PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR

FACULTAD DE INGENIERÍA

ESCUELA DE SISTEMAS

DISERTACIÓN PREVIA A LA OBTENCIÓN DEL TÍTULO DE INGENIERA EN SISTEMAS Y COMPUTACIÓN

ANÁLISIS, DESARROLLO E IMPLEMENTACIÓN DE UN SISTEMA WEB DE ADMINISTRACIÓN DE UNA RESIDENCIA UNIVERSITARIA. CASO DE ESTUDIO: RESIDENCIA UNIVERSITARIA TULPA

RAQUEL MISHELLE ZAMBONINO GÓMEZ

DIRECTOR: FABIÁN DE LA CRUZ DOMÍNGUEZ

QUITO, 2020

Dedicatoria

Quiero dedicar este trabajo de disertación a mi bisabuela, por siempre confiar en mí y motivarme a cumplir con todos los objetivos que me proponga, por siempre estar a mi lado para poder superar los obstáculos. A mi hermana, por siempre motivarme en cada etapa de mi vida y sobre todo en el desarrollo de este trabajo de disertación.

Agradecimiento

Quiero agradecer a mi familia, por siempre apoyarme a cumplir con mis sueños y estar presentas en cada etapa, además por brindarme los recursos necesarios para culminar con mis estudios en la universidad. A mis profesores, por enseñarme todo lo que sé, que me sirve y me servirá para mi vida profesional y personal. A mi director de tesis y revisores por guiarme y ser un apoyo en la realización de este trabajo.

Tabla de contenido

Índice de Figuras

Índice de Tablas

Resumen

En vista de la petición presentada por la Residencia Universitaria Tulpa de automatizar el registro de residentes, así como el control de sus actividades dentro de la residencia para poder mejorar su servicio, se decidió tomar esta organización como caso de estudio para la presente disertación.

Para esto, se definieron los requerimientos funcionales, dando así un alcance que tendrá la aplicación. Con las funcionalidades definidas, se procedió a diseñar la arquitectura de la solución que fue implementada como parte de esta disertación. Una vez terminado el desarrollo del sistema, se realizaron las pruebas correspondientes con las personas que usarán el sistema en la residencia, en este punto se resolvieron algunos defectos encontrados.

Se hicieron los ajustes al sistema hasta que los usuarios lo aprobaron y se dio por finalizado el trabajo de titulación, con su respectiva documentación.

Abstract

Due to the request presented by “Residencia Universitaria Tulpa” to automate the registration of residents as well as the control of their activities in order to improve their service, I decided to take this organization as a case study for this dissertation.

To achieve this goal, the functional requirements were defined, thus giving a scope that the application will have. With the defined functionalities, I proceeded to design the architecture of the solution that was implemented as part of this dissertation. Once the development of the system was finished, I tested the system with the people who will use it in the residence, at this point some defects were found and solved.

The adjustments to the system were made until the users approved it and the degree work was completed, with its respective documentation.

Glosario:

Trabajo de disertación:

1. CAPÍTULO 1: Introducción

En el primer capítulo, se presenta la introducción al trabajo de disertación, en la que detalla un problema identificado en el caso de estudio. Se detallan los objetivos a cumplir a lo largo de este trabajo, además de un corto resumen de lo que se realizó en cada capítulo. Esta información nos ayudará a entender de manera general el tema a tratar y sus diferentes puntos.

1.1 Justificación:

Tulpa es una residencia universitaria, la cual no maneja ningún sistema informático para su administración que incluye: Reservas de habitaciones y registro de actividades internas. El presente trabajo de titulación se realizará con la intención de facilitar dicha administración. Al finalizar el proyecto, se espera que el sistema presente ventajas tanto para las residentes, que tendrán más facilidad para el registro de las actividades, como a los directivos, y a las personas externas que realizan reservas. Lo cual, se podrá realizar desde cualquier lugar con conexión a Internet.

1.2 Planteamiento del problema:

Dado que la residencia carece de un sistema informático, ha venido causando diversos inconvenientes, ya que las personas no están al tanto de la información actualizada de la residencia y que por medio de un sistema centralizado todos pueden acceder a él. Otro

problema, es el manejo de registros de las comidas y de las actividades internas. El estado actual de la residencia hace que las residentes tengan que registrar sus asistencias en hojas, muchas veces se olvidan o cambian de planes y al no encontrarse en la residencia, no pueden cambiar horarios y la organización se ve afectada.

Todos los inconvenientes escritos anteriormente se podrían mejorar con la aplicación antes descrita. Al ser la aplicación web dicha gestión se podría realizar desde cualquier lugar sin inconvenientes. Consecuentemente la intención es tener información actualizada para tener un mejor manejo de la residencia y una mayor facilidad de acceso. Además, se intentará distribuir las actividades a diferentes roles, que son los administrativos y las residentes.

1.3 Objetivos:

1.3.1 Objetivo general:

Analizar, desarrollar e implementar un sistema web de administración de una residencia universitaria. Caso de estudio: Residencia Universitaria Tulpa

1.3.2 Objetivos específicos:

- Analizar, diseñar y desarrollar un sistema de acuerdo a la metodología escogida previamente.
- Implementar el sistema basándose en los estudios realizados previamente.
- Probar el funcionamiento del sistema corrigiendo posibles errores hasta que el sistema cumpla con las necesidades de la residencia.

1.4 Contenido

En el Capítulo 2 se desarrolló la parte teórica del proyecto de disertación, se incluye una breve descripción del caso de estudio, la Residencia Universitaria Tulpa, esto ayudó a conocer la situación de la organización, así como a entender de una mejor manera sus procesos para poder automatizarlos de la manera más efectiva.

Además, se incluyó información de las herramientas que se van a usar para el desarrollo, lo cual ayudó a familiarizarse de mejor manera con sus características. En cuanto a la metodología usada, se realizó una breve explicación de cada una de sus fases, una vez descritas se definieron los entregables.

En el Capítulo 3 se desarrolló la fase de diseño e implementación de la metodología cascada. Dentro de este capítulo, se encuentra el documento de especificación de requerimientos, es decir los diagramas de secuencia general y a detalle. Lo primero que se hizo fue definir, con la ayuda de la persona que maneja la residencia, las funcionalidades que tendrá la aplicación según el alcance del proyecto y las necesidades existentes; de este proceso se obtuvo el diagrama general de casos de uso, con un total de 14 funcionalidades. Además, se definieron los roles que van a intervenir en la aplicación, éstos son: Administrador que maneja todas las funcionalidades, y residente que interactúa con la funcionalidad de actividades.

A continuación, se definió a detalle lo que debe hacer cada funcionalidad. Dado que las diez primeras funcionalidades pertenecen al manejo de catálogos, en todas ellas se encontraron dos flujos: Ingresar y modificar. No se incluyó la opción de eliminar porque se desea almacenar registros históricos, y si se opta por esta alternativa, esto implica borrar

los registros de residentes y numerarias que tengan estos catálogos, sin embargo, se incluyó la opción de activar o desactivar los valores según sea su necesidad.

En cuanto al manejo de residentes, se decidió separar el manejo de las personas que ya han ido a una entrevista pero aún no confirman su estadía en la residencia y las personas que ya viven en la casa o que ya han confirmado su fecha de mudanza, cada una de ellas se maneja en una pantalla independiente, compartiendo los mismos datos, como son: Información personal, de sus estudios y de sus padres. Lo que les diferencia es que la residente tiene ya asignado un cuarto para vivir, la posible residente no lo tiene.

Por otro lado, el manejo de las numerarias se decidió separar del grupo de residentes, dado que no comparten todas sus características. Por un lado, cuando una numeraria llega a la casa no es necesario obtener la información de los padres, y por otro, a diferencia de las residentes que son solo estudiantes universitarias, las numerarias pueden ser estudiantes o tener una ocupación y un lugar de trabajo.

Después de definir el comportamiento del sistema, se procedió a construir el documento de especificación de diseño, en el cual se definieron los diagramas de secuencia, clases y entidad relación. Esto nos da una guía para el desarrollo de la aplicación. Tanto del manejo de las tres capas del patrón MVC (Modelo Vista Controlador), así como el manejo de datos en la Base de Datos.

En el Capítulo 4, después de realizar los diagramas, se procedió con la implementación según lo descrito, se incluyen algunos fragmentos de código, los más representativos. Una vez terminado el proceso de desarrollo, se realizaron las respectivas pruebas con la parte interesada, en este caso dos de las personas que van a usar la aplicación, se tuvieron dos

reuniones de pruebas, en las cuales se corrigieron algunos errores encontrados. Finalizada esta etapa, la administradora de la residencia firmó un documento de aceptación del sistema.

En este capítulo se incluye también el entregable de la fase de implantación y mantenimiento de la metodología cascada. Se tomó como referencia el documento usado para reportar errores de TSP (Team Software Process). Este documento debe usar en caso de que se encuentre algún error en la aplicación después de su implantación, el cuál entra como un control de cambios para que pueda ser revisado, solucionado, y validado.

En el Capítulo 5, finalizadas las fases de análisis, desarrollo e implementación de la aplicación, y basándose en la experiencia obtenida durante el presente trabajo de disertación, se incluyen las respectivas conclusiones, además de algunas recomendaciones a la Residencia Universitaria Tulpa para mejorar otros procesos y complementar los que ya están automatizados.

1.5 Glosario

1. Trabajo de disertación: Trabajo académico compuesto de introducción, desarrollo y conclusión que busca la respuesta de un planteamiento propuesto.
2. Metodología: Consiste en un conjunto de normas o procedimientos a seguir dentro de una investigación o estudio.
3. Lenguaje de programación: Herramienta que permite la construcción de sistemas informáticos mediante la escritura de una secuencia de instrucciones.
4. Motor de base de datos: Servicio que permite procesar los datos, según las peticiones que se requiera.
5. Framework: Es un entorno de trabajo que facilita la programación, gracias a sus módulos y artefactos que permiten una buena organización del código.
6. Sistema informático: Es un conjunto de partes de software y hardware que permiten procesar datos y almacenarlos.
7. Patrón de diseño: Son estándares diseñados para resolver problemas de diseño encontrados dentro del desarrollo de software.
8. UML: Es un lenguaje estándar para poder realizar una variedad de diagramas que ayudan en la definición de un sistema informático.
9. Sistema escalable: Permite que el software vaya creciendo y adaptándose a las necesidades sin perder su calidad.
10. Sistema mantenible: Significa que es fácil poder modificar componentes o corregir fallas dentro del sistema.
11. Usabilidad: Es la facilidad que tenga el usuario final para poder usar el software desarrollado, si es fácil, se puede decir que es usable.

12. Hosting: Es un servicio que permite alojar una aplicación o sistema informático, además de sus datos.
13. Modelo: Representa los datos que se van a manejar en el sistema informático.
14. Controlador: Gestiona la interacción entre el modelo y la vista. Ayuda a que cada uno reciba los datos de la forma que necesita.
15. Bucle: Permite repetir una secuencia de código hasta que se cumpla una condición establecida previamente.
16. Residencia universitaria: Construcción diseñada para que viva un conjunto de personas estudiantes de universidad.
17. Residente: Persona no perteneciente al Opus Dei que vive en la residencia universitaria y goza de todos sus servicios.
18. Numeraria: Persona perteneciente al Opus Dei que vive en la residencia universitaria, goza de sus servicios y ayuda en la administración de la casa.

2. CAPÍTULO 2: Marco teórico

Este capítulo nos da una introducción a la situación actual de la Residencia Universitaria Tulpa, a la cual se va a aplicar el proyecto, detallando la misión, visión, estructura de organización y una breve descripción de los procesos internos que se siguen.

Además se adjunta información acerca de las herramientas que se van a usar para el desarrollo del proyecto. Tales como la metodología, el framework, el lenguaje de programación, el motor de base de datos, y los diagramas UML que incluye esta disertación.

2.1 Situación actual Residencia Universitaria Tulpa

2.1.1 Misión

Brindar alojamiento de calidad, con habitaciones amuebladas a estudiantes de las diferentes provincias del Ecuador que vengan a estudiar en la ciudad de Quito, en un ambiente familiar y crecimiento personal, contando con atención personalizada por parte de las dirigentes de la casa, además de proveer todos los servicios necesarios para que las mujeres que vengan puedan centrarse solo en sus estudios.

2.1.2 Visión

Convertirnos en una residencia prestigiosa y reconocida para las mujeres de otras provincias del Ecuador que quieran estudiar en Quito y tener una estadía placentera y cómoda.

2.1.3 Estructura organizacional

La Residencia Universitaria Tulpa pertenece al grupo religioso del Opus Dei, mismo que tiene varios centros alrededor del mundo. Todos estos centros se comportan de la

misma forma en cuanto a estructura organizacional. Siempre existe una persona perteneciente al Opus Dei que lidera la residencia, un grupo de personas que se encargan de la limpieza y la comida de quienes viven en la casa, y otro grupo que se encarga de los temas contables. En el siguiente gráfico se muestran los diferentes roles que se manejan:

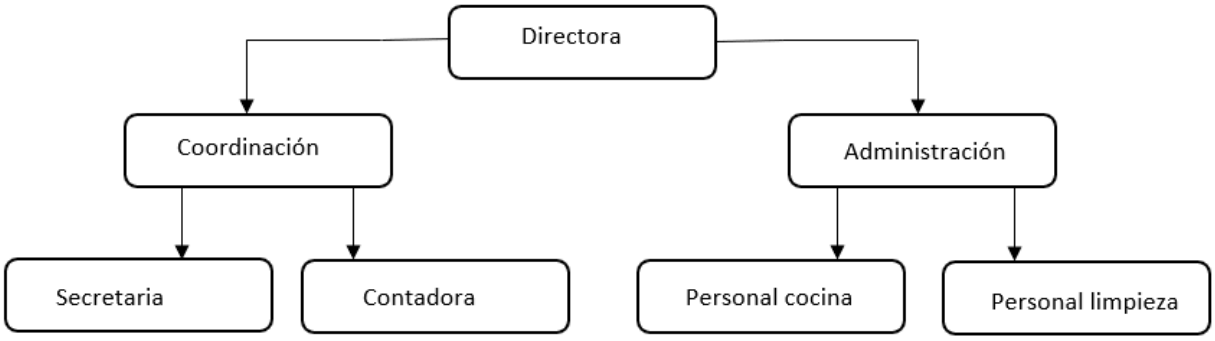


Figura 1: Estructura organizacional de la Residencia Universitaria Tulpa, (Mishelle Zambonino, 2020)

En la Figura 1, se ilustra la Estructura organizacional de la Residencia Universitaria Tulpa, a través de la cual se puede observar que existe una cabeza dentro de esta institución, la directora. Esta persona se encarga de coordinar todas las tareas de la casa, de verificar que todos los demás roles hagan su trabajo, así como de recibir y coordinar la solución a cualquier solicitud de las personas que pertenecen a la residencia en cuanto a daños en la estructura de la casa. Después de la directora, existes 2 grandes áreas que aportan al manejo de esta institución.

El primer grupo es la coordinación, en el que se encuentra la secretaria, una persona perteneciente al Opus Dei encargada de atender las solicitudes de entrevistas con las posibles residentes, además de ayudar a las residentes a solventar dudas acerca de reglas, actividades o funcionamiento de la casa. Esta persona que tiene contacto directo con la

directora, en caso de que alguien quiera hacer una petición lo puede hacer a la secretaria o a la directora directamente. En cuanto a la contadora, es alguien perteneciente al Opus Dei que se encarga de realizar los pagos de servicios básicos de la casa, así como los pagos al personal de administración y maneja el dinero de las pensiones de las residentes.

En el segundo grupo de la administración se encuentra el personal de cocina, un grupo de personas que se encarga de preparar y servir las 4 comidas del día: Desayuno, almuerzo, té y cena para las personas que viven en la residencia, se encargan además del aseo de la cocina. También forma parte de este grupo el personal de limpieza, son personas que se encargan de limpiar las habitaciones de las residentes, así como los espacios sociales de la casa, además se encargan de lavar la ropa de las residentes, así como las sábanas, edredones, toallas y manteles de la casa.

2.1.4 Situación actual de la residencia

En la Residencia Universitaria Tulpa se manejan algunos procesos durante la estadía de las residentes, mismas que ayudan a su manejo y control. El curso inicia cuando los padres de la persona que va a ingresar a la residencia llaman a la directora para agendar una cita y verificar si ésta es apta para quedarse, los aspectos que se toma en cuenta son las calificaciones del colegio y referencias que comprueben que están en capacidad de pagar la estadía. En caso de que la persona sea aceptada, se le asigna una habitación, se le da indicaciones del funcionamiento de la casa en cuanto a horarios y asistencia a las actividades diarias, y para finalizar su ingreso, se le da un recorrido por la casa para que pueda familiarizarse con el ambiente en el que se va a hospedar.

En cuanto al manejo de las actividades diarias, las residentes tienen una hora de entrada a la casa por las noches, que son las 21H30, la llegada de todas las residentes se comprueba en la tertulia a las 20H30, una actividad obligatoria realizada después de la cena donde las personas de la casa comparten un momento de socialización; sin embargo, a esa hora no se puede saber si todas las residentes han llegado y no hay control total. Otra de las actividades realizadas a diario, son los turnos de las comidas; existe un horario para el desayuno, de 06H00 a 08H00, para el almuerzo están disponibles tres turnos, uno a las 12H30, otro a las 13H00 y el último de 14H00 a 17H00, la cena es a las 20H00 el primer turno y a las 21H00 el segundo. Para controlar la cantidad de personas que van a comer en los diferentes horarios, se tiene una hoja impresa en la que las residentes se registran para las comidas diarias. Existen además actividades semanales y mensuales, como retiros y convivencias, para lo cual también se usa una hoja impresa donde se llena el nombre de las personas que va a asistir.

El proceso que se sigue cuando una persona decide dejar la residencia es simplemente una revisión de la habitación y los muebles, que estén en buen estado, si algo no está bien se hace un informe y se aplica un cargo económico extra antes de que la persona deje la residencia.

2.2 Metodología cascada

Antes de profundizar en este tema, se debe tener claro qué es y para qué sirve una metodología de desarrollo de software. Su principal función es, ser una guía que establece cómo realizar de manera correcta el proceso de creación de un sistema informático.

La metodología en cascada se aplica cuando la persona o grupo de personas que va a realizar un determinado proyecto tienen claro los pasos que se deben seguir “cuando el

trabajo desde la comunicación hasta el despliegue fluye en forma razonablemente lineal" (Pressman, 2010). Dado que el producto completo se lo presenta al final al cliente, si hubiese un error o un cambio, el resultado puede ser fatal porque se tendría que empezar el proceso nuevamente. Usualmente, este método es escogido cuando se van a hacer adaptaciones, mejoras, o como ya lo mencionamos anteriormente, con proyectos que tengan los requerimientos claros y estables, lo que es difícil de conseguir y no se da en la mayoría de los casos.

Para poder implementar esta metodología, se siguen ciertas fases separadas, ejecutadas de forma consecutiva y ordenada, primero se debe hacer un análisis y definición de requerimientos, a continuación se realiza el diseño del sistema, donde están los modelos del software, se codifica la aplicación y se realizan las respectivas pruebas. Por último, el sistema se implanta y se realiza el respectivo mantenimiento.

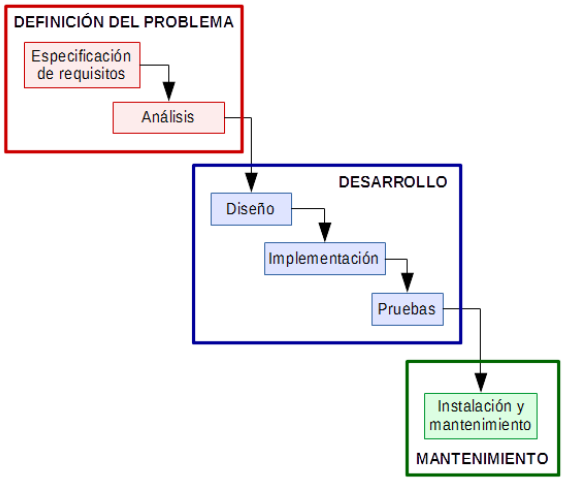


Figura 2: Ciclo de vida en cascada *(Gómez, Bejarano, González, Vergara, & Rodríguez-Sabio, 2020)*

En la Figura 2, se pueden observar las fases del ciclo de vida en cascada, a continuación se detalla cada una de ellas.

2.2.1 Análisis y definición de requerimientos

En esta fase se trabaja con los clientes, el encargado debe determinar las necesidades que se van a cubrir. A partir de estas actividades se genera el documento SRS[1], donde constan todos los requerimientos del usuario. Se incluyen los casos de uso generales y específicos, además del diseño conceptual.

El entregable de esta fase es el SRS.

2.2.2 Diseño del sistema

En la siguiente fase del desarrollo con el método de cascada, está el diseño del sistema, en el cual se descomponen los elementos del sistema, para poder trabajarlos por separado. Se deben definir las entidades y las relaciones que van a formar parte del software. Como resultado de esta etapa, vamos a obtener el SDS[2], en el cual se incluyen los diagramas de clase y los diagramas de secuencia.

El entregable de esta fase es el SDS.

2.2.3 Implementación

En la implementación, se codifican los elementos que especificamos en la fase anterior (diseño del sistema), este desarrollo se lo hace de forma individual, tomando en

[1] Software requirements specification: Documento de especificación de requerimientos.
[2] Software design specification: Documento de especificación de diseño

cuenta que se deben cumplir con todos los requerimientos que el cliente planteó. Lo que genera esta fase es el código de los componentes trabajados individualmente.

El entregable de esta fase es el código fuente de la aplicación.

2.2.4 Integración y pruebas del sistema

El siguiente paso es la integración de las partes individuales, de las cuales ya se probó la funcionalidad. Como en la fase anterior, debemos tomar en cuenta que el sistema cumpla con los requisitos planteados por el cliente. Lo que obtenemos de este proceso son los casos de prueba, que son realizados en base a los casos de uso, y el código del sistema completo. En este punto, el proyecto ya es entregado al cliente.

El entregable de este capítulo es el resultado de las reuniones de trabajo y el acta de aceptación del sistema.

2.2.5 Implantación y mantenimiento

Como último paso, tenemos la implantación, es decir la instalación y puesta en marcha del sistema. En cuanto al mantenimiento, se corrigen errores que no se vieron en las fases anteriores, esto hace que el sistema se vuelva más robusto.

El entregable de esta fase es tanto el manual de usuario como el técnico. Para el mantenimiento se adjunta el formulario de registro de incidentes.

2.3 Herramientas

2.3.1 PHP

Lo primero que debemos tomar en cuenta al hablar de PHP es que es un lenguaje multiplataforma, que se usa del lado del servidor, es decir, el desarrollo que se realice se

ejecuta en el servidor web y se usa la conexión de Internet que tenga la persona que esté usando el sitio web. Este tipo de lenguajes permite tener en la página web varias funcionalidades adicionales, como conexión a una base de datos por ejemplo. En la Figura 3, podemos observar la arquitectura que maneja este lenguaje.

Figura 3: Arquitectura del Lenguaje PHP *(Pelissier, 2002)*

"El lenguaje de programación PHP Hypertext Pre-processor, fue desarrollado puntualmente para diseñar páginas web dinámicas programando script del lado del servidor." (Latinoaméricana, 2010). Los scripts que se generan van vinculados siempre a las páginas HTML, es una ventaja para las páginas web porque separa el diseño de la data que contiene y con eso se consigue más organización y flexibilidad al momento de querer hacer algún cambio o encontrar algo de código.

Este lenguaje de programación nos da en cierta parte velocidad, ya que no consume recursos directamente de la máquina o del sistema, a pesar de que dependerá de la conexión a Internet que se tenga para su rendimiento. En cuanto a seguridad, PHP nos da la opción de configurar los distintos niveles de seguridad como lo desee el programador en el archivo .ini. Otra de las ventajas que nos da este lenguaje es su gran variedad de librerías que

permiten adaptar la página web a las necesidades del programador, incluso se puede añadir extensiones para facilitar el uso del lenguaje.

Sin embargo, PHP no es un lenguaje que pueda usarse solo, debemos tener conocimientos extra de HTML, además de otros lenguajes como CSS o JavaScript para poder realizar una página web completa y funcional. Esto no es un impedimento para su implementación, ya que los lenguajes mencionados anteriormente son sencillos de aprender, y hay suficiente documentación como para no complicar su aprendizaje.

2.3.2 JavaScript

JavaScript es un lenguaje de programación creado por Netscape tomando como base el lenguaje Java, la diferencia es que Java es un lenguaje con el que se pueden crear aplicaciones completas, mientras que JavaScript puede generar programas que funcionan solo con HTML. Este lenguaje genera programas llamados scripts[3], los mismos que dan a la página web un grado mayor de dinamismo. Generalmente los scripts cumplen su función cuando se ejecutan después de una acción en un componente HTML, por ejemplo cuando se da clic en algún botón.

Otra de las características que nos ofrece este lenguaje es el uso de variables. Se pueden usar variables de diferentes tipos, como numéricos, strings, entre otros. A diferencia de otros lenguajes, en JavaScript no es necesario asignar un tipo, sino que el intérprete lo clasifica.

[3] Script: Documento de instrucciones, escritas en lenguaje de programación que permite al computador ejecutar funciones o reglas descritas.

JavaScript trabaja con objetos que poseen características o propiedades por ejemplo una persona puede ser un objeto que tiene propiedades como nombre, apellido, teléfono entre otras cosas; además de poseer acciones, por ejemplo una persona puede ingresar a un sistema, salir de él o modificar sus datos. Cuando hablamos de una página web, estos objetos podemos tomarlos como los componentes de la página web y sus acciones son los scripts descritos anteriormente.

"La sintaxis de un lenguaje de programación se define como un conjunto de reglas que deben seguirse al escribir el código fuente" (Navarrete, 2006). La sintaxis de este lenguaje de programación se parece a Java y C. Algunas de las características son:

- Los espacios en blanco que se encuentren al inicio o al final de las líneas no se toman en cuenta, por lo que nos da una ventaja de orden en cuanto a tabulación dentro del código.
- Se debe tomar en cuenta que JavaScript distingue las letras minúsculas de las mayúsculas, esto nos advierte a recordar cómo escribirnos los nombres de las variables.
- Como ya lo dijimos anteriormente, en este lenguaje de programación no es necesario escribir el tipo de dato de una variable, sino que el compilador lo determina dependiendo de la naturaleza del dato.
- A diferencia de otros lenguajes, en éste no es necesario incluir el punto y coma al final de cada línea de programación, pero en caso de que se quiera mantener un estándar se puede colocar.
- Como en cualquier lenguaje de programación, se puede incluir comentarios de una sola línea (//comentario) o de varias líneas (/* comentario */)

2.3.3 HTML

HTML (Hyper Text Markup Language o Lenguaje de Marcas de Hipertexto) empieza su historia en 1998, actualmente nos encontramos en la versión 5. HTML usa etiquetas para su funcionamiento, dichas etiquetas van entre los símbolos de mayor y menor que (<>), además deben tener un inicio y un fin, por ejemplo, <label>something</label>. Este lenguaje de marcas fue creado como parte del servicio web (Wold Wide Web o www), mismo que se compone de 3 cosas: protocolo HTTP que permite la transmisión de datos a través de Internet, un conjunto de direcciones URL para poder ingresar a las páginas web y un lenguaje universal para incluir información (HTML).

El funcionamiento de HTML es: Escribir el código en cualquier editor de texto, guardarlo como extensión .html y al abrir el archivo, este se ejecutará en el navegador y será interpretado. Dentro de las páginas web podemos incluir diferentes tipos de contenido, entre ellos tenemos: Texto, imágenes, audio, video, animaciones y realidad virtual.

Este lenguaje ha ido evolucionando. Empezó con la versión 0, que establece la estructura base de una página en Internet. La versión 1 incluye estilos para los textos, al incluir las dos versiones anteriores sale a la luz la versión 1.0. En la siguiente versión, es decir la 2.0 se incluyen formularios, y las tablas se agregan en la versión 3.0. La próxima evolución es la 3.2 donde se agregan desarrollos en Netscape, posterior a esto ya se incluye lo que conocemos ahora, Javascript, objetos multimedia, hojas de estilo, esto pertenece a la versión 4. Actualmente HTML se encuentra en la versión 5 y se incluyen algunas mejoras incluyendo la gestión directa de video.

2.3.4 Modelo Vista Controlador

"MVC es un patrón de diseño que considera dividir una aplicación en tres módulos claramente identificables y con funcionalidad bien definida: El Modelo, las Vistas y el Controlador." (Patoja, 2004). Este patrón de diseño fue creado para evitar el desorden al momento de escribir código, existen ocasiones en las que los programadores tienden a caer en malos hábitos de mezclar demasiadas funcionalidades en un solo bloque de código, causando complicaciones al momento de querer modificarlo cuando ha pasado un período de tiempo, ya que tiende a ser inentendible, o muy difícil de comprender, como consecuencia, la modificación requerida no se la podrá hacer de forma fácil ni rápida. A continuación se explica cada una de las capas propuestas para el patrón Modelo Vista Controlador.

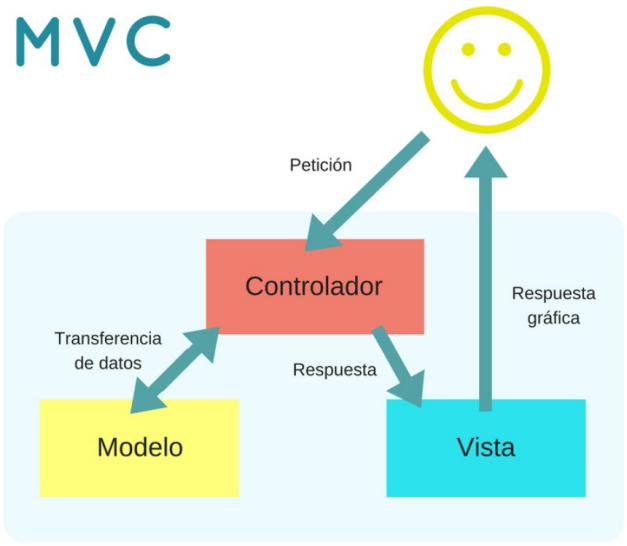


Figura 4: Modelo Vista Controlador *(García, 2020)*

Como podemos observar en la Figura 4, el usuario realiza una petición, la misma que llega al controlador que maneja la lógica de programación, en el desarrollo de la disertación

se incluirá en esta capa las funciones para poder manejar los datos y las solicitudes del usuario. Una vez realizada la petición, se usa la capa de modelo para la interacción con la base de datos, para poder insertar, modificar y consultar datos según la necesidad del usuario. Estos datos regresan al controlador y son presentados en la capa de vista. Esta última capa es la única que el cliente mira, de ella se generan y llegan todas las peticiones.

2.3.4.1Ventajas

- Al implementar la aplicación modularmente, los cambios o aumento de cosas serán más fáciles, ya que el código está ordenado y su manipulación es más rápida.
- Cuando se modifican cosas en vista, no influye en ninguna de las otras capas, por lo que hay riesgo mínimo que ese tipo de cambios afecten al funcionamiento en general de la aplicación.
- Este patrón es usado para realizar aplicaciones de gran tamaño en el mercado, por lo que es confiable aplicar el Modelo Vista Controlador.

2.3.4.2Desventajas

- Su implementación toma más tiempo, al ser modular, se necesita más cantidad de clases. Sin embargo, con el resultado obtenido es una inversión de tiempo necesaria y que generará ventajas al final del desarrollo.
- Tomando en cuenta que el patrón Modelo Vista Controlador fue diseñado para aplicaciones orientadas a objetos, muchas veces los lenguajes que no usan este paradigma tienen dificultad para lograr cumplir el patrón.

3. CAPÍTULO 3: Análisis y diseño del sistema

En este capítulo se van a presentar los diagramas de casos de uso general y a detalle de los procesos que realiza el sistema. También se presentarán los diagramas de secuencia. Recordemos que estos diagramas son la base del desarrollo, ya que describen cómo va a funcionar la aplicación. Adicional a esto se mostrarán los diagramas de clases, referentes a la arquitectura en la que está construido el programa y el diagrama conceptual, una representación de la estructura de la base de datos.

3.1 Especificación de requerimientos (SRS)

3.1.1 Introducción

El presente documento, describe el análisis y especificación de requerimientos de un sistema que automatiza las actividades y los datos de una residencia universitaria. Esto ayudará a un mejor manejo y gran facilidad para las residentes y administradoras al momento de saber y apuntarse a las diferentes actividades. La aplicación ayudará a la Residencia Universitaria Tulpa, que va a tener un extra que le ayudará en el mercado diferenciarse de las demás residencias universitarias

3.1.2 Requerimientos funcionales

- F1: Administrar ocupaciones
- F2: Administrar tipos de actividades
- F3: Administrar universidades
- F4: Administrar pisos
- F5: Administrar habitaciones
- F6: Administrar países

- F7: Administrar estados
- F8: Administrar ciudades
- F9: Administrar carreras
- F10: Administrar colegios
- F11: Administrar posibles residentes
- F12: Administrar residentes
- F13: Administrar numerarias
- F14: Administrar actividades

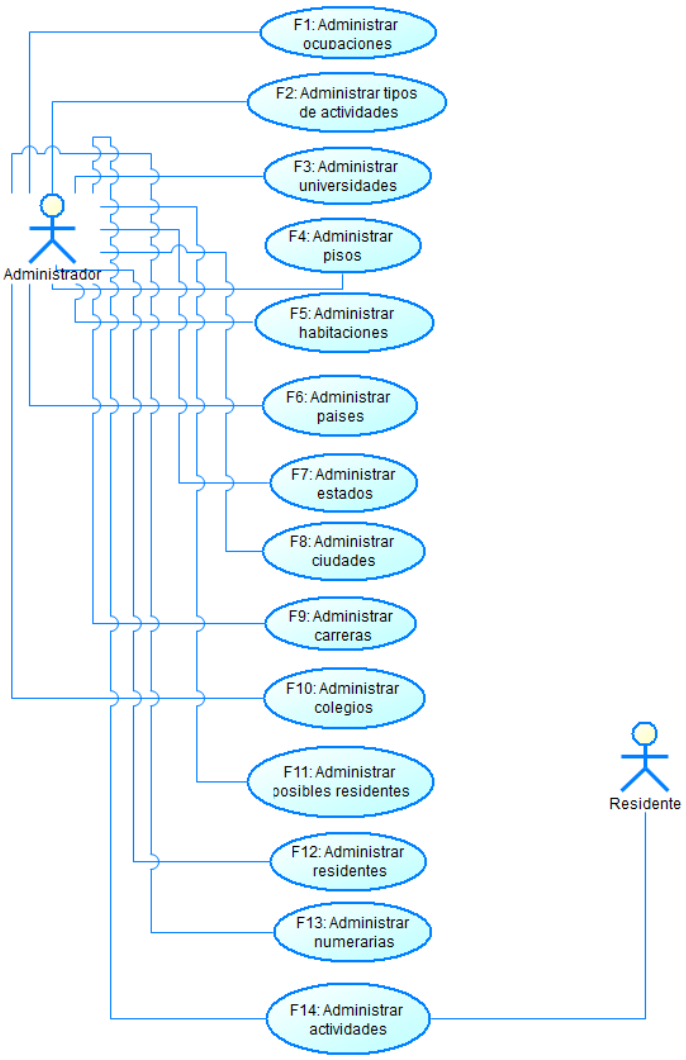
3.1.3 Requerimientos no funcionales

- Lenguaje de programación: PHP, Javascript, HTML.
- Base de Datos: MySQL
- Arquitectura: 3 capas orientado a la web
- Hosting: Hosting en la nube
- Metodología: Cascada
- Seguridad: Manejo de un super usuario a la base de datos para la conexión desde la aplicación

3.1.4 Diagramas

Caso de uso general

El diagrama de caso de uso general describe las acciones o actividades que va a permitir manejar el sistema, así como la relación que tiene con los diferentes actores que van a usar la aplicación. A continuación se va a presentar las funcionalidades del proyecto además de los roles que existen y sus accesos.

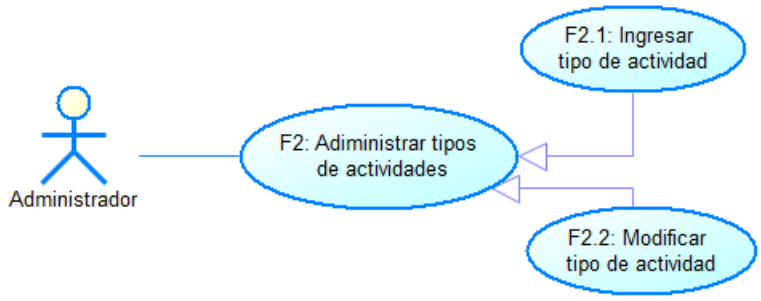


3.1.5 Casos de uso a detalle

Los casos de uso a detalle son un mecanismo que complementa el diagrama de caso de uso general, describe cada funcionalidad por separado, incluidas las acciones de dichas funcionalidades y los actores que intervienen en cada actividad en específico, de esta manera podemos ver de una mejor manera la interacción que se va a producir en la aplicación.

Dentro de la disertación los casos de uso que hacen referencia a catálogos son similares en su gestión y éstos son F1: Administrar ocupaciones, F2: Administrar tipos de actividades, F3: Administrar universidades, F4: Administrar pisos, F5: Administrar habitaciones, F6: Administrar países, F7: Administrar estados, F8: Administrar ciudades, F9: Administrar carreras, F10: Administrar colegios. Por lo tanto, se presenta a continuación los casos de uso a detalle del más representativo, en este caso es F2: Administrar tipos de actividades. El detalle del resto de los casos de uso para manejo de catálogos está en el Anexo 1.

F2: Administrar tipos de actividades



F2.1: Ingresar tipo de actividad

Descripción: Este caso de uso describe el flujo que debe seguir un actor para ingresar un nuevo tipo de actividad al sistema.

Actores: Administradores.

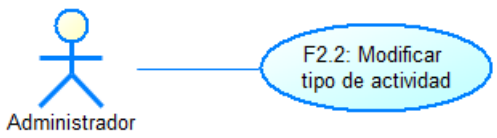
Flujo principal:

1. El actor selecciona la opción de Catálogos > Tipos Actividades del menú principal.
2. El sistema presenta la pantalla de tipos de actividades con la información de la base cargada.
3. El actor presiona el botón Nuevo Tipo de Actividad.
4. El sistema despiega un modal de ingreso del tipo de actividad.
5. El actor ingresa la información del tipo de actividad.
6. El actor pulsa el botón Guardar.
7. El sistema guarda los datos del tipo de actividad en la base de datos.

Excepciones:

Código	*Descripción*	*Solución*
E1	Error de conexión con la base de datos	Contactarse con el administrador de base de datos

F2.2: Modificar tipo de actividad



Descripción: Este caso de uso describe el flujo que debe seguir un actor para modificar un tipo de actividad existente.

Actores: Administradores.

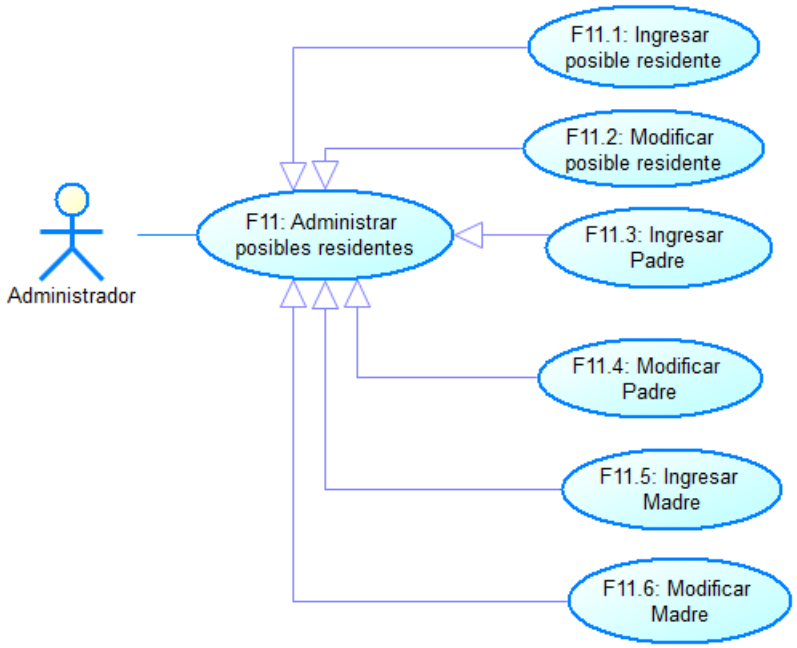
Flujo principal:

1. El actor escoge la opción Catálogos > Tipos Actividades del menú.
2. La aplicación carga la pantalla de tipos de actividades con la información de la base cargada.
3. El actor presiona el botón Modificar del tipo de actividad que desee.
4. El sistema muestra un modal con los datos del tipo de actividad.
5. El actor cambia los datos que el sistema lo permita.
6. El actor da clic en el botón Guardar.
7. El sistema modifica los datos del tipo de actividad en la base de datos.

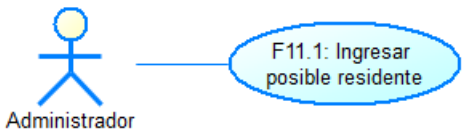
Excepciones:

Código	*Descripción*	*Solución*
E1	Error de conexión con la base de datos	Contactarse con el administrador de base de datos

F11: Administrar posibles residentes



F11.1: Ingresar posible residente



Descripción: Este caso de uso describe el flujo que debe seguir un actor para ingresar una nueva posible residente al sistema. Las posibles residentes son aquellas que han hecho la entrevista pero no han confirmado que se mudan a la residencia.

Actores: Administradores.

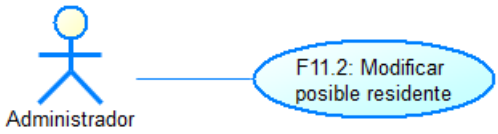
Flujo principal:

1. El actor elige la opción de Residencia > Posibles Residentes del menú.
2. El sistema carga la pantalla de residentes, con sus ciudades y colegios con la información de la base cargada.
3. El actor presiona el botón Nueva Residente.
4. El sistema presenta un modal de ingreso de residente con la geografía (Países, estados y ciudades) y los colegios activos de la Base de Datos.
5. El actor ingresa la información de la residente.
6. El actor presiona el botón con el texto Guardar.
7. El sistema verifica que la cédula de identidad ingresada no exista en el sistema.
8. El sistema guarda los datos de la posible residente en la base de datos.

Excepciones:

Código	*Descripción*	*Solución*
E1	Error de conexión con la base de datos	Contactarse con el administrador de base de datos

F11.2: Modificar posible residente



Descripción: Este caso de uso describe el flujo que debe seguir un actor para modificar una posible residente al sistema. Las posibles residentes son aquellas que han hecho la entrevista pero no han confirmado que se mudan a la residencia.

Actores: Administradores.

Flujo principal:

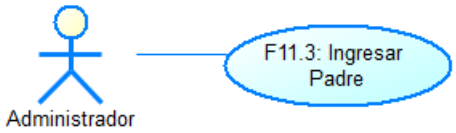
1. El actor escoge la opción de Residencia > Posibles Residentes del menú principal.
2. El sistema carga la pantalla de residentes, con sus ciudades y colegios con la información de la base cargada.
3. El actor presiona el botón Modificar de la posible residente que desee.
4. El sistema presenta un modal con los datos de la posible residente.
5. El actor modifica los datos que el sistema lo permita.
6. El actor oprime el botón con el textoGuardar.
7. El sistema modifica los datos de la posible residente en la base de datos. En caso de que se cambie el check box de Residente a activo, el registro se elimina de la tabla posibles residentes y pasa a la de Residentes.

Excepciones:

Código	*Descripción*	*Solución*

E1	Error de conexión con la base de datos	Contactarse con el administrador de base de datos

F11.3: Ingresar padre



Descripción: Este caso de uso describe el flujo que debe seguir un actor para ingresar el padre de una posible residente al sistema. Las posibles residentes son aquellas que han hecho la entrevista pero no han confirmado que se mudan a la residencia.

Actores: Administradores.

Flujo principal:

1. El actor escoge la opción de Residencia > Posibles Residentes del menú principal.
2. El sistema carga la pantalla de residentes, con sus ciudades y colegios con la información de la base cargada.
3. El actor presiona el botón “+” de la columna Padres de la posible residente que desee.
4. La aplicación carga un modal con las opciones de Padre y Madre.
5. El actor presiona el tab Padre.
6. El sistema carga el formulario de Ingreso de Padre.
7. El actor ingresa los datos que el sistema lo permita.
8. El actor oprime el botón Guardar.
9. El sistema ingresa los datos del Padre y lo enlaza con la posible residente.

Flujo Alterno:

6. Si e sistema carga los datos del padre ver *CU11.4: Modificar padre*.

Excepciones:

Código	*Descripción*	*Solución*
E1	Error de conexión con la base de datos	Contactarse con el administrador de base de datos

F11.4: Modificar padre

Descripción: Este caso de uso describe el flujo que debe seguir un actor para modificar el padre de una posible residente al sistema. Las posibles residentes son aquellas que han hecho la entrevista pero no han confirmado que se mudan a la residencia.

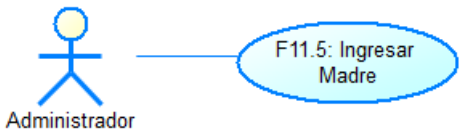
Actores: Administradores.

Flujo principal:

1. El actor selecciona Residencia > Posibles Residentes del menú principal.
2. El sistema carga la pantalla de residentes, con sus ciudades y colegios con la información de la base cargada.
3. El actor presiona el botón “+” de la columna Padres de la posible residente que desee.
4. La aplicación presenta un modal con las opciones de Padre y Madre.
5. El actor presiona el tab Padre.
6. El sistema carga el formulario de Ingreso de Padre y con los datos cargados.
7. El actor modifica los datos que el sistema admita.

8. El usuario presiona el botón Guardar.

F11.5: Ingresar madre



Descripción: Este caso de uso describe el flujo que debe seguir un actor para ingresar la madre de una posible residente al sistema. Las posibles residentes son aquellas que han hecho la entrevista pero no han confirmado que se mudan a la residencia.

Actores: Administradores.

Flujo principal:

1. El actor elige Residencia > Posibles Residentes del menú.
2. El sistema carga la pantalla de residentes, con sus ciudades y colegios con la información de la base cargada.
3. El actor presiona el botón “+” de la columna Padres de la posible residente que desee.
4. A aplicación muestra un modal con las opciones de Padre y Madre.
5. El actor presiona el tab Madre.
6. El sistema carga el formulario de Ingreso de Madre.
7. El actor ingresa los datos que el sistema lo permita.
8. El actor presiona el botón Guardar.
9. El sistema ingresa los datos de la Madre y lo enlaza con la posible residente.

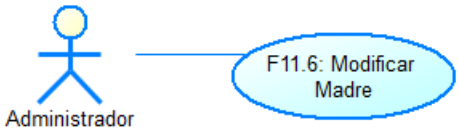
Flujo Alterno:

6. Si e sistema carga los datos del padre ver *CU11.6: Modificar madre*.

Excepciones:

Código	*Descripción*	*Solución*
E1	Error de conexión con la base de datos	Contactarse con el administrador de base de datos

F11.6: Modificar madre



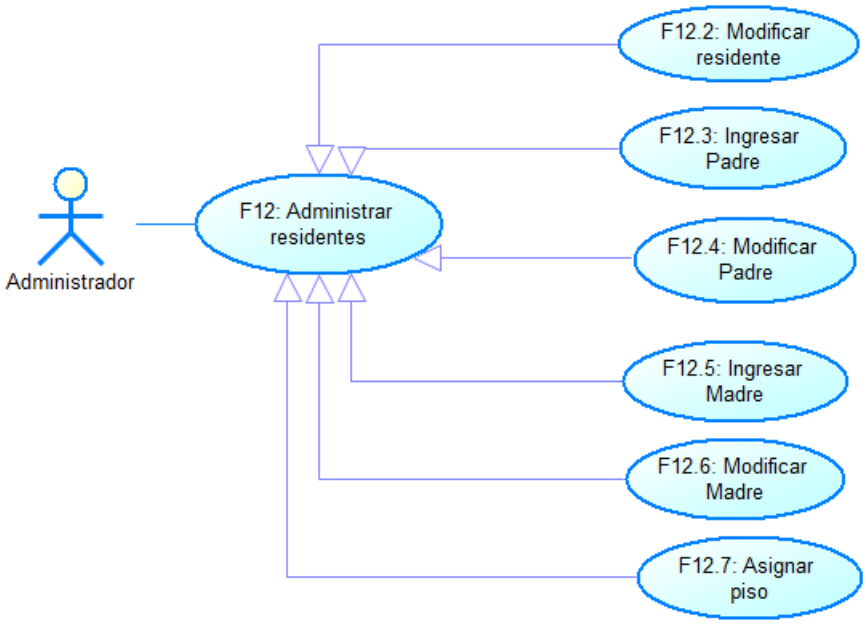
Descripción: Este caso de uso describe el flujo que debe seguir un actor para modificar la madre de una posible residente al sistema. Las posibles residentes son aquellas que han hecho la entrevista pero no han confirmado que se mudan a la residencia.

Actores: Administradores.

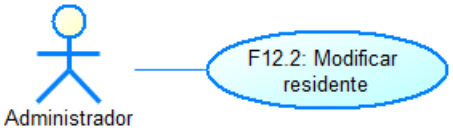
Flujo principal:

1. El actor escoge la opción de Residencia > Posibles Residentes del menú del lado izquierdo.
2. El sistema carga la pantalla de residentes, con sus ciudades y colegios con la información de la base cargada.
3. El actor presiona el botón “+” de la columna Padres de la posible residente que desee.
4. El sistema carga un modal con las opciones de Padre y Madre.
5. El actor presiona el tab Madre.
6. El sistema carga el formulario de Ingreso de Madre y con los datos cargados.
7. El actor modifica los datos que disponibles dentro del modal.
8. El actor pulsa el botón que contiene la palabra Guardar.

F12: Administrar residentes



F12.2: Modificar residente



Descripción: Este caso de uso describe el flujo que debe seguir un actor para modificar una residente al sistema. Las residentes son aquellas que viven en la residencia.

Actores: Administradores.

Flujo principal:

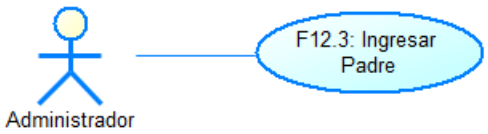
1. El actor selecciona la opción de Residencia > Residentes del menú izquierdo de la pantalla.
2. El sistema carga la pantalla de residentes, con sus ciudades y colegios con la información de la base cargada.

3. El actor presiona el botón Modificar de la residente que desee.
4. El sistema presenta un modal con los datos de la residente cargados.
5. El actor cambia los datos que el sistema lo permita.
6. El actor escoge el botón con el texto Guardar.
7. El sistema modifica los datos de la residente en la base de datos.

Excepciones:

Código	*Descripción*	*Solución*
E1	Error de conexión con la base de datos	Contactarse con el administrador de base de datos

F12.3: Ingresar padre



Descripción: Este caso de uso describe el flujo que debe seguir un actor para ingresar el padre de una residente al sistema. Las residentes son aquellas que viven en la residencia.

Actores: Administradores.

Flujo principal:

1. El actor escoge la opción de Residencia > Residentes del menú principal.
2. El sistema carga la pantalla de residentes, con sus ciudades y colegios con la información de la base cargada.
3. El actor presiona el botón “+” de la columna Padres de la posible residente que desee.
4. El sistema carga una pantalla con las opciones de Padre y Madre.

5. El actor presiona el tab Padre.
6. El sistema carga el formulario de Ingreso de Padre.
7. El actor ingresa los datos que el sistema lo permita.
8. El actor pulso el botón Guardar.
9. El sistema ingresa los datos del Padre y lo enlaza con la posible residente.

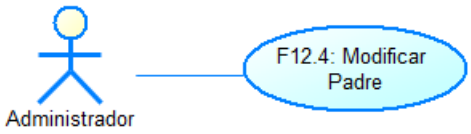
Flujo Alterno:

6. Si e sistema carga los datos del padre ver *CU11.4: Modificar padre*.

Excepciones:

Código	*Descripción*	*Solución*
E1	Error de conexión con la base de datos	Contactarse con el administrador de base de datos

F12.4: Modificar padre



Descripción: Este caso de uso describe el flujo que debe seguir un actor para modificar el padre de una residente al sistema. Las residentes son aquellas que viven en la residencia.

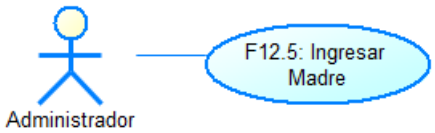
Actores: Administradores.

Flujo principal:

1. El actor elige Residencia > Residentes del menú principal.

2. El sistema carga la pantalla de residentes, con sus ciudades y colegios con la información de la base cargada.
3. El actor presiona el botón “+” de la columna Padres de la posible residente que desee.
4. El sistema presenta una pantalla tipo modal con las opciones de Padre y Madre.
5. El actor presiona el tab Padre.
6. El sistema carga el formulario de Ingreso de Padre y con los datos cargados.
7. El actor modifica los datos que el sistema lo permita.
8. El actor da clic el botón que contenga el texto Guardar.

F12.5: Ingresar madre



Descripción: Este caso de uso describe el flujo que debe seguir un actor para ingresar la madre de una residente al sistema. Las residentes son aquellas que viven en la residencia.

Actores: Administradores.

Flujo principal:

1. El actor selecciona la opción de Residencia > Residentes del menú principal.
2. El sistema carga la pantalla de residentes, con sus ciudades y colegios con la información de la base cargada.
3. El actor presiona el botón “+” de la columna Padres de la posible residente que desee.
4. La aplicación carga un modal con las opciones de Padre y Madre.
5. El actor presiona el tab Madre.

6. El sistema carga el formulario de Ingreso de Madre.
7. El actor ingresa los datos que el sistema lo permita.
8. El usuario presiona el botón que contenga el texto Guardar.
9. El sistema ingresa los datos de la Madre y lo enlaza con la posible residente.

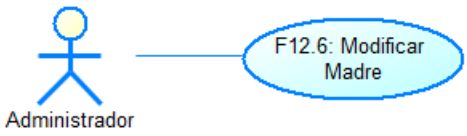
Flujo Alterno:

6. Si e sistema carga los datos del padre ver *CU11.6: Modificar madre*.

Excepciones:

Código	*Descripción*	*Solución*
E1	Error de conexión con la base de datos	Contactarse con el administrador de base de datos

F12.6: Modificar madre



Descripción: Este caso de uso describe el flujo que debe seguir un actor para modificar la madre de una residente al sistema. Las residentes son aquellas que viven en la residencia.

Actores: Administradores.

Flujo principal:

1. El actor da clic en la opción de Residencia > Residentes del menú principal.
2. El sistema carga la pantalla de residentes, con sus ciudades y colegios con la información de la base cargada.

3. El actor presiona el botón “+” de la columna Padres de la posible residente que desee.
4. El sistema carga un modal con las opciones de Padre y Madre.
5. El actor presiona el tab Madre.
6. El sistema carga el formulario de Ingreso de Madre y con los datos cargados.
7. El actor edita la información que el sistema le permita.
8. El actor pulsa el botón Guardar.
9. La aplicación modifica los datos de la madre de la residente.

F12.7: Asignar piso

Descripción: Este caso de uso describe el flujo que debe seguir un actor para asignar una habitación a una residente al sistema. Las posibles residentes son aquellas que han hecho la entrevista pero no han confirmado que se mudan a la residencia.

Actores: Administradores.

Flujo principal:

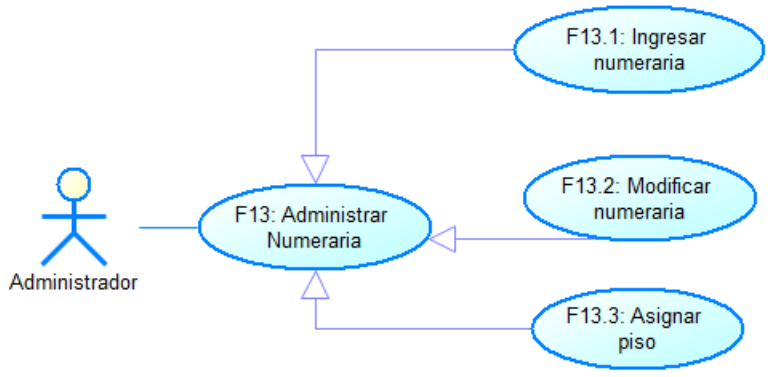
1. El actor selecciona la opción de Residencia > Residentes del menú principal.
2. El sistema carga la pantalla de residentes, con sus ciudades y colegios con la información de la base cargada.
3. El actor presiona el botón “+” de la columna Habitación de la residente que desee.
4. El sistema muestra una pantalla tipo modal con los pisos y las habitaciones activas que no se encuentren ocupadas.

5. El actor Selecciona el piso y la habitación que desea asignarle a la residente.
6. El actor oprime el botón Guardar.
7. El sistema ingresa la habitación asignada a la residente y cambia el estado de la habitación a ocupada.

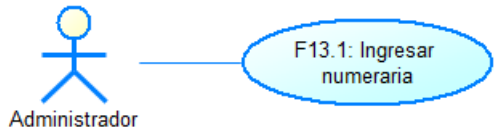
Flujo alterno:

4. Si la residente tiene una habitación asignada, se asignan a los combobox los valores pertenecientes a la residente.
5. El actor cambia los datos si lo desea.
6. El actor oprime el botón que contenga el texto Guardar.
7. El sistema modifica la habitación asignada a la residente y cambia el estado de la habitación a Libre.

F13: Administrar numeraria



F13.1: Ingresar numeraria



Descripción: Este caso de uso describe el flujo que debe seguir un actor para ingresar una numeraria al sistema.

Actores: Administradores.

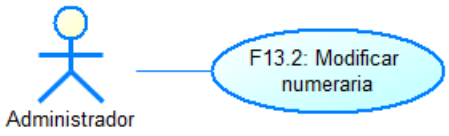
Flujo principal:

1. El actor elige la opción de Residencia > Numerarias del menú principal.
2. El sistema carga la pantalla de numerarias, con sus ciudades y con la información de la base cargada.
3. El actor presiona el botón Nueva Numeraria.
4. El sistema presenta un modal de ingreso de numeraria con la geografía (Países, estados y ciudades), las ocupaciones, universidades y careras activos de la Base de Datos.
5. El actor ingresa la información de la numeraria.
6. El actor presiona el botón Guardar.
7. El sistema verifica que la cédula de identidad ingresada no exista en el sistema.
8. El sistema guarda los datos de la numeraria en la base de datos.

Excepciones:

Código	*Descripción*	*Solución*
E1	Error de conexión con la base de datos	Contactarse con el administrador de base de datos

F13.2: Modificar numeraria



Descripción: Este caso de uso describe el flujo que debe seguir un actor para modificar una numeraria en el sistema.

Actores: Administradores.

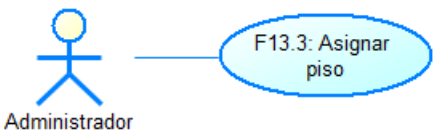
Flujo principal:

1. El actor elige Residencia > Numerarias del menú principal.
2. El sistema carga la pantalla de numerarias, con sus ciudades y respectiva información de la base cargada.
3. El actor presiona el botón Modificar de la numeraria que desee.
4. El sistema presenta una pantalla tipo modal con los datos de la numeraria, las ocupaciones, las universidades y las carreras.
5. El actor cambia los datos que el sistema lo permita.
6. El actor presiona el botón Guardar.
7. El sistema modifica los datos de la numeraria en la base de datos.

Excepciones:

Código	*Descripción*	*Solución*
E1	Error de conexión con la base de datos	Contactarse con el administrador de base de datos

F13.3 Asignar piso



Descripción: Este caso de uso describe el flujo que debe seguir un actor para asignar una habitación a una residente al sistema. Las posibles residentes son aquellas que han hecho la entrevista pero no han confirmado que se mudan a la residencia.

Actores: Administradores.

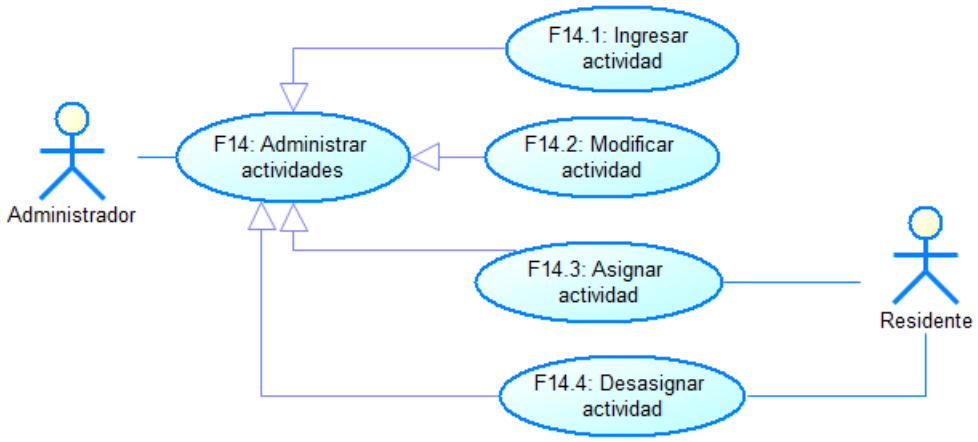
Flujo principal:

1. El actor escoge Residencia > Numerarias del menú izquierdo de la pantalla.
2. El sistema carga la pantalla de numerarias, con sus ciudades y con la información de la base cargada.
3. El actor presiona el botón "+" de la columna Habitación de la numeraria que desee.
4. A aplicación carga una pantalla tipo modal con los pisos y las habitaciones activas que no se encuentren ocupadas.
5. El actor Selecciona el piso y la habitación que desea asignarle a la numeraria.
6. El actor pulsa el botón Guardar.
7. El sistema ingresa la habitación asignada a la numeraria y cambia el estado de la habitación a ocupada.

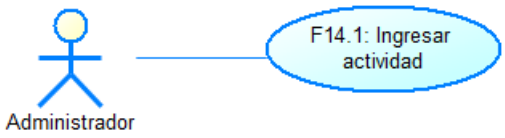
Flujo alterno:

4. Si la residente tiene una habitación asignada, se asignan a los combobox los valores pertenecientes a la numeraria.
8. El actor cambia los datos si lo desea.
9. El actor oprime el botón Guardar.
10. El sistema cambia la habitación asignada a la numeraria y modifica el estado de la habitación a Libre.

F14: Administrar actividades



F14.1: Ingresar actividad



Descripción: Este caso de uso describe el flujo que debe seguir un actor para ingresar una nueva actividad al sistema.

Actores: Administradores.

Flujo principal:

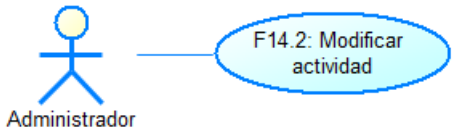
1. El actor selecciona la opción de Residencia > Actividades del menú principal.
2. El sistema muestra la pantalla de actividades con la información de la base cargada.
3. El actor presiona el botón Nueva Actividad.
4. El sistema presenta un modal de ingreso de actividades con los tipos de actividades activos de la Base de Datos.
5. El actor ingresa la información de la actividad.

6. El actor presiona el botón Guardar.
7. El sistema guarda los datos actividad en la base de datos.

Excepciones:

Código	*Descripción*	*Solución*
E1	Error de conexión con la base de datos	Contactarse con el administrador de base de datos

F14.2: Modificar actividad



Descripción: Este caso de uso describe el flujo que debe seguir un actor para modificar una actividad en el sistema.

Actores: Administradores.

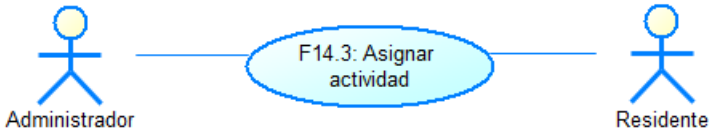
Flujo principal:

1. El actor escoge la opción de Residencia > Actividades del menú principal.
2. El sistema carga la pantalla de actividades, con su respectiva información de la base cargada.
3. El actor presiona el botón Modificar de la actividad que desee.
4. El sistema carga un modal con la información de la actividad.
5. El actor modifica los datos permitidos dentro del modal.
6. El actor presiona el botón Guardar.
7. El sistema modifica los datos de la actividad en la base de datos.

Excepciones:

Código	*Descripción*	*Solución*
E1	Error de conexión con la base de datos	Contactarse con el administrador de base de datos

F14.3 Asignar actividad



Descripción: Este caso de uso describe el flujo que debe seguir un actor para inscribirse a una actividad de la residencia.

Actores: Administradores y residentes. Tomar en cuenta, que por residentes nos referimos a todas las personas que vivan en la casa.

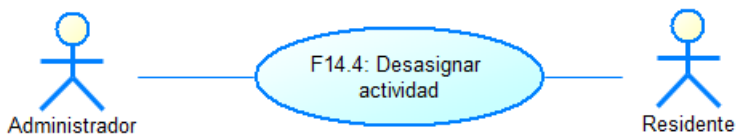
Flujo principal:

1. El actor elige la opción de Actividades del menú izquierdo.
2. El sistema presenta la pantalla con las actividades en que no esté asignada guardadas en la base de datos.
3. El actor presiona el botón “+” de la columna Inscribirse de la actividad que desee.
4. El sistema asigna la actividad al actor.

Excepciones:

Código	*Descripción*	*Solución*
E1	Error de conexión con la base de datos	Contactarse con el administrador de base de datos

F14.4 Desasignar actividad



Descripción: Este caso de uso describe el flujo que debe seguir un actor para inscribirse a una actividad de la residencia.

Actores: Administradores y residentes. Tomar en cuenta, que por residentes nos referimos a todas las personas que vivan en la casa.

Flujo principal:

1. El actor selecciona la opción de Actividades del menú.
2. El sistema muestra la pantalla con las actividades en que no esté asignada guardadas en la base de datos.
3. El actor presiona el botón “Mis Actividades”.
4. El sistema carga las actividades que tenga el actor.
5. El actor presiona el botón “Eliminar”.
6. El sistema elimina la inscripción del actor a la actividad.

Excepciones:

Código	*Descripción*	*Solución*
E1	Error de conexión con la base de datos	Contactarse con el administrador de base de datos

3.2 Especificaciones de diseño (SDS)

3.2.1 Introducción

El presente documento, describe la especificación de diseño del sistema descrito en el SRS. De aquí, obtendremos la base para poder construir el sistema, gracias al diagrama de clases, diagramas de secuencia y diagrama entidad-relación.

3.2.2 Diagrama de clases

Los diagramas de clase se los usa para plantear la estructura del sistema a implementar. En este caso, los objetos del mundo real se representan con clases, las cuales tienen atributos, que representan las características que poseen los objetos mencionados anteriormente. Todos ellos se relacionan entre sí, tal como se lo presenta en el Figura 5.

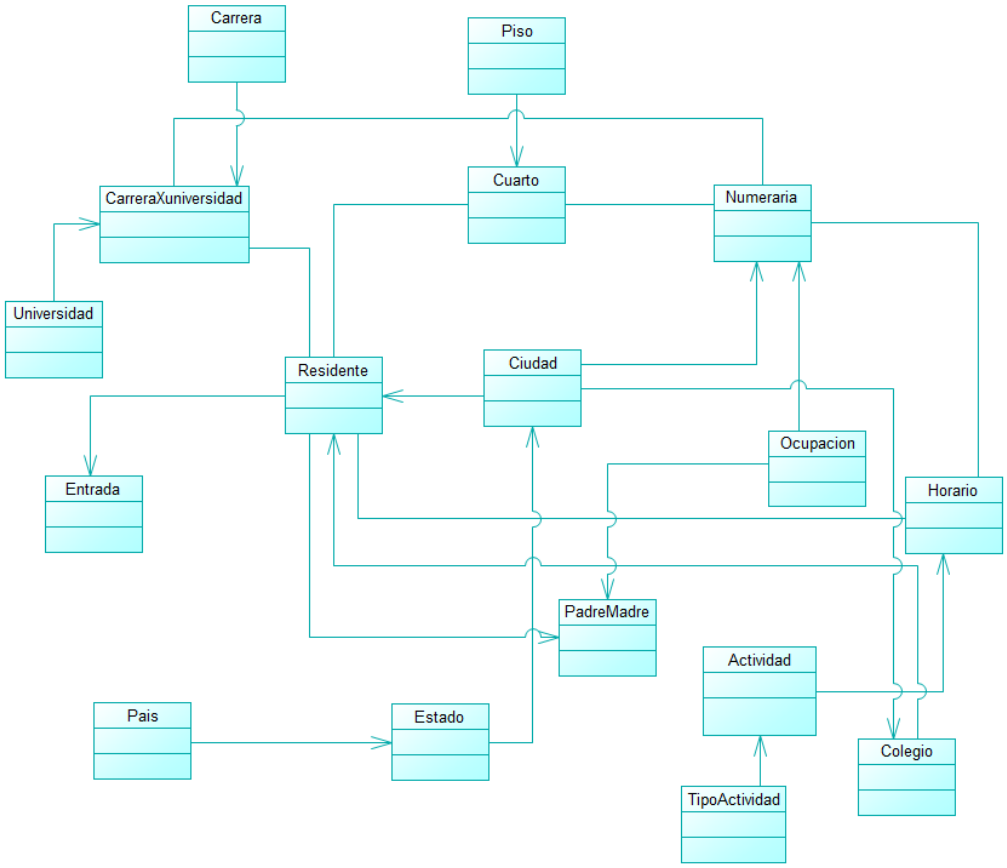


Figura 5: Diagrama de clases conceptual (Mishelle Zambonino, 2020)

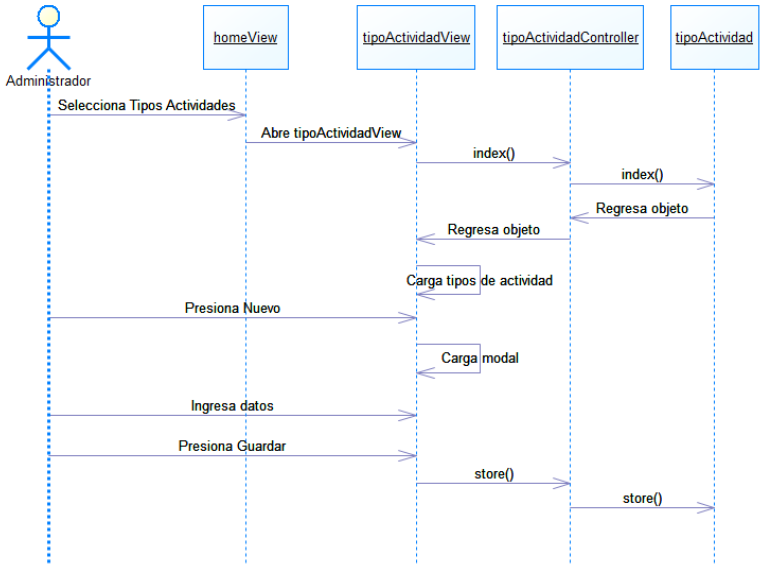
3.2.3 Diagramas de secuencia

"Los Diagramas de Secuencias muestran la forma en que un grupo de objetos se comunican (interactúan) entre sí a lo largo del tiempo" (Gutierrez, 2011). Básicamente lo que obtenemos de este diagrama es el flujo a detalle de los procesos y sus componentes. Para poder realizarlo, se deben incluir los actores y los objetos involucrados, cada objeto y actor debe poseer su línea de vida; entre las cuales es necesario manejar mensajes que representan la comunicación mencionada anteriormente. Recordemos que dichos mensajes se representan como vectores, y su orden debe ir en orden cronológico.

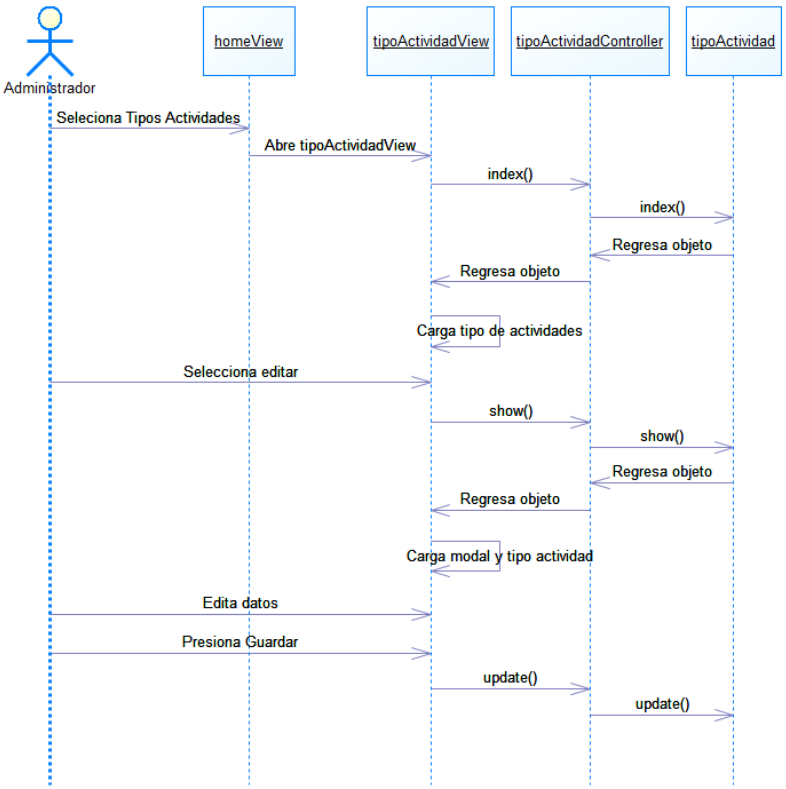
Al igual que en los casos de uso, los diagramas de secuencia que hacen referencia a catálogos son similares en su gestión y éstos son F1: Administrar ocupaciones, F2: Administrar tipos de actividades, F3: Administrar universidades, F4: Administrar pisos, F5: Administrar habitaciones, F6: Administrar países, F7: Administrar estados, F8: Administrar ciudades, F9: Administrar carreras, F10: Administrar colegios. Por lo tanto, se presenta a continuación los diagramas de secuencia del más representativo, en este caso es F2: Administrar tipos de actividades. El detalle del resto de diagramas de secuencia para manejo de catálogos está en el Anexo 2.

F2: Administrar tipo de actividad

F2.1: Ingresar tipo de actividad

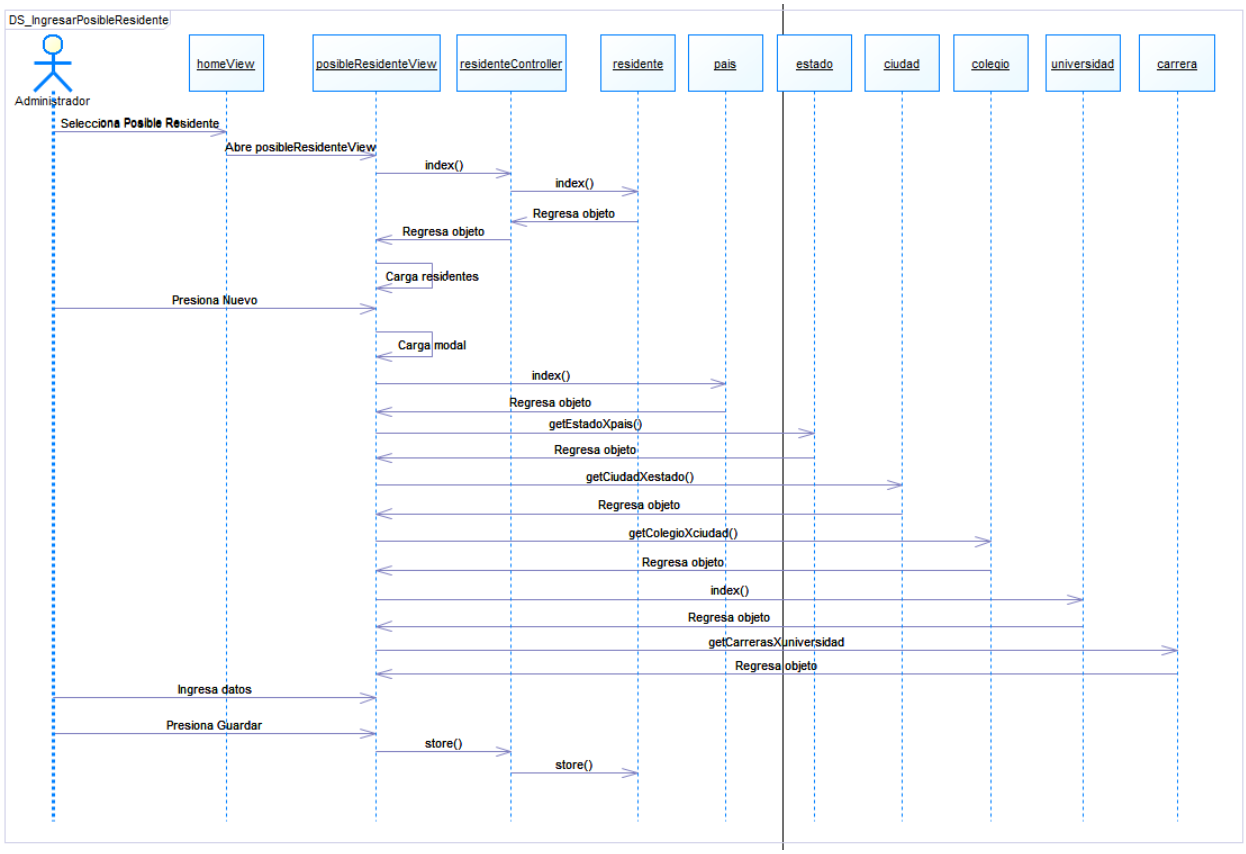


F2.2: Modificar tipo de actividad

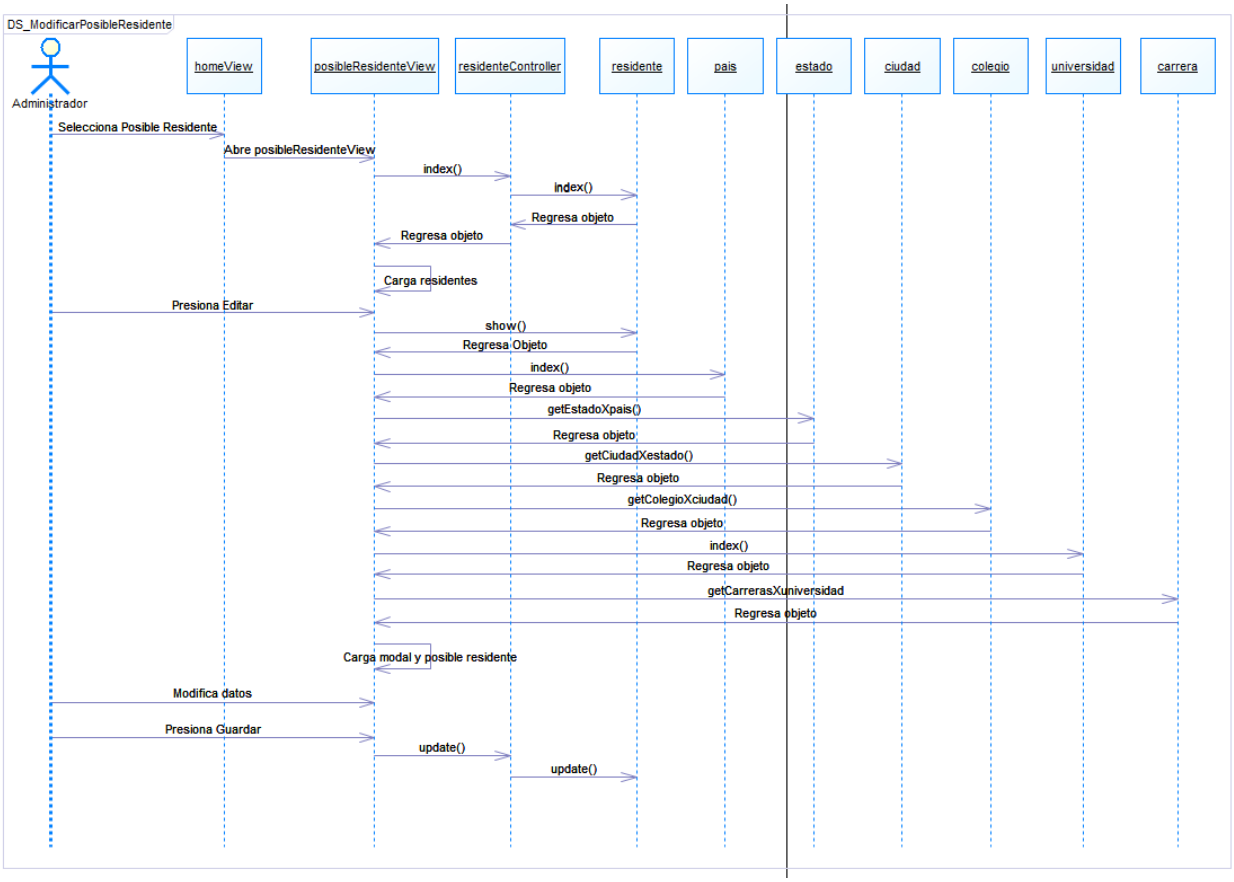

Administrador
homeView
tipoActividadView
tipoActividadController
tipoActividad
Seleciona Tipos Actividades
Abre tipoActividadView
index()
index()
Regresa objeto
Regresa objeto
Carga tipo de actividades
Selecciona editar
show()
show()
Regresa objeto
Regresa objeto
Carga modal y tipo actividad
Edita datos
Presiona Guardar
update()
update()

F11: Administrar posibles residentes

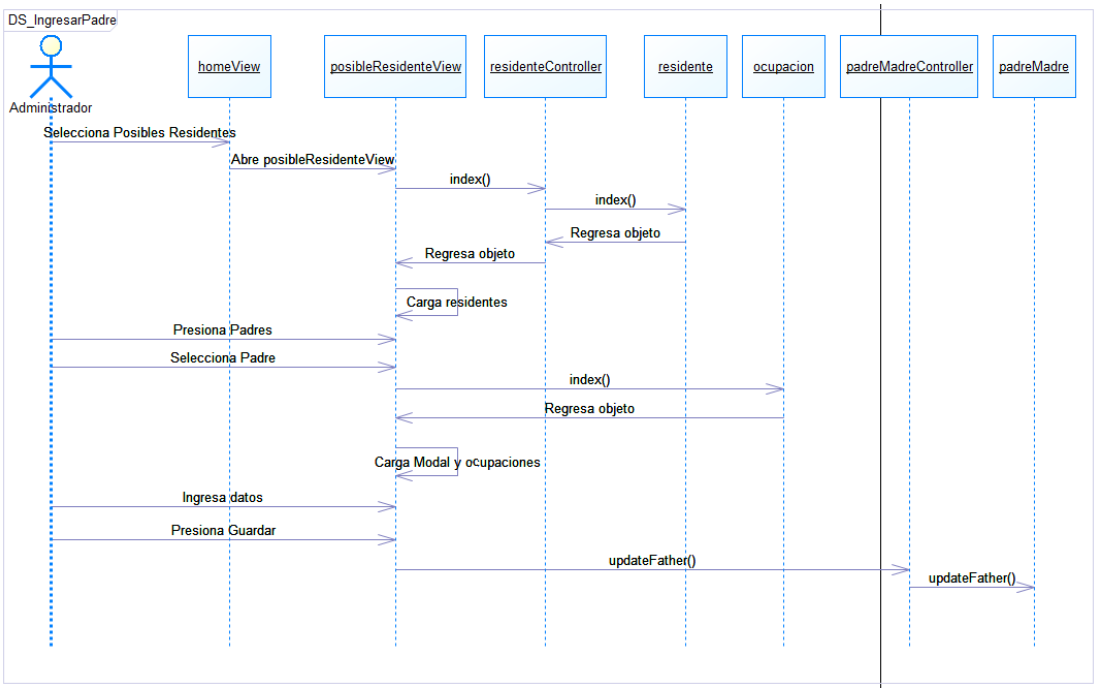
F11.1: Ingresar posible residente



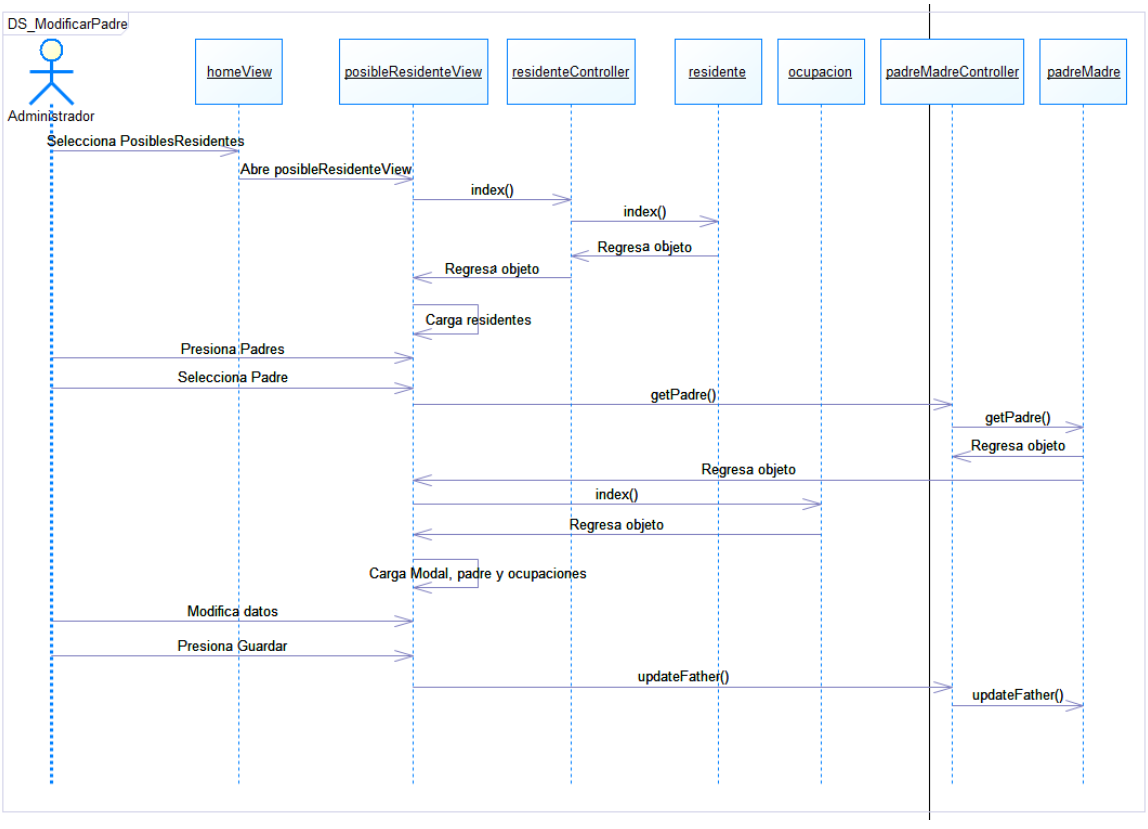
F11.2: Modificar posible residente



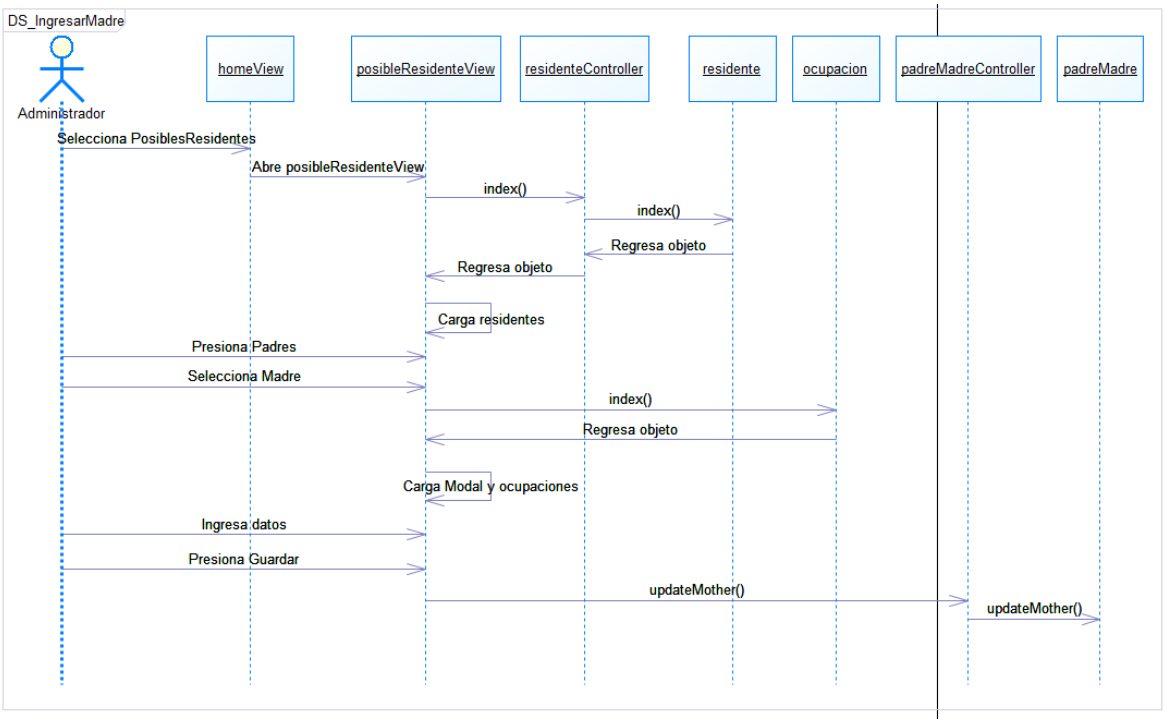
F11.3: Ingresar padre



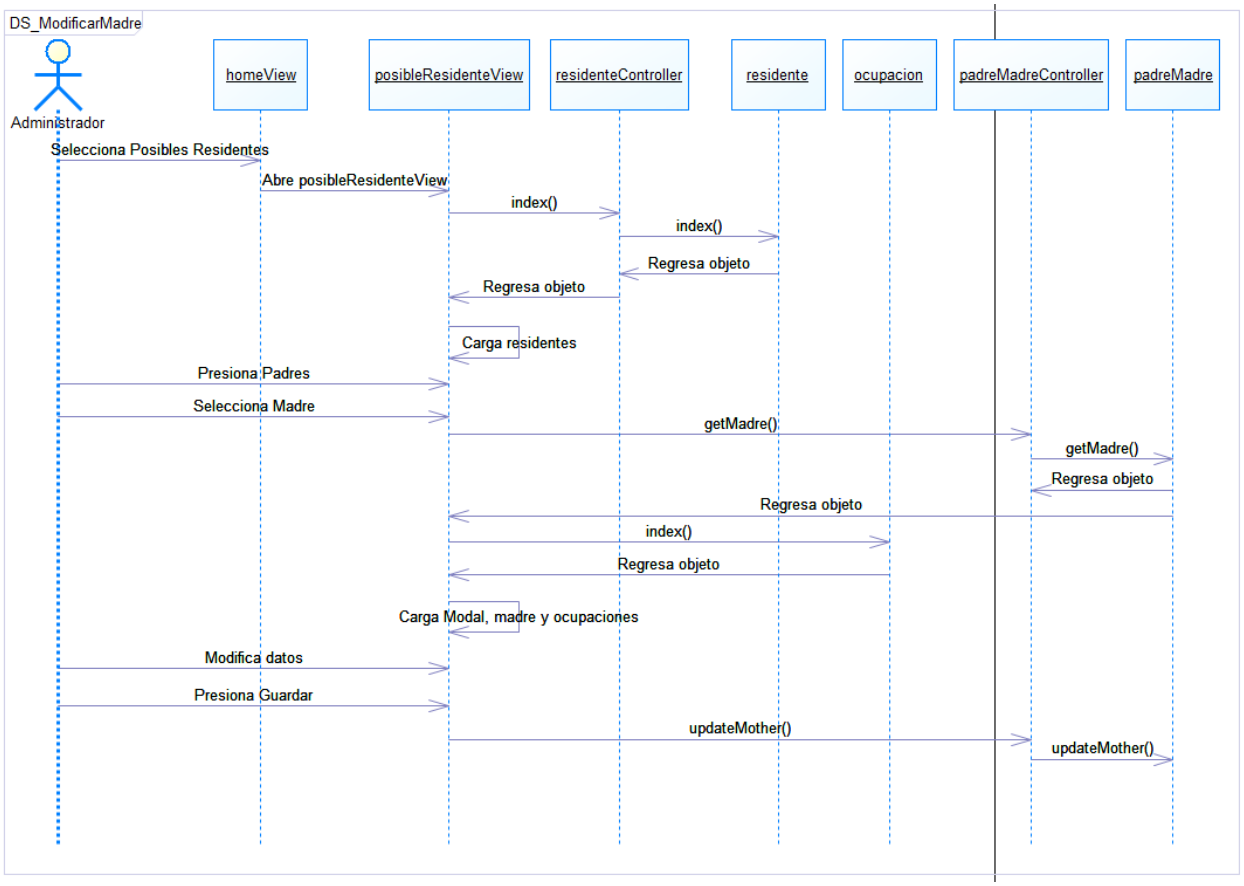
F11.4: Modificar padre



F11.5: Ingresar madre

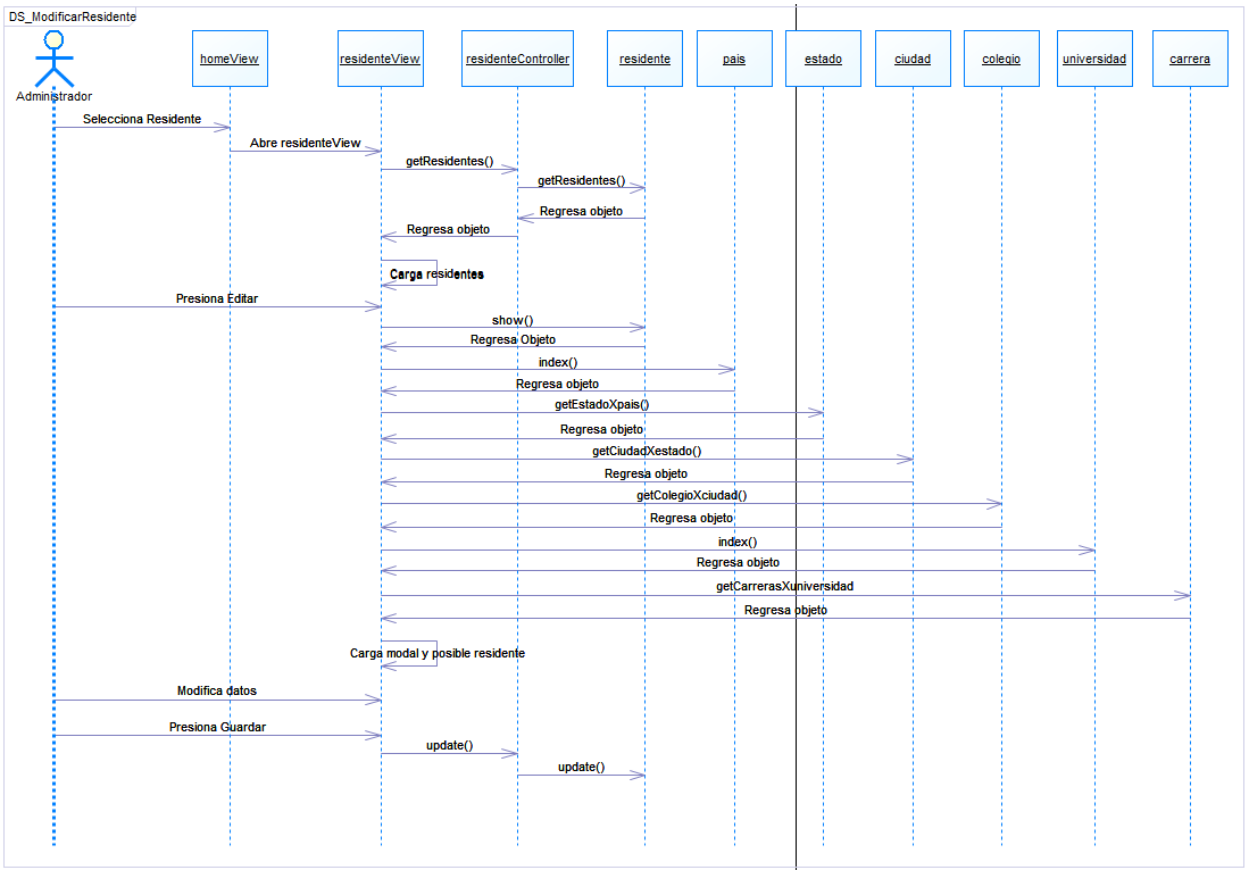


F11.6: Modificar madre

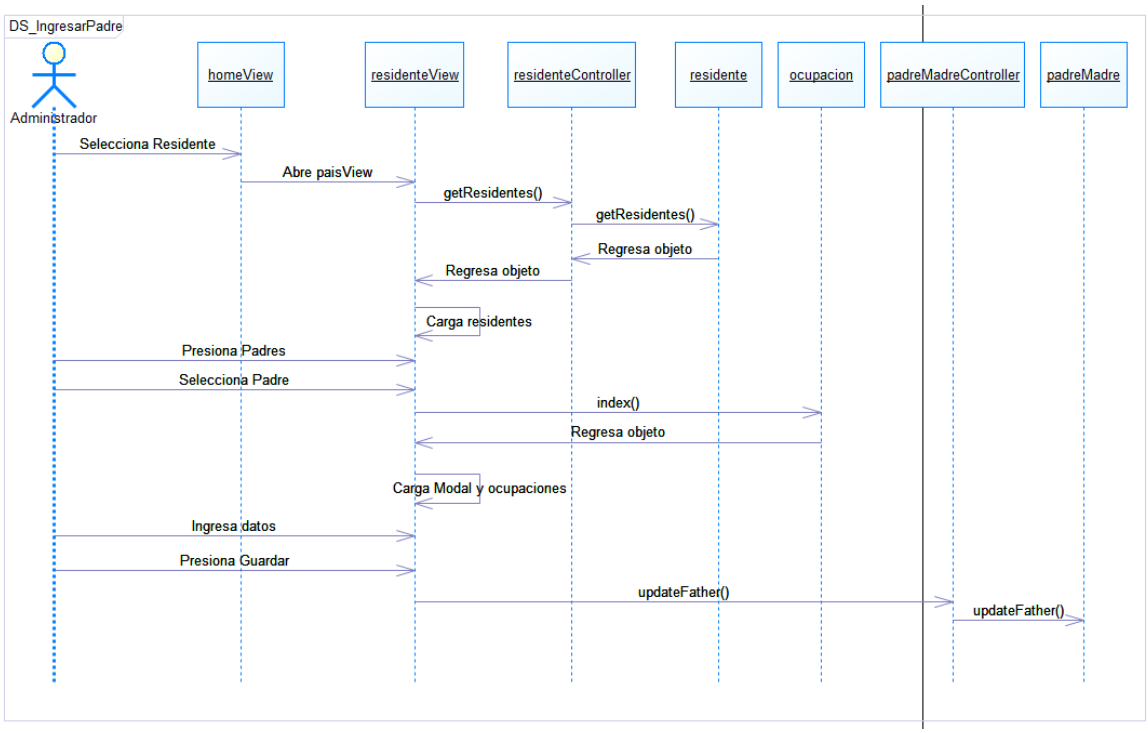


F12: Administrar residentes

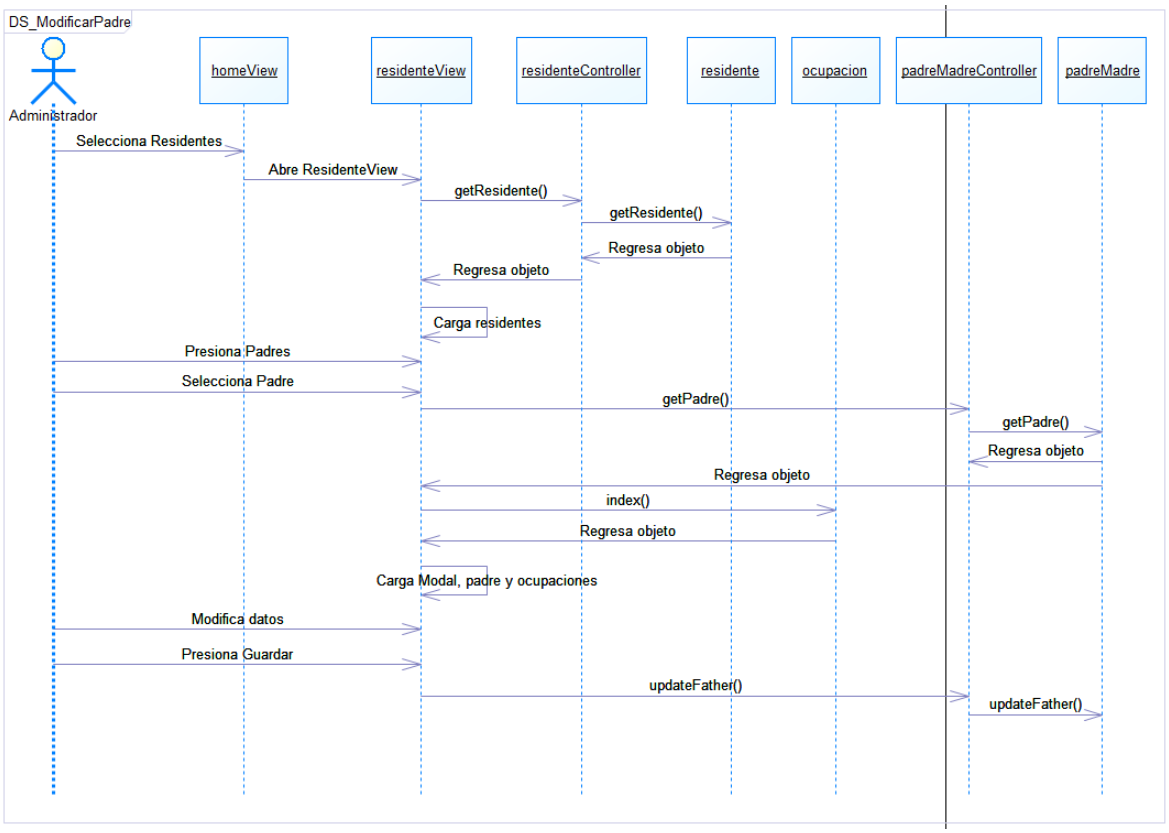
F12.2: Modificar residente



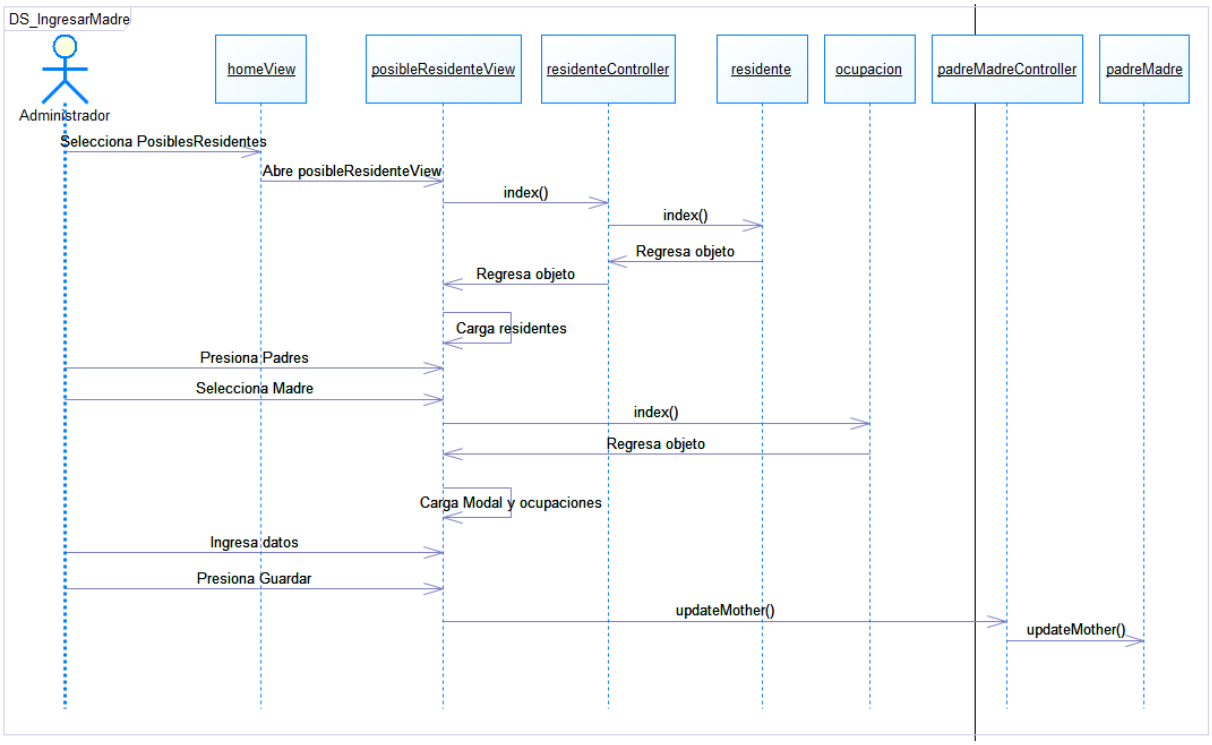
F12.3: Ingresar padre



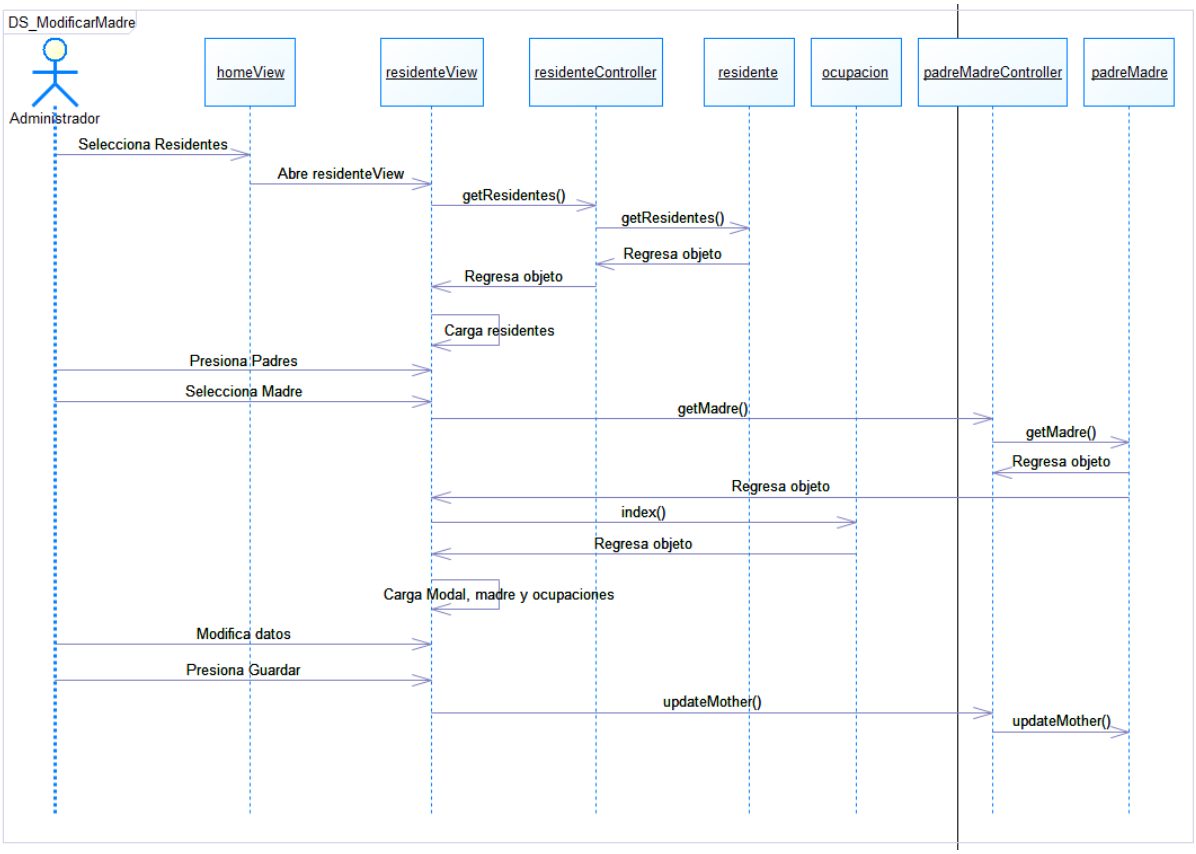
F12.4: Modificar padre



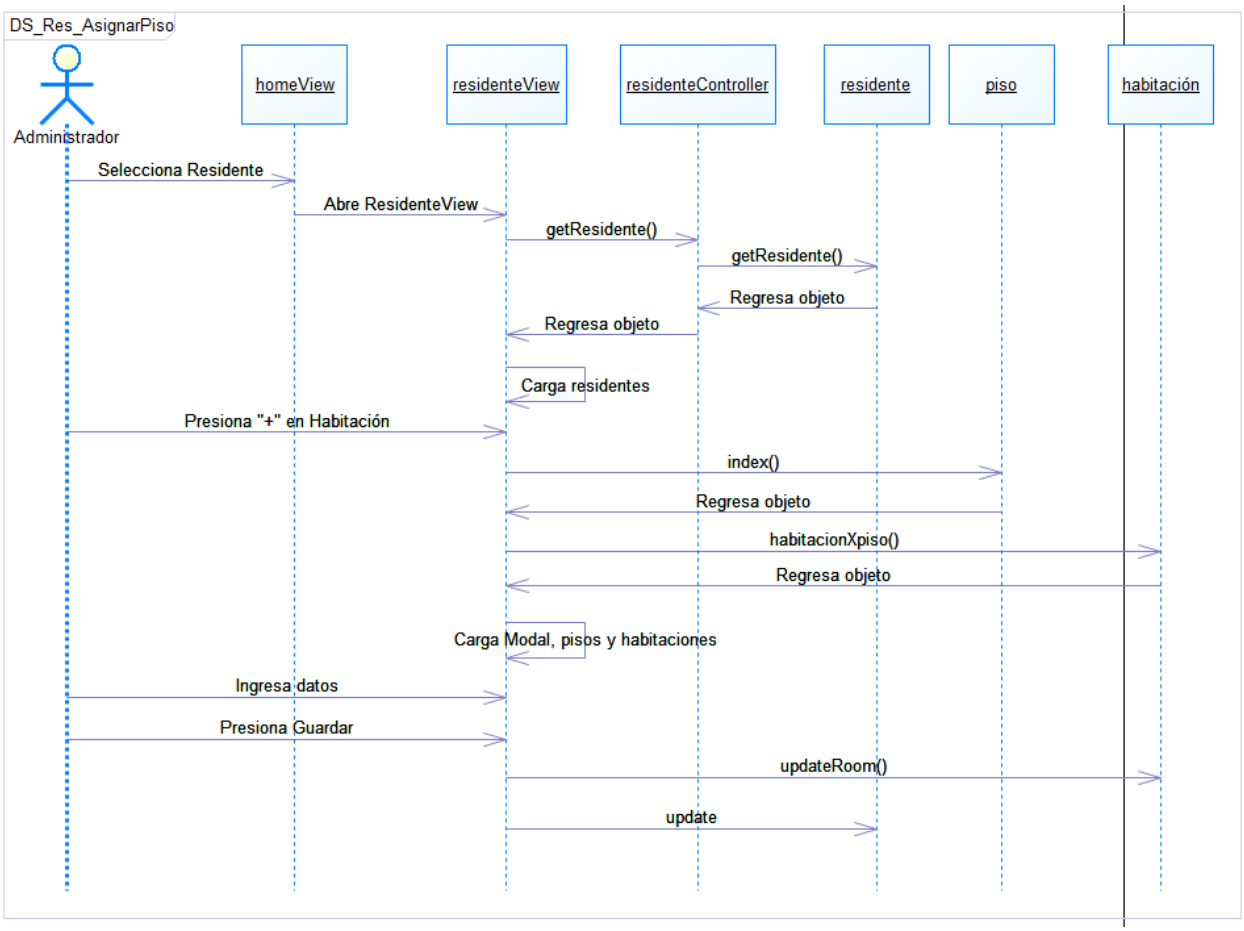
F12.5: Ingresar madre



F12.6: Modificar madre

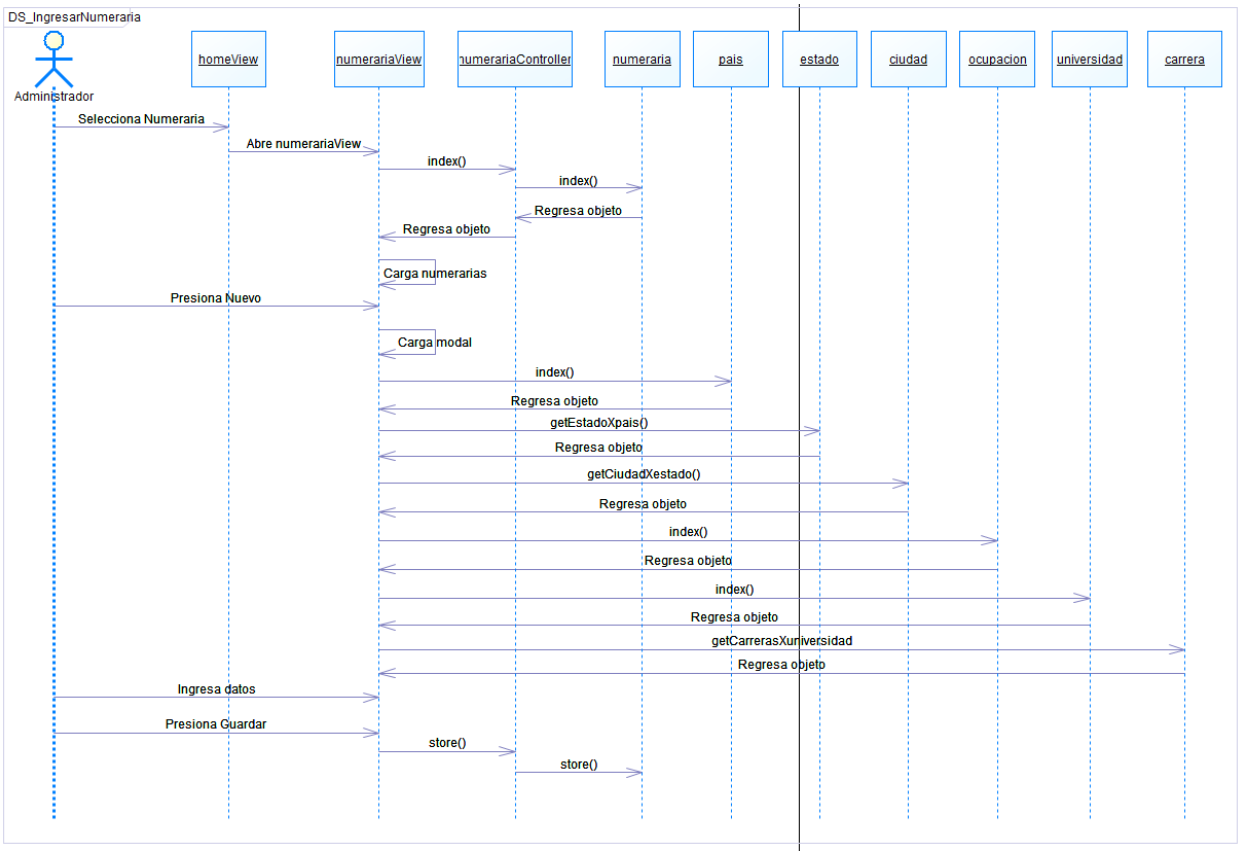


F12.7: Asignar piso

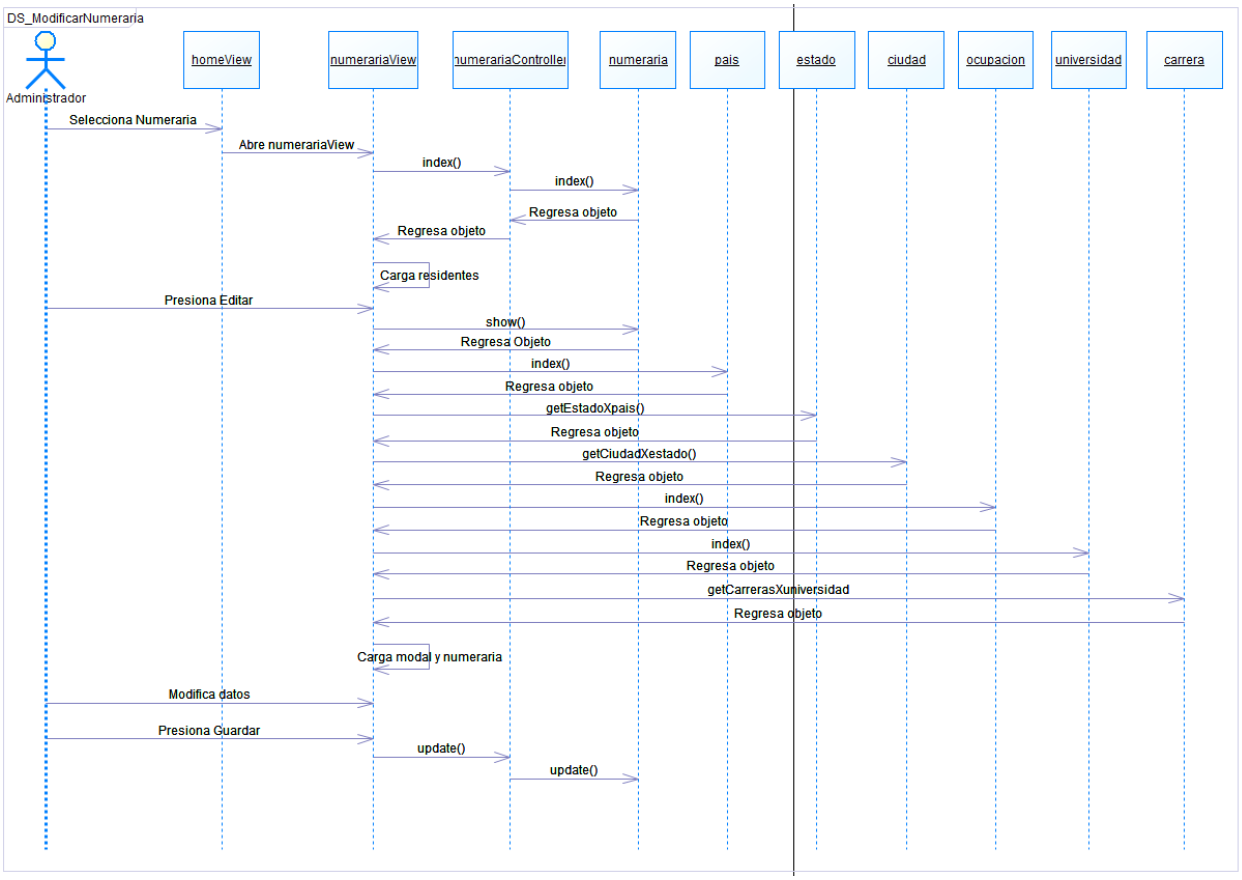


F13: Administrar numeraria

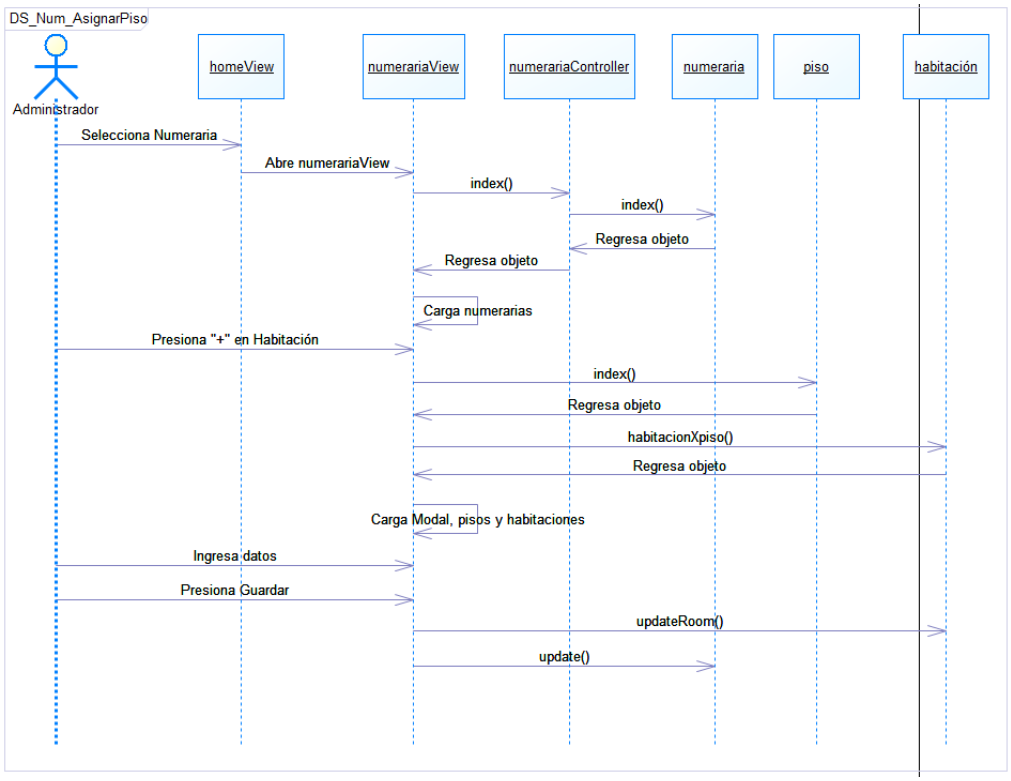
F13.1: Ingresar numeraria



F14.1: Modificar numeraria

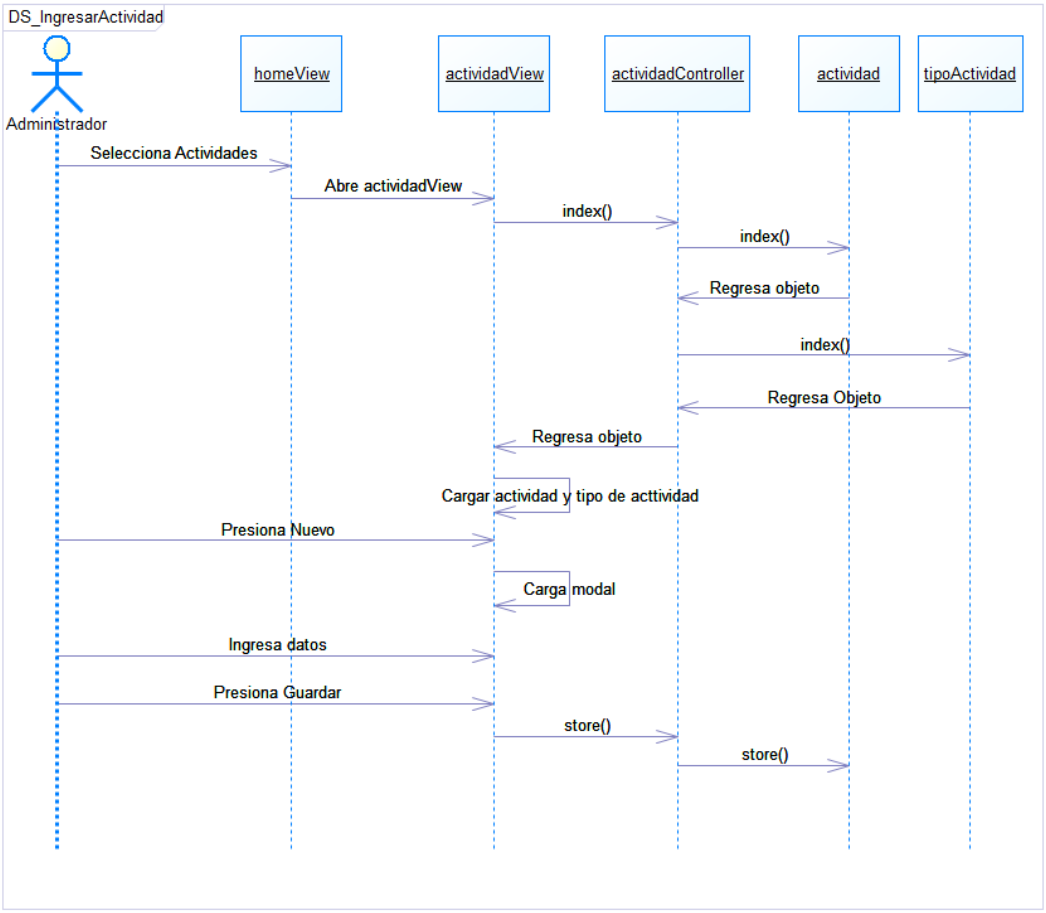


F14.2 Asignar piso

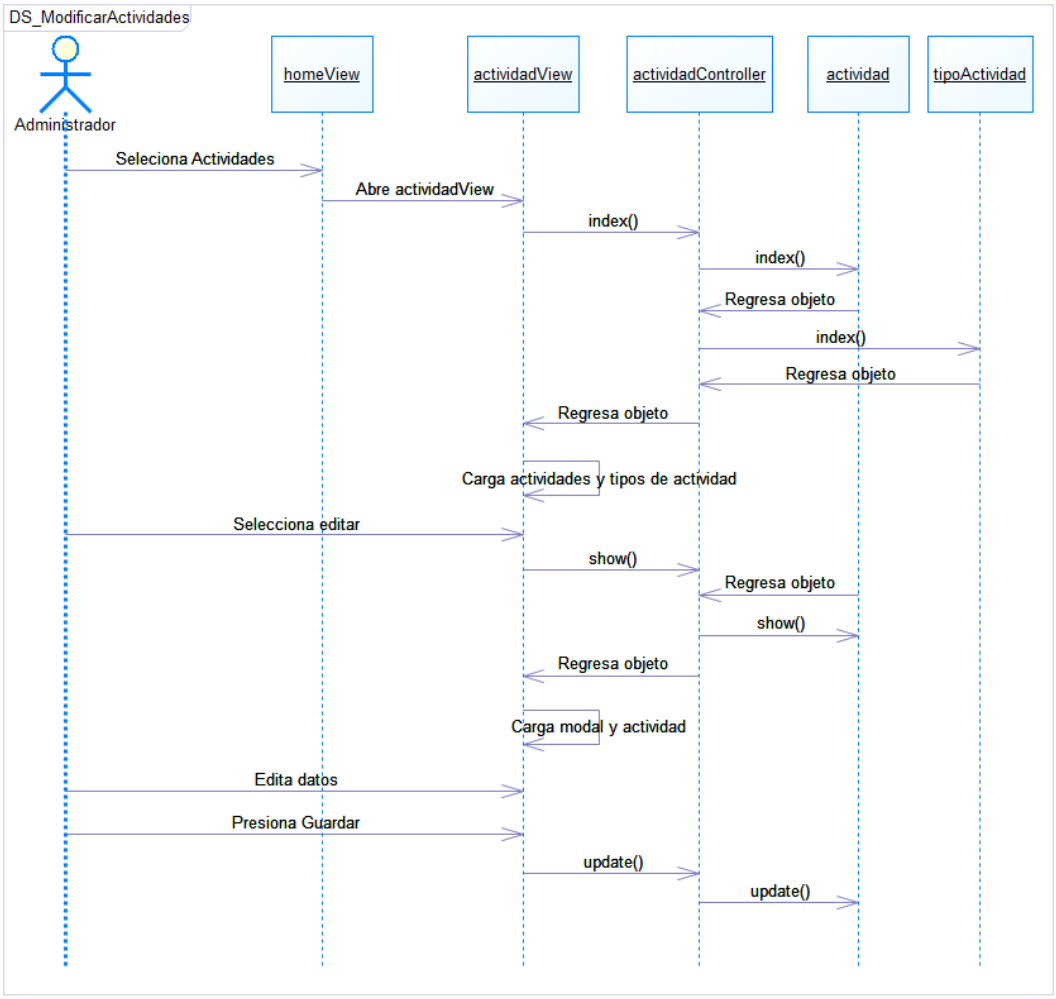


F14: Administrar actividades

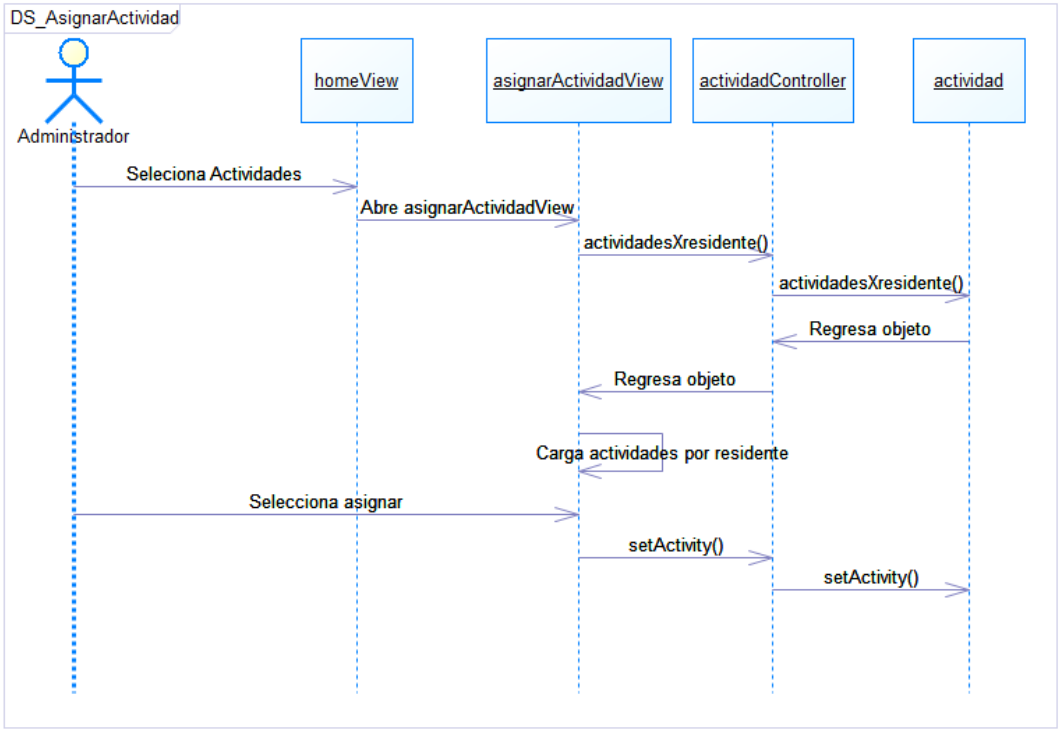
F14.1: Ingresar actividad



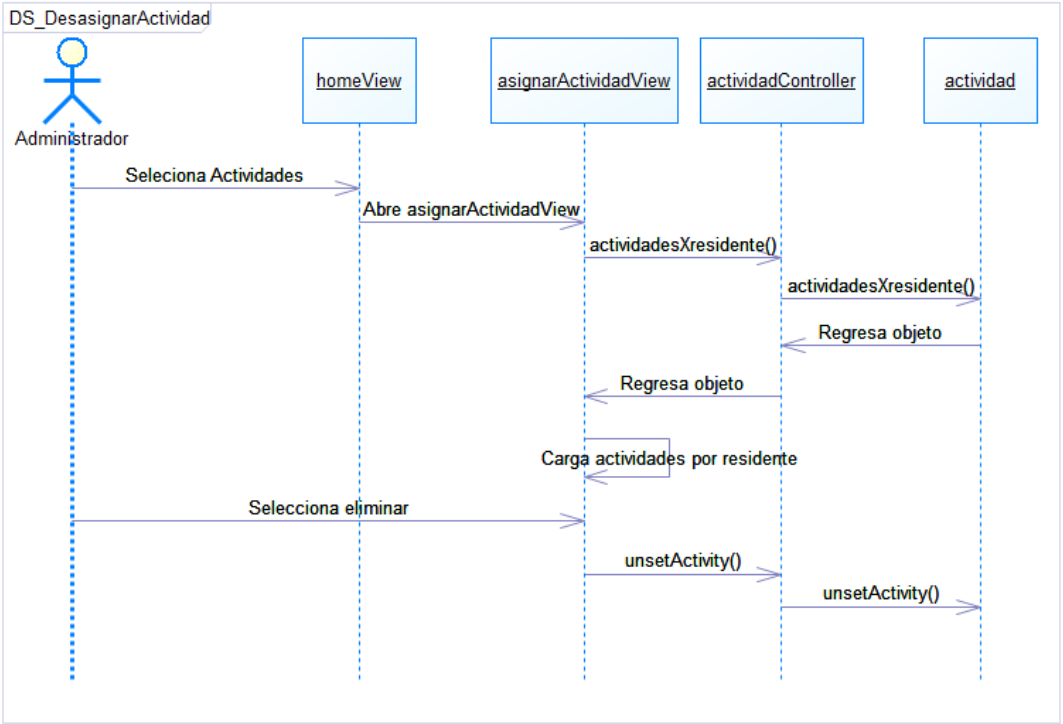
F14.2: Modificar actividad



F14.3: Asignar actividad



F14.4: Desasignar actividad



3.2.4 Diagrama entidad relación

Siempre que se va a crear un sistema es necesario tener un modelo de datos sobre el cual vamos a trabajar, podemos definir que un modelo es: “una representación de la realidad que contiene las características generales de algo que se va a realizar. En base dde datos, esta representación la elaboramos de forma gráfica” (Gómez, y otros, 2017).

El diagrama conceptual o también llamado Modelo Relacional se encarga de presentar al usuario la forma en que se van a relacionar los datos. Para poder construir este diagrama debemos crear tablas que representan los objetos que van a interactuar en el sistema, dentro de las tablas se debe incluir características de dichos objetos, representadas por columnas, por último estas tablas se deben relacionar unas con otras según su necesidad.

En la Figura 6 se ilustra el diagrama entidad-relación de este proyecto de disertación.

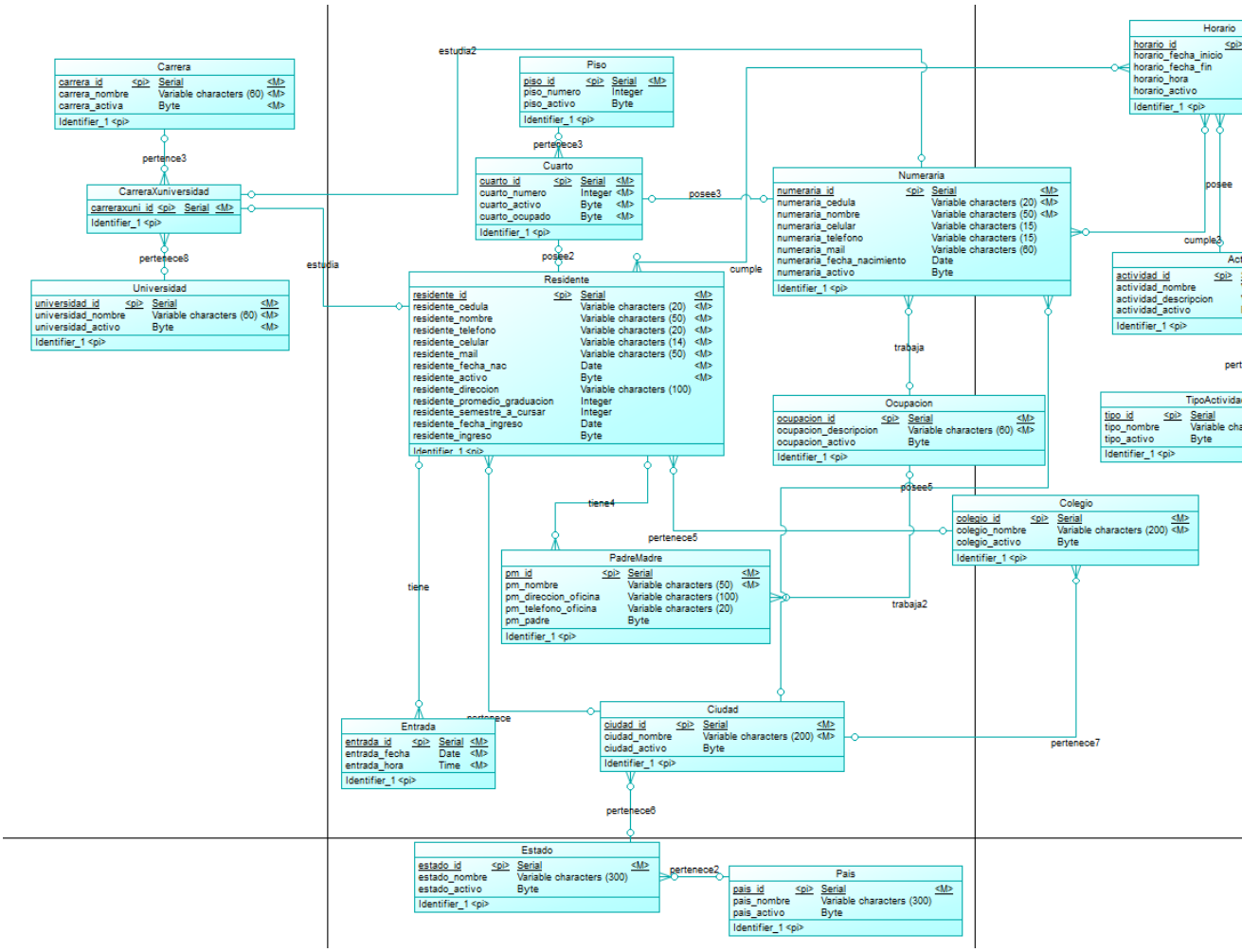


Figura 6: Diagrama entidad-relación (Mishelle Zambonino, 2020)

4. CAPÍTULO 4: Desarrollo y pruebas el sistema

En presente capítulo se detalla lo que se realizó en la fase de implementación, pruebas y mantenimiento de la disertación. También se incluye parte de código fuente, con sus aspectos más representativas, el resultado de las pruebas con las encargadas de la residencia, además de un documento que ayudará después de la implantación a reportar cualquier inconveniente que se presente, de esta manera se manejará el proceso con el respectivo orden y control.

4.1 Implementación:

Un aspecto importante al momento de implementar el código es la conexión con la base de datos, ya que es la información que se va a manejar, los parámetros necesarios para esta conexión se encuentran en el archivo *.env*, este lugar se especifican los siguientes elementos:

- DB_CONNECTION: Nombre del motor de base de datos.
- DB_HOST: Repositorio en el que se encuentra la base de datos, en este caso está en ambiente local.
- DB_PORT: Puerto en el que se encuentra la base de datos.
- DB_DATABASE: Nombre de la base de datos.
- DB_PASSWORD: Contraseña de la base de datos.

DB_CONNECTION=mysql ()

DB_HOST=127.0.0.1

DB_PORT=3306

DB_DATABASE=residencia

DB_USERNAME=root

DB_PASSWORD=

Por otro lado, en el archivo *database.php* se poblaron otros datos que complementan la conexión con la base de datos. En este archivo se colocan los mismos datos que en el anterior. Además de configuraciones extra que se pueden manejar en caso de ser necesario, en el caso de este proyecto no fue necesario llenar ningún otro campo.

```
'mysql' => [
        'driver' => 'mysql',
        'url' => env('DATABASE_URL'),
        'host' => env('DB_HOST', '127.0.0.1'),
        'port' => env('DB_PORT', '3306'),
        'database' => env('DB_DATABASE', 'residencia'),
        'username' => env('DB_USERNAME', 'root'),
        'password' => env('DB_PASSWORD', ''),
        'unix_socket' => env('DB_SOCKET', ''),
        'charset' => 'utf8mb4',
        'collation' => 'utf8mb4_unicode_ci',
        'prefix' => '',
        'prefix_indexes' => true,
        'strict' => true,
        'engine' => null,
        'options' => extension_loaded('pdo_mysql') ? array_filter([
            PDO::MYSQL_ATTR_SSL_CA => env('MYSQL_ATTR_SSL_CA'),
        ]) : [],
    ],
```

Dentro de este trabajo de disertación, el código utilizado para manejar los catálogos es similar en su gestión, por lo tanto, al igual que en los casos de uso se presentará a continuación el código más relevante, en este caso F2: Administrar tipos de actividades. El resto del código se encuentra en el Anexo 3. Dado que el trabajo se basa en el patrón MVC, se muestra a continuación los principales componentes para el CRUD.

En caso de la capa de modelo, en el archivo *tipoActividad.php* se debe colocar el nombre de la tabla, la llave primaria de dicha tabla. Además de la propiedad **filliable**, significa que los datos que se guarden en la tabla serán solo los especificados en el arreglo. En el caso de este proyecto, se debe incluir una línea de código "public $timestamps = false;", esto desactiva la opción que tiene por defecto el framework Laravel de agregar datos en los atributos **created_at** y **updated_at**, datos que no tienen las tablas de este proyecto.

```
<?php

namespace App;

use Illuminate\Database\Eloquent\Model;


class tipoActividad extends Model
{
    protected $table = 'tipoactividad';
    protected $primary_key = 'TIPO_ID';
    public $timestamps = false;

    protected $fillable =
```

```
        [
            'TIPO_ID',
            'TIPO_NOMBRE',
            'TIPO_ACTIVO',
        ];
    }
```

En cuanto a la capa de controlador, en el archivo *tipoActividadController.php* se encuentran todas las funciones que se van a usar en el manejo del CRUD.

La función **index** es la encargada de traer todos los datos se llama al modelo mencionado anteriormente, y se envían dichos datos a la vista. En cuanto a la función **store** se usa otro método, no se llama directamente al modelo, sino que se crea un arreglo con los datos y se ingresa a la base, al igual que el método **update**.

```
    public function index()
        {
            $tipoActividad = tipoActividad::get();
            json_encode($tipoActividad);
            return view('/tipoActividadView',['lista'=>$tipoActividad]);
        }
    public function store(Request $request)
        {
            $tipoActividadActivo=1;
            $data = array('TIPO_NOMBRE'=>$request-
>input('TIPO_NOMBRE'),'TIPO_ACTIVO'=>$tipoActividadActivo);
            DB::table('tipoactividad')->insert($data);
            return back();
```

}

Dentro de la capa de vista, en el archivo *tipoActividad.blade.php*, gracias a la integración que permite html con php, con el bucle "@foreach" permite lista en la tabla todos los datos que envía el controller, dentro del **foreach** se debe definir el nombre del elemento que se va a usar. Para poder listar un atributo de un objeto es necesario seguir la siguiente sintaxis: el elemento debe estar entre llaves dobles ({{ }}), el nombre del objeto debe ir junto con un símbolo de dólar ($), con una flecha se detalla el atributo del objeto a presentar ($elemento->tipoActividad_id)

```
@foreach($lista as $elemento)
            <tbody>
               <tr>
                  <td>
                     <span class="custom-checkbox">
                        <input type="checkbox" id="checkbox1" name="options[]" value
="1">
                        <label for="checkbox1"></label>
                     </span>
                  </td>
                  <td>{{$elemento->TIPO_ID}}</td>
                  <td>{{$elemento->TIPO_NOMBRE}}</td>
                  @if($elemento->TIPO_ACTIVO == 1)
                     <td><input type=checkbox class="custom-
checkbox" id="CB{{$elemento->TIPO_ACTIVO}}" checked disabled="true"></td>
```

```
                    @else
                        <td><input type=checkbox class="custom-
checkbox" id="CB{{$elemento->TIPO_ACTIVO}}" disabled="true"></td>
                    @endif
                    </td>
```

Otro aspecto importante a destacar es que, en los modales creados, es necesario incluir la url del servicio que se va a llamar, sea para ingresar o modificar los datos.

```
    <!-- Edit Modal HTML -->
    <div id="editModal-{{$elemento->TIPO_ID}}" class="modal fade">
        <div class="modal-dialog">
            <div class="modal-content">
                <form action="/tipoActividadUpdate/{{$elemento-
>TIPO_ID}}" method="get" id="forma">
                    <div class="modal-header">
                        <h4 class="modal-title">Editar</h4>
                        <button type="button" class="close" data-dismiss="modal" aria-
hidden="true">&times;</button>
                    </div>
                    <div class="modal-body">
                        <div class="form-group">
                            <label for="id">ID:</label>
                            <input type="text" name="TIPO_ID" id="TIPO_ID" class="form-
control" value="{{$elemento->TIPO_ID}}" disabled="true" required>
```

```
                    </div>
                    <div class="form-group">
                        <label for="descripcion">Nombre:</label>
                        <input type="text" id="TIPO_NOMBRE" name="TIPO_NOMBRE"
 class="form-control" value="{{$elemento->TIPO_NOMBRE}}" required>
                    </div>
                    <div class="form-group">
                        <label>Activo:</label><br>
                        @if($elemento->TIPO_ACTIVO == 1)
                            <input type=checkbox class="custom-
checkbox" id="CB{{$elemento->TIPO_ACTIVO}}" name="TIPO_ACTIVO" checked>
                        @else
                            <input type=checkbox class="custom-
checkbox" id="CB{{$elemento->TIPO_ACTIVO}}" name="TIPO_ACTIVO">
                        @endif
                    </div>
                </div>
                <div class="modal-footer">
                    <input type="button" class="btn btn-default" data-
dismiss="modal" value="Cancelar">
                    <input type="submit" class="btn btn-info" value="Guardar">
                </div>
            </form>
```

```
            </div>
        </div>
    </div>
```

En cuanto al proceso de administrar posibles residentes y residentes, usan el mismo modelo y controlador. En cuanto al manejo en el archivo *residente.php* es igual al descrito en F2: Administrar tipos de actividades.

```
<?php

namespace App;

use Illuminate\Database\Eloquent\Model;

class residente extends Model
{
    protected $table = 'residente';
    protected $primary_key = 'RESIDENTE_ID';
    public $timestamps = false;

    protected $fillable =
    [
        'RESIDENTE_ID',
        'CUARTO_ID',
```

```
        'COLEGIO_ID',
        'CIUDAD_ID',
        'CARRERAXUNI_ID',
        'RESIDENTE_CEDULA',
        'RESIDENTE_NOMBRE',
        'RESIDENTE_TELEFONO',
        'RESIDENTE_CELULAR',
        'RESIDENTE_MAIL',
        'RESIDENTE_FECHA_NAC',
        'RESIDENTE_ACTIVO',
        'RESIDENTE_DIRECCION',
        'RESIDENTE_PROMEDIO_GRADUACION',
        'RESIDENTE_SEMESTRE_A_CURSAR',
        'RESIDENTE_FECHA_INGRESO',
        'RESIDENTE_INGRESO'
    ];
}
```

En cuanto al **controller**, se puede destacar que, antes de ingresar las residentes, se valida en la base de datos que no exista un registro con el mismo número de cédula, esta parte del código se encuentra en el condicional if. En esta función también se puede recalcar que se fusiona un **query** de consulta con uno de ingreso para poder completar el registro, en este caso, para poder asignar el valor de carrera por universidad.

```
public function store(Request $request)
```

```
        {
            $carrerauni = 0;
            $carreraXuni = 0;
            $iddef=0;
            $residente = DB::table('residente')-
>where('RESIDENTE_CEDULA', '=', $request->RESIDENTE_CEDULA)->get();
            if(sizeof($residente) != 0){
                return back()->with('alert', 'Cédula ya ingresada');
            }
            else{
                $carreraXuni = DB::table('carreraxuniversidad')
                ->where('CARRERA_ID', '=', $request->CARRERA_ID)
                ->where('UNIVERSIDAD_ID', '=', $request->UNIVERSIDAD_ID)
                ->select('CARRERAXUNI_ID')
                ->get()
                ->toArray();
                json_encode($carreraXuni);
                $id=(int)$carreraXuni[0]->CARRERAXUNI_ID;
                $residenteActivo=1;
                $residenteIngreso=0;
                $data = array('RESIDENTE_NOMBRE'=>$request-
>input('RESIDENTE_NOMBRE'),
                'RESIDENTE_CEDULA'=>$request->input('RESIDENTE_CEDULA'),
```

```
            'CIUDAD_ID'=>$request->input('CIUDAD_ID'),
            'COLEGIO_ID'=>$request->input('COLEGIO_ID'),
            'RESIDENTE_MAIL'=>$request->input('RESIDENTE_MAIL'),
            'RESIDENTE_DIRECCION'=>$request-
>input('RESIDENTE_DIRECCION'),
            'RESIDENTE_TELEFONO'=>$request->input('RESIDENTE_TELEFONO'),
            'RESIDENTE_CELULAR'=>$request->input('RESIDENTE_CELULAR'),
            'RESIDENTE_SEMESTRE_A_CURSAR'=>$request-
>input('RESIDENTE_SEMESTRE_A_CURSAR'),
            'RESIDENTE_PROMEDIO_GRADUACION'=>$request-
>input('RESIDENTE_PROMEDIO_GRADUACION'),
            'RESIDENTE_FECHA_NAC'=>$request-
>input('RESIDENTE_FECHA_NAC'),
            'RESIDENTE_FECHA_INGRESO'=>$request-
>input('RESIDENTE_FECHA_INGRESO'),
            'RESIDENTE_INGRESO'=>$residenteIngreso,
            'RESIDENTE_ACTIVO'=>$residenteActivo,
            'CARRERAXUNI_ID'=>$id);
            DB::table('residente')->insert($data);
            return back();
        }
    }
```

Para la parte visual de residentes y posibles residentes, lo primero que se debe tomar en cuenta que se manejan en pantallas diferentes. La diferencia entre las dos es que, en la pantalla de posibles residentes, se encuentran los registros de la tabla que tengan valor 0 en el campo **esResidente**, y en la de residentes, las que tengan ese campo con valor 1. Otra diferencia entre los dos es que, en el de residentes podemos asignar un piso a la persona, en la otra vista no lo hace. Se presenta ese bloque de código que indica la diferencia de manejo. Podemos ver que dentro del formulario que vamos a llenar incluimos la url del servicio a usar, de esta manera, al dar clic en el botón de Guardar, ejecutará la función asignada que se encentra en el controller.

```
<div id="roomModal-{{$elemento->RESIDENTE_ID}}" class="modal fade">
    <div class="modal-dialog">
        <div class="modal-content">
            <form action="/residenteUpdateRoom/{{$elemento->RESIDENTE_ID}}" method="get">
                <div class="modal-header">
                    <h4 class="modal-title">Habitacion</h4>
                    <button type="button" class="close" data-dismiss="modal" aria-hidden="true">&times;</button>
                </div>
                <div class="modal-body">
                    <div class="form-group">
                        <label for="id">Piso:</label>
```

```
                        <select id="PISO{{$elemento-
>RESIDENTE_ID}}" class="comboPisos" onchange="roomFunction(this)" required>
                        @if($elemento->CUARTO_ID == '')
                            <option>- Seleccione -</option>
                        @endif
                        @foreach($listaPisos as $piso)
                        @if($piso->CUARTO_ID == $elemento->CUARTO_ID)
                            <option value="{{$piso->PISO_ID}}" selected>{{$piso-
>PISO_NUMERO}}</option>
                        @else
                            <option value="{{$piso->PISO_ID}}">{{$piso-
>PISO_NUMERO}}</option>
                        @endif
                        @endforeach
                        </select>
                    </div>
                    <div class="form-group">
                        <label for="id">Habitacion:</label>
                        <select name="CUARTO_ID" id="CUARTO-PISO{{$elemento-
>RESIDENTE_ID}}" required>
                        @foreach($listaPisos as $piso)
                        @if($piso->CUARTO_ID == $elemento->CUARTO_ID)
```

```
                    <option value="{{$piso->CUARTO_ID}}" selected>{{$piso->CUARTO_NUMERO}}</option>
                    @endif
                    @endforeach
                    </select>
                </div>
```

En cuanto al manejo de numerarias, la estructura del modelo y controlador es similar a las anteriores funcionalidades presentadas, una característica particular en el manejo de la vista de este punto, al igual que en residentes y posibles residentes, es el uso de javascript para incluir dinamismo a la página. En el siguiente bloque de código, vemos como se incluye un evento **onchange** al combobox de piso, para que, cada vez que se cambie de valor, se carguen los números de habitación del piso escogido. En la etiqueta html del combobox, se incluye el evento y el nombre de la función, y en el script, se describen los pasos a seguir cuando dicha función sea llamada, en este caso, consultar las habitaciones de piso seleccionado con la ayuda de Ajax.

```
<select name="CUARTO_ID" id="CUARTO-PISO{{$elemento->NUMERARIA_ID}}" required>
<script type="text/javascript">
                function roomFunction(select){
                    var selected = parseInt(select.value);
                    var nombreComboCuarto = "#CUARTO-"+select.id;
                    //alert("idSeleccion "+selected);
                    //alert("idCombo "+nombreComboCuarto);
                    $.ajax({
                        url: "cuartoXpiso/"+selected,
```

```
                type: "GET",
                dataType: "json",
                success: function(respuesta){
                    //alert("entro al ajax");
                    $(nombreComboCuarto).empty();
                respuesta.forEach(element => {
                    //alert(element.CARRERA_NOMBRE);
                    $(nombreComboCuarto).append('<option value='+element.CUARTO_ID
+'> '+element.CUARTO_NUMERO+' </option>')
                });
                }
            });
        }
    </script>
```

4.2 Pruebas

Las pruebas del sistema se realizaron con Ximena Endara, Directora de la residencia, y, María Magdalena Dávila, una de las personas que más interacción tendrá con el sistema. Las funcionalidades revisadas cubren lo especificado a través de los requerimientos funcionales y se resumen en la Tabla 1

Funcionalidades
F1: Administrar ocupaciones
F2: Administrar tipos de actividades
F3: Administrar universidades
F4: Administrar pisos
F5: Administrar habitaciones
F6: Administrar países
F7: Administrar estados
F8: Administrar ciudades
F9: Administrar carreras
F10: Administrar colegios

F11: Administrar posibles residentes
F12: Administrar residentes
F13: Administrar numerarias
F14: Administrar actividades

Tabla 1: Funcionalidades a probar (Mishelle Zambonino, 2020)

Para ello, se realizaron 2 sesiones de trabajo en las que se encontraron novedades que se fueron resolviendo y ajustes a la versión. Este proceso sirvió como retroalimentación por parte de la residencia para dejar la aplicación en un estado óptimo para su uso. A continuación se presenta el resumen de las pruebas realizadas, así como su resultado.

4.2.1 Reunión 1

En la primera sesión de pruebas realizada el lunes 13 de abril de 2020, se encontraron las siguientes observaciones:

- En la opción de Catálogos > Pisos, el campo de número permitía el ingreso de letras, y al momento de querer Guardar el registro, presentaba un error SQL, ya que en la base de datos no permite ingresar números.
- En la opción de Catálogos > Habitaciones, el campo de número permitía el ingreso de letras, y al momento de querer Guardar el registro, presentaba un error SQL, ya que en la base de datos no permite ingresar números.

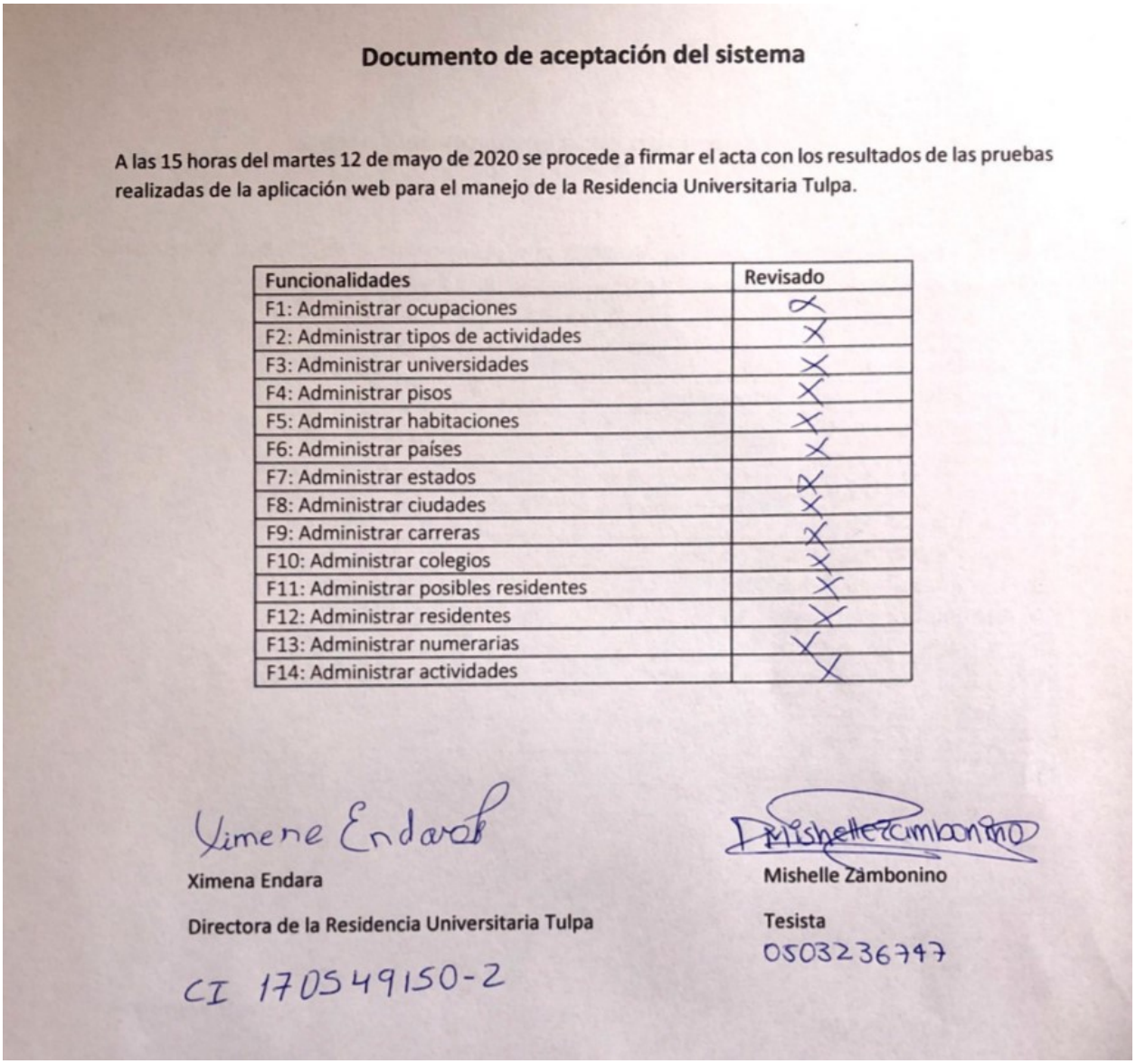
4.2.2 Reunión 2

En la segunda sesión de pruebas realizada el lunes 20 de abril de 2020, se validó la corrección de las observaciones encontradas en la primera reunión, se cambió el tipo de texto para que solo acepte números. Además se encontraron otras observaciones:

- En la opción Residencia > Numeraria, cuando se ingresa una numeraria con el mismo número de cédula que otra ya ingresada, la aplicación redirige a una pantalla de error SQL, lo que se debe hacer es poner un mensaje que diga que esa cédula ya está ingresada.

4.2.3 Documento de aceptación

Una vez ejecutadas las pruebas y ajustadas las observaciones se procedió a dejar constancia de lo realizado a través del documento de aceptación del sistema que se presenta en la Figura 7. En el documento, se detallaron las funcionalidades revisadas y probadas previamente, la Directora de la residencia Ximena Endara, procedió a firmar el documento de aceptación.



Documento de aceptación del sistema

A las 15 horas del martes 12 de mayo de 2020 se procede a firmar el acta con los resultados de las pruebas realizadas de la aplicación web para el manejo de la Residencia Universitaria Tulpa.

Funcionalidades	Revisado
F1: Administrar ocupaciones	X
F2: Administrar tipos de actividades	X
F3: Administrar universidades	X
F4: Administrar pisos	X
F5: Administrar habitaciones	X
F6: Administrar países	X
F7: Administrar estados	X
F8: Administrar ciudades	X
F9: Administrar carreras	X
F10: Administrar colegios	X
F11: Administrar posibles residentes	X
F12: Administrar residentes	X
F13: Administrar numerarias	X
F14: Administrar actividades	X

Ximena Endara

Ximena Endara

Directora de la Residencia Universitaria Tulpa

CI 1705491502

Mishelle Zambonino

Mishelle Zambonino

Tesista

0503236747

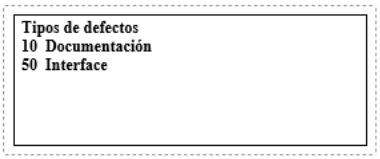
Figura 7: Aceptación del Usuario (Mishelle Zambonino, 2020)

4.3 Formulario de registro de registro de incidentes

A continuación se adjunta el formulario que se debe llenar para reportar un error. El documento tiene en la parte superior una cabecera, donde el usuario deberá colocar su nombre y la fecha de entrega del documento. Para reportar un error se necesitan llenar los siguientes datos:

- Fecha: Fecha en la que se encontró el error.
- Número: Número secuencial del error reportado.
- Tipo: Dependiendo del tipo de error, se ingresa el número respectivo.
- F. Inyección: Colocar la letra “P”, ya que el error que encontrado en producción.
- Solución: Colocar la letra “P”, ya que el error que encontrado en producción.
- Tiempo: Tiempo en el que se dio solución al error.
- Revisado: Se debe colocar una letra “X” cuando se haya resuelto.
- Descripción: Se coloca una descripción más detallada del error encontrado.

Este formulario fue realizado en base al documento de seguimiento de problemas (Formulario LOGD) de TSP (Team Software Process).



Formulario de registro de errores

Nombre: Fecha:

Fecha	Número	Tipo	F. Iny.	Solución	Tiempo	Revisado

Descripción: _____

Fecha	Número	Tipo	F. Iny.	Solución	Tiempo	Revisado

Descripción: _____

Fecha	Número	Tipo	F. Iny.	Solución	Tiempo	Revisado

Descripción: _____

5. Capítulo 5: Conclusiones y recomendaciones

5.1 Conclusiones

- Se logró analizar, desarrollar e implementar el sistema de administración de la Residencia Universitaria Tulpa ubicada en la ciudad de Quito.
- Con el uso de la metodología en cascada, se requiere que el alcance esté claro desde el inicio, dado que, el cliente no tiene una interacción constante en el proceso de desarrollo, se involucra únicamente al principio para definir los requerimientos y al final para las pruebas y revisiones de la aplicación terminada.
- Dentro de esta disertación, a pesar de que en la fase de pruebas se encontraron algunos defectos, éstos fueron de forma y no de funcionalidad, debido a que los requerimientos presentados por el cliente estaban los suficientemente definidos a causa de su conocimiento sobre los procesos que se manejan en la residencia.
- Laravel, la herramienta usada para el desarrollo permitió que se cumpla el patrón arquitectura Modelo Vista Controlador. Esto se logró creando controladores que manejan la lógica de programación, modelos que se encargan de los datos y vistas que contienen la parte visual de la aplicación.
- Dentro de la construcción de la parte visual, HTML permitió crear la estructura de todas las pantallas, Javascript ayudó a crear dinamismo en los componentes, al incluir eventos, y efectos visuales en las secciones necesarias, por ejemplo, al cargar la geografía en combobox para que se carguen los estados del país escogido, y que esta información cambie si se selecciona otro país.
- Debido a la integración de php dentro de la estructura HTML que permite Laravel, se pudieron incluir bucles para cargar la información de los controladores en las vistas.

- Las herramientas utilizadas para desarrollar el sistema, además de la arquitectura, le permiten ser escalable y mantenible a largo plazo. Esto sucede porque dentro de la arquitectura planteada, se tienen controladores, que pueden ser expuestas como apis, y éstas pueden ser llamadas y usadas dentro de otros componentes o vistas construidas según sea la necesidad del usuario.

5.2 Recomendaciones

- Para poder usar el sistema de manera correcta, se recomienda que el administrador tenga conocimientos básicos de tecnología y tenga el entrenamiento apropiado en el funcionamiento del sistema.
- Para conocer y explorar las facilidades de esta aplicación, se recomienda recurrir a los manuales de usuario y técnicos que acompañan al desarrollo de la aplicación, de esta forma se aprovechará al máximo las funcionalidades del sistema.
- Debido a que los procesos evolucionan con el tiempo, es recomendable hacer periódicamente una retroalimentación de la usabilidad de la aplicación, de donde se pueden tomar las observaciones más relevantes para hacer las modificaciones y mejoras que correspondan en la aplicación.
- Se recomienda que, se instale una herramienta de análisis de datos. De esta manera, la administración de la residencia podrá obtener información relevante del negocio, por ejemplo en qué época del año existe más afluencia de residentes, cuándo hay más entrevistas, entre otras. Esto puede ayudar a saber cuándo se debe promocionar más la residencia y marcar una diferencia competitiva para los demás centros del mismo tipo en la ciudad.

- Es recomendable complementar el módulo de registro de llegada de residentes con un registro biométrico, con esta implementación, se automatizará el proceso de registro de llegada de las residentes todos los días. Esta implementación puede optimizar el tiempo de la administradora de la residencia, además se reduce el riesgo de error humano.
- Debido a que la residencia no existe la infraestructura tecnológica como para instalar esta aplicación, se recomienda optar por un hosting compartido, ya que es económico y escalable en caso de requerir mayor capacidad de cómputo.

6. Bibliografía

Anciano, J. T. (2010). *Manual Introducción al Lenguaje HTML.* Madris: EDITRIAL CEP .

Bean, M. (2015). *Laravel 5 Essentials.* Livery Place: Packt Publishing Ltd.

Enfoques contemporaneos. (2010). Obtenido de Sistemas Inf. Gerencial: https://sistemasdeinformaciongerencial.webnode.es/enfoques-contemporaneos/

García, M. (2020). *MVC (Modelo-Vista-Controlador): ¿qué es y para qué sirve?* Obtenido de Coding or not.

Gauchat, J. D. (2012). *El gran libro de HTML5, CSS3 y Javascript.* Barcelona: MARCOMBO, S.A.

Gómez, Á. P., Jalca, J. J., García, J. G., Sánchez, O. Q., Parrales, K. M., & Merino, J. M. (2017). *Fundamentos sobre la Gestión de Base de Datos.* Alcoy: Area de Innovación y desarrollo, S. L.

Gómez, F. J., Bejarano, J. L., González, G. L., Vergara, A. P., & Rodríguez-Sabio, R. V. (2020). *5.1. Ciclo de vida clásico o en cascada*. Obtenido de ieda.

González, Y., & Romero, Y. (2012). Patrón Modelo-Vista-Controlador. *Revista Telem@ tica*, 47-57.

Gutierrez, D. (Mayo de 2011). *Universidad de los Andes Venezuela.* Obtenido de UML Diagrama de Secuencia: http://www.codecompiling.net/files/slides/UML_clase_06_UML_secuencia.pdf

Lancker, L. V. (2012). *HTML5 Los fundamentos del lenguaje* . Barcelona: Ediciones ENI.

Latinoaméricana, R. G. (2010). *El lenguaje de programación PHP*. Obtenido de http://redgrafica.com/IMG/article_PDF/article_a155.pdf

Navarrete, T. (2006). *El lenguaje Javascipt.* Obtenido de El lenguaje Javascipt: https://s3.amazonaws.com/academia.edu.documents/54012715/javascript.pdf?response-content-disposition=inline%3B%20filename%3DEl_lenguaje_JavaScript.pdf&X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=AKIAIWOWYYGZ2Y53UL3A%2F20191127%2Fus-east-1%2Fs3%2Faws

Nixon, R. (2012). *Learning PHP, MySQL, JavaScript, and CSS, 2E.* United States of America: O'Reilly Media, Inc.

Patoja, E. B. (2004). El patr´on de dise˜no Modelo-Vista-Controlador (MVC) y su implementaci´on en Java Swing. *Acta Nova*, 493-507.

Pelissier, C. (2002). *Programacion con PHP*.

Pressman, R. S. (2010). *Ingeniería del software. Un enfoque práctico.* Mexico: McGraw-Hill.

Puertas, J. P., & Borgues, E. L. (2016). *Creación de un sitio web con PHP y MySql.* Bogotá: Editorial Ra-ma.

Ruiz de la Peña, J., & Cuba Céspedes, I. (2010). Sistema de gestión de información para la Residencia Universitaria de la Universidad de Holguín "Oscar Lucero Moya". *Ciencias Holguín, XVI (1)*, 1-8.

Sagredo, J. G., Espinosa, A. T., Reyes, M. M., & García, M. d. (2013). Automatización de la codificación del patrón vista controlador en proyectos orientados a la web. *CIENCIA ergo sum*, 239-250.

Sierra, F., Acosta, J., Ariza, J., & Salas, M. (2017). Estudio y análisis de los framework en php basados en el modelo vista controlador para el. *Revista Investigación y Desarrollo en TIC*.

Zamora, S. (2018). *Enfoques contemporaneos de los sistemas de información*. Obtenido de Zamorar : https://izamorar.com/enfoques-contemporaneos-de-los-sistemas-de-informacion/

7. Anexos

7.1 Anexo 1: Casos de uso a detalle

F1: Administrar ocupaciones

F1.1: Ingresar ocupación

Descripción: Este caso de uso describe el flujo que debe seguir un actor para ingresar una nueva ocupación al sistema.

Actores: Administradores.

Flujo principal:

1. El actor selecciona la opción de Catálogos > Ocupaciones del menú principal.
2. El sistema carga la pantalla de ocupaciones con la información de la base cargada.
3. El actor presiona el botón Nueva Ocupación.
4. El sistema presenta un modal de ingreso de ocupación.
5. El actor ingresa la información de la ocupación.
6. El actor presiona el botón Guardar.
7. El sistema guarda los datos de la ocupación en la base de datos.

Excepciones:

Código	*Descripción*	*Solución*
E1	Error de conexión con la base de datos	Contactarse con el administrador de base de datos

F1.2: Modificar ocupación

Descripción: Este caso de uso describe el flujo que debe seguir un actor para modificar una ocupación existente.

Actores: Administradores.

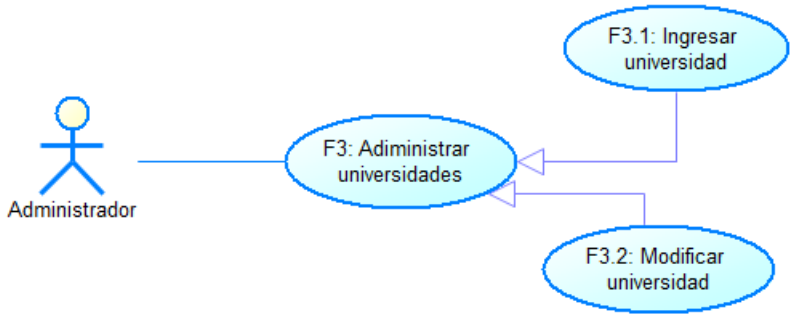
Flujo principal:

1. El actor selecciona la opción de Catálogos > Ocupaciones del menú principal.
2. El sistema carga la pantalla de ocupaciones con la información de la base cargada.
3. El actor presiona el botón Modificar de la ocupación que desee.
4. El sistema carga un modal con los datos de la ocupación.
5. El actor cambia los datos que el sistema lo permita.
6. El actor presiona el botón Guardar.
7. El sistema modifica los datos de la ocupación en la base de datos.

Excepciones:

Código	*Descripción*	*Solución*
E1	Error de conexión con la base de datos	Contactarse con el administrador de base de datos

F3: Administrar universidades



F3.1: Ingresar universidad

Descripción: Este caso de uso describe el flujo que debe seguir un actor para ingresar una nueva universidad al sistema.

Actores: Administradores.

Flujo principal:

1. El actor selecciona la opción de Catálogos > Universidades del menú principal.
2. El sistema carga la pantalla de universidades con la información de la base cargada.
3. El actor presiona el botón Nueva Universidad.
4. El sistema presenta un modal de ingreso de universidad.
5. El actor ingresa la información de la universidad.
6. El actor presiona el botón Guardar.
7. El sistema guarda los datos de la universidad en la base de datos.

Excepciones:

Código	*Descripción*	*Solución*

E1	Error de conexión con la base de datos	Contactarse con el administrador de base de datos

F3.2: Modificar universidad

Descripción: Este caso de uso describe el flujo que debe seguir un actor para modificar una universidad existente.

Actores: Administradores.

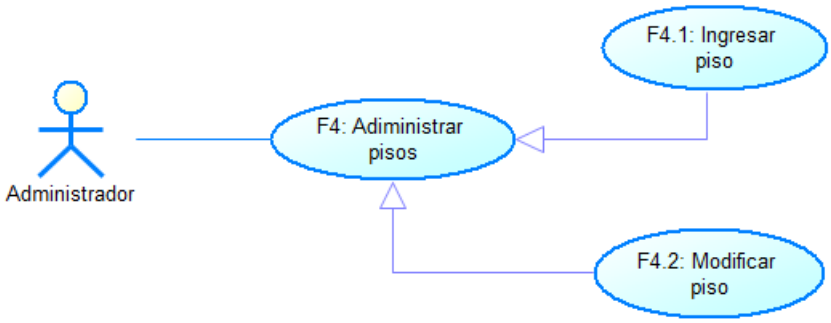
Flujo principal:

1. El actor selecciona la opción de Catálogos > Universidades del menú principal.
2. El sistema carga la pantalla de universidades con la información de la base cargada.
3. El actor presiona el botón Modificar de la universidad que desee.
4. El sistema carga un modal con los datos de la universidad.
5. El actor cambia los datos que el sistema lo permita.
6. El actor presiona el botón Guardar.
7. El sistema modifica los datos de la universidad en la base de datos.

Excepciones:

Código	*Descripción*	*Solución*
E1	Error de conexión con la base de datos	Contactarse con el administrador de base de datos

F4: Administrar pisos



F4.1: Ingresar piso

Descripción: Este caso de uso describe el flujo que debe seguir un actor para ingresar un nuevo piso al sistema.

Actores: Administradores.

Flujo principal:

1. El actor selecciona la opción de Catálogos > Pisos del menú principal.
2. El sistema carga la pantalla de pisos con la información de la base cargada.
3. El actor presiona el botón Nuevo Piso.
4. El sistema presenta un modal de ingreso de piso.
5. El actor ingresa la información del piso.
6. El actor presiona el botón Guardar.
7. El sistema guarda los datos del piso en la base de datos.

Excepciones:

Código	*Descripción*	*Solución*
E1	Error de conexión con la base de datos	Contactarse con el administrador de base de datos

F4.2: Modificar piso

Descripción: Este caso de uso describe el flujo que debe seguir un actor para modificar un piso existente.

Actores: Administradores.

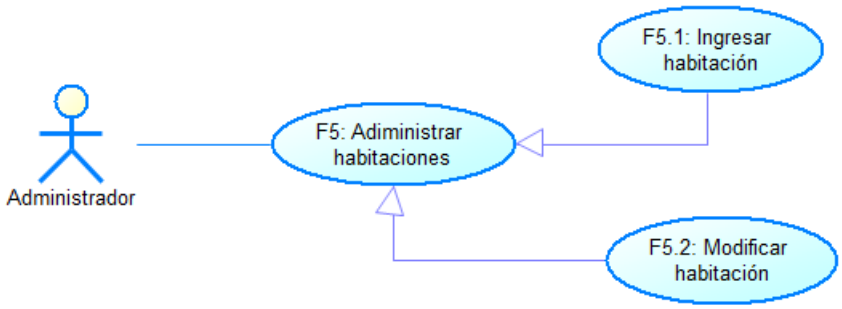
Flujo principal:

1. El actor selecciona la opción de Catálogos > Pisos del menú principal.
2. El sistema carga la pantalla de pisos con la información de la base cargada.
3. El actor presiona el botón Modificar del piso que desee.
4. El sistema carga un modal con los datos del piso.
5. El actor cambia los datos que el sistema lo permita.
6. El actor presiona el botón Guardar.
7. El sistema modifica los datos del piso en la base de datos.

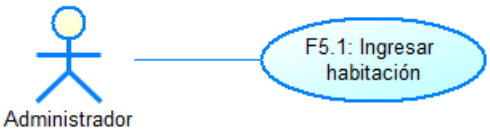
Excepciones:

Código	*Descripción*	*Solución*
E1	Error de conexión con la base de datos	Contactarse con el administrador de base de datos

F5: Administrar habitaciones



F5.1: Ingresar habitación



Descripción: Este caso de uso describe el flujo que debe seguir un actor para ingresar un nuevo piso al sistema.

Actores: Administradores.

Flujo principal:

1. El actor selecciona la opción de Catálogos > Habitaciones del menú principal.
2. El sistema carga la pantalla de habitaciones con sus respectivos pisos con la información de la base cargada.
3. El actor presiona el botón Nueva Habitación.
4. El sistema presenta un modal de ingreso de habitación.
5. El sistema carga los pisos que están activos.
6. El actor ingresa la información de la habitación.
7. El actor presiona el botón Guardar.
8. El sistema guarda los datos de la habitación en la base de datos.

Excepciones:

Código	*Descripción*	*Solución*

E1	Error de conexión con la base de datos	Contactarse con el administrador de base de datos

F5.2: Modificar habitación

Descripción: Este caso de uso describe el flujo que debe seguir un actor para modificar una habitación existente.

Actores: Administradores.

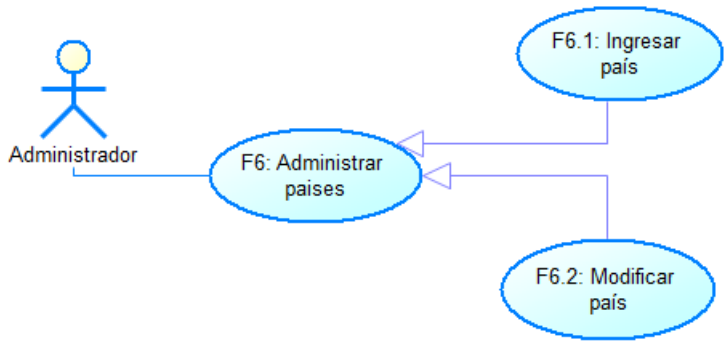
Flujo principal:

1. El actor selecciona la opción de Catálogos > Habitaciones del menú principal.
2. El sistema carga la pantalla de habitaciones con sus respectivos pisos con la información de la base cargada.
3. El actor presiona el botón Modificar de la habitación que desee.
4. El sistema carga un modal con los datos de la habitación.
5. El actor cambia los datos que el sistema lo permita.
6. El actor presiona el botón Guardar.
7. El sistema modifica los datos de la habitación en la base de datos.

Excepciones:

Código	*Descripción*	*Solución*
E1	Error de conexión con la base de datos	Contactarse con el administrador de base de datos

F6: Administrar países



F6.1: Ingresar país

Descripción: Este caso de uso describe el flujo que debe seguir un actor para ingresar un nuevo país al sistema.

Actores: Administradores.

Flujo principal:

1. El actor selecciona la opción de Catálogos > Geografía > Países del menú principal.
2. El sistema carga la pantalla de países con la información de la base cargada.
3. El actor presiona el botón Nuevo País.
4. El sistema presenta un modal de ingreso de país.
5. El actor ingresa la información del país.
6. El actor presiona el botón Guardar.
7. El sistema guarda los datos del país en la base de datos.

Excepciones:

Código	*Descripción*	*Solución*

E1	Error de conexión con la base de datos	Contactarse con el administrador de base de datos

F6.2: Modificar país

Descripción: Este caso de uso describe el flujo que debe seguir un actor para modificar un país existente.

Actores: Administradores.

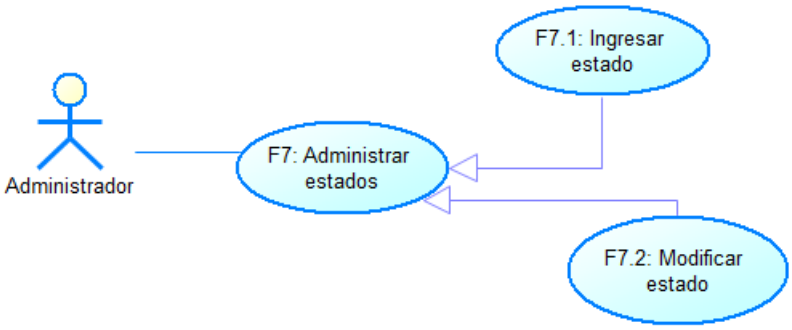
Flujo principal:

1. El actor selecciona la opción de Catálogos > Geografía > Países del menú principal.
2. El sistema carga la pantalla de países con la información de la base cargada.
3. El actor presiona el botón Modificar del país que desee.
4. El sistema carga un modal con los datos del país.
5. El actor cambia los datos que el sistema lo permita.
6. El actor presiona el botón Guardar.
7. El sistema modifica los datos del país en la base de datos.

Excepciones:

Código	*Descripción*	*Solución*
E1	Error de conexión con la base de datos	Contactarse con el administrador de base de datos

F7: Administrar estados



F7.1: Ingresar estado

Descripción: Este caso de uso describe el flujo que debe seguir un actor para ingresar un nuevo país al sistema.

Actores: Administradores.

Flujo principal:

1. El actor selecciona la opción de Catálogos > Geografía > Estados del menú principal.
2. El sistema carga la pantalla de estados con sus respectivos países con la información de la base cargada.
3. El actor presiona el botón Nuevo Estado.
4. El sistema presenta un modal de ingreso de estado.
5. El actor ingresa la información del estado.
6. El actor presiona el botón Guardar.
7. El sistema guarda los datos del estado en la base de datos.

Excepciones:

Código	*Descripción*	*Solución*

E1	Error de conexión con la base de datos	Contactarse con el administrador de base de datos

F7.2: Modificar estado

Descripción: Este caso de uso describe el flujo que debe seguir un actor para modificar un estado existente.

Actores: Administradores.

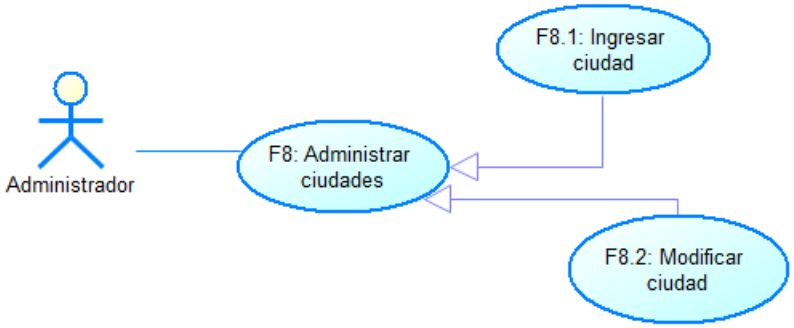
Flujo principal:

1. El actor selecciona la opción de Catálogos > Geografía > Estados del menú principal.
2. El sistema carga la pantalla de estados con sus respectivos países con la información de la base cargada.
3. El actor presiona el botón Modificar del estado que desee.
4. El sistema carga un modal con los datos del estado.
5. El actor cambia los datos que el sistema lo permita.
6. El actor presiona el botón Guardar.
7. El sistema modifica los datos del estado en la base de datos.

Excepciones:

Código	*Descripción*	*Solución*
E1	Error de conexión con la base de datos	Contactarse con el administrador de base de datos

F8: Administrar ciudades



F8.1: Ingresar ciudad

Descripción: Este caso de uso describe el flujo que debe seguir un actor para ingresar una nueva ciudad al sistema.

Actores: Administradores.

Flujo principal:

1. El actor selecciona la opción de Catálogos > Geografía > Ciudades del menú principal.
2. El sistema carga la pantalla de ciudades con sus respectivos estados y países con la información de la base cargada.
3. El actor presiona el botón Nueva Ciudad.
4. El sistema presenta un modal de ingreso de ciudad.
5. El actor ingresa la información de la ciudad.
6. El actor presiona el botón Guardar.
7. El sistema guarda los datos de la ciudad en la base de datos.

Excepciones:

Código	*Descripción*	*Solución*
E1	Error de conexión con la base de datos	Contactarse con el administrador de base de datos

F8.2: Modificar ciudad

Descripción: Este caso de uso describe el flujo que debe seguir un actor para modificar una ciudad existente.

Actores: Administradores.

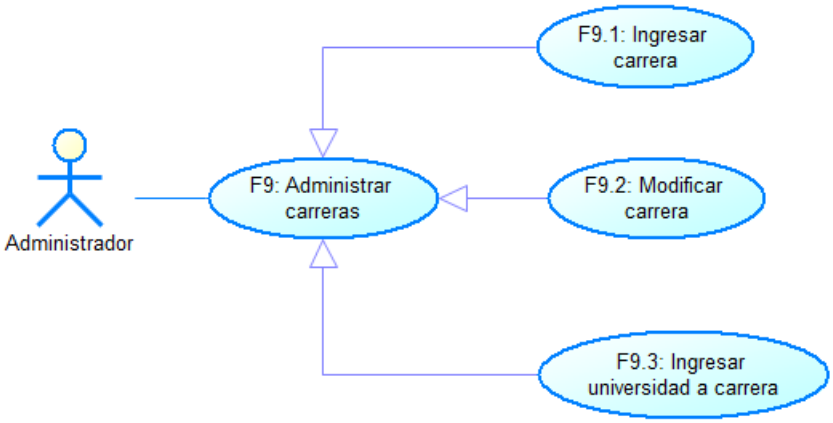
Flujo principal:

1. El actor selecciona la opción de Catálogos > Geografía > Ciudades del menú principal.
2. El sistema carga la pantalla de ciudades con sus respectivos estados y países con la información de la base cargada.
3. El actor presiona el botón Modificar de la ciudad que desee.
4. El sistema carga un modal con los datos de la ciudad.
5. El actor cambia los datos que el sistema lo permita.
6. El actor presiona el botón Guardar.
7. El sistema modifica los datos de la ciudad en la base de datos.

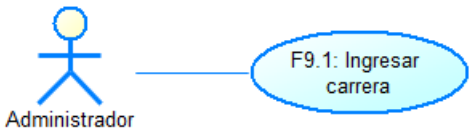
Excepciones:

Código	*Descripción*	*Solución*
E1	Error de conexión con la base de datos	Contactarse con el administrador de base de datos

F9: Administrar carreras



F9.1: Ingresas carrera



Descripción: Este caso de uso describe el flujo que debe seguir un actor para ingresar una nueva carrera al sistema.

Actores: Administradores.

Flujo principal:

1. El actor selecciona la opción de Catálogos > Carreras del menú principal.
2. El sistema carga la pantalla de carreras con la información de la base cargada.
3. El actor presiona el botón Nueva Carrera.
4. El sistema presenta un modal de ingreso de carrera.
5. El actor ingresa la información de la carrera.
6. El actor presiona el botón Guardar.
7. El sistema guarda los datos de la carrera en la base de datos.

Excepciones:

Código	*Descripción*	*Solución*
E1	Error de conexión con la base de datos	Contactarse con el administrador de base de datos

F9.2: Modificar carrera

Descripción: Este caso de uso describe el flujo que debe seguir un actor para modificar una carrera existente.

Actores: Administradores.

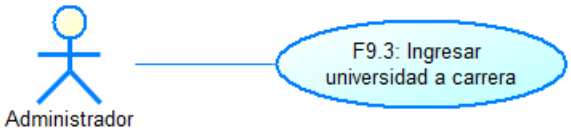
Flujo principal:

1. El actor selecciona la opción de Catálogos > Carreras del menú principal.
2. El sistema carga la pantalla de carreras con la información de la base cargada.
3. El actor presiona el botón Modificar de la carrera que desee.
4. El sistema carga un modal con los datos de la carrera.
5. El actor cambia los datos que el sistema lo permita.
6. El actor presiona el botón Guardar.
7. El sistema modifica los datos de la carrera en la base de datos.

Excepciones:

Código	*Descripción*	*Solución*
E1	Error de conexión con la base de datos	Contactarse con el administrador de base de datos

F9.3: Agregar universidad a carrera



Descripción: Este caso de uso describe el flujo que debe seguir un actor para enlazar una carrera existente con una universidad existente.

Actores: Administradores.

Flujo principal:

1. El actor selecciona la opción de Catálogos > Carreras del menú principal.
2. El sistema carga la pantalla de carreras con la información de la base cargada.
3. El actor presiona el botón Modificar de la carrera que desee.
4. El sistema carga un modal con los datos de la carrera, las universidades por carrera y todas las universidades.
5. El actor selecciona la universidad que desea aumentar.
6. El actor presiona el botón Guardar.
7. El sistema modifica los datos de la ciudad en la base de datos.

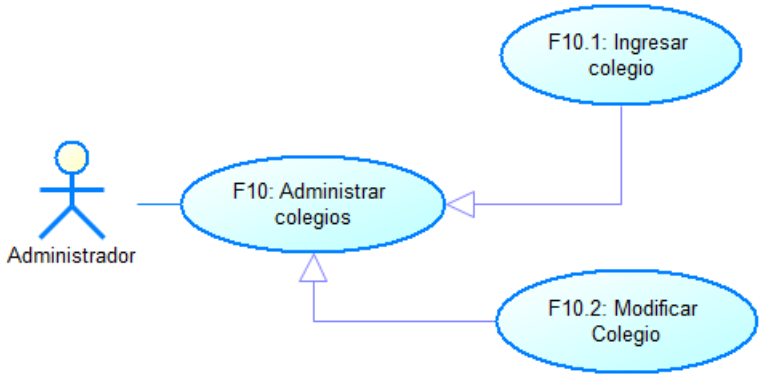
Flujo alterno:

1. En caso de que la universidad ya este asociada con la carrera el sistema no le permite ingresar y le da un mensaje.

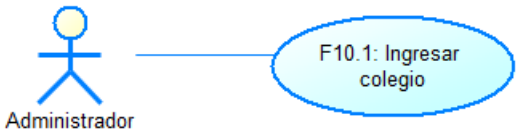
Excepciones:

Código	*Descripción*	*Solución*
E1	Error de conexión con la base de datos	Contactarse con el administrador de base de datos

F10: Administrar colegios



F10.1: Ingresar colegio



Descripción: Este caso de uso describe el flujo que debe seguir un actor para ingresar un nuevo colegio al sistema.

Actores: Administradores.

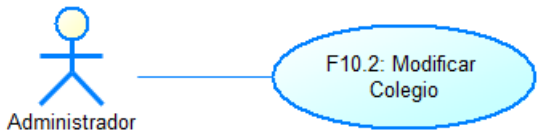
Flujo principal:

1. El actor selecciona la opción de Catálogos > Colegios del menú principal.
2. El sistema carga la pantalla de colegios, con sus respectivos, países, estados y ciudades con la información de la base cargada.
3. El actor presiona el botón Nuevo Colegio.
4. El sistema presenta un modal de ingreso de colegio con la geografía (Países, estados y ciudades).
5. El actor ingresa la información del colegio.
6. El actor presiona el botón Guardar.
7. El sistema guarda los datos del colegio en la base de datos.

Excepciones:

Código	*Descripción*	*Solución*
E1	Error de conexión con la base de datos	Contactarse con el administrador de base de datos

F10.2: Modificar colegio



Descripción: Este caso de uso describe el flujo que debe seguir un actor para modificar una carrera existente.

Actores: Administradores.

Flujo principal:

1. El actor selecciona la opción de Catálogos > Colegios del menú principal.
2. El sistema carga la pantalla de colegios, con sus respectivos, países, estados y ciudades con la información de la base cargada.
3. El actor presiona el botón Modificar del colegio que desee.
4. El sistema carga un modal con los datos del colegio.
5. El actor cambia los datos que el sistema lo permita.
6. El actor presiona el botón Guardar.
7. El sistema modifica los datos del colegio en la base de datos.

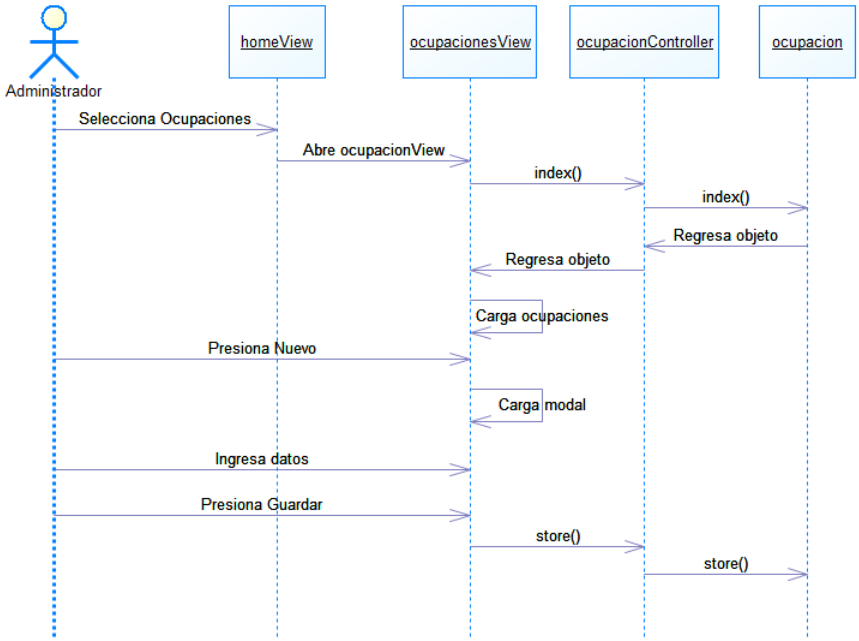
Excepciones:

Código	*Descripción*	*Solución*
E1	Error de conexión con la base de datos	Contactarse con el administrador de base de datos

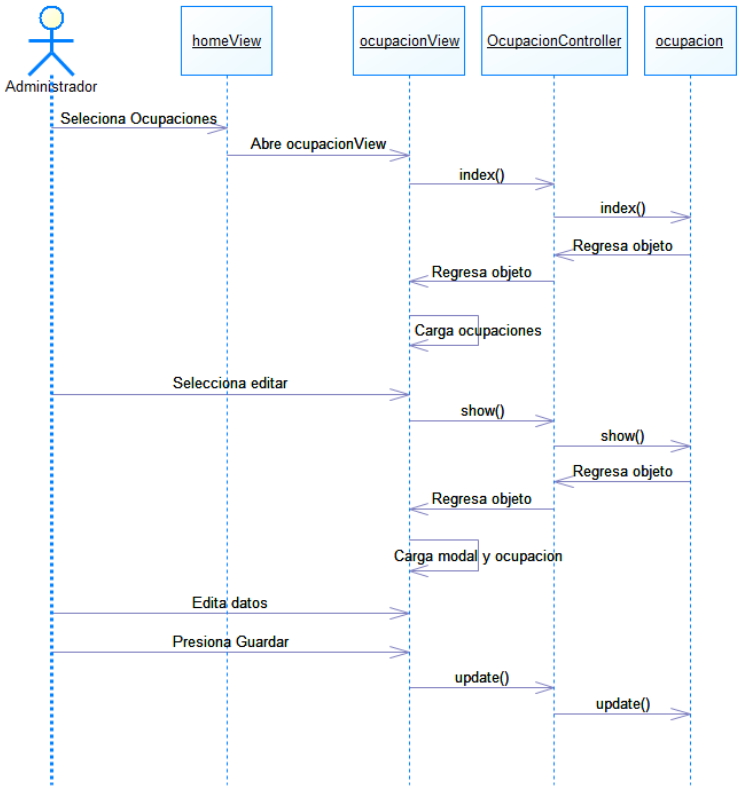
7.2 Anexo 2: Diagramas de secuencia

F1: Administrar Ocupaciones

F1.1 Ingresar ocupación

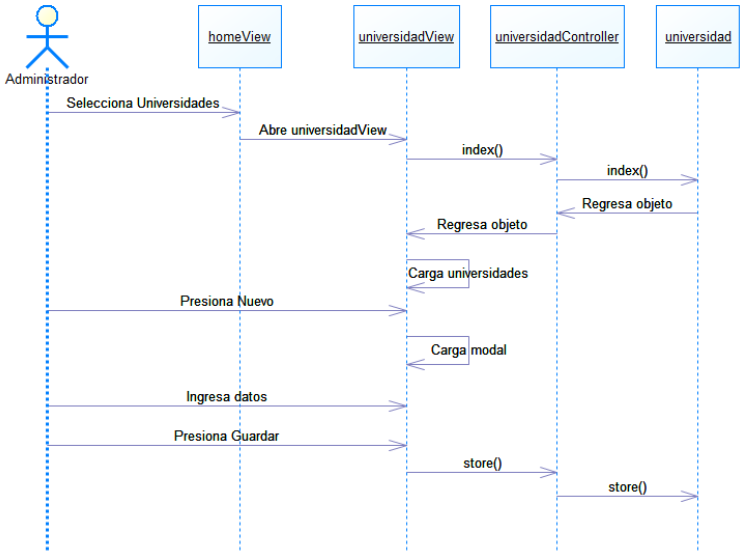


F1.2: Modificar ocupación

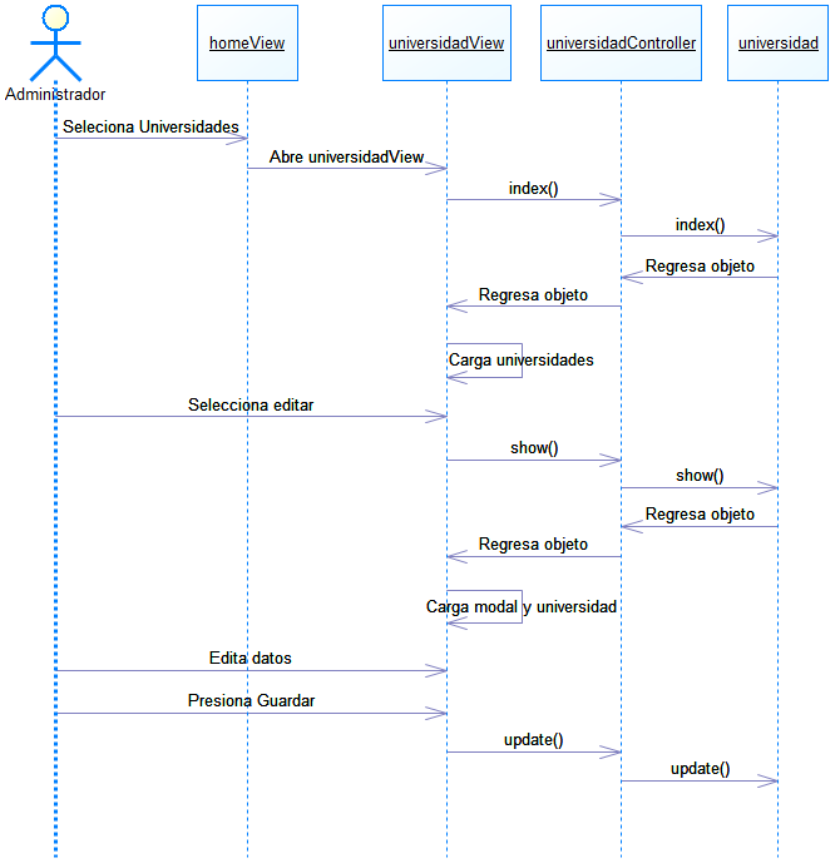


F3: Administrar universidades

F3.1: Ingresar universidad

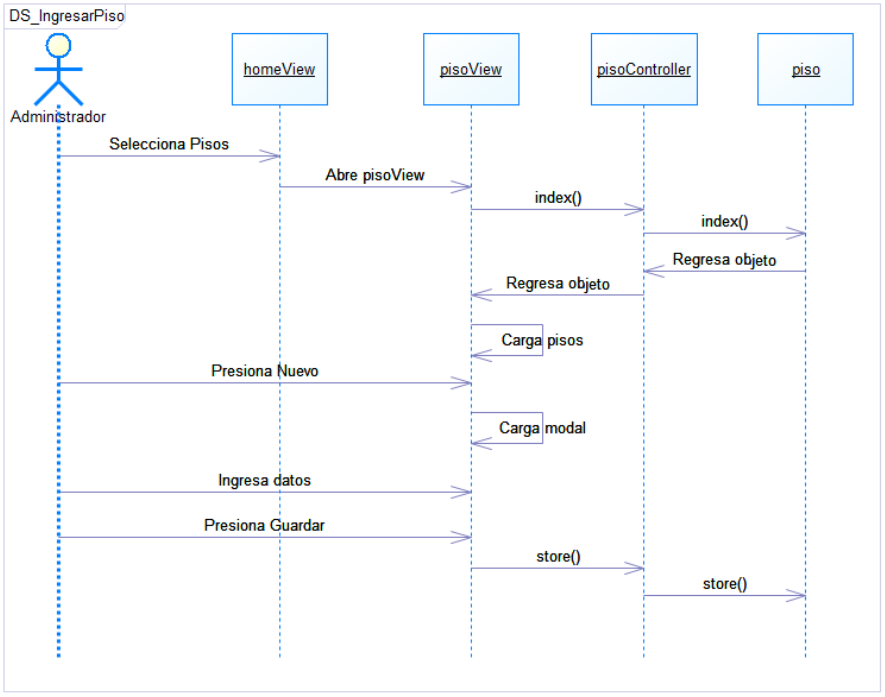


F3.2: Modificar universidad

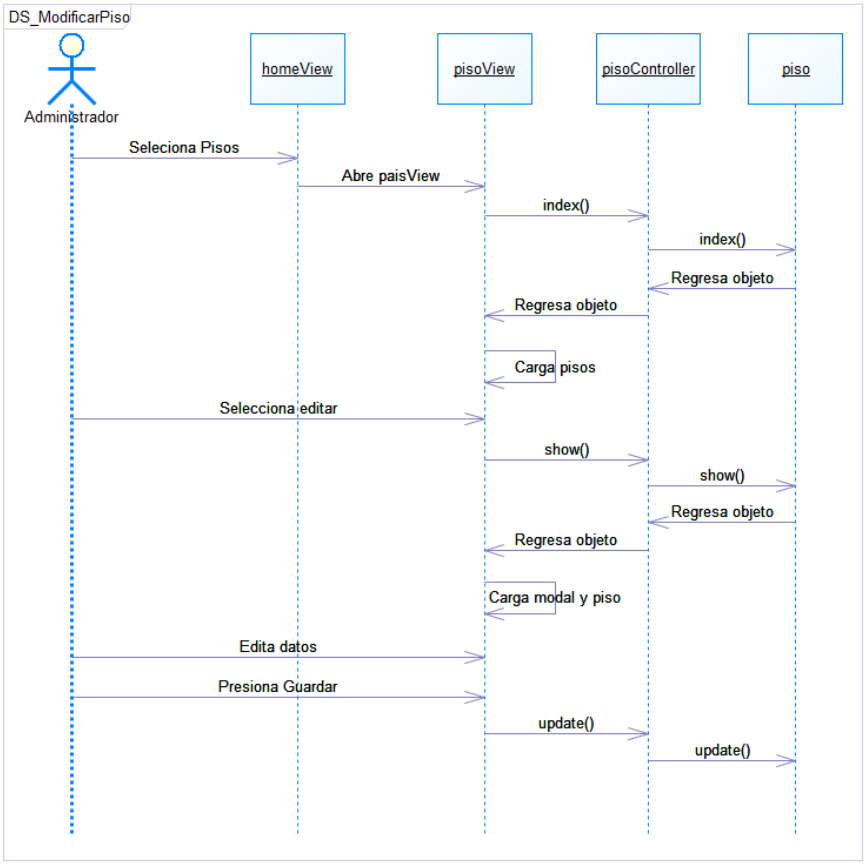


F4: Administrar pisos

F4.1: Ingresar piso

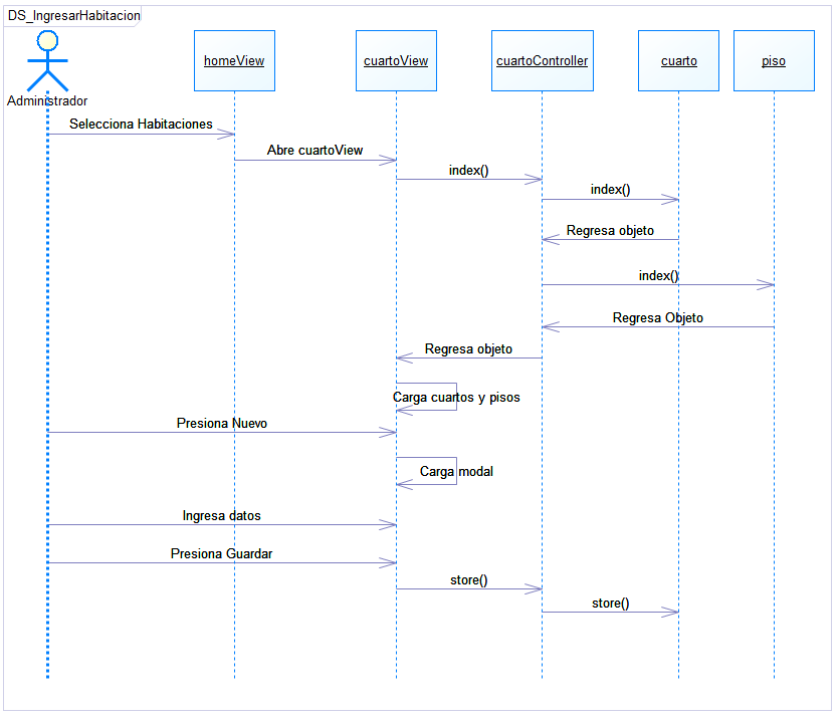


F4.2: Modificar piso

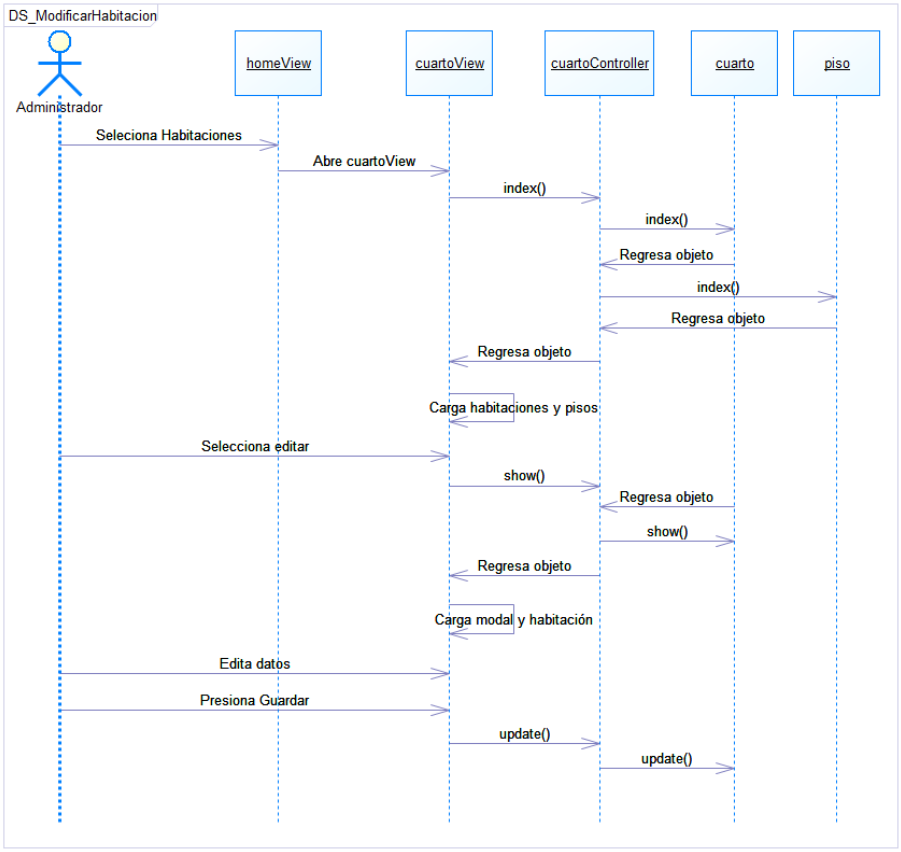


F5: Administrar habitaciones

F5.1: Ingresar habitación

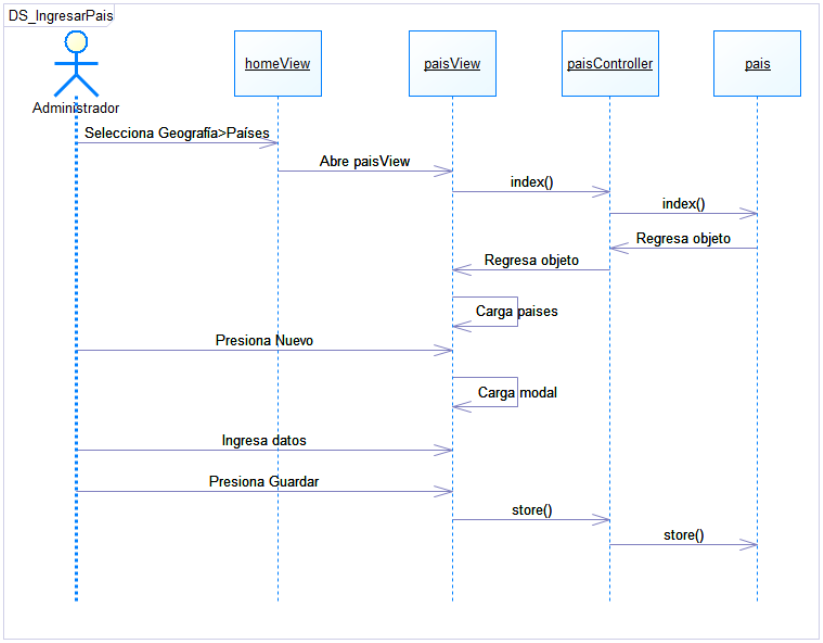


F5.2: Modificar habitación

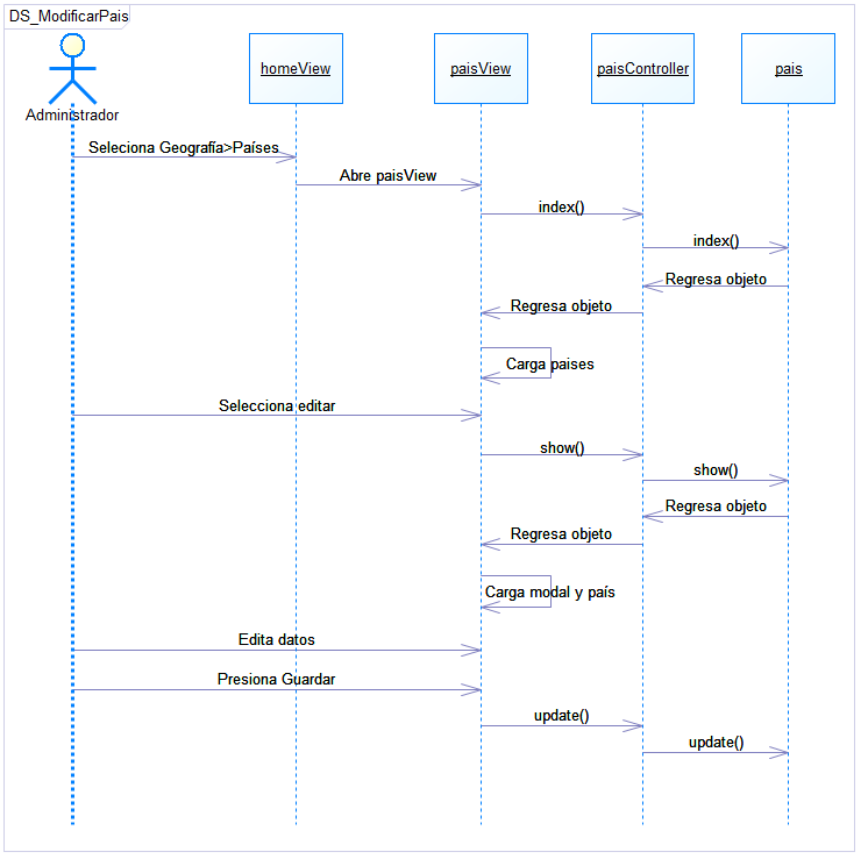


F6: Administrar países

F6.1: Ingresar país

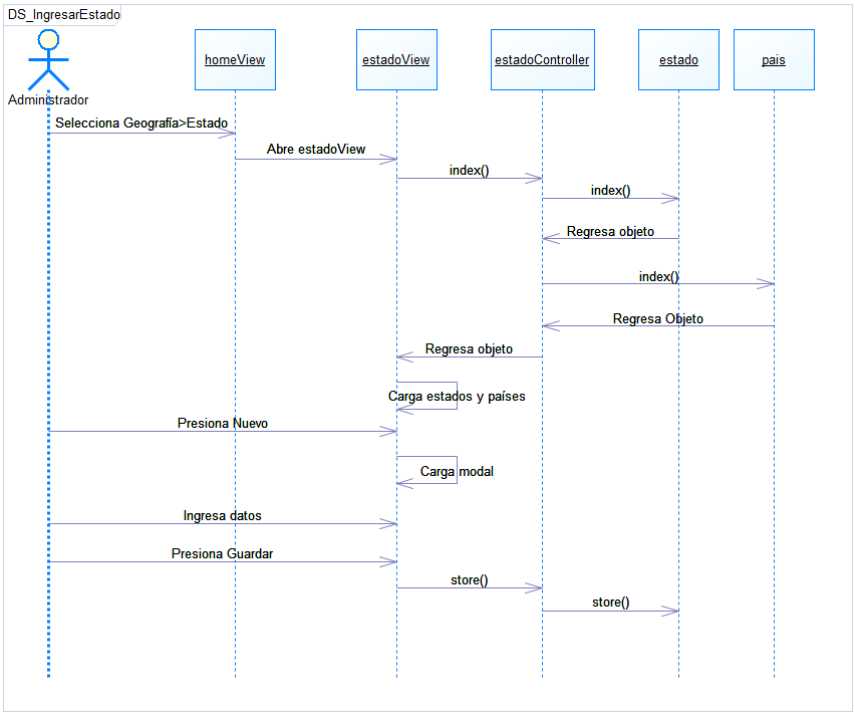


F6.2: Modificar país

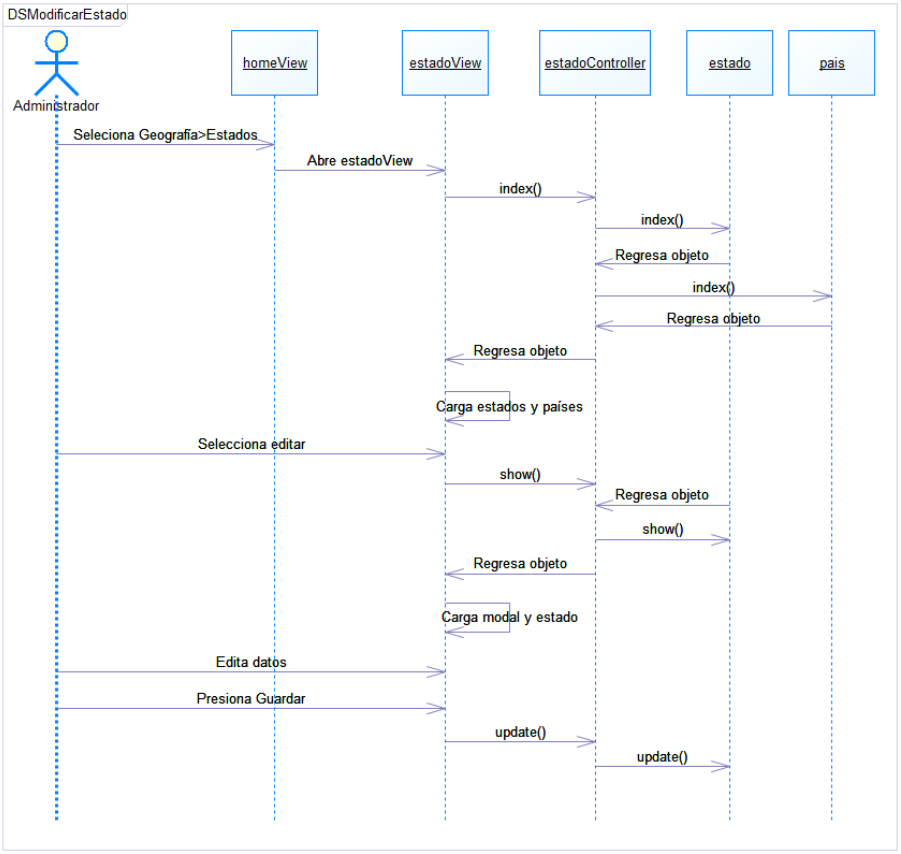


F7: Administrar estados

F7.1: Ingresar estado

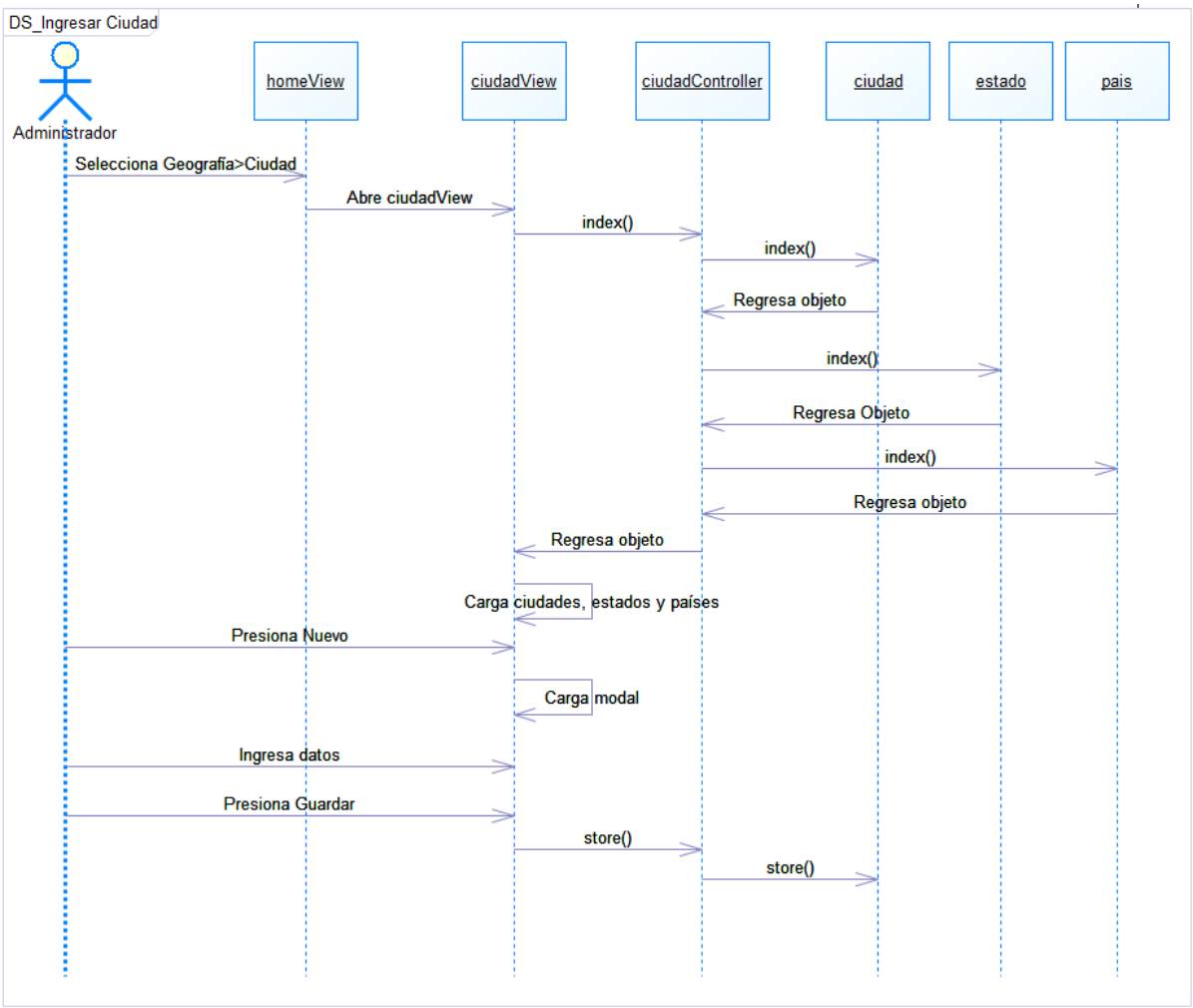


F7.2: Modificar estado

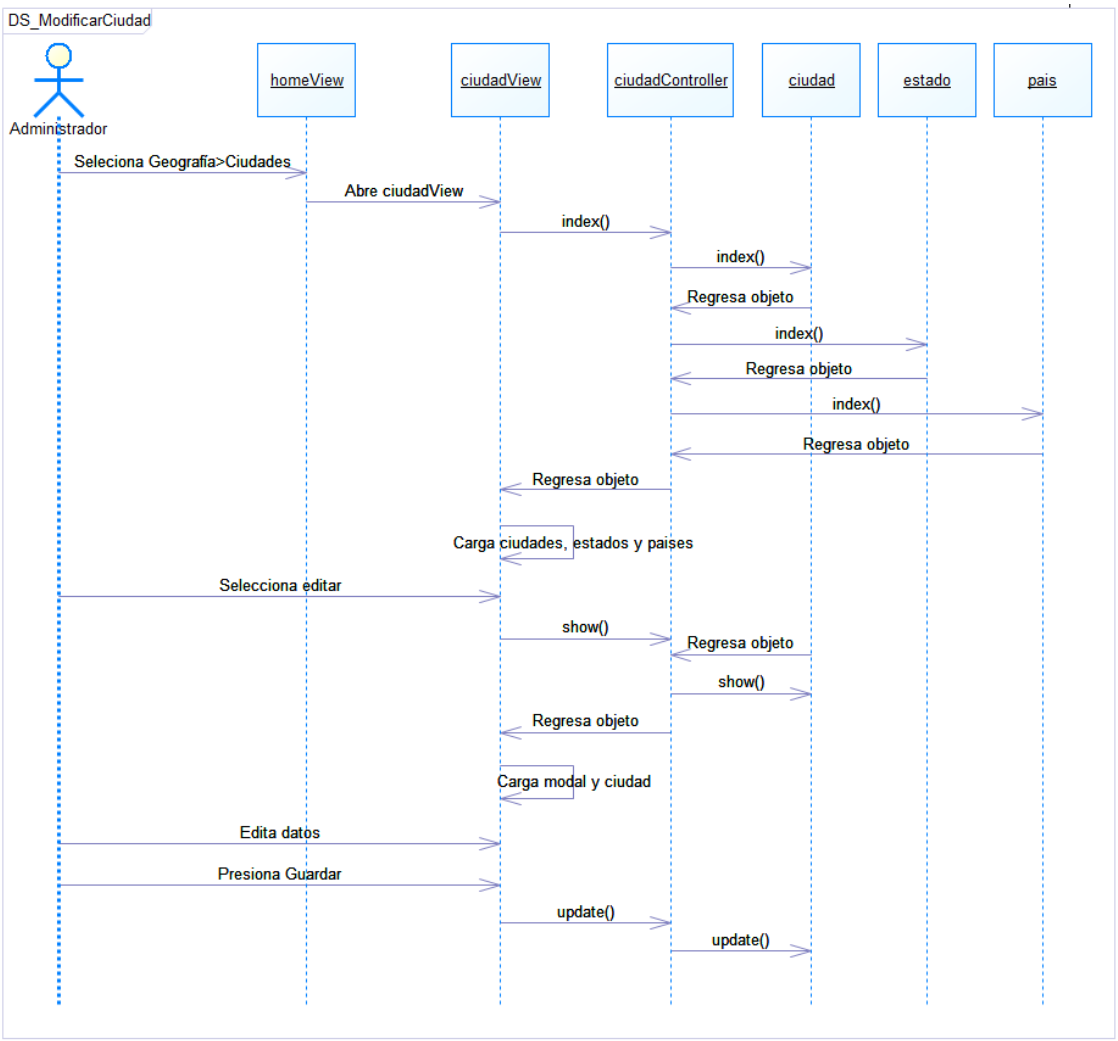


F8: Administrar ciudades

F8.1: Ingresar ciudad

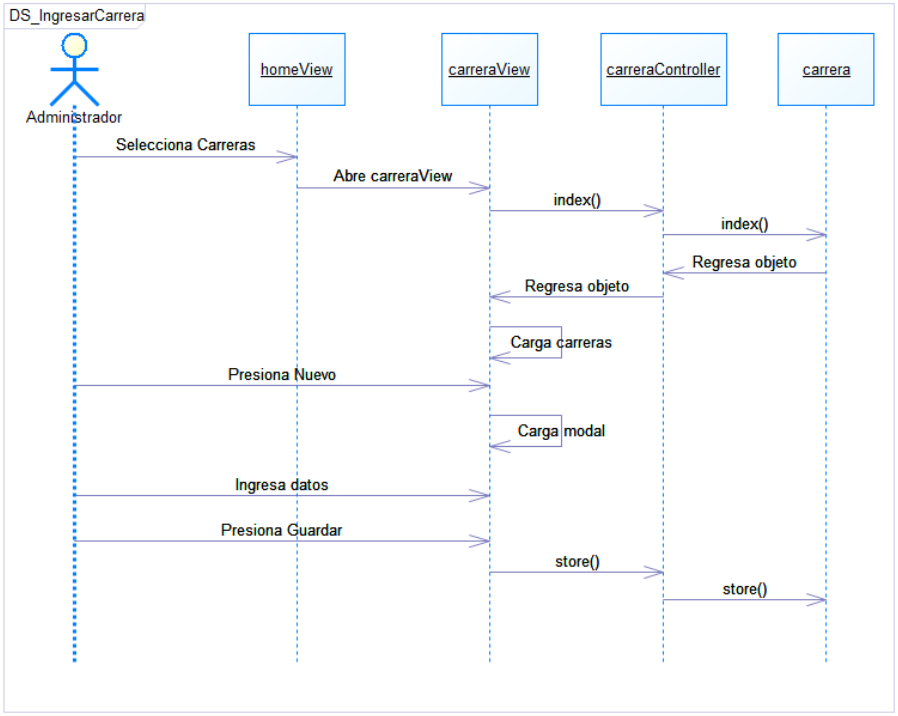


F8.2: Modificar ciudad

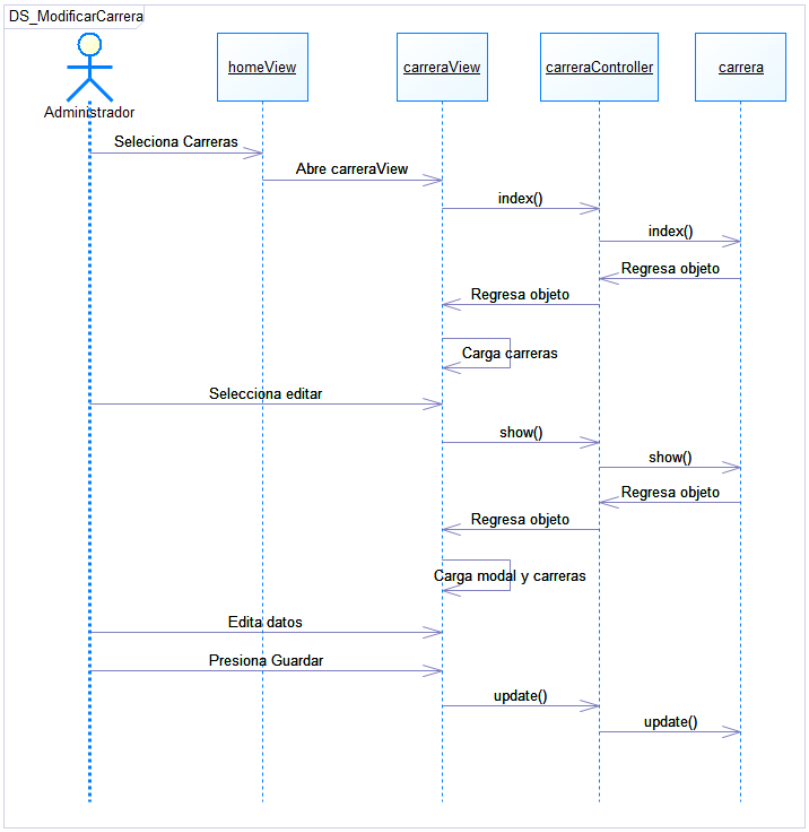

DS_ModificarCiudad
Administrador
homeView
ciudadView
ciudadController
ciudad
estado
pais
Seleciona Geografía>Ciudades
Abre ciudadView
index()
index()
Regresa objeto
index()
Regresa objeto
index()
Regresa objeto
Regresa objeto
Carga ciudades, estados y paises
Selecciona editar
show()
Regresa objeto
show()
Regresa objeto
Carga modal y ciudad
Edita datos
Presiona Guardar
update()
update()

F9: Administrar carrera

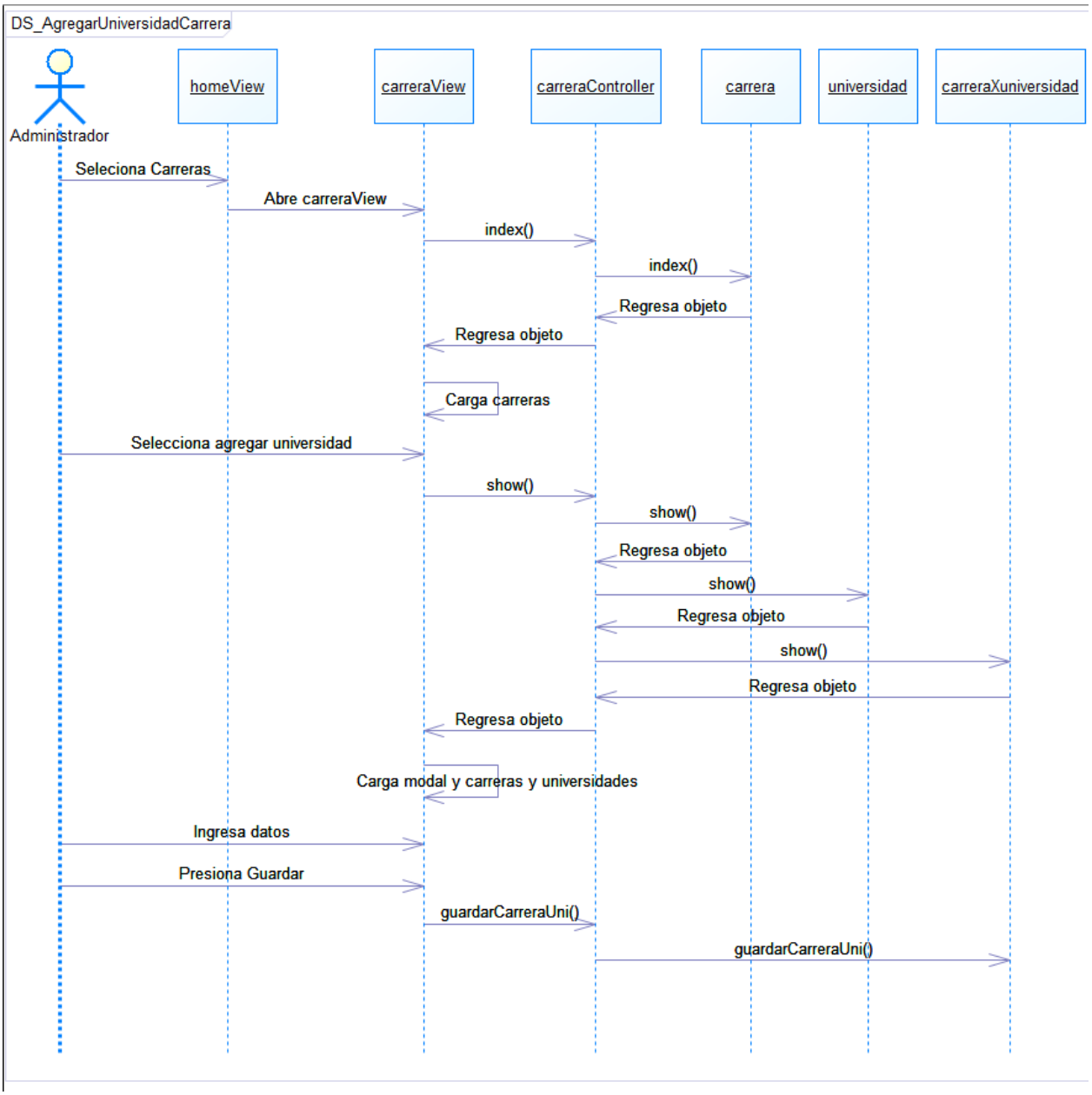
F9.1: Ingresar carrera



F9.2: Modificar carrera

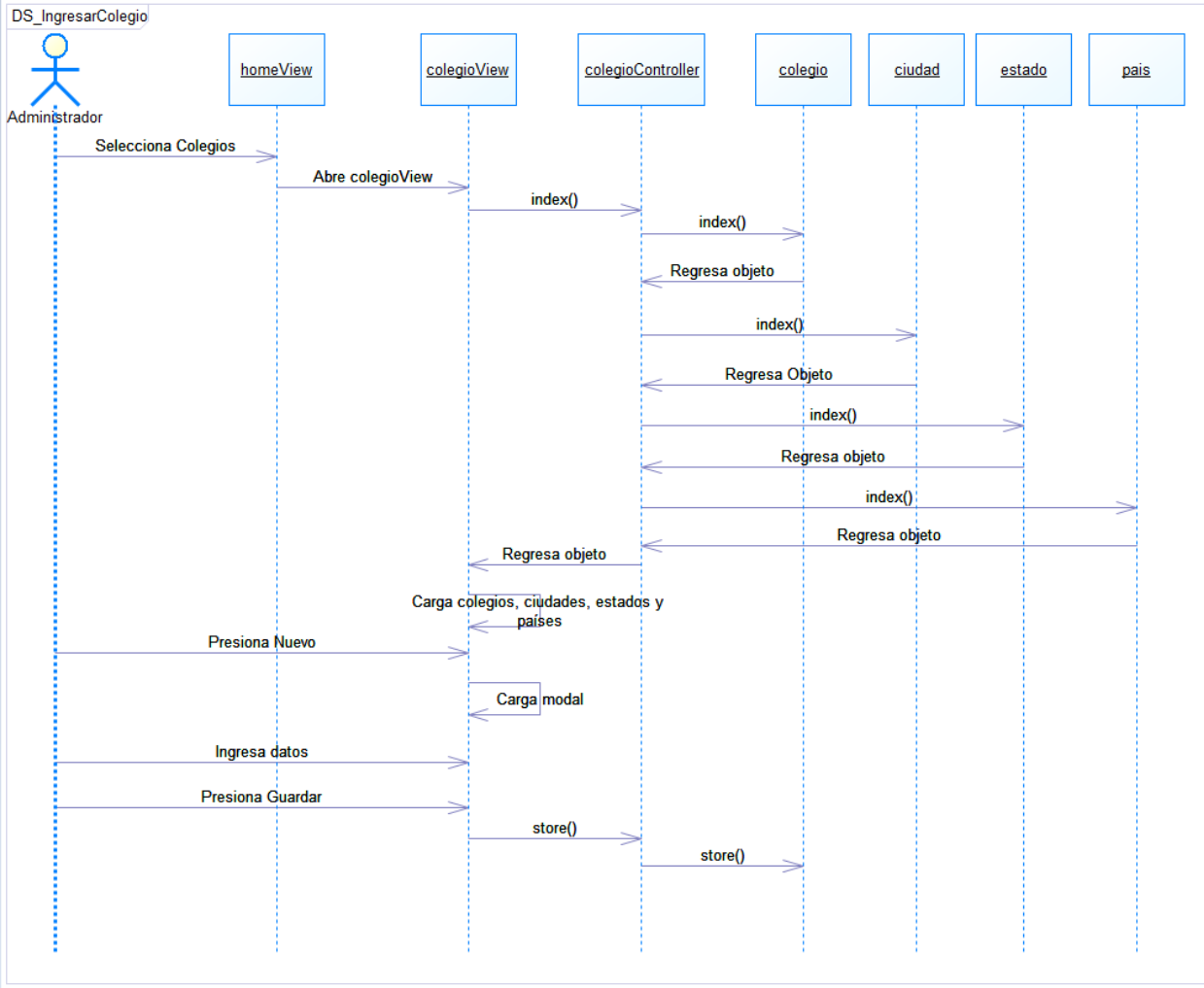


F9.3: Agregar universidad a carrera

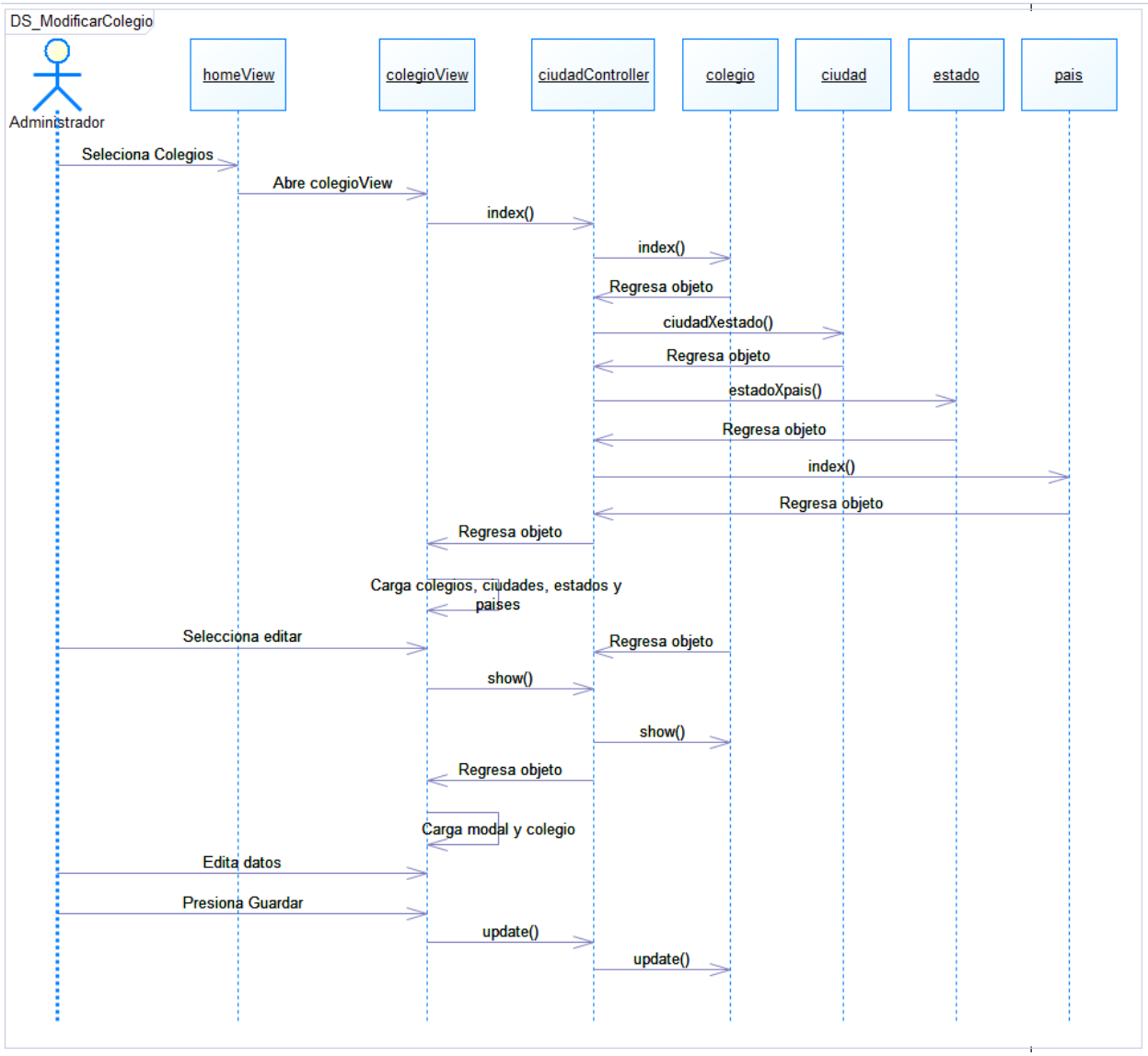


F10: Administrar colegios

F10.1: Ingresar colegio



F10.2: Modificar colegio



7.3 Anexo 3: Código fuente

Modelo

actividad.php

```php
<?php

namespace App;

use Illuminate\Database\Eloquent\Model;

class actividad extends Model
{
    protected $table = 'actividad';
    protected $primary_key = 'ACTIVIDAD_ID';
    public $timestamps = false;

    protected $fillable =
    [
        'ACTIVIDAD_ID',
        'TIPO_ID',
        'ACTIVIDAD_NOMBRE',
        'ACTIVIDAD_DESCRIPCION',
```

```
        'ACTIVIDAD_ACTIVO',
    ];
}
```

carrera.php

```
<?php

namespace App;

use Illuminate\Database\Eloquent\Model;

class carrera extends Model
{
    protected $table = 'carrera';
    protected $primary_key = 'CARRERA_ID';
    public $timestamps = false;

    protected $fillable =
```

```
    [
        'CARRERA_ID',
        'CARRERA_NOMBRE',
        'CARRERA_ACTIVO',
    ];
}
```

carreraXuni.php

```
<?php

namespace App;

use Illuminate\Database\Eloquent\Model;

class carreraXuni extends Model
{
    protected $table = 'carreraxuniversidad';
    public $timestamps = false;
```

```
    protected $fillable =
    [
        'CARRERAXUNI_ID',
        'CARRERA_ID',
        'UNIVERSIDAD_ID',
        'RESIDENTE_ID',
    ];
}
```

ciudad.php

```
<?php

namespace App;

use Illuminate\Database\Eloquent\Model;

class ciudad extends Model
```

```
{
    protected $table = 'ciudad';
    protected $primary_key = 'CIUDAD_ID';
    public $timestamps = false;

    protected $fillable =
    [
        'CIUDAD_ID',
        'ESTADO_ID',
        'CIUDAD_NOMBRE',
        'CIUDAD_ACTIVO',
    ];
}
```

colegio.php

```
<?php

namespace App;
```

```
use Illuminate\Database\Eloquent\Model;


class colegio extends Model
{
    protected $table = 'colegio';
    protected $primary_key = 'COLEGIO_ID';
    public $timestamps = false;


    protected $fillable =
    [
        'COLEGIO_ID',
        'CIUDAD_ID',
        'COLEGIO_NOMBRE',
        'COLEGIO_ACTIVO',
    ];
}
```

cuarto.php

```
<?php

namespace App;

use Illuminate\Database\Eloquent\Model;

class cuarto extends Model
{
    protected $table = 'cuarto';
    protected $primary_key = 'CUARTO_ID';
    public $timestamps = false;

    protected $fillable =
    [
        'CUARTO_ID',
        'RESIDENTE_ID',
```

```
        'PISO_ID',
        'NUMERARIA_ID',
        'CUARTO_NUMERO',
        'CUARTO_ACTIVO',
    ];
}
```

estado.php

```
<?php

namespace App;

use Illuminate\Database\Eloquent\Model;

class estado extends Model
{
    protected $table = 'estado';
    protected $primary_key = 'ESTADO_ID';
```

```
    public $timestamps = false;

    protected $fillable =
    [
        'ESTADO_ID',
        'PAIS_ID',
        'ESTADO_NOMBRE',
        'ESTADO_ACTIVO',
    ];
}
```

horario.php

```
<?php

namespace App;

use Illuminate\Database\Eloquent\Model;
```

```
class horario extends Model
{
    protected $table = 'horario';
    protected $primary_key = 'HORARIO_ID';
    public $timestamps = false;

    protected $fillable =
    [
        'HORARIO_ID',
        'ACTIVIDAD_ID',
        'HORARIO_FECHA_INICIO',
        'HORARIO_FECHA_FIN',
        'HORARIO_HORA',
        'HORARIO_ACTIVO',
    ];
}
```

numeraria.php

```php
<?php

namespace App;

use Illuminate\Database\Eloquent\Model;

class numeraria extends Model
{
    protected $table = 'numeraria';
    protected $primary_key = 'NUMERARIA_ID';
    public $timestamps = false;

    protected $fillable =
    [
        'NUMERARIA_ID',
        'OCUPACION_ID',
        'CARRERAXUNI_ID',
        'CUARTO_ID',
```

```
        'NUMERARIA_CEDULA',
        'NUMERARIA_NOMBRE',
        'NUMERARIA_CELULAR',
        'NUMERARIA_TELEFONO',
        'NUMERARIA_MAIL',
        'NUMERARIA_FECHA_NACIMIENTO',
        'NUMERARIA_ACTIVO',
    ];
}
```

ocupación.php

```
<?php

namespace App;

use Illuminate\Database\Eloquent\Model;

class ocupacion extends Model
```

```
{
    protected $table = 'ocupacion';
    protected $primary_key = 'OCUPACION_ID';
    public $timestamps = false;

    protected $fillable =
    [
        'OCUPACION_ID',
        'OCUPACION_DESCRIPCION',
        'OCUPACION_ACTIVO',
    ];
}
```

padreMadre.php

```
<?php

namespace App;
```

```
use Illuminate\Database\Eloquent\Model;


class padreMadre extends Model
{
    protected $table = 'padremadre';
    protected $primary_key = 'PM_ID';
    public $timestamps = false;


    protected $fillable =
    [
        'PM_ID',
        'RESIDENTE_ID',
        'OCUPACION_ID',
        'PM_NOMBRE',
        'PM_DIRECCION_OFICINA',
        'PM_TELEFONO_OFICINA'
    ];
}
```

pais.php

```
<?php

namespace App;

use Illuminate\Database\Eloquent\Model;

class pais extends Model
{
    protected $table = 'pais';
    protected $primary_key = 'PAIS_ID';
    public $timestamps = false;

    protected $fillable =
    [
        'PAIS_ID',
        'PAIS_NOMBRE',
```

```
        'PAIS_ACTIVO',
    ];
}
```

piso.php

```
<?php

namespace App;

use Illuminate\Database\Eloquent\Model;

class piso extends Model
{
    protected $table = 'piso';
    protected $primary_key = 'PISO_ID';
    public $timestamps = false;

    protected $fillable =
```

```
    [
        'PISO_ID',
        'PISO_NUMERO',
        'PISO_ACTIVO',
    ];
}
```

residente.php

```
<?php

namespace App;

use Illuminate\Database\Eloquent\Model;

class residente extends Model
{
    protected $table = 'residente';
    protected $primary_key = 'RESIDENTE_ID';
```

```
public $timestamps = false;


protected $fillable =
[
    'RESIDENTE_ID',
    'CUARTO_ID',
    'COLEGIO_ID',
    'CIUDAD_ID',
    'CARRERAXUNI_ID',
    'RESIDENTE_CEDULA',
    'RESIDENTE_NOMBRE',
    'RESIDENTE_TELEFONO',
    'RESIDENTE_CELULAR',
    'RESIDENTE_MAIL',
    'RESIDENTE_FECHA_NAC',
    'RESIDENTE_ACTIVO',
    'RESIDENTE_DIRECCION',
    'RESIDENTE_PROMEDIO_GRADUACION',
```

```
        'RESIDENTE_SEMESTRE_A_CURSAR',
        'RESIDENTE_FECHA_INGRESO',
        'RESIDENTE_INGRESO'
    ];
}
```

seg_rol.php

```
<?php

namespace App;

use Illuminate\Database\Eloquent\Model;

class seg_rol extends Model
{
    protected $table = 'rol';
    protected $primaryKey = 'rol_id';
```

```
    public $timestamps = false;


    protected $fillable = [
        'rol_id',
        'rol_descripcion'
    ];
}
```

seg_usuario.php

```
<?php

namespace App;

use Illuminate\Database\Eloquent\Model;

class seg_usuario extends Model
{
    protected $table = 'users';
```

```
    protected $primary_key = 'id';

    public $timestamps = false;



    protected $fillable =

    [

        'id',

        'name',

        'email',

        'password'

    ];



    public function rol()

    {

        return $this->hasOne('App\rol', 'rol_id', 'rol_id');

    }

}
```

tipoActividad.php

```php
<?php

namespace App;

use Illuminate\Database\Eloquent\Model;

class tipoActividad extends Model
{
    protected $table = 'tipoactividad';
    protected $primary_key = 'TIPO_ID';
    public $timestamps = false;

    protected $fillable =
    [
        'TIPO_ID',
        'TIPO_NOMBRE',
        'TIPO_ACTIVO',
    ];
```

```
}
```

universidad.php

```
<?php

namespace App;

use Illuminate\Database\Eloquent\Model;

class universidad extends Model
{
    protected $table = 'universidad';
    protected $primary_key = 'UNIVERSIDAD_ID';
    public $timestamps = false;

    protected $fillable =
    [
        'UNIVERSIDAD_ID',
```

```
        'UNIVERSIDAD_NOMBRE',
        'UNIVERSIDAD_ACTIVO',
    ];
}
```

Controlador

actividadController.php

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

use \App\actividad;

use DB;

class actividadController extends Controller
{
```

```
    public function index()
    {
        $actividades = DB::table('actividad')
            ->join('horario', 'horario.ACTIVIDAD_ID', '=', 'actividad.ACTIVIDAD_ID')
            ->join('tipoactividad', 'tipoactividad.TIPO_ID', '=', 'actividad.TIPO_ID')
            ->select('actividad.*', 'horario.*', 'tipoactividad.TIPO_NOMBRE', 'tipoactividad.TIPO_ID')
            ->get();
        json_encode($actividades);
        $tipoactividades = DB::table('tipoactividad')
        ->where('tipoactividad.TIPO_ACTIVO', '=', '1')
        ->get();
        json_encode($tipoactividades);
        return view('/actividadView')
        ->with([
            'lista' => $actividades,
            'listaTipos' => $tipoactividades
        ]);
```

```
    }


    public function reporteActividades()
    {
        $currentDate = date("Y-m-d");
        $actividadesResidentes = DB::table('cumple')
            ->join('residente', 'cumple.RESIDENTE_ID', 'residente.RESIDENTE_ID')
            ->join('horario', 'horario.HORARIO_ID', '=', 'cumple.HORARIO_ID')
            ->join('actividad', 'actividad.ACTIVIDAD_ID', '=', 'horario.ACTIVIDAD_ID')
            ->join('tipoactividad', 'tipoactividad.TIPO_ID', '=', 'actividad.TIPO_ID')
            -
>select('actividad.*', 'horario.*', 'residente.RESIDENTE_NOMBRE', 'tipoactividad.TIPO
_NOMBRE')
            ->where('horario.HORARIO_FECHA_INICIO', '=', $currentDate)
            ->get();
        json_encode($actividadesResidentes);
        $actividadesNumerarias = DB::table('cumple3')
            -
>join('numeraria', 'cumple3.NUMERARIA_ID', 'numeraria.NUMERARIA_ID')
```

```
            ->join('horario', 'horario.HORARIO_ID', '=', 'cumple3.HORARIO_ID')
            ->join('actividad', 'actividad.ACTIVIDAD_ID', '=', 'horario.ACTIVIDAD_ID')
            ->join('tipoactividad', 'tipoactividad.TIPO_ID', '=', 'actividad.TIPO_ID')
            -
>select('actividad.*', 'horario.*', 'numeraria.NUMERARIA_NOMBRE', 'tipoactividad.TI
PO_NOMBRE')
            ->where('horario.HORARIO_FECHA_INICIO', '=', $currentDate)
            ->get();
        json_encode($actividadesNumerarias);
        return view('/reporteActividadesView')
        ->with([
            'lista' => $actividadesResidentes,
            'listaNum' => $actividadesNumerarias
        ]);
    }

    public function registrarActividadResidente(Request $request){
        $user = $request->session()->get('user');
        $residente = DB::table('residente')
```

```
->where('id', '=', $user)
->select('RESIDENTE_ID')
->get()
->toArray();
json_encode($residente);
$residente_id = (int)$residente[0]->RESIDENTE_ID;
$currentDate = date("Y-m-d");
$actividadesRegistradas = DB::table('cumple')
    ->join('horario', 'horario.HORARIO_ID', '=', 'cumple.HORARIO_ID')
    ->join('actividad', 'actividad.ACTIVIDAD_ID', '=', 'horario.ACTIVIDAD_ID')
    ->join('tipoactividad', 'tipoactividad.TIPO_ID', '=', 'actividad.TIPO_ID')
    ->select('actividad.*', 'horario.*', 'tipoactividad.TIPO_NOMBRE', 'cumple.*')
    ->where('cumple.RESIDENTE_ID', '=', $residente_id)
    ->where('horario.HORARIO_FECHA_INICIO', '>=', $currentDate)
    ->get();
    json_encode($actividadesRegistradas);
$actividadesDisponibles = DB::table('horario')
    ->join('actividad', 'actividad.ACTIVIDAD_ID', '=', 'horario.ACTIVIDAD_ID')
```

```
        ->join('tipoactividad', 'tipoactividad.TIPO_ID', '=', 'actividad.TIPO_ID')
        ->select('actividad.*', 'horario.*', 'tipoactividad.TIPO_NOMBRE')
        ->where('horario.HORARIO_FECHA_INICIO', '>=', $currentDate)
        ->get();
        json_encode($actividadesDisponibles);
    return view('/registrarActividadResidente')
    ->with([
        'lista' => $actividadesRegistradas,
        'listaActividades' => $actividadesDisponibles,
        'residenteID' => $residente_id
    ]);
    return back();
}

public function registrarActividadNumeraria(Request $request){
    $user = $request->session()->get('user');
    $numeraria = DB::table('numeraria')
    ->where('id', '=', $user)
```

```
->select('NUMERARIA_ID')

->get()

->toArray();

json_encode($numeraria);

$numeraria_id = (int)$numeraria[0]->NUMERARIA_ID;

$currentDate = date("Y-m-d");

$actividadesRegistradas = DB::table('cumple3')

    ->join('horario', 'horario.HORARIO_ID', '=', 'cumple3.HORARIO_ID')

    ->join('actividad', 'actividad.ACTIVIDAD_ID', '=', 'horario.ACTIVIDAD_ID')

    ->join('tipoactividad', 'tipoactividad.TIPO_ID', '=', 'actividad.TIPO_ID')

    ->select('actividad.*', 'horario.*', 'tipoactividad.TIPO_NOMBRE', 'cumple3.*')

    ->where('cumple3.NUMERARIA_ID', '=', $numeraria_id)

    ->where('horario.HORARIO_FECHA_INICIO', '>=', $currentDate)

    ->get();

    json_encode($actividadesRegistradas);

$actividadesDisponibles = DB::table('horario')

    ->join('actividad', 'actividad.ACTIVIDAD_ID', '=', 'horario.ACTIVIDAD_ID')

    ->join('tipoactividad', 'tipoactividad.TIPO_ID', '=', 'actividad.TIPO_ID')
```

```
        ->select('actividad.*', 'horario.*', 'tipoactividad.TIPO_NOMBRE')
        ->where('horario.HORARIO_FECHA_INICIO', '>=', $currentDate)
        ->get();
        json_encode($actividadesDisponibles);
    return view('/registrarActividadNumeraria')
    ->with([
        'lista' => $actividadesRegistradas,
        'listaActividades' => $actividadesDisponibles,
        'numerariaID' => $numeraria_id
    ]);
    return back();
}

public function store(Request $request)
{
    $actividadActivo = 1;
    $data = array('TIPO_ID'=>$request->input('TIPO_ID'),
    'ACTIVIDAD_NOMBRE'=>$request->input('ACTIVIDAD_NOMBRE'),
```

```
            'ACTIVIDAD_DESCRIPCION'=>$request-
>input('ACTIVIDAD_DESCRIPCION'),
            'ACTIVIDAD_ACTIVO'=>$actividadActivo);
            DB::table('actividad')->insert($data);
            //ingresar el horario de la actividad
            $horarioActivo = 1;
            $actividad = DB::table('actividad')
               ->orderBy('ACTIVIDAD_ID','desc')
               ->select('ACTIVIDAD_ID')
               ->take(1)
               ->get()
               ->toArray();
               json_encode($actividad);
               $id=(int)$actividad[0]->ACTIVIDAD_ID;
            $data2 = array('ACTIVIDAD_ID'=>$id,
            'HORARIO_FECHA_INICIO'=>$request-
>input('HORARIO_FECHA_INICIO'),
            'HORARIO_FECHA_FIN'=>$request->input('HORARIO_FECHA_FIN'),
            'HORARIO_HORA'=>$request->input('HORARIO_HORA'),
```

```
    'HORARIO_ACTIVO'=>$horarioActivo);
    DB::table('horario')->insert($data2);
    return back();
}

public function show($actividad_id)
{
    $actividad = actividad::where('ACTIVIDAD_ID', '=', $actividad_id)->get();
    json_encode($actividad);
    return $actividad;
}

public function update($id, Request $request)
{
    if(!$request->ACTIVIDAD_ACTIVO){ //cuando el checkbox esta en off
        $actividadactivo=0;
        $horarioactivo=0;
    }
```

```
        else{
            $actividadactivo=1;
            $horarioactivo=1;
        }
        $data = array('TIPO_ID'=>$request->input('TIPO_ID'),
        'ACTIVIDAD_NOMBRE'=>$request->input('ACTIVIDAD_NOMBRE'),
        'ACTIVIDAD_DESCRIPCION'=>$request->input('ACTIVIDAD_DESCRIPCION'),
        'ACTIVIDAD_ACTIVO'=>$actividadactivo);
        DB::table('actividad')->where('ACTIVIDAD_ID','=', $request->input('ACTIVIDAD_ID'))->update($data);
        //modificar el horario
        $data2 = array('HORARIO_FECHA_INICIO'=>$request->input('HORARIO_FECHA_INICIO'),
        'HORARIO_FECHA_FIN'=>$request->input('HORARIO_FECHA_FIN'),
        'HORARIO_HORA'=>$request->input('HORARIO_HORA'),
        'HORARIO_ACTIVO'=>$horarioactivo);
        DB::table('horario')->where('HORARIO_ID','=', $request->input('HORARIO_ID'))->update($data2);
```

```
    return back();
}

public function registerResident($horario_id, $residente_id){
    $data = array('HORARIO_ID'=>$horario_id,
    'RESIDENTE_ID'=>$residente_id);
    DB::table('cumple')->insert($data);
    return back();
}

public function registerNumeraria($horario_id, $numeraria_id){
    $data = array('HORARIO_ID'=>$horario_id,
    'NUMERARIA_ID'=>$numeraria_id);
    $insert = DB::table('cumple3')->insert($data);
    if($insert == false){
        return back()->with('alert', 'No se puede escoger la misma actividad');
    }
    else{
```

```
        return back();
    }
}

public function unregisterResident($horario_id, $residente_id){
    DB::table('cumple')
    ->where('RESIDENTE_ID', '=', $residente_id)
    ->where('HORARIO_ID', '=', $horario_id)
    ->delete();
    return back();
}

public function unregisterNumeraria($horario_id, $numeraria_id){
    DB::table('cumple3')
    ->where('NUMERARIA_ID', '=', $numeraria_id)
    ->where('HORARIO_ID', '=', $horario_id)
    ->delete();
    return back();
```

```
    }

    public function destroy($id)
    {
        ocupacion::findOrFail($id)->delete();
        return 204;
    }
}
```

carreraController.php

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

use \App\carrera;
```

```
use \App\universidad;

use DB;

class carreraController extends Controller
{
    public function index()
    {
        $carrera=carrera::get();
        $carreraXuni = DB::table('carreraxuniversidad')
        ->join('universidad', 'carreraXuniversidad.UNIVERSIDAD_ID' ,'=', 'universidad.UNIVERSIDAD_ID')
        ->select('carreraxuniversidad.CARRERA_ID', 'carreraxuniversidad.UNIVERSIDAD_ID','universidad.UNIVERSIDAD_NOMBRE')
        ->get();
        json_encode($carreraXuni);
        $universidades = DB::table('universidad')
```

```
            ->select('UNIVERSIDAD_ID','UNIVERSIDAD_NOMBRE')
            ->get();
            json_encode($universidades);
            return view('/carreraXuniView')
                        ->with([
                            'lista' => $carrera,
                            'listaCarreraXuni' => $carreraXuni,
                            'listaUniversidades' => $universidades
                           ]);
        }
        public function carreraxUniversidad(Request $request, $universidad_id){
            if($request->ajax())
            {
                $lista = DB::table('carreraxuniversidad')
                -
>join('carrera', 'carrera.CARRERA_ID', '=', 'carreraxuniversidad.CARRERA_ID')
                ->select('carreraxuniversidad.CARRERA_ID','carrera.CARRERA_NOMBRE')
                ->where('carreraxuniversidad.UNIVERSIDAD_ID', '=', $universidad_id)
                ->orderBy('UNIVERSIDAD_ID','asc')
```

```
            ->get();
            return $lista;
        }
    }
    public function store(Request $request)
    {
        $carreraActivo=1;
        $data = array('CARRERA_NOMBRE'=>$request-
>input('CARRERA_NOMBRE'),'CARRERA_ACTIVA'=>$carreraActivo);
        $resultado=DB::table('carrera')->insert($data);
        return back();
    }
    public function guardadCarreraUni($id, Request $request)
    {
        $carreraXuni = DB::table('carreraxuniversidad')
        ->where('CARRERA_ID', '=', $request->CARRERA_ID)
        ->where('UNIVERSIDAD_ID', '=', $request->UNIVERSIDAD_ID)
        ->get();
        if(sizeof($carreraXuni) != 0){
```

```
            return back()->with('alert', 'Registro ya ingresado');
        }
        else{
        $data = array('CARRERA_ID'=>$id,'UNIVERSIDAD_ID'=>$request->input('UNIVERSIDAD_ID'));

            DB::table('carreraxuniversidad')->insert($data);
            return back()->with('alert', 'Creado con éxito');
        }
    }
    public function show($carrera_id)
    {
        $carrera = carrera::where('CARRERA_ID', '=', $carrera_id)->get();
        json_encode($carrera);
        return $carrera;
    }

    public function update($id, Request $request)
    {
```

```
            if(!$request->CARRERA_ACTIVA) //cuando el checkbox esta en off
            {
               $carreraActivo=0;
            }
            else
            {
               $carreraActivo=1;
            }
            $data = array('CARRERA_NOMBRE'=>$request-
>input('CARRERA_NOMBRE'),'CARRERA_ACTIVA'=>$carreraActivo);
            DB::table('carrera')->where('CARRERA_ID','=', $id)->update($data);
            return back();
        }

        public function destroy($id)
        {
            ocupacion::findOrFail($id)->delete();
            return 204;
        }
```

```
}
```

carreraXuniController.php

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

use \App\carreraXuni;

use DB;

class carreraXuniController extends Controller
{
    public function store(Request $request)
    {
```

```
            $data = array('CARRERA_ID'=>$request-
>input('CARRERA_ID'),'UNIVERSIDAD_ID'=>$request-
>input('UNIVERSIDAD_ID'));
            $resultado = DB::table('carreraxuniversidad')->insert($data);
            return back();
        }

        public function show($carrera_id)
        {
            $carrera = carrera::where('CARRERA_ID', '=', $carrera_id)->get();
            json_encode($carrera);
            return $carrera;
        }

        public function update($id, Request $request)
        {
            if(!$request->CARERRA_ACTIVO) //cuando el checkbox esta en off
            {
                $carreraActivo=0;
```

```
            }
            else
            {
               $carreraActivo=1;
            }
            $data = array('CARRERA_NOMBRE'=>$request-
>input('CARRERA_NOMBRE'),'CARRERA_ACTIVA'=>$carreraActivo);
            DB::table('carrera')->where('CARRERA_ID','=', $id)->update($data);
            return back();
        }

        public function destroy($id)
        {
            ocupacion::findOrFail($id)->delete();
            return 204;
        }
    }
```

ciudadController.php

```php
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

use \App\ciudad;

use \App\pais;

use DB;

class ciudadController extends Controller
{
    public function index()
    {
        $ciudad = DB::table('ciudad')
        ->join('estado', 'estado.ESTADO_ID', '=', 'ciudad.ESTADO_ID')
```

```
    ->join('pais', 'estado.PAIS_ID', '=', 'pais.PAIS_ID')
    ->select('ciudad.*','estado.ESTADO_NOMBRE', 'PAIS_NOMBRE')
    ->orderBy('ESTADO_ID','asc')
    ->get();
    json_encode($ciudad);
    $pais = pais::get();
    json_encode($pais);
    return view('/ciudadView')
                ->with([
                    'lista' => $ciudad,
                    'listaPaises' => $pais
                   ]);
}

public function getCiudadXestado(Request $request, $estado_id){
    if($request->ajax())
    {
        $lista = DB::table('ciudad')
```

```
            ->join('estado', 'estado.ESTADO_ID', '=', 'ciudad.ESTADO_ID')
            ->join('pais', 'estado.PAIS_ID', '=', 'pais.PAIS_ID')
            ->select('ciudad.*','estado.ESTADO_NOMBRE', 'PAIS_NOMBRE')
            ->where('estado.ESTADO_ID', '=', $estado_id)
            ->orderBy('ESTADO_ID','asc')
            ->get();
            return $lista;
        }
    }
    public function store(Request $request)
    {
        $ciudadActivo=1;
        $data = array('CIUDAD_NOMBRE'=>$request-
>input('CIUDAD_NOMBRE'),'ESTADO_ID'=>$request-
>input('ESTADO_ID'),'CIUDAD_ACTIVO'=>$ciudadActivo);
        DB::table('ciudad')->insert($data);
        return back();
    }
```

```
    public function show($piso_id)
    {
        $estado = estado::where('ESTADO_ID', '=', $estado_id)->get();
        json_encode($estado);
        return $estado;
    }

    public function update($id, Request $request)
    {
        if(!$request->CIUDAD_ACTIVO) //cuando el checkbox esta en off
        {
            $ciudadActivo=0;
        }
        else
        {
            $ciudadActivo=1;
        }
        $data = array('CIUDAD_NOMBRE'=>$request-
>input('CIUDAD_NOMBRE'),'CIUDAD_ACTIVO'=>$ciudadActivo);
```

```
            DB::table('ciudad')->where('CIUDAD_ID','=', $request->CIUDAD_ID)-
>update($data);
            return back();
        }

        public function destroy($id)
        {
            ocupacion::findOrFail($id)->delete();
            return 204;
        }
    }
```

colegioController.php

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
```

```
use \App\colegio;

use \App\ciudad;

use \App\pais;

use DB;

class colegioController extends Controller
{
   public function index()
   {
      $colegioXciudad = DB::table('colegio')
      ->join('ciudad', 'colegio.CIUDAD_ID', '=', 'ciudad.CIUDAD_ID')
      ->join('estado', 'ciudad.ESTADO_ID', '=', 'estado.ESTADO_ID')
      ->join('pais', 'estado.PAIS_ID', '=', 'pais.PAIS_ID')
      -
>select('colegio.*','ciudad.CIUDAD_NOMBRE', 'estado.ESTADO_NOMBRE', 'pais.PAI
S_NOMBRE', 'pais.PAIS_ID')
```

```
    ->orderBy('PAIS_ID','asc')
    ->get();
    json_encode($colegioXciudad);
    $paises = pais::get();
    json_encode($paises);
    $ciudad = ciudad::get();
    json_encode($ciudad);
    return view('/colegioView')
                ->with([
                    'lista' => $colegioXciudad,
                    'listaPaises' => $paises,
                    'listaCiudades' => $ciudad
                   ]);
}
public function getColegioXciudad(Request $request, $ciudad_id){
    if($request->ajax())
    {
        $lista = DB::table('colegio')
```

```
            ->join('ciudad', 'colegio.CIUDAD_ID', '=', 'ciudad.CIUDAD_ID')
            ->select('colegio.*')
            ->where('colegio.CIUDAD_ID', '=', $ciudad_id)
            ->where('colegio.COLEGIO_ACTIVO', '=', '1')
            ->orderBy('CIUDAD_ID','asc')
            ->get();
            return $lista;
        }
    }
    public function store(Request $request)
    {
        $colegioActivo=1;
        $data = array('COLEGIO_NOMBRE'=>$request-
>input('COLEGIO_NOMBRE'),'CIUDAD_ID'=>$request-
>input('CIUDAD_ID'),'COLEGIO_ACTIVO'=>$colegioActivo);
        DB::table('colegio')->insert($data);
        return back();
    }
```

```
    public function show($colegioo_id)
    {
        $colegio = colegio::where('COLEGIO_ID', '=', $colegio_id)->get();
        json_encode($colegio);
        return $colegio;
    }

    public function update($id, Request $request)
    {
        if(!$request->COLEGIO_ACTIVO) //cuando el checkbox esta en off
        {
            $colegioActivo=0;
        }
        else
        {
            $colegioActivo=1;
        }
        $data = array('COLEGIO_NOMBRE'=>$request-
>input('COLEGIO_NOMBRE'),'COLEGIO_ACTIVO'=>$colegioActivo);
```

```
            DB::table('colegio')->where('COLEGIO_ID','=', $request->COLEGIO_ID)-
>update($data);
            return back();
        }

        public function destroy($id)
        {
            ocupacion::findOrFail($id)->delete();
            return 204;
        }
    }
```

cuartoController.php

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
```

```
use \App\cuarto;

use DB;

class cuartoController extends Controller
{
    public function index()
    {
        $cuarto = DB::table('cuarto')
        ->join('piso', 'piso.PISO_ID', '=', 'cuarto.PISO_ID')
        ->select('piso.PISO_NUMERO','piso.PISO_ID','cuarto.CUARTO_ID','cuarto.CUARTO_NUMERO','cuarto.CUARTO_ACTIVO')
        ->orderBy('PISO_ID','asc')
        ->get();
        json_encode($cuarto);
        $pisos = DB::table('piso')
        ->select('PISO_NUMERO','PISO_ID')
        ->where('PISO_ACTIVO','=','1')
```

```
    ->get();
    json_encode($pisos);
    return view('/cuartoView')
                ->with([
                    'lista' => $cuarto,
                    'listaPisos' => $pisos
                   ]);
}

public function getCuartoXpiso(Request $request, $piso_id){
    if($request->ajax())
    {
        $lista = DB::table('cuarto')
        ->join('piso', 'piso.PISO_ID', '=', 'cuarto.PISO_ID')
        ->select('cuarto.*')
        ->where('cuarto.CUARTO_ACTIVO', '=', '1')
        ->where('cuarto.CUARTO_OCUPADO', '=', '0')
        ->where('cuarto.PISO_ID', '=', $piso_id)
```

```
            ->orderBy('PISO_ID','asc')
            ->get();
            return $lista;
        }
    }

    public function store(Request $request)
    {
        $cuartoActivo=1;
        $cuartoOcupado=0;
        $data = array('CUARTO_NUMERO'=>$request-
>input('CUARTO_NUMERO'),'PISO_ID'=>$request-
>input('PISO_ID'),'CUARTO_ACTIVO'=>$cuartoActivo,'CUARTO_OCUPADO'=>$cu
artoOcupado);
        DB::table('cuarto')->insert($data);
        return back();
    }

    public function show($cuarto_id)
```

```
        {
            $cuarto = cuarto::where('CUARTO_ID', '=', $cuarto_id)->get();
            json_encode($cuarto);
            return $cuarto;
        }

        public function update($id, Request $request)
        {
            if(!$request->CUARTO_ACTIVO) //cuando el checkbox esta en off
            {
                $cuartoActivo=0;
            }
            else
            {
                $cuartoActivo=1;
            }
            $data = array('CUARTO_NUMERO'=>$request-
>input('CUARTO_NUMERO'),'CUARTO_ACTIVO'=>$cuartoActivo);
```

```
            DB::table('cuarto')->where('CUARTO_ID','=', $request->CUARTO_ID)-
>update($data);
            return back();
        }

        public function destroy($id)
        {
            ocupacion::findOrFail($id)->delete();
            return 204;
        }
    }
```

estadoController.php

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
```

```
use \App\estado;


use DB;


class estadoController extends Controller
{
   public function index()
   {
      $estado = DB::table('estado')
      ->join('pais', 'pais.PAIS_ID', '=', 'estado.PAIS_ID')
      ->select('estado.*','pais.PAIS_NOMBRE')
      ->orderBy('PAIS_ID','asc')
      ->get();
      json_encode($estado);
      $pais = DB::table('pais')
      ->select('PAIS_NOMBRE','PAIS_ID')
      ->where('PAIS_ACTIVO','=','1')
      ->get();
```

```
    json_encode($pais);
    return view('/estadoView')
                ->with([
                    'lista' => $estado,
                    'listaPaises' => $pais
                   ]);
}

public function getEstadoXpais(Request $request, $pais_id){
    if($request->ajax())
    {
        $lista = DB::table('ESTADO')
        ->join('pais', 'estado.PAIS_ID', '=', 'pais.PAIS_ID')
        ->select('estado.*','pais.PAIS_NOMBRE')
        ->where('pais.PAIS_ID', '=', $pais_id)
        ->orderBy('ESTADO_ID','asc')
        ->get();
        return $lista;
```

```
        }
    }
    public function store(Request $request)
    {
        $estadoActivo=1;
        $data = array('ESTADO_NOMBRE'=>$request-
>input('ESTADO_NOMBRE'),'PAIS_ID'=>$request-
>input('PAIS_ID'),'ESTADO_ACTIVO'=>$estadoActivo);
        DB::table('estado')->insert($data);
        return back();
    }

    public function show($piso_id)
    {
        $estado = estado::where('ESTADO_ID', '=', $estado_id)->get();
        json_encode($estado);
        return $estado;
    }
```

```
    public function update($id, Request $request)
    {
        if(!$request->ESTADO_ACTIVO) //cuando el checkbox esta en off
        {
            $estadoActivo=0;
        }
        else
        {
            $estadoActivo=1;
        }
        $data = array('ESTADO_NOMBRE'=>$request->input('ESTADO_NOMBRE'),'ESTADO_ACTIVO'=>$estadoActivo);
        DB::table('estado')->where('ESTADO_ID','=', $request->ESTADO_ID)->update($data);
        return back();
    }

    public function destroy($id)
    {
```

```
        ocupacion::findOrFail($id)->delete();
        return 204;
    }
}
```

ingresoController.php

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

use App\ingreso;

use DB;

class ingresoController extends Controller
{
```

```
public function __construct()
{
 $admin = 1;
}

public function redirect(Request $request)
{
    $check = $request->check;
    $id = $request->user_id;
    $rol = usuario::findOrFail($id)->rol;
    $request->session()->put('rol', $rol->rol_descripcion);
    if($check){
        return view('welcome');
    }
    else{
        return back();
    }
}
```

```
public function ingresar(Request $request){
    $resultado = DB::table('users')
        ->where('email', '=', $request->email)
        ->where('password', '=', $request->password)
        ->get()
        ->toArray();
        json_encode($resultado);
    if(count($resultado)>0){
        $user_id=(int)$resultado[0]->id;
        $residente = DB::table('residente')
            ->where('id', '=', $user_id)
            ->select('RESIDENTE_ID')
            ->get()
            ->toArray();
        json_encode($residente);
        if(count($residente)==0){
            $numeraria = DB::table('numeraria')
```

```
            ->where('id', '=', $user_id)
            ->select('NUMERARIA_ID')
            ->get()
            ->toArray();
        json_encode($numeraria);
        $residence_id = (int)$numeraria[0]->NUMERARIA_ID;
        $actividadesRegistradas = DB::table('cumple3')
        ->join('horario', 'horario.HORARIO_ID', '=', 'cumple3.HORARIO_ID')
        ->join('actividad', 'actividad.ACTIVIDAD_ID', '=', 'horario.ACTIVIDAD_ID')
        ->join('tipoactividad', 'tipoactividad.TIPO_ID', '=', 'actividad.TIPO_ID')
        ->select('actividad.*', 'horario.*', 'tipoactividad.TIPO_NOMBRE', 'cumple3.*')
        ->where('cumple3.NUMERARIA_ID', '=', $residence_id)
        ->get();
        json_encode($actividadesRegistradas);
        $actividadesDisponibles = DB::table('horario')
            -
>join('actividad', 'actividad.ACTIVIDAD_ID', '=', 'horario.ACTIVIDAD_ID')
            ->join('tipoactividad', 'tipoactividad.TIPO_ID', '=', 'actividad.TIPO_ID')
            ->select('actividad.*', 'horario.*', 'tipoactividad.TIPO_NOMBRE')
```

```
                ->get();
                json_encode($actividadesDisponibles);
                return view('/registrarActividadNumeraria')
                ->with([
                    'lista' => $actividadesRegistradas,
                    'listaActividades' => $actividadesDisponibles,
                    'numerariaID' => $residence_id
                ]);
                session(['admin' => '1']);
            }
            else{
                $residence_id = (int)$residente[0]->RESIDENTE_ID;
                $actividadesRegistradas = DB::table('cumple')
                ->join('horario', 'horario.HORARIO_ID', '=', 'cumple.HORARIO_ID')
                -
>join('actividad', 'actividad.ACTIVIDAD_ID', '=', 'horario.ACTIVIDAD_ID')
                ->join('tipoactividad', 'tipoactividad.TIPO_ID', '=', 'actividad.TIPO_ID')
                -
>select('actividad.*', 'horario.*', 'tipoactividad.TIPO_NOMBRE', 'cumple.*')
```

```
                ->where('cumple.RESIDENTE_ID', '=', $residence_id)
                ->get();
                json_encode($actividadesRegistradas);
                $actividadesDisponibles = DB::table('horario')
                -
>join('actividad', 'actividad.ACTIVIDAD_ID', '=', 'horario.ACTIVIDAD_ID')
                ->join('tipoactividad', 'tipoactividad.TIPO_ID', '=', 'actividad.TIPO_ID')
                ->select('actividad.*', 'horario.*', 'tipoactividad.TIPO_NOMBRE')
                ->get();
                json_encode($actividadesDisponibles);
                return view('/registrarActividadResidente')
                ->with([
                    'lista' => $actividadesRegistradas,
                    'listaActividades' => $actividadesDisponibles,
                    'residenteID' => $residence_id
                ]);
                session(['admin' => '0']);
            }
        }
```

```
        else{
            return back()->with('alert', 'Cédula ya ingresada');
        }
    }
}
```

numerariaController.php

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

use \App\numeraria;

use \App\pais;

use DB;
```

```
class numerariaController extends Controller
{
    public function index()
    {
        $numeraria = DB::table('numeraria')
            -
>join('ocupacion', 'ocupacion.OCUPACION_ID', '=', 'numeraria.OCUPACION_ID')
            -
>join('carreraxuniversidad', 'carreraxuniversidad.CARRERAXUNI_ID', '=', 'numeraria.C
ARRERAXUNI_ID')
            -
>join('carrera', 'carrera.CARRERA_ID', '=', 'carreraxuniversidad.CARRERA_ID')
            ->join('ciudad', 'ciudad.CIUDAD_ID', '=', 'numeraria.CIUDAD_ID')
            -
>select('numeraria.*', 'ocupacion.OCUPACION_DESCRIPCION', 'carrera.CARRERA_
NOMBRE', 'carreraxuniversidad.CARRERA_ID', 'carreraxuniversidad.UNIVERSIDAD_
ID', 'ciudad.CIUDAD_NOMBRE')
            ->orderBy('NUMERARIA_ID','asc')
            ->get();
```

```
        json_encode($numeraria);
        $ocupacion = DB::table('ocupacion')
        ->where('OCUPACION_ACTIVO', '=', '1')
        ->get();
        json_encode($ocupacion);
        $universidades = DB::table('universidad')
        ->where('UNIVERSIDAD_ACTIVO', '=', '1')
        ->select('universidad.*')
        ->get();
        json_encode($universidades);
        $carrerasXuniversidades = DB::table('carreraxuniversidad')
        -
>join('carrera', 'carrera.CARRERA_ID', '=', 'carreraxuniversidad.CARRERA_ID')
        -
>join('universidad', 'universidad.UNIVERSIDAD_ID', '=', 'carreraxuniversidad.UNIVER
SIDAD_ID')
        ->get();
        json_encode($carrerasXuniversidades);
        $piso = DB::table('piso')
```

```
    ->join('cuarto', 'cuarto.PISO_ID', '=', 'piso.PISO_ID')
    ->where('PISO_ACTIVO', '=', '1')
    ->get();
    json_encode($piso);
    $pais = pais::get();
    json_encode($pais);
    return view('/numerariaView')
                ->with([
                    'lista' => $numeraria,
                    'listaPaises' => $pais,
                    'listaUniversidades' => $universidades,
                    'listaCarreras' => $carrerasXuniversidades,
                    'listaOcupaciones' => $ocupacion,
                    'listaPisos' => $piso
                   ]);
}
public function store(Request $request)
{
```

```
        $numerariaActivo = 1;
        $numeraria = DB::table('numeraria')-
>where('NUMERARIA_CEDULA', '=', $request->NUMERARIA_CEDULA)->get();
        if(sizeof($numeraria) != 0){
            return back()->with('alert', 'Cédula ya ingresada');
        }
        else{
        $carreraXuni = DB::table('carreraxuniversidad')
            ->where('CARRERA_ID', '=', $request->CARRERA_ID)
            ->where('UNIVERSIDAD_ID', '=', $request->UNIVERSIDAD_ID)
            ->select('CARRERAXUNI_ID')
            ->get()
            ->toArray();
            json_encode($carreraXuni);
            $idCarrera=(int)$carreraXuni[0]->CARRERAXUNI_ID;
        $data = array('OCUPACION_ID'=>$request->input('OCUPACION_ID'),
        'CIUDAD_ID'=>$request->input('CIUDAD_ID'),
        'CARRERAXUNI_ID'=>$idCarrera,
        'CUARTO_ID'=>$request->input('CUARTO_ID'),
```

```
            'NUMERARIA_CEDULA'=>$request->input('NUMERARIA_CEDULA'),
            'NUMERARIA_NOMBRE'=>$request->input('NUMERARIA_NOMBRE'),
            'NUMERARIA_CELULAR'=>$request->input('NUMERARIA_CELULAR'),
            'NUMERARIA_TELEFONO'=>$request->input('NUMERARIA_TELEFONO'),
            'NUMERARIA_MAIL'=>$request->input('NUMERARIA_MAIL'),
            'NUMERARIA_FECHA_NACIMIENTO'=>$request-
>input('NUMERARIA_FECHA_NACIMIENTO'),
            'NUMERARIA_ACTIVO'=>$numerariaActivo);
            DB::table('numeraria')->insert($data);
            $rol = 1;
            $contrasena = "1q2w3e4r";
            $datauser = array('NAME'=>$request->input('NUMERARIA_NOMBRE'),
            'EMAIL'=>$request->input('NUMERARIA_MAIL'),
            'PASSWORD'=>$contrasena,
            'ROL_ID'=>$rol
            );
            DB::table('users')->insert($datauser);
            $userId = DB::table('users')
            ->orderBy('ID', 'desc')
```

```
->limit(1)

->select('ID')

->get()

->toArray();

json_encode($userId);

$idUser=(int)$userId[0]->ID;



$numId = DB::table('numeraria')

->orderBy('NUMERARIA_ID', 'desc')

->limit(1)

->select('NUMERARIA_ID')

->get()

->toArray();

json_encode($numId);

$idNum=(int)$numId[0]->NUMERARIA_ID;



$dataUpdate = array(

    'ID'=>$idUser);
```

```
            DB::table('numeraria')->where('NUMERARIA_ID','=', $idNum)-
>update($dataUpdate);


            return back();
            }
        }

        public function show($numeraria_id)
        {
            $numeraria = numeraria::where('NUMERARIA_ID', '=', $numeraria_id)->get();
            json_encode($numeraria);
            return $numeraria;
        }

        public function updateRoom($id, Request $request){
            $data2 = array(
                'CUARTO_OCUPADO'=>'1');
                DB::table('cuarto')->where('CUARTO_ID','=', $request-
>input('CUARTO_ID'))->update($data2);
```

```
    $data = array(
    'CUARTO_ID'=>$request->input('CUARTO_ID'));
    DB::table('numeraria')->where('NUMERARIA_ID','=', $id)->update($data);
    return back();
}
public function update($id, Request $request)
{
    if(!$request->NUMERARIA_ACTIVO){ //cuando el checkbox esta en off
        $numerariaActivo=0;
    }
    else{
        $numerariaActivo=1;
    }
    $carreraXuni = DB::table('carreraxuniversidad')
        ->where('CARRERA_ID', '=', $request->CARRERA_ID)
        ->where('UNIVERSIDAD_ID', '=', $request->UNIVERSIDAD_ID)
        ->select('CARRERAXUNI_ID')
        ->get()
```

```
            ->toArray();
            json_encode($carreraXuni);
            $idCarrera=(int)$carreraXuni[0]->CARRERAXUNI_ID;
        $data = array('OCUPACION_ID'=>$request->input('OCUPACION_ID'),
        'CARRERAXUNI_ID'=>$idCarrera,
        'CUARTO_ID'=>$request->input('CUARTO_ID'),
        'NUMERARIA_CEDULA'=>$request->input('NUMERARIA_CEDULA'),
        'NUMERARIA_NOMBRE'=>$request->input('NUMERARIA_NOMBRE'),
        'NUMERARIA_CELULAR'=>$request->input('NUMERARIA_CELULAR'),
        'NUMERARIA_TELEFONO'=>$request->input('NUMERARIA_TELEFONO'),
        'NUMERARIA_MAIL'=>$request->input('NUMERARIA_MAIL'),
        'NUMERARIA_FECHA_NACIMIENTO'=>$request-
>input('NUMERARIA_FECHA_NACIMIENTO'),
        'NUMERARIA_ACTIVO'=>$numerariaActivo);
        DB::table('numeraria')->where('NUMERARIA_ID','=', $id)->update($data);
        return back();
    }

    public function destroy($id)
```

```
    {
        ocupacion::findOrFail($id)->delete();
        return 204;
    }
}
```

ocupacionController.php

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

use \App\ocupacion;

use DB;

class ocupacionController extends Controller
```

```
{
    public function index()
    {
        $ocupacion = ocupacion::get();
        json_encode($ocupacion);
        return view('/ocupacionView',['lista'=>$ocupacion]);
        //return $ocupacion;
    }

    public function getOcupaciones(Request $request){
        if($request->ajax())
        {
            $lista = DB::table('ocupacion')
            ->where('OCUPACION_ACTIVO', '=', '1')
            ->orderBy('OCUPACION_ID','asc')
            ->get();
            return $lista;
        }
```

```
    }

    public function store(Request $request)
    {
        $ocupacionActivo=1;
        $data = array('OCUPACION_DESCRIPCION'=>$request-
>input('OCUPACION_DESCRIPCION'),'OCUPACION_ACTIVO'=>$ocupacionActivo)
;
        DB::table('ocupacion')->insert($data);
        return back();
    }

    public function show($ocupacion_id)
    {
        $ocupacion = ocupacion::where('OCUPACION_ID', '=', $ocupacion_id)->get();
        json_encode($ocupacion);
        return $ocupacion;
    }
```

```
    public function update($id, Request $request)
    {
        if(!$request->OCUPACION_ACTIVO) //cuando el checkbox esta en off
        {
            $ocupacionActivo=0;
        }
        else
        {
            $ocupacionActivo=1;
        }
        $data = array('OCUPACION_DESCRIPCION'=>$request->input('OCUPACION_DESCRIPCION'),'OCUPACION_ACTIVO'=>$ocupacionActivo);
        DB::table('ocupacion')->where('OCUPACION_ID','=', $request->OCUPACION_ID)->update($data);
        return back();
    }

    public function destroy($id)
```

```
    {
        ocupacion::findOrFail($id)->delete();
        return 204;
    }
}
```

padreMadreController.php

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

use \App\padreMadre;

use DB;

class padreMadreController extends Controller
```

```
    {
        public function index()
        {
            $residente = DB::table('residente')
                ->join('colegio', 'colegio.COLEGIO_ID', '=', 'residente.COLEGIO_ID')
                ->join('ciudad', 'ciudad.CIUDAD_ID', '=', 'residente.CIUDAD_ID')
                -
>select('residente.*', 'ciudad.CIUDAD_NOMBRE', 'colegio.COLEGIO_NOMBRE')
                ->where('residente.RESIDENTE_INGRESO', '=', '0')
                ->orderBy('RESIDENTE_ID','asc')
                ->get();
            json_encode($residente);
            $pais = pais::get();
            json_encode($pais);
            return view('/posibleResidenteView')
                        ->with([
                            'lista' => $residente,
                            'listaPaises' => $pais
                           ]);
```

```
}

public function getPadre(Request $request, $id_residente){
    if($request->ajax())
    {
        $padre = DB::table('padremadre')
        ->where('RESIDENTE_ID', '=', $id_residente)
        ->where('PM_PADRE', '=', '1')
        ->get();
        return ($padre);
    }
}

public function getMadre(Request $request, $id_residente){
    if($request->ajax())
    {
        $madre = DB::table('padremadre')
        ->where('RESIDENTE_ID', '=', $id_residente)
```

```
            ->where('PM_PADRE', '=', '0')
            ->get();
            return ($madre);
        }
    }

    public function store(Request $request)
    {
        /*$data = array('RESIDENTE_ID'=>$request->input('RESIDENTE_ID'),
        'OCUPACION_ID'=>$request->input('OCUPACION_ID'),
        'PM_NOMBRE'=>$request->input('PM_NOMBRE'),
        'PM_DIRECCION_OFICINA'=>$request->input('PM_DIRECCION_OFICINA'),
        'PM_DIRECCION_OFICINA'=>$request->input('PM_DIRECCION_OFICINA'));
        DB::table('padremadre')->insert($data);*/
        return padreMadre::create($request->all());
        return back();
    }
```

```
        public function show($padreMadre_id)
        {
            $padreMadre = padremadre::where('RESIDENTE_ID', '=', $padreMadre_id)->get();
            json_encode($padreMadre);
            return $padreMadre;
        }

        //id es de la residente para poder linkearlo con el padre o madre
        public function fatherCreate($id, Request $request){
            $pmPadre = 1;
            $data = array('PM_NOMBRE'=>$request->input('PM_NOMBRE'),
                'OCUPACION_ID'=>$request->input('OCUPACION_ID'),
                'PM_DIRECCION_OFICINA'=>$request->input('PM_DIRECCION_OFICINA'),
                'PM_TELEFONO_OFICINA'=>$request->input('PM_TELEFONO_OFICINA'),
                'PM_PADRE'=>$pmPadre,
```

```
            'RESIDENTE_ID'=>$id);
            DB::table('padremadre')->insert($data);
            return back();
        }

        public function fatherUpdate($id, Request $request){
            if($request->input('PM_ID') == 0){
                $pmPadre = 1;
                $data = array('PM_NOMBRE'=>$request->input('PM_NOMBRE'),
                    'OCUPACION_ID'=>$request->input('OCUPACION_ID'),
                    'PM_DIRECCION_OFICINA'=>$request-
>input('PM_DIRECCION_OFICINA'),
                    'PM_TELEFONO_OFICINA'=>$request-
>input('PM_TELEFONO_OFICINA'),
                    'PM_PADRE'=>$pmPadre,
                    'RESIDENTE_ID'=>$id);
                    DB::table('padremadre')->insert($data);
            }
            else{
```

```
            $data = array('PM_NOMBRE'=>$request->input('PM_NOMBRE'),
            'OCUPACION_ID'=>$request->input('OCUPACION_ID'),
            'PM_DIRECCION_OFICINA'=>$request->input('PM_DIRECCION_OFICINA'),
            'PM_TELEFONO_OFICINA'=>$request->input('PM_TELEFONO_OFICINA'));
            DB::table('padremadre')->where('PM_ID','=', $request->input('PM_ID'))->update($data);
        }

        return back();
    }

    public function motherCreate($id, Request $request){
        $pmPadre = 0;
        $data = array('PM_NOMBRE'=>$request->input('PM_NOMBRE'),
        'OCUPACION_ID'=>$request->input('OCUPACION_ID'),
        'PM_DIRECCION_OFICINA'=>$request->input('PM_DIRECCION_OFICINA'),
```

```
            'PM_TELEFONO_OFICINA'=>$request->input('PM_TELEFONO_OFICINA'),
            'PM_PADRE'=>$pmPadre,
            'RESIDENTE_ID'=>$id);
            DB::table('padremadre')->insert($data);
            return back();
        }

        public function motherUpdate($id, Request $request){
            if($request->input('PM_ID') == 0){
                $pmPadre = 0;
                $data = array('PM_NOMBRE'=>$request->input('PM_NOMBRE'),
                'OCUPACION_ID'=>$request->input('OCUPACION_ID'),
                'PM_DIRECCION_OFICINA'=>$request-
>input('PM_DIRECCION_OFICINA'),
                'PM_TELEFONO_OFICINA'=>$request-
>input('PM_TELEFONO_OFICINA'),
                'PM_PADRE'=>$pmPadre,
                'RESIDENTE_ID'=>$id);
                DB::table('padremadre')->insert($data);
```

```
            }
            else{
                $data = array('PM_NOMBRE'=>$request->input('PM_NOMBRE'),
                'OCUPACION_ID'=>$request->input('OCUPACION_ID'),
                'PM_DIRECCION_OFICINA'=>$request-
>input('PM_DIRECCION_OFICINA'),
                'PM_TELEFONO_OFICINA'=>$request-
>input('PM_TELEFONO_OFICINA'));
                DB::table('padremadre')->where('PM_ID','=', $request->input('PM_ID'))-
>update($data);
            }

            return back();
        }

        public function update($id, Request $request)
        {
            $data = array('RESIDENTE_NOMBRE'=>$request-
>input('RESIDENTE_NOMBRE'),
```

```
            'RESIDENTE_CEDULA'=>$request->input('RESIDENTE_CEDULA'),
            'RESIDENTE_MAIL'=>$request->input('RESIDENTE_MAIL'),
            'RESIDENTE_DIRECCION'=>$request->input('RESIDENTE_DIRECCION'),
            'RESIDENTE_TELEFONO'=>$request->input('RESIDENTE_TELEFONO'),
            'RESIDENTE_CELULAR'=>$request->input('RESIDENTE_CELULAR'),
            'RESIDENTE_SEMESTRE_A_CURSAR'=>$request-
>input('RESIDENTE_SEMESTRE_A_CURSAR'),
            'RESIDENTE_PROMEDIO_GRADUACION'=>$request-
>input('RESIDENTE_PROMEDIO_GRADUACION'),
            'RESIDENTE_FECHA_NAC'=>$request->input('RESIDENTE_FECHA_NAC'),
            'RESIDENTE_FECHA_INGRESO'=>$request-
>input('RESIDENTE_FECHA_INGRESO'),
            'RESIDENTE_ACTIVO'=>$residenteActivo,
            'RESIDENTE_INGRESO'=>$residenteIngreso);
            DB::table('residente')->where('RESIDENTE_ID','=', $id)->update($data);
            return back();
        }

        public function destroy($id)
```

```
    {
        ocupacion::findOrFail($id)->delete();
        return 204;
    }
}
```

paisController.php

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

use \App\pais;

use DB;

class paisController extends Controller
```

```
{
    public function index()
    {
        $pais = pais::get();
        json_encode($pais);
        return view('/paisView',['lista'=>$pais]);
        //return $ocupacion;
    }

    public function verPaises()
    {
        $pais = pais::get();
        json_encode($pais);
        return view('/ciudadView',['listaPaises'=>$pais]);
    }
    public function store(Request $request)
    {
        $paisActivo=1;
```

```
        $data = array('PAIS_NOMBRE'=>$request-
>input('PAIS_NOMBRE'),'PAIS_ACTIVO'=>$paisActivo);
        DB::table('pais')->insert($data);
        return back();
    }

    public function show($pais_id)
    {
        $pais = pais::where('PAIS_ID', '=', $pais_id)->get();
        json_encode($pais);
        return $pais;
    }

    public function update($id, Request $request)
    {
        if(!$request->PAIS_ACTIVO) //cuando el checkbox esta en off
        {
            $paisActivo=0;
        }
```

```
            else
            {
               $paisActivo=1;
            }
            $data = array('PAIS_NOMBRE'=>$request-
>input('PAIS_NOMBRE'),'PAIS_ACTIVO'=>$paisActivo);
            DB::table('pais')->where('PAIS_ID','=', $id)->update($data);
            return back();
        }

        public function destroy($id)
        {
            ocupacion::findOrFail($id)->delete();
            return 204;
        }
    }
```

pisoController.php

```
<?php
```

```
namespace App\Http\Controllers;

use Illuminate\Http\Request;

use \App\piso;

use DB;

class pisoController extends Controller
{
    public function index()
    {
        $piso = piso::get();
        json_encode($piso);
        return view('/pisoView',['lista'=>$piso]);
    }
```

```
    public function verPisos()
    {
        $piso = DB::table('piso')
        ->where('piso.PISO_ACTIVO', '=', '1')
        ->select('piso.PISO_ID', 'piso.PISO_NUMERO')
        ->get();
        json_encode($piso);
        return view('/cuartoView',['listaPisos'=>$piso]);
    }

    public function store(Request $request)
    {
        $pisoActivo=1;
        $data = array('PISO_NUMERO'=>$request-
>input('PISO_NUMERO'),'PISO_ACTIVO'=>$pisoActivo);
        DB::table('piso')->insert($data);
        return back();
    }
```

```
    public function show($piso_id)
    {
        $piso = piso::where('PISO_ID', '=', $piso_id)->get();
        json_encode($piso);
        return $piso;
    }

    public function update($id, Request $request)
    {
        if(!$request->PISO_ACTIVO) //cuando el checkbox esta en off
        {
            $pisoActivo=0;
        }
        else
        {
            $pisoActivo=1;
        }
        $data = array('PISO_NUMERO'=>$request-
>input('PISO_NUMERO'),'PISO_ACTIVO'=>$pisoActivo);
```

```
        DB::table('piso')->where('PISO_ID','=', $request->PISO_ID)->update($data);
        return back();
    }

    public function destroy($id)
    {
        ocupacion::findOrFail($id)->delete();
        return 204;
    }
}
```

residenteController.php

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
```

```
use \App\residente;

use \App\pais;

use \App\universidad;

use DB;

class residenteController extends Controller
{
   public function index()
   {
      $residente = DB::table('residente')
         ->join('colegio', 'colegio.COLEGIO_ID', '=', 'residente.COLEGIO_ID')
         ->join('ciudad', 'ciudad.CIUDAD_ID', '=', 'residente.CIUDAD_ID')
         -
>join('carreraxuniversidad', 'carreraxuniversidad.CARRERAXUNI_ID', '=', 'residente.CA
RRERAXUNI_ID')
```

```
            -
>join('carrera', 'carrera.CARRERA_ID', '=', 'carreraxuniversidad.CARRERA_ID')
            -
>select('residente.*', 'ciudad.CIUDAD_NOMBRE', 'colegio.COLEGIO_NOMBRE', 'carr
era.CARRERA_NOMBRE', 'carreraxuniversidad.CARRERA_ID', 'carreraxuniversidad.
UNIVERSIDAD_ID')
            ->where('residente.RESIDENTE_INGRESO', '=', '0')
            ->orderBy('RESIDENTE_ID','asc')
            ->get();
        json_encode($residente);
        $universidades = DB::table('universidad')
        ->where('UNIVERSIDAD_ACTIVO', '=', '1')
        ->get();
        $pais = pais::get();
        $carrerasXuniversidades = DB::table('carreraxuniversidad')
        -
>join('carrera', 'carrera.CARRERA_ID', '=', 'carreraxuniversidad.CARRERA_ID')
        -
>join('universidad', 'universidad.UNIVERSIDAD_ID', '=', 'carreraxuniversidad.UNIVER
SIDAD_ID')
```

```
    ->get();
    json_encode($pais);
    $ocupacion = DB::table('ocupacion')
    ->where('OCUPACION_ACTIVO', '1')
    ->get();
    json_encode($ocupacion);
    return view('/posibleResidenteView')
            ->with([
                'lista' => $residente,
                'listaPaises' => $pais,
                'listaUniversidades' => $universidades,
                'listaCarreras' => $carrerasXuniversidades,
                'listaOcupaciones' =>$ocupacion
                ]);
}
public function getResidentes()
{
    $residente = DB::table('residente')
```

```
        ->join('colegio', 'colegio.COLEGIO_ID', '=', 'residente.COLEGIO_ID')
        ->join('ciudad', 'ciudad.CIUDAD_ID', '=', 'residente.CIUDAD_ID')
        ->join('carreraxuniversidad', 'carreraxuniversidad.CARRERAXUNI_ID', '=', 'residente.CARRERAXUNI_ID')
        ->join('carrera', 'carrera.CARRERA_ID', '=', 'carreraxuniversidad.CARRERA_ID')
        ->select('residente.*', 'ciudad.CIUDAD_NOMBRE', 'colegio.COLEGIO_NOMBRE', 'carrera.CARRERA_NOMBRE', 'carreraxuniversidad.CARRERA_ID', 'carreraxuniversidad.UNIVERSIDAD_ID')
        ->where('residente.RESIDENTE_INGRESO', '=', '1')
        ->orderBy('RESIDENTE_ID','asc')
        ->get();
        json_encode($residente);
        $universidades = DB::table('universidad')
        ->where('UNIVERSIDAD_ACTIVO', '=', '1')
        ->get();
        $pais = pais::get();
        $carrerasXuniversidades = DB::table('carreraxuniversidad')
```

```
            -
>join('carrera', 'carrera.CARRERA_ID', '=', 'carreraxuniversidad.CARRERA_ID')
            -
>join('universidad', 'universidad.UNIVERSIDAD_ID', '=', 'carreraxuniversidad.UNIVER
SIDAD_ID')
            ->get();
            json_encode($pais);
            $ocupacion = DB::table('ocupacion')
            ->where('OCUPACION_ACTIVO', '1')
            ->get();
            json_encode($ocupacion);
            $piso = DB::table('piso')
            ->join('cuarto', 'cuarto.PISO_ID', '=', 'piso.PISO_ID')
            ->where('PISO_ACTIVO', '=', '1')
            ->get();
            json_encode($piso);
            return view('/residenteView')
                  ->with([
                        'lista' => $residente,
```

```
                    'listaPaises' => $pais,
                    'listaUniversidades' => $universidades,
                    'listaCarreras' => $carrerasXuniversidades,
                    'listaOcupaciones' =>$ocupacion,
                    'listaPisos' =>$piso
                   ]);
        }

        public function store(Request $request)
        {
            $carrerauni = 0;
            $carreraXuni = 0;
            $iddef=0;
            $residente = DB::table('residente')-
>where('RESIDENTE_CEDULA', '=', $request->RESIDENTE_CEDULA)->get();
            if(sizeof($residente) != 0){
                return back()->with('alert', 'Cédula ya ingresada');
            }
            else{
```

```
            $carreraXuni = DB::table('carreraxuniversidad')
            ->where('CARRERA_ID', '=', $request->CARRERA_ID)
            ->where('UNIVERSIDAD_ID', '=', $request->UNIVERSIDAD_ID)
            ->select('CARRERAXUNI_ID')
            ->get()
            ->toArray();
            json_encode($carreraXuni);
            $id=(int)$carreraXuni[0]->CARRERAXUNI_ID;
            $residenteActivo=1;
            $residenteIngreso=0;
            $data = array('RESIDENTE_NOMBRE'=>$request-
>input('RESIDENTE_NOMBRE'),
            'RESIDENTE_CEDULA'=>$request->input('RESIDENTE_CEDULA'),
            'CIUDAD_ID'=>$request->input('CIUDAD_ID'),
            'COLEGIO_ID'=>$request->input('COLEGIO_ID'),
            'RESIDENTE_MAIL'=>$request->input('RESIDENTE_MAIL'),
            'RESIDENTE_DIRECCION'=>$request-
>input('RESIDENTE_DIRECCION'),
            'RESIDENTE_TELEFONO'=>$request->input('RESIDENTE_TELEFONO'),
```

```
            'RESIDENTE_CELULAR'=>$request->input('RESIDENTE_CELULAR'),
            'RESIDENTE_SEMESTRE_A_CURSAR'=>$request-
>input('RESIDENTE_SEMESTRE_A_CURSAR'),
            'RESIDENTE_PROMEDIO_GRADUACION'=>$request-
>input('RESIDENTE_PROMEDIO_GRADUACION'),
            'RESIDENTE_FECHA_NAC'=>$request-
>input('RESIDENTE_FECHA_NAC'),
            'RESIDENTE_FECHA_INGRESO'=>$request-
>input('RESIDENTE_FECHA_INGRESO'),
            'RESIDENTE_INGRESO'=>$residenteIngreso,
            'RESIDENTE_ACTIVO'=>$residenteActivo,
            'CARRERAXUNI_ID'=>$id);
            DB::table('residente')->insert($data);
            return back();
        }
    }

    public function show($residented_id)
    {
```

```
    $residente = residente::where('RESIDENTE_ID', '=', $residente_id)->get();
    json_encode($residente);
    return $residente;
}

public function updateRoom($id, Request $request){
    $data = array(
    'CUARTO_ID'=>$request->input('CUARTO_ID'));
    DB::table('residente')->where('RESIDENTE_ID','=', $id)->update($data);
    return back();
}

public function update($id, Request $request)
{
    $idUser=0;
    if(!$request->RESIDENTE_ACTIVO){ //cuando el checkbox esta en off
        $residenteActivo=0;
    }
```

```
else{
    $residenteActivo=1;
}
if(!$request->RESIDENTE_INGRESO){ //cuando el checkbox esta en off
    $residenteIngreso=0;
    $idUser=0;
}
else{
    $residenteIngreso=1;

    //ingresa el usuario
    $rol = 2;
    $contrasena = "1q2w3e4r";
    $datauser = array('NAME'=>$request->input('RESIDENTE_NOMBRE'),
    'EMAIL'=>$request->input('RESIDENTE_MAIL'),
    'PASSWORD'=>$contrasena,
    'ROL_ID'=>$rol
    );
```

```
    DB::table('users')->insert($datauser);
    $userId = DB::table('users')
    ->orderBy('ID', 'desc')
    ->limit(1)
    ->select('ID')
    ->get()
    ->toArray();
    json_encode($userId);
    $idUser=(int)$userId[0]->ID;
}
$carreraXuni = DB::table('carreraxuniversidad')
    ->where('CARRERA_ID', '=', $request->CARRERA_ID)
    ->where('UNIVERSIDAD_ID', '=', $request->UNIVERSIDAD_ID)
    ->select('CARRERAXUNI_ID')
    ->get()
    ->toArray();
    json_encode($carreraXuni);
    $idCarrera=(int)$carreraXuni[0]->CARRERAXUNI_ID;
```

```
        $data = array('RESIDENTE_NOMBRE'=>$request-
>input('RESIDENTE_NOMBRE'),
        'RESIDENTE_CEDULA'=>$request-
>input('RESIDENTE_CEDULAUPDATE'),
        'RESIDENTE_MAIL'=>$request->input('RESIDENTE_MAIL'),
        'RESIDENTE_DIRECCION'=>$request->input('RESIDENTE_DIRECCION'),
        'RESIDENTE_TELEFONO'=>$request->input('RESIDENTE_TELEFONO'),
        'RESIDENTE_CELULAR'=>$request->input('RESIDENTE_CELULAR'),
        'RESIDENTE_SEMESTRE_A_CURSAR'=>$request-
>input('RESIDENTE_SEMESTRE_A_CURSAR'),
        'RESIDENTE_PROMEDIO_GRADUACION'=>$request-
>input('RESIDENTE_PROMEDIO_GRADUACION'),
        'RESIDENTE_FECHA_NAC'=>$request->input('RESIDENTE_FECHA_NAC'),
        'RESIDENTE_FECHA_INGRESO'=>$request-
>input('RESIDENTE_FECHA_INGRESO'),
        'RESIDENTE_ACTIVO'=>$residenteActivo,
        'RESIDENTE_INGRESO'=>$residenteIngreso,
        'CARRERAXUNI_ID'=>$idCarrera,
        'RESIDENTE_ID'=>$idUser);
```

```
        DB::table('residente')->where('RESIDENTE_ID','=', $id)->update($data);
        return back();
    }

    public function destroy($id)
    {
        ocupacion::findOrFail($id)->delete();
        return 204;
    }
}
```

tipoActividadController.php

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
```

```
use \App\tipoActividad;

use DB;

class tipoActividadController extends Controller
{
    public function index()
    {
        $tipoActividad = tipoActividad::get();
        json_encode($tipoActividad);
        return view('/tipoActividadView',['lista'=>$tipoActividad]);
    }

    public function store(Request $request)
    {
        $tipoActividadActivo=1;
        $data = array('TIPO_NOMBRE'=>$request-
>input('TIPO_NOMBRE'),'TIPO_ACTIVO'=>$tipoActividadActivo);
        DB::table('tipoactividad')->insert($data);
```

```
    return back();
}

public function show($tipoActividad_id)
{
    $tipoActividad = tipoActividad::where('TIPO_ID', '=', $tipoActividad_id)->get();
    json_encode($tipoActividad);
    return $tipoActividad;
}

public function update($id, Request $request)
{
    if(!$request->TIPO_ACTIVO) //cuando el checkbox esta en off
    {
        $tipoActividadActivo=0;
    }
    else
    {
```

```
            $tipoActividadActivo=1;
        }
        $data = array('TIPO_NOMBRE'=>$request->input('TIPO_NOMBRE'),'TIPO_ACTIVO'=>$tipoActividadActivo);
        DB::table('tipoactividad')->where('TIPO_ID','=', $id)->update($data);
        return back();
    }

    public function destroy($id)
    {
        ocupacion::findOrFail($id)->delete();
        return 204;
    }
}
```

universidadController.php

```
<?php

namespace App\Http\Controllers;
```

```
use Illuminate\Http\Request;

use \App\universidad;

use DB;

class universidadController extends Controller
{
    public function index()
    {
        $universidad = universidad::get();
        json_encode($universidad);
        return view('/universidadView',['lista'=>$universidad]);
    }

    public function store(Request $request)
    {
```

```
            $universidadActivo=1;
            $data = array('UNIVERSIDAD_NOMBRE'=>$request-
>input('UNIVERSIDAD_NOMBRE'),'UNIVERSIDAD_ACTIVO'=>$universidadActivo
);
            DB::table('universidad')->insert($data);
            return back();
        }

        public function show($universidad_id)
        {
            $universidad = universidad::where('UNIVERSIDAD_ID', '=', $universidad_id)-
>get();
            json_encode($universidad);
            return $universidad;
        }

        public function update($id, Request $request)
        {
            if(!$request->UNIVERSIDAD_ACTIVO) //cuando el checkbox esta en off
```

```
            {
                $universidadActivo=0;
            }
            else
            {
                $universidadActivo=1;
            }
            $data = array('UNIVERSIDAD_NOMBRE'=>$request->input('UNIVERSIDAD_NOMBRE'),'UNIVERSIDAD_ACTIVO'=>$universidadActivo);
            DB::table('universidad')->where('UNIVERSIDAD_ID','=', $request->UNIVERSIDAD_ID)->update($data);
            return back();
        }

        public function destroy($id)
        {
            ocupacion::findOrFail($id)->delete();
            return 204;
```

```
    }
}
```

Vista

actividadView.blade.php

```
@extends('layouts.app')

@section('title', 'Adm. Clientes')

@section('content')
    <div class="container">
    <form class="form-inline d-flex justify-content-center md-form form-
sm searchbox">
        <input class="form-control form-control-sm mr-3 w-
75" type="text" placeholder="Search" id="myInput" name="myInput"
            aria-label="Search" onkeyup="searchFunction()">
    </form>

    <script type="text/javascript">
```

```
function searchFunction(){
    // Variables
    var input, filter, table, tr, td, i, txtValue;
    input = document.getElementById("myInput");
    filter = input.value.toUpperCase();
    table = document.getElementById("myTable");
    tr = table.getElementsByTagName("tr");



    for (i = 0; i < tr.length; i++) {
        td = tr[i].getElementsByTagName("td")[2];
        if (td) {
            txtValue = td.textContent || td.innerText;
            if (txtValue.toUpperCase().indexOf(filter) > -1) {
                tr[i].style.display = "";
            } else {
                tr[i].style.display = "none";
            }
```

```
        }
        td2 = tr[i].getElementsByTagName("td")[3];
        if (td2) {
            txtValue = td2.textContent || td2.innerText;
            if (txtValue.toUpperCase().indexOf(filter) > -1) {
                tr[i].style.display = "";
            } else {
                tr[i].style.display = "none";
            }
        }
    }
}
</script>
    <div class="table-wrapper">
        <div class="table-title">
            <div class="row">
                <div class="col-sm-6">
                    <h2>Administrar <b>Actividades</b></h2>
```

```
                    </div>
                    <div class="col-sm-6">
                        <a href="#addModal" class="btn btn-success" data-
toggle="modal"><i class="material-
icons">&#xE147;</i> <span>Nueva Actividad</span></a>
                    </div>
                </div>
            </div>
            <!-- Add Modal HTML -->
        <div id="addModal" class="modal fade">
            <div class="modal-dialog">
                <div class="modal-content">
                    <form action="/actividadCreate" method="get" id="form">
                        <div class="modal-header">
                            <h4 class="modal-title">Crear</h4>
                            <button type="button" class="close" data-dismiss="modal" aria-
hidden="true">&times;</button>
                        </div>
                        <div class="modal-body">
```

```
                    <div class="form-group" hidden>
                        <label for="id">ID:</label>
                        <input type="text" id="TIPO_ID" class="form-control">
                    </div>
                    <div class="form-group">
                        <label for="descripcion">Tipo de actividad:</label>
                        <select name="TIPO_ID" id="TIPO_ID" required>
                        @foreach($listaTipos as $tipo)
                            <option value="{{$tipo->TIPO_ID}}">{{$tipo-
>TIPO_NOMBRE}}</option>
                        @endforeach
                        </select>
                    </div>
                    <div class="form-group">
                        <label for="id">Nombre:</label>
                        <input type="text" id="ACTIVIDAD_NOMBRE" name="ACTIVID
AD_NOMBRE" class="form-control">
                    </div>
                    <div class="form-group">
```

```
                    <label for="id">Descripcion:</label>
                    <textarea id="ACTIVIDAD_DESCRIPCION" name="ACTIVIDAD
_DESCRIPCION" class="form-control"></textarea>
                </div>
                <div class="form-group">
                    <label for="id">Fecha inicio:</label>
                    <input type="date" id="HORARIO_FECHA_INICIO" name="HOR
ARIO_FECHA_INICIO">
                </div>
                <div class="form-group">
                    <label for="id">Hora:</label><br/>
                    <input type="time" id="HORARIO_HORA" name="HORARIO_H
ORA">
                </div>
                <div class="form-group">
                    <label for="id">Fecha fin:</label><br/>
                    <input type="date" id="HORARIO_FECHA_FIN" name="HORARI
O_FECHA_FIN">
                </div>
```

```
                    <div class="form-group">
                        <label>Activo:</label><br>
                        <input type="checkbox" class="custom-
checkbox" id="ACTIVIDAD_ACTIVO" name="ACTIVIDAD_ACTIVO" disabled="tru
e" value="1" checked>
                    </div>
                    <!--<div class="form-group">
                        <label>Repetir diariamente:</label><br>
                        <input type="radio" class="custom-
checkbox" id="OCUPACION_ACTIVO" name="OCUPACION_ACTIVO" value="1">
                    </div>
                    <div class="form-group">
                        <label>Repetir semanalmente:</label><br>
                        <input type="radio" class="custom-
checkbox" id="OCUPACION_ACTIVO" name="OCUPACION_ACTIVO" value="1">
                    </div>-->
                </div>
                <div class="modal-footer">
```

```
                    <input type="button" class="btn btn-default" data-
dismiss="modal" value="Cancelar">
                    <input type="submit" class="btn btn-success" value="Agregar">
                </div>
            </form>
        </div>
    </div>
</div>
    <table class="table table-striped table-hover" id="myTable">
        <thead>
            <tr>
                <th>
                    <span class="custom-checkbox">
                        <input type="checkbox" id="selectAll">
                        <label for="selectAll"></label>
                    </span>
                </th>
                <th>ID</th>
                <th>Nombre</th>
```

```
                    <th>Descripción</th>
                    <th>Fecha Inicio</th>
                    <th>Horario</th>
                    <th>Fecha Fin</th>
                    <!--<th>Repeticion</th>-->
                    <th>Activo</th>
                </tr>
            </thead>
        @foreach($lista as $elemento)
            <tbody>
                <tr>
                    <td>
                        <span class="custom-checkbox">
                            <input type="checkbox" id="checkbox1" name="options[]" value
="1">
                            <label for="checkbox1"></label>
                        </span>
                    </td>
                    <td>{{$elemento->ACTIVIDAD_ID}}</td>
```

```
                    <td>{{$elemento->ACTIVIDAD_NOMBRE}}</td>

                    <td>{{$elemento->ACTIVIDAD_DESCRIPCION}}</td>

                    <td>{{$elemento->HORARIO_FECHA_INICIO}}</td>

                    <td>{{$elemento->HORARIO_HORA}}</td>

                    <td>{{$elemento->HORARIO_FECHA_FIN}}</td>

                    @if($elemento->ACTIVIDAD_ACTIVO == 1)

                       <td><input type=checkbox class="custom-
checkbox" id="CB{{$elemento-
>ACTIVIDAD_ACTIVO}}" checked disabled="true"></td>

                    @else

                       <td><input type=checkbox class="custom-
checkbox" id="CB{{$elemento->ACTIVIDAD_ACTIVO}}" disabled="true"></td>

                    @endif

                    </td>



    <!-- Edit Modal HTML -->

    <div id="editModal-{{$elemento->ACTIVIDAD_ID}}" class="modal fade">

       <div class="modal-dialog">

            <div class="modal-content">
```

```
                <form action="/actividadUpdate/{{$elemento->ACTIVIDAD_ID}}" method="get" id="forma">
                    <div class="modal-header">
                        <h4 class="modal-title">Editar</h4>
                        <button type="button" class="close" data-dismiss="modal" aria-hidden="true">&times;</button>
                    </div>
                    <div class="modal-body">
                        <div class="form-group" hidden>
                            <label for="id">HORARIO ID:</label>
                            <input type="text" name="HORARIO_ID" id="HORARIO_ID" class="form-control" value="{{$elemento->HORARIO_ID}}" disabled="true" required>
                        </div>
                        <div class="form-group">
                            <label for="id">ID:</label>
                            <input type="text" name="ACTIVIDAD_ID" id="ACTIVIDAD_ID" class="form-control" value="{{$elemento->ACTIVIDAD_ID}}" disabled="true" required>
                        </div>
```

```
                    <div class="form-group">
                        <label for="descripcion">Tipo de actividad:</label>
                        <select name="TIPO_ID" id="TIPO_ID" required>
                        @foreach($listaTipos as $tipo)
                            @if($tipo->TIPO_ID == $elemento->TIPO_ID)
                            <option value="{{$tipo->TIPO_ID}}" selected>{{$tipo-
>TIPO_NOMBRE}}</option>
                            @else
                            <option value="{{$tipo->TIPO_ID}}">{{$tipo-
>TIPO_NOMBRE}}</option>
                            @endif
                        @endforeach
                        </select>
                    </div>
                    <div class="form-group">
                        <label for="id">Nombre:</label>
                        <input type="text" id="ACTIVIDAD_NOMBRE" name="ACTIVID
AD_NOMBRE" class="form-control" value="{{$elemento-
>ACTIVIDAD_NOMBRE}}">
```

```
                    </div>

                    <div class="form-group">

                        <label for="id">Descripcion:</label>

                        <textarea id="ACTIVIDAD_DESCRIPCION" name="ACTIVIDAD
_DESCRIPCION" class="form-control" value="">{{$elemento-
>ACTIVIDAD_DESCRIPCION}}</textarea>

                    </div>

                    <div class="form-group">

                        <label for="id">Fecha inicio:</label>

                        <input type="date" id="HORARIO_FECHA_INICIO" name="HOR
ARIO_FECHA_INICIO" value="{{$elemento->HORARIO_FECHA_INICIO}}">

                    </div>

                    <div class="form-group">

                        <label for="id">Hora:</label><br/>

                        <input type="time" id="HORARIO_HORA" name="HORARIO_H
ORA" value="{{$elemento->HORARIO_HORA}}">

                    </div>

                    <div class="form-group">

                        <label for="id">Fecha fin:</label><br/>
```

```
                        <input type="date" id="HORARIO_FECHA_FIN" name="HORARI
O_FECHA_FIN" value="{{$elemento->HORARIO_FECHA_FIN}}">
                    </div>
                    <div class="form-group">
                        <label>Activo:</label><br>
                        @if($elemento->ACTIVIDAD_ACTIVO == 1)
                            <input type=checkbox class="custom-
checkbox" id="CB{{$elemento-
>ACTIVIDAD_ACTIVO}}" name="ACTIVIDAD_ACTIVO" checked>
                        @else
                            <input type=checkbox class="custom-
checkbox" id="CB{{$elemento-
>ACTIVIDAD_ACTIVO}}" name="ACTIVIDAD_ACTIVO">
                        @endif
                    </div>
                </div>
                <div class="modal-footer">
                    <input type="button" class="btn btn-default" data-
dismiss="modal" value="Cancelar">
                    <input type="submit" class="btn btn-info" value="Guardar">
```

```
                </div>
            </form>
        </div>
    </div>
</div>


                <td>
                    <a href="#" class="edit" data-toggle="modal" data-
target="#editModal-{{$elemento->ACTIVIDAD_ID}}"><i class="material-icons" data-
toggle="tooltip"
                    title="Edit">&#xE254;</i></a>
                </td>
            </tr>
        </tbody>


        @endforeach
        </table>
</div>
    </div>
```

```
    </div>
```

carreraXuniView.balde.php

```
    @extends('layouts.app')


    @section('title', 'Adm. Clientes')


    @section('content')
    @if (session('alert'))
        <div class="alert alert-success" id="mensaje">
            {{ session('alert') }}


        </div>
    @endif
        <div class="container">
        <form class="form-inline d-flex justify-content-center md-form form-
sm searchbox">
            <input class="form-control form-control-sm mr-3 w-
75" type="text" placeholder="Search" id="myInput" name="myInput"
```

```
        aria-label="Search" onkeyup="searchFunction()">
</form>

<script type="text/javascript">
function searchFunction(){
    // Variables
    var input, filter, table, tr, td, i, txtValue;
    input = document.getElementById("myInput");
    filter = input.value.toUpperCase();
    table = document.getElementById("myTable");
    tr = table.getElementsByTagName("tr");


    for (i = 0; i < tr.length; i++) {
        td = tr[i].getElementsByTagName("td")[2];
        if (td) {
            txtValue = td.textContent || td.innerText;
            if (txtValue.toUpperCase().indexOf(filter) > -1) {
```

```
                tr[i].style.display = "";
            } else {
                tr[i].style.display = "none";
            }
        }
    }
}
</script>
    <div class="table-wrapper">
        <div class="table-title">
            <div class="row">
                <div class="col-sm-6">
                    <h2>Administrar <b>Carreras</b></h2>
                </div>
                <div class="col-sm-6">
                    <a href="#addModal" class="btn btn-success" data-
toggle="modal"><i class="material-
icons">&#xE147;</i> <span>Nueva Carrera</span></a>
                </div>
```

```
            </div>
        </div>
        <!-- Add Modal HTML -->
    <div id="addModal" class="modal fade">
        <div class="modal-dialog">
            <div class="modal-content">
                <form action="/carreraCreate" method="get">
                    <div class="modal-header">
                        <h4 class="modal-title">Crear</h4>
                        <button type="button" class="close" data-dismiss="modal" aria-
hidden="true">&times;</button>
                    </div>
                    <div class="modal-body">
                        <div class="form-group" hidden>
                            <label for="id">ID:</label>
                            <input type="text" id="CARRERA_ID" class="form-control">
                        </div>
                        <div class="form-group">
                            <label for="descripcion">Nombre:</label>
```

```
                        <input type="text" id="CARRERA_NOMBRE" name="CARRERA
_NOMBRE" class="form-control" required>
                    </div>
                    <div class="form-group">
                        <label>Activo:</label><br>
                        <input type="checkbox" class="custom-
checkbox" id="CARRERA_ACTIVA" name="CARRERA_ACTIVA" disabled="true" v
alue="1" checked>
                    </div>
                </div>
                <div class="modal-footer">
                    <input type="button" class="btn btn-default" data-
dismiss="modal" value="Cancelar">
                    <input type="submit" class="btn btn-success" value="Agregar">
                </div>
            </form>
        </div>
    </div>
</div>
```

```
<table class="table table-striped table-hover" id="myTable">
    <thead>
        <tr>
            <th>
                <span class="custom-checkbox">
                    <input type="checkbox" id="selectAll">
                    <label for="selectAll"></label>
                </span>
            </th>
            <th>ID</th>
            <th>Nombre</th>
            <th>Activo</th>
            <th>Universidades</th>
        </tr>
    </thead>
@foreach($lista as $elemento)
    <tbody>
        <tr>
```

```
                    <td>
                        <span class="custom-checkbox">
                            <input type="checkbox" id="checkbox1" name="options[]" value
="1">
                            <label for="checkbox1"></label>
                        </span>
                    </td>
                    <td>{{$elemento->CARRERA_ID}}</td>
                    <td>{{$elemento->CARRERA_NOMBRE}}</td>
                    @if($elemento->CARRERA_ACTIVA == 1)
                        <td><input type=checkbox class="custom-
checkbox" id="CB{{$elemento-
>CARRERA_ACTIVA}}" checked disabled="true"></td>
                    @else
                        <td><input type=checkbox class="custom-
checkbox" id="CB{{$elemento->CARRERA_ACTIVA}}" disabled="true"></td>
                    @endif
                    </td>
```

```
    <!-- Edit Modal HTML -->
    <div id="editModal-{{$elemento->CARRERA_ID}}" class="modal fade">
        <div class="modal-dialog">
            <div class="modal-content">
                <form action="/carreraUpdate/{{$elemento->CARRERA_ID}}" method="get" id="forma">
                    <div class="modal-header">
                        <h4 class="modal-title">Editar</h4>
                        <button type="button" class="close" data-dismiss="modal" aria-hidden="true">&times;</button>
                    </div>
                    <div class="modal-body">
                        <div class="form-group">
                            <label for="id">ID:</label>
                            <input type="text" name="CARRERA_ID" id="CARRERA_ID" class="form-control" value="{{$elemento->CARRERA_ID}}" disabled="true" required>
                        </div>
                        <div class="form-group">
                            <label for="descripcion">Nombre:</label>
```

```
                        <input type="text" id="CARRERA_NOMBRE" name="CARRERA
_NOMBRE" class="form-control" value="{{$elemento-
>CARRERA_NOMBRE}}" required>
                    </div>
                    <div class="form-group">
                        <label>Activo:</label><br>
                        @if($elemento->CARRERA_ACTIVA == 1)
                            <input type=checkbox class="custom-
checkbox" id="CB{{$elemento-
>CARRERA_ACTIVA}}" name="CARRERA_ACTIVA" checked>
                        @else
                            <input type=checkbox class="custom-
checkbox" id="CB{{$elemento-
>CARRERA_ACTIVA}}" name="CARRERA_ACTIVA">
                        @endif
                    </div>
                </div>
                <div class="modal-footer">
                    <input type="button" class="btn btn-default" data-
dismiss="modal" value="Cancelar">
```

```
                    <input type="submit" class="btn btn-info" value="Guardar">
                </div>
            </form>
        </div>
    </div>
</div>

<!-- University Modal HTML -->
<div id="universityModal-{{$elemento->CARRERA_ID}}" class="modal fade">
    <div class="modal-dialog">
        <div class="modal-content">
            <form action="/carreraXuniCreate/{{$elemento->CARRERA_ID}}" method="get">
                <div class="modal-header">
                    <h4 class="modal-title">Universidades</h4>
                    <button type="button" class="close" data-dismiss="modal" aria-hidden="true">&times;</button>
                </div>
```

```
                    <div class="modal-body">
                        <div class="form-group">
                            <label for="id">ID:</label>
                            <input type="text" name="CARRERAUNI_ID" id="CARRERAUNI
_ID" class="form-control" value="{{$elemento-
>CARRERA_ID}}" disabled="true" required>
                        </div>
                        <div class="form-group">
                            <label for="nombre">Carrera:</label>
                            <input type="text" name="CARRERA_NOMBRE" id="CARRERA
_NOMBRE" class="form-control" value="{{$elemento-
>CARRERA_NOMBRE}}" disabled="true" required>
                        </div>
                        <div class="form-group">
                            <label for="id">Universidades:</label>
                            <select name="UNIVERSIDAD_ID" id="UNIVERSIDAD_ID">
                            @foreach($listaUniversidades as $universidad)
                                <option value="{{$universidad-
>UNIVERSIDAD_ID}}">{{$universidad->UNIVERSIDAD_NOMBRE}}</option>
```

```
                    @endforeach
                    </select>
                </div>
                <div class="form-group">
                    <input type="submit" class="btn btn-
info" value="Agregar Universidad">
                </div>
                <div class="form-group">
                    <ul>
                    @foreach($listaCarreraXuni as $carreraXuni)
                        @if($carreraXuni->CARRERA_ID == $elemento-
>CARRERA_ID)
                        <li>{{$carreraXuni->UNIVERSIDAD_NOMBRE}}</li>
                        @endif
                    @endforeach
                    </ul>
                </div>
            </div>
```

```
                    <div class="modal-footer">
                        <input type="button" class="btn btn-info" data-
dismiss="modal" value="Cerrar">
                    </div>
                </form>
            </div>
        </div>
    </div>

                    <td>
                        <a href="#" class="delete" data-toggle="modal" data-
target="#universityModal-{{$elemento->CARRERA_ID}}">
                        <input type="button" class="btn btn-info" data-
dismiss="modal" value="+">
                    </td>
                    <td>
                        <a href="#" class="edit" data-toggle="modal" data-
target="#editModal-{{$elemento->CARRERA_ID}}"><i class="material-icons" data-
toggle="tooltip"
```

```
                        title="Edit">&#xE254;</i></a>
                    </td>
                </tr>
            </tbody>

            @endforeach
            </table>
        </div>
        </div>
    </div>
```

ciudadView.blade.php

```
@extends('layouts.app')

@section('title', 'Adm. Clientes')

@section('content')
<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.4/jquery.min.js"></scrip
t>
```

```
<div class="container">

<form class="form-inline d-flex justify-content-center md-form form-
sm searchbox">

    <input class="form-control form-control-sm mr-3 w-
75" type="text" placeholder="Search" id="myInput" name="myInput"

        aria-label="Search" onkeyup="searchFunction()">

</form>



<script type="text/javascript">

function searchFunction(){

    // Variables

    var input, filter, table, tr, td, i, txtValue;

    input = document.getElementById("myInput");

    filter = input.value.toUpperCase();

    table = document.getElementById("myTable");

    tr = table.getElementsByTagName("tr");



    for (i = 0; i < tr.length; i++) {

        td = tr[i].getElementsByTagName("td")[4];
```

```
        if (td) {
            txtValue = td.textContent || td.innerText;
            if (txtValue.toUpperCase().indexOf(filter) > -1) {
                tr[i].style.display = "";
            } else {
                tr[i].style.display = "none";
            }
        }
    }
}
</script>
    <div class="table-wrapper">
        <div class="table-title">
            <div class="row">
                <div class="col-sm-6">
                    <h2>Administrar <b>Ciudades</b></h2>
                </div>
                <div class="col-sm-6">
```

```
                    <a href="#addModal" class="btn btn-success" data-
toggle="modal"><i class="material-
icons">&#xE147;</i> <span>Nueva Ciudad</span></a>
                </div>
            </div>
        </div>
        <!-- Add Modal HTML -->
    <div id="addModal" class="modal fade">
        <div class="modal-dialog">
            <div class="modal-content">
                <form action="/ciudadCreate" method="get" id="form">
                    <div class="modal-header">
                        <h4 class="modal-title">Crear</h4>
                        <button type="button" class="close" data-dismiss="modal" aria-
hidden="true">&times;</button>
                    </div>
                    <div class="modal-body">
                        <div class="form-group" hidden>
                            <label for="id">ID:</label>
```

```
                    <input type="text" id="CIUDAD_ID" class="form-control">
                </div>
                <div class="form-group">
                    <label for="id">Pais:</label>
                    <select name="PAIS" id="PAIS">
                    <option value="0" >- Seleccione -</option>
                    @foreach($listaPaises as $pais)
                        <option value="{{$pais->PAIS_ID}}" >{{$pais->PAIS_NOMBRE}}</option>
                    @endforeach
                    </select>
                </div>
                <div class="form-group">
                    <label for="id">Estado:</label>
                    <select name="ESTADO_ID" id="ESTADO" required>
                    </select>
                </div>
    <script type="text/javascript">
    $(document).ready(function(){
```

```
    //cuando se cambie el combo de pais
    $('#PAIS').on('change', function() {
        var pais_id=$('#PAIS').val();
    $.ajax({
        url: "estadoXpais/"+pais_id,
        type: "GET",
        dataType: "json",
        success: function(respuesta){
            $('#ESTADO').empty();
            $('#CIUDAD').empty();
          respuesta.forEach(element => {
            $('#ESTADO').append('<option value='+element.ESTADO_ID+'> '+element.ES
TADO_NOMBRE+' </option>')
          });
        }
      });
    });


    });
```

```
    </script>
                    <div class="form-group">
                        <label for="descripcion">Nombre:</label>
                        <input type="text" id="CIUDAD_NOMBRE" name="CIUDAD_NOMBRE" class="form-control" required>
                    </div>
                    <div class="form-group">
                        <label>Activo:</label><br>
                        <input type="checkbox" class="custom-checkbox" id="CIUDAD_ACTIVO" name="CIUDAD_ACTIVO" value="1" disabled="true" checked>
                    </div>
                </div>
                <div class="modal-footer">
                    <input type="button" class="btn btn-default" data-dismiss="modal" value="Cancelar">
                    <input type="submit" class="btn btn-success" value="Agregar">
                </div>
            </form>
```

```
        </div>
    </div>
</div>
        <table class="table table-striped table-hover" id="myTable">
            <thead>
                <tr>
                    <th>
                        <span class="custom-checkbox">
                            <input type="checkbox" id="selectAll">
                            <label for="selectAll"></label>
                        </span>
                    </th>
                    <th>Pais</th>
                    <th>Estado</th>
                    <th>ID</th>
                    <th>Nombre</th>
                    <th>Activo</th>
                </tr>
```

```
            </thead>
        @foreach($lista as $elemento)
            <tbody>
                <tr>
                    <td>
                        <span class="custom-checkbox">
                            <input type="checkbox" id="checkbox1" name="options[]" value
="1">
                            <label for="checkbox1"></label>
                        </span>
                    </td>
                    <td>{{$elemento->PAIS_NOMBRE}}</td>
                    <td>{{$elemento->ESTADO_NOMBRE}}</td>
                    <td>{{$elemento->CIUDAD_ID}}</td>
                    <td>{{$elemento->CIUDAD_NOMBRE}}</td>
                    @if($elemento->CIUDAD_ACTIVO == 1)
                        <td><input type=checkbox class="custom-
checkbox" id="CB{{$elemento->CIUDAD_ACTIVO}}" checked disabled="true"></td>
                    @else
```

```
                        <td><input type=checkbox class="custom-
checkbox" id="CB{{$elemento->CIUDAD_ACTIVO}}" disabled="true"></td>
                        @endif
                        </td>

    <!-- Edit Modal HTML -->
    <div id="editModal-{{$elemento->CIUDAD_ID}}" class="modal fade">
        <div class="modal-dialog">
            <div class="modal-content">
                <form action="/ciudadUpdate/{{$elemento-
>CIUDAD_ID}}" method="get" id="forma">
                    <div class="modal-header">
                        <h4 class="modal-title">Editar</h4>
                        <button type="button" class="close" data-dismiss="modal" aria-
hidden="true">&times;</button>
                    </div>
                    <div class="modal-body">
                        <div class="form-group">
                            <label for="id">ID:</label>
```

```
                    <input type="text" name="CIUDAD_ID" id="CIUDAD_ID" class="
form-control" value="{{$elemento->CIUDAD_ID}}" disabled="true" required>
                </div>
                <div class="form-group">
                    <label for="piso">Pais:</label>
                    <input type="text" id="PAIS_NOMBRE" name="PAIS_NOMBRE"
class="form-control" value="{{$elemento-
>PAIS_NOMBRE}}" disabled="true" required>
                </div>
                <div class="form-group">
                    <label for="piso">Estado:</label>
                    <input type="text" id="ESTADO_NOMBRE" name="ESTADO_N
OMBRE" class="form-control" value="{{$elemento-
>ESTADO_NOMBRE}}" disabled="true" required>
                </div>
                <div class="form-group">
                    <label for="descripcion">Nombre:</label>
                    <input type="text" id="CIUDAD_NOMBRE" name="CIUDAD_NO
MBRE" class="form-control" value="{{$elemento->CIUDAD_NOMBRE}}" required>
                </div>
```

```
                    <div class="form-group">
                        <label>Activo:</label><br>
                        @if($elemento->CIUDAD_ACTIVO == 1)
                            <input type=checkbox class="custom-
checkbox" id="CB{{$elemento-
>CIUDAD_ACTIVO}}" name="CIUDAD_ACTIVO" checked>
                        @else
                            <input type=checkbox class="custom-
checkbox" id="CB{{$elemento->CIUDAD_ACTIVO}}" name="CIUDAD_ACTIVO">
                        @endif
                    </div>
                </div>
                <div class="modal-footer">
                    <input type="button" class="btn btn-default" data-
dismiss="modal" value="Cancelar">
                    <input type="submit" class="btn btn-info" value="Guardar">
                </div>
            </form>
        </div>
```

```
    </div>
  </div>

                <td>
                    <a href="#" class="edit" data-toggle="modal" data-
target="#editModal-{{$elemento->CIUDAD_ID}}"><i class="material-icons" data-
toggle="tooltip"
                    title="Edit">&#xE254;</i></a>
                </td>
              </tr>
            </tbody>

            @endforeach
            </table>
  </div>
    </div>
  </div>
```

colegioView.blade.php

```
@extends('layouts.app')

@section('title', 'Adm. Clientes')

@section('content')
<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.4/jquery.min.js"></scrip
t>
    <div class="container">
    <form class="form-inline d-flex justify-content-center md-form form-
sm searchbox">
        <input class="form-control form-control-sm mr-3 w-
75" type="text" placeholder="Search" id="myInput" name="myInput"
            aria-label="Search" onkeyup="searchFunction()">
    </form>

    <script type="text/javascript">
    function searchFunction(){
        // Variables
        var input, filter, table, tr, td, i, txtValue;
```

```
input = document.getElementById("myInput");

filter = input.value.toUpperCase();

table = document.getElementById("myTable");

tr = table.getElementsByTagName("tr");




for (i = 0; i < tr.length; i++) {

   td = tr[i].getElementsByTagName("td")[5];

   if (td) {

      txtValue = td.textContent || td.innerText;

      if (txtValue.toUpperCase().indexOf(filter) > -1) {

         tr[i].style.display = "";

      } else {

         tr[i].style.display = "none";

      }

   }

   td2 = tr[i].getElementsByTagName("td")[3];

   if (td2) {
```

```
            txtValue = td2.textContent || td2.innerText;
            if (txtValue.toUpperCase().indexOf(filter) > -1) {
                tr[i].style.display = "";
            } else {
                tr[i].style.display = "none";
            }
        }
    }
}
</script>
    <div class="table-wrapper">
        <div class="table-title">
            <div class="row">
                <div class="col-sm-6">
                    <h2>Administrar <b>Colegios</b></h2>
                </div>
                <div class="col-sm-6">
```

```
                    <a href="#addModal" class="btn btn-success" data-
toggle="modal"><i class="material-
icons">&#xE147;</i> <span>Nuevo Colegio</span></a>
                </div>
            </div>
        </div>
        <!-- Add Modal HTML -->
    <div id="addModal" class="modal fade">
        <div class="modal-dialog">
            <div class="modal-content">
                <form action="/colegioCreate" method="get" id="form">
                    <div class="modal-header">
                        <h4 class="modal-title">Crear</h4>
                        <button type="button" class="close" data-dismiss="modal" aria-
hidden="true">&times;</button>
                    </div>
                    <div class="modal-body">
                        <div class="form-group" hidden>
                            <label for="id">ID:</label>
```

```
                    <input type="text" id="COLEGIO_ID" class="form-control">
                </div>
                <div class="form-group">
                    <label for="id">Pais:</label>
                    <select name="PAIS" id="PAIS" require>
                    <option value="0" >- Seleccione -</option>
                    @foreach($listaPaises as $pais)
                        <option value="{{$pais->PAIS_ID}}" >{{$pais->PAIS_NOMBRE}}</option>
                    @endforeach
                    </select>
                </div>
                <div class="form-group">
                    <label for="id">Estado:</label>
                    <select name="ESTADO" id="ESTADO">
                    </select>
                </div>
                <div class="form-group">
                    <label for="id">Ciudad:</label>
```

```
                        <select name="CIUDAD_ID" id="CIUDAD" required>
                        </select>
                    </div>
    <script type="text/javascript">
    $(document).ready(function(){
        //cuando se cambie el combo de pais
    $('#PAIS').on('change', function() {
        var pais_id=$('#PAIS').val();
    $.ajax({
        url: "estadoXpais/"+pais_id,
        type: "GET",
        dataType: "json",
        success: function(respuesta){
            $('#ESTADO').empty();
            $('#CIUDAD').empty();
            $('#ESTADO').append('<option value=0> '+'- Seleccione -'+' </option>')
          respuesta.forEach(element => {
            $('#ESTADO').append('<option value='+element.ESTADO_ID+'> '+element.ES
TADO_NOMBRE+' </option>')
```

```
        });
      }
    });
  });

  //cuando se cambie un combo de estado
  $('#ESTADO').on('change', function() {
    var estado_id=$('#ESTADO').val();
    $.ajax({
      url: "ciudadXestado/"+estado_id,
      type: "GET",
      dataType: "json",
      success: function(respuesta){
        $('#CIUDAD').empty();
        respuesta.forEach(element => {
          $('#CIUDAD').append('<option value='+element.CIUDAD_ID+'> '+element.C
IUDAD_NOMBRE+' </option>')
        });
      }
```

```
        });
      });
    });
    </script>
                    <div class="form-group">
                        <label for="descripcion">Nombre:</label>
                        <input type="text" id="COLEGIO_NOMBRE" name="COLEGIO_
NOMBRE" class="form-control" required>
                    </div>
                    <div class="form-group">
                        <label>Activo:</label><br>
                        <input type="checkbox" class="custom-
checkbox" id="COLEGIO_ACTIVO" name="COLEGIO_ACTIVO" value="1" disabled
="true" checked>
                    </div>
                </div>
                <div class="modal-footer">
                    <input type="button" class="btn btn-default" data-
dismiss="modal" value="Cancelar">
```

```
                    <input type="submit" class="btn btn-success" value="Agregar">
                </div>
            </form>
        </div>
    </div>
</div>
        <table class="table table-striped table-hover" id="myTable">
            <thead>
                <tr>
                    <th>
                        <span class="custom-checkbox">
                            <input type="checkbox" id="selectAll">
                            <label for="selectAll"></label>
                        </span>
                    </th>
                    <th>Pais</th>
                    <th>Estado</th>
                    <th>Ciudad</th>
```

```
                    <th>ID</th>
                    <th>Nombre</th>
                    <th>Activo</th>
                </tr>
            </thead>
        @foreach($lista as $elemento)
            <tbody>
                <tr>
                    <td>
                        <span class="custom-checkbox">
                            <input type="checkbox" id="checkbox1" name="options[]" value
="1">
                            <label for="checkbox1"></label>
                        </span>
                    </td>
                    <td>{{$elemento->PAIS_NOMBRE}}</td>
                    <td>{{$elemento->ESTADO_NOMBRE}}</td>
                    <td>{{$elemento->CIUDAD_NOMBRE}}</td>
                    <td>{{$elemento->COLEGIO_ID}}</td>
```

```
                    <td>{{$elemento->COLEGIO_NOMBRE}}</td>

                    @if($elemento->COLEGIO_ACTIVO == 1)

                        <td><input type=checkbox class="custom-
checkbox" id="CB{{$elemento-
>COLEGIO_ACTIVO}}" checked disabled="true"></td>

                    @else

                        <td><input type=checkbox class="custom-
checkbox" id="CB{{$elemento->COLEGIO_ACTIVO}}" disabled="true"></td>

                    @endif

                    </td>



    <!-- Edit Modal HTML -->

    <div id="editModal-{{$elemento->COLEGIO_ID}}" class="modal fade">

        <div class="modal-dialog">

            <div class="modal-content">

                <form action="/colegioUpdate/{{$elemento-
>COLEGIO_ID}}" method="get" id="forma">

                    <div class="modal-header">

                        <h4 class="modal-title">Editar</h4>
```

```
                    <button type="button" class="close" data-dismiss="modal" aria-
hidden="true">&times;</button>
                </div>
                <div class="modal-body">
                    <div class="form-group">
                        <label for="id">ID:</label>
                        <input type="text" name="COLEGIO_ID" id="COLEGIO_ID" class
="form-control" value="{{$elemento->COLEGIO_ID}}" disabled="true" required>
                    </div>
                    <div class="form-group">
                        <label for="piso">Pais:</label>
                        <input type="text" id="PAIS_NOMBRE" name="PAIS_NOMBRE"
class="form-control" value="{{$elemento-
>PAIS_NOMBRE}}" disabled="true" required>
                    </div>
                    <div class="form-group">
                        <label for="piso">Estado:</label>
                        <input type="text" id="ESTADO_NOMBRE" name="ESTADO_N
OMBRE" class="form-control" value="{{$elemento-
>ESTADO_NOMBRE}}" disabled="true" required>
```

```
                    </div>
                    <div class="form-group">
                        <label for="piso">Ciudad:</label>
                        <input type="text" id="CIUDAD_NOMBRE" name="CIUDAD_NO
MBRE" class="form-control" value="{{$elemento-
>CIUDAD_NOMBRE}}" disabled="true" required>
                    </div>
                    <div class="form-group">
                        <label for="descripcion">Nombre:</label>
                        <input type="text" id="COLEGIO_NOMBRE" name="COLEGIO_
NOMBRE" class="form-control" value="{{$elemento-
>COLEGIO_NOMBRE}}" required>
                    </div>
                    <div class="form-group">
                        <label>Activo:</label><br>
                        @if($elemento->COLEGIO_ACTIVO == 1)
                            <input type=checkbox class="custom-
checkbox" id="CB{{$elemento-
>COLEGIO_ACTIVO}}" name="COLEGIO_ACTIVO" checked>
                        @else
```

```
                        <input type=checkbox class="custom-
checkbox" id="CB{{$elemento-
>COLEGIO_ACTIVO}}" name="COLEGIO_ACTIVO">
                        @endif
                    </div>
                </div>
                <div class="modal-footer">
                    <input type="button" class="btn btn-default" data-
dismiss="modal" value="Cancelar">
                    <input type="submit" class="btn btn-info" value="Guardar">
                </div>
            </form>
        </div>
    </div>
</div>

                    <td>
```

COLEGIO_ID}}"><i class="material-icons" data-toggle="tooltip"

title="Edit"></i>

</td>

</tr>

</tbody>

@endforeach

</table>

cuartoView.blade.php

@extends('layouts.app')

@section('title', 'Adm. Clientes')

```
@section('content')
    <div class="container">
        <div class="table-wrapper">
            <div class="table-title">
                <div class="row">
                    <div class="col-sm-6">
                        <h2>Administrar <b>Habitaciones</b></h2>
                    </div>
                    <div class="col-sm-6">
                        <a href="#addModal" class="btn btn-success" data-
toggle="modal"><i class="material-
icons">&#xE147;</i> <span>Nueva Habitacion</span></a>
                    </div>
                </div>
            </div>
            <!-- Add Modal HTML -->
        <div id="addModal" class="modal fade">
            <div class="modal-dialog">
                <div class="modal-content">
```

```
<form action="/cuartoCreate" method="get" id="form">
    <div class="modal-header">
        <h4 class="modal-title">Crear</h4>
        <button type="button" class="close" data-dismiss="modal" aria-
hidden="true">&times;</button>
    </div>
    <div class="modal-body">
        <div class="form-group" hidden>
            <label for="id">ID:</label>
            <input type="text" id="CUARTO_ID" class="form-control">
        </div>
        <div class="form-group">
            <label for="id">Piso:</label>
            <select name="PISO_ID" id="PISO_ID">
            @foreach($listaPisos as $elemento)
                <option value="{{$elemento-
>PISO_ID}}" name="PISO_ID">{{$elemento->PISO_NUMERO}}</option>
            @endforeach
            </select>
```

```
                    </div>
                    <div class="form-group">
                        <label for="descripcion">Número:</label>
                        <input type="number" id="CUARTO_NUMERO" name="CUARTO_NUMERO" class="form-control" required>
                    </div>
                    <div class="form-group">
                        <label>Activo:</label><br>
                        <input type="checkbox" class="custom-checkbox" id="CUARTO_ACTIVO" name="CUARTO_ACTIVO" value="1" disabled="true" checked>
                    </div>
                </div>
                <div class="modal-footer">
                    <input type="button" class="btn btn-default" data-dismiss="modal" value="Cancelar">
                    <input type="submit" class="btn btn-success" value="Agregar">
                </div>
            </form>
```

```
        </div>
    </div>
</div>
        <table class="table table-striped table-hover">
            <thead>
                <tr>
                    <th>
                        <span class="custom-checkbox">
                            <input type="checkbox" id="selectAll">
                            <label for="selectAll"></label>
                        </span>
                    </th>
                    <th>Piso</th>
                    <th>ID</th>
                    <th>Número</th>
                    <th>Activo</th>
                </tr>
            </thead>
```

```
        @foreach($lista as $elemento)
            <tbody>
                <tr>
                    <td>
                        <span class="custom-checkbox">
                            <input type="checkbox" id="checkbox1" name="options[]" value
="1">
                            <label for="checkbox1"></label>
                        </span>
                    </td>
                    <td>{{$elemento->PISO_NUMERO}}</td>
                    <td>{{$elemento->CUARTO_ID}}</td>
                    <td>{{$elemento->CUARTO_NUMERO}}</td>
                    @if($elemento->CUARTO_ACTIVO == 1)
                        <td><input type=checkbox class="custom-
checkbox" id="CB{{$elemento-
>CUARTO_ACTIVO}}" checked disabled="true"></td>
                    @else
```

```
                        <td><input type=checkbox class="custom-
checkbox" id="CB{{$elemento->CUARTO_ACTIVO}}" disabled="true"></td>
                        @endif
                        </td>

    <!-- Edit Modal HTML -->
    <div id="editModal-{{$elemento->CUARTO_ID}}" class="modal fade">
        <div class="modal-dialog">
            <div class="modal-content">
                <form action="/cuartoUpdate/{{$elemento-
>CUARTO_ID}}" method="get" id="forma">
                    <div class="modal-header">
                        <h4 class="modal-title">Editar</h4>
                        <button type="button" class="close" data-dismiss="modal" aria-
hidden="true">&times;</button>
                    </div>
                    <div class="modal-body">
                        <div class="form-group">
                            <label for="id">ID:</label>
```

```
                    <input type="text" name="PISO_ID" id="CUARTO_ID" class="for
m-control" value="{{$elemento->CUARTO_ID}}" disabled="true" required>
                </div>
                <div class="form-group">
                    <label for="piso">Piso:</label>
                    <input type="text" id="PISO_NUMERO" name="PISO_NUMERO"
 class="form-control" value="{{$elemento-
>PISO_NUMERO}}" disabled="true" required>
                </div>
                <div class="form-group">
                    <label for="descripcion">Número:</label>
                    <input type="number" id="CUARTO_NUMERO" name="CUART
O_NUMERO" class="form-control" value="{{$elemento-
>CUARTO_NUMERO}}" required>
                </div>
                <div class="form-group">
                    <label>Activo:</label><br>
                    @if($elemento->CUARTO_ACTIVO == 1)
```

```
                        <input type=checkbox class="custom-
checkbox" id="CB{{$elemento-
>CUARTO_ACTIVO}}" name="CUARTO_ACTIVO" checked>
                        @else
                        <input type=checkbox class="custom-
checkbox" id="CB{{$elemento-
>CUARTO_ACTIVO}}" name="CUARTO_ACTIVO">
                        @endif
                    </div>
                </div>
                <div class="modal-footer">
                    <input type="button" class="btn btn-default" data-
dismiss="modal" value="Cancelar">
                    <input type="submit" class="btn btn-info" value="Guardar">
                </div>
            </form>
        </div>
    </div>
</div>
```

```
                    <td>
                        <a href="#" class="edit" data-toggle="modal" data-
target="#editModal-{{$elemento->CUARTO_ID}}"><i class="material-icons" data-
toggle="tooltip"
                        title="Edit">&#xE254;</i></a>
                    </td>
                </tr>
            </tbody>

            @endforeach
            </table>
    </div>
        </div>
    </div>
```

estadoView.blade.php

```
@extends('layouts.app')
```

```
    @section('title', 'Adm. Clientes')



    @section('content')

        <div class="container">

        <form class="form-inline d-flex justify-content-center md-form form-
sm searchbox">

            <input class="form-control form-control-sm mr-3 w-
75" type="text" placeholder="Search" id="myInput" name="myInput"

                aria-label="Search" onkeyup="searchFunction()">

        </form>



        <script type="text/javascript">

        function searchFunction(){

            // Variables

            var input, filter, table, tr, td, i, txtValue;

            input = document.getElementById("myInput");

            filter = input.value.toUpperCase();

            table = document.getElementById("myTable");

            tr = table.getElementsByTagName("tr");
```

```
    for (i = 0; i < tr.length; i++) {
        td = tr[i].getElementsByTagName("td")[3];
        if (td) {
            txtValue = td.textContent || td.innerText;
            if (txtValue.toUpperCase().indexOf(filter) > -1) {
                tr[i].style.display = "";
            } else {
                tr[i].style.display = "none";
            }
        }
    }
}
</script>

    <div class="table-wrapper">
        <div class="table-title">
            <div class="row">
```

```
                    <div class="col-sm-6">
                        <h2>Administrar <b>Estados</b></h2>
                    </div>
                    <div class="col-sm-6">
                        <a href="#addModal" class="btn btn-success" data-
toggle="modal"><i class="material-
icons">&#xE147;</i> <span>Nuevo Estado</span></a>
                    </div>
                </div>
            </div>
            <!-- Add Modal HTML -->
        <div id="addModal" class="modal fade">
            <div class="modal-dialog">
                <div class="modal-content">
                    <form action="/estadoCreate" method="get" id="form">
                        <div class="modal-header">
                            <h4 class="modal-title">Crear</h4>
                            <button type="button" class="close" data-dismiss="modal" aria-
hidden="true">&times;</button>
```

```
                </div>
                <div class="modal-body">
                    <div class="form-group" hidden>
                        <label for="id">ID:</label>
                        <input type="text" id="ESTADO_ID" class="form-control">
                    </div>
                    <div class="form-group">
                        <label for="id">Pais:</label>
                        <select name="PAIS_ID" id="PAIS_ID">
                        @foreach($listaPaises as $elemento)
                            <option value="{{$elemento-
>PAIS_ID}}" name="PAIS_ID">{{$elemento->PAIS_NOMBRE}}</option>
                        @endforeach
                        </select>
                    </div>
                    <div class="form-group">
                        <label for="descripcion">Nombre:</label>
                        <input type="text" id="ESTADO_NOMBRE" name="ESTADO_N
OMBRE" class="form-control" required>
```

```
                    </div>
                    <div class="form-group">
                        <label>Activo:</label><br>
                        <input type="checkbox" class="custom-
checkbox" id="ESTADO_ACTIVO" name="ESTADO_ACTIVO" value="1" disabled="t
rue" checked>
                    </div>
                </div>
                <div class="modal-footer">
                    <input type="button" class="btn btn-default" data-
dismiss="modal" value="Cancelar">
                    <input type="submit" class="btn btn-success" value="Agregar">
                </div>
            </form>
        </div>
    </div>
</div>
    <table class="table table-striped table-hover" id="myTable">
        <thead>
```

```
            <tr>
                <th>
                    <span class="custom-checkbox">
                        <input type="checkbox" id="selectAll">
                        <label for="selectAll"></label>
                    </span>
                </th>
                <th>Pais</th>
                <th>ID</th>
                <th>Nombre</th>
                <th>Activo</th>
            </tr>
        </thead>
@foreach($lista as $elemento)
        <tbody>
            <tr>
                <td>
                    <span class="custom-checkbox">
```

```
                        <input type="checkbox" id="checkbox1" name="options[]" value
="1">
                        <label for="checkbox1"></label>
                    </span>
                </td>
                <td>{{$elemento->PAIS_NOMBRE}}</td>
                <td>{{$elemento->ESTADO_ID}}</td>
                <td>{{$elemento->ESTADO_NOMBRE}}</td>
                @if($elemento->ESTADO_ACTIVO == 1)
                    <td><input type=checkbox class="custom-
checkbox" id="CB{{$elemento->ESTADO_ACTIVO}}" checked disabled="true"></td>
                @else
                    <td><input type=checkbox class="custom-
checkbox" id="CB{{$elemento->ESTADO_ACTIVO}}" disabled="true"></td>
                @endif
                </td>

    <!-- Edit Modal HTML -->
    <div id="editModal-{{$elemento->ESTADO_ID}}" class="modal fade">
```

```
<div class="modal-dialog">
    <div class="modal-content">
        <form action="/estadoUpdate/{{$elemento->ESTADO_ID}}" method="get" id="forma">
            <div class="modal-header">
                <h4 class="modal-title">Editar</h4>
                <button type="button" class="close" data-dismiss="modal" aria-hidden="true">&times;</button>
            </div>
            <div class="modal-body">
                <div class="form-group">
                    <label for="id">ID:</label>
                    <input type="text" name="ESTADO_ID" id="ESTADO_ID" class="form-control" value="{{$elemento->ESTADO_ID}}" disabled="true" required>
                </div>
                <div class="form-group">
                    <label for="piso">Pais:</label>
```

```
                    <input type="text" id="PAIS_NOMBRE" name="PAIS_NOMBRE"
class="form-control" value="{{$elemento-
>PAIS_NOMBRE}}" disabled="true" required>
                </div>
                <div class="form-group">
                    <label for="descripcion">Nombre:</label>
                    <input type="text" id="ESTADO_NOMBRE" name="ESTADO_N
OMBRE" class="form-control" value="{{$elemento-
>ESTADO_NOMBRE}}" required>
                </div>
                <div class="form-group">
                    <label>Activo:</label><br>
                    @if($elemento->ESTADO_ACTIVO == 1)
                        <input type=checkbox class="custom-
checkbox" id="CB{{$elemento-
>ESTADO_ACTIVO}}" name="ESTADO_ACTIVO" checked>
                    @else
                        <input type=checkbox class="custom-
checkbox" id="CB{{$elemento->ESTADO_ACTIVO}}" name="ESTADO_ACTIVO">
                    @endif
```

```
                    </div>
                </div>
                <div class="modal-footer">
                    <input type="button" class="btn btn-default" data-
dismiss="modal" value="Cancelar">
                    <input type="submit" class="btn btn-info" value="Guardar">
                </div>
            </form>
        </div>
    </div>
</div>


                    <td>
                        <a href="#" class="edit" data-toggle="modal" data-
target="#editModal-{{$elemento->ESTADO_ID}}"><i class="material-icons" data-
toggle="tooltip"
                        title="Edit">&#xE254;</i></a>
                    </td>
                </tr>
```

```
                </tbody>


                @endforeach

                </table>

        </div>

            </div>

        </div>
```

loginView.blade.php

```
<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1">

<title>Ingreso</title>

<link href="https://fonts.googleapis.com/css?family=Roboto|Varela+Round" rel="styl
esheet">
```

```
    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/font-
awesome/4.7.0/css/font-awesome.min.css">
    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/boot
strap.min.css">

    <script src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.4/jquery.min.js"></scrip
t>
    <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"></s
cript>
    <link href="{{ asset('assets/css/login.css') }}" rel="stylesheet">
    </head>
    <body>
    @if (session('alert'))
        <div class="alert alert-success" id="mensaje">
            {{ session('alert') }}

        </div>
    @endif
    <!-- Modal HTML -->
```

```
<div id="myModal" >
    <div class="modal-dialog modal-login">
        <div class="modal-content">
            <div class="modal-header">
                <div class="avatar">
                    <img src="avatar.png" alt="Avatar">
                </div>
                <h4 class="modal-title">Ingreso al sistema</h4>
                <button type="button" class="close" data-dismiss="modal" aria-
hidden="true">&times;</button>
            </div>
            <div class="modal-body">
                <form action="/autenticacion" method="get">
                    <div class="form-group">
                        <input type="text" class="form-
control" name="email" id="email" placeholder="Email" required="required">
                    </div>
                    <div class="form-group">
```

```
                    <input type="password" class="form-
control" name="password" id="password" placeholder="Contraseña" required="required
">
                </div>
                <div class="form-group">
                    <button type="submit" class="btn btn-primary btn-lg btn-block login-
btn">Ingresar</button>
                </div>
            </form>
        </div>
        <!-- <div class="modal-footer">
            <a href="#">Olvidé mi contraseña</a> -->
        </div>
    </div>
</div>
</body>
</html>
```

numerariaView.blade.php

```
@extends('layouts.app')

@section('title', 'Adm. Clientes')

@section('content')
@if (session('alert'))
    <div class="alert alert-success" id="mensaje">
        {{ session('alert') }}

    </div>
@endif
<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.4/jquery.min.js"></scrip
t>
    <div class="container">
    <form class="form-inline d-flex justify-content-center md-form form-
sm searchbox">
```

```
        <input class="form-control form-control-sm mr-3 w-
75" type="text" placeholder="Search" id="myInput" name="myInput"
            aria-label="Search" onkeyup="searchFunction()">
    </form>

    <script type="text/javascript">
    function searchFunction(){
        // Variables
        var input, filter, table, tr, td, i, txtValue;
        input = document.getElementById("myInput");
        filter = input.value.toUpperCase();
        table = document.getElementById("myTable");
        tr = table.getElementsByTagName("tr");

        for (i = 0; i < tr.length; i++) {
            td = tr[i].getElementsByTagName("td")[2];
            if (td) {
                txtValue = td.textContent || td.innerText;
                if (txtValue.toUpperCase().indexOf(filter) > -1) {
```

```
                tr[i].style.display = "";
            } else {
                tr[i].style.display = "none";
            }
        }
    }
}
</script>
    <div class="table-wrapper">
        <div class="table-title">
            <div class="row">
                <div class="col-sm-6">
                    <h2>Administrar <b>Numerarias</b></h2>
                </div>
                <div class="col-sm-6">
                    <a href="#addModal" class="btn btn-success" data-
toggle="modal"><i class="material-
icons">&#xE147;</i> <span>Nueva Numeraria</span></a>
                </div>
```

```
            </div>
        </div>
        <!-- Add Modal HTML -->
    <div id="addModal" class="modal fade">
        <div class="modal-dialog">
            <div class="modal-content">
                <form action="/numerariaCreate" method="get" id="form">
                    <div class="modal-header">
                        <h4 class="modal-title">Crear</h4>
                        <button type="button" class="close" data-dismiss="modal" aria-
hidden="true">&times;</button>
                    </div>
                    <div class="modal-body">
                        <div class="form-group" hidden>
                            <label for="id">ID:</label>
                            <input type="text" id="NUMERARIA_ID" class="form-control">
                        </div>
                        <div class="form-group">
                            <label for="id">Pais:</label>
```

```
                        <select name="PAIS" id="PAIS" require>
                        <option value="0" >- Seleccione -</option>
                        @foreach($listaPaises as $pais)
                            <option value="{{$pais->PAIS_ID}}" >{{$pais->PAIS_NOMBRE}}</option>
                        @endforeach
                        </select>
                    </div>
                    <div class="form-group">
                        <label for="id">Estado:</label>
                        <select name="ESTADO" id="ESTADO">
                        </select>
                    </div>
                    <div class="form-group">
                        <label for="id">Ciudad:</label>
                        <select name="CIUDAD_ID" id="CIUDAD" required>
                        </select>
                    </div>
                    <div class="form-group">
```

```
                    <label for="descripcion">CI:</label>

                    <input type="text" id="NUMERARIA_CEDULA" name="NUMER
ARIA_CEDULA" class="form-control" required>

                </div>

                <div class="form-group">

                    <label for="descripcion">Nombre:</label>

                    <input type="text" id="NUMERARIA_NOMBRE" name="NUMER
ARIA_NOMBRE" class="form-control" required>

                </div>

                <div class="form-group">

                    <label>Telefono:</label><br>

                    <input type="text" id="NUMERARIA_TELEFONO" name="NUM
ERARIA_TELEFONO" class="form-control" required>

                </div>

                <div class="form-group">

                    <label>Celular:</label><br>

                    <input type="text" id="NUMERARIA_CELULAR" name="NUME
RARIA_CELULAR" class="form-control" required>

                </div>
```

```
                    <div class="form-group">
                        <label>Mail:</label><br>
                        <input type="email" id="NUMERARIA_MAIL" name="NUMERA
RIA_MAIL" class="form-control" required>
                    </div>
                    <div class="form-group">
                        <label>Fecha Nacimiento:</label><br>
                        <input type="date" id="NUMERARIA_FECHA_NACIMIENTO" n
ame="NUMERARIA_FECHA_NACIMIENTO" class="form-control date" required>
                    </div>
                    <div class="form-group">
                        <label for="id">Universidad:</label>
                        <select name="UNIVERSIDAD_ID" id="UNIVERSIDAD" required
>
                        <option value="0" >- Seleccione -</option>
                        @foreach($listaUniversidades as $universidad)
                            <option value="{{$universidad-
>UNIVERSIDAD_ID}}" >{{$universidad->UNIVERSIDAD_NOMBRE}}</option>
                        @endforeach
```

```
                    </select>
                </div>
                <div class="form-group">
                    <label for="id">Carrera:</label>
                    <select name="CARRERA_ID" id="CARRERA" required>
                    </select>
                </div>
                <div class="form-group">
                    <label for="id">Ocupacion:</label>
                    <select name="OCUPACION_ID" id="OCUPACION_ID" required
>
                    @foreach($listaOcupaciones as $ocupacion)
                        <option value="{{$ocupacion-
>OCUPACION_ID}}" >{{$ocupacion->OCUPACION_DESCRIPCION}}</option>
                    @endforeach
                    </select>
                </div>
                <div class="form-group">
                    <label>Activo:</label><br>
```

```
                        <input type="checkbox" class="custom-
checkbox" id="NUMERARIA_ACTIVO" name="NUMERARIA_ACTIVO" disabled="t
rue" value="1" checked>
                    </div>
                </div>
                <div class="modal-footer">
                    <input type="button" class="btn btn-default" data-
dismiss="modal" value="Cancelar">
                    <input type="submit" class="btn btn-success" value="Agregar">
                </div>
            </form>
        </div>
    </div>
</div>
        <table class="table table-striped table-hover" id="myTable">
            <thead>
                <tr>
                    <th>
                        <span class="custom-checkbox">
```

```
                        <input type="checkbox" id="selectAll">
                        <label for="selectAll"></label>
                    </span>
                </th>
                <th>ID</th>
                <th>Nombre</th>
                <th>Celular</th>
                <th>Ciudad</th>
                <th>Activo</th>
                <th>Habitacion</th>
            </tr>
        </thead>
    @foreach($lista as $elemento)
        <tbody>
            <tr>
                <td>
                    <span class="custom-checkbox">
                        <input type="checkbox" id="checkbox1" name="options[]" value
="1">
```

```
                        <label for="checkbox1"></label>
                    </span>
                </td>
                <td>{{$elemento->NUMERARIA_ID}}</td>
                <td>{{$elemento->NUMERARIA_NOMBRE}}</td>
                <td>{{$elemento->NUMERARIA_CELULAR}}</td>
                <td>{{$elemento->CIUDAD_NOMBRE}}</td>
                @if($elemento->NUMERARIA_ACTIVO == 1)
                    <td><input type=checkbox class="custom-
checkbox" id="CB{{$elemento-
>NUMERARIA_ACTIVO}}" checked disabled="true"></td>
                @else
                    <td><input type=checkbox class="custom-
checkbox" id="CB{{$elemento->NUMERARIA_ACTIVO}}" disabled="true"></td>
                @endif
                </td>

    <!-- Edit Modal HTML -->
    <div id="editModal-{{$elemento->NUMERARIA_ID}}" class="modal fade">
```

```
        <div class="modal-dialog">
            <div class="modal-content">
                <form action="/numerariaUpdate/{{$elemento->NUMERARIA_ID}}" method="get" id="forma">
                    <div class="modal-header">
                        <h4 class="modal-title">Editar</h4>
                        <button type="button" class="close" data-dismiss="modal" aria-hidden="true">&times;</button>
                    </div>
                    <div class="modal-body">
                        <div class="form-group">
                            <label for="id">ID:</label>
                            <input type="text" name="NUMERARIA_ID" id="NUMERARIA_ID" class="form-control" value="{{$elemento->NUMERARIA_ID}}" disabled="true" required>
                        </div>
                        <div class="form-group">
                            <label for="descripcion">CI:</label>
```

```
                    <input type="text" id="NUMERARIA_CEDULA" name="NUMER
ARIA_CEDULA" class="form-control" value="{{$elemento-
>NUMERARIA_CEDULA}}" required>
                </div>
                <div class="form-group">
                    <label for="descripcion">Nombre:</label>
                    <input type="text" id="NUMERARIA_NOMBRE" name="NUMER
ARIA_NOMBRE" class="form-control" value="{{$elemento-
>NUMERARIA_NOMBRE}}" required>
                </div>
                <div class="form-group">
                    <label>Ciudad:</label><br>
                    <input type="text" id="CIUDAD_NOMBRE" name="CIUDAD_NO
MBRE" class="form-control" value="{{$elemento->CIUDAD_NOMBRE}}" disabled>
                </div>
                <div class="form-group">
                    <label>Telefono:</label><br>
                    <input type="text" id="NUMERARIA_TELEFONO" name="NUM
ERARIA_TELEFONO" class="form-control" value="{{$elemento-
>NUMERARIA_TELEFONO}}" required>
```

```
                </div>

                <div class="form-group">

                    <label>Celular:</label><br>

                    <input type="text" id="NUMERARIA_CELULAR" name="NUME
RARIA_CELULAR" class="form-control" value="{{$elemento-
>NUMERARIA_CELULAR}}" required>

                </div>

                <div class="form-group">

                    <label>Mail:</label><br>

                    <input type="email" id="NUMERARIA_MAIL" name="NUMERA
RIA_MAIL" class="form-control" value="{{$elemento-
>NUMERARIA_MAIL}}" required>

                </div>

                <div class="form-group">

                    <label>Fecha Nacimiento:</label><br>

                    <input type="date" id="NUMERARIA_FECHA_NACIMIENTO" n
ame="NUMERARIA_FECHA_NACIMIENTO" class="form-
control" value="{{$elemento->NUMERARIA_FECHA_NACIMIENTO}}" required>

                </div>

                <div class="form-group">
```

```
                    <label for="id">Universidad:</label>

                    <select name="UNIVERSIDAD_ID" id="{{$elemento-
>NUMERARIA_ID}}" class="comboUniversidad" onchange="myFunction(this)" requir
ed>

                    @foreach($listaUniversidades as $universidad)

                        @if($universidad->UNIVERSIDAD_ID == $elemento-
>UNIVERSIDAD_ID)

                        <option value="{{$universidad-
>UNIVERSIDAD_ID}}" selected>{{$universidad-
>UNIVERSIDAD_NOMBRE}}</option>

                        @else

                        <option value="{{$universidad-
>UNIVERSIDAD_ID}}">{{$universidad->UNIVERSIDAD_NOMBRE}}</option>

                        @endif

                    @endforeach

                    </select>

                </div>

                <script type="text/javascript">

                    function myFunction(select){

                        var selected = parseInt(select.value);
```

```
                    var nombreComboCarrera = "#CARRERA-"+select.id;
                    //alert("idSeleccion "+selected);
                    //alert("idCombo "+nombreComboCarrera);
                    $.ajax({
                        url: "carrerasXuniversidad/"+selected,
                        type: "GET",
                        dataType: "json",
                        success: function(respuesta){
                            //alert("entro al ajax");
                            $(nombreComboCarrera).empty();
                        respuesta.forEach(element => {
                            //alert(element.CARRERA_NOMBRE);
                            $(nombreComboCarrera).append('<option value='+element.C
ARRERA_ID+'> '+element.CARRERA_NOMBRE+' </option>')
                        });
                        }
                    });
                }
            </script>
```

```
                    <div class="form-group">
                        <label for="id">Carrera:</label>
                        <select name="CARRERA_ID" id="CARRERA-{{$elemento->NUMERARIA_ID}}" required>
                        @foreach($listaCarreras as $carrera)
                            @if($carrera->UNIVERSIDAD_ID == $elemento->UNIVERSIDAD_ID)
                            @if($carrera->CARRERA_ID == $elemento->CARRERA_ID)
                            <option value="{{$carrera->CARRERA_ID}}" selected>{{$carrera->CARRERA_NOMBRE}}</option>
                            @else
                            <option value="{{$carrera->CARRERA_ID}}">{{$carrera->CARRERA_NOMBRE}}</option>
                            @endif
                            @endif
                        @endforeach
                        </select>
                    </div>
                    <div class="form-group">
```

```
                    <label for="id">Ocupacion:</label>
                    <select name="OCUPACION_ID" id="OCUPACION_ID" required
>
                    @foreach($listaOcupaciones as $ocupacion)
                        @if($ocupacion->OCUPACION_ID == $elemento-
>OCUPACION_ID)
                        <option value="{{$ocupacion-
>OCUPACION_ID}}" selected>{{$ocupacion-
>OCUPACION_DESCRIPCION}}</option>
                        @else
                        <option value="{{$ocupacion-
>OCUPACION_ID}}">{{$ocupacion->OCUPACION_DESCRIPCION}}</option>
                        @endif
                    @endforeach
                    </select>
                </div>
                <div class="form-group">
                    <label>Activo:</label><br>
                    @if($elemento->NUMERARIA_ACTIVO == 1)
```

```
                        <input type=checkbox class="custom-
checkbox" id="CB{{$elemento-
>NUMERARIA_ACTIVO}}" name="NUMERARIA_ACTIVO" checked>
                    @else
                        <input type=checkbox class="custom-
checkbox" id="CB{{$elemento-
>NUMERARIA_ACTIVO}}" name="NUMERARIA_ACTIVO">
                    @endif
                </div>
            </div>
            <div class="modal-footer">
                <input type="button" class="btn btn-default" data-
dismiss="modal" value="Cancelar">
                <input type="submit" class="btn btn-info" value="Guardar">
            </div>
        </form>
    </div>
</div>
</div>
```

```
        <!-- Piso Modal HTML -->
    <div id="roomModal-{{$elemento->NUMERARIA_ID}}" class="modal fade">
        <div class="modal-dialog">
            <div class="modal-content">
                <form action="/numerariaUpdateRoom/{{$elemento->NUMERARIA_ID}}" method="get" id="forma">
                    <div class="modal-header">
                        <h4 class="modal-title">Habitacion</h4>
                        <button type="button" class="close" data-dismiss="modal" aria-hidden="true">&times;</button>
                    </div>
                    <div class="modal-body">
                        <div class="form-group">
                            <label for="id">Piso:</label>
                            <select id="PISO{{$elemento->NUMERARIA_ID}}" class="comboPisos" onchange="roomFunction(this)" required>
                            @if($elemento->CUARTO_ID == '')
                                <option>- Seleccione -</option>
                            @endif
```

```
                    @foreach($listaPisos as $piso)
                    @if($piso->CUARTO_ID == $elemento->CUARTO_ID)
                        <option value="{{$piso->PISO_ID}}" selected>{{$piso->PISO_NUMERO}}</option>
                    @else
                        <option value="{{$piso->PISO_ID}}">{{$piso->PISO_NUMERO}}</option>
                    @endif
                    @endforeach
                    </select>
                </div>
                <div class="form-group">
                    <label for="id">Habitacion:</label>
                    <select name="CUARTO_ID" id="CUARTO-PISO{{$elemento->NUMERARIA_ID}}" required>
                    @foreach($listaPisos as $cuarto)
                    @if($cuarto->CUARTO_ID == $elemento->CUARTO_ID)
                        <option value="{{$cuarto->CUARTO_ID}}" selected>{{$cuarto->CUARTO_NUMERO}}</option>
```

```
    @endif
    @endforeach
    </select>
</div>
<script type="text/javascript">
    function roomFunction(select){
        var selected = parseInt(select.value);
        var nombreComboCuarto = "#CUARTO-"+select.id;
        //alert("idSeleccion "+selected);
        //alert("idCombo "+nombreComboCuarto);
        $.ajax({
            url: "cuartoXpiso/"+selected,
            type: "GET",
            dataType: "json",
            success: function(respuesta){
                //alert("entro al ajax");
                $(nombreComboCuarto).empty();
            respuesta.forEach(element => {
```

```
                                //alert(element.CARRERA_NOMBRE);

                                $(nombreComboCuarto).append('<option value='+element.C
UARTO_ID+'> '+element.CUARTO_NUMERO+' </option>')

                            });

                            }

                        });

                    }

                </script>

            </div>

            <div class="modal-footer">

                <input type="button" class="btn btn-default" data-
dismiss="modal" value="Cancelar">

                <input type="submit" class="btn btn-info" value="Guardar">

            </div>

        </form>

    </div>

</div>

</div>

                <td>
```

```
                    <a href="#" class="delete" data-toggle="modal" data-
target="#roomModal-{{$elemento->NUMERARIA_ID}}">

                    <input type="button" class="btn btn-info" data-
dismiss="modal" value="+">

                </td>

                <td>

                    <a href="#" class="edit" data-toggle="modal" data-
target="#editModal-{{$elemento->NUMERARIA_ID}}"><i class="material-
icons" data-toggle="tooltip"

                    title="Edit">&#xE254;</i></a>

                </td>

            </tr>

        </tbody>



        @endforeach

        </table>

    </div>

    </div>

</div>
```

```
    <script type="text/javascript">
$(document).ready(function(){
//cuando se cambie el combo de pais
$('#PAIS').on('change', function() {
    var pais_id=$('#PAIS').val();
$.ajax({
    url: "estadoXpais/"+pais_id,
    type: "GET",
    dataType: "json",
    success: function(respuesta){
        $('#ESTADO').empty();
        $('#CIUDAD').empty();
        $('#ESTADO').append('<option value=0> '+'- Seleccione -'+' </option>')
      respuesta.forEach(element => {
        $('#ESTADO').append('<option value='+element.ESTADO_ID+'> '+element.ES
TADO_NOMBRE+' </option>')
      });
    }
  });
```

```
});


//cuando se cambie un combo de estado
$('#ESTADO').on('change', function() {
    var estado_id=$('#ESTADO').val();
    $.ajax({
        url: "ciudadXestado/"+estado_id,
        type: "GET",
        dataType: "json",
        success: function(respuesta){
            $('#CIUDAD').empty();
            $('#COLEGIO').empty();
          respuesta.forEach(element => {
            $('#CIUDAD').append('<option value='+element.CIUDAD_ID+'> '+element.C
IUDAD_NOMBRE+' </option>')
          });
        }
      });
    });
```

```
        $('#UNIVERSIDAD').on('change', function() {
        var universidad_id=$('#UNIVERSIDAD').val();
        $.ajax({
           url: "carrerasXuniversidad/"+universidad_id,
           type: "GET",
           dataType: "json",
           success: function(respuesta){
              $('#CARRERA').empty();
             respuesta.forEach(element => {
              $('#CARRERA').append('<option value='+element.CARRERA_ID+'> '+eleme
nt.CARRERA_NOMBRE+' </option>')
             });
           }
         });
        });
    });
    </script>
```

ocupacionView.blade.php

```
@extends('layouts.app')

@section('title', 'Adm. Clientes')

@section('content')
<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.4/jquery.min.js"></scrip
t>
    <div class="container">

    <form class="form-inline d-flex justify-content-center md-form form-
sm searchbox">
        <input class="form-control form-control-sm mr-3 w-
75" type="text" placeholder="Search" id="myInput" name="myInput"
            aria-label="Search" onkeyup="searchFunction()">
    </form>

    <script type="text/javascript">
```

```
function searchFunction(){
    // Variables
    var input, filter, table, tr, td, i, txtValue;
    input = document.getElementById("myInput");
    filter = input.value.toUpperCase();
    table = document.getElementById("myTable");
    tr = table.getElementsByTagName("tr");



    for (i = 0; i < tr.length; i++) {
        td = tr[i].getElementsByTagName("td")[2];
        if (td) {
        txtValue = td.textContent || td.innerText;
        if (txtValue.toUpperCase().indexOf(filter) > -1) {
            tr[i].style.display = "";
        } else {
            tr[i].style.display = "none";
        }
```

```
            }
        }
    }
    </script>

        <div class="table-wrapper">
            <div class="table-title">
                <div class="row">
                    <div class="col-sm-6">
                        <h2>Administrar <b>Ocupaciones</b></h2>
                    </div>
                    <div class="col-sm-6">
                        <a href="#addModal" class="btn btn-success" data-
toggle="modal"><i class="material-
icons">&#xE147;</i> <span>Nueva Ocupacion</span></a>
                    </div>
                </div>
            </div>
            <!-- Add Modal HTML -->
```

```
<div id="addModal" class="modal fade">
    <div class="modal-dialog">
        <div class="modal-content">
            <form action="/ocupacionCreate" method="get" id="form">
                <div class="modal-header">
                    <h4 class="modal-title">Crear</h4>
                    <button type="button" class="close" data-dismiss="modal" aria-
hidden="true">&times;</button>
                </div>
                <div class="modal-body">
                    <div class="form-group" hidden>
                        <label for="id">ID:</label>
                        <input type="text" id="OCUPACION_ID" class="form-control">
                    </div>
                    <div class="form-group">
                        <label for="descripcion">Descripción:</label>
                        <input type="text" id="OCUPACION_DESCRIPCION" name="OC
UPACION_DESCRIPCION" class="form-control" required>
                    </div>
```

```
                        <div class="form-group">
                            <label>Activo:</label><br>
                            <input type="checkbox" class="custom-
checkbox" id="OCUPACION_ACTIVO" name="OCUPACION_ACTIVO" disabled="tr
ue" value="1" checked>
                        </div>
                    </div>
                    <div class="modal-footer">
                        <input type="button" class="btn btn-default" data-
dismiss="modal" value="Cancelar">
                        <input type="submit" class="btn btn-success" value="Agregar">
                    </div>
                </form>
            </div>
        </div>
    </div>
        <table class="table table-striped table-hover" id="myTable">
            <thead>
                <tr>
```

```
                    <th>
                        <span class="custom-checkbox">
                            <input type="checkbox" id="selectAll">
                            <label for="selectAll"></label>
                        </span>
                    </th>
                    <th>ID</th>
                    <th>Descripcion</th>
                    <th>Activo</th>
                </tr>
            </thead>
        @foreach($lista as $elemento)
            <tbody>
                <tr>
                    <td>
                        <span class="custom-checkbox">
                            <input type="checkbox" id="checkbox1" name="options[]" value
="1">
                            <label for="checkbox1"></label>
```

```
                    </span>
                </td>
                <td>{{$elemento->OCUPACION_ID}}</td>
                <td>{{$elemento->OCUPACION_DESCRIPCION}}</td>
                @if($elemento->OCUPACION_ACTIVO == 1)
                    <td><input type=checkbox class="custom-
checkbox" id="CB{{$elemento-
>OCUPACION_ACTIVO}}" checked disabled="true"></td>
                @else
                    <td><input type=checkbox class="custom-
checkbox" id="CB{{$elemento->OCUPACION_ACTIVO}}" disabled="true"></td>
                @endif
                </td>

    <!-- Edit Modal HTML -->
    <div id="editModal-{{$elemento->OCUPACION_ID}}" class="modal fade">
        <div class="modal-dialog">
            <div class="modal-content">
```

```
                <form action="/ocupacionUpdate/{{$elemento->OCUPACION_ID}}" method="get" id="forma">
                    <div class="modal-header">
                        <h4 class="modal-title">Editar</h4>
                        <button type="button" class="close" data-dismiss="modal" aria-hidden="true">&times;</button>
                    </div>
                    <div class="modal-body">
                        <div class="form-group">
                            <label for="id">ID:</label>
                            <input type="text" name="OCUPACION_ID" id="OCUPACION_ID" class="form-control" value="{{$elemento->OCUPACION_ID}}" disabled="true" required>
                        </div>
                        <div class="form-group">
                            <label for="descripcion">Descripción:</label>
                            <input type="text" id="OCUPACION_DESCRIPCION" name="OCUPACION_DESCRIPCION" class="form-control" value="{{$elemento->OCUPACION_DESCRIPCION}}" required>
                        </div>
```

```
                    <div class="form-group">
                        <label>Activo:</label><br>
                        @if($elemento->OCUPACION_ACTIVO == 1)
                            <input type=checkbox class="custom-
checkbox" id="CB{{$elemento-
>OCUPACION_ACTIVO}}" name="OCUPACION_ACTIVO" checked>
                        @else
                            <input type=checkbox class="custom-
checkbox" id="CB{{$elemento-
>OCUPACION_ACTIVO}}" name="OCUPACION_ACTIVO">
                        @endif
                    </div>
                </div>
                <div class="modal-footer">
                    <input type="button" class="btn btn-default" data-
dismiss="modal" value="Cancelar">
                    <input type="submit" class="btn btn-info" value="Guardar">
                </div>
            </form>
```

```
            </div>
          </div>
        </div>


                    <td>
                        <a href="#" class="edit" data-toggle="modal" data-
target="#editModal-{{$elemento->OCUPACION_ID}}"><i class="material-icons" data-
toggle="tooltip"
                        title="Edit">&#xE254;</i></a>
                    </td>
                </tr>
            </tbody>


            @endforeach
            </table>
        </div>
          </div>
        </div>
```

paisView.blade.php

```
@extends('layouts.app')

@section('title', 'Adm. Clientes')

@section('content')
    <div class="container">
    <form class="form-inline d-flex justify-content-center md-form form-sm searchbox">
        <input class="form-control form-control-sm mr-3 w-75" type="text" placeholder="Search" id="myInput" name="myInput"
            aria-label="Search" onkeyup="searchFunction()">
    </form>

    <script type="text/javascript">
    function searchFunction(){
        // Variables
        var input, filter, table, tr, td, i, txtValue;
```

```
  input = document.getElementById("myInput");
  filter = input.value.toUpperCase();
  table = document.getElementById("myTable");
  tr = table.getElementsByTagName("tr");



  for (i = 0; i < tr.length; i++) {
    td = tr[i].getElementsByTagName("td")[2];
    if (td) {
      txtValue = td.textContent || td.innerText;
      if (txtValue.toUpperCase().indexOf(filter) > -1) {
        tr[i].style.display = "";
      } else {
        tr[i].style.display = "none";
      }
    }
  }
}
```

```
</script>
    <div class="table-wrapper">
        <div class="table-title">
            <div class="row">
                <div class="col-sm-6">
                    <h2>Administrar <b>Paises</b></h2>
                </div>
                <div class="col-sm-6">
                    <a href="#addModal" class="btn btn-success" data-
toggle="modal"><i class="material-
icons">&#xE147;</i> <span>Nuevo Pais</span></a>
                </div>
            </div>
        </div>
        <!-- Add Modal HTML -->
    <div id="addModal" class="modal fade">
        <div class="modal-dialog">
            <div class="modal-content">
                <form action="/paisCreate" method="get">
```

```
                    <div class="modal-header">
                        <h4 class="modal-title">Crear</h4>
                        <button type="button" class="close" data-dismiss="modal" aria-
hidden="true">&times;</button>
                    </div>
                    <div class="modal-body">
                        <div class="form-group" hidden>
                            <label for="id">ID:</label>
                            <input type="text" id="PAIS_ID" class="form-control">
                        </div>
                        <div class="form-group">
                            <label for="descripcion">Nombre:</label>
                            <input type="text" id="PAIS_NOMBRE" name="PAIS_NOMBRE"
class="form-control" required>
                        </div>
                        <div class="form-group">
                            <label>Activo:</label><br>
```

```
                        <input type="checkbox" class="custom-
checkbox" id="PAIS_ACTIVO" name="PAIS_ACTIVO" disabled="true" value="1" che
cked>
                    </div>
                </div>
                <div class="modal-footer">
                    <input type="button" class="btn btn-default" data-
dismiss="modal" value="Cancelar">
                    <input type="submit" class="btn btn-success" value="Agregar">
                </div>
            </form>
        </div>
    </div>
</div>
        <table class="table table-striped table-hover" id="myTable">
            <thead>
                <tr>
                    <th>
                        <span class="custom-checkbox">
```

```
                        <input type="checkbox" id="selectAll">
                        <label for="selectAll"></label>
                    </span>
                </th>
                <th>ID</th>
                <th>Nombre</th>
                <th>Activo</th>
            </tr>
        </thead>
    @foreach($lista as $elemento)
        <tbody>
            <tr>
                <td>
                    <span class="custom-checkbox">
                        <input type="checkbox" id="checkbox1" name="options[]" value
="1">
                        <label for="checkbox1"></label>
                    </span>
                </td>
```

```
                    <td>{{$elemento->PAIS_ID}}</td>

                    <td>{{$elemento->PAIS_NOMBRE}}</td>

                    @if($elemento->PAIS_ACTIVO == 1)

                        <td><input type=checkbox class="custom-
checkbox" id="CB{{$elemento->PAIS_ACTIVO}}" checked disabled="true"></td>

                    @else

                        <td><input type=checkbox class="custom-
checkbox" id="CB{{$elemento->PAIS_ACTIVO}}" disabled="true"></td>

                    @endif

                    </td>



    <!-- Edit Modal HTML -->

    <div id="editModal-{{$elemento->PAIS_ID}}" class="modal fade">

        <div class="modal-dialog">

            <div class="modal-content">

                <form action="/paisUpdate/{{$elemento-
>PAIS_ID}}" method="get" id="forma">

                    <div class="modal-header">

                        <h4 class="modal-title">Editar</h4>
```

```
                    <button type="button" class="close" data-dismiss="modal" aria-
hidden="true">&times;</button>
                </div>
                <div class="modal-body">
                    <div class="form-group">
                        <label for="id">ID:</label>
                        <input type="text" name="PAIS_ID" id="PAIS_ID" class="form-
control" value="{{$elemento->PAIS_ID}}" disabled="true" required>
                    </div>
                    <div class="form-group">
                        <label for="descripcion">Nombre:</label>
                        <input type="text" id="PAIS_NOMBRE" name="PAIS_NOMBRE"
class="form-control" value="{{$elemento->PAIS_NOMBRE}}" required>
                    </div>
                    <div class="form-group">
                        <label>Activo:</label><br>
                        @if($elemento->PAIS_ACTIVO == 1)
                            <input type=checkbox class="custom-
checkbox" id="CB{{$elemento->PAIS_ACTIVO}}" name="PAIS_ACTIVO" checked>
```

```
                    @else
                        <input type=checkbox class="custom-checkbox" id="CB{{$elemento->PAIS_ACTIVO}}" name="PAIS_ACTIVO">
                    @endif
                </div>
            </div>
            <div class="modal-footer">
                <input type="button" class="btn btn-default" data-dismiss="modal" value="Cancelar">
                <input type="submit" class="btn btn-info" value="Guardar">
            </div>
        </form>
    </div>
</div>
</div>

                    <td>
```

```
                        <a href="#" class="edit" data-toggle="modal" data-
target="#editModal-{{$elemento->PAIS_ID}}"><i class="material-icons" data-
toggle="tooltip"
                        title="Edit">&#xE254;</i></a>
                    </td>
                </tr>
            </tbody>

            @endforeach
            </table>
    </div>
        </div>
    </div>
```

pisoView.blade.php

```
@extends('layouts.app')

@section('title', 'Adm. Clientes')
```

```
@section('content')
    <div class="container">
        <div class="table-wrapper">
            <div class="table-title">
                <div class="row">
                    <div class="col-sm-6">
                        <h2>Administrar <b>Pisos</b></h2>
                    </div>
                    <div class="col-sm-6">
                        <a href="#addModal" class="btn btn-success" data-
toggle="modal"><i class="material-
icons">&#xE147;</i> <span>Nuevo Piso</span></a>
                    </div>
                </div>
            </div>
            <!-- Add Modal HTML -->
        <div id="addModal" class="modal fade">
            <div class="modal-dialog">
                <div class="modal-content">
```

```
<form action="/pisoCreate" method="get" id="form">
    <div class="modal-header">
        <h4 class="modal-title">Crear</h4>
        <button type="button" class="close" data-dismiss="modal" aria-
hidden="true">&times;</button>
    </div>
    <div class="modal-body">
        <div class="form-group" hidden>
            <label for="id">ID:</label>
            <input type="text" id="PISO_ID" class="form-control">
        </div>
        <div class="form-group">
            <label for="descripcion">Número:</label>
            <input type="number" id="PISO_NUMERO" name="PISO_NUME
RO" class="form-control" required>
        </div>
        <div class="form-group">
            <label>Activo:</label><br>
```

```
                        <input type="checkbox" class="custom-
checkbox" id="PISO_ACTIVO" name="PISO_ACTIVO" value="1" disabled="true" che
cked>
                    </div>
                </div>
                <div class="modal-footer">
                    <input type="button" class="btn btn-default" data-
dismiss="modal" value="Cancelar">
                    <input type="submit" class="btn btn-success" value="Agregar">
                </div>
            </form>
        </div>
    </div>
</div>
        <table class="table table-striped table-hover">
            <thead>
                <tr>
                    <th>
                        <span class="custom-checkbox">
```

```
                        <input type="checkbox" id="selectAll">
                        <label for="selectAll"></label>
                    </span>
                </th>
                <th>ID</th>
                <th>Número</th>
                <th>Activo</th>
            </tr>
        </thead>
    @foreach($lista as $elemento)
        <tbody>
            <tr>
                <td>
                    <span class="custom-checkbox">
                        <input type="checkbox" id="checkbox1" name="options[]" value
="1">
                        <label for="checkbox1"></label>
                    </span>
                </td>
```

```
                    <td>{{$elemento->PISO_ID}}</td>

                    <td>{{$elemento->PISO_NUMERO}}</td>

                    @if($elemento->PISO_ACTIVO == 1)

                        <td><input type=checkbox class="custom-
checkbox" id="CB{{$elemento->PISO_ACTIVO}}" checked disabled="true"></td>

                    @else

                        <td><input type=checkbox class="custom-
checkbox" id="CB{{$elemento->PISO_ACTIVO}}" disabled="true"></td>

                    @endif

                    </td>



    <!-- Edit Modal HTML -->

    <div id="editModal-{{$elemento->PISO_ID}}" class="modal fade">

        <div class="modal-dialog">

            <div class="modal-content">

                <form action="/pisoUpdate/{{$elemento-
>PISO_ID}}" method="get" id="forma">

                    <div class="modal-header">

                        <h4 class="modal-title">Editar</h4>
```

```
                    <button type="button" class="close" data-dismiss="modal" aria-
hidden="true">&times;</button>
                </div>
                <div class="modal-body">
                    <div class="form-group">
                        <label for="id">ID:</label>
                        <input type="text" name="PISO_ID" id="PISO_ID" class="form-
control" value="{{$elemento->PISO_ID}}" disabled="true" required>
                    </div>
                    <div class="form-group">
                        <label for="descripcion">Número:</label>
                        <input type="number" id="PISO_NUMERO" name="PISO_NUME
RO" class="form-control" value="{{$elemento->PISO_NUMERO}}" required>
                    </div>
                    <div class="form-group">
                        <label>Activo:</label><br>
                        @if($elemento->PISO_ACTIVO == 1)
                            <input type=checkbox class="custom-
checkbox" id="CB{{$elemento->PISO_ACTIVO}}" name="PISO_ACTIVO" checked>
```

```
                        @else
                            <input type=checkbox class="custom-
checkbox" id="CB{{$elemento->PISO_ACTIVO}}" name="PISO_ACTIVO">
                        @endif
                    </div>
                </div>
                <div class="modal-footer">
                    <input type="button" class="btn btn-default" data-
dismiss="modal" value="Cancelar">
                    <input type="submit" class="btn btn-info" value="Guardar">
                </div>
            </form>
        </div>
    </div>
</div>

                    <td>
```

```
                        <a href="#" class="edit" data-toggle="modal" data-
target="#editModal-{{$elemento->PISO_ID}}"><i class="material-icons" data-
toggle="tooltip"
                        title="Edit">&#xE254;</i></a>
                    </td>
                </tr>
            </tbody>

            @endforeach
            </table>
    </div>
        </div>
    </div>
```

posibleResidenteView.blade.php

```
@extends('layouts.app')

@section('title', 'Adm. Clientes')
```

```
    @section('content')
    @if (session('alert'))
        <div class="alert alert-success" id="mensaje">
            {{ session('alert') }}

        </div>
    @endif
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.4/jquery.min.js"></scrip
t>
        <div class="container">
        <form class="form-inline d-flex justify-content-center md-form form-
sm searchbox">
            <input class="form-control form-control-sm mr-3 w-
75" type="text" placeholder="Search" id="myInput" name="myInput"
                aria-label="Search" onkeyup="searchFunction()">
        </form>

        <script type="text/javascript">
        function searchFunction(){
```

```
// Variables
var input, filter, table, tr, td, i, txtValue;
input = document.getElementById("myInput");
filter = input.value.toUpperCase();
table = document.getElementById("myTable");
tr = table.getElementsByTagName("tr");

for (i = 0; i < tr.length; i++) {
  td = tr[i].getElementsByTagName("td")[2];
  if (td) {
    txtValue = td.textContent || td.innerText;
    if (txtValue.toUpperCase().indexOf(filter) > -1) {
      tr[i].style.display = "";
    } else {
      tr[i].style.display = "none";
    }
  }
}
```

```
        }
        </script>
            <div class="table-wrapper">
                <div class="table-title">
                    <div class="row">
                        <div class="col-sm-6">
                            <h2>Administrar <b>Posibles Residentes</b></h2>
                        </div>
                        <div class="col-sm-6">
                            <a href="#addModal" class="btn btn-success" data-
toggle="modal"><i class="material-
icons">&#xE147;</i> <span>Nueva Residente</span></a>
                        </div>
                    </div>
                </div>
                <!-- Add Modal HTML -->
        <div id="addModal" class="modal fade">
            <div class="modal-dialog">
                <div class="modal-content">
```

```
<form action="/residenteCreate" method="get" id="form">
    <div class="modal-header">
        <h4 class="modal-title">Crear</h4>
        <button type="button" class="close" data-dismiss="modal" aria-hidden="true">&times;</button>
    </div>
    <div class="modal-body">
        <div class="form-group" hidden>
            <label for="id">ID:</label>
            <input type="text" id="RESIDENTE_ID" class="form-control">
        </div>
        <div class="form-group">
            <label for="id">Pais:</label>
            <select name="PAIS" id="PAIS" required>
            <option value="0" >- Seleccione -</option>
            @foreach($listaPaises as $pais)
                <option value="{{$pais->PAIS_ID}}" >{{$pais->PAIS_NOMBRE}}</option>
            @endforeach
```

```
                    </select>
                </div>
                <div class="form-group">
                    <label for="id">Estado:</label>
                    <select name="ESTADO" id="ESTADO">
                    </select>
                </div>
                <div class="form-group">
                    <label for="id">Ciudad:</label>
                    <select name="CIUDAD_ID" id="CIUDAD" required>
                    </select>
                </div>
                <div class="form-group">
                    <label for="id">Colegio:</label>
                    <select name="COLEGIO_ID" id="COLEGIO" required>
                    </select>
                </div>
<script type="text/javascript">
```

```
$(document).ready(function(){
    //cuando se cambie el combo de pais
$('#PAIS').on('change', function() {
    var pais_id=$('#PAIS').val();
$.ajax({
    url: "estadoXpais/"+pais_id,
    type: "GET",
    dataType: "json",
    success: function(respuesta){
        $('#ESTADO').empty();
        $('#CIUDAD').empty();
        $('#ESTADO').append('<option value=0> '+'- Seleccione -'+' </option>')
      respuesta.forEach(element => {
        $('#ESTADO').append('<option value='+element.ESTADO_ID+'> '+element.ES
TADO_NOMBRE+' </option>')
      });
    }
  });
});
```

```
    //cuando se cambie un combo de estado
    $('#ESTADO').on('change', function() {
        var estado_id=$('#ESTADO').val();
        $.ajax({
            url: "ciudadXestado/"+estado_id,
            type: "GET",
            dataType: "json",
            success: function(respuesta){
                $('#CIUDAD').empty();
                $('#COLEGIO').empty();
                $('#CIUDAD').append('<option value=0> '+'- Seleccione -'+' </option>')
              respuesta.forEach(element => {
                $('#CIUDAD').append('<option value='+element.CIUDAD_ID+'> '+element.C
IUDAD_NOMBRE+' </option>')
              });
            }
          });
        });
```

```
    //cuando se cambie un combo de ciudad
    $('#CIUDAD').on('change', function() {
        var ciudad_id=$('#CIUDAD').val();
        $.ajax({
            url: "colegioXciudad/"+ciudad_id,
            type: "GET",
            dataType: "json",
            success: function(respuesta){
                $('#COLEGIO').empty();
              respuesta.forEach(element => {
                $('#COLEGIO').append('<option value='+element.COLEGIO_ID+'> '+element
.COLEGIO_NOMBRE+' </option>')
              });
            }
          });
        });
        $('#UNIVERSIDAD').on('change', function() {
        var universidad_id=$('#UNIVERSIDAD').val();
```

```
$.ajax({
    url: "carrerasXuniversidad/"+universidad_id,
    type: "GET",
    dataType: "json",
    success: function(respuesta){
      $('#CARRERA').empty();
     respuesta.forEach(element => {
      $('#CARRERA').append('<option value='+element.CARRERA_ID+'> '+eleme
nt.CARRERA_NOMBRE+' </option>')
     });
    }
  });
});
$('#UNIVERSIDAD_MODAL').on('change', function() {
var universidad_id=$('#UNIVERSIDAD_MODAL').val();
$.ajax({
    url: "carrerasXuniversidad/"+universidad_id,
    type: "GET",
    dataType: "json",
```

```
            success: function(respuesta){
                $('#CARRERA_MODAL').empty();
                respuesta.forEach(element => {
                $('#CARRERA_MODAL').append('<option value='+element.CARRERA_ID+'
> '+element.CARRERA_NOMBRE+' </option>')
                });
            }
          });
        });
    });
    </script>
                        <div class="form-group">
                            <label for="descripcion">CI:</label>
                            <input type="text" id="RESIDENTE_CEDULA" name="RESIDEN
TE_CEDULA" class="form-control"
                            minlength="10" maxlength="10" onfocusout="validateId(this)" requi
red>
                        </div>
                        <div id="salida" class="messagediv"></div>
```

```
<script type="text/javascript">

    function validateId(select){

        var cad = select.value.trim();

        var total = 0;

        var longitud = cad.length;

        var longcheck = longitud - 1;



        if (cad !== "" && longitud === 10){

        for(i = 0; i < longcheck; i++){

            if (i%2 === 0) {

            var aux = cad.charAt(i) * 2;

            if (aux > 9) aux -= 9;

            total += aux;

            } else {

            total += parseInt(cad.charAt(i)); // parseInt o concatenará en lug
ar de sumar

            }

        }
```

```
                    total = total % 10 ? 10 - total % 10 : 0;



                    if (cad.charAt(longitud-1) == total) {

                        document.getElementById('salida').style.color = '#008000';

                        setTimeout(function(){ document.getElementById("salida").inn
erHTML = ("Cedula Válida");



                    }else{

                        document.getElementById('salida').style.color = '#FF0000';

                        document.getElementById("salida").innerHTML = ("Cedula In
válida");

                        document.getElementById("RESIDENTE_CEDULA").value =
"";

                    }

                    }

                }

            </script>

            <div class="form-group">

                <label for="descripcion">Nombre:</label>
```

```
                    <input type="text" id="RESIDENTE_NOMBRE" name="RESIDEN
TE_NOMBRE" class="form-control" required>
                </div>
                <div class="form-group">
                    <label>Telefono:</label><br>
                    <input type="number" id="RESIDENTE_TELEFONO" name="RES
IDENTE_TELEFONO" class="form-control"
                    minlength="9" maxlength="10" required>
                </div>
                <div class="form-group">
                    <label>Celular:</label><br>
                    <input type="number" id="RESIDENTE_CELULAR" name="RESI
DENTE_CELULAR" class="form-control"
                    minlength="9" maxlength="10" required>
                </div>
                <div class="form-group">
                    <label>Mail:</label><br>
                    <input type="email" id="RESIDENTE_MAIL" name="RESIDENT
E_MAIL" class="form-control" required>
```

```
                </div>
                <div class="form-group">
                    <label>Fecha Nacimiento:</label><br>
                    <input type="date" id="RESIDENTE_FECHA_NAC" name="RESI
DENTE_FECHA_NAC" class="form-control date" required>
                </div>
                <div class="form-group">
                    <label>Dirección:</label><br>
                    <input type="text" id="RESIDENTE_DIRECCION" name="RESID
ENTE_DIRECCION" class="form-control" required>
                </div>
                <div class="form-group">
                    <label>Promedio de graduación:</label><br>
                    <input type="number" id="RESIDENTE_PROMEDIO_GRADUAC
ION" name="RESIDENTE_PROMEDIO_GRADUACION" class="form-
control" required>
                </div>
                <div class="form-group">
                    <label for="id">Universidad:</label>
```

```
                        <select name="UNIVERSIDAD_ID" id="UNIVERSIDAD" require
>
                        <option value="0" >- Seleccione -</option>
                        @foreach($listaUniversidades as $universidad)
                           <option value="{{$universidad-
>UNIVERSIDAD_ID}}" >{{$universidad->UNIVERSIDAD_NOMBRE}}</option>
                        @endforeach
                        </select>
                    </div>
                    <div class="form-group">
                        <label for="id">Carrera:</label>
                        <select name="CARRERA_ID" id="CARRERA" required>
                        </select>
                    </div>
                    <div class="form-group">
                        <label>Semestre a cursar:</label><br>
                        <input type="number" id="RESIDENTE_SEMESTRE_A_CURSAR
" name="RESIDENTE_SEMESTRE_A_CURSAR" class="form-control" required>
                    </div>
```

```
                    <div class="form-group">
                        <label>Fecha Ingreso:</label><br>
                        <input type="date" id="RESIDENTE_FECHA_INGRESO" name="
RESIDENTE_FECHA_INGRESO" class="form-control" required>
                    </div>
                    <div class="form-group">
                        <label>Activo:</label><br>
                        <input type="checkbox" class="custom-
checkbox" id="RESIDENTE_ACTIVO" name="RESIDENTE_ACTIVO" disabled="true
" value="1" checked>
                    </div>
                </div>
                <div class="modal-footer">
                    <input type="button" class="btn btn-default" data-
dismiss="modal" value="Cancelar">
                    <input type="submit" class="btn btn-success" value="Agregar">
                </div>
            </form>
        </div>
```

```
    </div>
</div>
        <table class="table table-striped table-hover" id="myTable">
            <thead>
                <tr>
                    <th>
                        <span class="custom-checkbox">
                            <input type="checkbox" id="selectAll">
                            <label for="selectAll"></label>
                        </span>
                    </th>
                    <th>ID</th>
                    <th>Nombre</th>
                    <th>Celular</th>
                    <th>Ciudad</th>
                    <th>Activo</th>
                    <th>Residente</th>
                    <th>Padres</th>
```

```
                </tr>
            </thead>
        @foreach($lista as $elemento)
            <tbody>
                <tr>
                    <td>
                        <span class="custom-checkbox">
                            <input type="checkbox" id="checkbox1" name="options[]" value
="1">
                            <label for="checkbox1"></label>
                        </span>
                    </td>
                    <td>{{$elemento->RESIDENTE_ID}}</td>
                    <td>{{$elemento->RESIDENTE_NOMBRE}}</td>
                    <td>{{$elemento->RESIDENTE_CELULAR}}</td>
                    <td>{{$elemento->CIUDAD_NOMBRE}}</td>
                    @if($elemento->RESIDENTE_ACTIVO == 1)
```

```
                    <td><input type="checkbox" class="custom-
checkbox" id="CB{{$elemento-
>RESIDENTE_ACTIVO}}" checked disabled="true"></td>
                @else
                    <td><input type="checkbox" class="custom-
checkbox" id="CB{{$elemento->RESIDENTE_ACTIVO}}" disabled="true"></td>
                @endif
                @if($elemento->RESIDENTE_INGRESO == 1)
                    <td><input type="checkbox" class="custom-
checkbox" id="CB{{$elemento-
>RESIDENTE_INGRESO}}" checked disabled="true"></td>
                @else
                    <td><input type="checkbox" class="custom-
checkbox" id="CB{{$elemento->RESIDENTE_INGRESO}}" disabled="true"></td>
                @endif
                </td>

    <!-- Edit Modal HTML -->
    <div id="editModal-{{$elemento->RESIDENTE_ID}}" class="modal fade">
        <div class="modal-dialog">
```

```
            <div class="modal-content">

                <form action="/residenteUpdate/{{$elemento->RESIDENTE_ID}}" method="get" id="forma">

                    <div class="modal-header">

                        <h4 class="modal-title">Editar</h4>

                        <button type="button" class="close" data-dismiss="modal" aria-hidden="true">&times;</button>

                    </div>

                    <div class="modal-body">

                        <div class="form-group">

                            <label for="id">ID:</label>

                            <input type="text" name="RESIDENTE_ID" id="RESIDENTE_ID" class="form-control" value="{{$elemento->RESIDENTE_ID}}" disabled="true" required>

                        </div>

                        <div class="form-group">

                            <label for="descripcion">CI:</label>

                            <input type="text" id="RESIDENTE_CEDULAUPDATE" name="RESIDENTE_CEDULAUPDATE" class="form-control" value="{{$elemento->RESIDENTE_CEDULA}}" disabled>
```

```
                </div>


                <div class="form-group">
                    <label for="descripcion">Nombre:</label>
                    <input type="text" id="RESIDENTE_NOMBRE" name="RESIDEN
TE_NOMBRE" class="form-control" value="{{$elemento-
>RESIDENTE_NOMBRE}}" required>
                </div>
                <div class="form-group">
                    <label>Ciudad:</label><br>
                    <input type="text" id="CIUDAD_NOMBRE" name="CIUDAD_NO
MBRE" class="form-control" value="{{$elemento->CIUDAD_NOMBRE}}" disabled>
                </div>
                <div class="form-group">
                    <label>Telefono:</label><br>
                    <input type="text" id="RESIDENTE_TELEFONO" name="RESID
ENTE_TELEFONO" class="form-control" value="{{$elemento-
>RESIDENTE_TELEFONO}}" required>
                </div>
```

```
                    <div class="form-group">
                        <label>Celular:</label><br>
                        <input type="text" id="RESIDENTE_CELULAR" name="RESIDE
NTE_CELULAR" class="form-control" value="{{$elemento-
>RESIDENTE_CELULAR}}" required>
                    </div>
                    <div class="form-group">
                        <label>Mail:</label><br>
                        <input type="email" id="RESIDENTE_MAIL" name="RESIDENT
E_MAIL" class="form-control" value="{{$elemento-
>RESIDENTE_MAIL}}" required>
                    </div>
                    <div class="form-group">
                        <label>Fecha Nacimiento:</label><br>
                        <input type="date" id="RESIDENTE_FECHA_NAC" name="RESI
DENTE_FECHA_NAC" class="form-control" value="{{$elemento-
>RESIDENTE_FECHA_NAC}}" required>
                    </div>
                    <div class="form-group">
                        <label>Dirección:</label><br>
```

```
                    <input type="text" id="RESIDENTE_DIRECCION" name="RESID
ENTE_DIRECCION" class="form-control" value="{{$elemento-
>RESIDENTE_DIRECCION}}" required>
                </div>
                <div class="form-group">
                    <label>Colegio:</label><br>
                    <input type="text" id="COLEGIO_NOMBRE" name="COLEGIO_
NOMBRE" class="form-control" value="{{$elemento-
>COLEGIO_NOMBRE}}" disabled>
                </div>
                <div class="form-group">
                    <label>Promedio de graduación:</label><br>
                    <input type="text" id="RESIDENTE_PROMEDIO_GRADUACIO
N" name="RESIDENTE_PROMEDIO_GRADUACION" class="form-
control" value="{{$elemento-
>RESIDENTE_PROMEDIO_GRADUACION}}" required>
                </div>
                <div class="form-group">
                    <label for="id">Universidad:</label>
```

```
                    <select name="UNIVERSIDAD_ID" id="{{$elemento-
>RESIDENTE_ID}}" onchange="myFunction(this)" required>
                    @foreach($listaUniversidades as $universidad)
                        @if($universidad->UNIVERSIDAD_ID == $elemento-
>UNIVERSIDAD_ID)
                        <option value="{{$universidad-
>UNIVERSIDAD_ID}}" selected>{{$universidad-
>UNIVERSIDAD_NOMBRE}}</option>
                        @else
                        <option value="{{$universidad-
>UNIVERSIDAD_ID}}">{{$universidad->UNIVERSIDAD_NOMBRE}}</option>
                        @endif
                    @endforeach
                    </select>
                </div>
                <script type="text/javascript">
                    function myFunction(select){
                        var selected = parseInt(select.value);
                        var nombreComboCarrera = "#CARRERA-"+select.id;
```

```
                    //alert("idSeleccion "+selected);
                    //alert("idCombo "+nombreComboCarrera);
                    $.ajax({
                        url: "carrerasXuniversidad/"+selected,
                        type: "GET",
                        dataType: "json",
                        success: function(respuesta){
                            //alert("entro al ajax");
                            $(nombreComboCarrera).empty();
                        respuesta.forEach(element => {
                            //alert(element.CARRERA_NOMBRE);
                            $(nombreComboCarrera).append('<option value='+element.C
ARRERA_ID+'> '+element.CARRERA_NOMBRE+' </option>')
                        });
                        }
                    });
                }
            </script>
            <div class="form-group">
```

```
                    <label for="id">Carrera:</label>

                    <select name="CARRERA_ID" id="CARRERA-{{$elemento->RESIDENTE_ID}}" required>

                    @foreach($listaCarreras as $carrera)

                        @if($carrera->UNIVERSIDAD_ID == $elemento->UNIVERSIDAD_ID)

                        @if($carrera->CARRERA_ID == $elemento->CARRERA_ID)

                        <option value="{{$carrera->CARRERA_ID}}" selected>{{$carrera->CARRERA_NOMBRE}}</option>

                        @else

                        <option value="{{$carrera->CARRERA_ID}}">{{$carrera->CARRERA_NOMBRE}}</option>

                        @endif

                        @endif

                    @endforeach

                    </select>

                </div>

                <div class="form-group">

                    <label>Semestre a cursar:</label><br>
```

```
                    <input type="txt" id="RESIDENTE_SEMESTRE_A_CURSAR" na
me="RESIDENTE_SEMESTRE_A_CURSAR" class="form-
control" value="{{$elemento->RESIDENTE_SEMESTRE_A_CURSAR}}" required>
                </div>
                <div class="form-group">
                    <label>Fecha Ingreso:</label><br>
                    <input type="date" id="RESIDENTE_FECHA_INGRESO" name="
RESIDENTE_FECHA_INGRESO" class="form-control" value="{{$elemento-
>RESIDENTE_FECHA_INGRESO}}" required>
                </div>

                <div class="form-group">
                    <label>Activo:</label><br>
                    @if($elemento->RESIDENTE_ACTIVO == 1)
                        <input type="checkbox" class="custom-
checkbox" id="CB{{$elemento-
>RESIDENTE_ACTIVO}}" name="RESIDENTE_ACTIVO" checked>
                    @else
```

```
                        <input type="checkbox" class="custom-
checkbox" id="CB{{$elemento-
>RESIDENTE_ACTIVO}}" name="RESIDENTE_ACTIVO">
                    @endif
                </div>
                <div class="form-group">
                    <label>Es residente:</label><br>
                    @if($elemento->RESIDENTE_INGRESO == 1)
                        <input type="checkbox" class="custom-
checkbox" id="CB{{$elemento-
>RESIDENTE_INGRESO}}" name="RESIDENTE_INGRESO" checked>
                    @else
                        <input type="checkbox" class="custom-
checkbox" id="CB{{$elemento-
>RESIDENTE_INGRESO}}" name="RESIDENTE_INGRESO">
                    @endif
                </div>
            </div>
            <div class="modal-footer">
```

```
                    <input type="button" class="btn btn-default" data-
dismiss="modal" value="Cancelar">
                    <input type="submit" class="btn btn-info" value="Guardar">
                </div>
            </form>
        </div>
    </div>
</div>
<!-- Parent Modal HTML -->
<div id="parentModal-{{$elemento->RESIDENTE_ID}}" class="modal fade tab">
    <div class="modal-dialog">
        <div class="modal-content" class="">
            <div class="modal-header">
                <button class="tablinks" onclick="openParent(event, 'padre-{{$elemento-
>RESIDENTE_ID}}')">Padre</button>
                <button class="tablinks" onclick="openParent(event, 'madre-
{{$elemento->RESIDENTE_ID}}')">Madre</button>
                <button type="button" class="close" data-dismiss="modal" aria-
hidden="true">&times;</button>
```

```
            <script type="text/javascript">

                function openParent(evt, parentName) {

                    var i, tabcontent, tablinks;

tabcontent = document.getElementsByClassName("tabcontent");

for (i = 0; i < tabcontent.length; i++) {

  tabcontent[i].style.display = "none";

}

tablinks = document.getElementsByClassName("tablinks");

for (i = 0; i < tablinks.length; i++) {

  tablinks[i].className = tablinks[i].className.replace(" active", "");

}



// Show the current tab, and add an "active" class to the button that opened the tab

document.getElementById(parentName).style.display = "block";

evt.currentTarget.className += " active";

    var residente_id = parentName.substr(6);

    var nombreModalPadre = "padre-" + residente_id;

    var nombreModalMadre = "madre-" + residente_id;
```

```
        if(parentName == nombreModalPadre){
            $.ajax({
                url: "getPadre/"+residente_id,
                type: "GET",
                dataType: "json",
                success: function(respuesta){
                    if(respuesta.length > 0)
                        {
                            //alert ('Entro');
                            document.getElementById('PPM_ID'+residente_id).value = respuesta[0
].PM_ID;
                            document.getElementById('PPM_NOMBRE'+residente_id).value = res
puesta[0].PM_NOMBRE;
                            document.getElementById('PPM_DIRECCION_OFICINA'+residente_
id).value = respuesta[0].PM_DIRECCION_OFICINA;
                            document.getElementById('PPM_TELEFONO_OFICINA'+residente_i
d).value = respuesta[0].PM_TELEFONO_OFICINA;
                            //document.getElementById('POCUPACION_ID'+residente_id).selecte
dIndex = respuesta[0].PM_OCUPACION_ID;
```

```
                }
                else{
                    document.getElementById('PPM_ID'+residente_id).value = 0;
                }
            }
        });
    }
    else{
        $.ajax({
            url: "getMadre/"+residente_id,
            type: "GET",
            dataType: "json",
            success: function(respuesta){
                if(respuesta.length > 0)
                {
                    //alert ('Entro');
                    document.getElementById('MPM_ID'+residente_id).value = respuesta[0].
PM_ID;
```

```
                document.getElementById('MPM_NOMBRE'+residente_id).value = resp
uesta[0].PM_NOMBRE;

                document.getElementById('MPM_DIRECCION_OFICINA'+residente_id
).value = respuesta[0].PM_DIRECCION_OFICINA;

                document.getElementById('MPM_TELEFONO_OFICINA'+residente_id)
.value = respuesta[0].PM_TELEFONO_OFICINA;

                //document.getElementById('MOCUPACION_ID'+residente_id).selected
Index = respuesta[0].PM_OCUPACION_ID;

            }

            else{

                document.getElementById('MPM_ID'+residente_id).value = 0;

            }

        }

    });

}

}

</script>

</div>
```

```
        <div id="padre-{{$elemento-
>RESIDENTE_ID}}" class="tabcontent" style="display:none">
        <form method="get" action="/parentUpdate/{{$elemento-
>RESIDENTE_ID}}" id="forma{{$elemento->RESIDENTE_ID}}">


            <div class="modal-body">
                <div class="form-group">
                    <label for="id">ID:</label>
                    <input type="text" name="PM_ID" id="PPM_ID{{$elemento-
>RESIDENTE_ID}}" class="form-control" readonly>
                </div>
                <div class="form-group">
                    <label for="id">Nombre:</label>
                    <input type="text" name="PM_NOMBRE" id="PPM_NOMBRE{{$elemen
to->RESIDENTE_ID}}" class="form-control" required>
                </div>
                <div class="form-group">
                    <label for="id">Ocupacion:</label>
```

```
                <select name="OCUPACION_ID" id="POCUPACION_ID{{$elemento->RESIDENTE_ID}}" required>
                @foreach($listaOcupaciones as $ocupacion)
                  <option value="{{$ocupacion->OCUPACION_ID}}">{{$ocupacion->OCUPACION_DESCRIPCION}}</option>
                @endforeach
                </select>
              </div>
              <div class="form-group">
                <label for="descripcion">Dirección Oficina:</label>
                <input type="text" id="PPM_DIRECCION_OFICINA{{$elemento->RESIDENTE_ID}}" name="PM_DIRECCION_OFICINA" class="form-control" required>
              </div>
              <div class="form-group">
                <label for="descripcion">Teléfono Oficina:</label>
                <input type="text" id="PPM_TELEFONO_OFICINA{{$elemento->RESIDENTE_ID}}" name="PM_TELEFONO_OFICINA" class="form-control" required>
              </div>
```

```
            </div>
            <div class="modal-footer">
                <input type="button" class="btn btn-default" data-
dismiss="modal" value="Cancelar">
                <input type="submit" class="btn btn-info" value="Guardar">
            </div>
        </form>
        </div>
        <div id="madre-{{$elemento-
>RESIDENTE_ID}}" class="tabcontent" style="display:none">
        <form action="/motherUpdate/{{$elemento-
>RESIDENTE_ID}}" method="get" id="form{{$elemento->RESIDENTE_ID}}">

            <div class="modal-body">
                <div class="form-group">
                    <label for="id">ID:</label>
                    <input type="text" name="PM_ID" id="MPM_ID{{$elemento-
>RESIDENTE_ID}}" class="form-control" readonly>
                </div>
```

```
<div class="form-group">
    <label for="id">Nombre:</label>
    <input type="text" name="PM_NOMBRE" id="MPM_NOMBRE{{$elemento->RESIDENTE_ID}}" class="form-control" required>
</div>
<div class="form-group">
    <label for="id">Ocupacion:</label>
    <select name="OCUPACION_ID" id="MOCUPACION_ID" required>
    @foreach($listaOcupaciones as $ocupacion)
        <option value="{{$ocupacion->OCUPACION_ID}}">{{$ocupacion->OCUPACION_DESCRIPCION}}</option>
    @endforeach
    </select>
</div>
<div class="form-group">
    <label for="descripcion">Dirección Oficina:</label>
    <input type="text" id="MPM_DIRECCION_OFICINA{{$elemento->RESIDENTE_ID}}" name="PM_DIRECCION_OFICINA" class="form-control" required>
```

```
            </div>
            <div class="form-group">
               <label for="descripcion">Teléfono Oficina:</label>
               <input type="text" id="MPM_TELEFONO_OFICINA{{$elemento->RESIDENTE_ID}}" name="PM_TELEFONO_OFICINA" class="form-control" required>
            </div>
         </div>
         <div class="modal-footer">
            <input type="button" class="btn btn-default" data-dismiss="modal" value="Cancelar">
            <input type="submit" class="btn btn-info" value="Guardar">
         </div>
      </form>
      </div>
            </div>
         </div>
      </div>
                  <td>
```

```
                        <a href="#" class="delete" data-toggle="modal" data-
target="#parentModal-{{$elemento->RESIDENTE_ID}}">
                        <input type="button" class="btn btn-info" data-
dismiss="modal" value="+">
                    </td>
                    <td>
                        <a href="#" class="edit" data-toggle="modal" data-
target="#editModal-{{$elemento->RESIDENTE_ID}}"><i class="material-icons" data-
toggle="tooltip"
                        title="Edit">&#xE254;</i></a>
                    </td>
                </tr>
            </tbody>

            @endforeach
            </table>
        </div>
    </div>
</div>
```

registrarActividadNumeraria.blade.php

```
@extends('layouts.app')

@section('title', 'Adm. Clientes')

@section('content')
    <div class="container">
        <div class="table-wrapper">
            <div class="table-title">
                <div class="row">
                    <div class="col-sm-6">
                        <h2>Actividades <b>Disponibles</b></h2>
                    </div>
                </div>
            </div>
            <table class="table table-striped table-hover">
                <thead>
```

```
                    <tr>
                        <th>
                            <span class="custom-checkbox">
                                <input type="checkbox" id="selectAll">
                                <label for="selectAll"></label>
                            </span>
                        </th>
                        <th>TipoActividad</th>
                        <th>Actividad</th>
                        <th>Fecha Inicio</th>
                        <th>Horario</th>
                        <th>Fecha Fin</th>
                        <th>Inscribirse</th>
                    </tr>
                </thead>
            @foreach($listaActividades as $actividad)
                <tbody>
                <form action="/actividadNumeraria/{{$actividad-
>HORARIO_ID}}/{{$numerariaID}}" method="get">
```

```
                    <tr>
                        <td>
                            <span class="custom-checkbox">
                                <input type="checkbox" id="checkbox1" name="options[]" value
="1">
                                <label for="checkbox1"></label>
                            </span>
                        </td>
                        <td>{{$actividad->TIPO_NOMBRE}}</td>
                        <td>{{$actividad->ACTIVIDAD_NOMBRE}}</td>
                        <td>{{$actividad->HORARIO_FECHA_INICIO}}</td>
                        <td>{{$actividad->HORARIO_HORA}}</td>
                        <td>{{$actividad->HORARIO_FECHA_FIN}}</td>

                        <td>
                            <input type="submit" class="btn btn-info" data-
dismiss="modal" value="+">
                        </td>
```

```
                </tr>
            </tbody>
            </form>

            @endforeach
            </table>
</div>
    </div>
</div>

<!-- tabla de actividades que si tiene el residente -->
<div class="container">
    <div class="table-wrapper">
        <div class="table-title">
            <div class="row">
                <div class="col-sm-6">
                    <h2>Actividades <b>Asignadas</b></h2>
                </div>
```

```
    </div>
</div>
<table class="table table-striped table-hover">
    <thead>
        <tr>
            <th>
                <span class="custom-checkbox">
                    <input type="checkbox" id="selectAll">
                    <label for="selectAll"></label>
                </span>
            </th>
            <th>TipoActividad</th>
            <th>Actividad</th>
            <th>Fecha Inicio</th>
            <th>Horario</th>
            <th>Fecha Fin</th>
            <th>Eiminar</th>
        </tr>
```

```
            </thead>
        @foreach($lista as $elemento)
            <tbody>
            <form action="/actividadNumerariaDelete/{{$elemento->HORARIO_ID}}/{{$numerariaID}}" method="get">
                <tr>
                    <td>
                        <span class="custom-checkbox">
                            <input type="checkbox" id="checkbox1" name="options[]" value="1">
                            <label for="checkbox1"></label>
                        </span>
                    </td>
                    <td>{{$elemento->TIPO_NOMBRE}}</td>
                    <td>{{$elemento->ACTIVIDAD_NOMBRE}}</td>
                    <td>{{$elemento->HORARIO_FECHA_INICIO}}</td>
                    <td>{{$elemento->HORARIO_HORA}}</td>
                    <td>{{$elemento->HORARIO_FECHA_FIN}}</td>
```

```
                    <td>

                        <input type="submit" class="btn btn-info" data-
dismiss="modal" value="-">

                    </td>

                </tr>

            </tbody>



            @endforeach

            </table>

        </div>

        </div>

    </div>
```

7.4 Anexo 4: Manual de usuario

MANUAL DE USUARIO

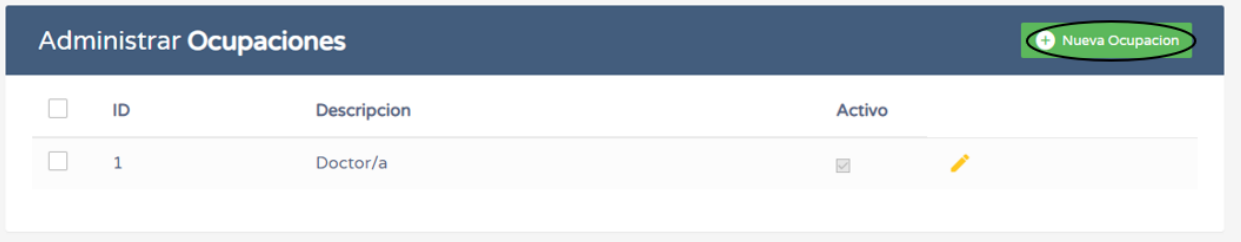
Contenido

Administración de catálogos

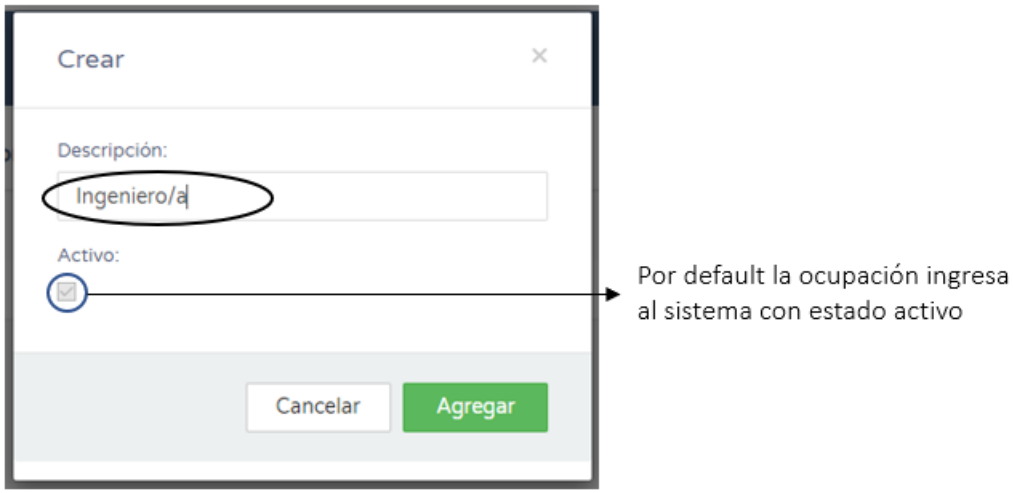
Ocupaciones

Ingresar Ocupaciones

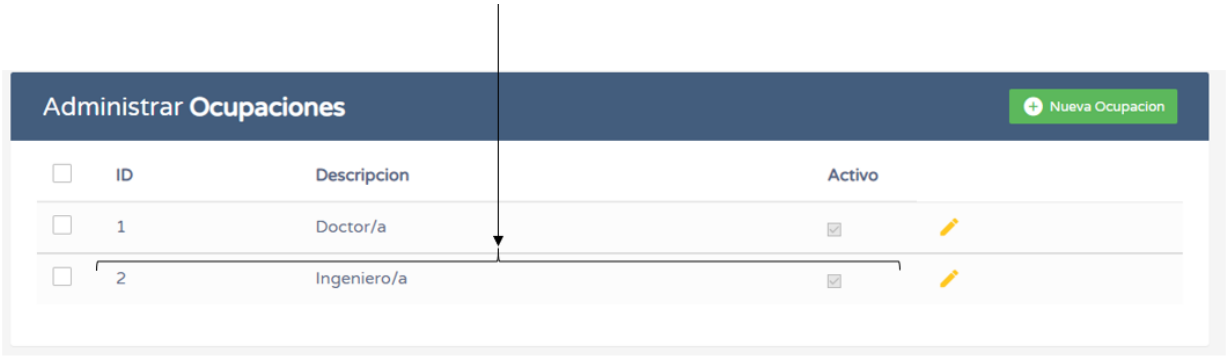
1. Seleccionar de la opción de Menú Catálogos > Ocupaciones.
2. Seleccionar el botón Nueva Ocupación.



3. Llenar los datos del modal.



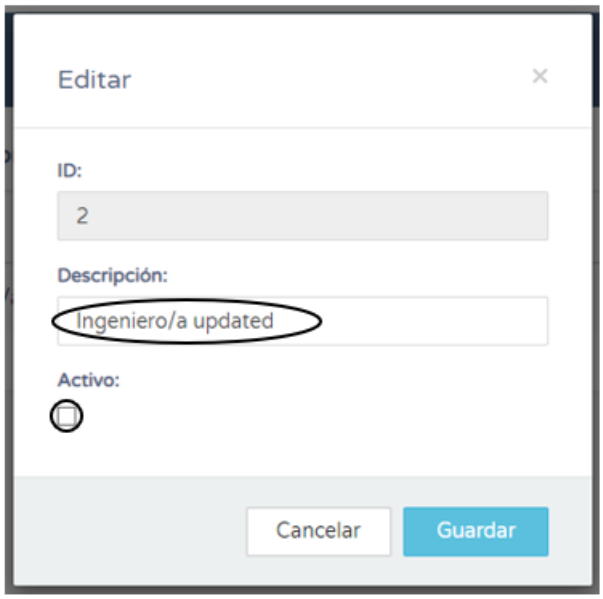
4. Presionar el botón Agregar.



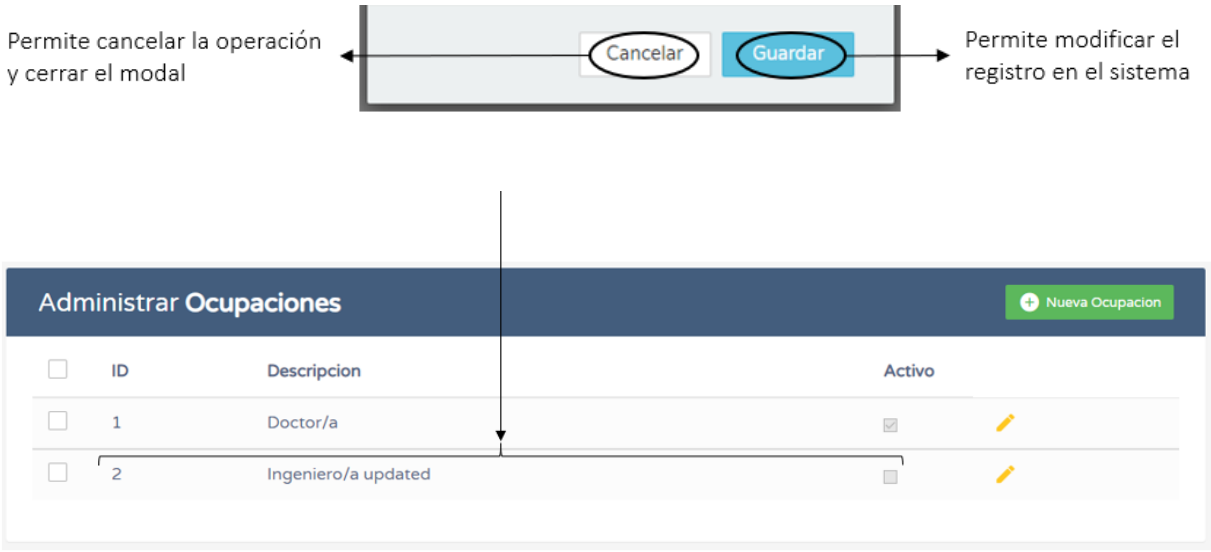
Modificar Ocupación

1. Seleccionar de la opción de Menú Catálogos > Ocupaciones.
2. Presionar el símbolo de edición de la ocupación que se desea editar.

3. Cambiar los datos que están activos para ser editados.



4. Presionar el botón Guardar.

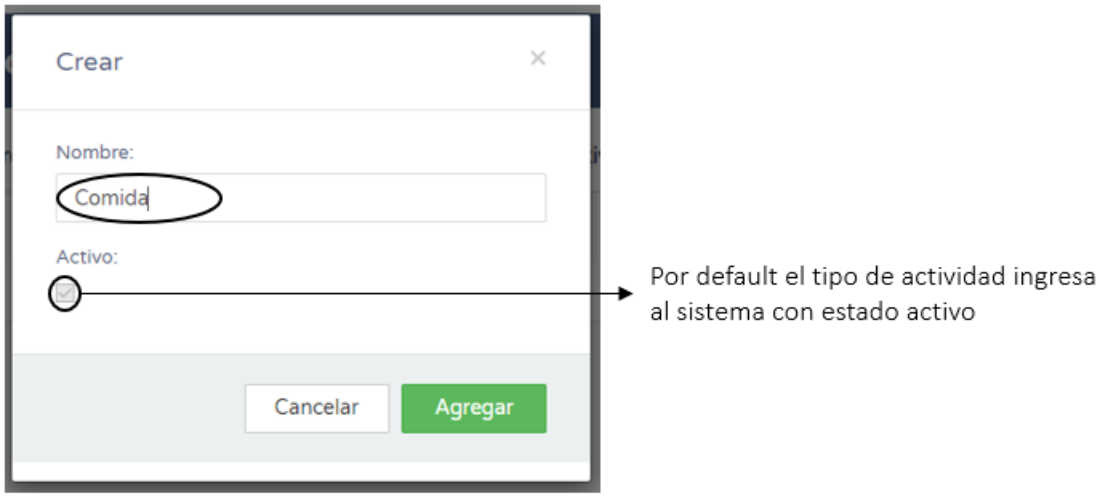


Tipos de actividades

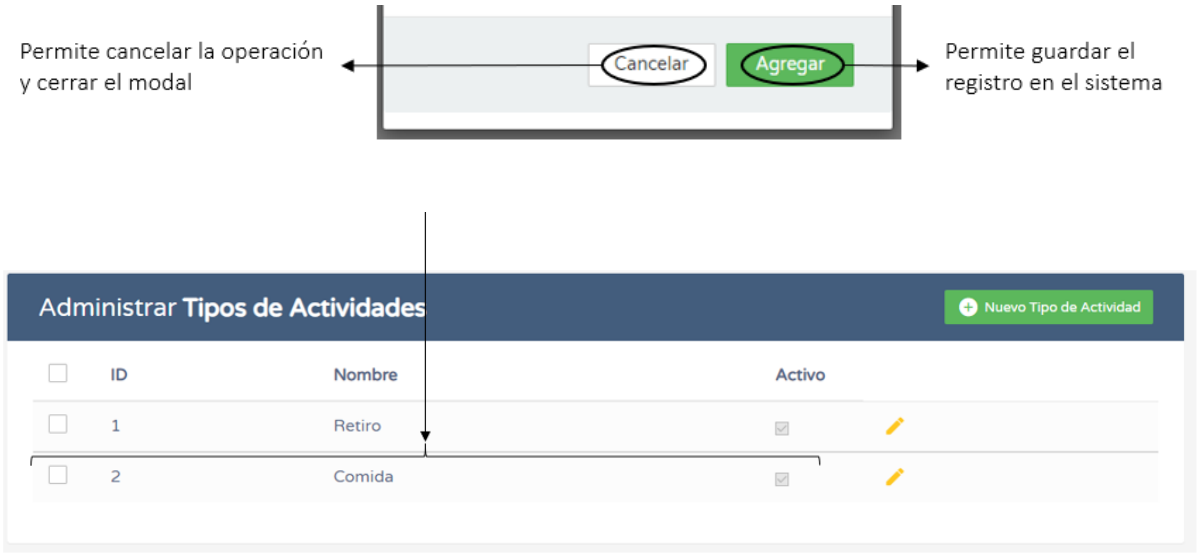
Ingresar Tipo de Actividad

1. Seleccionar de la opción de Menú Catálogos > Tipos Actividades.
2. Seleccionar el botón Nuevo Tipo de Actividad.

3. Llenar los datos del modal.

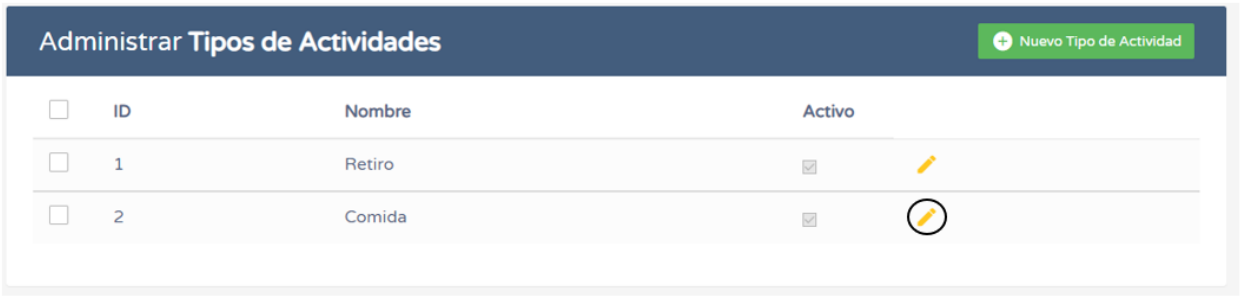


4. Presionar el botón Agregar.

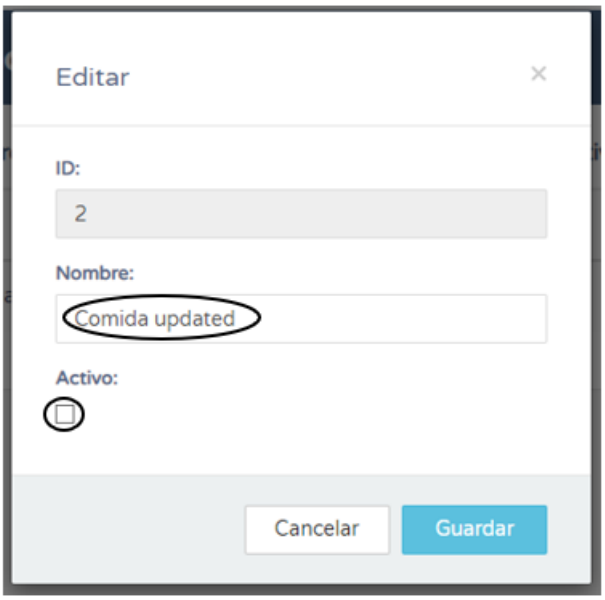


Modificar Tipo de Actividad

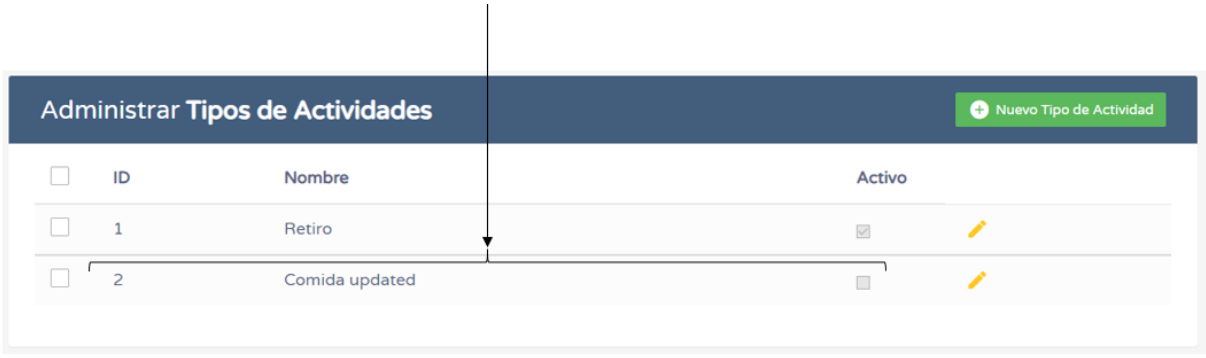
1. Seleccionar de la opción de Menú Catálogos > Tipos Actividades.
2. Presionar el símbolo de edición del tipo de actividad que se desea editar.



3. Cambiar los datos que están activos para ser editados.



4. Presionar el botón Guardar.

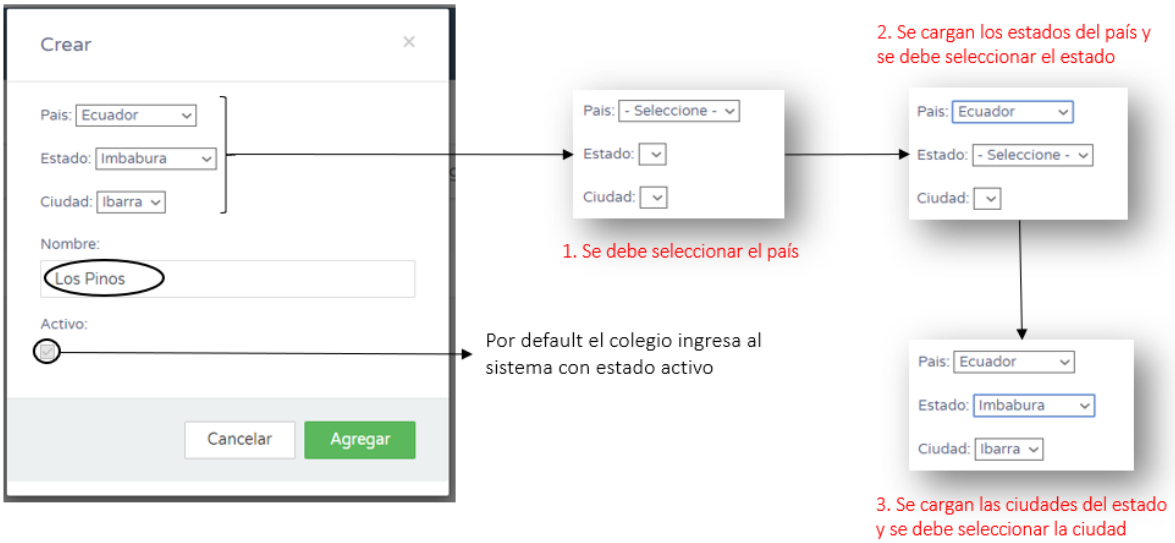


Colegios

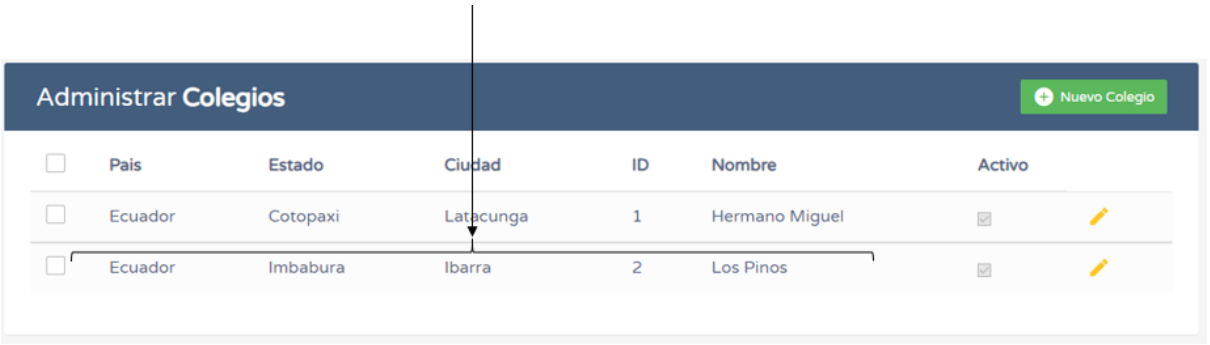
Ingresar Colegio

1. Seleccionar de la opción de Menú Catálogos > Colegios.
2. Seleccionar el botón Nuevo Colegio.

3. Llenar los datos del modal.

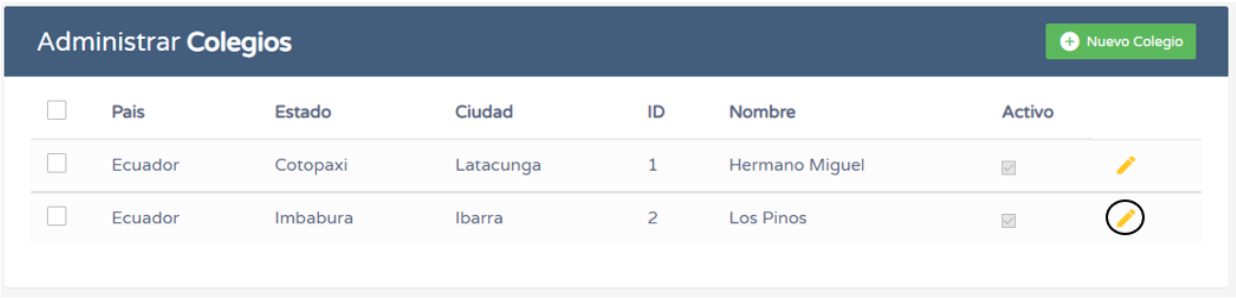


4. Presionar el botón Agregar.

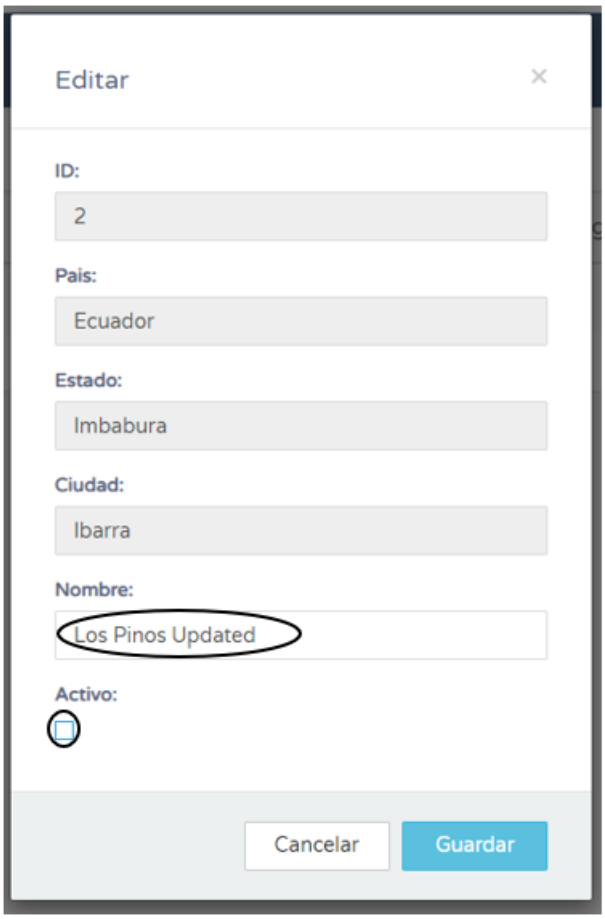


Modificar Colegio

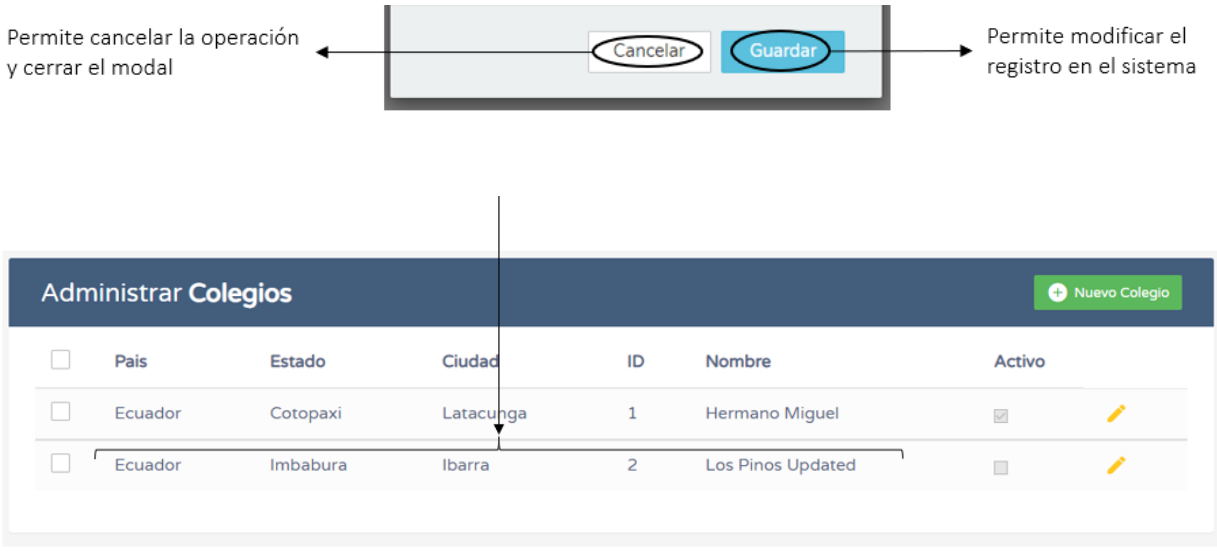
1. Seleccionar de la opción de Menú Catálogos > Colegios.
2. Presionar el símbolo de edición del colegio que se desea editar.



3. Cambiar los datos que están activos para ser editados.



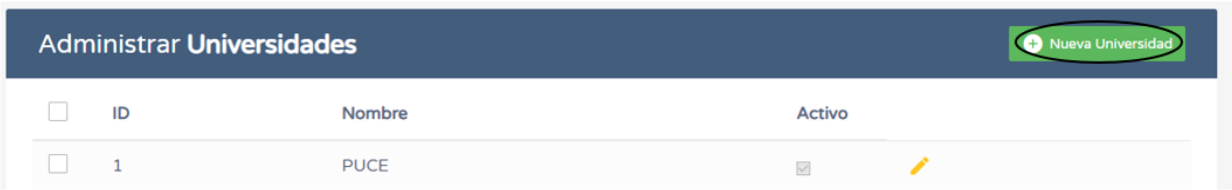
4. Presionar el botón Guardar.



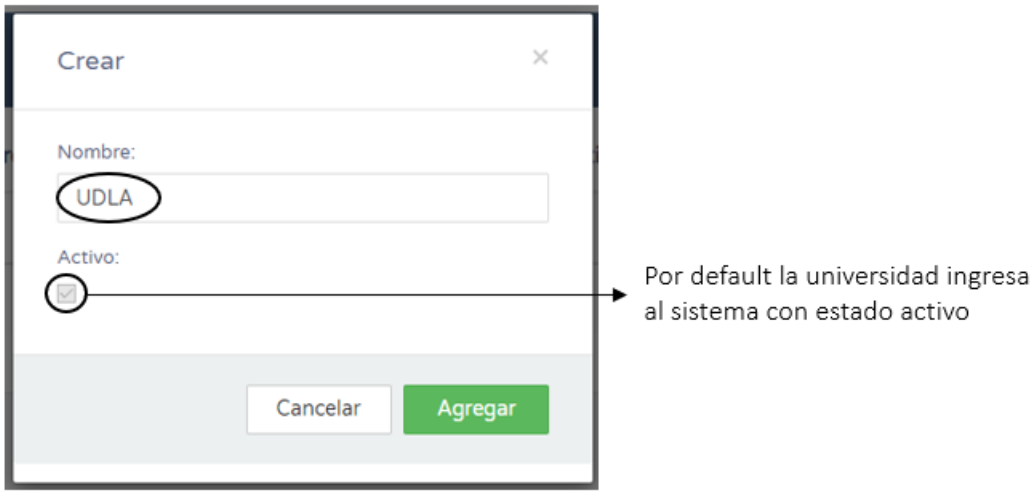
Universidades

Ingresar Universidad

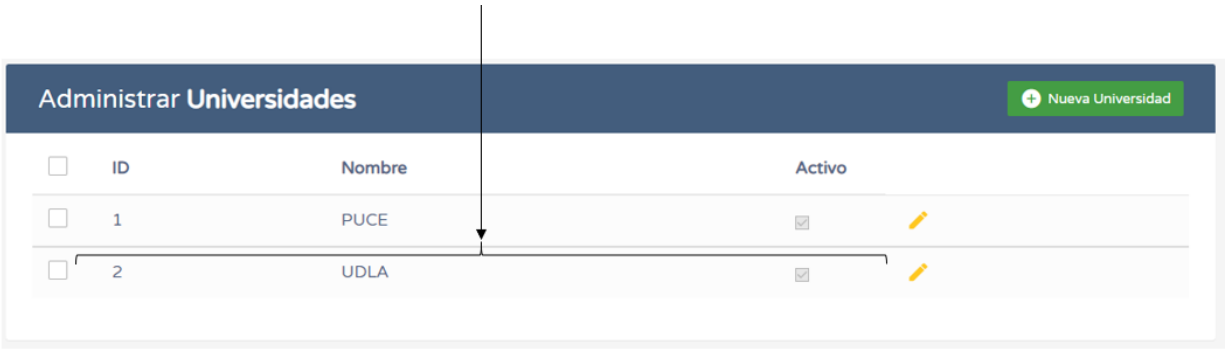
1. Seleccionar de la opción de Menú Catálogos > Universidades.
2. Seleccionar el botón Nueva Universidad.



3. Llenar los datos del modal.



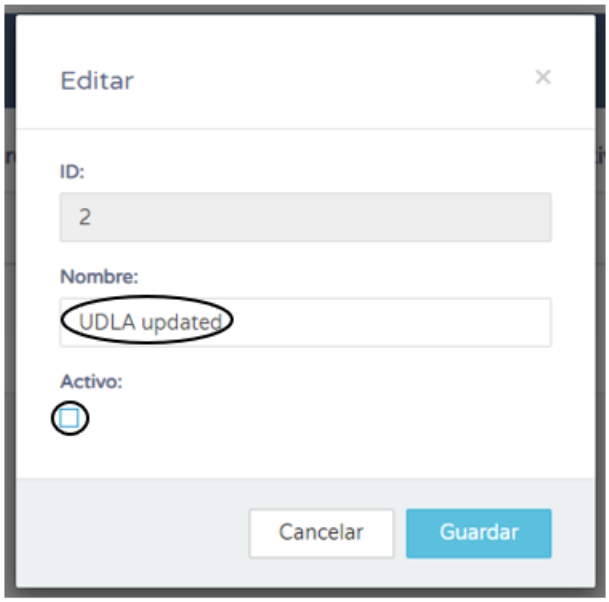
4. Presionar el botón Agregar.



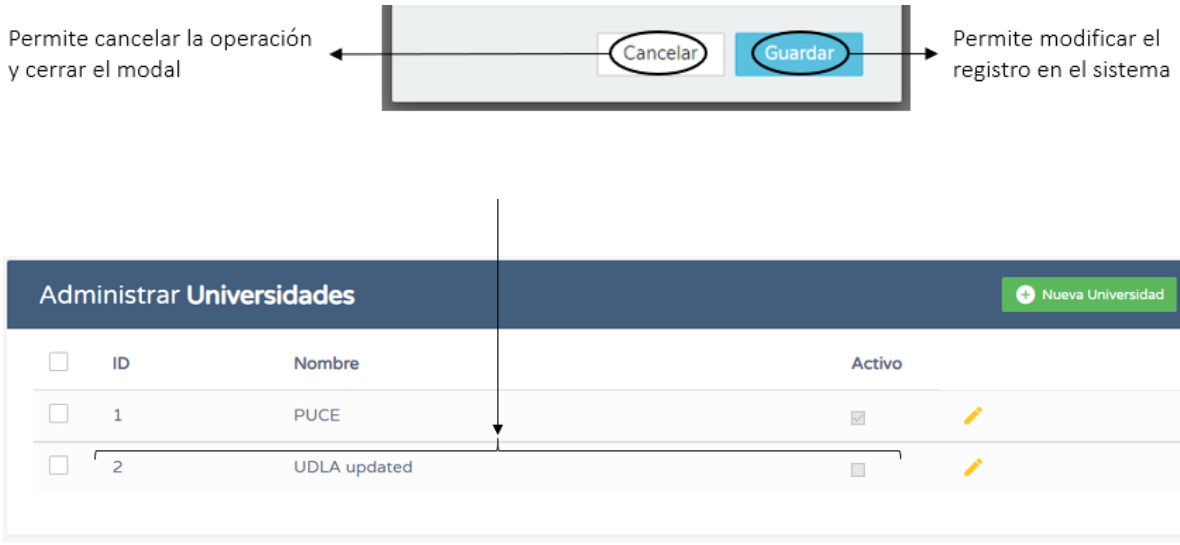
Modificar Universidad

1. Seleccionar de la opción de Menú Catálogos > Universidades.
2. Presionar el símbolo de edición de la universidad que se desea editar.

3. Cambiar los datos que están activos para ser editados.



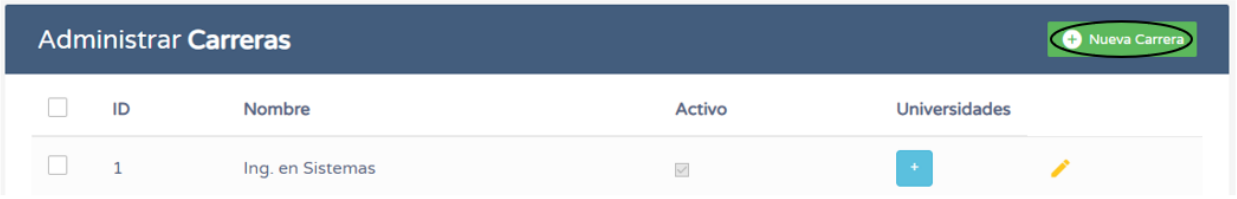
4. Presionar el botón Guardar.



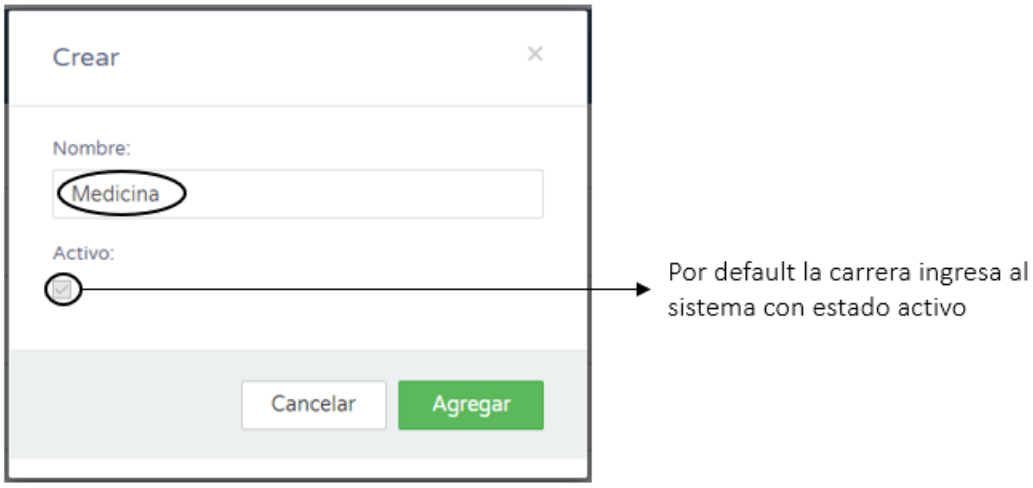
Carreras

Ingresar Carrera

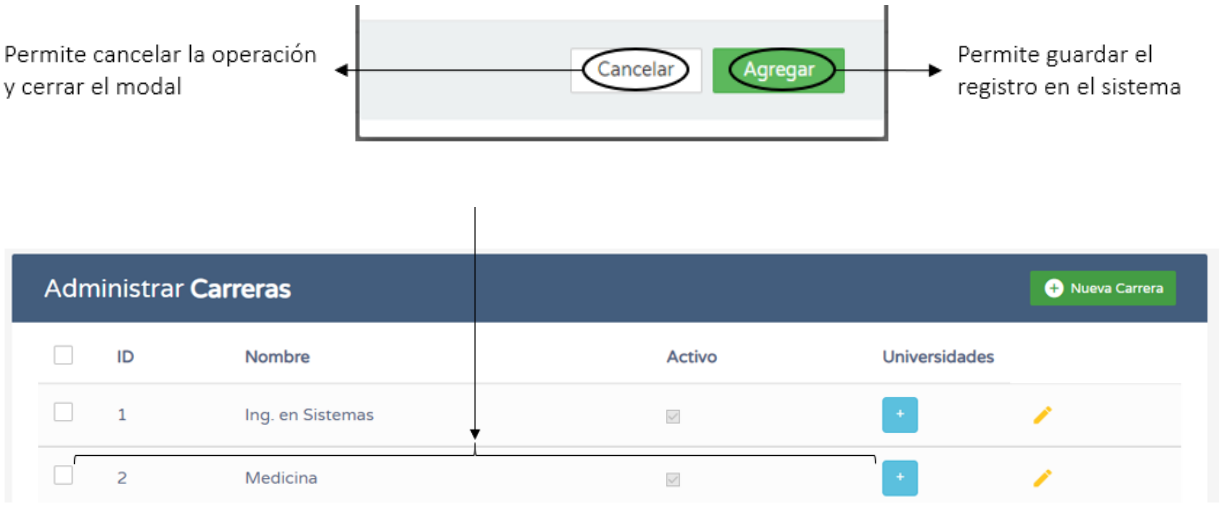
1. Seleccionar de la opción de Menú Catálogos > Carreras.
2. Seleccionar el botón Nueva Carrera.



3. Llenar los datos del modal.

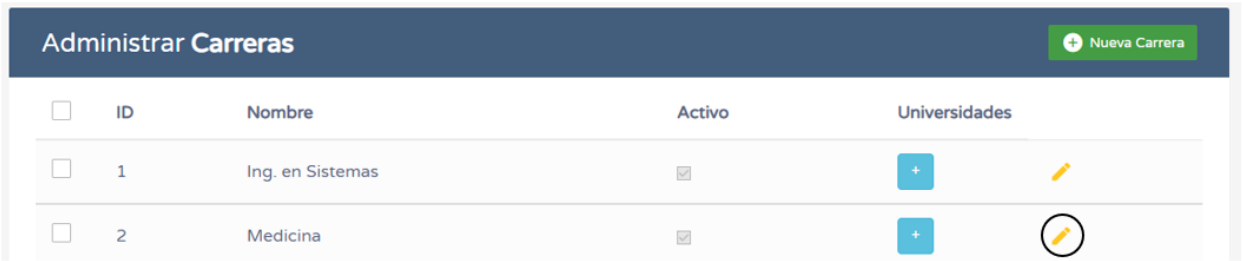


4. Presionar el botón Agregar.

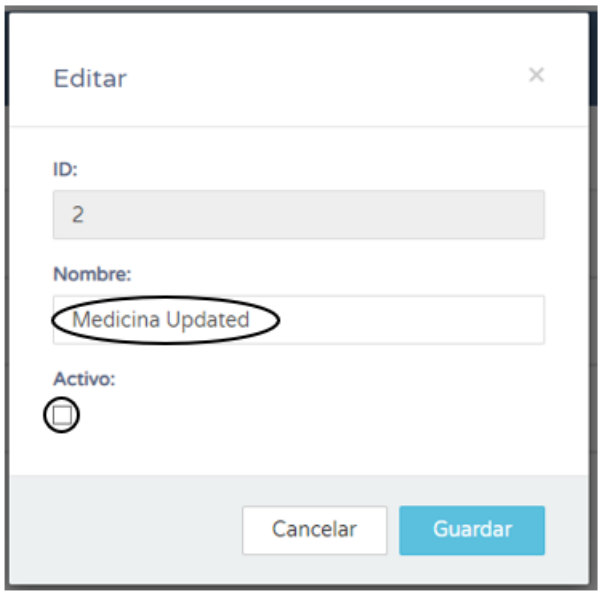


Modificar Carrera

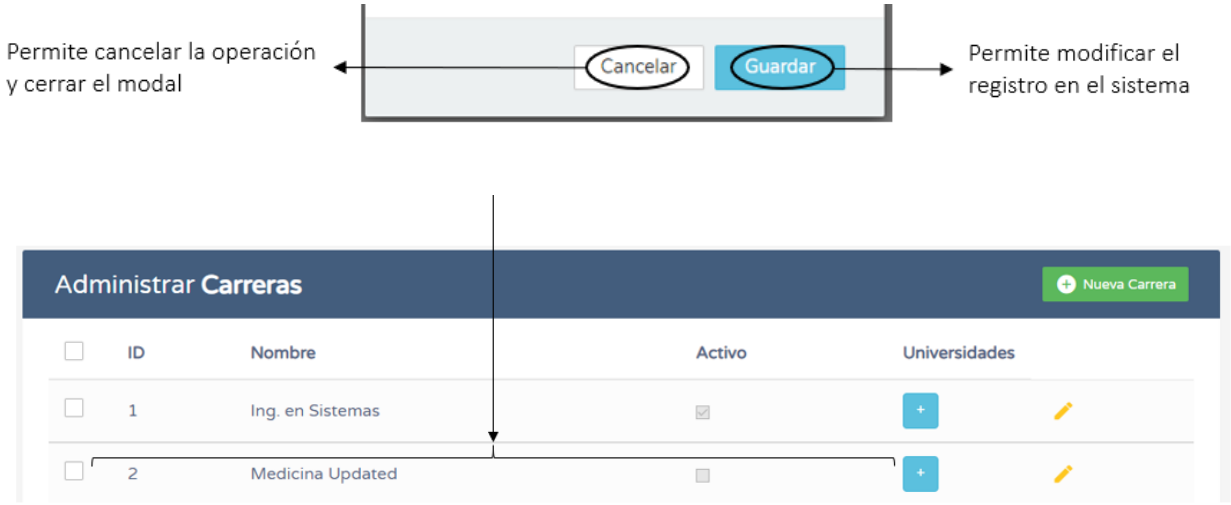
1. Seleccionar de la opción de Menú Catálogos > Carreras.
2. Presionar el símbolo de edición de la carrera que se desea editar.



3. Cambiar los datos que están activos para ser editados.

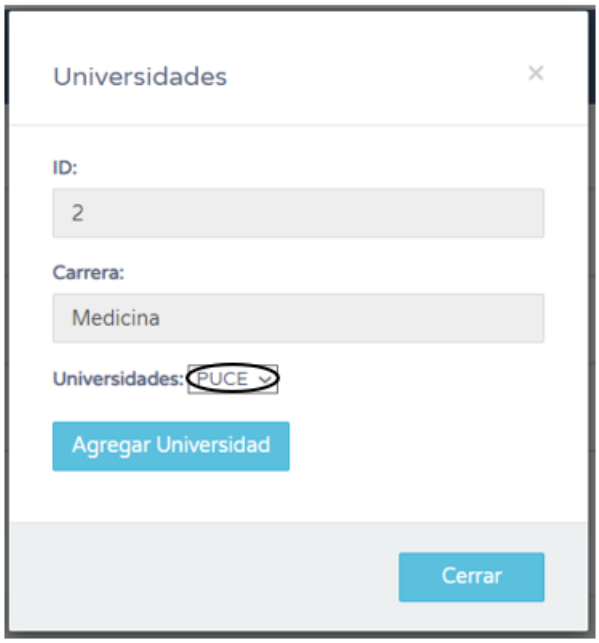


4. Presionar el botón Guardar.

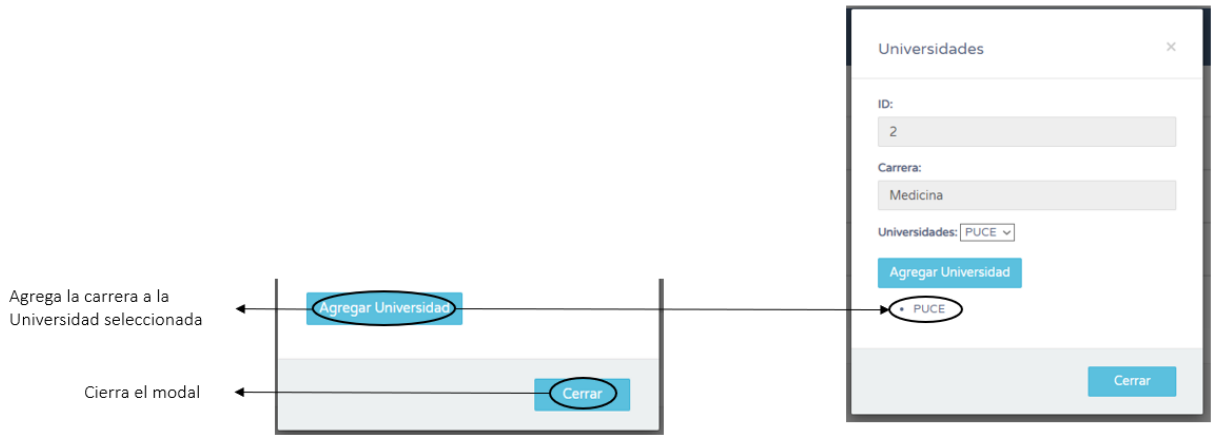


Asignar Universidades

1. Seleccionar de la opción de Menú Catálogos > Carreras.
2. Presionar el símbolo “+” de la columna Universidades de la carrera que se desea.
3. Seleccionar la universidad a la que se desee agregar.



4. Presionar el botón Agregar Universidad.

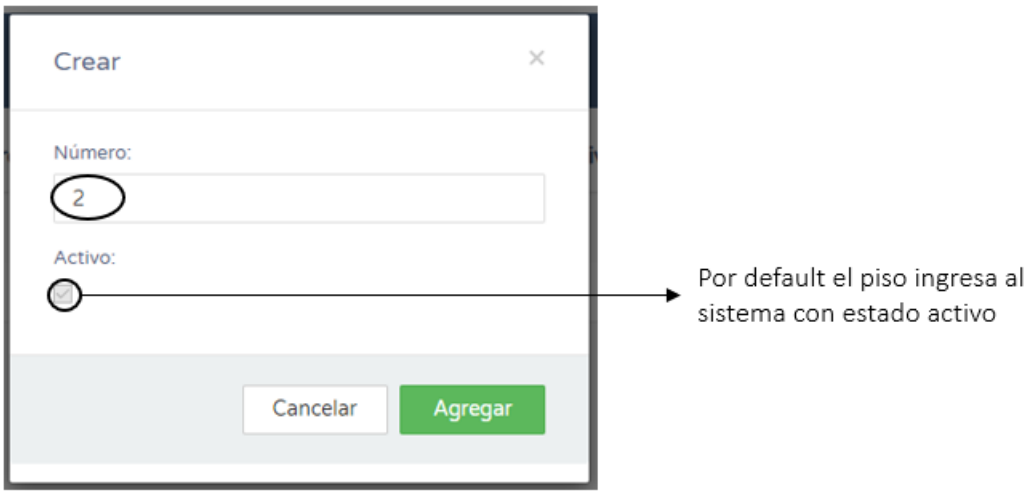


Pisos

Ingresar Piso

1. Seleccionar de la opción de Menú Catálogos > Pisos.
2. Seleccionar el botón Nuevo Piso.

3. Llenar los datos del modal.

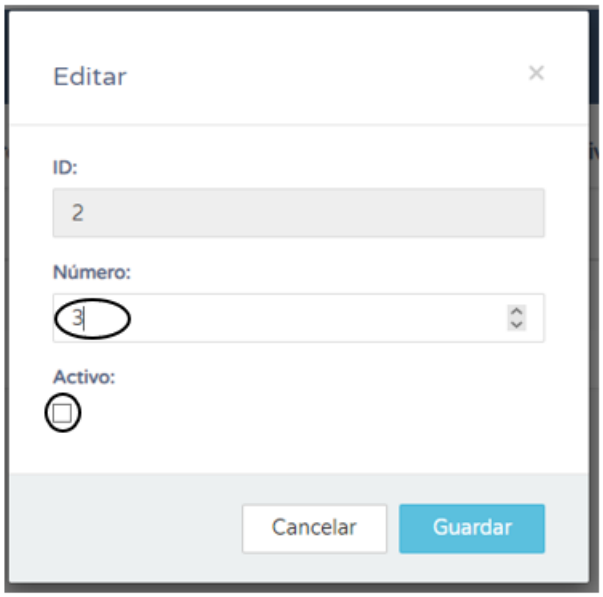


4. Presionar el botón Agregar.

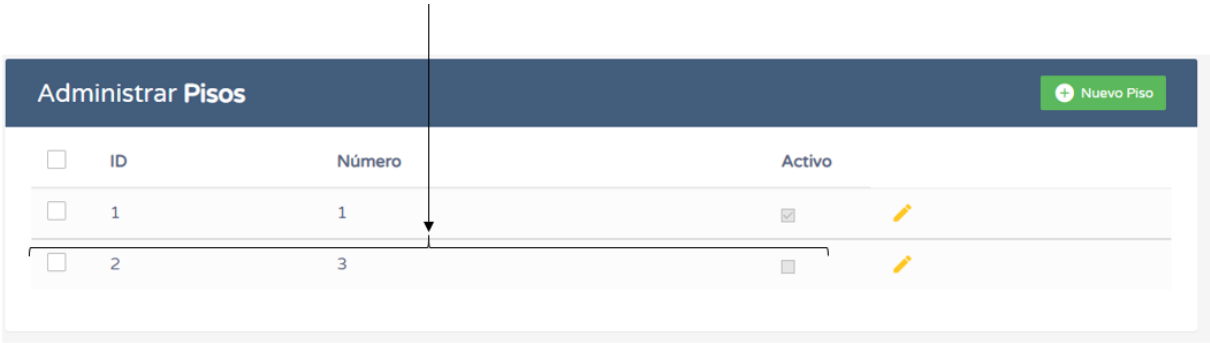
Modificar Piso

1. Seleccionar de la opción de Menú Catálogos > Pisos.
2. Presionar el símbolo de edición del piso que se desea editar.

3. Cambiar los datos que están activos para ser editados.



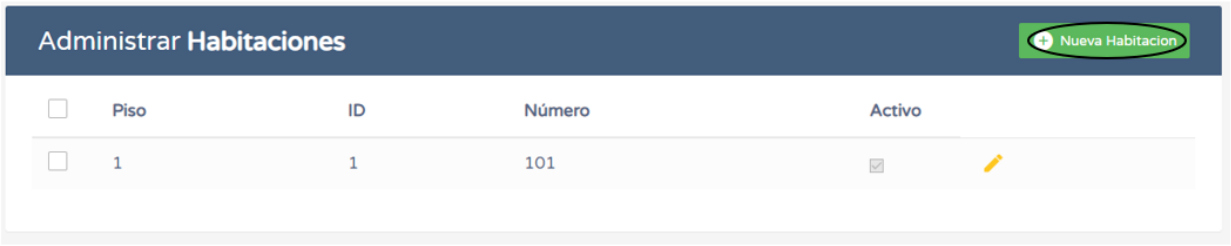
4. Presionar el botón Guardar.



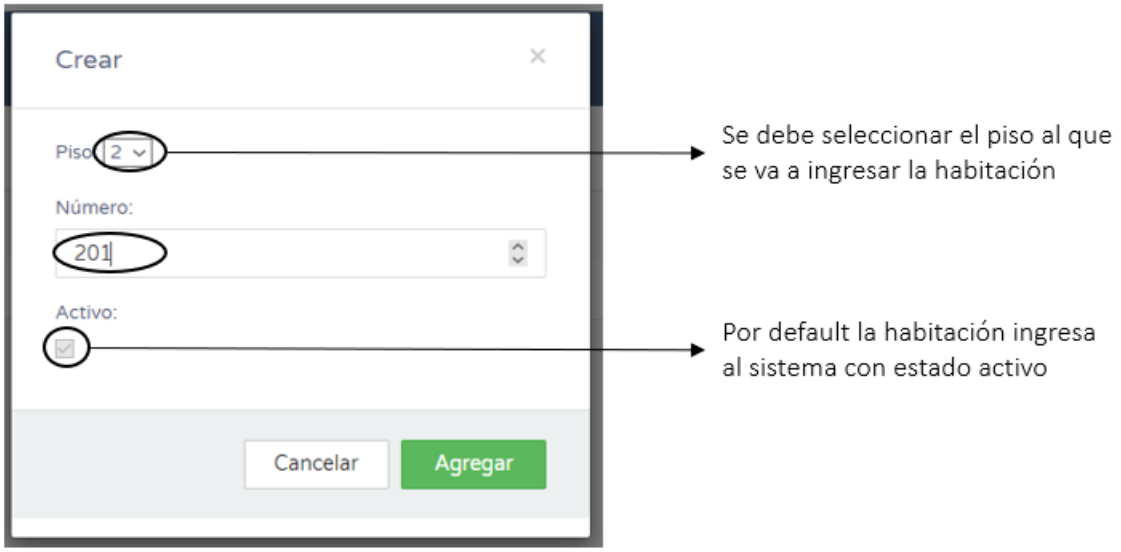
Habitaciones

Ingresar Habitación

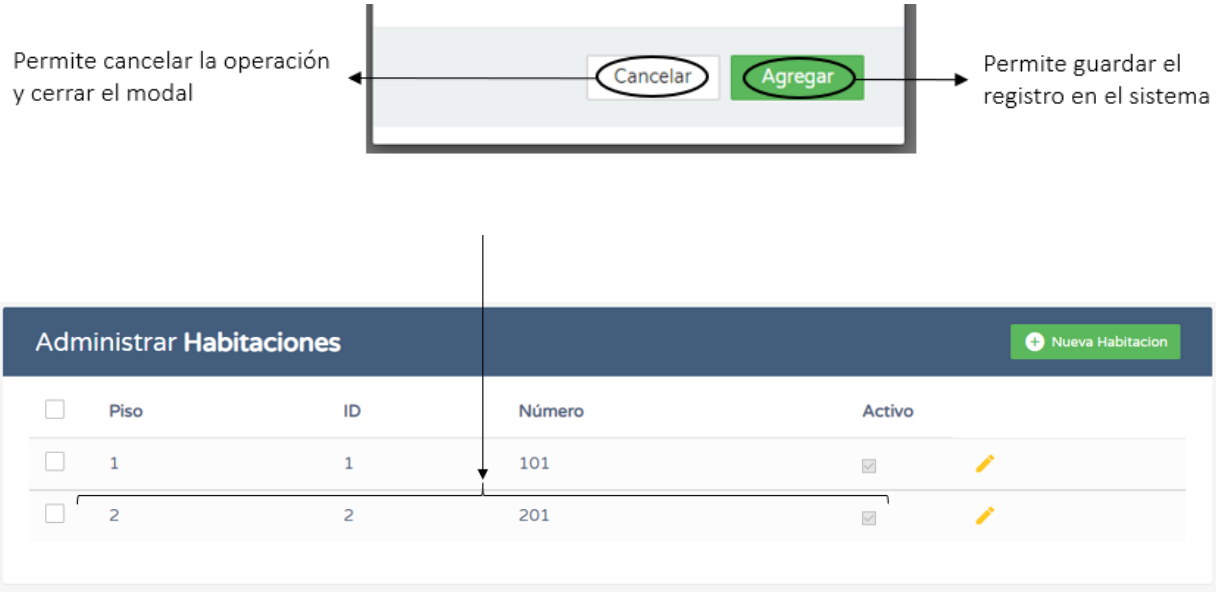
1. Seleccionar de la opción de Menú Catálogos > Habitaciones.
2. Seleccionar el botón Nueva Habitación.



3. Llenar los datos del modal.

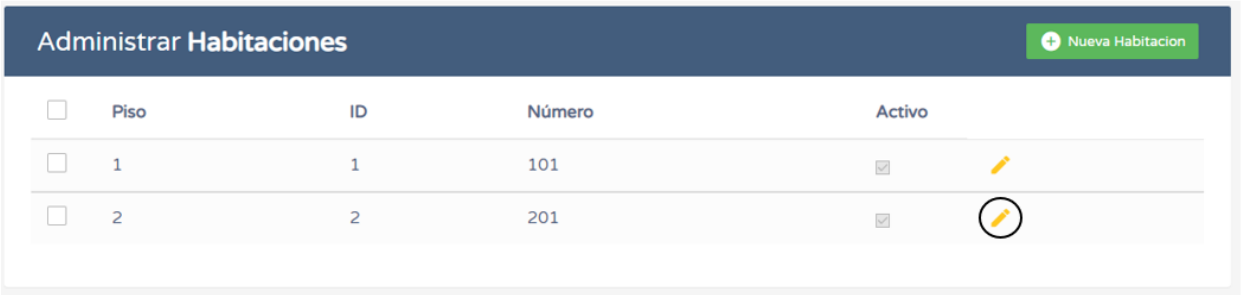


4. Presionar el botón Agregar.

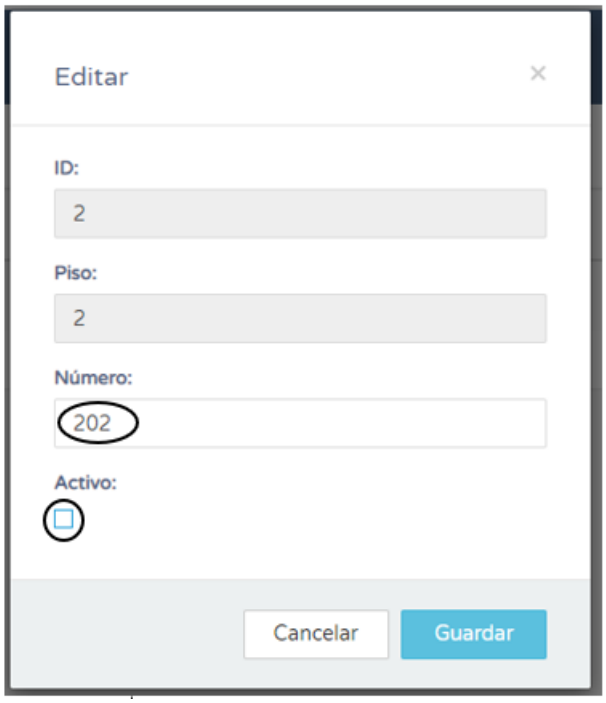


Modificar Habitación

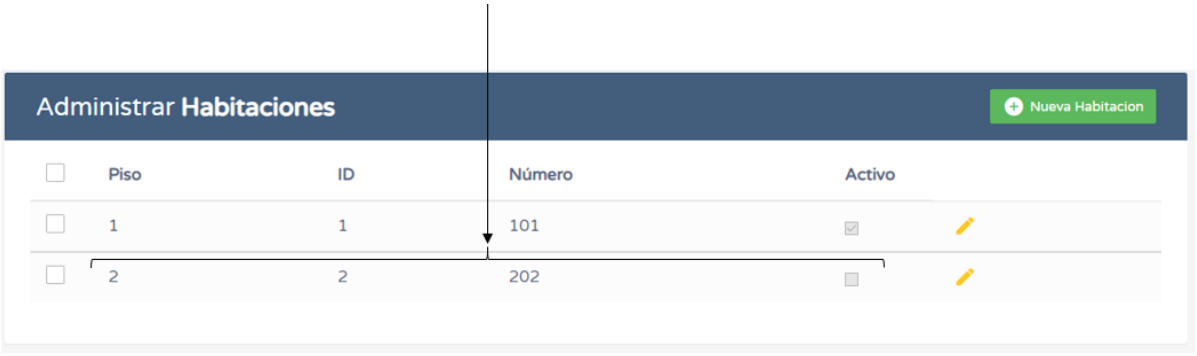
1. Seleccionar de la opción de Menú Catálogos > Habitaciones.
2. Presionar el símbolo de edición de la habitación que se desea editar.



3. Cambiar los datos que están activos para ser editados.



4. Presionar el botón Guardar.

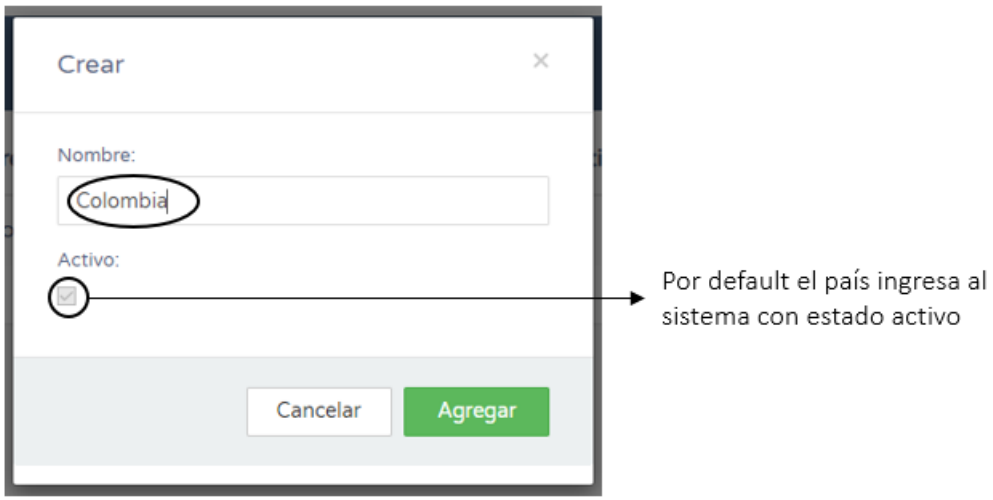


Países

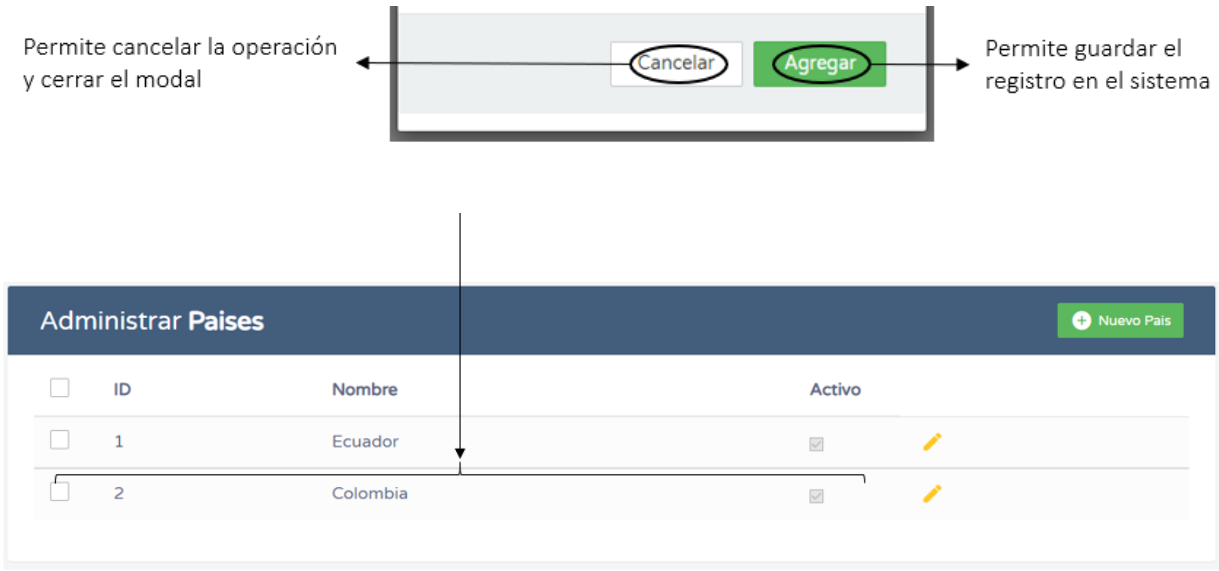
Ingresar País

1. Seleccionar de la opción de Menú Catálogos > Geografía > Países.
2. Seleccionar el botón Nuevo País.

3. Llenar los datos del modal.

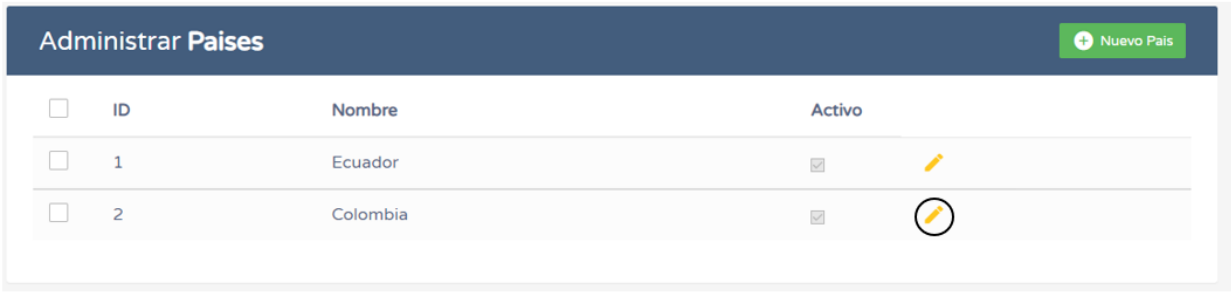


4. Presionar el botón Agregar.

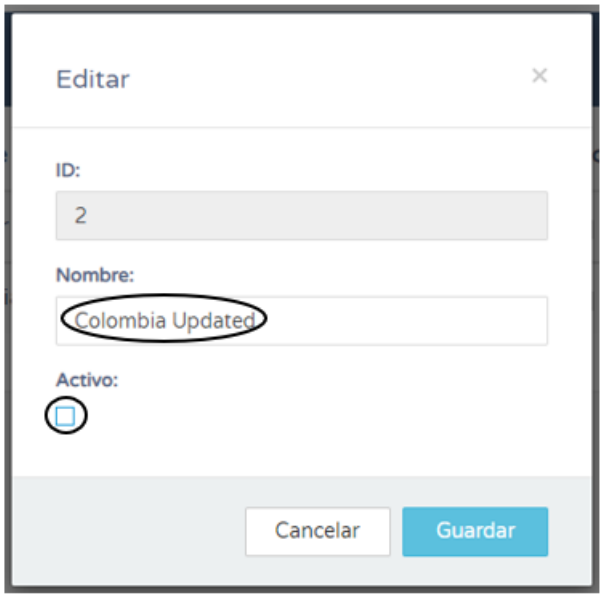


Modificar País

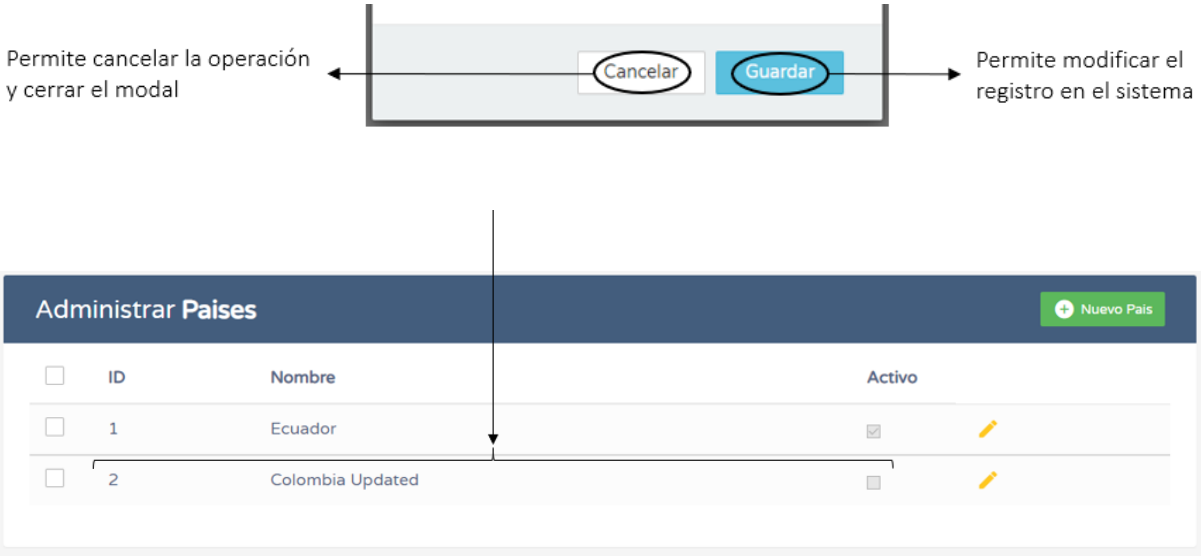
1. Seleccionar de la opción de Menú Catálogos > Geografía > Países.
2. Presionar el símbolo de edición del país que se desea editar.



3. Cambiar los datos que están activos para ser editados.



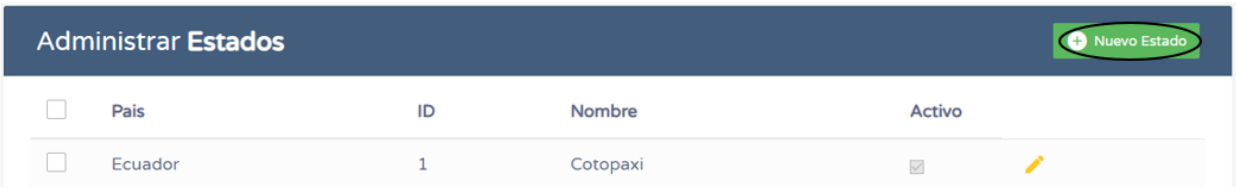
4. Presionar el botón Guardar.



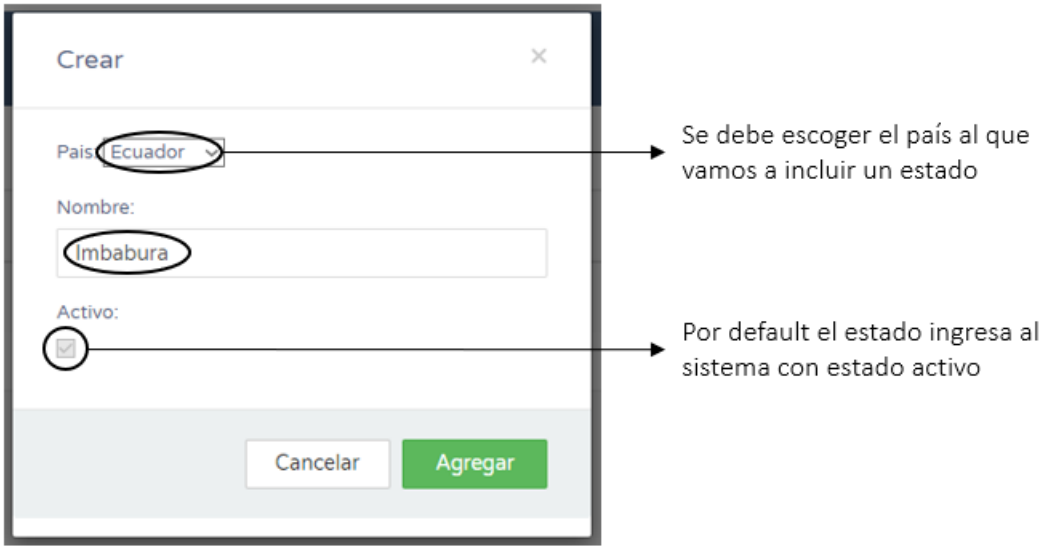
Estados

Ingresar Estado

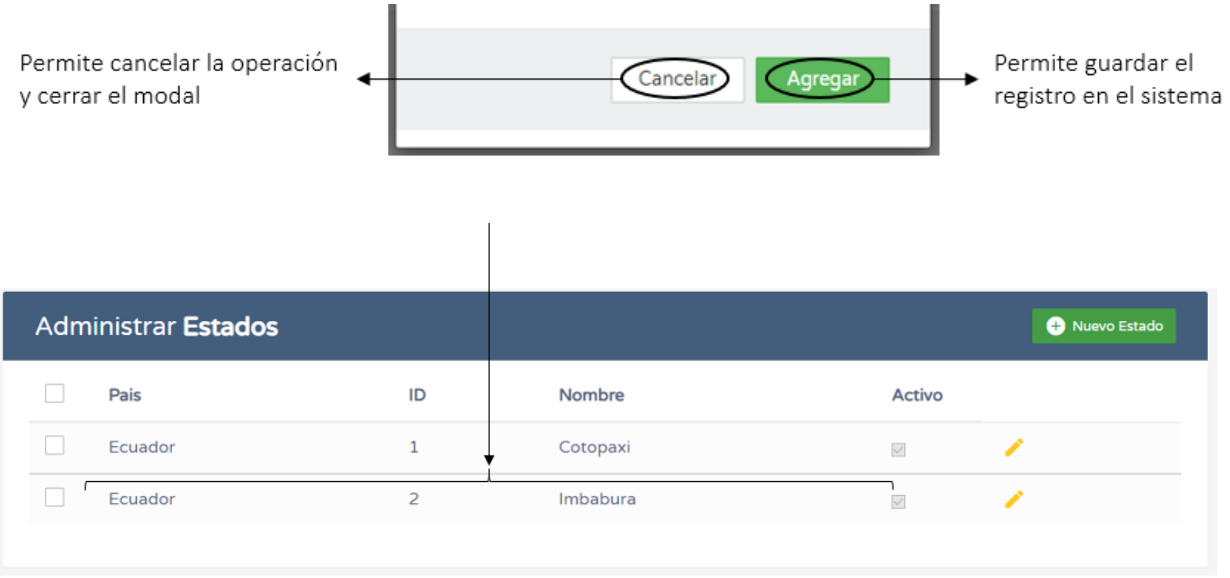
1. Seleccionar de la opción de Menú Catálogos > Geografía > Estados.
2. Seleccionar el botón Nuevo Estado.



3. Llenar los datos del modal.

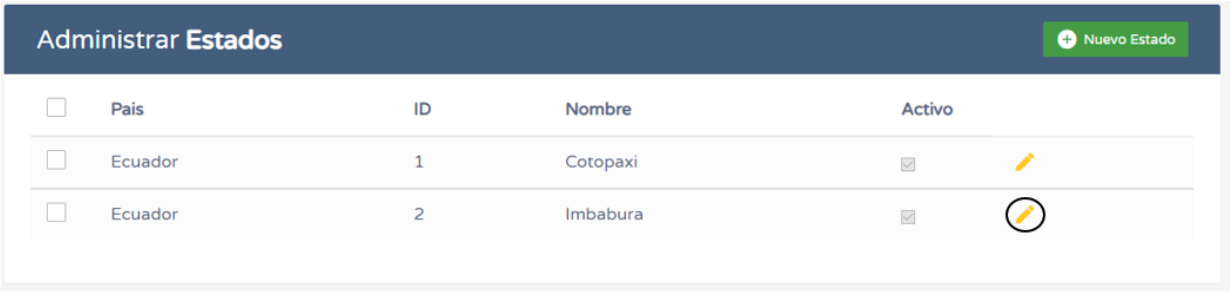


4. Presionar el botón Agregar.

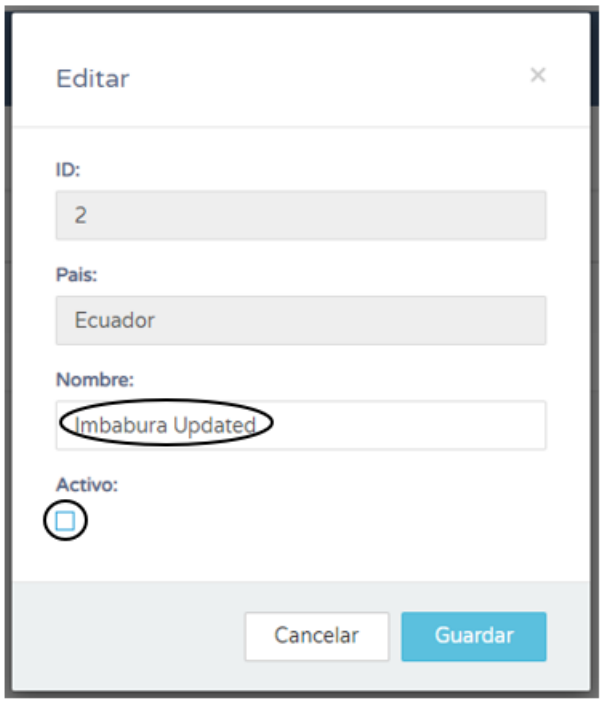


Modificar Estado

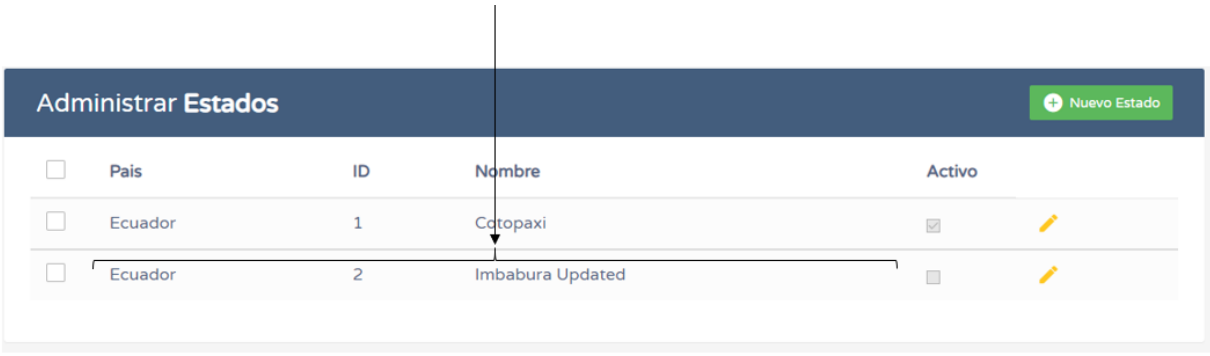
1. Seleccionar de la opción de Menú Catálogos > Geografía > Estados.
2. Presionar el símbolo de edición del estado que se desea editar.



3. Cambiar los datos que están activos para ser editados.



4. Presionar el botón Guardar.

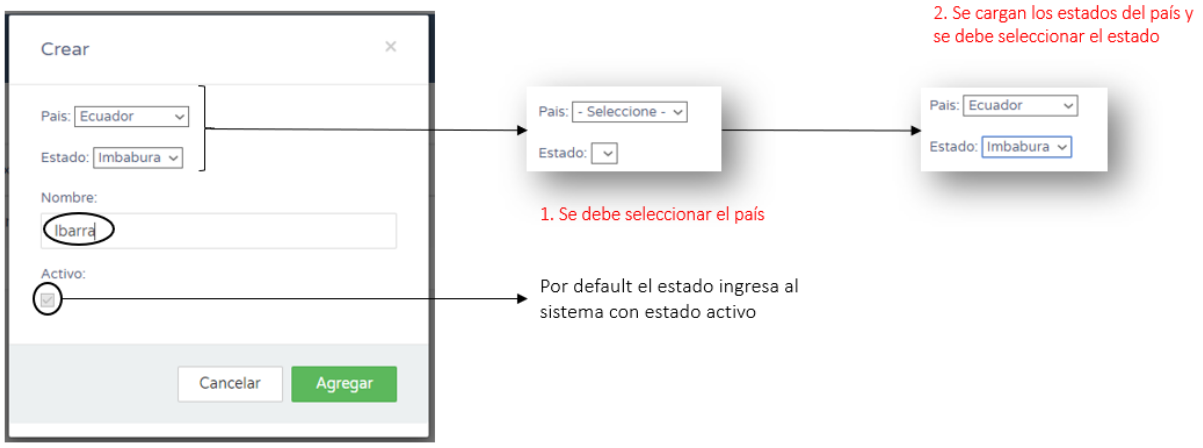


Ciudades

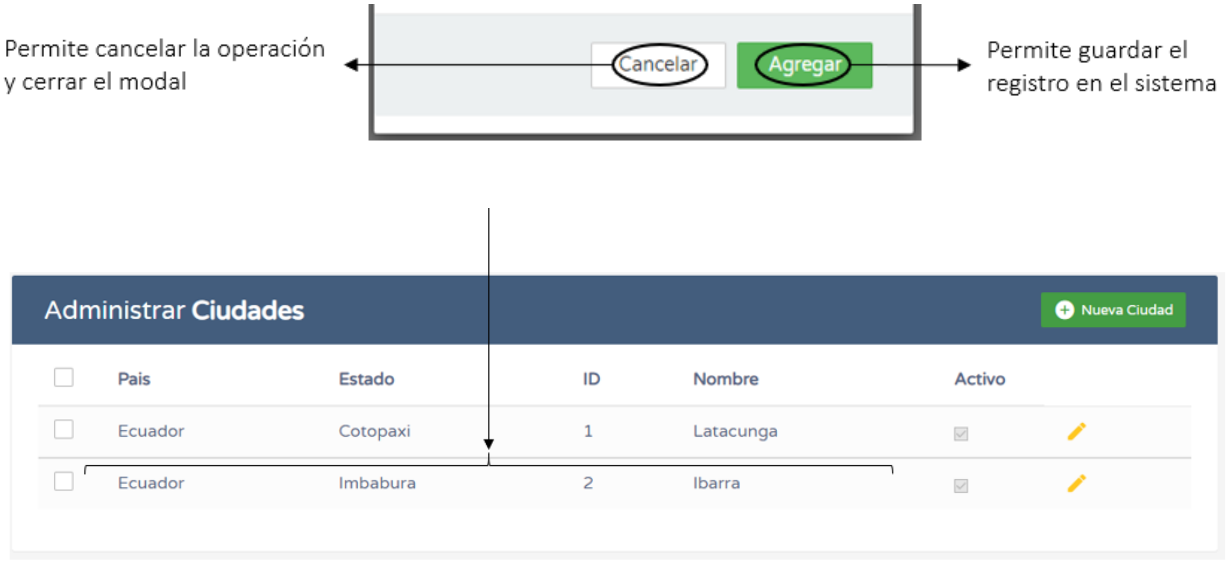
Ingresar Ciudad

1. Seleccionar de la opción de Menú Catálogos > Geografía > Ciudades.
2. Seleccionar el botón Nueva Ciudad.

3. Llenar los datos del modal.



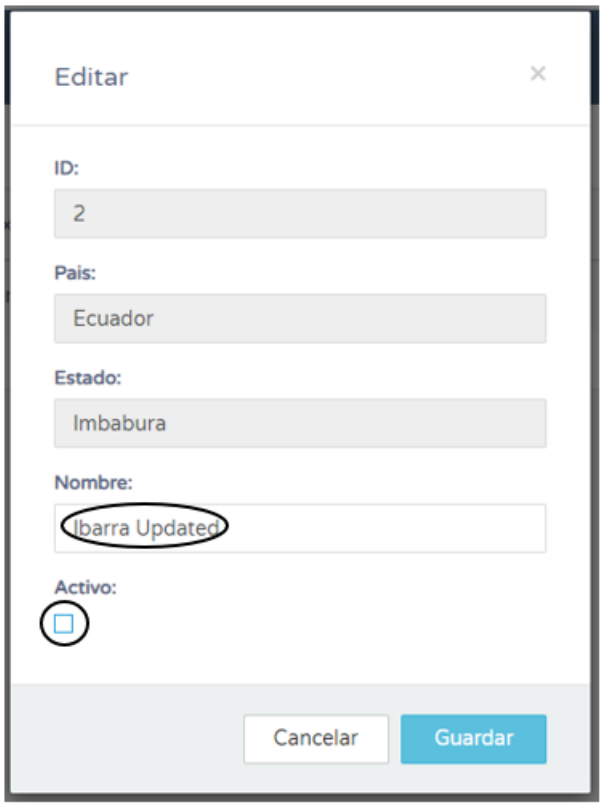
4. Presionar el botón Agregar.



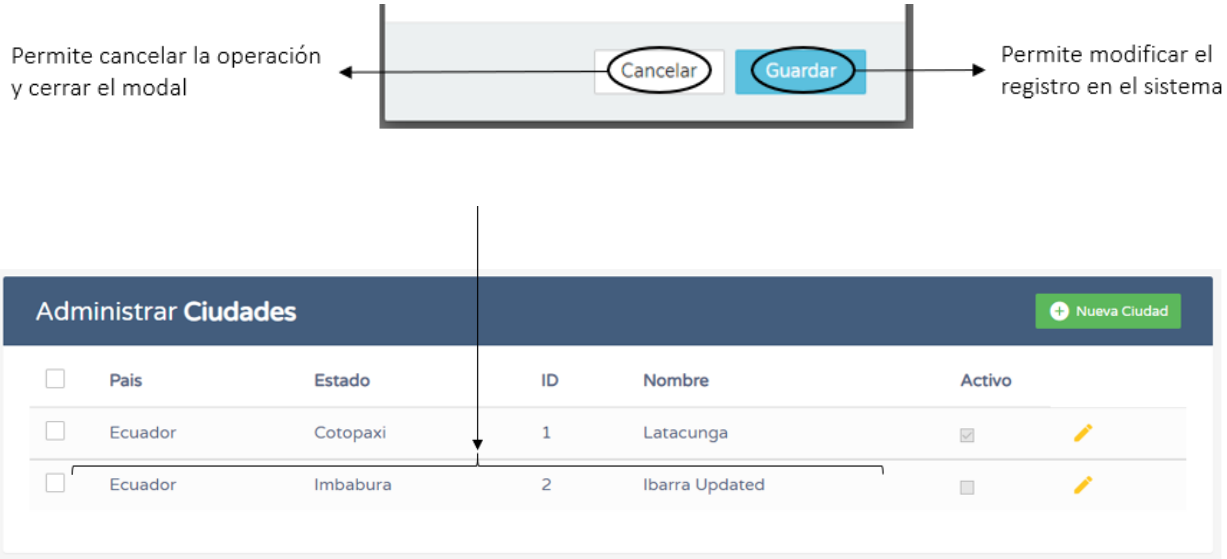
Modificar Ciudad

1. Seleccionar de la opción de Menú Catálogos > Geografía > Ciudades.
2. Presionar el símbolo de edición de la ciudad que se desea editar.

3. Cambiar los datos que están activos para ser editados.



4. Presionar el botón Guardar.

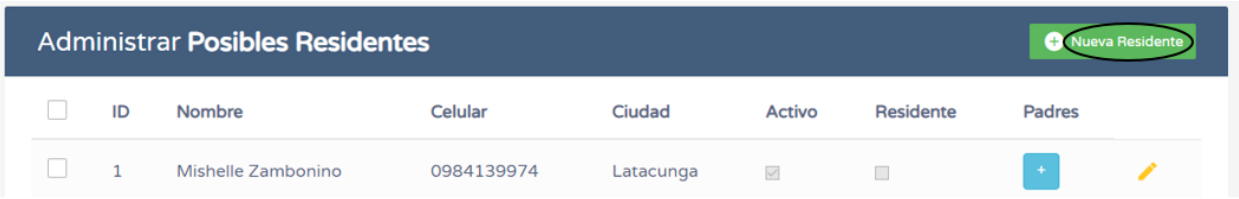


Administración de Residencia

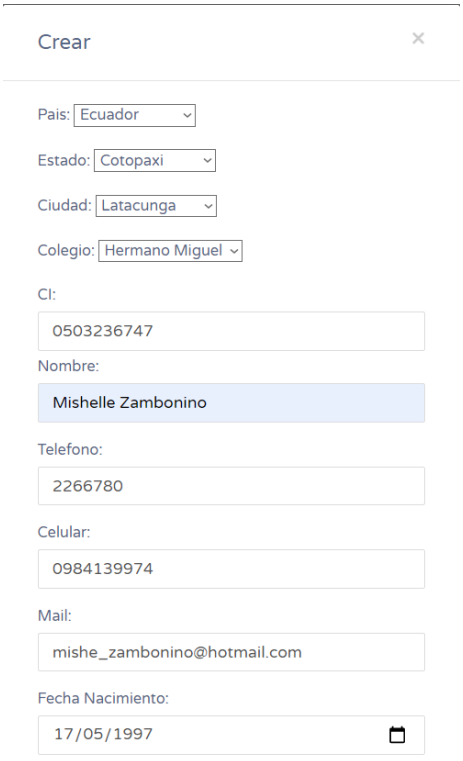
Posibles Residentes

Ingresar posible residente

1. Seleccionar de la opción de Menú Residencia > Posibles Residentes.
2. Seleccionar el botón Nueva Residente.



3. Llenar los datos del modal.

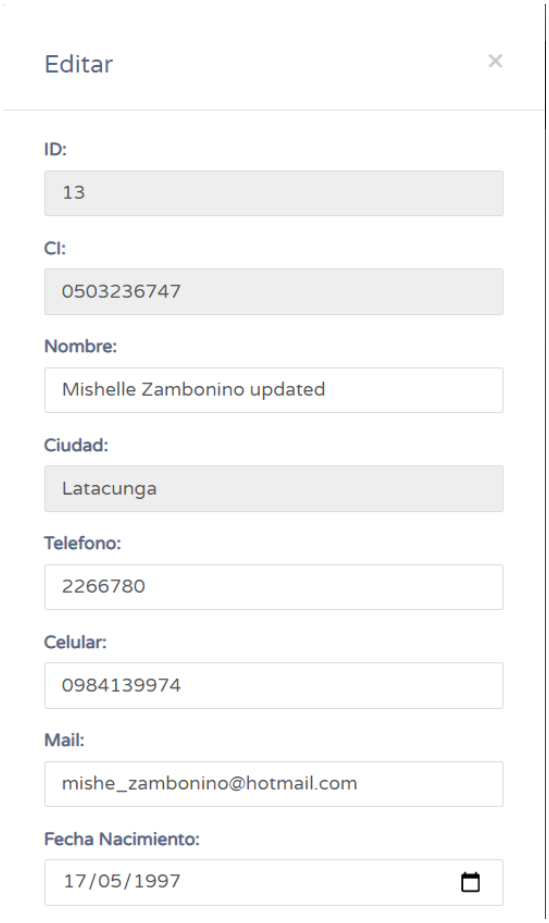


4. Presionar el botón Agregar.

Modificar posible residente

1. Seleccionar de la opción de Menú Residencia > Posibles Residentes.
2. Presionar el símbolo de edición de la residente que se desea editar.

3. Cambiar los datos que están activos para ser editados.


Editar
ID:
13
CI:
0503236747
Nombre:
Mishelle Zambonino updated
Ciudad:
Latacunga
Telefono:
2266780
Celular:
0984139974
Mail:
mishe_zambonino@hotmail.com
Fecha Nacimiento:
17/05/1997

4. Presionar el botón Guardar. Cuando se activa el check de Es residente, el registro pasa a la pantalla de residentes.

Ingresar/Modificar padre y madre

1. Seleccionar de la opción de Menú Residencia > Posibles Residentes.
2. Presionar el símbolo “+” de la columna Padres.

3. Agregar o modificar los datos de un padre o madre.

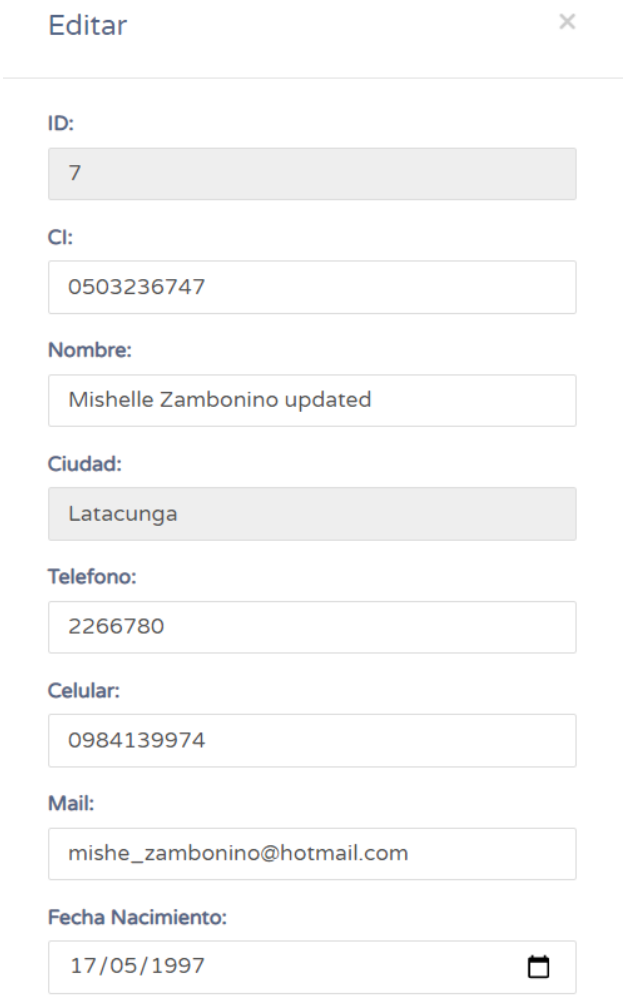
4. Presionar el botón Guardar. Cuando se activa el check de Es residente, el registro pasa a la pantalla de residentes.

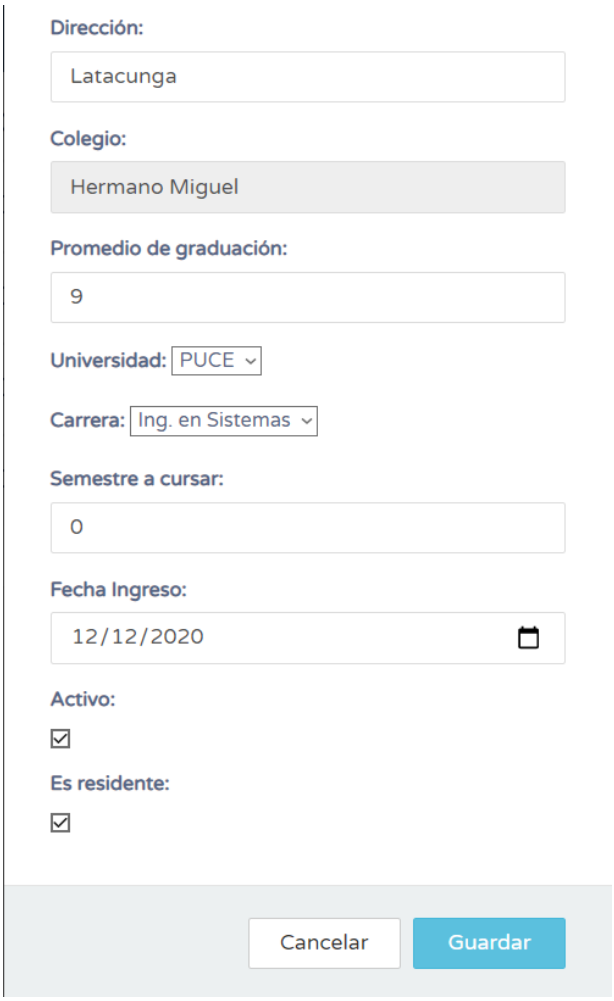
Residentes

Modificar residente

1. Seleccionar de la opción de Menú Residencia > Residentes.
2. Presionar el símbolo de edición de la residente que se desea editar.

3. Cambiar los datos que están activos para ser editados.





4. Presionar el botón Guardar.

Ingresar/Modificar padre y madre

1. Seleccionar de la opción de Menú Residencia > Residentes.
2. Presionar el símbolo “+” de la columna Padres.

3. Agregar o modificar los datos de un padre o madre.

4. Presionar el botón Guardar. Cuando se activa el check de Es residente, el registro pasa a la pantalla de residentes.

Asignar piso

1. Seleccionar de la opción de Menú Residencia > Residentes.
2. Presionar el símbolo “+” de la columna Habitación.

3. Asignar o modificar la habitación de la residente

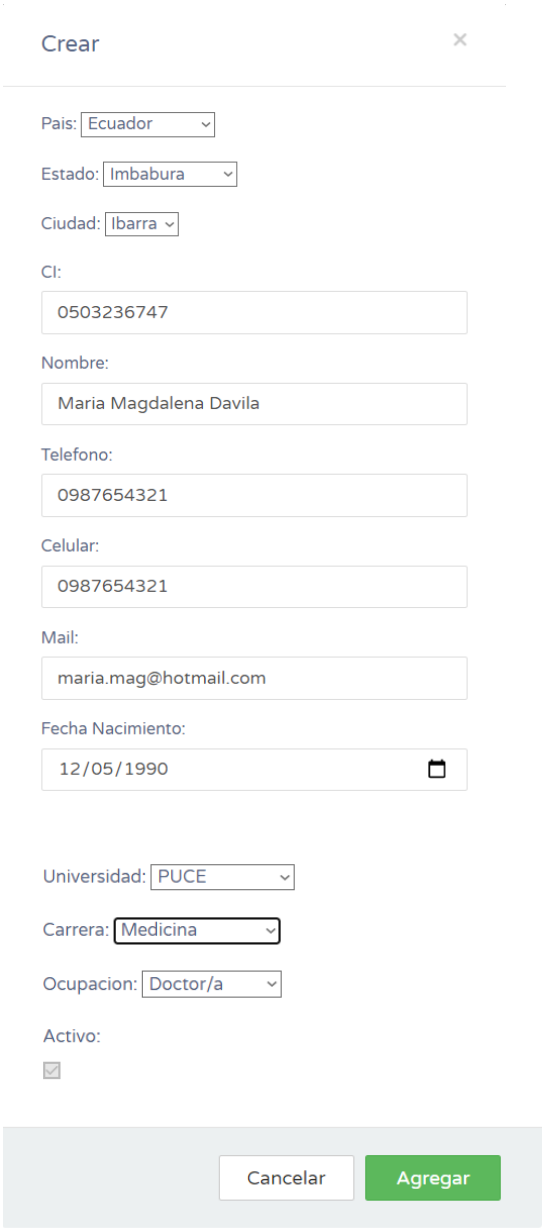
4. Presionar el botón Guardar. Cuando se activa el check de Es residente, el registro pasa a la pantalla de residentes.

Numerarias

Ingresar numeraria

1. Seleccionar de la opción de Menú Residencia > Numerarias.
2. Seleccionar el botón Nueva Numeraria.

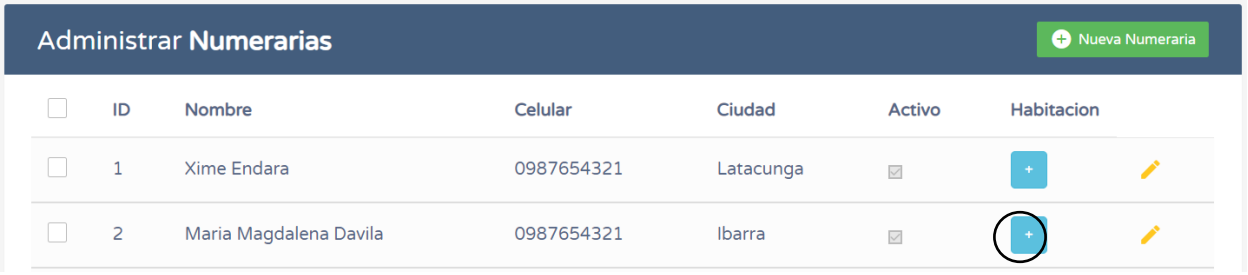
3. Llenar los datos del modal.



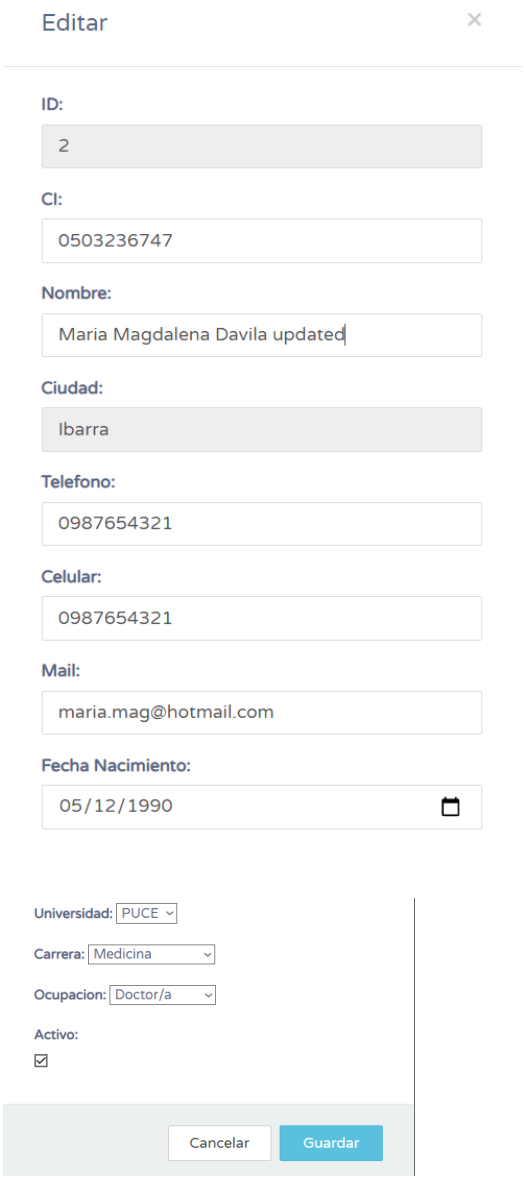
4. Presionar el botón Agregar.

Modificar Numeraria

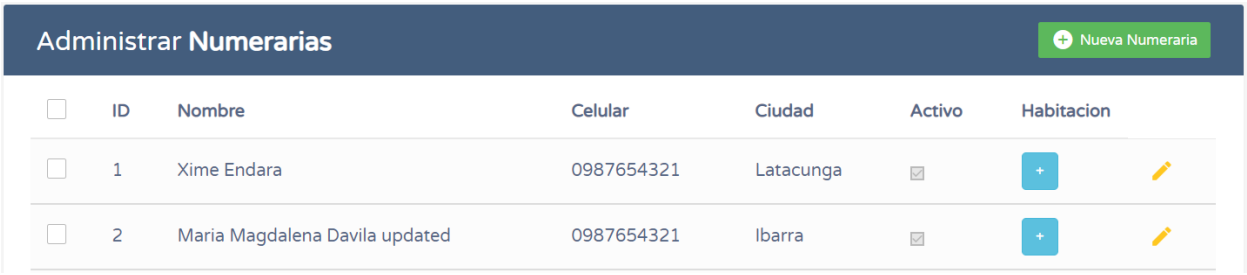
1. Seleccionar de la opción de Menú Residencia > Numerarias.
2. Presionar el símbolo de edición de la numeraria que se desea editar.



3. Cambiar los datos que están activos para ser editados.

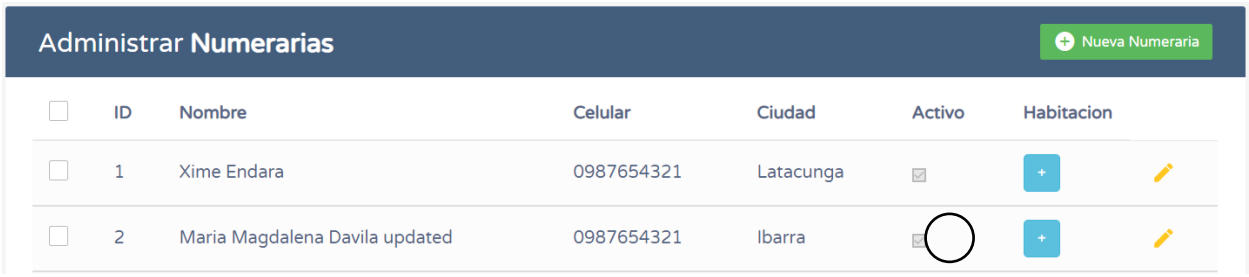


4. Presionar el botón Guardar.

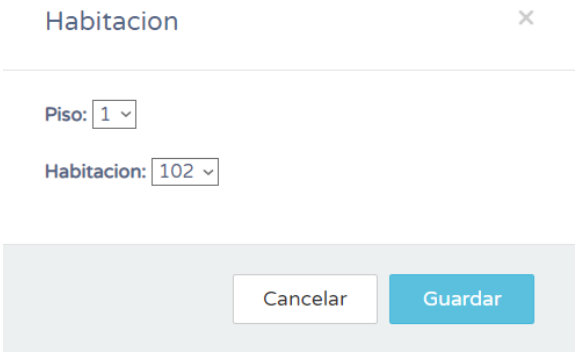


Asignar piso

1. Seleccionar de la opción de Menú Residencia > Residentes.
2. Presionar el símbolo “+” de la columna Habitación.



3. Asignar o modificar la habitación de la numeraria.

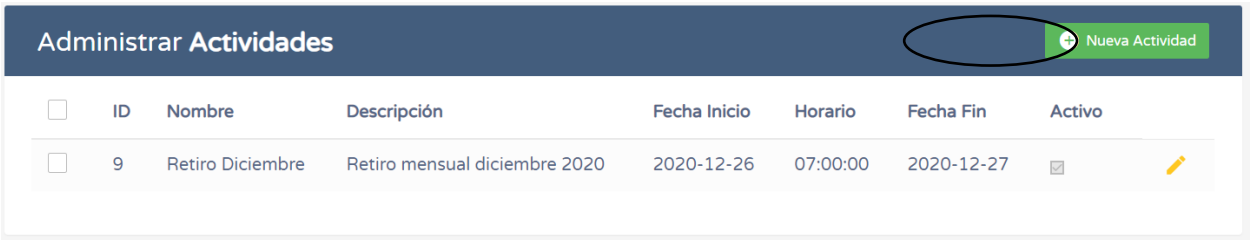


4. Presionar el botón Guardar. Cuando se activa el check de Es residente, el registro pasa a la pantalla de residentes.

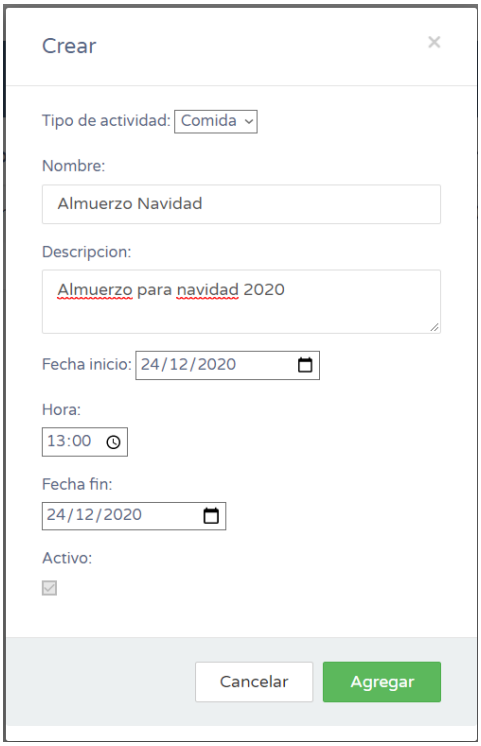
Actividades

Ingresar actividad

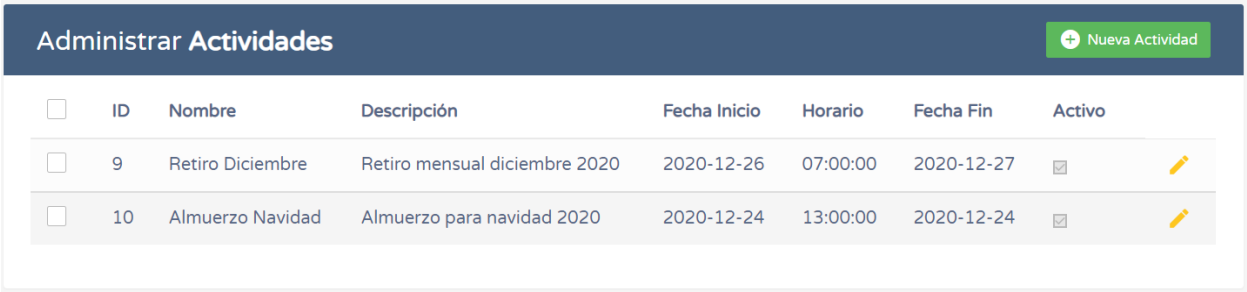
1. Seleccionar de la opción de Menú Residencia > Actividades.
2. Seleccionar el botón Nueva Actividad.



3. Llenar los datos del modal.

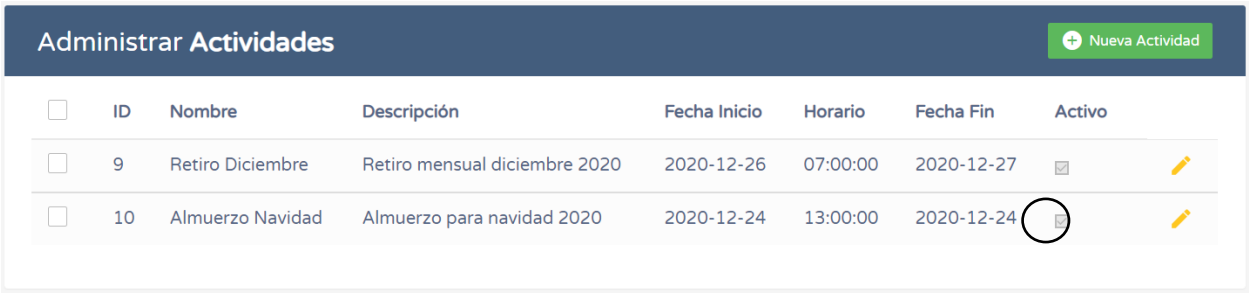


4. Presionar el botón Agregar.

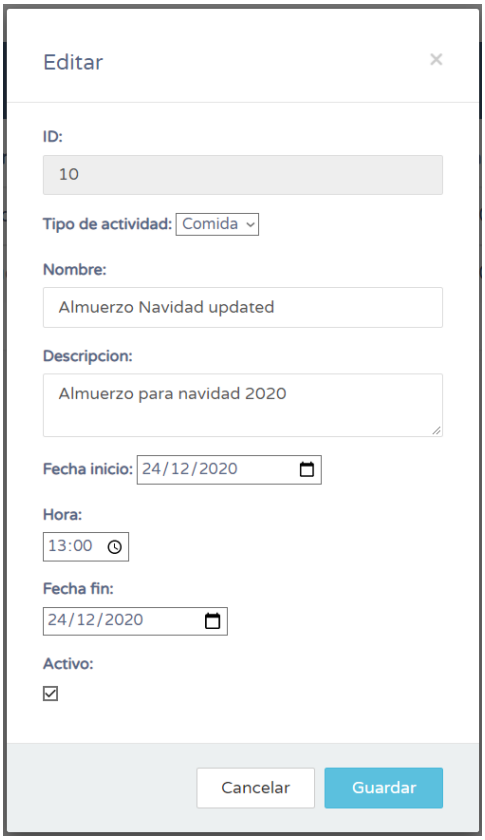


Modificar actividad

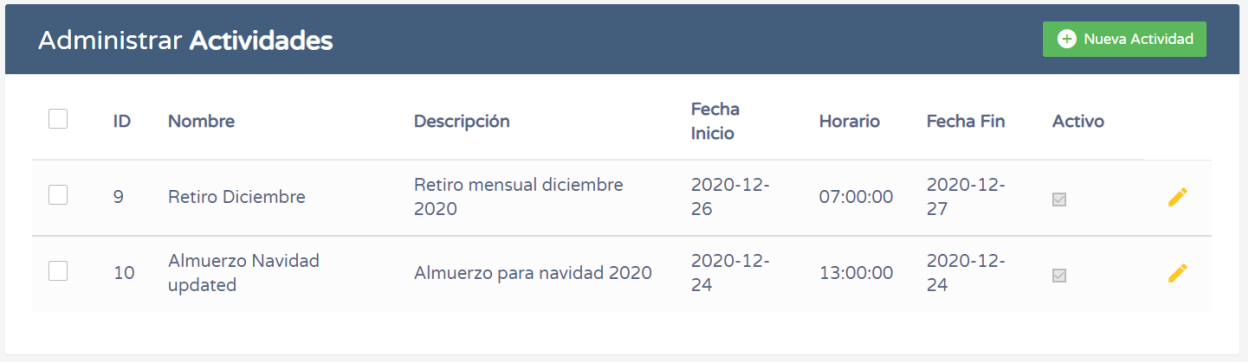
1. Seleccionar de la opción de Menú Residencia > Actividades.
2. Presionar el símbolo de edición de la actividad que se desea editar.



3. Cambiar los datos que están activos para ser editados.

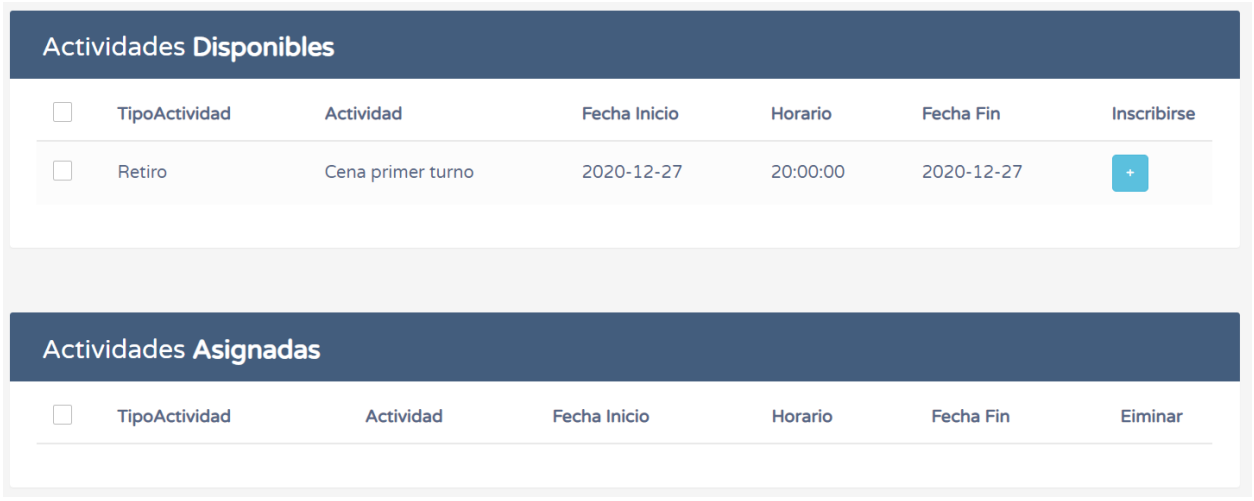


4. Presionar el botón Guardar.

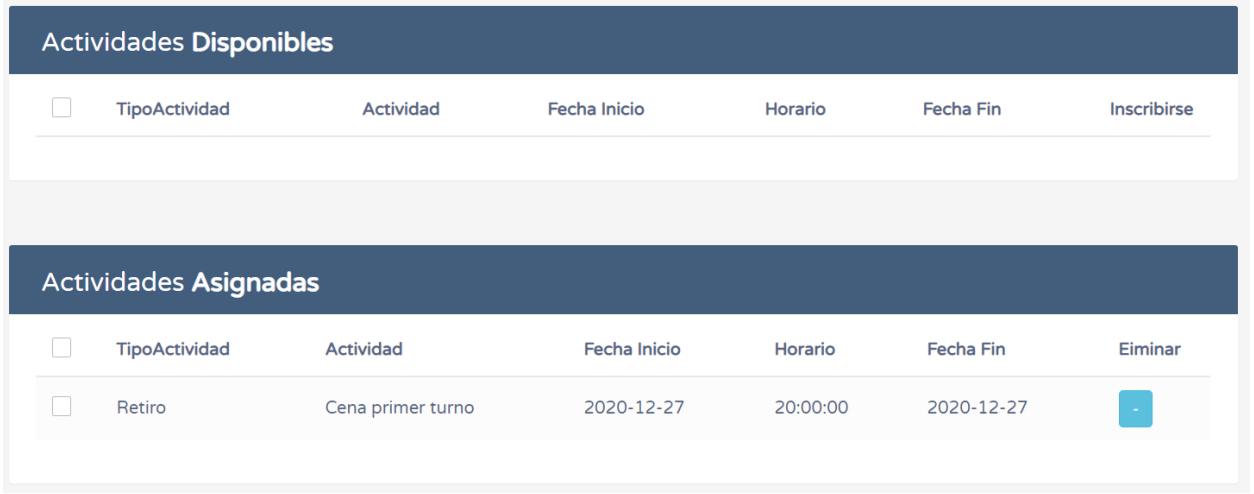


Registrar Actividad

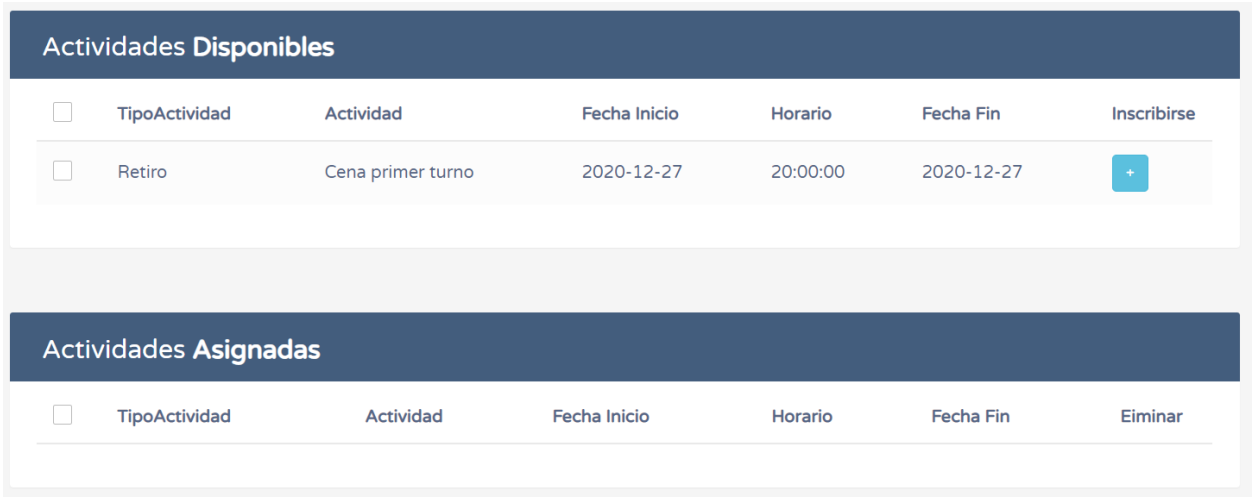
1. Seleccionar de la opción de Menú Registrar actividad.
2. Se presentarán dos tablas, una con actividades disponibles, y otra con las actividades en las que el usuario se haya registrado.



3. Para poder inscribirse se debe dar clic en el botón "+" de la columna Inscribirse de la tabla Actividades disponibles.
4. La actividad se eliminará de la tabla Actividades disponibles y se aumentará en la tabla Actividades asignadas.



5. Para poder eliminar una actividad, se debe dar clic en el botón "+" de la columna Eliminar de la tabla Actividades Asignadas.
6. La actividad se eliminará de la tabla Actividades asignadas y se aumentará en la tabla Actividades disponibles.



Reporte Actividades

Esta opción está disponible únicamente para el administrador

1. Seleccionar de la opción de menú Reporte Actividades
2. Se presenta un reporte con las personas que se han registrado en las actividades del día actual

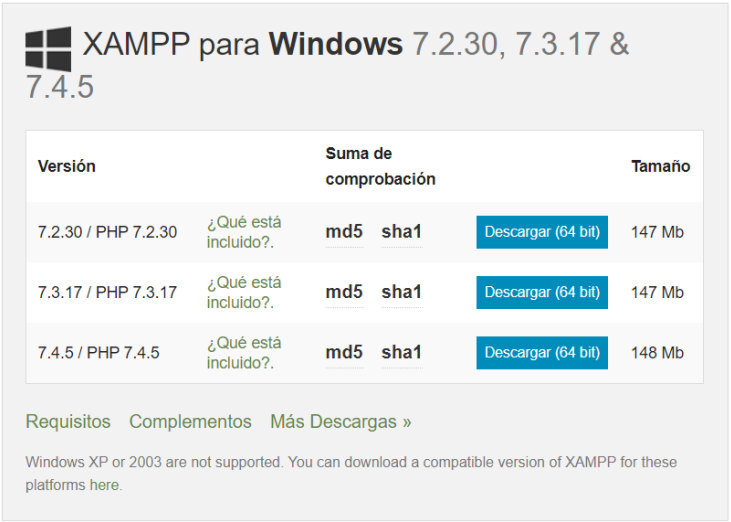
7.5 Anexo 5: Manual técnico

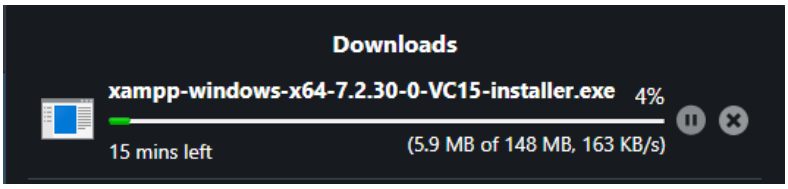
Manual técnico de instalación

En el siguiente manual se presentarán los pasos necesarios para poder instalar las herramientas que ayudaron al desarrollo de la aplicación, estos pasos deben ser aplicados usando el sistema operativo Windows 10.

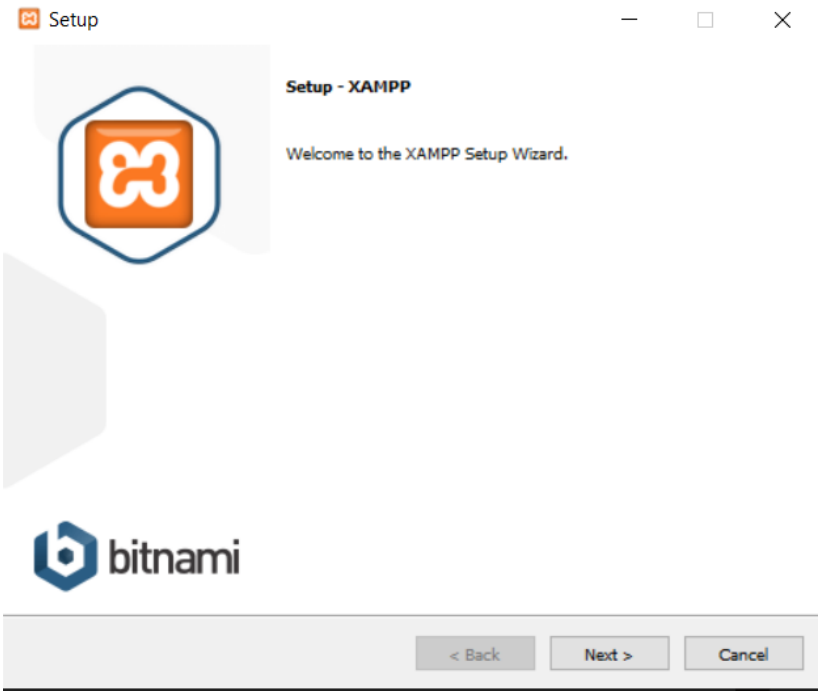
Instalación xampp

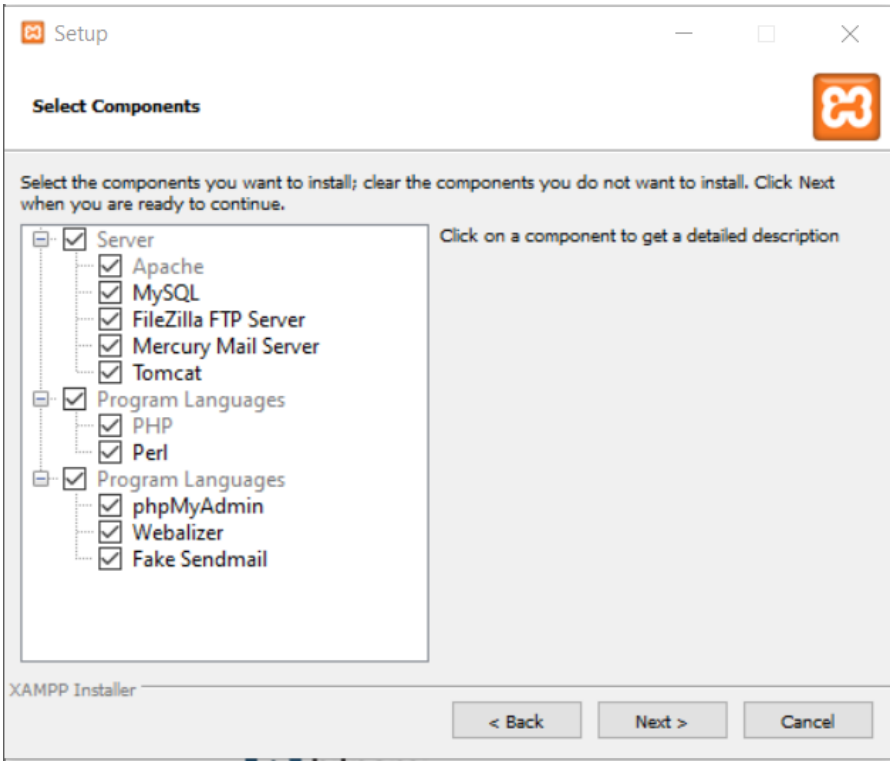
1. Ingresar a la url https://www.apachefriends.org/es/download.html
2. En la parte de Windows, descargar la versión 7.2.30 / PHP 7.2.30.



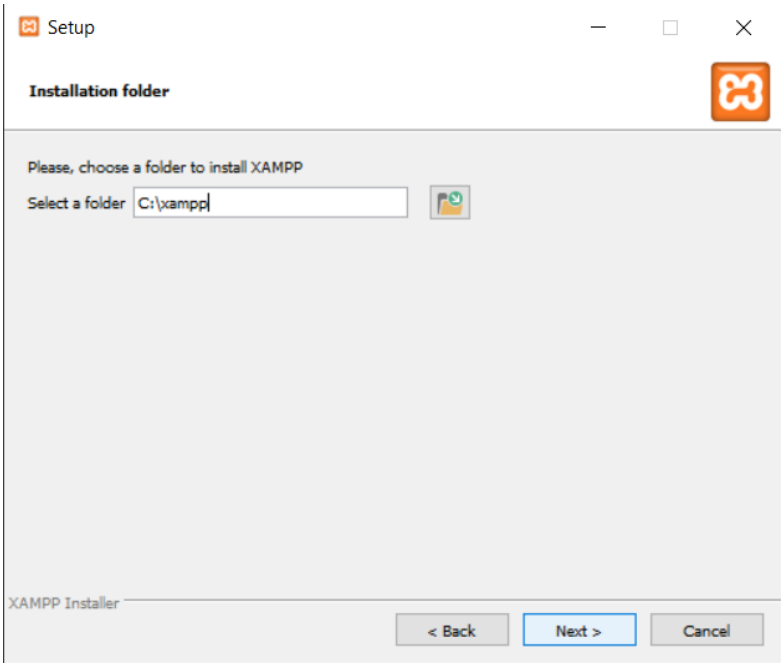


3. Una vez descargado el archivo con extensión “.exe” se debe ejecutar.

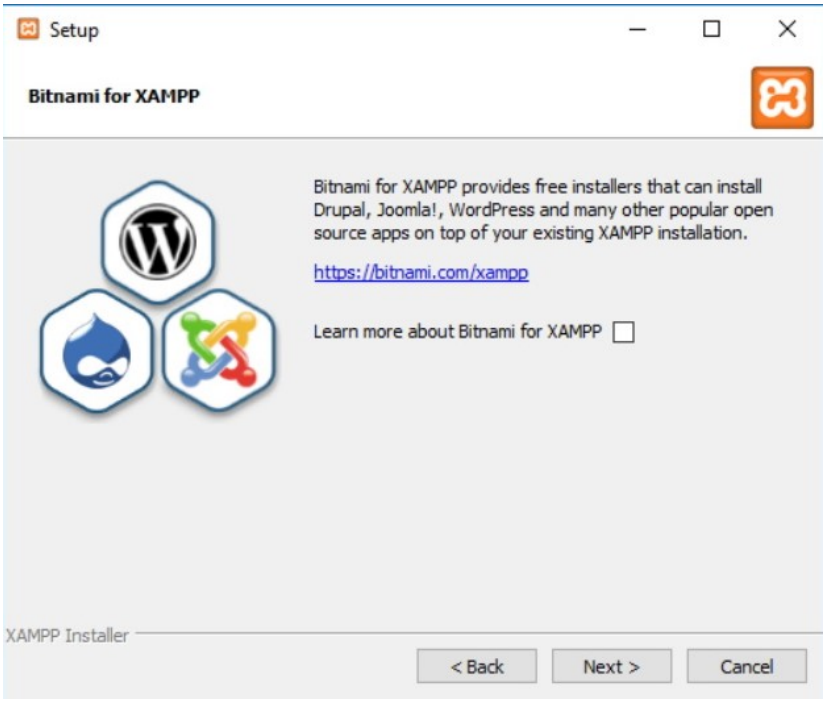


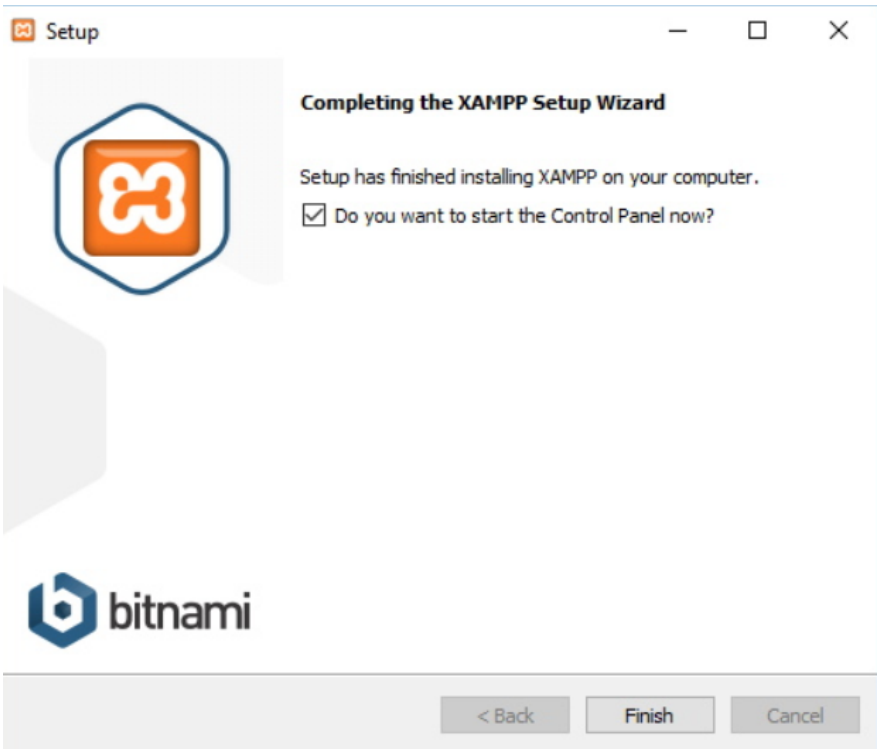


4. Seleccionar lo que se desea instalar, no olvidar verificar que la opción de MySQL esté activa, porque es la base de datos que usaremos.



5. Seleccionar la carpeta donde se va a instalar.
6. Continuar con el proceso hasta la pantalla de finalización.





Instalación MySQL

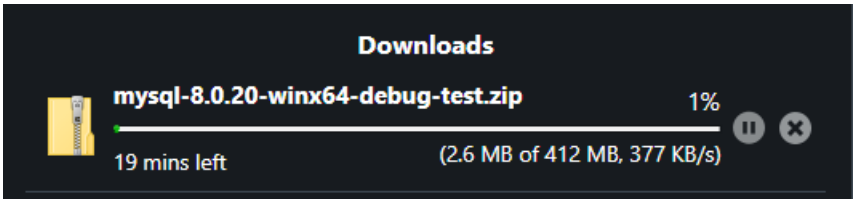
1. Ingresar a la url https://dev.mysql.com/downloads/
2. Seleccionar esta opción

   - MySQL Community Server

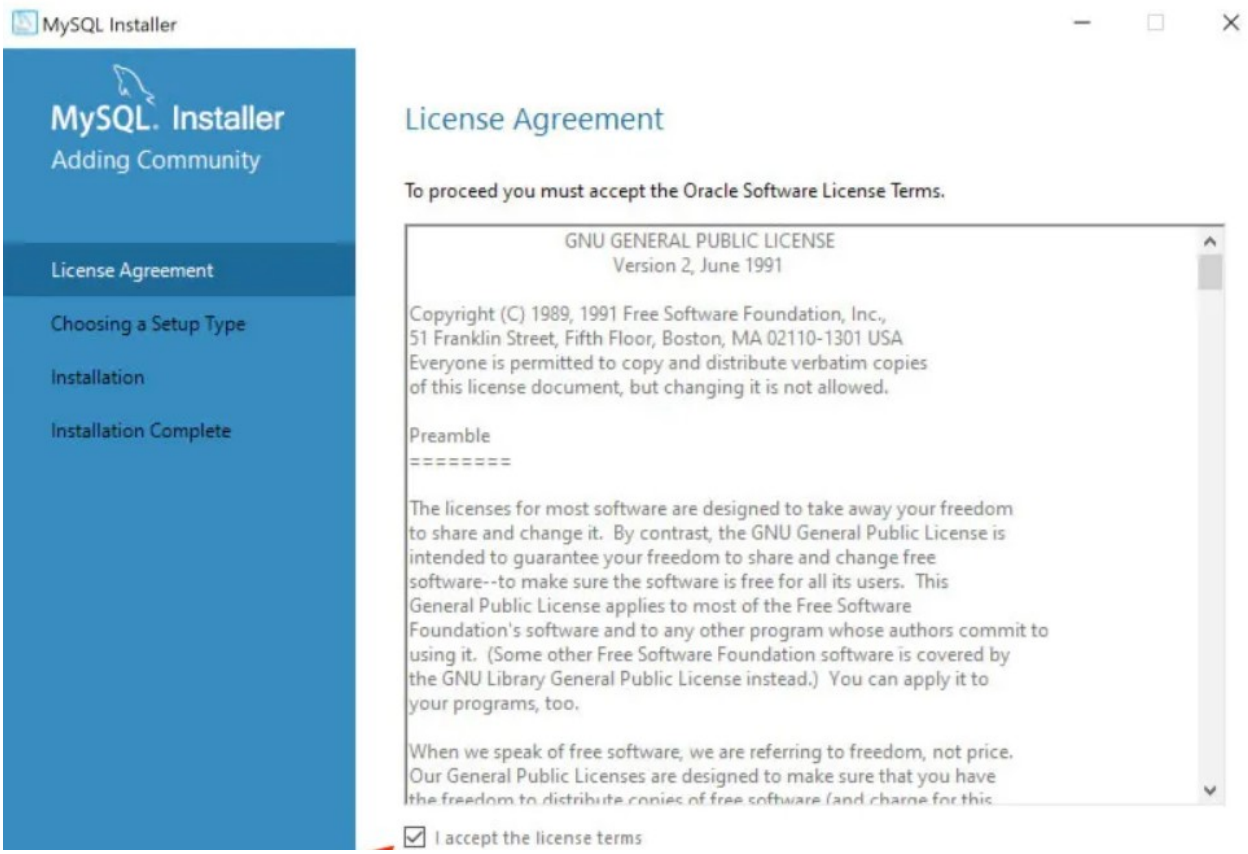
3. Seleccionar esta opci´ón y dar clic en Download.

4. Clic en la siguiente opción.

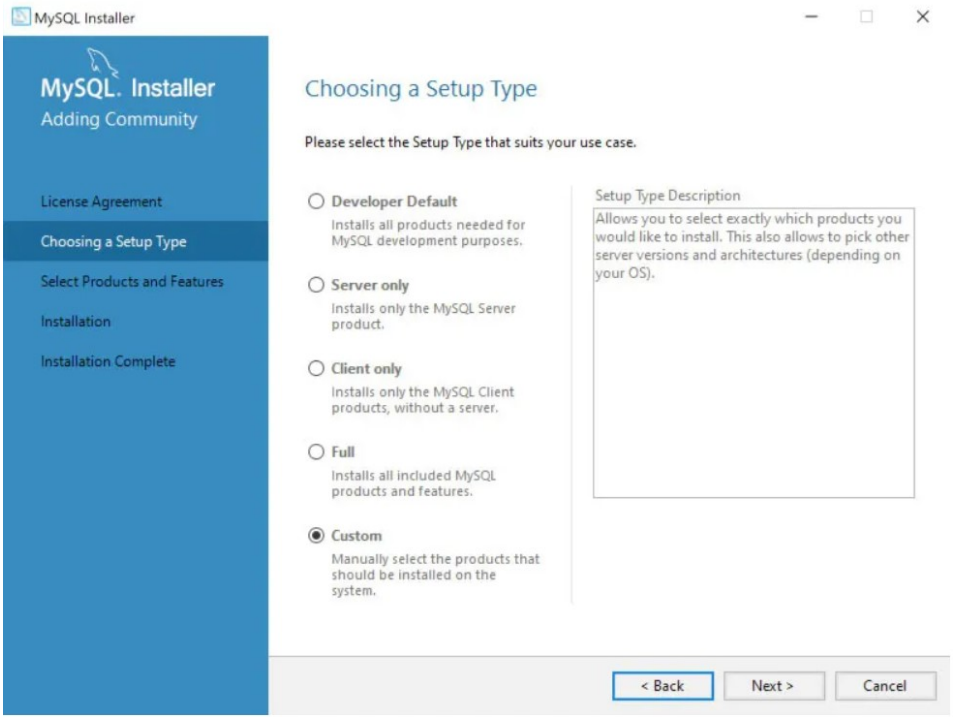
   No thanks, just start my download.



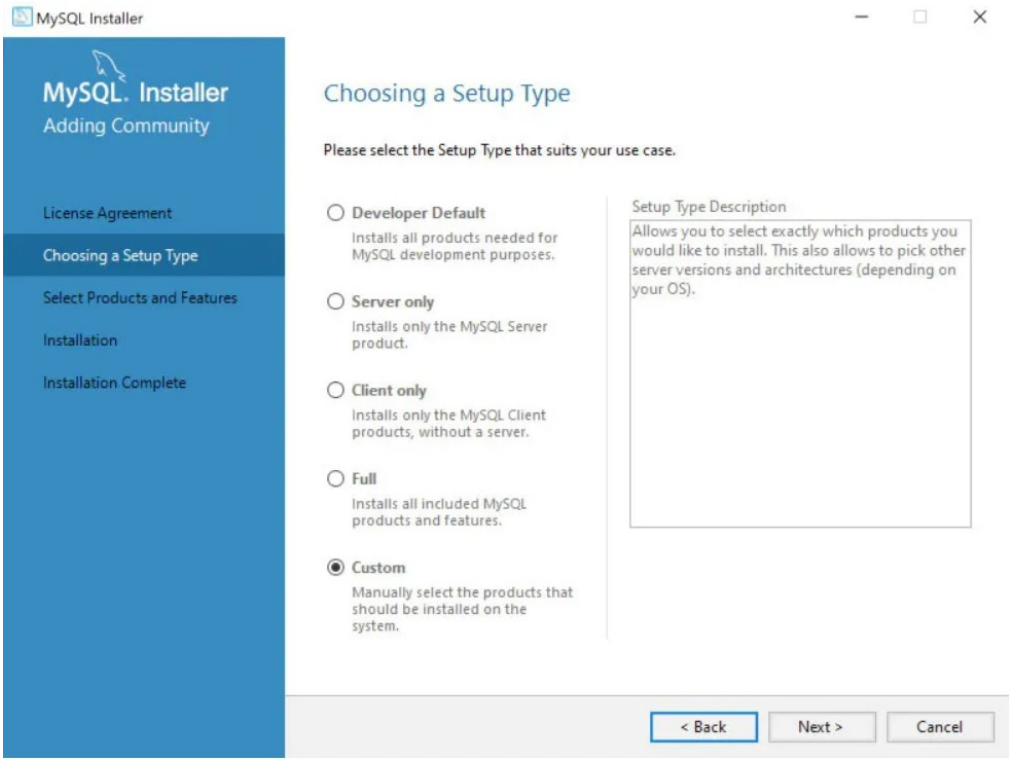
5. Una vez descargado, ejecutar el archivo ".msi". Se deben aceptar los términos y condiciones.



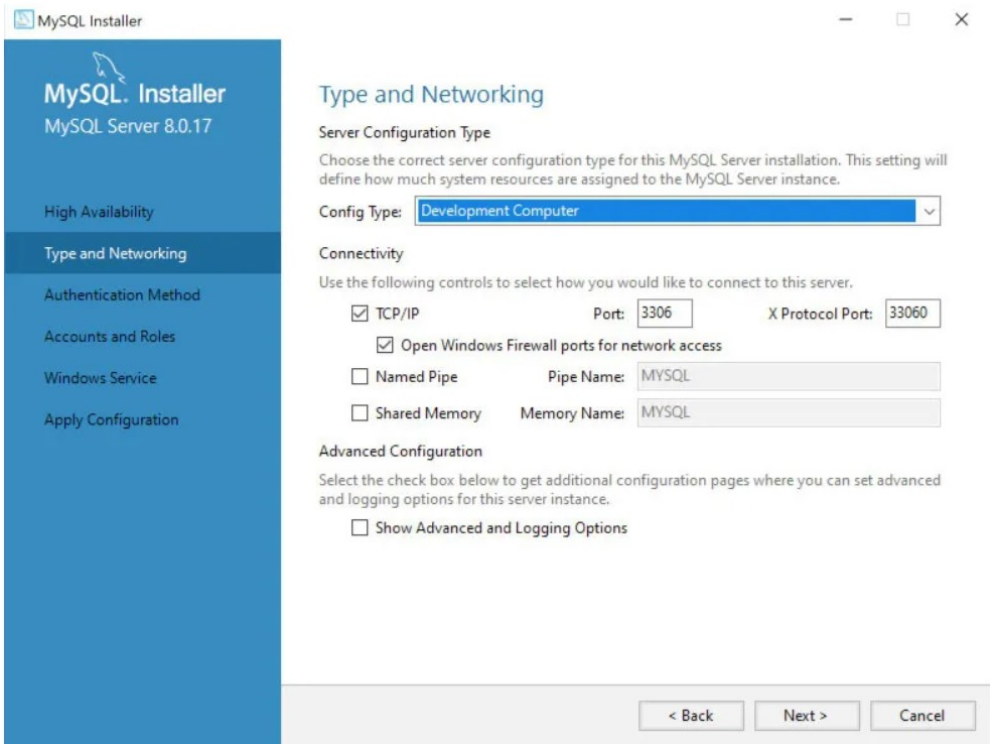
6. Se debe seleccionar la opción para escoger lo que se va a instalar



7. Se debe verificar que se intale MySQL Workbench



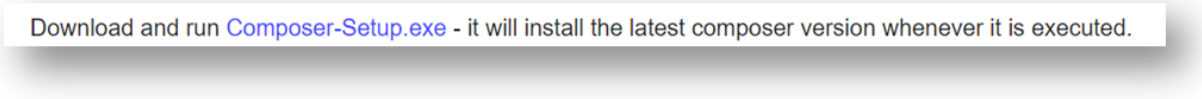
8. El instalador presenta un resumen de lo instalado. Por último hay que configurar el servidor.

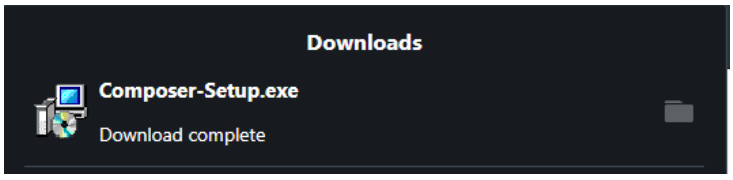


Instalación Laravel

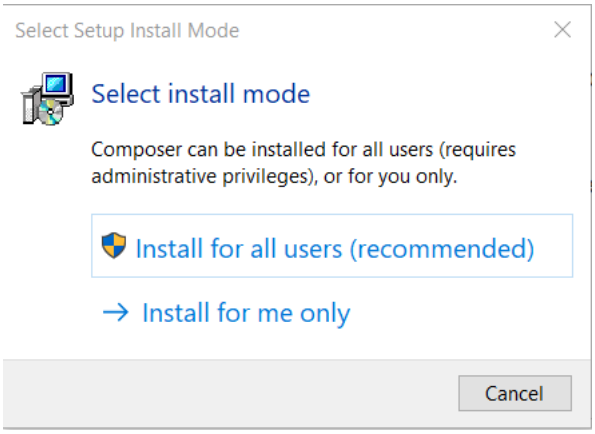
El primer paso es instalar composer.

1. Ingresar a la url https://getcomposer.org/download/
2. Clic en el link de descarga

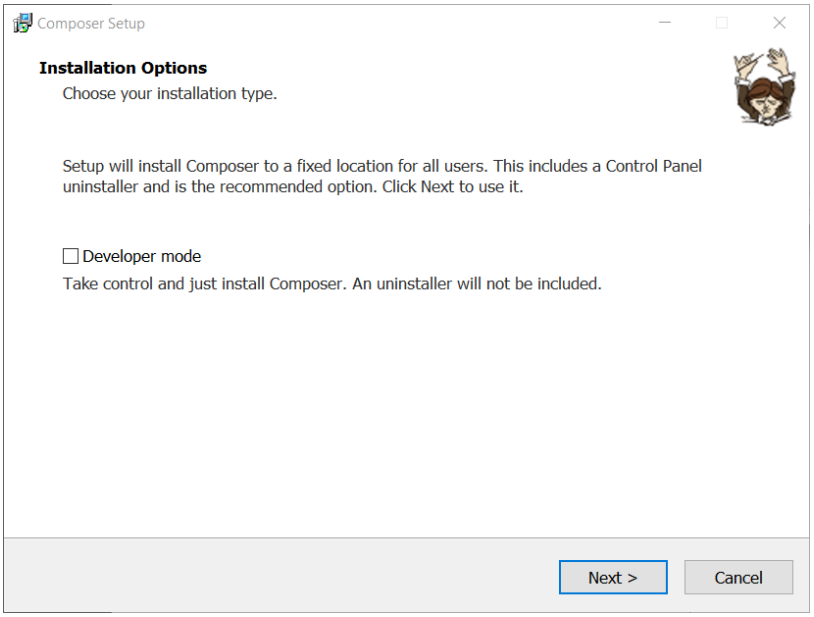




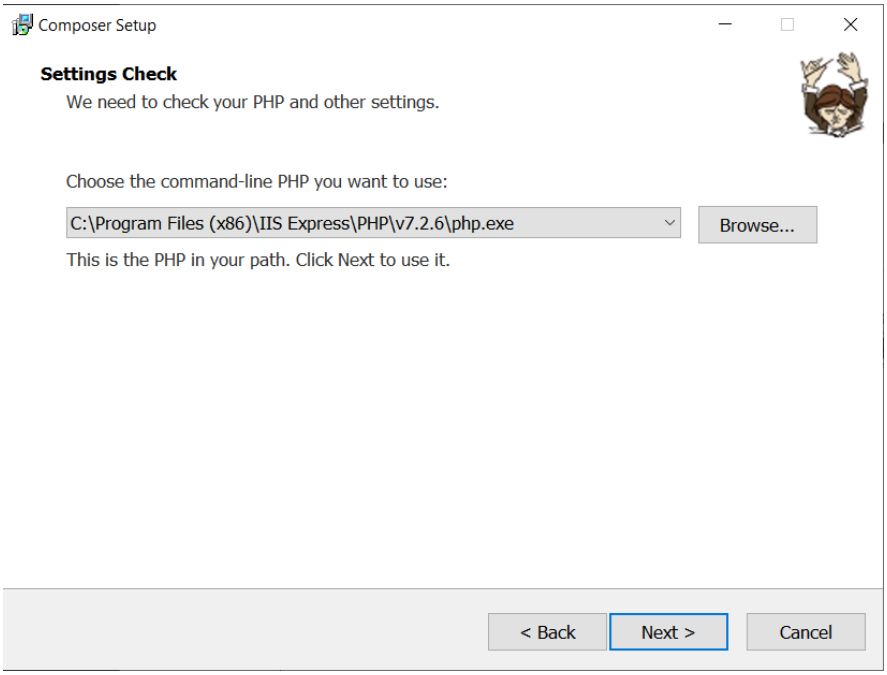
3. Una vez descargado el archivo con extensión “.exe” se debe ejecutar.



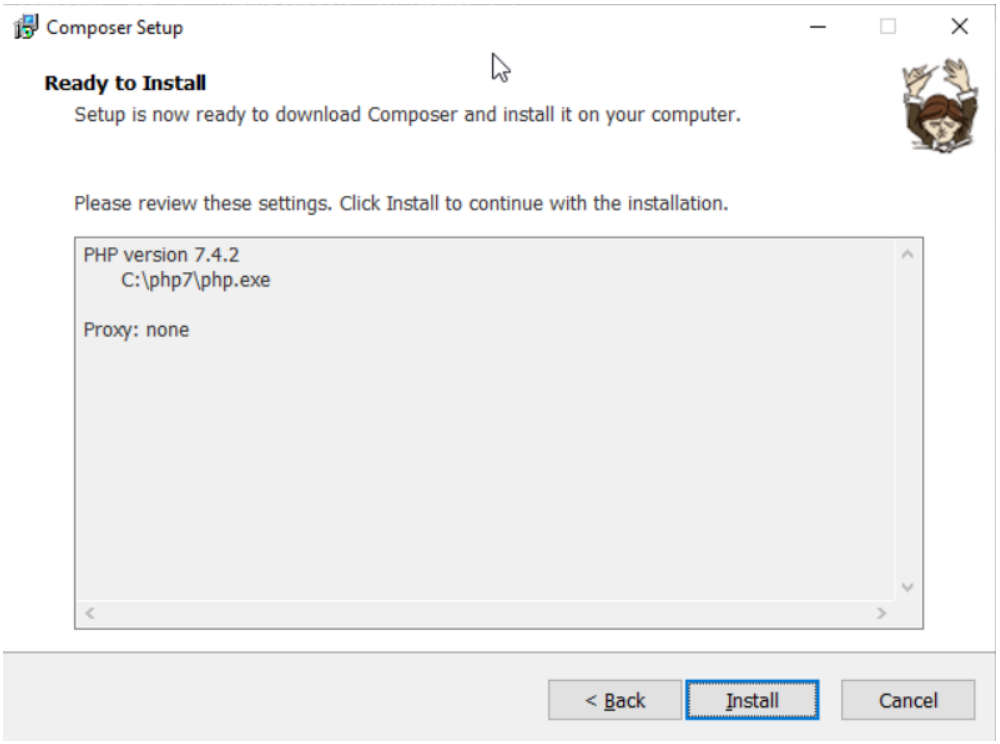
4. Seleccionar la instalación recomendada.



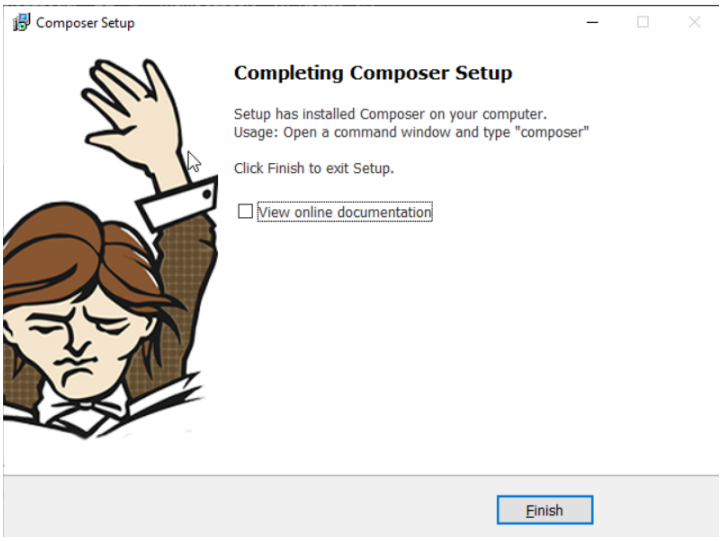
5. Quitar la opción de Developer mode



6. Por default, dejar la carpeta de almacenamiento que propone el instalador.
7. Se confirma la versión de PHP.



8. Se finaliza el proceso de instalación.



Una vez instalado composer, podemos crear nuestro proyecto en laravel. Esto lo vamos a realizar desde la línea de comandos.

1. Ubicarnos en la carpeta que queremos que esté nuestro proyecto.

```
C:\Users\mishe>cd Desktop/Tesis

C:\Users\mishe\Desktop\Tesis>
```

2. Crear el proyecto.

```
C:\Users\mishe\Desktop\Tesis>composer create-project laravel/laravel proyectoTesis
PHP Warning:  Module 'mysqli' already loaded in Unknown on line 0
PHP Warning:  Module 'mbstring' already loaded in Unknown on line 0
PHP Warning:  Module 'openssl' already loaded in Unknown on line 0
PHP Warning:  Module 'pdo_mysql' already loaded in Unknown on line 0
Installing laravel/laravel (v7.6.0)
  - Installing laravel/laravel (v7.6.0): Downloading (100%)
Created project in proyectoTesis
> @php -r "file_exists('.env') || copy('.env.example', '.env');"
PHP Warning:  Module 'mysqli' already loaded in Unknown on line 0
PHP Warning:  Module 'mbstring' already loaded in Unknown on line 0
PHP Warning:  Module 'openssl' already loaded in Unknown on line 0
PHP Warning:  Module 'pdo_mysql' already loaded in Unknown on line 0
Loading composer repositories with package information
Updating dependencies (including require-dev)
Package operations: 92 installs, 0 updates, 0 removals
  - Installing voku/portable-ascii (1.4.10): Downloading (100%)
  - Installing symfony/polyfill-ctype (v1.17.0): Downloading (100%)
  - Installing phpoption/phpoption (1.7.3): Downloading (100%)
  - Installing vlucas/phpdotenv (v4.1.5): Downloading (100%)
  - Installing symfony/css-selector (v5.0.8): Downloading (100%)
  - Installing tijsverkoyen/css-to-inline-styles (2.2.2): Loading from cache
  - Installing symfony/polyfill-mbstring (v1.17.0): Downloading (100%)
  - Installing symfony/var-dumper (v5.0.8): Downloading (100%)
  - Installing symfony/routing (v5.0.8): Downloading (100%)
  - Installing symfony/process (v5.0.8): Downloading (100%)
```

3. Para crear un controller, usar el siguiente comando (tomar en cuenta que debe estar dentro del proyecto para crear el controller).

```
C:\Users\mishe\Desktop\Tesis\proyectoTesis>php artisan make:controller controllerTest
PHP Warning:  Module 'mysqli' already loaded in Unknown on line 0
PHP Warning:  Module 'mbstring' already loaded in Unknown on line 0
PHP Warning:  Module 'openssl' already loaded in Unknown on line 0
PHP Warning:  Module 'pdo_mysql' already loaded in Unknown on line 0
Controller created successfully.
```

4. Para crear un model, usar el siguiente comando (tomar en cuenta que debe estar dentro del proyecto para crear el model).

```
C:\Users\mishe\Desktop\Tesis\proyectoTesis>php artisan make:model modelTest
PHP Warning:  Module 'mysqli' already loaded in Unknown on line 0
PHP Warning:  Module 'mbstring' already loaded in Unknown on line 0
PHP Warning:  Module 'openssl' already loaded in Unknown on line 0
PHP Warning:  Module 'pdo_mysql' already loaded in Unknown on line 0
Model created successfully.
```

5. Para correr la aplicación creada es necesario usar el siguiente comando. La aplicación se levanta en el localhost, en el puerto 8000

```
C:\Users\mishe\Desktop\Tesis\proyectoTesis>php artisan serve
PHP Warning:  Module 'mysqli' already loaded in Unknown on line 0
PHP Warning:  Module 'mbstring' already loaded in Unknown on line 0
PHP Warning:  Module 'openssl' already loaded in Unknown on line 0
PHP Warning:  Module 'pdo_mysql' already loaded in Unknown on line 0
Laravel development server started: http://127.0.0.1:8000
[Tue May 12 23:21:04 2020] PHP Warning:  Module 'mysqli' already loaded in Unknown on line 0
[Tue May 12 23:21:04 2020] PHP Warning:  Module 'mbstring' already loaded in Unknown on line 0
[Tue May 12 23:21:04 2020] PHP Warning:  Module 'openssl' already loaded in Unknown on line 0
[Tue May 12 23:21:04 2020] PHP Warning:  Module 'pdo_mysql' already loaded in Unknown on line 0
```

6. Si colocamos esa url en el navegador, podemos ver que nuestra aplicación ya está corriendo.

Instalación en la nube (Formulario de registro)

Se recomienda usar infranetworking para alojar la app en un hosting en la nube.

Elija Ciclo de Facturación

- (•) 1 Mes - US$19.90
- ( ) 3 Meses - US$19.30
- ( ) 6 Meses - US$18.70
- ( ) 12 Meses - US$16.58
- ( ) 24 Meses - US$15.75
- ( ) Precio de 36 meses - US$15.47

Opciones Configurables

Servicio de Backup
- (•) Yo mismo haré Respaldos de mis Webs (Gratis)
- ( ) Respaldo Premium Semanal y Mensual de tus Webs US$4.00

Sumario de Pedido

Hosting Multidominio - **Multidominios Beta**

Multidominios Beta	US$19.90
» Servicio de Backup: Yo mismo haré Respaldos de mis Webs (Gratis)	US$0.00
Costo de Instalación:	US$0.00
Mensual:	US$19.90
Importe a la Fecha:	**US$19.90**

Continuar

Ver Carrito

Configuración de Dominios

Las siguientes opciones y ajustes están disponibles para los dominios que usted ha elegido. Los campos obligatorios están indicados con un *.

residenciatupa.com - 1 Año(s) [Este dominio ya tiene Alojamiento]

Hosting:	[Este dominio ya tiene Alojamiento]
Período de Registro:	1 Año(s)
Complementos:	☐ Gestionar DNS (¡GRATIS!)
	☐ Protección de ID (US$5.00)

Continuar

Sus Datos

Nuevo Cliente | Cliente Existente

Nombre
Mishelle

Apellido
Zambonino

Nombre de Compañía
Residencia Universitaria Tulpa

Dirección de E-Mail
mishe_zambonino@hotmail.com

Contraseña
••••••••••

Consejos Útiles:
* Utilice ambos caracteres, mayúsculas y minúsculas
* Incluya al menos un símbolo (# $! % & etc...)
* No utilice palabras del diccionario

Dirección 1
Av Gonzales Suarez y Gonnessiat

Dirección 2

Ciudad
Quito

Provincia/Región
Pichincha

Código Postal
170102

País
Ecuador

Número de Teléfono
+593 98 413 9974

Número de Móvil

Confirmar Contraseña
••••••••••

Titular de la Cuenta
Persona

Número de Identidad del Titular
0503236747
Número de Identidad Personal (cédula / DNI) o número de tu Empresa.