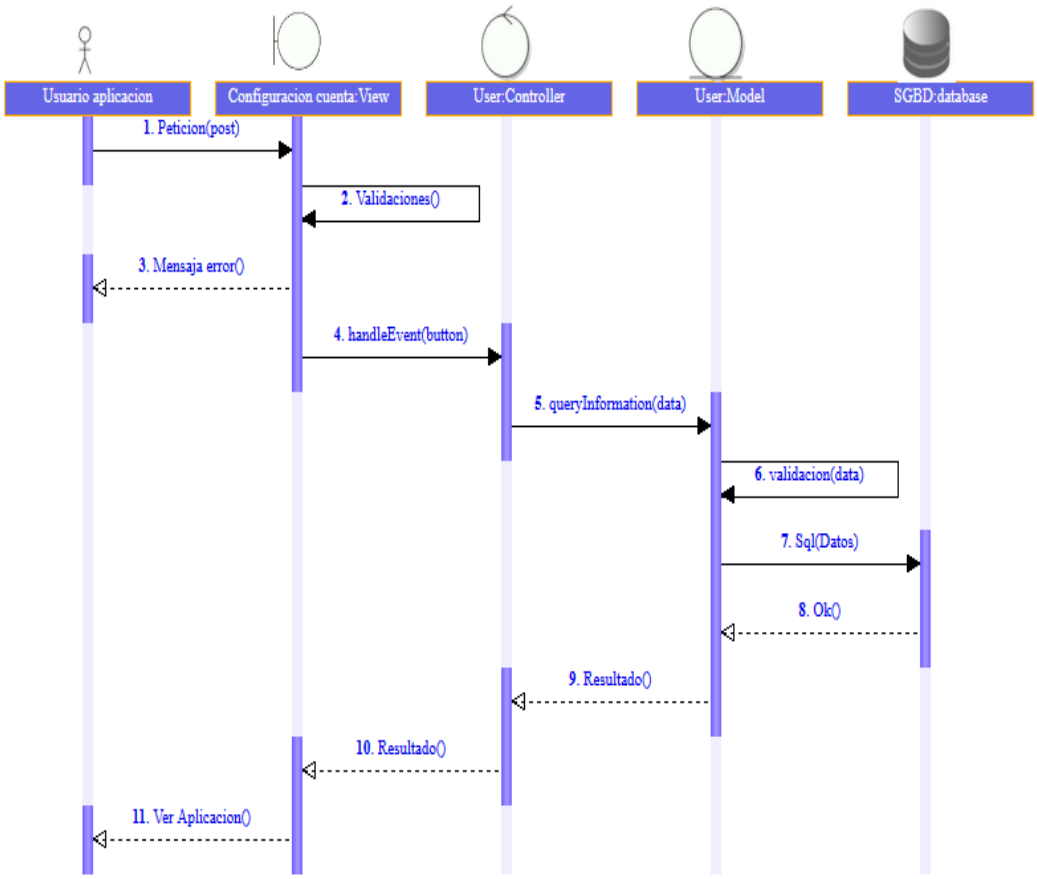

Figura 4.9 : Diagrama de secuencias MVC configuración cuenta.

Elaborado por: Wilson Pérez – Investigador

Descripción
5. Usuario aplicación ase una llamada a Configuración cuenta:View->Peticion(post)
6. Configuración cuenta:View ase una llamada a Configuración cuenta:View->Validaciones()
7. Usuario aplicación recibe de Configuración cuenta:View->Mensaja error()
8. Configuración cuenta:View ase una llamada a User:Controller->handleEvent(button)
9. User:Controller ase una llamada a User:Model->queryInformation(data)
10. User:Model ase una llamada a User:Model->validacion(data)
11. User:Model ase una llamada a SGBD:database->Sql(Datos)
12. User:Model recibe de SGBD:database->Ok()
13. User:Controller recibe de User:Model->Resultado()
14. Configuración cuenta:View recibe de User:Controller->Resultado()
15. Usuario aplicacion recibe de Configuracion cuenta:View->Ver Aplicacion()

Tabla 4.22: Descripción MVC configuración cuenta.

Elaborado por: Wilson Pérez – Investigador

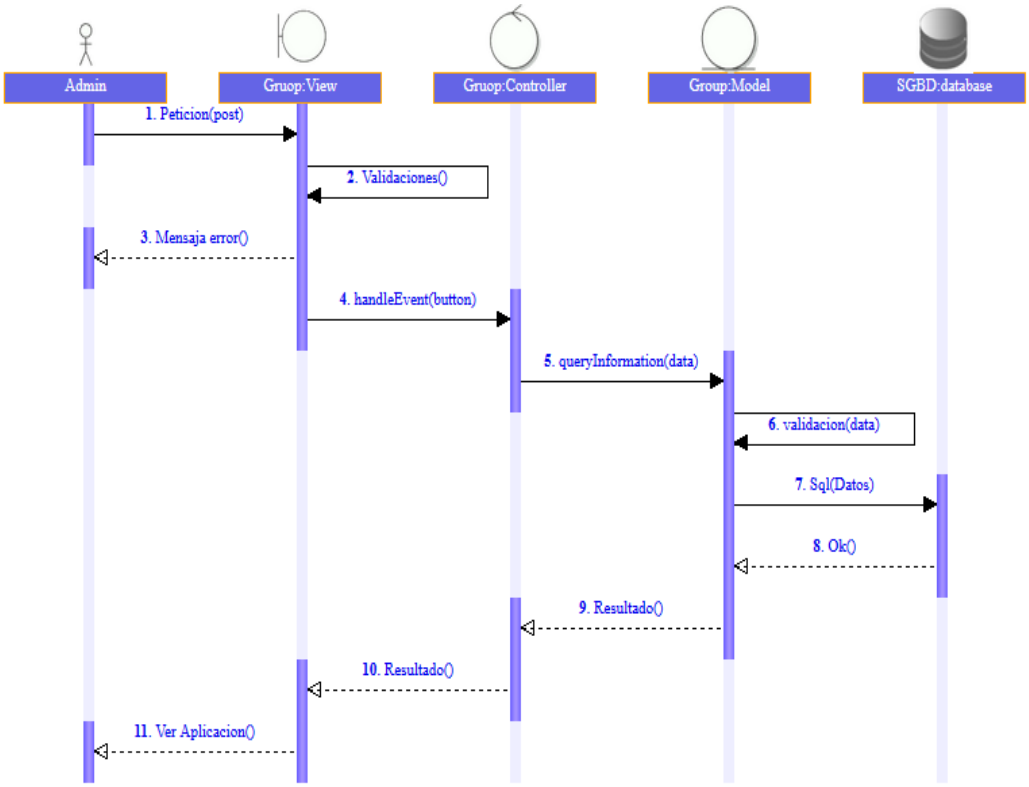

Figura 4.10: Diagrama de secuencias MVC agregar crear grupos de usuarios.

Elaborado por: Wilson Pérez – Investigador

Descripción
1. Admin ase una petición ha Gruop:View->Petición(post)
2. Gruop:View realiza una validación Gruop:View->Validaciones()
3. Admin recibe una mensaje Gruop:View->Mensaje error()
4. Gruop:View hase una petición ha Gruop:Controller->handleEvent(button)
5. Gruop:Controller envía información Group:Model->queryInformation(data)
6. Group:Model realiza una validación de datos en Group:Model->validacion(data)
7. Group:Model envía SGBD:database->Sql(Datos)

8. Group:Model devuelve de SGBD:database->Ok()
9. Gruop:Controller recibe de Group:Model->Resultado()
10. Gruop:View recibe de Gruop:Controller->Resultado()
11. Admin recibe de Gruop:View->Ver Aplicacion()

Tabla 4.23: Descripción MVC crear más grupos de usuarios.

Elaborado por: Wilson Pérez – Investigador

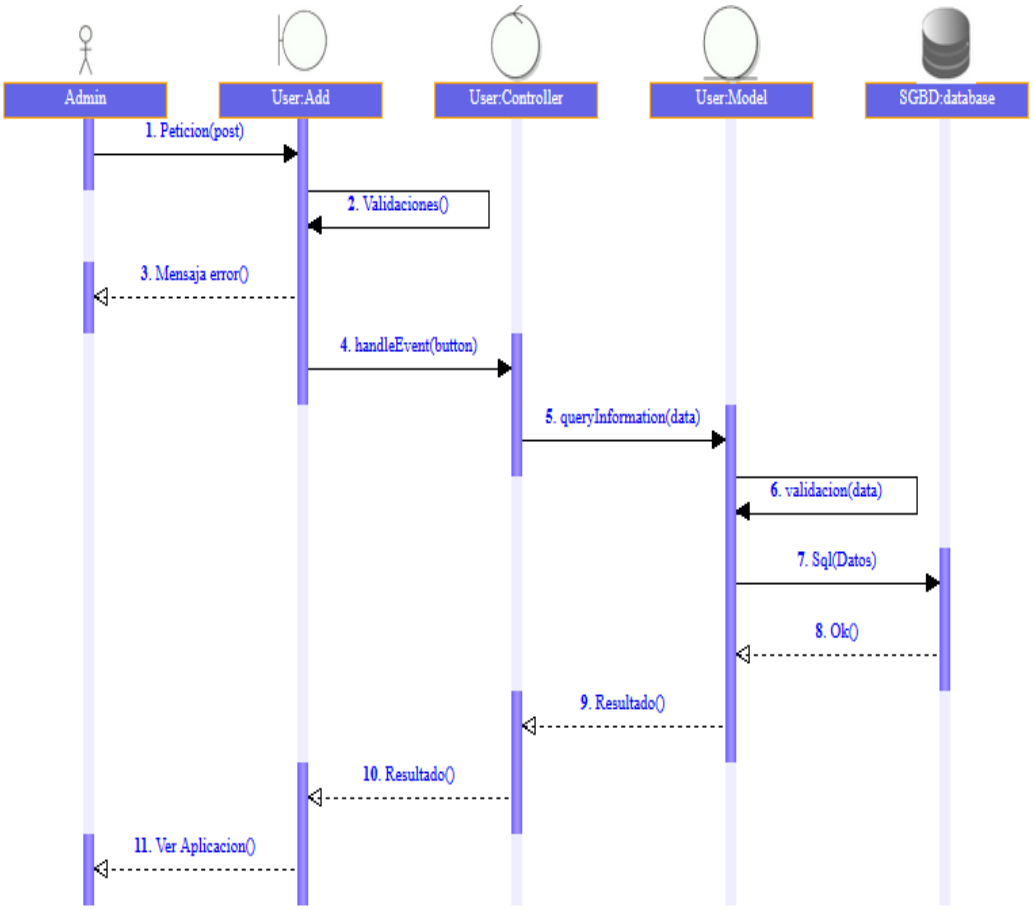

Figura 4.11: Diagrama de secuencias MVC agregar usuarios.

Elaborado por: Wilson Pérez – Investigador

Descripción
1. Admin petición User:Add->Petición(post)
2. User:Add realiza una llamada validación User:Add->Validaciones()
3. Admin recibe un una respuesta de User:Add->Mensaja error()
4. User:Add envía una petición ha User:Controller->handleEvent(button)
5. User:Controller envía la información ha User:Model->queryInformation(data)
6. User:Model hace validaciones en User:Model->validacion(data)
7. User:Model envía a SGBD:database->Sql(Datos)
8. User:Model recibe de SGBD:database->Ok()
9. User:Controller recibe de User:Model->Resultado()
10. User:Add recibe de User:Controller->Resultado()
11. Admin recibe de User:Add->Ver Aplicacion()

Tabla 4.24: Descripción MVC agregar usuarios.

Elaborado por: Wilson Pérez – Investigador

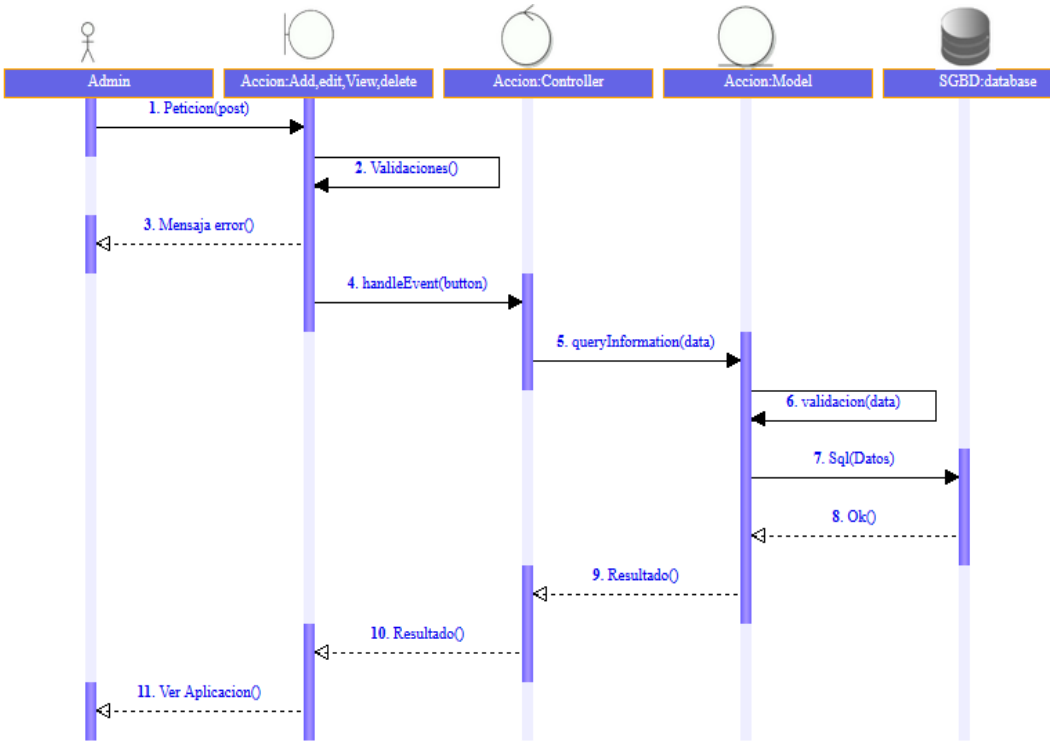

Figura 4.12: Diagrama de secuencias MVC Administrar información de la aplicación.

Elaborado por: Wilson Pérez – Investigador

Descripción
1. Admin llamada a la Acción:Add,edit,View,delete->Peticion(post) de la acción a realizarse.
2. Acción:Add,edit,View,delete llamada a la acción Acción:Add,edit,View,delete->Validaciones()
3. Admin recibe la información de la Acción:Add,edit,View,delete->Mensaja error()
4. Acción:Add,edit,View,delete realiza una petición a las Acción:Controller->handleEvent(button)
5. Acción:Controller envía a la Accion:Model->queryInformation(data)
6. Acción:Model realiza la verificación en la Accion:Model->validacion(data)
7. Acción:Model envía al SGBD:database->Sql(Datos)

8. Acción:Model recibe de SGBD:database->Ok()
9. Acción:Controller recibe de Accion:Model->Resultado()
10. Acción:Add,edit,View,delete recibe de Accion:Controller->Resultado()
11. Admin recibe de Accion:Add,edit,View,delete->Ver Aplicacion()

Tabla 4.25: Descripción MVC Administrar información de la aplicación.

Elaborado por: Wilson Pérez – Investigador

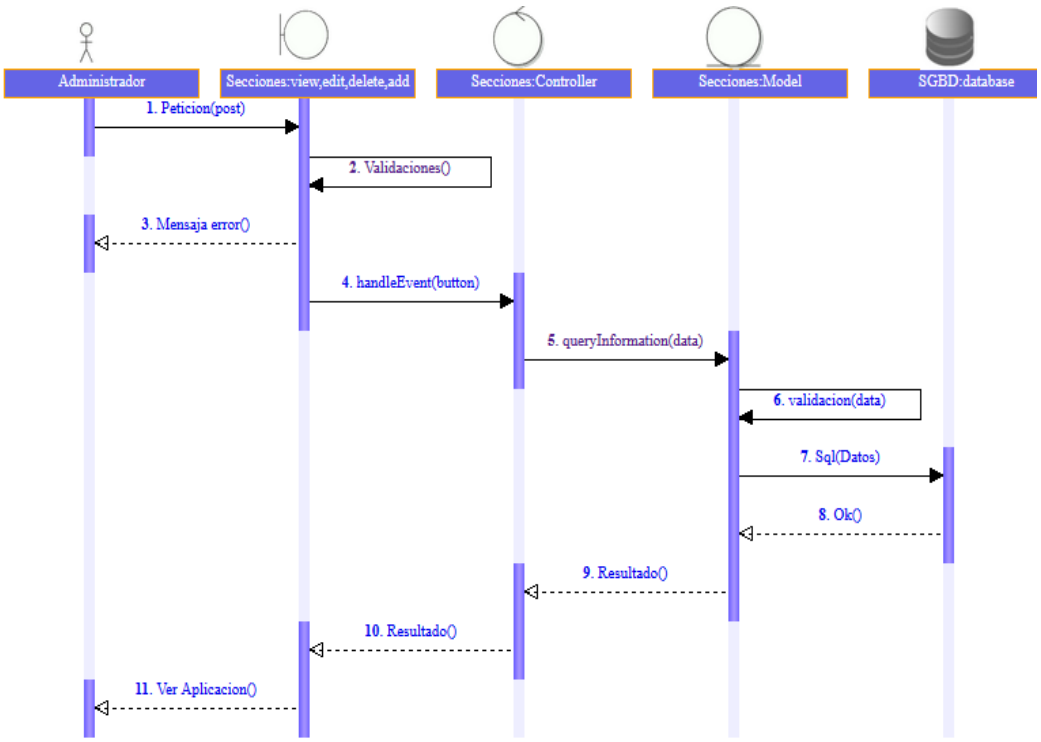

Figura 4.13: Diagrama de secuencias MVC Administrar secciones de parqueadero.

Elaborado por: Wilson Pérez – Investigador

Descripción
1. Administrador realiza una llamada a Secciones:View,edit,delete,add->Peticion(post)
2. Secciones:View,edit,delete,add llama a si mismo Secciones:View,edit,delete,add->Validaciones()
3. Administrador recibe de Secciones:View,edit,delete,add->Mensaja error()
4. Secciones:View,edit,delete,add envía una llamada mediante Secciones:Controller->handleEvent(button)
5. Secciones:Controller envía la información Secciones:Model->queryInformation(data)
6. Secciones:Model realiza una llamada a Secciones:Model->validacion(data)
7. Secciones:Model envía al SGBD:database->Sql(Datos)
8. Secciones:Model recibe de SGBD:database->Ok()
9. Secciones:Controller recibe de Secciones:Model->Resultado()
10. Secciones:View,edit,delete,add recibe de Secciones:Controller->Resultado()
11. Administrador recibe de Secciones:View,edit,delete,add->Ver Aplicacion()

Tabla 4.26: Descripción MVC Administrar secciones de parqueadero.

Elaborado por: Wilson Pérez – Investigador

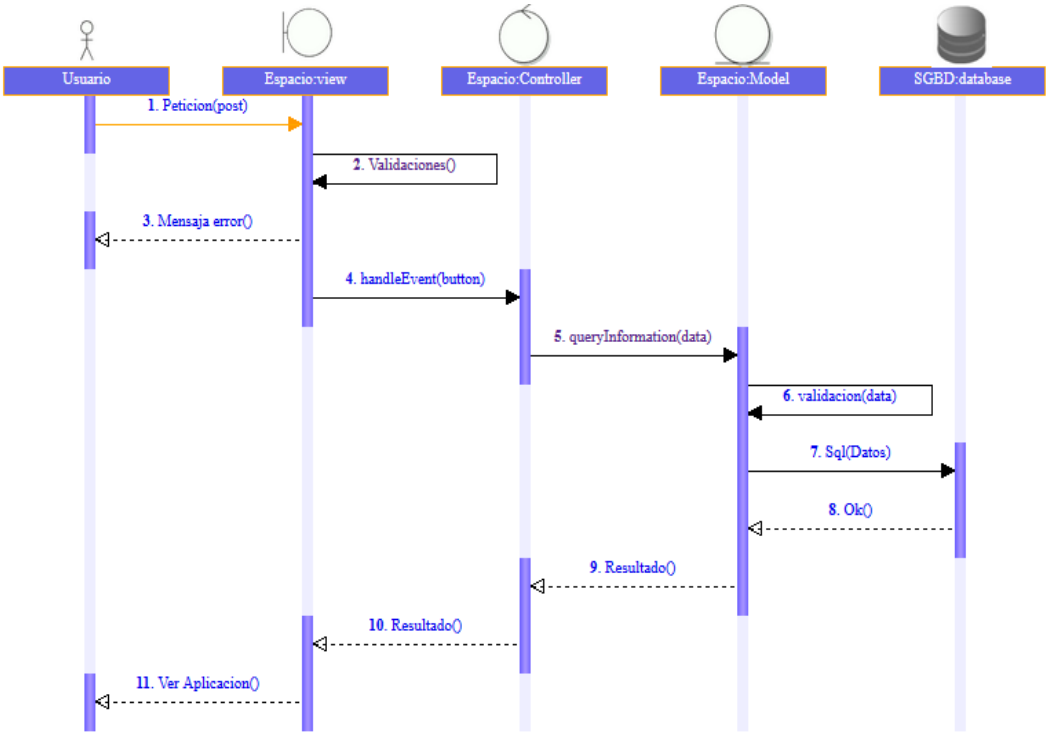

Figura 4.14: Diagrama de secuencias MVC Disponibilidad de espacios de parqueadero.

Elaborado por: Wilson Pérez – Investigador

Descripción
1. Usuario realiza una llamada a Espacios:View->Peticion(post)
2. Espacios:View llama a si mismo Espacios:View->Validaciones()
3. Usuario recibe de Espacios:View->Mensaja error()
4. Espacios:View envía una llamada mediante Espacios:Controller->handleEvent(button)
5. Espacios:Controller envía la información Espacios:Model->queryInformation(data)
6. Espacios:Model realiza una llamada a Espacios:Model->validacion(data)
7. Espacios:Model envía al SGBD:database->Sql(Datos)
8. Espacios:Model recibe de SGBD:database->Ok()
9. Espacios:Controller recibe de Espacios:Model->Resultado()
10. Espacios:View recibe de Espacios:Controller->Resultado()
11. Usuario recibe de Espacios:View->Ver Aplicacion()

Tabla 4.27: Descripción MVC Disponibilidad de espacios de parqueadero.

Elaborado por: Wilson Pérez – Investigador

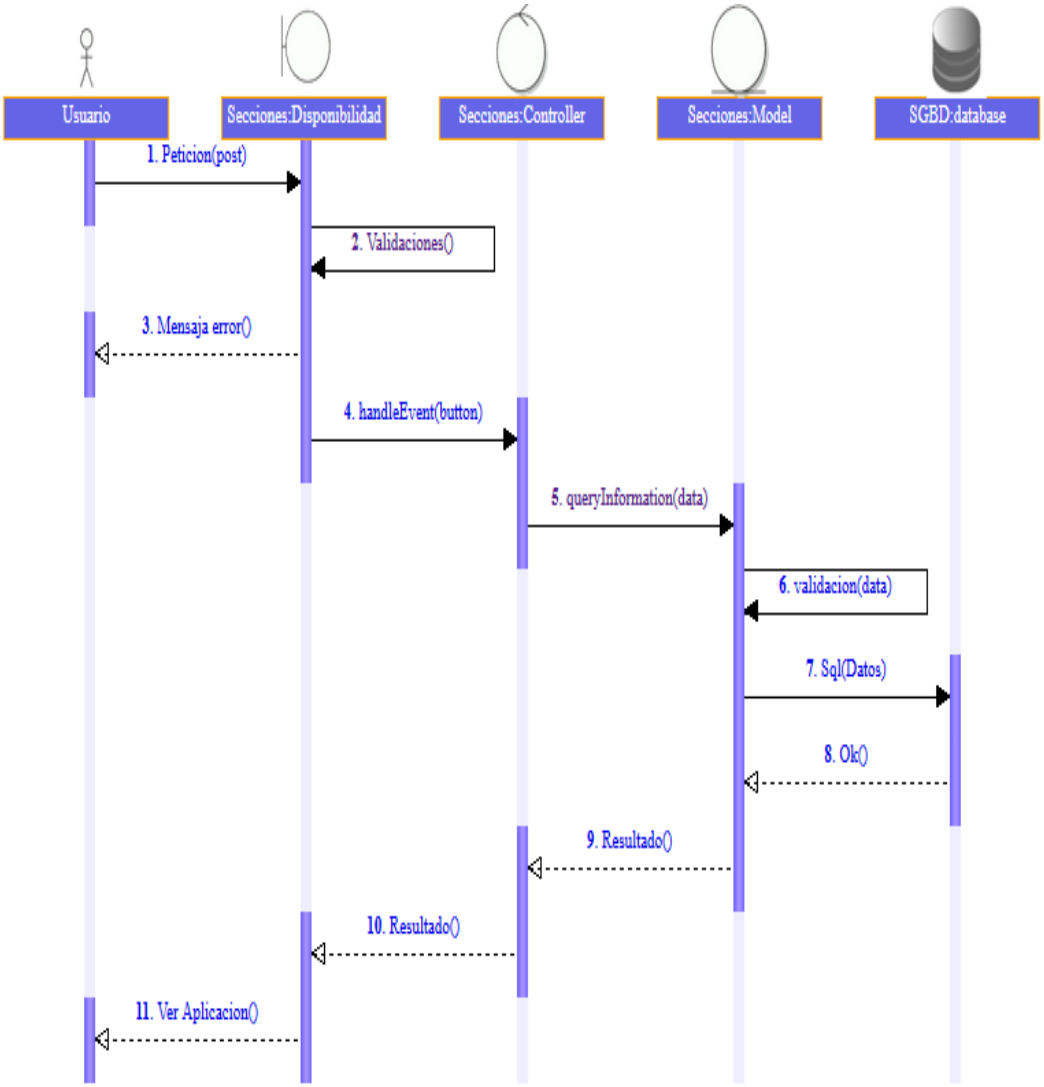

Figura 4.15: Diagrama de secuencias MVC Información de secciones de espacios disponibles.

Elaborado por: Wilson Pérez – Investigador

Descripción
1. Usuario realiza una llamada a Secciones:Disponiblidad->Peticion(post)
2. Secciones:Disponiblidad llama a si mismo Secciones:Disponiblidad->Validaciones()
3. Usuario recibe de Secciones:Disponiblidad->Mensaja error()
4. Secciones:Disponiblidad envía una llamada mediante Secciones:Seccionesler->handleEvent(button)
5. Secciones:Seccionesler envía la información Secciones:Model->queryInformation(data)
6. Secciones:Model realiza una llamada a Secciones:Model->validacion(data)
7. Secciones:Model envía al SGBD:database->Sql(Datos)
8. Secciones:Model recibe de SGBD:database->Ok()
9. Secciones:Seccionesler recibe de Secciones:Model->Resultado()
10. Secciones:Disponiblidad recibe de Secciones:Seccionesler->Resultado()
11. Usuario recibe de Secciones:Disponiblidad->Ver Aplicacion()

Tabla 4.28: Descripción MVC Información de secciones de espacios disponibles.

Elaborado por: Wilson Pérez – Investigador

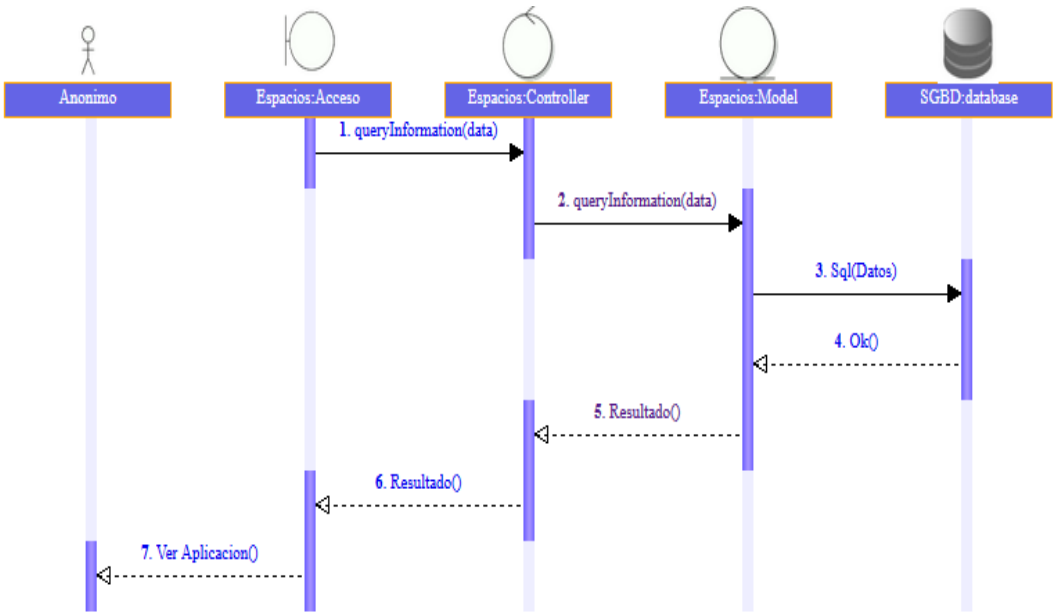

Figura 4.16: Diagrama de secuencias MVC Ver si accede o no al parqueadero.

Elaborado por: Wilson Pérez – Investigador

Descripción
1. Espacios:Acceso llama a Espacios:Controller->queryInformation(data)
2. Espacios:Controller llama a Espacios:Model->queryInformation(data)
3. Espacios:Model llama a SGBD:database->Sql(Datos)
4. Espacios:Model recibe de SGBD:database->Ok()
5. Espacios:Controller recibe de Espacios:Model->Resultado()
6. Espacios:Acceso recibe de Espacios:Controller->Resultado()
7. Anonimo recibe de Espacios:Acceso->Ver Aplicacion()

Tabla 4.29: Descripción MVC Ver si accede o no al parqueadero.

Elaborado por: Wilson Pérez – Investigador

4.6.2.1. **Características de los usuarios**

La aplicación va dirigida a personal administrativo de la institución y más concretamente al área de tecnologías. Por tanto, se encamina a un usuario con alto nivel de conocimientos y de un grado de experiencia alto para poder extraer toda la utilidad a nuestro sistema.

Además también está orientado a los guardias de seguridad que tienen una amplia experiencia ya ellos manejan la parte de cobros de acceso y uso del parqueadero de la institución.

La Aplicación también está orientada al usuario inexperto que es que que solo recibe información más nunca realiza alguna interacción con el sistema.

4.6.2.2. **Restricciones**

El sistema está basado en plataforma web por lo cual no poder el cual solo podrá ser utilizada en plataformas que soporten la mismas características.

La aplicación también solo está basado el proceso de control de usos de parqueaderos de la institución antes mencionada como fuente del proyecto y por lo tanto deberán realizar modificaciones si se quiere adaptar para cualquier otra institución, o por lo menos tener semejanza con los procesos que se manejan en la institución para la cual se realizó la aplicación

4.6.2.3. **Suposiciones y Dependencias**

Se realiza la aplicación para una plataforma web, No existe requerirnos de tiempo de respuesta ni limitaciones de memoria.

4.6.2.4. **Requisitos Futuros**

Se omite para futuras versiones la construcción de un interfaz gráfico completo que permite realizar trabajo interactivo con la aplicación en entornos móviles.

De la misma manera, se sugiere la utilización de un sistema adyacente u otras técnicas apropiadas para la toma de decisiones del proceso.

4.7. **Requisitos Específicos**

4.7.1. **Interfaces Externas**

Especificaremos aquellos requisitos que intervienen en el proceso de desarrollo del software como la interfaz le gustaría al usuario, la interfaz del hardware y que entorno el sistema necesita.

4.7.1.1. **Interfaz de usuarios**

Con el diseño de la interfaz gráfica de usuario se especifica cómo se presentara la información y los controles utilizados para cada una de los roles o grupos de usuarios que utilizaran la aplicación web, dando una vista previa de las mismas antes de ser implementadas con una herramienta de desarrollo.

4.7.1.2. **Interfaz de inicio de sesión**

La interfaz de ingreso a la aplicación se la realizara mediante un usuario y contraseña como se muestra en la siguiente figura.

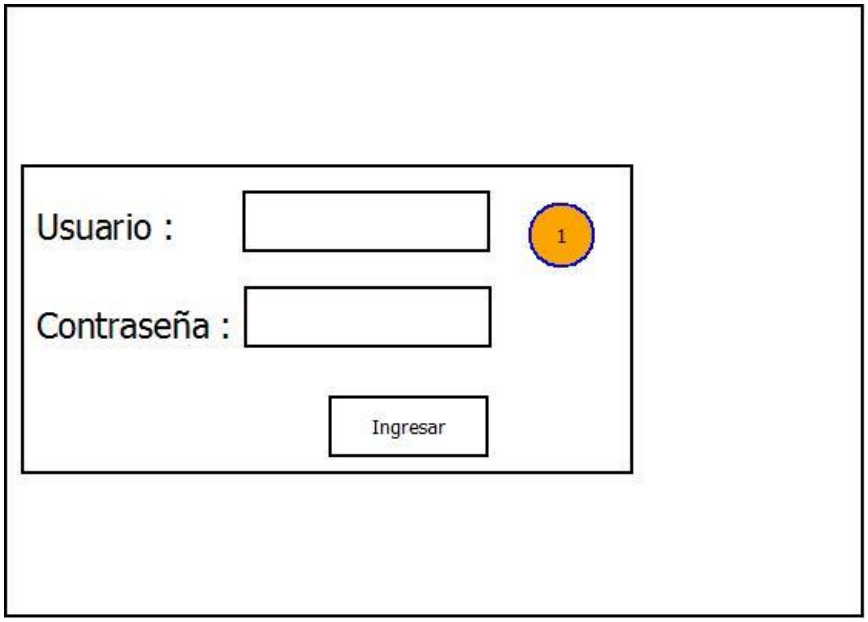

Figura 4.17: Interfaz de inicio de sesión.

Elaborado por: Wilson Pérez – Investigador

1. **Área de inicio de sesión**: Esta área permite a los diferentes usuarios autentificarse en la aplicación web, mediante el ingreso del nombre de usuario y la contraseña.

4.7.1.3. **Interfaz página principal de la aplicación web**

Ventana principal con todas las opción que presentara al usuario que inicio sesión dependiendo del tipo de rol desempeñado por el usuario que inicio sesión.

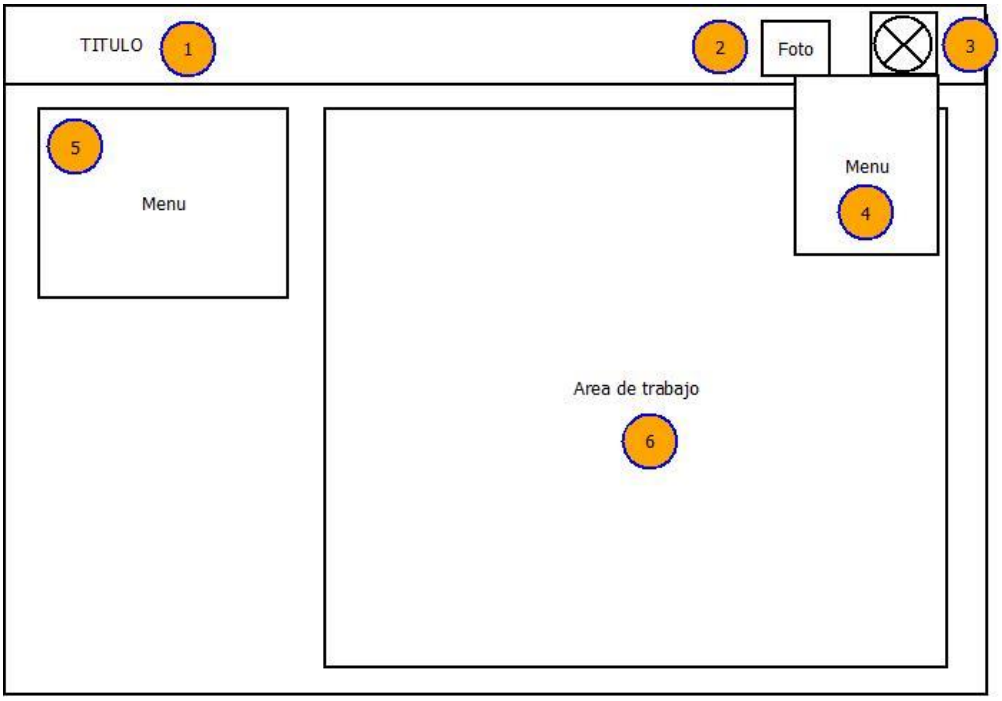

Figura 4.18: Interfaz página principal de la aplicación web

Elaborado por: Wilson Pérez – Investigador

1. **Título de la institución o logo del mismo**: en esta are ira el nombre de la institución o el logotipo que identifique el mismo.

2. **Foto del usuario**: en esta sección ira la foto del usuario que ha iniciado la sesión en la aplicación.

3. **Configuración de usuarios**: en esta sección se despliega un menú con opciones de salida y configuración de las opciones del usuario que inicio sesión en la aplicación.

4. **Menú Configuración**: en esta sección se despliegan opciones del menú de configuración de la aplicación así como ayuda e información del mismo.

5. **Menú Principal:** en esta sección están las opciones a realizar en la aplicación con las diferentes acciones permitidas por el rol que desempeñe el usuario que ha iniciado sesión en la aplicación.

6. **Área de trabajo**: en esta sección se presenta y realiza las opciones de manipulación de los procesos realizados por el proceso de que realiza la aplicación.

4.7.1.4. **Interfaz de grupos**

Ventana de administración de grupos con todas las opción que presentara al usuario que inicio sesión con los permisos necesarios para realizar la operación.

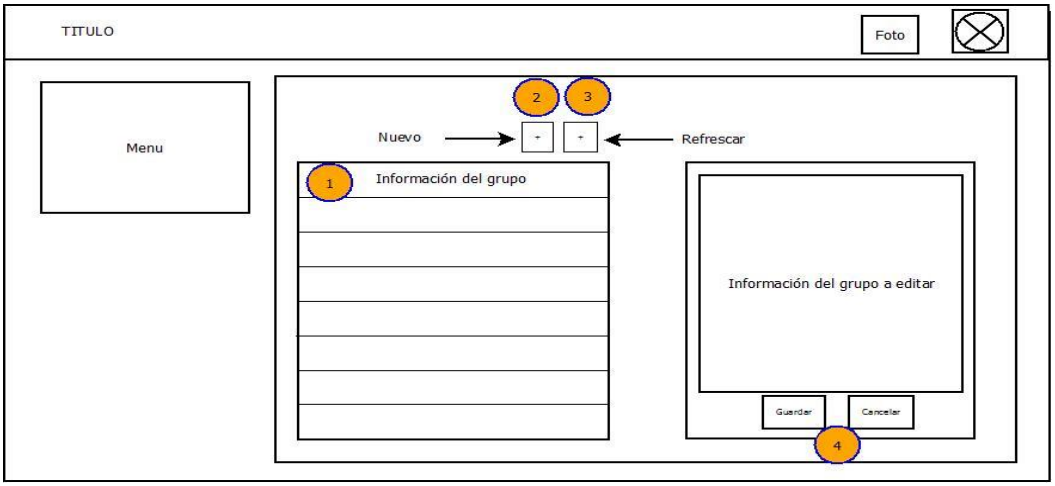

Figura 4.19: Interfaz de grupos de la aplicación web

Elaborado por: Wilson Pérez – Investigador

1. **Información del grupo**: en esta sección se presenta y realiza las opciones de la manipulación de la información de los diferentes grupos de usuarios.

2. **Nuevo:** En esta opción no permitirá agregar grupos a la aplicación dependiendo de si tiene o no permisos para realizar la opción.

3. **Refrescar**: esta opción permitirá refrescar la información de los grupos.

4. **Tratamiento de la información**: en esta sección se presentara la información del grupo seleccionado para la manipulación de la información del mismo.

4.7.1.5. **Interfaz usuarios**

Ventana de administración de usuarios con todas las opción que presentara al inicio sesión con los permisos necesarios para realizar la operación.

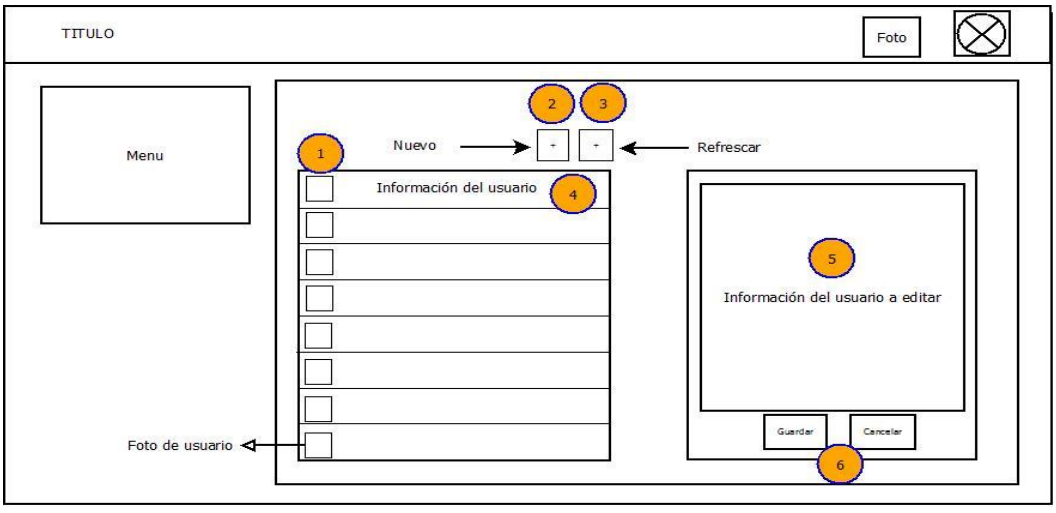

Figura 4.20: Interfaz de usuarios de la aplicación web

Elaborado por: Wilson Pérez – Investigador

1. **Foto**: en esta sección se mostrara la foto de los diferentes usuarios que manipulan la aplicación web.

2. **Información de usuarios**: en esta sección se presenta y realiza las opciones de la manipulación de la información de los diferentes usuarios.

3. **Nuevo:** En esta opción no permitirá agregar usuarios a la aplicación dependiendo de si tiene o no permisos para realizar la opción.

4. **Refrescar**: esta opción permitirá refrescar la información de los usuarios.

5. **Tratamiento de la información**: en esta sección se presentara la información del grupo seleccionado para la manipulación de la información del mismo.

6. **Opciones de información**: en esta sección estarán las diferentes opciones que se podrá realizar con los usuarios que se está administrando la información.

4.7.1.6. **Interfaz del tratamiento de la información**

Ventana de tratamiento general para todas las operaciones y opción que presentara al usuario que inicio sesión con los permisos necesarios para realizar la operación, según la información seleccionada para la manipulación.

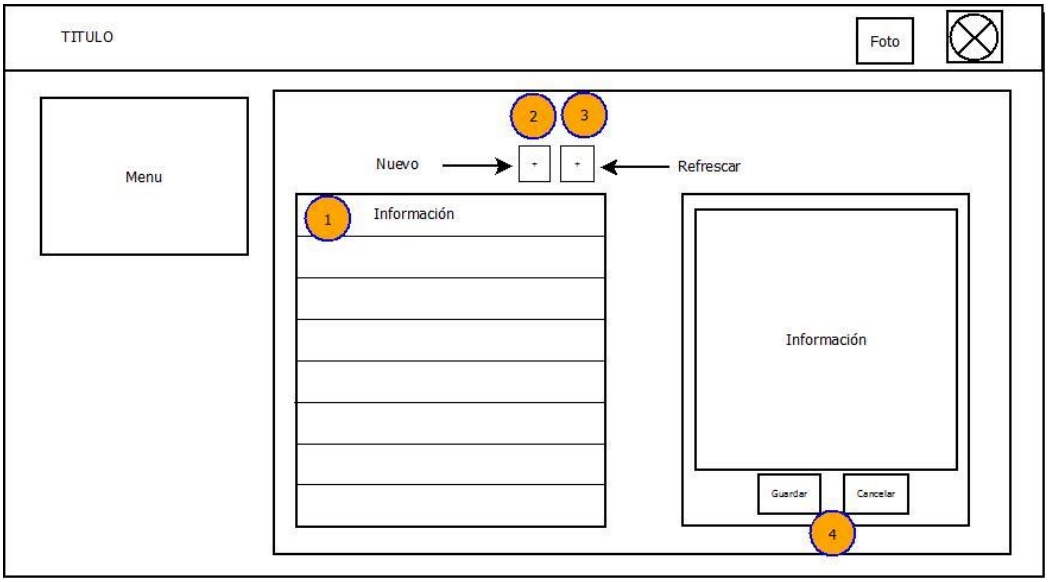

Figura 4.21: Interfaz del tratamiento de la información.

Elaborado por: Wilson Pérez – Investigador

1. **Información**: en esta sección se presenta y realiza las opciones de la manipulación de la información de los proceso a realizarse.

2. **Nuevo:** En esta opción no permitirá agregar información a la aplicación dependiendo de si tiene o no permisos para realizar la opción.

3. **Refrescar**: esta opción permitirá refrescar la información de los procesos que se está manipulando.

4. **Tratamiento de la información**: en esta sección se presentara la información que se ha seleccionado para la manipulación y almacenamiento del mismo.

4.7.1.7. **Cuadros modales de ingreso de información**

Ventana modal para el ingreso de información de los diferentes procesos de que realiza la aplicación web.

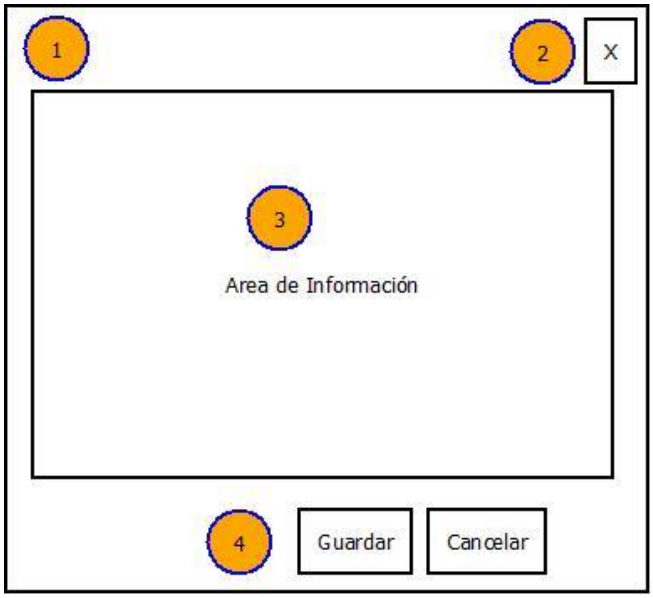

Figura 4.22: Cuadros modales de ingreso de información.

Elaborado por: Wilson Pérez – Investigador

1. **Modal**: en esta sección se presenta el contenido de todo el cuadro modal.

2. **Opción de cierre de modal:** opción de cierre del cuadro modal de ingreso de información.

3. **Área de información**: en esta sesión se ingresa zara la información requerida por el proceso que se manipula por el usuario que inicio sesión.

4. **Opciones de manipulación**: en esta sección se presentara la las operaciones realizarse con la información a ingresarse.

4.7.1.8. **Modal de eliminación de información**

Ventana modal para la eliminación de información de los diferentes procesos de que realiza la aplicación web.

Figura 4.23: Modal de eliminación de información.

Elaborado por: Wilson Pérez – Investigador

1. **Modal**: en esta sección se presenta el contenido de todo el cuadro modal.

2. **Opción de cierre de modal:** opción de cierre del cuadro modal de eliminación de información.

3. **Información**: en esta sesión se mostrara la información a eliminarse.

4. **Opciones de manipulación**: en esta sección se presentara la las operaciones realizarse con la información a eliminar.

4.7.1.9. **Interfaz de ver espacios disponibles**

Ventana que permitirá ver la disponibilidad por secciones de los espacios disponibles de parqueo en la universidad técnica de Ambato.

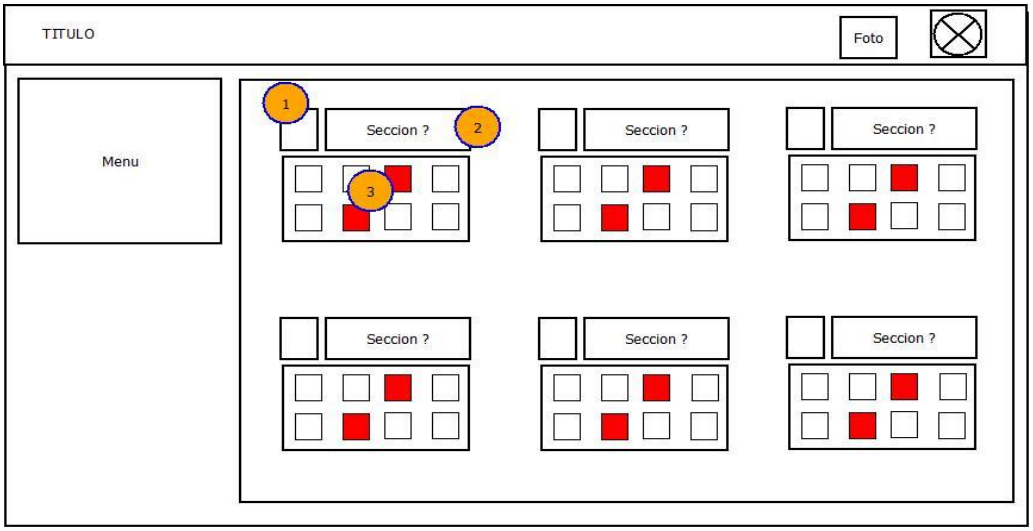

Figura 4.24: Interfaz de ver espacios disponibles.

Elaborado por: Wilson Pérez – Investigador

1. **Foto**: en esta sección se presenta la imagen de la sección o facultad que este el o los espacios de parqueadero.

2. **Nombre de la sección:** en esta sección estará el nombre de la sección o facultad de la institución.

3. **Espacios de parqueadero:** en esta sesión se mostrara la información de los espacios que tiene cada una de las secciones, así como la administración de la misma.

4.7.1.10. **Interfaz de administración de espacios de parqueaderos**

Ventana que permitirá administrar los espacios de parqueo en la universidad técnica de Ambato.

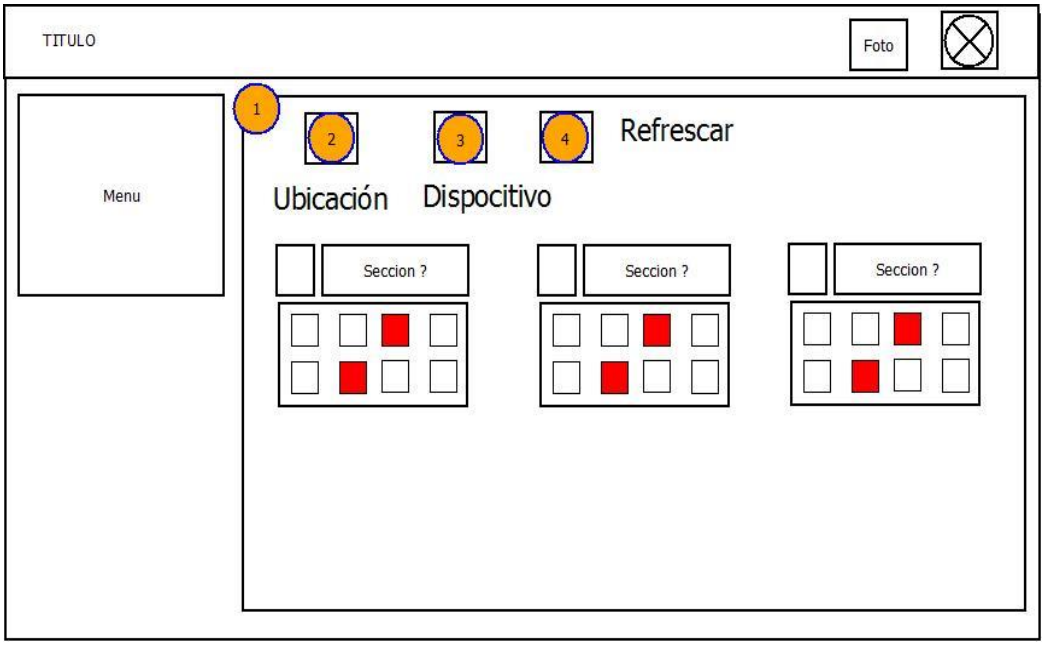

Figura 4.25: Interfaz de administración de espacios de parqueaderos

Elaborado por: Wilson Pérez – Investigador

1. **Espacio de trabajo**: en esta sección se presenta el contenido de todos los espacios disponibles.

2. **Ubicación:** opción no permitirá agregar más ubicaciones o secciones de parqueadero en la institución.

3. **Dispositivo**: en esta opción ayudara a agregar más dispositivos para asociar a los espacios disponibles.

4. **Refrescar**: en esta opción utilizara para refrescar los cambios de espacios de parqueadero.

4.7.1.11. **Interfaz de configuración de la cuenta**

Ventana que permitirá administrar la información del usuario que inicio sesión en la aplicación.

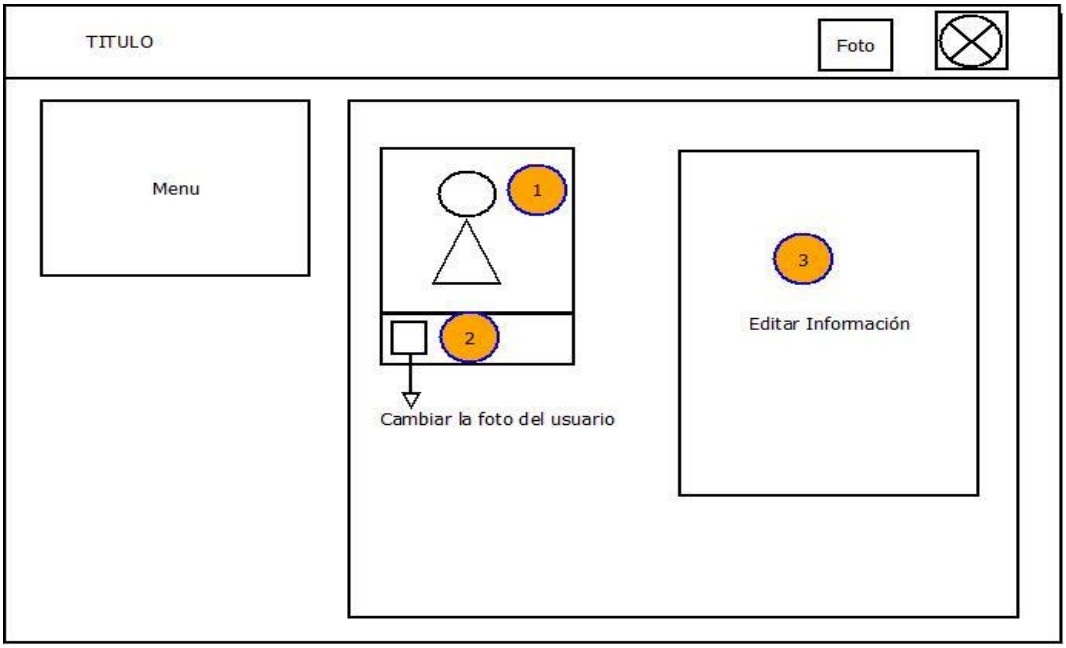

Figura 4.26: Interfaz de configuración de la cuenta

Elaborado por: Wilson Pérez – Investigador

1. **Foto**: en esta sección se presenta una imagen de usuario que ha iniciado sección.

2. **Cambio de imagen:** opción no permitirá modificar la imagen de perfil de usuario que inicio sesión.

3. **Editar Información**: en esta sesión se manipulara la información y se modifica la misma.

4.7.1.12. **Interfaz espacios por sección de parqueadero**

Ventana que mostrara la información de la cantidad de espacios por sección de parqueadero

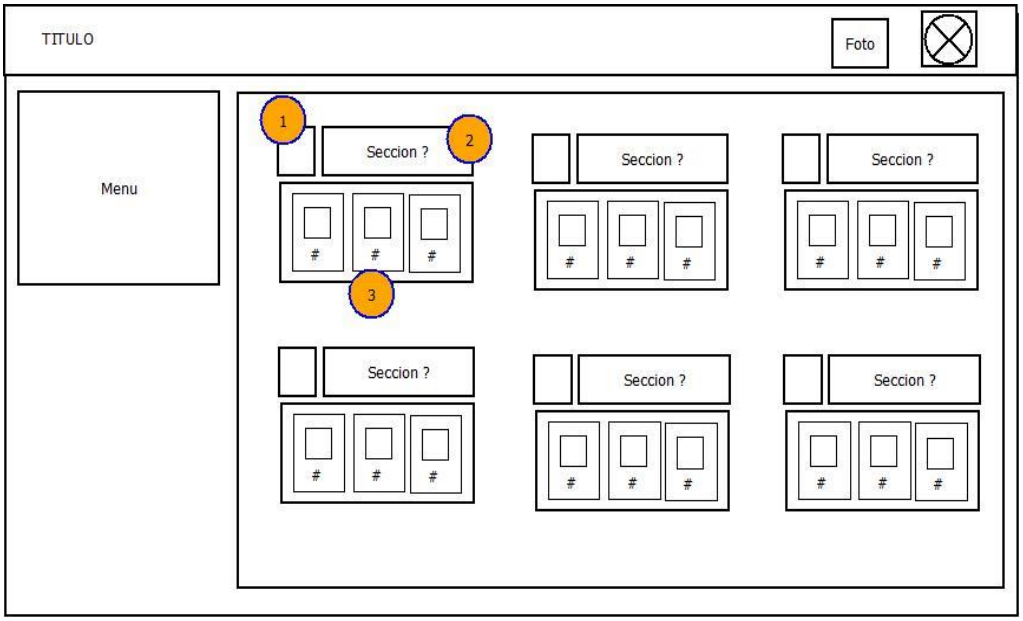

Figura 4.27: Interfaz espacios por sección de parqueadero

Elaborado por: Wilson Pérez – Investigador

1. **Foto**: en esta sección se presenta la imagen de la sección o facultad que este el o los espacios de parqueadero.

2. **Nombre de la sección:** en esta sección estará el nombre de la sección o facultad de la institución.

3. **Espacios de parqueadero:** en esta sesión se mostrara la información resumida de la cantidad de espacios así como también la cantidad de espacios disponibles y ocupados del mismo.

4.7.1.13. **Interfaz de espacios disponibles**

Ventana que mostrara la información de la cantidad de espacios por sección de parqueadero al usuario final.

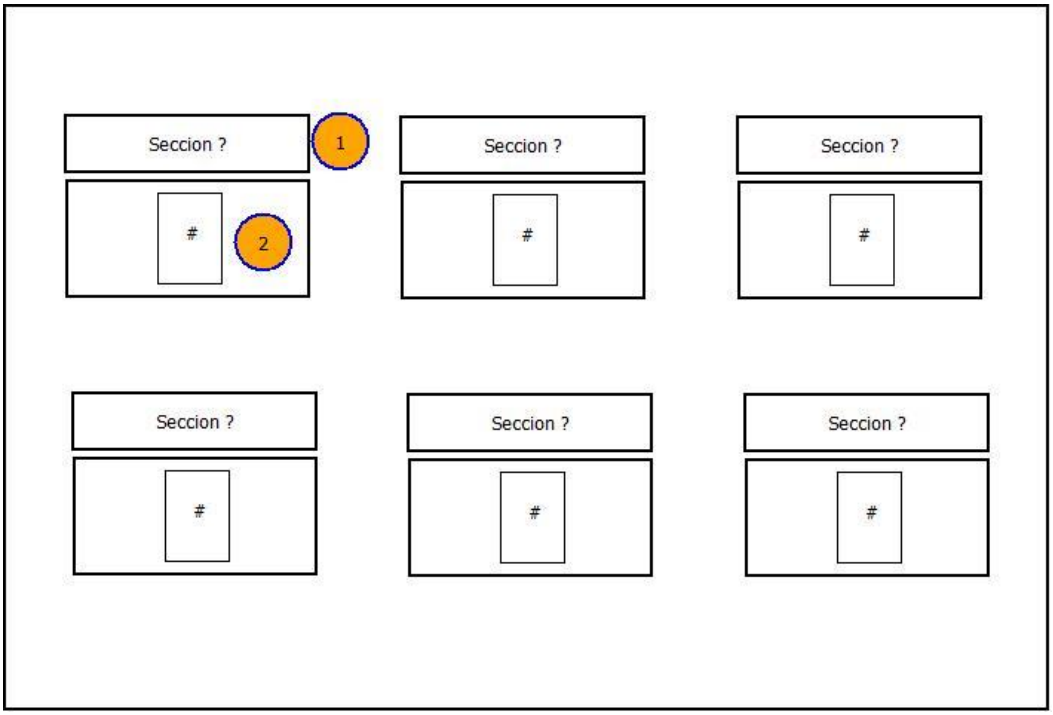

Figura 4.28: Interfaz de espacios disponibles

Elaborado por: Wilson Pérez – Investigador

1. **Nombre de la sección:** en esta sección estará el nombre de la sección o facultad de la institución.

2. **Espacios de parqueadero:** en esta sesión se mostrara la información cantidad de espacios así como también la información de si puede o no ingresar al parqueadero.

4.7.1.14. Interfaz de mensajes de la aplicación

Ventana que mostrara los mensajes de ingreso, actualización, advertencias errores e eliminación de la manipulación de los datos de la aplicación.

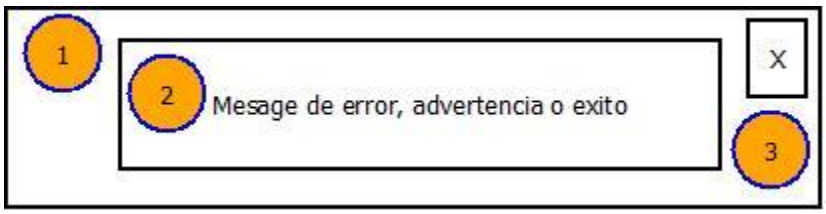

Figura 4.29: Interfaz de mensajes de la aplicación

Elaborado por: Wilson Pérez – Investigador

1. **Área mensaje**: en esta sección se presenta los distintos mensajes de manera individual.

2. **Descripción del mensaje:** en esta sección esta la descripción del mensaje e indicadores del problema y solución del mismo.

3. **Cerrar:** en esta opción permitirá cerrar el mensaje.

4.7.1.15. Interfaz hardware de la aplicación

Descripción de la topología de funcionalidad de la aplicación en el entorno de trabajo.

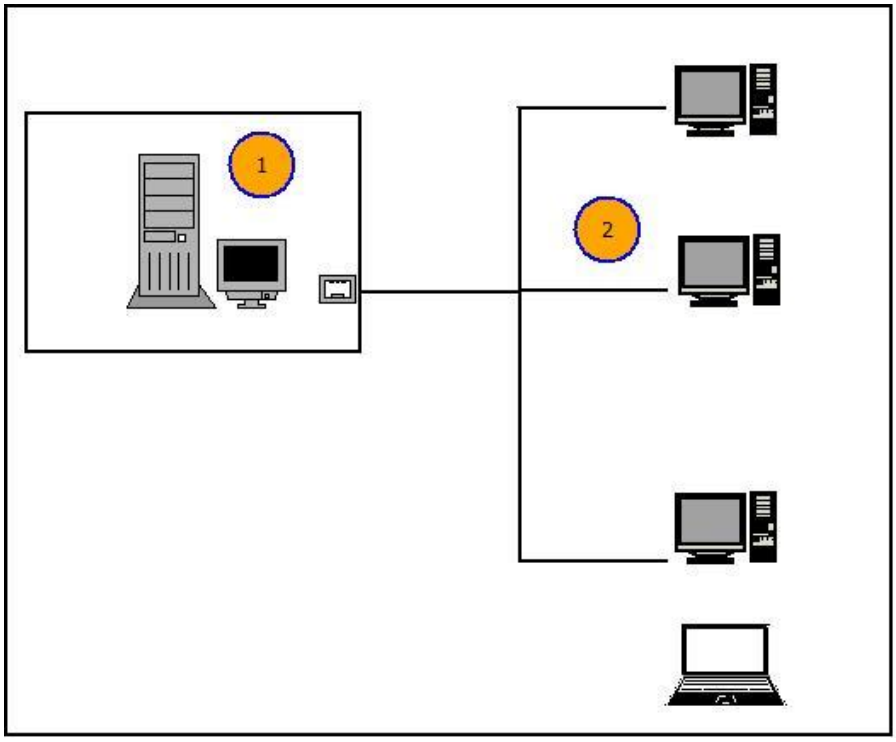

Figura 4.30: Interfaz hardware de la aplicación

Elaborado por: Wilson Pérez – Investigador

1. **Servidor**: en esta sección se representa el servidor donde se alojara nuestra aplicación web y la base de datos.

2. **Cliente:** en esta sección representa los clientes y las facilidades de acceso que tienen dependiendo de la topología de red, ya sea de manera cableada o inalámbrica.

4.7.1.16. Interfaz hardware de dispositivos

Descripción hardware donde describe la los dispositivos de recepción y emisión de señales para la aplicación.

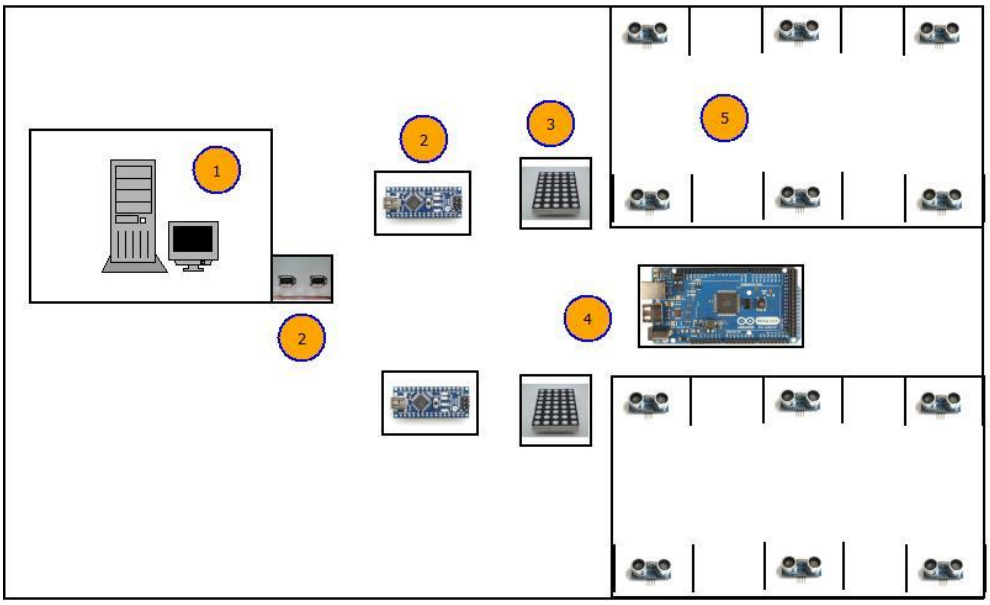

Figura 4.31: Interfaz hardware de la dispositivos

Elaborado por: Wilson Pérez – Investigador

1. **Servidor**: en esta sección el servidor que adquiere los datos generados por los receptores de señales (Arduinos).

2. **Puerto COM:** en esta sección se muestra el medio utilizado por el servidor receptar las señales emitidas por los sensores.

3. **Arduido Nano :** con este dispositivo utilizado para el manejo de la matriz de leds.

4. **Pantallas led**: este dispositivo es donde se visualizan los indicadores de espacios disponibles de parqueaderos, así como las señales de indicción de espacios disponibles.

5. **Ardunino Mega**: es donde obtenemos y tratamos las señales para el uso posterior de la aplicación web.

6. **Sensores ultrasónicos**: son los encargados de censar si hay o no algún objeto que bloque, reconociendo y tomando este dato como un espacio ocupado.

4.7.2. **Requisitos de Rendimiento**

Se recalcar que la aplicación debe funcionar sobre un sistema cualquier plataforma que tenga configurada un servidor web apache y las dependencias de php y su motor de base de datos correspondientes.

4.7.3. **Restricciones de Diseño**

Se establece que la aplicación solo deberá de ser utilizada en navegadores web de equipos de escritorio y actualizados para un correcto funcionamiento de y facilidad de manejo de los diferentes usuarios de la misma.

4.7.4. **Atributos del Sistema**

Los atributos del sistema son cualidades no funcionales que a menudo se confunden con las funciones.
Por ejemplo:
Facilidad de uso, tolerancia a fallas, tiempo de respuesta, metáfora de interfaz, plataformas.

Atributo	Descripción
Tiempo de respuesta paginas simples.	3 a 4 segundos de respuesta, para cada una de la manipulación de los formularios web.
Tiempo de respuesta lectura de los sensores.	0.5 segundo al recibir la información de los sensores ultrasónicos. El tiempo de respuesta de la matriz de led 0.5 segundos ya que solo recibe la señal e imprime la respuesta esperada.
Tiempo de envió de señales de la aplicación a la aplicación.	El tiempo de envió de señales a los dispositivos desde la aplicación, es de 4 a 6 segundos en el envió de la información, pero esto no es una limitación en la aplicación ya que no afecta en nada en la aplicación y el tiempo de respuesta están dentro de los rangos de respuesta esperada.
Facilidad de uso	En intuitivo para cada uno de los usuarios ya que proporciona una interfaz amigable y fácil de manipular.
Tolerancia a fallos	Si la transacción no se completa y existe algún fallo no guarda la información si no es procesada ates del fallo correspondiente perdiendo el dato a almacenar en los momentos del fallo.
Interfaz	La interfaz es amigable, y es presentada en colores referentes a la institución.
Plataformas	Las plataformas soportadas para la instalación son Linux, Windows, Mac, Unix para instalación del servidor. Para el cliente: son soportados para los navegadores Firefox, Chrome, Explorer, Opera y otros que estén en las últimas versiones y con soporte javascript.

Tabla 4.30: Atributos del Sistema

Elaborado por: Wilson Pérez – Investigador

4.7.5. **Otros Requisitos**

No se definen otros requisitos.

4.8. **Implementación y pruebas de la aplicación**

En esta sección se define el desarrollo de la base de datos, aplicación web, y las pruebas de la misma, de los formularios y segmentos de código de cómo está desarrollada la aplicación web.

4.9. **Base de datos**

El desarrollo de la base de datos se la realiza en Postgres que es un motor de base de datos no propietaria el cual no proporciona las herramientas necesarias para la implementación de la aplicación web, además las características del motor de base de datos, es multiplataforma, que puede ser montada tanto en un sistema operativo propietario o libre.

También un beneficio fundamental de la base de datos es el soporte que tiene ya que por ser libre y robusta la tenemos en foros páginas oficiales y redes sociales con respuesta inmediata a cualquier inquietud sobre la herramienta de base de datos.

La base de datos se diseñó en un entorno SQL que facilita la búsqueda de la información de manera más rápida ya que se crean las relaciones ayudando con la integridad de datos.

4.10. **Diseño de la base de datos**

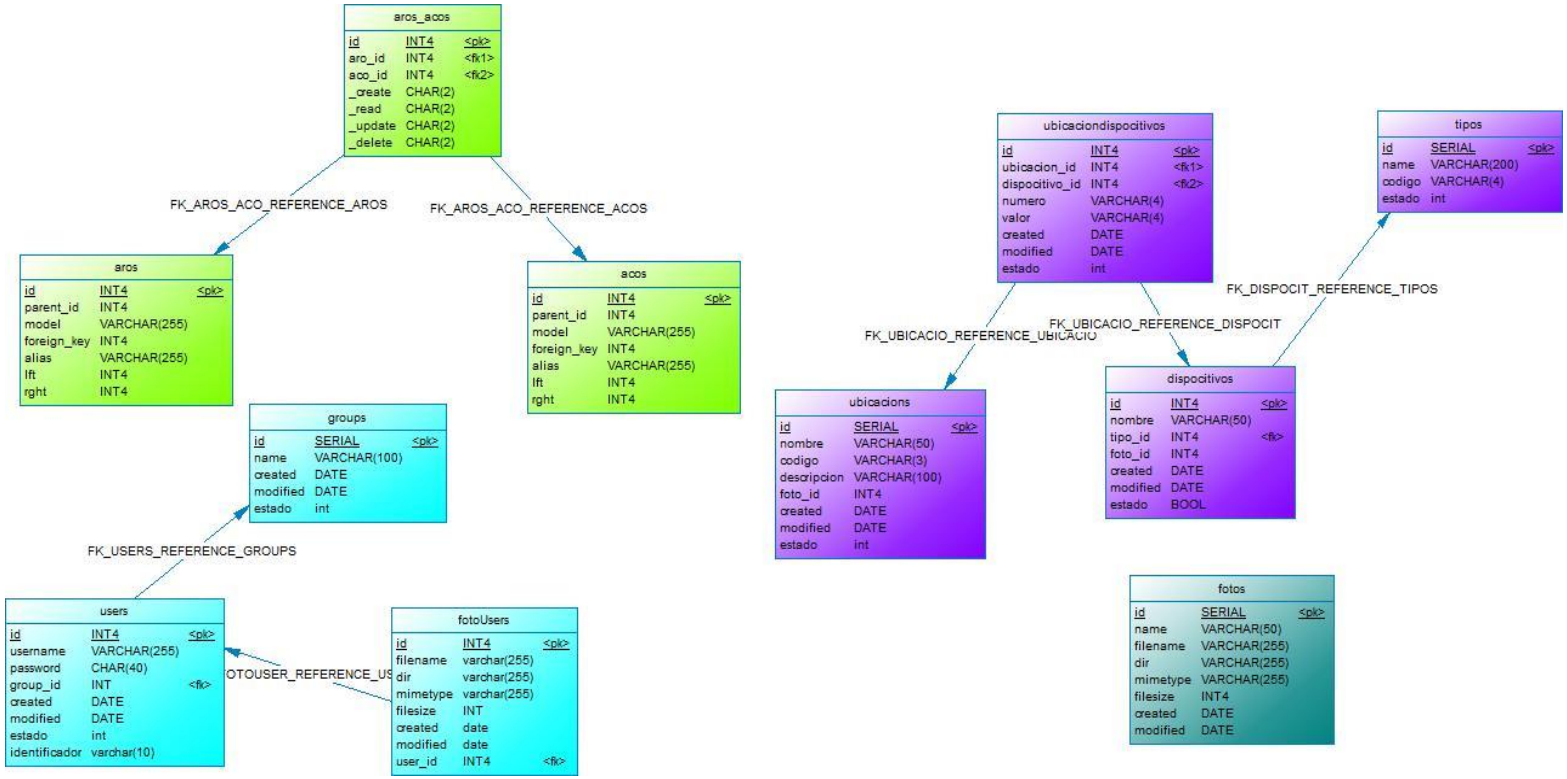

Figura 4.32: Diseño de la base de datos

Elaborado por: Wilson Pérez – Investigador

4.11. Diccionario de datos

Descripción de las tablas de la base de datos y cada uno de los campos.

Tabla: tipos					
Nombre	**Tipo**	**Tamaño**	**P.K.**	**F.K.**	**Obligatorio**
Id	Serial		Si	no	Si
Name	Varchar	200	No	no	No
Código	Varchar	4	No	no	No
Estado	Int		No	no	No

Tabla 4.31: Diccionario de datos tabla tipos

Elaborado por: Wilson Pérez – Investigador

Tabla: dispositivos					
Nombre	**Tipo**	**Tamaño**	**P.K.**	**F.K.**	**Obligatorio**
Id	SERIAL		SI	NO	SI
Nombre	VARCHAR	50	NO	NO	NO
tipo_id	INT4	4	NO	SI	NO
foto_id	INT4	4	NO	NO	NO
Created	DATE		NO	NO	NO
modified	DATE		NO	NO	NO
Estado	BOOL		NO	NO	NO

Tabla 4.32: Diccionario de datos tabla dispositivos

Elaborado por: Wilson Pérez – Investigador

Tabla: fotos					
Nombre	**Tipo**	**Tamaño**	**P.K.**	**F.K.**	**Obligatorio**
Id	SERIAL		SI	NO	SI
Name	VARCHAR	50	NO	NO	NO
filename	VARCHAR	255	NO	NO	NO
Dir	VARCHAR	255	NO	NO	NO
mimetype	VARCHAR	255	NO	NO	NO
Filesize	INT4	4	NO	NO	NO
Created	DATE		NO	NO	NO
modified	DATE		NO	NO	NO

Tabla 4.33: Diccionario de datos tabla fotos

Elaborado por: Wilson Pérez – Investigador

Tabla: ubicacions					
Nombre	**Tipo**	**Tamaño**	**P.K.**	**F.K.**	**Obligatorio**
Id	SERIAL		SI	NO	SI
Nombre	VARCHAR	50	NO	NO	NO
Código	VARCHAR	3	NO	NO	NO
descripcion	VARCHAR	100	NO	NO	NO
foto_id	INT4	4	NO	NO	NO
Created	DATE		NO	NO	NO
modified	DATE		NO	NO	NO
Estado	INT		NO	NO	NO

Tabla 4.34: Diccionario de datos tabla ubucacions

Elaborado por: Wilson Pérez – Investigador

Tabla: ubicaciondispocitivos					
Nombre	**Tipo**	**Tamaño**	**P.K.**	**F.K.**	**Obligatorio**
Id	SERIAL		SI	NO	SI
ubicacion_id	INT4	4	NO	SI	NO
dispocitivo_id	INT4	4	NO	SI	NO
Numero	VARCHAR	4	NO	NO	NO
Valor	VARCHAR	4	NO	NO	NO
Created	DATE		NO	NO	NO
modified	DATE		NO	NO	NO
Estado	INT		NO	NO	NO

Tabla 4.35: Diccionario de datos tabla ubucaciondispocitivos

Elaborado por: Wilson Pérez – Investigador

Tabla: groups					
Nombre	**Tipo**	**Tamaño**	**P.K.**	**F.K.**	**Obligatorio**
Id	SERIAL		SI	NO	SI
Name	VARCHAR	100	NO	NO	SI
Created	DATE		NO	NO	NO
modified	DATE		NO	NO	NO
Estado	INT		NO	NO	NO

Tabla 4.36: Diccionario de datos tabla groups

Elaborado por: Wilson Pérez – Investigador

Tabla: users					
Nombre	**Tipo**	**Tamaño**	**P.K.**	**F.K.**	**Obligatorio**
Id	SERIAL		SI	NO	SI
username	VARCHAR	255	NO	NO	SI
password	CHAR	40	NO	NO	SI
group_id	INT		NO	SI	SI
Created	DATE		NO	NO	NO
modified	DATE		NO	NO	NO
Estado	INT		NO	NO	NO
identificador	VARCHAR	10	NO	NO	NO

Tabla 4.37: Diccionario de datos tabla users

Elaborado por: Wilson Pérez – Investigador

Tabla: fotousers					
Nombre	**Tipo**	**Tamaño**	**P.K.**	**F.K.**	**Obligatorio**
Id	SERIAL		SI	NO	SI
filename	VARCHAR	255	NO	NO	NO
Dir	VARCHAR	255	NO	NO	NO
mimetype	VARCHAR	255	NO	NO	NO
Filesize	INT		NO	NO	NO
Created	DATE		NO	NO	NO
modified	DATE		NO	NO	NO
user_id	INT4	4	NO	SI	NO

Tabla 4.38: Diccionario de datos tabla fotousers

Elaborado por: Wilson Pérez – Investigador

Tabla: Aros					
Nombre	**Tipo**	**Tamaño**	**P.K.**	**F.K.**	**Obligatorio**
Id	INT4	4	SI	NO	SI
parent_id	INT4	4	NO	NO	NO
Model	VARCHAR	255	NO	NO	NO
foreign_key	INT4	4	NO	NO	NO
Alias	VARCHA	255	NO	NO	NO
Lft	INT4	4	NO	NO	NO
Rght	INT4	4	NO	NO	NO

Tabla 4.39: Diccionario de datos tabla Aros

Elaborado por: Wilson Pérez – Investigador

Tabla: Acos					
Nombre	**Tipo**	**Tamaño**	**P.K.**	**F.K.**	**Obligatorio**
Id	INT4	4	SI	NO	SI
parent_id	INT4	4	NO	NO	NO
Model	VARCHAR	255	NO	NO	NO
foreign_key	INT4	4	NO	NO	NO
Alias	VARCHAR	255	NO	NO	NO
Lft	INT4	4	NO	NO	NO
Rght	INT4	4	NO	NO	NO

Tabla 4.40: Diccionario de datos tabla Acos

Elaborado por: Wilson Pérez – Investigador

Tabla: aros_acos					
Nombre	**Tipo**	**Tamaño**	**P.K.**	**F.K.**	**Obligatorio**
Id	INT4	4	SI	NO	SI
aro_id	INT4	4	NO	SI	SI
aco_id	INT4	4	NO	SI	SI
_create	CHAR	2	NO	NO	SI
_read	CHAR	2	NO	NO	SI
_update	CHAR	2	NO	NO	SI
_delete	CHAR	2	NO	NO	SI

Tabla 4.41: Diccionario de datos tabla aros_acos

Elaborado por: Wilson Pérez – Investigador

4.12. Implementación de la aplicación web

Se describe el lenguaje de programación así como sus clases objetos e interfaz de la misma y las herramientas de desarrollo.

4.12.1. **Interfaz de la aplicación web**

La interfaz de la aplicación es realizada con Bootstrap el cual ayuda la presentación de la interfaz mediante la utilización de Html, Css, Js el que ayuda al diseño responsivo de la página.

El cual se utiliza de la siguiente manera la ubicación de los componentes en la distribución de las filas y columnas donde albergamos el contenido de la aplicación.

AJAX, acrónimo de Asíncronos JavaScript (JavaScript asíncrono), es una técnica de desarrollo web para crear aplicaciones interactivas. Estas aplicaciones se ejecutan en el cliente, es decir, en el navegador de los usuarios mientras se mantiene la comunicación asíncrona con el servidor en segundo plano. De esta forma es posible realizar cambios sobre las páginas sin necesidad de recargarlas, mejorando la interactividad, velocidad y usabilidad en las aplicaciones.

JavaScript, guardan funciones y variables globales que se ejecutarán en la página web, pudiendo llamar a sus funciones desde cualquier subpágina sin tener que incrustar scripts en cada una de ellas y ahorrando así código.

Css, es un lenguaje usado para definir la presentación de un documento estructurado escrito en HTML con la especificación de las hojas de estilo que servirán de estándar para los agentes de usuario o navegadores.

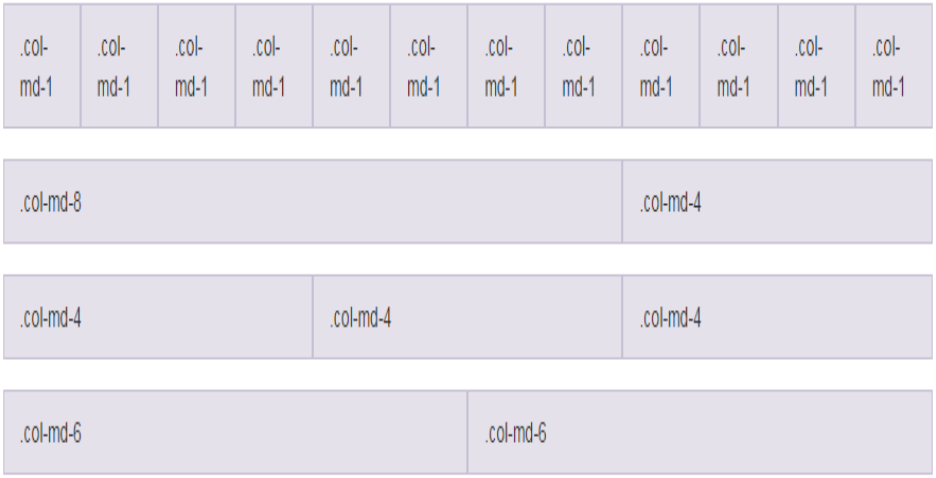

Figura 4.33: Esquema grid bootstrap

Elaborado por: getbootstrap.com

Fragmento de código de un formulación de la aplicación web

```
<div role="navigation" class="navbar navbar-default navbar-static-top"
id="menuHorizontal">
<div class="row">
<div class="col-md-12">
<div class="navbar-header">
<button data-target=".navbar-collapse" data-toggle="collapse" class="navbar-
toggle" type="button">
<span class="sr-only">Toggle navigation</span>
<span class="icon-bar"></span>
<span class="icon-bar"></span>
<span class="icon-bar"></span>
</button>
<?php echo $this->Html->link(
$this->Html->image('escudo.jpg', array('style'=>'width:60px;')),
array('controller'=>'aplicacion','action'=>'index') ,
array('escapeTitle' => false,'class'=>'navbar-brand')
);
echo $this->Html->link(
'Universidad Técnica de Ambato',
array('controller'=>'aplicacion','action'=>'index') ,
array('class'=>'navbar-brand')
);?>
</div>
<div class="navbar-collapse collapse">
<form class="navbar-form navbar-right">
<!-- Large button group -->
<div class="btn-group">
<button id="menu1" class="btn btn-default btn-sm dropdown-toggle"
type="button" data-toggle="dropdown">
Menu <span class="glyphicon glyphicon-cog"></span>
```

```
</button>
<ul class="dropdown-menu" role="menu" aria-labelledby="menu1">
<li id="editCuenta"><a href="#"><span class="fa fa-cogs fa-fw"></span>
Configurar cuenta</a></li>
<li id="pop"><a href="#"><span class="fa fa-question fa-fw"> </span>
Ayuda</a></li>
<li><a href="#"><span class="fa fa-eye fa-fw"></span> Acerca de ...</a></li>
<li class="divider"></li>
<li>
<?php
echo $this->Html->link(
'<span class="fa fa-times fa-fw"></span> Salir',
array('controller'=>'Users','action'=>'logout'),
array('escapeTitle' => false)
);
?>
</li>
</ul>
</div>
</form>
<form class="navbar-form navbar-right">
<div class="btn-group">
<a href="#" id="menu2" class="dropdown-toggle" data-toggle="dropdown" data-
container="body" data-placement="bottom"
data-content="#pruebalashfljhsadjlf">
<?php
echo $this->Html->image(null,
array('style'=>'width:35px;','id'=>'FotoUsuario','class'=>'img-circle'));
?>
</a>
</div>
</form>
<form class="navbar-form navbar-right" role="search">
<div class="form-group">
<input type="text" class="form-control" placeholder="Buscar">
</div>
<button type="submit" class="btn btn-default " > <i class="fa fa-
search"></i></button>
</form></div><!--/.navbar-collapse    --
  </div>
   </div>
</div>
<!--<div id="demo" class="collapse in">...</div>-->
<div class="row">
<div class="col-md-3 col-sm-3 col-xs-3" >
<?php echo $this->element('menuadministrador'); ?>
</div>
<div class="col-md-9" style=" ">
<div class="panel panel-default" >
<div class="panel-body" id="ajaxApp2">
<div class="row" >
<div class="col-md-12" >
<h1>UTAPARK</h1>
</div>
</div>
</div>
</div>
</div>
```

```
<div class="nav navbar-fixed-bottom" style="z-index: 1100" >
<div class="col-md-3" id="alertas">
<?php echo $this->Session->flash();?>
</div>
</div>
</div>
```

Donde el resultado del código es el siguiente.

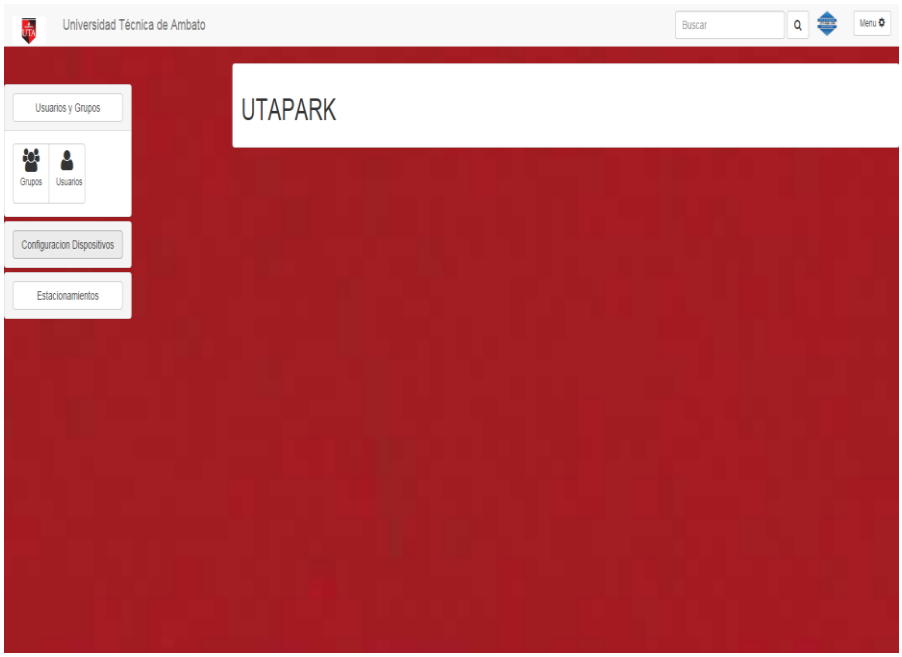

Figura 4.34: Resultado del código de la interfaz

Elaborado por: Wilson Pérez – Investigador

4.12.2. **Desarrollo de formularios de la aplicación**

4.12.2.1. **Modelo**

Es la representación de la información con la cual el sistema opera, por lo tanto gestiona todos los accesos a dicha información, tanto consultas como actualizaciones, implementando también los privilegios de acceso que se hayan descrito en las especificaciones de la aplicación

Fragmento de código de un modelo de la aplicación web.

```
<?php
App::uses('AppModel', 'Model');
/**
 * Ubicaciondispocitivo Model
 *
 * @property Ubicacion $Ubicacion
 * @property Dispocitivo $Dispocitivo
 */
class Ubicaciondispocitivo extends AppModel {


        //The Associations below have been created with all possible keys,
those that are not needed can be removed

/**
 * belongsTo associations
 *
 * @var array
 */
public $belongsTo = array(
'Ubicacion' => array(
'className' => 'Ubicacion',
'foreignKey' => 'ubicacion_id',
'conditions' => '',
'fields' => '',
'order' => ''
'Dispocitivo' => array(
'className' => 'Dispocitivo',
'foreignKey' => 'dispocitivo_id',
'conditions' => '',
'fields' => '',
'order' => ''
));

public $validate = array(
'numero' => array(
'notEmpty' => array(
'rule' => array('notEmpty'),
'message' => 'El Número del parqueadero es requerido',
//'allowEmpty' => false,
'required' => false,
//'last' => false, // Stop validation after this rule
//'on' => 'create', // Limit validation to 'create' or 'update' operations
),
'alphaNumeric' => array(
'rule' => array('custom','/^[0-9[:space:]]*$/'),
'required' => true,
'message' => 'El Número del parqueadero, Sólo numeros'
),'between' => array(
'rule' => array('between', 1, 4),
'message' => 'El Número del parqueadero, Entre 5 y 50 caracteres'
),),'ubicacion_id' => array(
'notEmpty' => array(
'rule' => array('notEmpty'),
'message' => 'Debe tener registrado ubicaciones',
```

```
//'allowEmpty' => false,
'required' => false,
//'last' => false, // Stop validation after this rule
//'on' => 'create', // Limit validation to 'create' or 'update' operations
),),
'valor' => array(
'notEmpty' => array(
'rule' => array('notEmpty'),
'message' => 'El Valor es requerido',
//'allowEmpty' => false,
'required' => false,
//'last' => false, // Stop validation after this rule
//'on' => 'create', // Limit validation to 'create' or 'update' operations
),'alphaNumeric' => array(
'rule' => array('custom','/^[0-1[:space:]]*$/'),
'required' => true,
'message' => 'El Valor, Sólo numeros 00 o 11'
),'between' => array(
'rule' => array('between', 2, 2),
'message' => 'El valor, debe tener dos caracteres'
),),'ubicacion_id' => array(
'notEmpty' => array(
'rule' => array('notEmpty'),
'message' => 'Debe tener registrado ubicaciones',
//'allowEmpty' => false,
'required' => false,
//'last' => false, // Stop validation after this rule
//'on' => 'create', // Limit validation to 'create' or 'update' operations
),),'dispocitivo_id' => array(
'notEmpty' => array(
'rule' => array('notEmpty'),
'message' => 'Debe tener registrado dispositivos',
//'allowEmpty' => false,
'required' => false,
//'last' => false, // Stop validation after this rule
//'on' => 'create', // Limit validation to 'create' or 'update' operations
),),);}
```

4.12.2.2. **Vista**

Presenta el modelo en un formato adecuado para interactuar usualmente la interfaz de usuario por tanto representar al modelo como salida.

Fragmento de código de las vistas:

Index.ctp

```
<?php $this->Paginator->options(array(
   'update' => '#ajaxApp2'
          ));
```

```
?>
<div class="row">
        <div class="col-md-6">
        <div class="panel panel-default">
        <div class="panel-heading">
        <h3 class="panel-title">Ubicaciones</h3>
</div>
<div class="panel-body">
<div class="row">
<div class="col-md-12">
<div class="btn-group navbar-right opciones">
<button type="button" id="btnAgregarUbicacion" data-
target="#ModalNuevoUbicacion" data-placement="top" title="Nueva
Ubicación" class="btn btn-primary btn-sm tol">
<span class="fa fa-plus"><span class="fa fa-
user"></span></span></button>
<button type="button" id="btnRefrescarUbicacion" data-toggle="tooltip"
data-placement="top" title="Actualizar" class="btn btn-primary btn-sm tol">
<span class="fa fa-refresh"></span></button></div></div></div>
<div class="row ">
<div class="col-md-12 content_5">
<table class="table table-condensed  table-responsive  table-hover">
<thead>
<tr>
<th><?php echo $this->Paginator->sort('nombre','Nombre'); ?></th>
<th><?php echo $this->Paginator->sort('descripcion','Descripción');
?></th><th><?php echo $this->Paginator->sort('foto_id','Foto'); ?></th>
<th><?php echo $this->Paginator->sort('estado','Estado');?></th></tr>
</thead>
<tbody>
<?php  foreach ($ubicacions as $ubicacion): ?>
<tr class="datosLista" data-Ubicacion="<?php echo
$ubicacion['Ubicacion']['id'] ?>" > <td><?php echo
h($ubicacion['Ubicacion']['nombre']); ?> </td>
<td><?php echo h($ubicacion['Ubicacion']['descripcion']); ?> </td>
<td><?php echo $this->Html-
>image('../'.$ubicacion['Foto']['dir'].'/thumb/small/'.$ubicacion['Foto']['filenam
e'], array('style'=>'width:50px;')); ?>  
</td>
<td><?php if (h($ubicacion['Ubicacion']['estado'])=='1'){echo
__('Activo');}else{echo __('Inactivo');} ?> </td>
        </tr>     <?php endforeach; ?>   </tbody>
        </table>
        <p>
        <?php   echo $this->Paginator->counter(array(
        'format' => __('Page {:page} of {:pages}, showing {:current} records
out of {:count} total, starting on record {:start}, ending on {:end}')));?>
        </p>

<ul class="paging pagination pagination-sm">
<?php echo '<li>'.$this->Paginator->prev('«', array('tag' => false),
null).'</li>'; echo $this->Paginator->numbers(array( 'separator' => '',
                                        'currentClass' => 'active',
                                         'currentTag' => 'a',
                                         'tag' => 'li', 'modulus'=>5 ));
echo '<li>'.$this->Paginator->next('»', array('tag' => false), null).'</li>';?>
</ul>
                </div>
```

```
                    </div>
                </div>
                </div>
                </div>
                <div class="col-md-6" id="div1IndexUbicacion">
                    <div class="loderAjax"></div>
                    <div class="panel panel-default">
                        <div class="panel-body">
                            <div id="ajaxEditUbicacion">
                            </div>
                        </div>
                    </div>
                </div>
</div>
<div class="modal fade" id="ModalNuevoUbicacion" tabindex="-1"
role="dialog" aria-labelledby="myModalLabel" aria-hidden="true">
  <div class="modal-dialog">
    <div class="modal-content">
      <div class="modal-header">
        <button type="button" class="close" data-dismiss="modal" aria-
hidden="true">&times;</button>
        <h4 class="modal-title">Nueva Ubicaci&#243n</h4>
      </div>
      <div class="modal-body" id="ajaxNuevoUbicacion">
      </div>
    </div><!-- /.modal-content -->
  </div><!-- /.modal-dialog -->
</div><!-- /.modal -->

<script type="text/javascript">

            $(function() {
                    $('.tol').tooltip();
                    $(".datosLista").bind("click",function(){
             $.ajax({
                    async:true,
                    beforeSend: function (data, textStatus) {
                    $("#div1IndexUbicacion").hide('slide');
                            } ,
                    complete: function (data, textStatus) {
                    loderAremover();
                    } ,
                    success:function (data, textStatus) {
                    loderA('.loderAjax');
                    $("#div1IndexUbicacion").show('slide',
                    function  () {
                    $('#ajaxEditUbicacion').html(data);
                    });},
                    type:"post",
                    url:"\/index.php\/ubicacions\/view\/"
                    +$(this).attr('data-Ubicacion')
                     });
                            return false;
                       });
                      $("#div1IndexUbicacion").hide();});
</script>
<?php
            $this->Js->get('#btnAgregarUbicacion');
```

```
$this->Js->event('click',
$this->Js->request(array('action' => 'add'),array('async' => true,
        'method'=>'get',
        'update' => '#ajaxNuevoUbicacion',
        'success'=>'$("#ModalNuevoUbicacion").modal("show");'
        )));
$this->Js->get('#btnRefrescarUbicacion');
$this->Js->event('click', '$(
"#btnPanelUbicaciones").trigger("click");');?>
```

El resultado del código del index.ctp

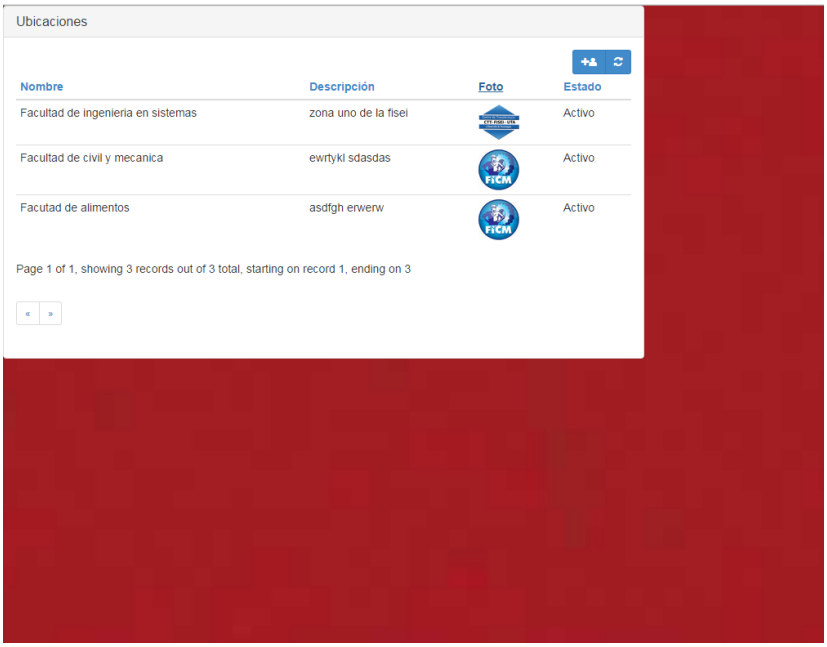

Figura 4.35: Resultado del código de la vista index
Elaborado por: Wilson Pérez – Investigador

Add.ctp

```
<?php echo $this->Html->css(array('bootstrap-select')); ?>
<?php echo $this->Html->script(array('bootstrap-select'));?>
<div class="groups form">
<?php echo $this->Form-
>create('Ubicaciondispocitivo',array('class'=>'form-horizontal')); ?>
<fieldset>
<div class="form-group">
<label for="UbicaciondispocitivoUbicacion_Id" class="col-sm-2 control-
label">Ubicacion:</label>
<div class="col-sm-10">
<select class="form-control selectpicker" data-live-search="true"
name="data[Ubicaciondispocitivo][ubicacion_id]"
id="UbicaciondispocitivoUbicacionId">
<?php foreach ($imagesu as $image):  ?>
<option value="<?php echo $image['Ubicacion']['id']; ?>" data-
content='<?php echo $this->Html-
>image('../'.$image['Foto']['dir'].'/thumb/small/'.$image['Foto']['filename'],
```

```
array('style'=>'width:25px;')); ?>   <?php echo
$image['Ubicacion']['nombre'];?>' >
<?php endforeach; ?>
</select>
</div>
</div>
<div class="form-group">
<label for="UbicaciondispocitivoDispocitivo_id" class="col-sm-2 control-
label">Dispocitivo:</label>
<div class="col-sm-10">
<select class="form-control selectpicker" data-live-search="true"
name="data[Ubicaciondispocitivo][dispocitivo_id]"
id="UbicaciondispocitivoDispocitivoId">
<?php foreach ($images as $image):  ?>
<option value="<?php echo $image['Dispocitivo']['id']; ?>" data-
content='<?php echo $this->Html-
>image('../'.$image['Foto']['dir'].'/thumb/small/'.$image['Foto']['filename'],
array('style'=>'width:25px;')); ?>   <?php echo
$image['Dispocitivo']['nombre'];?>' >
<?php endforeach; ?>
</select>
</div>
</div>
<div class="form-group">
<label for="UbicaciondispocitivoNumero" class="col-sm-2 control-
label">Numero:</label>
<div class="col-sm-10">
<?php if (($ubicaciondispocitivo[0][0]['max'])<9) { ?>
<?php echo $this->Form->input('numero',
array('readonly','div'=>false,'value'=>'0'.($ubicaciondispocitivo[0][0]['max']+1
),'label'=>false,'class'=>'form-control'));?>
<?php } else { ?>
<?php echo $this->Form->input('numero', array('readonly','div'=>false,
'value'=>($ubicaciondispocitivo[0][0]['max']+1),'label'=>false,'class'=>'form-
control'));?>
<?php } ?>
</div>
</div>
<div class="form-group">
<label for="UbicaciondispocitivoValor" class="col-sm-2 control-
label">Valor:</label>
<div class="col-sm-10">
<?php echo $this->Form->input('valor',
array('readonly','value'=>'00','div'=>false,'label'=>false,'class'=>'form-
control'));?>
</div>
</div>
<div class="form-group">
<label for="UbicaciondispocitivoEstado" class="col-sm-2 control-
label">Estado:</label>
<div class="col-sm-10">
<?php
$options = array('1'=>' Activo  ','0'=>' Inactivo');
$attributes = array('legend'=>false,'div'=>false,'label'=>true, 'class'=>'radio-
inline', 'type' => 'radio', 'options' => $options,'default'=>'1');
echo $this->Form->input('estado',$attributes);
?>
</div>
```

```
</div>
</fieldset>
            <div class="form-group">
            <div class="col-sm-offset-2 col-sm-10">
                        <div class="navbar-right">
<?php echo $this->Js->submit( 'Guardar',
array('id'=>'btnNuevoUbicaciondispocitivo',
'class'=>'btn btn-primary',
'div'=>false,
'url' =>  array('action' => 'add'),
'success' => '  var a = eval("(" + data + ")");

if(a["estado"]){
$("#btnCancelarNuevoUbicaciondispocitivo").click();
$("#ModalAgregarParqueadero").on("hidden.bs.modal",function  (e) {
$("#btnRefrescarDispocitivoDetalle").trigger("click");
                                                                    });
                                                        }
                                                $("#alertas").append(a["html"]);
                                    'method'=>'post'
                        )
                                                        );
                                                        ?>
<button class="btn btn-default" id="btnCancelarNuevoUbicaciondispocitivo"
data-dismiss="modal" >Cancelar</button>
</div>
</div>
</div>
</div>
<script type="text/javascript">
$(function  () {
 $('.selectpicker').selectpicker({
'selectedText': 'cat'
            });
             });

</script>
```

Resultado del código de la vista add.ctp

Figura 4.36: Resultado del código de la vista add
Elaborado por: Wilson Pérez – Investigador

Edit.ctp

```
<?php echo $this->Html->css(array('bootstrap-select')); ?>
<?php echo $this->Html->script(array('bootstrap-select'));?>
<div class="groups form">
<?php echo $this->Form->create('Ubicacion',array('class'=>'form-
horizontal')); ?>
<fieldset>
<?php echo $this->Form->input('id');?>
<div class="form-group">
<label for="UbicacionNombre" class="col-sm-2 control-
label">Nombre:</label>
<div class="col-sm-10">
<?php echo $this->Form->input('nombre',
array('div'=>false,'label'=>false,'class'=>'form-control'));?>
</div>   </div>
<div class="form-group">
<label for="UbicacionDescripcion" class="col-sm-3 control-
label">Descripción:</label>
<div class="col-sm-9">
<?php echo $this->Form->input('descripcion', array('type' =>
'textarea','maxlength'=> '100','div'=>false,'label'=>false,'class'=>'form-
control'));?>
</div>   </div>
<div class="form-group">
<label for="UbicacionFoto_id" class="col-sm-2 control-label">Foto:</label>
<div class="col-sm-10">
<select class="form-control selectpicker" data-live-search="true"
name="data[Ubicacion][foto_id]" id="UbicacionFotoId">
<?php foreach ($images as $image):  ?>
<?php if (($ubicacion['Foto']['id'])==($image['Foto']['id'])) { ?>
<option selected="selected" value="<?php echo $image['Foto']['id']; ?>"
data-content='<?php echo $this->Html-
>image('../'.$image['Foto']['dir'].'/thumb/small/'.$image['Foto']['filename'],
array('style'=>'width:25px;')); ?>   <?php echo
$image['Foto']['name'];?>' >
<?php   } else {?>
<option value="<?php echo $image['Foto']['id']; ?>" data-content='<?php
echo $this->Html-
>image('../'.$image['Foto']['dir'].'/thumb/small/'.$image['Foto']['filename'],
array('style'=>'width:25px;')); ?>   <?php echo
$image['Foto']['name'];?>' >
<?php   }?>
<?php endforeach; ?>
</select>
</div>
</div>
<div class="form-group">
<label for="UbicacionEstado" class="col-sm-2 control-
label">Estado:</label>
<div class="col-sm-10">
<?php   $options =
array('1'=>' Activo  ','0'=>' Inactivo');
$attributes = array('legend'=>false,'div'=>false,'label'=>true, 'class'=>'radio-
inline', 'type' => 'radio', 'options' => $options,'default'=>'1');
echo $this->Form->input('estado',$attributes);
```

```
?>
</div>
</div>
</fieldset>
<div class="form-group">
<div class="col-sm-offset-2 col-sm-10">
<div class="navbar-right">
<?php echo $this->Js->submit( 'Cancelar',
array('id'=>'btnCancelarEditarUbicacion',
'class'=>'btn btn-default',
'div'=>false,
'update' => '#ajaxEditUbicacion',
'url' =>  array('action' => 'view',$ubicacion['Ubicacion']['id']),
'method'=>'get'
));?>
<?php echo $this->Js->submit( 'Guardar', array(
'class'=>'btn btn-primary',
'div'=>false
,'success' => 'var a = eval("(" + data + ")");
if(a["estado"]){
$("#btnRefrescarUbicacion").trigger("click");
}
$("#alertas").append(a["html"]);
)
);
?>
</div>
</div>
</div>
</div>
<script type="text/javascript">
$(function  () {
$('.selectpicker').selectpicker({
'selectedText': 'cat'
});
});
</script>
```

El resultado de la vista edit.ctp

Figura 4.37: Resultado del código de la vista edit
Elaborado por: Wilson Pérez – Investigador

View.ctp

```
<div class="row">
<div class="col-md-3">
<div class="related well well-sm" id="infoFoto" >
<?php echo $this->Html-
>image('../'.$ubicacion['Foto']['dir'].'/'.$ubicacion['Foto']['filename'],
array('class'=>'img-responsive img-thumbnail')); ?>
</div>
</div>
<div class="col-md-9">
<div class="loderAjax2"></div>
<div class="well well-sm" id="infoUbicacion">
<div class="btn-group navbar-right operaciones">
<button type="button" id="btnEditarUbicacion" data-toggle="tooltip" data-
placement="top" title="Editar Información" class="btn btn-primary btn-sm
tol">
<span class="fa fa-pencil-square-o"></span>
</button>
<button type="button" class="btn btn-primary btn-sm tol" data-
toggle="modal" data-placement="top" title="Eliminar Ubicación" data-
target="#ModalEliminarUbicacion">
<span class="fa fa-trash-o"></span>
</button>
</div>
<div class="groups view" id="ajaxUbicacion">
<dl>
<dt><?php echo __('Id'); ?></dt>
<dd>
<?php echo h($ubicacion['Ubicacion']['id']); ?>
 
</dd>
<dt><?php echo __('Nombre'); ?></dt>
<dd>
<?php echo  h($ubicacion['Ubicacion']['nombre']); ?>
 
</dd>
<dt><?php echo __('Descripción'); ?></dt>
<dd>
<?php echo h($ubicacion['Ubicacion']['descripcion']); ?>
 
</dd>
<dt><?php echo __('Creado'); ?></dt>
<dd>
<?php echo h($ubicacion['Ubicacion']['created']); ?>
 
</dd>
<dt><?php echo __('Modificado'); ?></dt>
<dd>
<?php echo h($ubicacion['Ubicacion']['modified']); ?>
 
</dd>
<dt><?php echo __('Estado'); ?></dt>
<dd>
<?php if (h($ubicacion['Ubicacion']['estado'])=='1'){echo
```

```
__('Activo');}else{echo __('Inactivo');} ?>
 
</dd>
</dl>
</div>
</div>
</div>
<div class="col-md-12 ">
<div class="panel panel-default">
<div class="panel-heading">
<h3 class="panel-title">Dispocitivos</h3>
</div>
<div class="panel-body">
<?php if (!empty($ubicacion['Ubicaciondispocitivo'])): ?>
<div class="container col-md-12">
<ul class="row">
<?php foreach ($ubicacion['Ubicaciondispocitivo'] as $ubicaciondetalle): ?>
<li class="col-lg-4 col-md-5 col-sm-2 col-xs-3">
<?php if( $ubicaciondetalle['estado']==0){?>
<div class="panel  panel-danger">
<div class="panel-heading">
<button value="<?php echo $ubicaciondetalle['id']; ?>" type="button"
class="btn btn-primary btn-sm tol datosListaubicacion" data-toggle="modal"
data-placement="top" title="ubicacion" data-
target="#ModalVerubicacionUbicacio">
<span class="fa fa-search"></span>
</button>
<h6># <?php echo $ubicaciondetalle['numero']; ?></h6>
</div>
</div>
<?php } else {?>
<div class="panel  panel-success">
<div class="panel-heading">
<button value="<?php echo $ubicaciondetalle['id']; ?>" type="button"
class="btn btn-primary btn-sm tol datosListaubicacion" data-toggle="modal"
data-placement="top" title="ubicacion" data-
target="#ModalVerDispocitivoUbicacio">
<span class="fa fa-search"></span>
</button>
<h6># <?php echo $ubicaciondetalle['numero']; ?></h6>
</div>
</div>
<?php }?>
</li>
<?php endforeach; ?>
</ul>
</div>
<?php endif; ?>
</div>
</div>
</div>
</div>
<div class="modal fade" id="ModalEliminarUbicacion" tabindex="-1"
role="dialog" aria-labelledby="myModalLabel" aria-hidden="true">
<div class="modal-dialog modal-sm">
<div class="modal-content">
    <div class="modal-header">
<button type="button" class="close" data-dismiss="modal" aria-
```

```
hidden="true">&times;</button>
        <h4 class="modal-title">Eliminar Ubicación</h4>
      </div>
      <div class="modal-body">
        <p>Desea eliminar la Ubicación <?php echo
$ubicacion['Ubicacion']['nombre'] ?></p>
      </div>
      <div class="modal-footer">
<button type="button" id="btnCancelarEliminarUbicacion" class="btn btn-
default" data-dismiss="modal">Cancelar</button>
<?php echo $this->Form-
>button('Eliminar',array('type'=>'submit','id'=>'btnEliminarUbicacion','class'=
>'btn btn-primary','div'=>false)); ?>
      </div>
    </div><!-- /.modal-content -->
  </div><!-- /.modal-dialog -->
</div><!-- /.modal -->


<script type="text/javascript">
            $(function  () {
              $('.tol').tooltip();
              $('.operaciones').hide();
              $('#infoUbicacion').mouseover(function(){
                        $('.operaciones').show('slide');
              });
              $('#infoUbicacion').mouseleave(function(){
                        $('.operaciones').hide('scale');
              });

               $('.operacionesf').hide();
              $('#infoFoto').mouseover(function(){
                        $('.operacionesf').show('slide');
              });
              $('#infoFoto').mouseleave(function(){
                        $('.operacionesf').hide('scale');
              });

              $("#ModalEliminarUbicacion").on('hidden.bs.modal',function  (e) {
$('#btnRefrescarEliminarUbicacion').trigger('click');
$('#btnPanelUbicacions').trigger('click');
            });
 $(".datosListaubicacion").bind("click",function(){
 $.ajax({
async:true,
success:function (data, textStatus) {
$('#ajaxVerDispocitivoUbicacio').html(data);
},
type:"post",
url:"\/index.php\/Ubicaciondispocitivos\/viewdetalle\/"+$(this).val()
});
});
});
</script>
<?php
$this->Js->get('#btnEditarUbicacion');
$this->Js->event('click',
$this->Js->request(
```

```
array('action' => 'edit', $ubicacion['Ubicacion']['id']),
array('async' => true,
                    'update' => '#ajaxUbicacion',
                    'success'=>'$("#ajaxUbicacion").show("slide");',
                    'complete'=>'$(".operaciones").remove(); loderAremover();',
'beforeSend'=>'$("#ajaxUbicacion").hide("slide");loderA(".loderAjax2");'
                    )));
$this->Js->get('#btnEliminarUbicacion');
$this->Js->event('click',
$this->Js->request(
array('action' => 'delete', $ubicacion['Ubicacion']['id']),
array('async' => true,
          'method'=>'post',
          'success'=>'
          var a = eval("(" + data + ")");
          if(a["estado"]){
          $( "#btnCancelarEliminarUbicacion" ).trigger( "click" );
          $("#ModalEliminarUbicacion").on("hidden.bs.modal",function  (e) {
          $("#btnRefrescarUbicacion").trigger("click");
                                                                  });
                                                        }
          $("#alertas").append(a["html"]);
                              ')));
          ?>
<div class="modal fade" id="ModalVerDispocitivoUbicacio" tabindex="-1"
role="dialog" aria-labelledby="myModalLabel" aria-hidden="true">
  <div class="modal-dialog">
    <div class="modal-content">
      <div class="modal-header">
        <button type="button" class="close" data-dismiss="modal" aria-
hidden="true">&times;</button>
        <h4 class="modal-title">Ubicacion del Dispocitivo</h4>
      </div>
      <div class="modal-body" id="ajaxVerDispocitivoUbicacio">

      </div>
    </div><!-- /.modal-content -->
  </div><!-- /.modal-dialog -->
</div><!-- /.modal-dialog -->
```

Resultado de código de la vista view.ctp

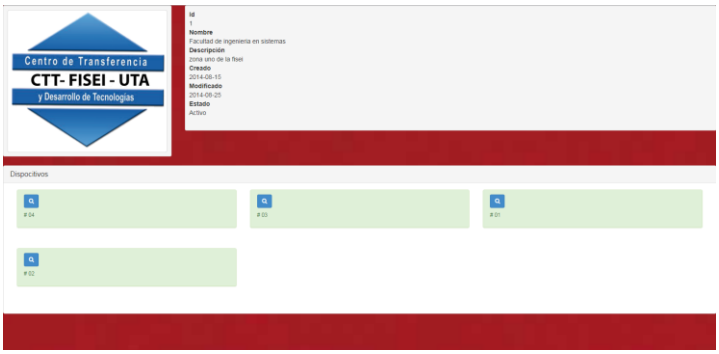

Figura 4.38: Resultado del código de la vista view
Elaborado por: Wilson Pérez – Investigador

4.12.2.3. **Controlador**

Responde a eventos usualmente acciones del usuario e invoca peticiones al modelo cuando se hace alguna solicitud sobre la información como editar un documento o un registro en una base de datos, también puede enviar comandos a su vista asociada si se solicita un cambio en la forma en que se presenta de modelo.

Fragmento de código del controlador de la aplicación web.

```
<?php
App::uses('AppController', 'Controller');
/**
 * Ubicacions Controller
 *
 * @property Ubicacion $Ubicacion
 * @property PaginatorComponent $Paginator
 */
class UbicacionsController extends AppController {

/**
 * Components
 *
 * @var array
 */
public $components = array('Paginator','RequestHandler');

/**
 * index method
 *
 * @return void
 */
public function index() {
                    $this->Paginator->settings = $this->paginate;
                    $this->Ubicacion->recursive = 1;
                    $this->set('ubicacions', $this->Paginator->paginate());
          }

/**
 * view method
 *
 * @throws NotFoundException
 * @param string $id
 * @return void
 */
public function view($id = null) {
if (!$this->Ubicacion->exists($id)) {
throw new NotFoundException(__('Invalid ubicacion'));
} $options = array('conditions' => array('Ubicacion.' . $this->Ubicacion-
>primaryKey => $id));
$this->set('ubicacion', $this->Ubicacion->find('first', $options));}
```

```
/**
 * add method
 *
 * @return void
 */
public function add() {
$a = array('estado' => FALSE, 'datos'=>null, 'html'=>null );
if ($this->request->is('post')) {
$this->autoRender = FALSE;
$this->Ubicacion->set($this->data);
if ($this->Ubicacion->validates()) {
$this->Ubicacion->create();
if ($this->Ubicacion->save($this->request->data)) {
$a['estado'] = true;
$a['html']=$this->mensajes("Ubicación ".$this->request-
>data['Ubicacion']['nombre']." Guardado", "correcto");
} else {
$a['html'] = $this->mensajes("No se guardo la Ubicación, intente
nuevamente", "error");
}
}
else {
$errors = $this->Ubicacion->validationErrors;
//print_r($errors["name"]);
foreach ($errors as $key => $var) {
$a['html'] .= $this->mensajes(" " . $var[0].PHP_EOL."\n", "advertencia");
}
}
return json_encode($a);
}
$fotos = $this->Ubicacion->Foto->find('list');
$this->set('images',$this->Ubicacion->Foto->find('all'));
$this->set(compact('fotos'));
}

/**
 * edit method
 *
 * @throws NotFoundException
 * @param string $id
 * @return void
 */
public function edit($id = null) {
$a = array('estado' => FALSE, 'datos'=>null, 'html'=>null );
if (!$this->Ubicacion->exists($id)) {
throw new NotFoundException(__('Invalid Ubicacion'));
}
if ($this->request->is(array('post', 'put'))) {
$this->Ubicacion->set($this->data);
$this->autoRender = FALSE;
if ($this->Ubicacion->validates()) {
if ($this->Ubicacion->save($this->request->data)) {
$a['estado'] = true;
$a['html']=$this->mensajes("Grupo ".$this->request-
>data['Ubicacion']['nombre']." Guardado", "correcto");
} else {
$a['html'] = $this->mensajes("No se guardo el Grupo, intente nuevamente",
"error");
```

```
}
}else{
$errors = $this->Ubicacion->validationErrors;
//print_r($errors["name"]);
foreach ($errors as $key => $var) {
$a['html'] .= $this->mensajes($key. " " . $var[0].PHP_EOL."\n",
"advertencia");
}
}
return json_encode($a);
} else {
$options = array('conditions' => array('Ubicacion.' . $this->Ubicacion-
>primaryKey => $id));
$this->request->data = $this->Ubicacion->find('first', $options);
$this->set('ubicacion', $this->request->data);
$this->set('images',$this->Ubicacion->Foto->find('all'));
}
$fotos = $this->Ubicacion->Foto->find('list');
$this->set(compact('fotos'));
}
/**
 * delete method
 *
 * @throws NotFoundException
 * @param string $id
 * @return void
 */
public function delete($id = null) {
$a = array('estado' => FALSE, 'datos'=>null, 'html'=>null );
$this->Ubicacion->id = $id;
$options = array('conditions' => array('Ubicacion.' . $this->Ubicacion-
>primaryKey => $id));
$this->request->data = $this->Ubicacion->find('first', $options);
if (!$this->Ubicacion->exists()) {
throw new NotFoundException(__('Invalid Ubicacion'));
}
if($this->request->is(array('post', 'delete'))){
$this->autoRender = false;
if(count($this->request->data['Ubicaciondispocitivo'])==0)
{
if ($this->Ubicacion->delete()) {
$a['estado'] = true;
$a['html']=$this->mensajes("Grupo ".$this->request-
>data['Ubicacion']['nombre']." Eliminado", "correcto");
} else {
$a['html']= $this->mensajes("No se pudo eliminar al Ubicacion ".$this-
>request->data['Ubicacion']['name'],"error");
}}
else {
$a['html'] = $this->mensajes("No se pudo eliminar al Ubicacion tiene
Dispocitivos", "advertencia");
}
return json_encode($a);
}}}
```

4.12.3. **Programación de arruino mega**

Descripción de los dispositivos electrónicos

Mostraremos los distintos dispositivos utilizados para la emisión y recepción de las señales para nuestra aplicación:

Sensores ultrasónicos descripción del funcionamiento

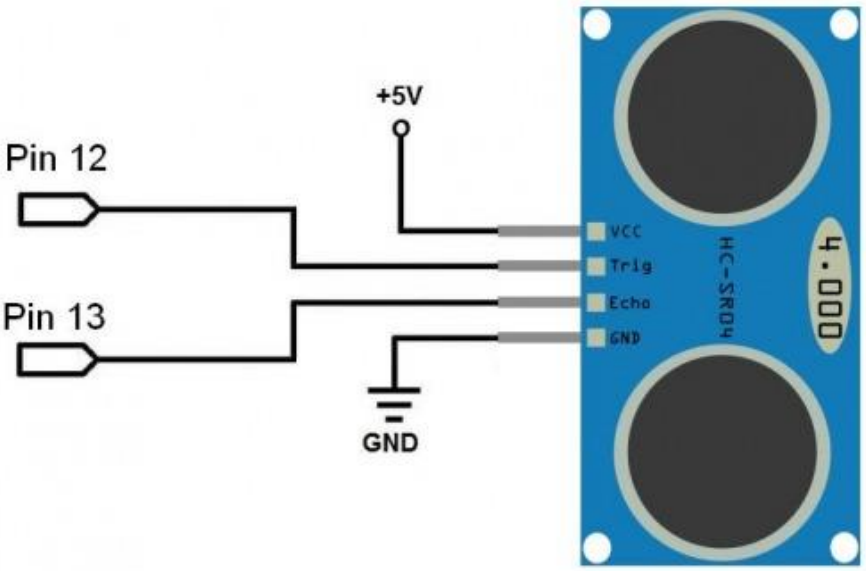

Figura 4.39: Estructura de sensor ultrasónico

Elaborado por: La Web

La imagen anterior muestra como está conformado el sensor ultrasónico el cual tenemos cuatro pines los cuales son utilizados para:

- El Vcc es utilizado para la alimentación del dispositivo
- El Trig el cual utilizamos para el envió de señales al sensor
- El Echo es el que ayuda a la recepción de señales desde sensor
- El GND que es masa o tierra

La forma de funcionamiento es como se muestra en la siguiente figura:

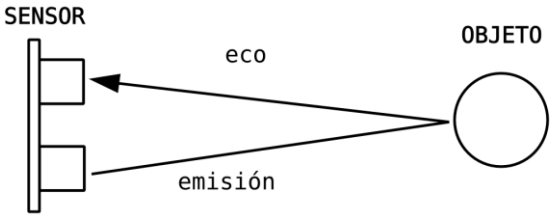

Figura 4.40: Funcionamiento del sensor ultrasónico

Elaborado por: La Web

Trabajando en conjunto con el arduino con el sensor se representa de la siguiente manera.

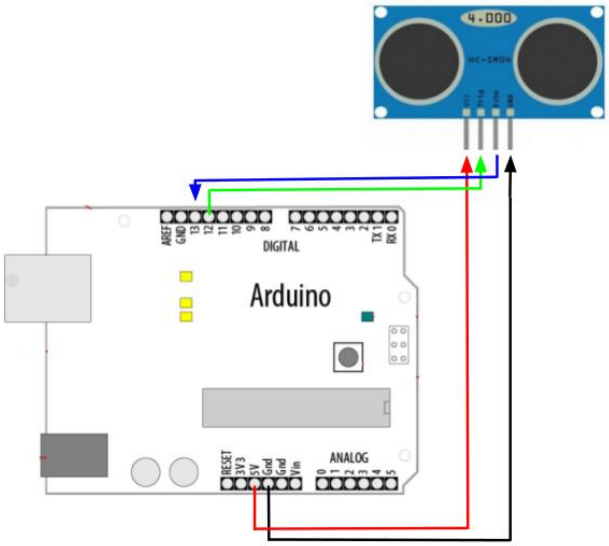

Figura 4.41 Sensor ultrasónico conexión con arduino

Elaborado por: La Web

El código de programación de arduino uno mega para la recepción de los dispositivos que ayudan al sensor de si hay o no espacios disponibles de parqueadero son los siguientes.

```
const int tris1=2;
const int echo1=38;
const int tris2=3;
const int echo2=40;
const int tris3=4;
const int echo3=42;
const int tris4=5;
const int echo4=44;
int am1=34;
int ver1=36;
int am2=30;
int ver2=32;

int am3=26;
int ver3=28;
int am4=22;
int ver4=24;
float dis1;
float dis2;
float dis3;
float dis4;
void setup(){
  Serial.begin(9600);
  pinMode(tris1,OUTPUT);
  pinMode(echo1,INPUT);
  pinMode(tris2,OUTPUT);
  pinMode(echo2,INPUT);
  pinMode(tris3,OUTPUT);
  pinMode(echo3,INPUT);
  pinMode(tris4,OUTPUT);
  pinMode(echo4,INPUT);
  pinMode(tris3,OUTPUT);
  pinMode(echo3,INPUT);
  pinMode(tris4,OUTPUT);
  pinMode(echo4,INPUT);
  digitalWrite(ver1, LOW);
  digitalWrite(ver2, LOW);
  digitalWrite(ver3, LOW);
  digitalWrite(ver4, LOW);
  digitalWrite(am1, LOW);
  digitalWrite(am2, LOW);
  digitalWrite(am3, LOW);
  digitalWrite(am4, LOW);

}

 String sensor1(){
  String valor="";
```

```
 digitalWrite(tris1,LOW);
 delayMicroseconds(5);
 digitalWrite(tris1,HIGH);
 delayMicroseconds(10);
 digitalWrite(tris1,LOW);
dis1=pulseIn(echo1,HIGH);
dis1=dis1*0.0001657;

 if ((dis1 < 0.15)&& (dis1>0.03))
   {
    digitalWrite(am1, HIGH);
    delay(10);
    valor+="01011101:";
    digitalWrite(ver1, LOW);
    delay(10);
   }
 else
   {
    digitalWrite(ver1, HIGH);
    delay(10);
    valor+="01010001:";
    digitalWrite(am1, LOW);
    delay(10);
   }
return valor;
 }

 String sensor2(){
 String valor="";
 digitalWrite(tris2,LOW);
 delayMicroseconds(5);
 digitalWrite(tris2,HIGH);
 delayMicroseconds(10);
 digitalWrite(tris2,LOW);
dis2=pulseIn(echo2,HIGH);
dis2=dis2*0.0001657;
if ((dis2 < 0.15)&& (dis2>0.03))
   {
    digitalWrite(am2, HIGH);
    delay(10);
    valor+="01021101:";
    digitalWrite(ver2, LOW);
    delay(10);
   }
 else
   {
    digitalWrite(ver2, HIGH);
    delay(10);
```

```
    valor+="01020001:";
    digitalWrite(am2, LOW);
    delay(10);
  }
return valor;
}


 String sensor3(){
 String valor="";
 digitalWrite(tris3,LOW);
 delayMicroseconds(5);
 digitalWrite(tris3,HIGH);
 delayMicroseconds(10);
 digitalWrite(tris3,LOW);
dis3=pulseIn(echo3,HIGH);
dis3=dis3*0.0001657;
if ((dis3 < 0.07)&& (dis3>0.03))
  {
    digitalWrite(am3, HIGH);
    delay(10);
     valor="01031102:";
    digitalWrite(ver3, LOW);
    delay(10);
  }
 else
  {
    digitalWrite(ver3, HIGH);
    delay(10);
     valor="01030002:";
    digitalWrite(am3, LOW);
    delay(10);
  }
return valor;
}


 String sensor4(){
 String valor="";
 digitalWrite(tris4,LOW);
 delayMicroseconds(5);
 digitalWrite(tris4,HIGH);
 delayMicroseconds(10);
 digitalWrite(tris4,LOW);
dis4=pulseIn(echo4,HIGH);
dis4=dis4*0.0001657;
```

```
if ((dis4 < 0.15)&& (dis4>0.03))
   {
    digitalWrite(am4, HIGH);
    delay(10);
    valor="01041103";
    digitalWrite(ver4, LOW);
    delay(10);
   }
 else
   {
    digitalWrite(ver4, HIGH);
    delay(10);
    valor="01040003";
    digitalWrite(am4, LOW);
    delay(10);
   }
 return valor;
 }

void loop(){

 String res="";
 res+=sensor1();
 res+=sensor2();
 res+=sensor3();
 res+=sensor4();
 Serial.println(res);
}
```

En el cual generamos una trama para que pueda leer la aplicación web y poder tratar la información en un lenguaje más tratarle para determinar los espacios ubicación y la disponibilidad del mismo así como el tipo de dispositivo utilizado para la obtención de la los datos.

La trama se forma de la siguiente manera:

Descripción	**Trama**	**Estados**
Los dos primeros dígitos son utilizados para identificar el tipo de dispositivo.	00	0 – 99
Los dos siguientes dígitos son utilizados para identificar el número de espacio de parqueadero.	00	0 - 99
Los dos siguientes dígitos representan el estado si está o no ocupado el espacio de parqueadero.	00	00
		11
Los dos últimos dígitos se utiliza para la sesión o ubicación en la cual está ubicado el dispositivo.	00	0 – 99

Tabla 4.42: Armado de trama de cada dispositivo

Elaborado por: Wilson Pérez – Investigador

Ejemplo:

De la trama del dispositivo ultrasónico utilizado para ver si un espacio está o no disponible de cada una de las secciones o ubicación de la institución.

Trama de espacio ocupado, que tiene el tipo uno con un número de espacio uno y que está ocupado y que pertenece a la sección o ubicación uno:

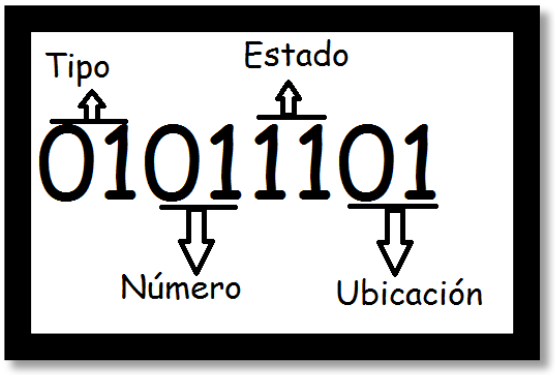

Figura 4.42: Tama de espacio ocupado

Elaborado por: Wilson Pérez – Investigador

Trama de espacio disponible, que tiene el tipo uno con un número de espacio uno y que está ocupado y que pertenece a la sección o ubicación uno:

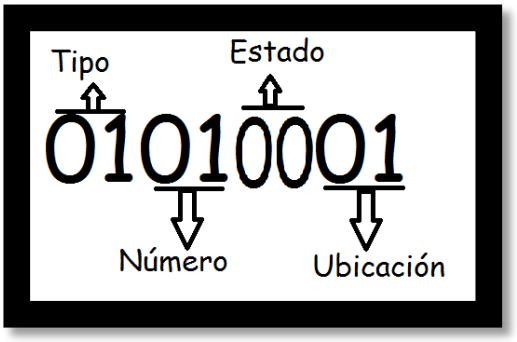

Figura 4.43: Tama de espacio ocupado

Elaborado por: Wilson Pérez – Investigador

Código de obtención de la trama desde la aplicación:

```
public function leerdatodispocitivo(){
$fp =fopen("/dev/ttyACM0", "r+");
if( !$fp) {
        die("error");
}
$valor=fgets($fp,500);
// echo $valor;
$this->loadModel('Tipo');
$this->loadModel('Dispocitivo');
$idUbcacionDispocitivo="";
$dispocitivo = explode(":", $valor);
for ($i=0; $i<count($dispocitivo); $i++) {
$codigo = substr($dispocitivo[$i],0,2);
$numero = substr($dispocitivo[$i],2,2);
$valor = substr($dispocitivo[$i],4,2);
$num = substr($dispocitivo[$i],6,2);
$idsss=$this->Ubicacion->find('first',array('conditions'=>array(
'Ubicacion.codigo'=>$num),
'fields'=>array('Ubicacion.id'),
'recursive'=>'-1'));
if(count($idsss)>0)
{
$ubi=$idsss['Ubicacion']['id'];
}
$idTipos= $this->Tipo->find('first',array('conditions'=>
array('Tipo.codigo'=>$codigo),
'fields'=>array('Tipo.id'),   'recursive'=>'-1' ));
if(count($idTipos)>=1){
$idsDispocitivos=$this->Dispocitivo->find('all',array('conditions'=>
array('Dispocitivo.tipo_id'=>$idTipos['Tipo']['id']),
```

```
'fields'=>array('Dispocitivo.id'),
'recursive'=>'-1'));
for($j=0;$j<count($idsDispocitivos);$j++)
{
$idsUbicacionDispocitivo=$this->Ubicaciondispocitivo->find('first',
array('conditions'=>array(
'Ubicaciondispocitivo.dispocitivo_id'=>$idsDispocitivos[$j]['Dispocitivo']['id'],
'Ubicaciondispocitivo.numero'=>$numero,
'Ubicaciondispocitivo.ubicacion_id'=>$ubi),
'fields'=>array('Ubicaciondispocitivo.id'),
'recursive'=>'-1'));
if(count($idsUbicacionDispocitivo)==1)
{
$idUbcacionDispocitivo=$idsUbicacionDispocitivo;
}}
$this->Ubicaciondispocitivo-
>read(null,$idUbcacionDispocitivo['Ubicaciondispocitivo']['id']);
$this->Ubicaciondispocitivo->set(array('valor'=>$valor));
$this->Ubicaciondispocitivo->save(); }}
$this->Session->setFlash(__($valor));
fclose($fp);
}
```

4.12.4. **Programación de arduino nano**

Se muestra en la figura siguiente como está formada la matriz de led:

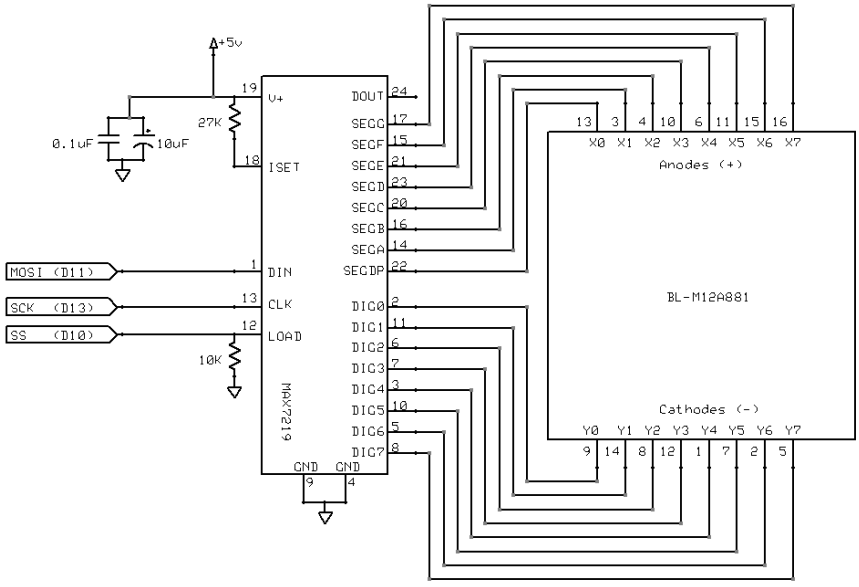

Figura 4.44: Circuito de matriz de leds.

Elaborado por: Internet

Describimos como funciona en conjunto con el arduino.

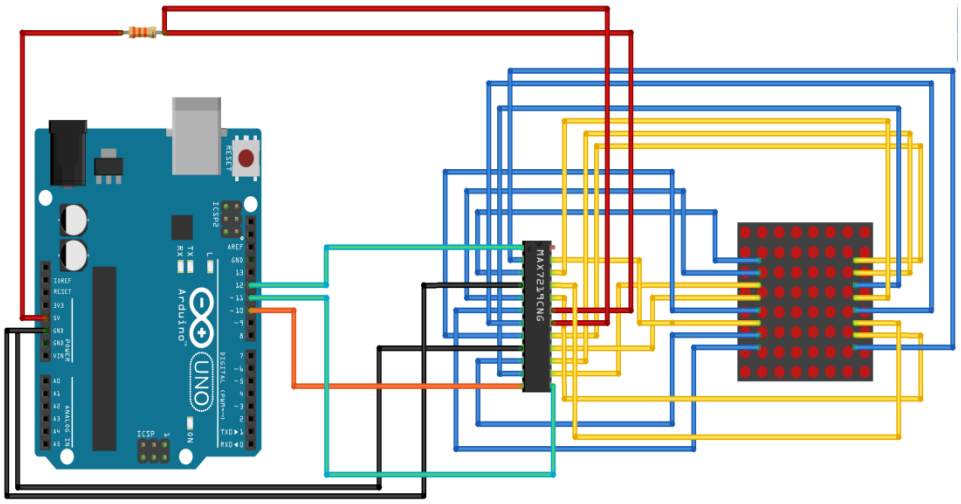

Figura 4.45: Matriz de leds y arduino.

Elaborado por: Internet

El código que se presenta a continuación permite mostrar en las pantallas leds para determinar las rutas en las cuales existen espacios disponibles de parqueaderos en la institución.

```
#include <avr/pgmspace.h>
#include <LedControl.h>
 const int numDevices = 1;      // number of MAX7219s used
const long scrollDelay = 75;   // adjust scrolling speed
String ahora;
unsigned long bufferLong [14] = {0};
LedControl lc=LedControl(9,10,11,numDevices);
unsigned long delaytime=60;
void setup() {
  lc.setIntensity(0,8);
  lc.clearDisplay(0);

  Serial.begin(9600);
```

```
}
void siguaizquirda() {
  for(int row=8;row>=0;row--) {
    if(row>=4)    {
       delay(delaytime);
       lc.setLed(0,row,5,true);
       lc.setLed(0,row,4,true);
       lc.setLed(0,row,3,true);
       lc.setLed(0,row,2,true);
       delay(delaytime);
    }
    else if(row==3)
    {
       delay(delaytime);
       lc.setLed(0,row,7,true);
       lc.setLed(0,row,6,true);
       lc.setLed(0,row,5,true);
       lc.setLed(0,row,4,true);
       lc.setLed(0,row,3,true);
       lc.setLed(0,row,2,true);
       lc.setLed(0,row,1,true);
       lc.setLed(0,row,0,true);
       delay(delaytime);
    }
    else if(row==2)
    {
       delay(delaytime);
        lc.setLed(0,row,6,true);
       lc.setLed(0,row,5,true);
       lc.setLed(0,row,4,true);
       lc.setLed(0,row,3,true);
       lc.setLed(0,row,2,true);
```

```
      lc.setLed(0,row,1,true);
      delay(delaytime);
    }
    else if(row==1)
    {
      delay(delaytime);
      lc.setLed(0,row,5,true);
      lc.setLed(0,row,4,true);
      lc.setLed(0,row,3,true);
      lc.setLed(0,row,2,true);
      delay(delaytime);
    }
    else if(row==0)
    {
      delay(delaytime);
       lc.setLed(0,row,4,true);
      lc.setLed(0,row,3,true);
      delay(delaytime);
    }
  }
}
void siguaderecha() {
  for(int row=0;row<=8;row++) {
    if(row<=3)
     {
      delay(delaytime);
      lc.setLed(0,row,5,true);
      lc.setLed(0,row,4,true);
      lc.setLed(0,row,3,true);
      lc.setLed(0,row,2,true);
      delay(delaytime);
    }
```

```
else if(row==4)
{
   delay(delaytime);
   lc.setLed(0,row,7,true);
   lc.setLed(0,row,6,true);
   lc.setLed(0,row,5,true);
   lc.setLed(0,row,4,true);
   lc.setLed(0,row,3,true);
   lc.setLed(0,row,2,true);
   lc.setLed(0,row,1,true);
   lc.setLed(0,row,0,true);
   delay(delaytime);
}
else if(row==5)
{
   delay(delaytime);
    lc.setLed(0,row,6,true);
   lc.setLed(0,row,5,true);
   lc.setLed(0,row,4,true);
   lc.setLed(0,row,3,true);
   lc.setLed(0,row,2,true);
   lc.setLed(0,row,1,true);
   delay(delaytime);
}
else if(row==6)
{
   delay(delaytime);
   lc.setLed(0,row,5,true);
   lc.setLed(0,row,4,true);
   lc.setLed(0,row,3,true);
   lc.setLed(0,row,2,true);
   delay(delaytime);
```

```
    }
    else if(row==7)
    {
       delay(delaytime);
        lc.setLed(0,row,4,true);
       lc.setLed(0,row,3,true);
       delay(delaytime);
    }
  }
}
void siguabajo() {
  for(int row=8;row>=0;row--) {
    if(row>3)
     {
       delay(delaytime);
       lc.setLed(0,5,row,true);
       lc.setLed(0,4,row,true);
       lc.setLed(0,3,row,true);
       lc.setLed(0,2,row,true);
       delay(delaytime);
    }
    else if(row==3)
    {
       delay(delaytime);
       lc.setLed(0,7,row,true);
       lc.setLed(0,6,row,true);
       lc.setLed(0,5,row,true);
       lc.setLed(0,4,row,true);
       lc.setLed(0,3,row,true);
       lc.setLed(0,2,row,true);
       lc.setLed(0,1,row,true);
       lc.setLed(0,0,row,true);
```

```
      delay(delaytime);
    }
    else if(row==2)
    {
      delay(delaytime);
       lc.setLed(0,6,row,true);
      lc.setLed(0,5,row,true);
      lc.setLed(0,4,row,true);
      lc.setLed(0,3,row,true);
      lc.setLed(0,2,row,true);
      lc.setLed(0,1,row,true);
      delay(delaytime);
    }
    else if(row==1)
    {
      delay(delaytime);
      lc.setLed(0,5,row,true);
      lc.setLed(0,4,row,true);
      lc.setLed(0,3,row,true);
      lc.setLed(0,2,row,true);
      delay(delaytime);
    }
    else if(row==0)
    {
      delay(delaytime);
       lc.setLed(0,4,row,true);
      lc.setLed(0,3,row,true);
      delay(delaytime);
    }

  }
}
```

```
void siguariba() {
  for(int row=0;row<=8;row++) {
    if(row<=3)
     {
        delay(delaytime);
        lc.setLed(0,5,row,true);
        lc.setLed(0,4,row,true);
        lc.setLed(0,3,row,true);
        lc.setLed(0,2,row,true);
        delay(delaytime);
    }
    else if(row==4)
    {
        delay(delaytime);
        lc.setLed(0,7,row,true);
        lc.setLed(0,6,row,true);
        lc.setLed(0,5,row,true);
        lc.setLed(0,4,row,true);
        lc.setLed(0,3,row,true);
        lc.setLed(0,2,row,true);
        lc.setLed(0,1,row,true);
        lc.setLed(0,0,row,true);
        delay(delaytime);
    }
    else if(row==5)
    {
        delay(delaytime);
         lc.setLed(0,6,row,true);
        lc.setLed(0,5,row,true);
        lc.setLed(0,4,row,true);
        lc.setLed(0,3,row,true);
        lc.setLed(0,2,row,true);
```

```
      lc.setLed(0,1,row,true);
      delay(delaytime);
    }
    else if(row==6)
    {
      delay(delaytime);
      lc.setLed(0,5,row,true);
      lc.setLed(0,4,row,true);
      lc.setLed(0,3,row,true);
      lc.setLed(0,2,row,true);
      delay(delaytime);
    }
    else if(row==7)
    {
      delay(delaytime);
       lc.setLed(0,4,row,true);
      lc.setLed(0,3,row,true);
      delay(delaytime);
    }
  }
}
void nosigua() {
      delay(delaytime);
      lc.setLed(0,0,1,true);
      lc.setLed(0,0,2,true);
      lc.setLed(0,0,3,true);
      lc.setLed(0,0,4,true);
      lc.setLed(0,0,5,true);
      lc.setLed(0,0,6,true);
      lc.setLed(0,7,1,true);
      lc.setLed(0,7,2,true);
      lc.setLed(0,7,3,true);
```

```
      lc.setLed(0,7,4,true);
      lc.setLed(0,7,5,true);
      lc.setLed(0,7,6,true);
      lc.setLed(0,1,0,true);
      lc.setLed(0,2,0,true);
      lc.setLed(0,3,0,true);
      lc.setLed(0,4,0,true);
      lc.setLed(0,5,0,true);
      lc.setLed(0,6,0,true);
      lc.setLed(0,1,7,true);
      lc.setLed(0,2,7,true);
      lc.setLed(0,3,7,true);
      lc.setLed(0,4,7,true);
      lc.setLed(0,5,7,true);
      lc.setLed(0,6,7,true);
      lc.setLed(0,1,6,true);
      lc.setLed(0,2,5,true);
      lc.setLed(0,3,4,true);
      lc.setLed(0,4,3,true);
      lc.setLed(0,5,2,true);
      lc.setLed(0,6,1,true);
      delay(delaytime);
}
void loop() {
  int stado, auxstado;
  int var = 0;
  int aux=0;
  while(var == 0){
    stado= Serial.read();
    if(stado != -1)
    {
      auxstado=stado;
```

```
      aux=aux+1;
     }
     if(aux>1){
      var=1;
     }
    if( auxstado==53)
     {
      nosigua();
     }
     if( auxstado==49)
     {
      lc.clearDisplay(0);
      siguaizquirda();
      lc.clearDisplay(0);
     }
     if( auxstado==50)    {
      lc.clearDisplay(0);
      siguaderecha();
      lc.clearDisplay(0);    }
     if( auxstado==51)    {
      lc.clearDisplay(0);
      siguabajo();
      lc.clearDisplay(0);    }
     if( auxstado==52)    {
      lc.clearDisplay(0);
      siguariba();
      lc.clearDisplay(0);   }
     delay(300);
    }
}
```

4.13. **Pruebas de funcionamiento**

Se las realiza con la finalidad de obtener un producto final de calidad se los desarrolla a nivel de funcionalidad y procesos de la aplicación o producto software desarrollado, con las cuales podemos descubrir defectos probando los componentes individuales.

Estos objetos o componentes pueden ser fusiones objetos que están dentro de la aplicación, demostrando que el sistema satisface los requerimientos del procesos que se está automatizando en el software, y que no se comporte de manera inesperada.

4.13.1. **Pruebas de caja blanca**

Este tipo de pruebas también se las denomina como pruebas de software ya que se realiza sobre las funciones internas de un módulo en el desarrollo de software.

4.13.2. **Pruebas del función de obtención de trama de dispositivos**

0	public function leerdatodispocitivo(){
1	$fp =fopen("/dev/ttyACM0", "r+");
2	if(!$fp)
3	{die("error"); }
4	else{ $valor=fgets($fp,500); $this->loadModel('Tipo'); $this->loadModel('Dispocitivo'); $idUbcacionDispocitivo=""; $dispocitivo = explode(":", $valor);
5	for ($i=0; $i<count($dispocitivo); $i++) {
6	$codigo = substr($dispocitivo[$i],0,2); $numero = substr($dispocitivo[$i],2,2); $valor = substr($dispocitivo[$i],4,2); $num = substr($dispocitivo[$i],6,2);
7	$idsss=$this->Ubicacion->find('first',array('conditions'=>array('Ubicacion.codigo'=>$num), 'fields'=>array('Ubicacion.id'), 'recursive'=>'-1'));
8	if(count($idsss)>0)

9	{$ubi=$idsss['Ubicacion']['id']; }
10	$idTipos= $this->Tipo->find('first',array('conditions'=> array('Tipo.codigo'=>$codigo), 'fields'=>array('Tipo.id'), 'recursive'=>'-1'));
11	if(count($idTipos)>=1)
12	{ $idsDispocitivos=$this->Dispocitivo->find('all',array('conditions'=> array('Dispocitivo.tipo_id'=>$idTipos['Tipo']['id']),'fields'=>array('Dispocitivo.id'),'recursive'=>'-1'));
13	for($j=0;$j<count($idsDispocitivos);$j++) {
14	$idsUbicacionDispocitivo=$this->Ubicaciondispocitivo->find('first', array('conditions'=>array('Ubicaciondispocitivo.dispocitivo_id'=> $idsDispocitivos[$j]['Dispocitivo']['id'], 'Ubicaciondispocitivo.numero'=>$numero, 'Ubicaciondispocitivo.ubicacion_id'=>$ubi), 'fields'=>array('Ubicaciondispocitivo.id'), 'recursive'=>'-1'));
15	if(count($idsUbicacionDispocitivo)==1) {
16	$idUbcacionDispocitivo=$idsUbicacionDispocitivo; }}
17	$this->Ubicaciondispocitivo->read(null,$idUbcacionDispocitivo['Ubicaciondispocitivo']['id']); $this->Ubicaciondispocitivo->set(array('valor'=>$valor)); $this->Ubicaciondispocitivo->save(); }}
18	$this->Session->setFlash(__($valor));}
19	fclose($fp);
20	}

Tabla 4.43: Pruebas caja blanca del función leerdatodispocitivo

Elaborado por: Wilson Pérez – Investigador

4.1.1. **Grafo prueba obtención**

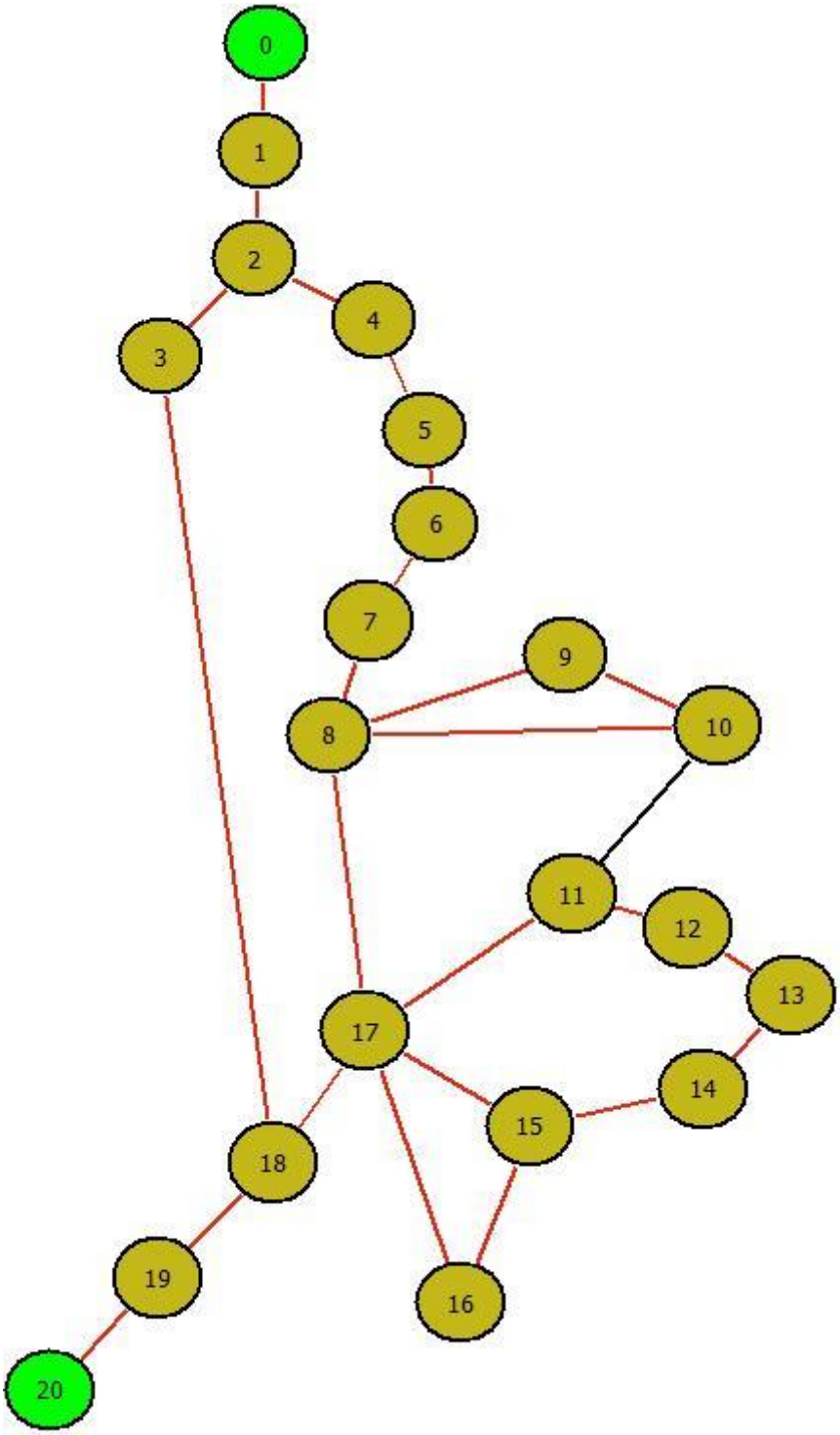

Figura 4.46: Grafo de obtención de trama prueba de caja blanca.

Elaborado por: Wilson Pérez – Investigador

Complejidad Ciclomática

V (G)= A (arista) – N (nodos)+2

V (G) = 25 – 20 + 2

V (G)= 3

Caminos Básicos

N	Nodos	# nodos
1	0-1-2-3-18-19-20	7
2	0-1-2-3-4-5-6-7-8-9-10-11-17-18-19-20	15
3	0-1-2-3-4-5-6-7-8-9-10-11-12-13-14-15-16-17-18-19-20	20

Tabla 4.44: Camino Básico.

Elaborado por: Wilson Pérez – Investigador

Comprobación de la prueba del camino básico

Se tomara como referencia el camino N:1	
0	Función leerdatodispocitivo
1	Asignar a una variable el archivo del puerto com.
2	Preguntamos si el archivo se asignó correctamente.
3	Error el archivo no se asignó correctamente, mensaje de error.
18	Imprimimos el mensaje
19	Cerramos todo
20	Termínanos el proceso.

Tabla 4.45: Comprobación del Camino Básico.

Elaborado por: Wilson Pérez – Investigador

4.1.2. **Pruebas de caja negra**

Las pruebas de caja negra ejercitan los requisitos funcionales desde el exterior de cada módulo es decir desde su interfaz.

Para la comprobación de dicha prueba se ha contado con la participación de la persona encargada de manejar la aplicación web, el cual se maneja el proceso de control de espacios disponibles de parqueaderos:

- Entrada y salida de datos
- Comprobación de que cada proceso cumpla con a los requerimientos de ingreso, eliminación, actualización y selección.
- Que la información sea mostrada correctamente

Dentro de los ingresos, eliminación, actualización las pruebas de caja negra se basan en la verificación de datos que se envían por medio de la interfaz mediante una petición de ingreso de datas necesarias para alterar la información de la base de datos.

En el proceso de selección las pruebas se basan en la presentación de la información en los componentes utilizados para mostrar la información mostrando de manera fácil e intuitiva para el usuario.

Ejemplo de pruebas de caja negra utilizando un procesos de la aplicación web que es el tratamiento de ubicaciones o espacios de parqueadero de la institución.

Ubicaciones		
Acciones	**Casos esperado**	**Resultados Obtenidos**
Ingreso	Mensajes de error	Mensajes de error describiendo el motivo del error y la solución.
	Información agregada correctamente	Mensaje de confirmación de inserción.
Actualización	Mensajes de error	Mensajes de error describiendo el motivo del error y la solución.
	Información actualizada correctamente	Información actualizada más mensaje de confirmación de actualización.
Eliminación	Mensajes de error	Mensajes de error describiendo el motivo del error y la solución.
	Datos eliminados correctamente	Datos eliminados más mensaje de confirmación de eliminación.
Selección	Despliegue de la información del proceso.	Muestra la información de los procesos.

Tabla 4.46: Pruebas de caja negra Ubicaciones

Elaborado por: Wilson Pérez – Investigador

Demostración de las pruebas de caja negra y sus interfaces

El resultado esperado dependiendo si solos datos están correctos o esta vacíos no presenta la información esperada del mensaje de error con una descripción de la solución.

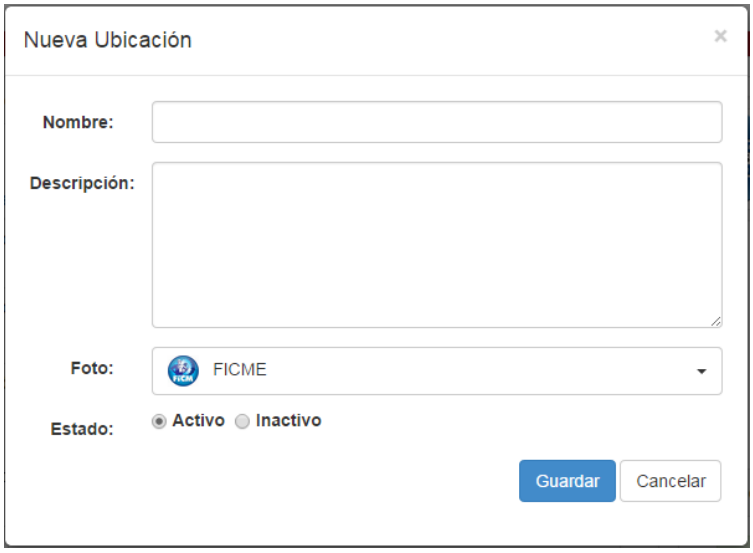

Figura 4.47: Pruebas de caja negra Ubicaciones agregar.

Elaborado por: Wilson Pérez – Investigador

Mensaje de erro con la solución o indicador del error.

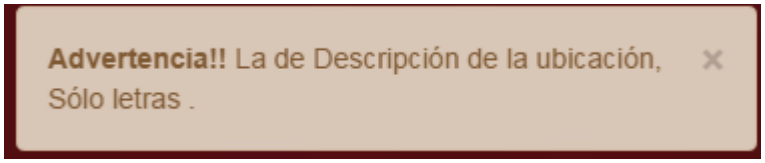

Figura 4.48: Menaje de error esperado en la agregación.

Elaborado por: Wilson Pérez – Investigador

Para el caso en el toda la información ingresado solo tendremos un mensaje de confirmación de ingreso de datos.

Figura 4.49: Mensaje de confirmación esperado por la agregación

Elaborado por: Wilson Pérez – Investigador

Para la actualización de la información se tiene los siguientes casos:

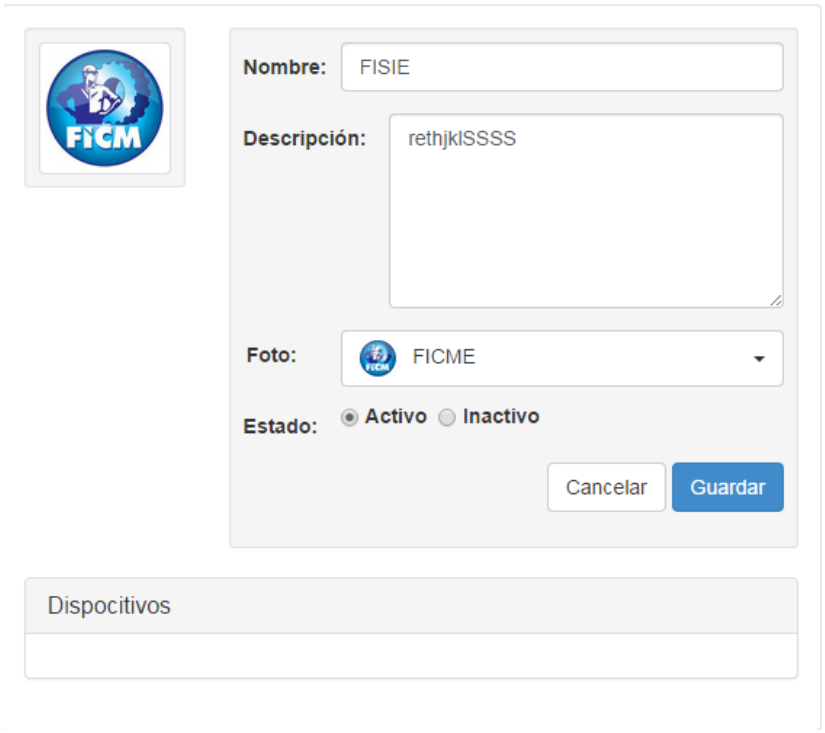

Figura 4.50: Pruebas de caja negra Ubicaciones edición

Elaborado por: Wilson Pérez – Investigador

Mensaje de error esperado cundo existe un error en la actualización o edición de información

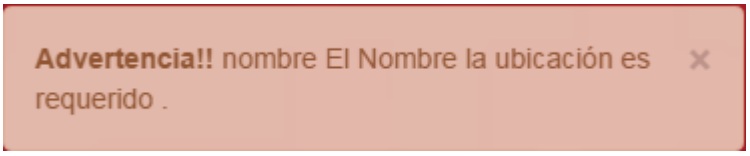

Figura 4.51: Mensaje de error esperado en la actualización

Elaborado por: Wilson Pérez – Investigador

Para cuando todo este correcto se muestra el mensaje de que se guardó correctamente.

Figura 4.52: Mensaje esperado en la confirmación de la actualización o edición

Elaborado por: Wilson Pérez – Investigador

Para el caso de elijación de información se muestra un cuadro de dialogo de confirmación.

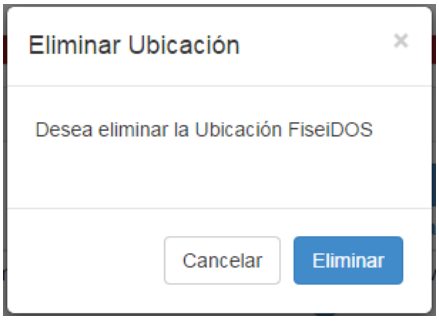

Figura 4.53: Cuadro de diálogo de pruebas de caja negra de eliminación

Elaborado por: Wilson Pérez – Investigador

Mostrando un mensaje de eliminación de información correcta.

Figura 4.54: Mensaje esperado de eliminación de ubicación

Elaborado por: Wilson Pérez – Investigador

El despliegue de información se lo muestra de la siguiente manera en el elemento u objeto de despliegue de información.

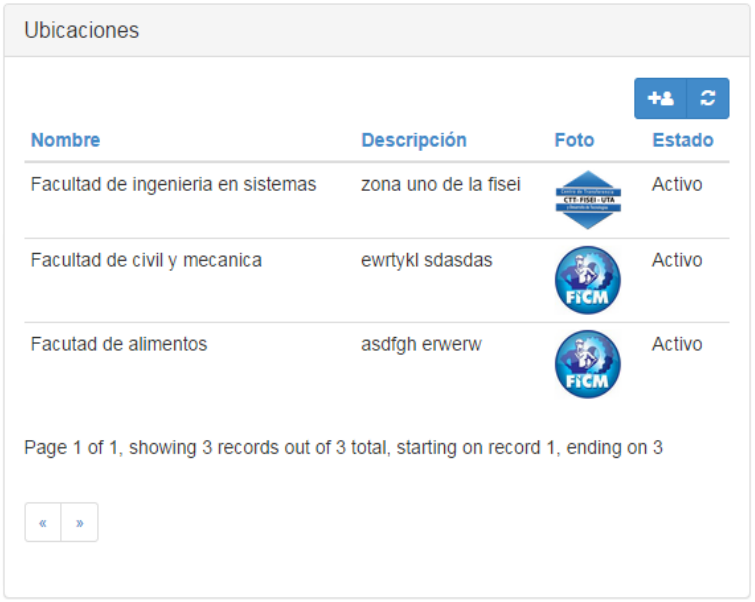

Figura 4.55: Pruebas de caja negra Ubicaciones selección

Elaborado por: Wilson Pérez – Investigador

CAPÍTULO V

5. CONCLUSIONES Y RECOMENDACIONES

5.1. **Conclusiones**

- El análisis mediante fichas de observación y acercamiento con el personal responsable del proceso de parqueo en la Universidad Técnica De Ambato campus Huachi Chico, permitieron conocer la inexistencia de un medio de distribución de espacios de parqueadero.

- El estándar EEE 830, de levantamientos de requerimientos permitió realizar un análisis y diseño de la aplicación web, con mayor rapidez y fidelidad.

- La elección y utilización de MVC en PHP, facilito la realización de la aplicación web, permitiendo un fácil desarrollo del mismo haciendo que sea intuitiva y con un diseño amigable para el usuario final.

- Las pruebas de la aplicación web ayudaron a probar el funcionamiento funcional del mismo del mismo tanto a nivel de producto final y desarrollo del mismo.

- El desarrollo de la aplicación permitirá automatizar el proceso de distribución de espacios disponibles de estacionamiento, de tal manera que se ahorre tiempo para el usuario, al buscar un sitio ya que podrá guiarse por indicadores que ayudaran a localizar un lugar disponible, también ayudara en la administración del mismo por el personal encargado del parqueadero de la institución.

5.2. **Recomendaciones**

- Capacitación al personal encargado de administrar la aplicación web, para su correcta utilización.

- Los usuarios de la aplicación web deben ser cuidadosos en el manejo de sus contraseñas con el fin de prevenir manipulación inadecuada de la información.

- Se recomienda a la persona encargada de la administración de la aplicación web, obtener respaldos periódicos de la base de datos para evitar pérdidas de información.

- Para un mejor funcionamiento de la aplicación web los usuarios clientes deben utilizar de preferencia Mozilla Firefox, Chrome como navegadores predeterminados.

- Utilizar por cada espacio de estacionamiento utilizar un dispositivo que facilite determinar la disponibilidad del mismo.

- Configurar los dispositivos electrónicos en horarios que no afecten el desempeño del mismo.

- Para la implantación de la aplicación web, es recomendable de una red LAN para la ubicación de los equipos de control del mismo, así como determinara los tipos de dispositivos de acuerdo al entorno de la sección de parqueadero.

Bibliografía

[1] A. E. Pérez Bautista, "Sistema web integrado para la gestión de cobranza de valores en el Gobierno Provincial de Tungurahua," Amabto, 2012.

[2] L. M. Viteri Medina, "Servicio Web para realizar transferencias de fondos en línea entre Instituciones financieras del país a través del Banco Central del Ecuador para la Cooperativa de Ahorro y Crédito "Chibuleo Ltda. De la Ciudad de Ambato," Ambato, 2013.

[3] N. S. Nise, «freelibros,» 08 02 2012. Disponible web: http://www.freelibros.org/electronica/sistemas-de-control-para-ingenieria-3ra-edicion-norman-s-nise.html. [Último acceso: 12 07 2013].

[4] X. A. Brotons, "upcommons.upc.edu," 2012. Disponible web: https://upcommons.upc.edu/pfc/bitstream/2099.1/3330/5/34059-5.pdf. [Accedida 12 06 2013].

[5] I. J. R. Vignoni, "ing.unlp.edu.ar," 2002. Disponible web: http://www.ing.unlp.edu.ar/electrotecnia/procesos/transparencia/Control_de_Procesos.pdf. [Accedida 12 06 2013].

[6] P. Moreno, "Mavainsa," 03 2011. Disponible web: http://pastranamoreno.files.wordpress.com/2011/03/control_procesos-valvulas.pdf. [Accedida 12 06 2013].

[7] L. A. S. A. Villegas, "exa.unne.edu.ar," Disponible web: http://exa.unne.edu.ar/informatica/sistemas.adm1/material/tema-7.pdf. [Accedida 15 06 2013].

[8] 14 07 2010. Disponible web: http://seguridadensistemascomputacionale.org/Control%20De%20Acceso%20En%20Los%20Sistemas%20Computacionales.pdf. [Accedida 15 06 2013].

[9] 2011. Junta de Andalucia, sistemas Automaticos Disponible web: http://www.juntadeandalucia.es/averroes/~23005153/d_tecnologia/bajables/2%20bachillerato/SISTEMAS%20AUTOMATICOS%20DE%20CONTROL.pdf. [Accedida 16 06 2013].

[10] i. industriales, "ingenierosindustriales," Disponible web: http://ingenierosindustriales.jimdo.com. [Accedida 16 06 2013].

[11] S. Mesoamérica, "SIEMENS," siemens, 2008. Disponible web: http://industria.siemens.com.mx/Trafico%20dos/HTML/3_2_5.html. [Accedida 20 06 2013].

[12] A. S. Foundation., "Apache," 1997-2014. Disponible web: http://httpd.apache.org/.

[13] S. Lujan, Programacion de aplicaciones web: historia, principios basicos y clientes web, España: Club Universitario, 2002.

[14] PHP, "Web php," 2008. Disponible web: http://php.net/manual/es/language.oop5.php. [Accedida 27 06 2013].

[15] postgresql, "web postgresql," 23 10 2002. Disponible web: http://www.postgresql.org.es/. [Accedida 28 06 2013].

[16] IEEE, 22 10 2008. Disponible web: http://www.fdi.ucm.es/profesor/gmendez/docs/is0809/ieee830.pdf. [Accedida 26 06 2013].

[17] G. d. Magdalena, "Anomimo," Disponible web: http://www.magdalena.gov.co/apc-aa-files/61306630636336616166653232336536/manual_de_procesos_y_procedimientos.pdf. [Accedida 12 06 2013].

GLOSARIO DE TÉRMINOS

APW: Aplicación Web

ERS: Especificación de Requisitos Software

SCAFFOLDING: Scaffolding es un método para construir aplicaciones basadas en bases de datos, esta técnica está soportada por algunos Frameworks del tipo MVC

MVC: Modelo vista controlador, es un patrón de arquitectura de software que separa los datos y la lógica de negocio de una aplicación de la interfaz de usuario.

CAMELCASE: Es un estilo de escritura que se aplica a palabras compuestas.

MODEL: Contiene el núcleo de la funcionalidad de la aplicación.

VISTA: Es la presentación del Modelo, puede acceder al Modelo pero nunca cambiar su estado, puede ser notificada cuando hay un cambio de estado en el Modelo.

CONTROLLER: Reacciona a la petición del Cliente, ejecutando la acción adecuada y creando el modelo pertinente.

POO: Programación Orientado a Objetos

ARDUINO: es una plataforma de hardware libre, basada en una placa con un microcontrolador y un entorno de desarrollo, diseñada para facilitar el uso de la electrónica en proyectos multidisciplinares.

TRAMA: Es una secuencia de números que se utiliza para leer los estados de los dispositivos electrónicos.

SENSOR ULTRASÓNICO: son detectores de proximidad que trabajan libres de roces mecánicos y que detectan objetos a distancias de hasta 8m.

VCC: Alimentación positiva del circuito, es el voltaje de entrada o de alimentación de cierto circuito.

GND: Alimentación negativa del circuito.

ANEXOS

ANEXO #1: FICHA DE OBSERVACIÓN PROCESOS MANUAL DE ASIGNACIÓN

FICHA DE OBSERVACIÓN 1 ÁREA: Universidad Técnica De Ambato campus Huachi Chico. Observador: Investigador	
Objetivo: Analizar el proceso manual de control de espacios disponibles de parqueaderos Observaciones	
Como es el proceso manual de control de lugares de estacionamiento.	Como, información guardada, resultados obtenidos.
Observaciones:	

ANEXO #2: FICHA DE OBSERVACIÓN MÉTODOS DE ASIGNACIÓN

FICHA DE OBSERVACIÓN 2 ÁREA: Universidad Técnica De Ambato campus Huachi Chico Observador: Investigador	
Objetivo: Determinar los métodos de asignación existentes de espacios de parqueo Observación	
Métodos existentes	
Ventajas	
Desventajas	
Problemas ocasionados	
Observaciones:	

ANEXO #3: FOTOS DESARROLLO

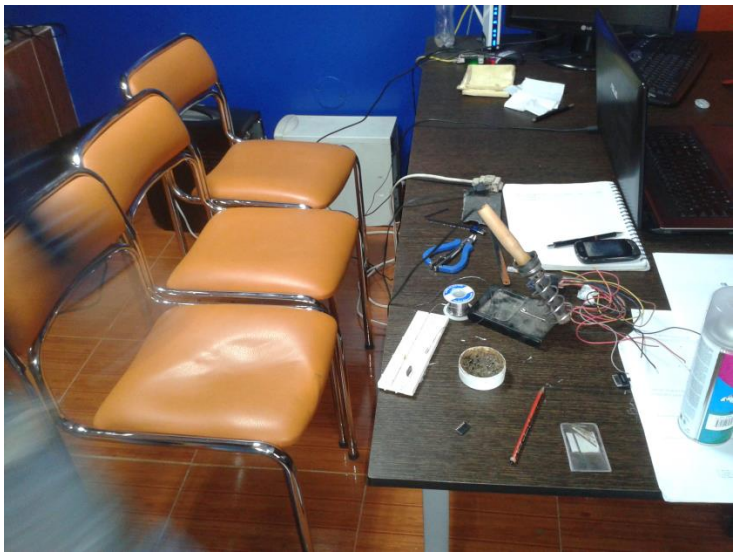

Foto 1: Equipos
Fuente: Wilson Pérez-Investigador

Foto 2: Instalación dispositivos
Fuente: Wilson Pérez-Investigador

Foto 3: Funcionamiento de matrices led
Fuente: Wilson Pérez-Investigador

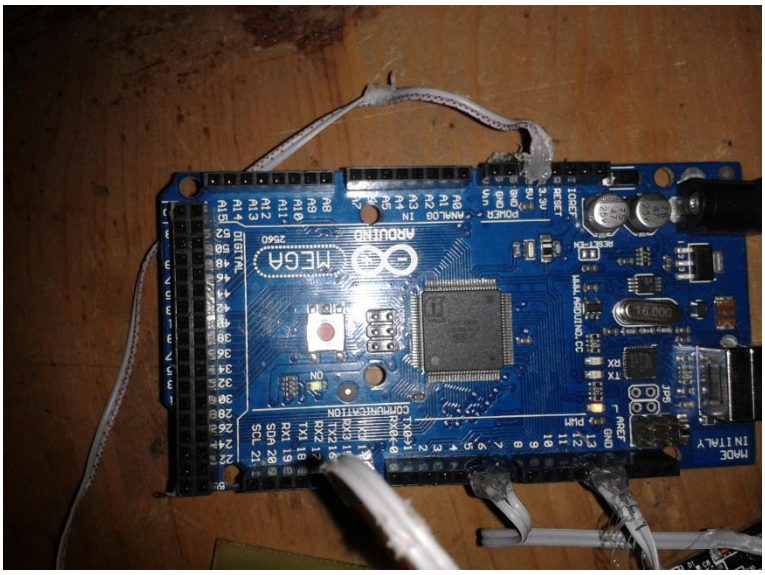

Foto 4: Funcionamiento de matrices led
Fuente: Wilson Pérez-Investigador