UNIVERSIDAD AUTÓNOMA DEL
ESTADO DE QUINTANA ROO

DIVISIÓN DE CIENCIAS, INGENIERÍA Y TECNOLOGÍA

IMPLEMENTACIÓN DE HERRAMIENTAS PARA EL FRAMEWORK "BITLY" PARA EL DESARROLLO ÁGIL DE SISTEMAS INFORMÁTICOS DE LA UNIVERSIDAD DE QUINTANA ROO

TESIS

PARA OBTENER EL GRADO DE

INGENIERO EN REDES

PRESENTA

KEVIN ALFARO CARRILLO

DIRECTOR

LIC. SERGIO ALBERTO SOLÍS SOSA

ASESORES

DR. JAIME SILVERIO ORTEGÓN AGUILAR

M.T.I. MELISSA BLANQUETO ESTRADA

M.S.I. RUBÉN ENRIQUE GONZÁLEZ ELIXAVIDE

M.T.I. VLADIMIR VENIAMIN CABAÑAS VICTORIA

CHETUMAL QUINTANA ROO, MÉXICO, JUNIO DE 2023

UNIVERSIDAD AUTÓNOMA DEL
ESTADO DE QUINTANA ROO

DIVISIÓN DE CIENCIAS, INGENIERÍA Y TECNOLOGÍA

TRABAJO DE TESIS TITULADO

"IMPLEMENTACIÓN DE HERRAMIENTAS PARA EL FRAMEWORK "BITLY" PARA EL DESARROLLO ÁGIL DE SISTEMAS INFORMÁTICOS DE LA UNIVERSIDAD DE QUINTANA ROO"

ELABORADO POR

KEVIN ALFARO CARRILLO

BAJO SUPERVISIÓN DEL COMITÉ DEL PROGRAMA DE LICENCIATURA Y APROBADO COMO REQUISITO PARCIAL PARA OBTENER EL GRADO DE:

INGENIERO EN REDES

COMITÉ DE TESIS

DIRECTOR: _____

LIC. SERGIO ALBERTO SOLÍS SOSA

ASESOR: _____

DR. JAIME SILVERIO ORTEGÓN AGUILAR

ASESORA: _____

M.T.I. MELISSA BLANQUETO ESTRADA

ASESOR SUPLENTE: _____

MS.I. RUBÉN ENRIQUE GONZÁLEZ ELIXA

ASESOR SUPLENTE: _____

M.T.I. VLADIMIR VENIAMIN CABAÑAS VICTORIA

UNIVERSIDAD AUTÓNOMA DEL ESTADO DE QUINTANA ROO
ÁREA DE TITULACIÓN

UNIVERSIDAD AUTÓNOMA DEL ESTADO DE QUINTANA ROO
DIVISIÓN DE CIENCIAS, INGENIERÍA Y TECNOLOGÍA

CHETUMAL QUINTANA ROO, MÉXICO, JUNIO DE 2023

UNIVERSIDAD AUTÓNOMA DEL ESTADO DE QUINTANA ROO

Declaración de Originalidad

Chetumal, Quintana Roo a 27 de junio del 2023

En mi calidad de autor de la Tesis titulada **Implementación de herramientas para el Framework "Bitly" para el desarrollo ágil de Sistemas Informáticos de la Universidad de Quintana Roo**, que presento para obtener el título de **Ingeniero en Redes**, declaro bajo protesta de decir verdad que:

Este trabajo de tesis es original e inédito, de mi propia autoría intelectual.

Si bien contiene parcialidades del contenido de obras, las mismas son citadas y respaldadas en el reconocimiento del derecho moral de los autores; por lo que no es una traducción ni una versión mejorada de otro documento publicado o aún sin publicar.

No ha sido utilizada anteriormente para obtener algún grado académico, ni ha sido publicado por cualquier medio.

En todas las citas y las paráfrasis que utilizo, identifico las fuentes originales e incluyo las referencias completas en el apartado correspondiente.
Identifico la procedencia de las tablas y figuras (gráficas, mapas, diagramas, esquemas ilustraciones, arte digital, fotografías u otros) previamente publicadas, reconociendo el derecho moral de los autores.

Todos los contenidos de esta tesis están libres de cualquier violación en materia de derechos de autor, por lo que asumo la responsabilidad de cualquier litigio o reclamación relacionada con derechos de propiedad intelectual, eximiendo de toda responsabilidad a la Universidad Autónoma del Estado de Quintana Roo.

Reconozco que la Universidad Autónoma del Estado de Quintana Roo no comparte necesariamente las afirmaciones que en esta Tesis se plantean.

Firma Autor

Kevin Alfaro Carrillo

15-17991

UNIVERSIDAD AUTÓNOMA DEL ESTADO DE QUINTANA ROO

Carta de Cesión de Derechos

En mi calidad de autor(a) de la Tesis titulada "Implementación de herramientas para Framework "Bitly" para el desarrollo ágil de Sistemas Informáticos de la Universidad de Quintana Roo", presentada para obtener el título de licenciado en Ingeniería en, es de mi plena voluntad:

- Autorizar a la Universidad Autónoma del Estado de Quintana Roo para que se encargue de la divulgación científica total o parcial de esta obra, en formato impreso o digital, sin limitación en el tiempo, por los medios que dicha institución decida, y con fines educativos o académicos exclusivamente.
- Aceptar que los lectores puedan descargar, almacenar, copiar y distribuir gratuitamente la versión final aprobada de la Tesis, siempre y cuando se realice sin fines comerciales, no se generen obras derivadas y se mencione la autoría de la obra.
- Reutilizar la versión final aprobada de la Tesis con propósitos educativos o académicos y a publicar la obra, en formato impreso o digital, siempre y cuando no se generen nuevos derechos que impidan a la Universidad Autónoma del Estado de Quintana Roo continuar con la divulgación científica de la obra.
- Aceptar que, si la Tesis es publicada con fines comerciales, esta no debe denotar, contener, insertar o incluir en ninguna parte interna o externa de la publicación el escudo, emblema, logotipo o nombre de la Universidad Autónoma del Estado de Quintana Roo. En caso contrario, debo obtener previamente la autorización por escrito del representante legal de la Universidad.
- Ingresar al repositorio que me indique el Área de Biblioteca del Campus al que pertenezco el archivo final de mi trabajo de titulación en el formato que se me solicite.
- Autorizo en este acto a la Universidad Autónoma del Estado de Quintana Roo para que difunda mi información personal, tales como mi nombre y/o seudónimo, correo electrónico en la plataforma utilizada por la Institución, por lo que, la eximo de cualquier responsabilidad y/o futura reclamación presente por la protección de datos personales señalados en la Ley General de Protección de Datos Personales en Posesión de Sujetos Obligados.

Chetumal, Quintana Roo, a 27 de junio de 2023

Firma

Kevin Alfaro Carrillo
15-17991

DIRECCIÓN GENERAL DE INNOVACIÓN EDUCATIVA
BIBLIOTECAS

Sello de la Biblioteca

CERTIFICADO DE ANÁLISIS
magister

Tesis_Kevin_Alfaro_FINAL

8% Similitudes

2% Texto entre comillas
2% similitudes entre comillas
< 1% Idioma no reconocido

Nombre del documento: Tesis_Kevin_Alfaro_FINAL.pdf
ID del documento: d66a14c70275014be12f786b6eba919274c00264
Tamaño del documento original: 2,71 MB

Depositante: Niuris Guerrero González
Fecha de depósito: 27/6/2023
Tipo de carga: interface
fecha de fin de análisis: 27/6/2023

Número de palabras: 18.303
Número de caracteres: 129.974

Ubicación de las similitudes en el documento:

Fuentes

Fuentes principales detectadas

Nº	Descripciones	Similitudes	Ubicaciones	Datos adicionales
1	**www.sergiosolis.com** | Generator - Sergio Solis http://www.sergiosolis.com/documentacion/framework/generator/	2%		Palabras idénticas : 2% (524 palabras)
2	**www.sergiosolis.com** | MVC: Modelo - Vista - Controlador - Sergio Solis http://www.sergiosolis.com/documentacion/framework/mvc/ **2 fuentes similares**	2%		Palabras idénticas : 2% (415 palabras)
3	**1library.co** | Requerimientos funcionales y no funcionales https://1library.co/document/qv969jdy-requerimientos-funcionales-y-no-funcionales.html#:~:text=Los r... **1 fuente similar**	< 1%		Palabras idénticas : < 1% (176 palabras)
4	**mundokramer.wordpress.com** | Especificaciones de requerimiento, características ... https://mundokramer.wordpress.com/2011/05/19/requerimientos-caracteristicas-y-limitaciones-del-s...	< 1%		Palabras idénticas : < 1% (66 palabras)
5	**www.sergiosolis.com** | Condicionamiento IDE - Sergio Solis http://www.sergiosolis.com/documentacion/framework/condicionamiento-ide/#:~:text=Variables de en...	< 1%		Palabras idénticas : < 1% (60 palabras)

Fuentes con similitudes fortuitas

Nº	Descripciones	Similitudes	Ubicaciones	Datos adicionales
1	**www.studocu.com** | Identificación de Requerimientos funcionales y no funcionales -... https://www.studocu.com/es-mx/document/instituto-tecnologico-superior-de-poza-rica/fundamentos-d...	< 1%		Palabras idénticas : < 1% (34 palabras)
2	**www.eumed.net** | Requerimientos Funcionales - Libro Gratis https://www.eumed.net/libros-gratis/2012b/1232/requerimientos-funcionales.html#:~:text=Los requeri...	< 1%		Palabras idénticas : < 1% (29 palabras)
3	**hdl.handle.net** | Scrum como estrategia para el aprendizaje colaborativo a través de ... http://hdl.handle.net/10481/53159	< 1%		Palabras idénticas : < 1% (28 palabras)
4	**hdl.handle.net** | Desarrollo e implementación de un sistema de gestión vacacional p... http://hdl.handle.net/20.500.12894/5899	< 1%		Palabras idénticas : < 1% (27 palabras)
5	**Documento de otro usuario** #5eb354 El documento proviene de otro grupo	< 1%		Palabras idénticas : < 1% (20 palabras)

Fuentes mencionadas (sin similitudes detectadas)

Estas fuentes han sido citadas en el documento sin encontrar similitudes.

1 https://www.oracle.com/database/sqldeveloper/
2 https://www.oracle.com/database/technologies/instant-client/winx64-64
3 https://www.oracle.com/database/technologies/instant-client/winx64-64-downloads.html
4 https://www.acens.com/wp
5 https://rua.ua.es/dspace/bitstream/10045/4042/1/tema11.pdf

Resumen

Derivado de las crecientes solicitudes de desarrollo de software del Departamento de Recursos Humanos de la Universidad Autónoma del Estado de Quintana Roo, se decide desarrollar un framework que se adapte a las necesidades institucionales. No obstante, dichas solicitudes se volvieron demandantes lo que hace necesario la mejora de esta herramienta de desarrollo de software, ya que, al ser de reciente creación, requiere que sus procesos de ejecución sean optimizados para garantizar eficiencia y eficacia de los sistemas que se generen, así como brindar la seguridad de la información que se gestiona. El principal objetivo de este trabajo es el desarrollo de componentes y nuevas implementaciones para el framework, que permitan el desarrollo rápido de proyectos de software. Se utilizará la metodología de Scrum con el fin de mantener un control, una buena administración y avance eficiente en todas las fases del proyecto. Esta herramienta es desarrollada con el lenguaje de programación PHP, con la posibilidad de emplear la base de datos Oracle y MySQL, incorporando diversas tecnologías (HTML, CSS, JavaScript) y librerías como jQuery, PhpSpreadsheet (para el manejo de documentos de texto y hojas de cálculo), FPDF, AdminLTE, BootStrap, entre otros. La mejora del framework podrá garantizar el desarrollo óptimo de los sistemas que sean necesarios para la universidad.

Agradecimientos

Le agradezco a mis padres, por la confianza y por alentarme a seguir adelante. Algunos momentos fueron difíciles, pero ustedes siempre me apoyaron en todo momento. Sin duda han sido un gran soporte y ejemplo para mí.

Agradezco a esa persona que siempre ha estado a mi lado, brindándome su amor y su apoyo para lograr terminar este proyecto. Sin su apoyo, parte de esto no podría ser posible y le doy parte del crédito por ello.

Gracias a mi buen amigo y director de tesis Sergio Solís, por toda la ayuda y por guiarme en todo este proceso. Un gran maestro y sin duda un gran amigo en quien apoyarme.

Agradezco infinitamente al Dr. Jaime Ortegón, quién siempre estuvo dispuesto a ayudarme en todo momento, escuchándome, aconsejándome y alentándome para la terminación de este proyecto.

Y a mis profesores, por guiarme por todo mi camino de estudiante universitario, brindándome sus experiencias y conocimientos, así como aconsejándome y asesorándome. Son un ejemplo para seguir. Muchas gracias.

A las amistades que logré realizar durante toda mi estancia en la universidad, las buenas personas que pude conocer, compartiendo todo tipo de momentos juntos, realizando tareas, proyectos, prácticas, riendo juntos en todo momento y apoyándonos mutuamente. Gracias por todo chicos.

Dedicatoria

Este proyecto se lo dedico a:

Mi familia, quienes me brindaron todo su apoyo para que yo pueda salir adelante, haciendo todo lo posible para que pueda estudiar la Universidad. Muchas gracias familia.

A esa persona que siempre ha estado a mi lado, brindándome su apoyo y amor todos estos años, echándole ganas para poder salir adelante juntos.

A mi director de tesis Sergio Solís, quien es un excelente maestro y amigo, por brindarme las herramientas necesarias y guiarme para realizar este proyecto, así como resolviendo mis dudas que surgieron en la realización de este. Mil gracias.

A mi asesor de tesis, el Dr. Jaime, quien siempre estuvo al tanto de mí para poder terminar este proyecto, ayudándome con mi documento y resolviendo mis dudas. Muchas gracias.

Y finalmente, dedico este proyecto a todas las personas que me apoyaron en mi camino como estudiante y en la terminación de este proyecto, con el cual, sin su apoyo, esto no hubiera sido posible.

Contenido

Índice de ilustraciones

Índice de tablas

Capítulo 1 Introducción

1.1 Antecedentes

La Universidad Autónoma del Estado de Quintana Roo es una institución relativamente joven, fundada en el año de 1991, y se ha caracterizado por estar siempre a la vanguardia tecnológica sirviendo de referencia a otras instituciones. Es el Área de Sistemas de la Dirección General de Tecnologías de la Información y Comunicación (DGTIC) de la misma universidad, la encargada de liderar estas implementaciones tecnológicas, pero debido al crecimiento y aumento de solicitudes de creación de software, el Departamento de Recursos Humanos (DRH), decidió crear su propia área de desarrollo de sistemas informáticos.

Con la necesidad de crear sistemas con prontitud, se decide por el desarrollo de un *framework* (herramienta de desarrollo) adaptado también a las necesidades de la misma institución y al DRH. Esta herramienta es desarrollada con el lenguaje de programación PHP, con la posibilidad de emplear la base de datos Oracle y MySQL, incorporando diversas tecnologías (HTML, CSS, JavaScript) y librerías como jQuery, PhpSpreadsheet (Para el manejo de Documentos de texto y hojas de cálculo), FPDF, AdminLTE, Bootstrap, etc.

El mercado actual de frameworks presenta algunas limitantes puesto que la mayoría se vuelven obsoletos con el paso del tiempo, ya sea porque el proceso de aprendizaje resulta tardado, o la falta de incorporación de nuevas tecnologías para el desarrollo de software. De igual manera, algunos de los frameworks son inflexibles ya no que no permiten realizar nuevos desarrollos de sistemas y, además, las tecnologías incorporadas no se adaptan con facilidad a otros entornos.

No obstante, debido a las necesidades del Departamento de Recursos Humanos de la Universidad Autónoma del Estado de Quintana Roo, se requiere mejorar el Framework existente, ofreciendo sistemas más detallados y amigables para el usuario, así como la incorporación de herramientas propias, que ayuden a optimizar el desarrollo de sistemas.

1.2 Justificación

El Departamento de Recursos Humanos de la Universidad Autónoma del Estado de Quintana Roo emplea un framework en desarrollo de software denominado "Bitly", el cual surgió por la necesidad de sustituir cualquier otro framework existente en el mercado orientado al desarrollo de sistemas, con la finalidad de cumplir las necesidades propias del departamento.

"Bitly" es un framework artesanal que utiliza el patrón de diseño de software MVC (Modelo-Vista-Controlador) en el cual se tiene control total de todas las partes de la aplicación, permitiendo así modificar las funcionalidades, controlando la lógica de seguridad y otros aspectos, sin estar implementada por terceros.

Las solicitudes de desarrollo de sistemas informáticos en el DRH son demandantes, lo que justifica la mejora del framework "Bitly" para la programación rápida de sistemas.

Bitly, es un framework de reciente creación, por el cual requiere que sus procesos de ejecución sean optimizados para garantizar eficiencia y eficacia de los sistemas que se generen, así como brindar la seguridad de la información que se gestiona. Asimismo, es importante la creación de nuevas herramientas y la incorporación de nuevas funcionalidades que permitan optimizar el tiempo del desarrollo de sistemas.

1.3 Objetivo general

Desarrollar componentes y nuevas implementaciones para el framework "Bitly" que permitan el desarrollo rápido de proyectos de software para el Departamento de Recursos Humanos de la Universidad Autónoma del Estado de Quintana Roo.

1.4 Objetivos particulares

- Completar componentes para la herramienta *Generator*.
- Concluir módulos prioritarios: instalador y roles de usuario.

- Detectar y corregir posibles errores y vulnerabilidades de seguridad.
- Desarrollar el módulo "Generación de código inicial para nuevos Módulos".
- Completar documentación de operación del framework para su mantenimiento.

1.5 Alcances

El presente proyecto pretende desarrollar nuevos componentes al Módulo de "Generator" del framework Bitly, para poder agilizar los desarrollos de sistemas que hagan uso de esta herramienta.

Además, será necesario realizar una revisión exhaustiva al núcleo de este framework con la finalidad de encontrar vulnerabilidades, actualizar las librerías requeridas, y como consecuencia garantizar la estabilidad y seguridad de los futuros sistemas, documentando su estructura para posibles adecuaciones posteriores.

Por otra parte, también será necesario la creación de un módulo de instalación del framework, con su respectivo instructivo, para facilitar a los nuevos programadores hacer uso de esta herramienta.

Capítulo 2 Marco Teórico

2.1 Herramientas para el desarrollo de sistemas (Frameworks)

De acuerdo con Lafosse (2010) un framework es aquel conjunto de bibliotecas, herramientas y normas que se siguen, los cuales ayudan a desarrollar aplicaciones, y se encuentra constituido por varios segmentos/componentes que interactúan entre sí. Estos permiten reutilizar código, estandarizar el desarrollo y utilizar el ciclo de desarrollo de tipo interactivo-incremental (especificación, codificación, mantenimiento y evolución).

Por otra parte, Acens Technologies (2014) señala que un framework no es un software ni una herramienta que se pueda ejecutar y brinde una interfaz gráfica en la cual trabajar, sino más bien lo define como "un conjunto de archivos y directorios que facilitan la creación de aplicaciones, ya que incorporan funcionalidades ya desarrolladas y probadas, implementadas en un determinado lenguaje de programación." (p. 3).

El principal objetivo de los frameworks es agilizar el proceso, al momento de desarrollar una aplicación, permitiendo así enfocarse en el problema y no en implementar funcionalidades de uso común. (Acens Technologies, 2014)

Los Frameworks agregan funcionalidad extendida a un lenguaje de programación, automatizan varios de los patrones de programación con el fin de orientarlos a un determinado propósito, ofreciendo una estructura al código y permitiendo la realización de mejoras, de manera que se convierta en una versión más entendible y sostenible. Asimismo, estos permiten separar la aplicación, generalmente en tres capas (Villalobos, Sánchez, & Gutiérrez, 2010):

- Lógica de presentación: administra las interacciones entre el usuario y el software.
- Lógica de datos: concede el acceso a un agente de almacenamiento persistente u otros.
- Lógica de dominio o de negocio: maneja los modelos de datos de acuerdo con los comandos admitidos desde la presentación.

Por otra parte, los frameworks permiten la construcción sencilla de aplicaciones, y por consiguiente, la reutilización de código, gracias a las librerías de componentes con las que cuenta. Dichos componentes pueden emplearse junto con otros, puesto que comparten la misma interfaz del framework (Delía, 2019).

No obstante, no todos los aspectos de una aplicación pueden ser creados de manera flexible, de manera que algunas características del framework no podrán ser modificables, ya que son compartidas por todas las aplicaciones que han sido generadas a partir del Framework (Markiewicz & Lucena, 2000).

Existen diversos tipos de frameworks en todos los dominios de aplicación, y casi cualquier idioma, entre los cuales podemos encontrar (Lafosse, 2010):

- De infraestructura de sistemas, los cuales permiten desarrollar los sistemas de explotación, herramientas gráficas y plataformas de red.
- Comunicativos (software).
- De empresa (desarrollos específicos).
- De gestión de contenido (tipo Content Management System).

2.1.1 Framework artesanal

Los frameworks actuales contiene muchas funcionalidades precargadas, los cuales buscan ahorrar trabajo y tiempo al usuario de descargar y añadir a un proyecto. No obstante, esto no siempre resulta útil ya que vuelven a la aplicación desarrollada muy pesada.

Por lo tanto, para resolver este problema surgieron los frameworks artesanales, puesto que permiten modificar las funcionalidades volviéndolo más ligero.

Además, los frameworks artesanales permiten un desarrollo a medida, lo que ayuda a solucionar problemas de escalabilidad de una aplicación y el acceso y personalización del código, puesto que se tiene el control total de todas las partes de la aplicación. De igual manera estos no están implementados por terceros y permite tener un control de la lógica de seguridad de la aplicación teniendo así una seguridad más robusta.

En general, se puede observar que tanto los frameworks artesanales como los tradicionales ofrecen soluciones accesibles para pequeñas y grandes empresas.

Una vez que se ha comprendido lo que es un framework artesanal, solo queda reafirmar que los frameworks, tanto artesanales como los no-artesanales ofrecen soluciones accesibles para pequeñas y grandes empresas.

2.2 Modelo-Vista-Controlador

Muñoz (2013) define el Modelo Vista Controlador (MVC) como aquel patrón de arquitectura de software en el cual se separan los datos y la lógica de negocio de la interfaz de usuario y el módulo que se encarga de administrar los eventos y las comunicaciones. Asimismo, está basado en la reutilización de código y la separación de conceptos, de manera que se facilite el desarrollo de aplicaciones y posteriormente, su mantenimiento.

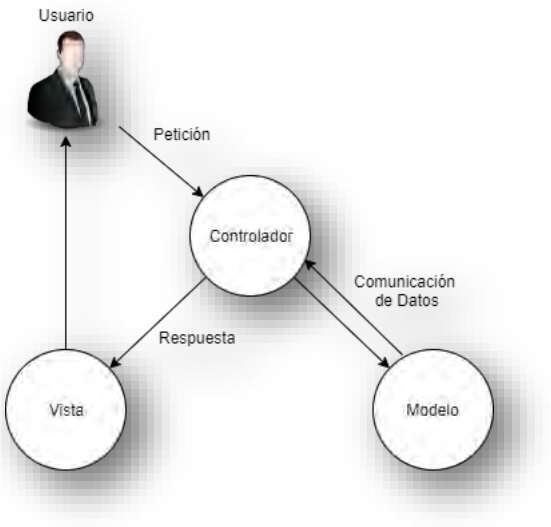

Ilustración 1: Modelo-Vista-Controlador. Fuente: Elaboración propia.

Como su nombre lo indica, propone la construcción de tres componentes: el modelo, la vista y el controlador; es decir que, se definen los componentes tanto para la representación de la información como para la interacción del usuario, de manera independiente, lo que permite que cualquier cambio realizado en el Modelo puede verse reflejado de manera automática en cada una de las vistas. (Muñoz, 2013)

De acuerdo con Mariscal (2015) el uso de este patrón presenta diversas ventajas, las cuales son:

- La implementación se realiza a través de módulos.
- Las vistas siempre muestran información actualizada, ya que esta se realiza de manera automática por medio del modelo de la aplicación.
- Cualquier modificación realizada no afectará el mecanismo de comunicación y actualización entre los modelos.
- No se modifica el tratamiento de la información sino solo su representación.

No obstante, también se han observado problemas, entre los cuales se encuentran (Mariscal, 2015):

- En un primer momento, para el desarrollo de una aplicación será necesario emplear mucho tiempo, ya que normalmente es necesario desarrollar un mayor número de clases a diferencia de otros entornos de desarrollo. Sin embargo, esto permite que después sea más fácil mantener la aplicación en comparación a otras aplicaciones que no hacen uso de este modelo.
- El MVC requiere una estructura inicial sobre la que se construirán las clases e interfaces que permitirán desarrollar, modificar y comunicar los módulos de la aplicación. Esta estructura deberá contener las acciones o eventos para poder proporcionar las notificaciones que el modelo genere. Asimismo, se requerirá de una clase modelo, una clase vista y otra clase controlador que se ocupen de la comunicación y la actualización, y para el desarrollo de la aplicación serán totalmente transparentes

2.2.1 Modelo

Romero & González (2012) definen el Modelo como la representación de los datos del programa, la cual maneja la información y mantiene un controladas todas sus transformaciones. El Modelo es independiente de los controladores y de las vistas, ya que no tiene conocimientos ni referencia de ellos, y es el propio sistema el que mantiene los enlaces entre el Modelo y sus Vistas, y notifica a las Vistas cuando hay algún cambio en el Modelo.

2.2.2 Vista

La Vista es el objeto que se encarga de generar una presentación visual de los datos que representa el Modelo, y muestra los datos al usuario. Asimismo, aunque interactúa de manera preferencia con el Controlador, es posible que se comunique de manera directa con el Modelo mediante una referencia al propio Modelo. (Romero & González, 2012)

2.2.3 Controlador

De acuerdo con Romero & González (2012) el Controlador es el objeto que actúa como un intermediario entre la Vista y el Modelo, y entra en acción al momento de realizar algún cambio, ya sea en la información del Modelo o por modificaciones de la Vista.

2.3 Programación Orientada a Objetos

La programación orienta a objetos es un paradigma de programación totalmente diferente al método clásico de programación, el cual utiliza objetos y su comportamiento para resolver problemas y generar programas y aplicaciones informáticas. Dentro este paradigma, el objeto es considerado una entidad con un estado (datos o variables de instancia) y unas funciones (métodos) que permiten tener acceso y realizar modificaciones en dicho estado.

Con la programación orientada a objetos (POO) se aumenta el modularidad de los programas y la reutilización de los mismo. Además, la POO se diferencia de la programación clásica porque utiliza técnicas nuevas como el polimorfismo, el encapsulamiento, la herencia, etc.

Generalmente, los lenguajes de última generación permiten la programación orientada a objetos, así como la programación clásica. Por esta razón, puede entenderse la POO como una evolución de la programación clásica (programación estructurada). (Pérez, 2015)

La POO permite el modelaje de un dominio, por ejemplo, un problema a programar, de manera que este sea lo más cercano a la realidad. A su vez, los objetos del programa simulan los objetos (sustantivos) del dominio, ya sea una automóvil o una carretera, por ejemplo. Por otro lado, los métodos de los objetos son los que permiten modelar sin ningún problema, las acciones que pueden realizar, tales como caminar o manejar. (Alicante, 2006)

2.4 Programación por Capas

La programación por capas es una técnica de ingeniería de software propia de la programación por objetos, el cual resulta complejo y por lo tanto debe realizarse de forma eficiente con el fin de obtener una aplicación final consistente y que, a su vez, satisfaga las necesidades del usuario. (Ricardo J. Vargas Del Valle, 2006)

De igual manera, la programación por capas involucra técnicas efectivas de programación en la implementación del modelo de componentes, coherentes con el patrón arquitectónico en capas del modelo de diseño, así como con el diseño detallado de las capas. En este modelo, las capas poseen un significado esencialmente lógico (layer) y su distribución puede resultar en sistemas de un proceso o de múltiples procesos (aplicaciones). (Vázquez, 2010)

Este modelo se organiza principalmente en 3 capas: capa de presentación, capa lógica y capa de datos. Primeramente, se encuentra la capa de presentación, la cual se encuentra compuesta por los objetos que se encargan de la interacción con el usuario, tales como los formularios y las interfaces de la aplicación. Por otro lado, se aprecia la capa lógica en la cual están los objetos que realizan la mayor parte del trabajo interno del programa, en donde se hace énfasis en la lógica de la aplicación, así como en la funcionalidad de servir de enlace entre las otras capas. Finalmente se encuentra la capa de datos, la cual está compuesta por los objetos que envían y obtienen información al comunicarse con los sistemas de información, como las bases de datos, los cuales colaboran con el programa. (Ricardo J. Vargas Del Valle, 2006)

2.5 Tecnologías Web

2.5.1 La World Wide Web

El diseño de interfaces Web es un tema complejo en el que no sólo intervienen procesos de diseño gráfico y programación, sino que también resultan imprescindibles aspectos de la arquitectura de la información, navegación, funcionalidad y, sobre todo, de la usabilidad (Belmonte, 2003).

2.5.2 PHP

Cobo et al. (2005) define PHP como un lenguaje que se interpreta del lado del servido caracterizado por su potencia, versatilidad, robustez y modularidad. Además, es un lenguaje multiplataforma, ya que los programas funcionan sin ningún problema en diversas plataformas, trabaja sobre la mayoría de los servidores web y está listo para interactuar con distintos tipos de base de datos. Sin embargo, sus propiedades pueden ser mejor aprovechadas con el sistema operativo Unix, ya que es un lenguaje concebido inicialmente para este entorno.

Los programas escritos en PHP se incluyen directamente en el código HTML, y son ejecutados por el servidor web mediante un intérprete antes de transferir un resultado, en forma de código de HTML puro, al cliente que lo requiere. (Cobo, Gómez, Pérez, & Rocha, 2005)

2.5.3 HTML

De acuerdo con Cobo et al. (2005) HTML es un lenguaje de definición de hipertexto que se compone de comandos, marcas o etiquetas, también llamadas "Tags" los cuales permiten precisar la estructura lógica de un documento web y determinar sus atributos tales como el color del texto, contenidos multimedia, hipervínculos, entre otros.

Los comandos son insertados en el texto que compone el contenido de la página, siempre incluidos entre los signos < >, los cuales sirven para especificar su estructura y formato. De igual manera, permiten insertar contenidos especiales como las imágenes, videos, sonidos, entre otros. (Cobo, Gómez, Pérez, & Rocha, 2005)

2.5.4 CSS

Marín et al. (2013) definen CSS como un estándar Web que se encarga de asignar el formato a los objetos, fuentes y elementos principales en un diseño Web, y es utilizado como un patrón vinculado al resto de los contenidos. Asimismo, permite el desarrollo de los sitios de manera versátil, ya que al cambiar una hoja de estilos se puede cambiar al mismo tiempo diversas páginas, puesto que generalmente una hoja se encuentra asociada a todas ellas.

2.5.5 JavaScript

JavaScript es un lenguaje de guión del lado del cliente que puede ser utilizado en páginas HTML y es uno de los lenguajes script más empleados y documentados de Internet, el cual se encuentra adoptado como estándar por mayor parte de los fabricantes, debido a que ofrece mayor compatibilidad con la mayoría de los navegadores Web en existencia. (Sanz, y otros, 2015)

Este lenguaje está basado vagamente en el lenguaje de programación Java, sin embargo, a pesar de tener una metodología de programación y sintaxis similares, no puede ser considerada una versión sencilla de Java. (Dimes, 2015)

2.5.6 Librería JQuery

Lancker (204) define jQuery como un Framework JavaScript libre y Open Source, del lado del cliente enfocado en la interacción entre el DOM (Document Object Model), JavaScript, AJAX (Asynchronous JavaScript and XML) y HTML. Tiene como objetivo la simplificación de comandos comunes de JavaScript.

Aunque las especificaciones de jQuery son cuantiosas, tiene como prioridad asegurar la flexibilidad que aporta para dar acceso a todas las secciones del documento HTML mediante la multitud de selectores existentes. Por lo tanto, dicha característica se empleó para nombrar a este framework, ya que "J" proviene de JavaScript y Query hace referencia a la búsqueda o acceso a los elementos. (Lancker, 2014)

2.5.7 Librería PhpSpredsheet

PhpSpreadsheet es una biblioteca escrita en PHP, la cual proporciona un conjunto de clases que permiten la lectura y escritura en diversos formatos de archivo de hoja de cálculo, tales como Excel y LibreOffice Cal. (PhpSpreadsheet, 2019)

2.5.8 FPDF

FPDF es una clase escrita en PHP, el cual permite generar documentos PDF directamente desde PHP, sin la necesidad de usar la biblioteca PDFlib. Además, es libre y gratuito, por lo que se puede usar para cualquier fin y modificarla de acuerdo con las necesidades que se tengan. (FPDF, 2019)

2.5.9 AdminLTE

Lemus et al. (2019) define Admin LTE como una plantilla web open source basada en BootStrap, la cual engloba diversas funcionalidades, estilos y contenido dinámico. Dicha plantilla contiene una extensa y detallada documentación acerca de su uso, que va desde la implementación hasta la modificación de componentes para uso más personalizado; y facilita la interacción con el usuario mediante una interfaz amigable y entendible.

2.5.10 Bootstrap

Bootstrap es un framework originalmente desarrollado por Twitter, el cual permite la creación de interfaces web con CSS y JavaScript, adaptando de manera automática la interfaz del sitio web al tamaño de un dispositivo móvil, como una computadora o una Tablet. Dicha característica es denominada "responsive design". (Lainez, 2016)

2.6 Metodología para el desarrollo de sistemas informáticos

Scrum es un método de trabajo colaborativo que se divide por etapas y con el que un grupo de personas puede obtener mejores resultados que trabajando de forma tradicional (Hundermark, 2011).

De acuerdo con (Hundermark, 2011), Scrum es una metodología de trabajo colaborativo, el cual está compuesto por una serie de fases, y a diferencia de la forma tradicional, esta permite obtener mejores resultados.

De acuerdo con (Navarro Cadavid, Fernández Martínez, & Morales Vélez, 2013) la metodología Scrum es un marco de trabajo colaborativo de proyectos, compuesto por un conjunto de reglas e instrumentos, los cuales generan la estructura necesaria para su correcto funcionamiento.

Asimismo, emplea grupos de trabajo, denominados equipos Scrum, en donde los miembros desempeñan roles específicos, con el fin de entregar bienes y servicios en ciertos periodos de tiempo, denominados *Sprints*, los cuales tienen una duración aproximada de 1 a 4 semanas, y suceden una detrás de otra. Al inicio de cada Sprint, el equipo realiza una selección de los requisitos solicitados por el cliente, la cual no puede ser modificada durante el periodo en cuestión, y se comprometen a cumplirlos al término del Sprint. Posteriormente,

al final del periodo, el equipo presenta los resultados obtenidos y realiza una revisión con el cliente. (Mariño & Alfonzo, 2014)

La metodología Scrum, emplea un enfoque incremental que tiene sus bases en la teoría de control empírico de proceso, la cual se encuentra fundamentada en la transparencia, inspección y adaptación. Todo ello permite que durante el proceso se garantice la visibilidad de aquellas cosas que pudieran afectar el resultado, se detecten variaciones indeseables, y se realicen los ajustes pertinentes para minimizar el impacto de estas. (Navarro Cadavid, Fernández Martínez, & Morales Vélez, 2013)

Capítulo 3 Desarrollo

El presente capítulo plasma lo realizado en este proyecto, desde su fase de planeación, diseño, desarrollo, hasta las pruebas, corrección de errores y documentación.
La metodología que se empleó para la realización de este proyecto es la Metodología Scrum, la cual ayudó a mantener un control, una buena administración y un avance eficiente en todas las fases a desarrollar del proyecto.

Como base metodológica, se implementó la programación orientada a objetos, siendo una parte fundamental de este proyecto, debido a que este permite que el código generado sea reutilizable, organizado y fácil de mantener, es decir que, en caso de que se presente algún fallo, es posible arreglarlo, ya que solo es necesario sustraer la parte dañada sin tener afectaciones en las demás partes del sistema, permitiendo que éste siga funcionando de forma correcta y sin algún problema.

Es importante resaltar que, para la realización de este apartado del proyecto, es necesario tomar en consideración la participación de los involucrados de la Universidad Autónoma del Estado de Quintana Roo, ya que este servirá como una herramienta para los diversos departamentos que lo requieran, permitiendo que el trabajo se realice de forma más eficiente. En el apartado final, se encuentra la fase de pruebas donde se verificó el correcto funcionamiento de Framework.

3.1 Fase de planificación

3.1.1 Descripción de los interesados

Apegado a la metodología elegida para el desarrollo del proyecto, definimos como "Stakeholders" a los interesados en el proyecto, ya sean personas o entidades, los cuales son necesarios para el funcionamiento de este proyecto. Los identificados son:

- **La Universidad Autónoma del Estado de Quintana Roo:** Como institución pública la universidad se ve obligada a tener un sistema que administre sus procesos, debido a esto, es la entidad de mayor interés para que el desarrollo de sistemas informáticos logre otorgar beneficios para esta misma.
- **Departamento de Recursos Humanos:** Existe una necesidad de cálculos más exactos que puedan ser aplicables a la nómina, que garanticen la veracidad en la administración del personal. Por lo tanto, es necesario la implementación y uso de un sistema eficiente que se apegue a la normas y requerimientos, el cual ayude a recortar procesos de trabajo de dicho departamento.
- **Departamento de Sistemas:** Se encarga de la instalación, desarrollo y mantenimiento de los sistemas informáticos en toda la universidad. De igual manera, son responsables de la vigilancia de los sistemas que se han desarrollado para asegurar que estos se apeguen a los estándares de calidad.

3.1.2 Requerimientos funcionales

Los requerimientos funcionales de un sistema describen lo que el sistema debe hacer. Estos requerimientos dependen del tipo de software que se desarrolle, de los posibles usuarios del software y del enfoque general tomado por la organización al redactar requerimientos. Cuando se expresan como requerimientos del usuario, habitualmente se describen de una forma bastante abstracta. Sin embargo, los requerimientos funcionales del sistema describen con detalle la función de éste, sus entradas y salidas, excepciones, etc. (Sommerville, 2005)

De acuerdo con la información antes mencionada, en los requerimientos funcionales se presentan las actividades realizadas a lo largo de este proyecto, con la implementación de la metodología *Scrum*, la cual, ayudó a mantener el control y un avance eficiente de todas las fases del proyecto hasta la versión final.

- Archivo de instalación del framework:
    - Creación de una base de datos a elegir el uso de MySQL u Oracle en la instalación.
    - Creación de usuario para inicio de sesión.

  - Inicio de sesión e implementación del uso de CAPCHAP.

- Implementación de herramientas para el módulo Generator.
  - Pantalla de creación de un Generator.
  - Importador y exportador del módulo Generator.
  - Adaptación de componentes y funciones.

- Desarrollo del módulo "Roles de usuario:
  - Pantalla para el registro de usuarios.
  - Administración de roles de usuario.
  - Administración de permisos de usuario.

En el desarrollo de este proyecto se realizaron otras tareas para mantener el óptimo uso del framework, tanto para el desarrollador como para el usuario, de manera que ambos puedan hacer uso de este sistema.

3.1.3 Requerimientos no funcionales

Los requerimientos no funcionales son aquellos requerimientos que no se refieren directamente a las funciones específicas que proporciona el sistema, sino a las propiedades emergentes de éste; como la fiabilidad, el tiempo de respuesta y la capacidad de almacenamiento. De forma alternativa, definen las restricciones del sistema como la capacidad de los dispositivos de entrada/salida y las representaciones de datos que se utilizan en las interfaces del sistema. (Sommerville, 2005)

Los requerimientos no funcionales rara vez se asocian con características particulares del sistema. Más bien, estos requerimientos especifican o restringen las propiedades emergentes del sistema. Por lo tanto, se puede especificar el rendimiento del sistema, la protección, la disponibilidad y otras propiedades emergentes. A continuación, se mencionan algunos requerimientos no funcionales:

3.1.4. Restricciones

Las restricciones pueden representar parámetros de control de calidad o las limitaciones por la cual no se pueden avanzar más allá de algo establecido. Estos buscan una mejora continua, identificando las restricciones o limitaciones encontradas en un sistema informático.

Las restricciones del framework son fundamentales para el desarrollo ágil de los sistemas, por lo cual es importante mencionarlos y explicar brevemente cada uno de ellos. Algunos de estos fueron los siguientes:

- **Tiempo de respuesta:** El tiempo de ejecución del sistema debe ser moderado (No más de 2 segundos).
- **Tamaño:** El peso del Framework debe ser moderado, con un peso bajo para su fácil y rápida instalación. (No más de 100 MB).
- **Interfaz de usuario:** Presentar una interfaz sencilla, que sea fácil de manejar para los usuarios del sistema.
- **Disponibilidad:** La disponibilidad del sistema debe ser continua con un nivel de servicio para los usuarios de 7 días por 24 horas.
- **Desempeño:** El sistema debe de mantenerse en un servidor eficiente, con el propósito de que los usuarios puedan usar el sistema sin problema.
- **Registro:** Control sobre las acciones y cambios realizados en el sistema.
- **Sistema robusto:** El acceso a los datos debe ser de forma segura.

3.2 Fase de diseño

3.2.1. Diagrama arquitectónico

Para entender el diagrama arquitectónico, es necesario describir el procedimiento de operación del framework, así como su funcionamiento en cada parte de esta. En la siguiente lista se enumera este procedimiento:

1. El usuario realiza una solicitud al Framework por medio de una URL (dominio + URI).
2. El archivo *.htaccess* captura la URL y entrega la URI al archivo *index.php*

3. El *index.php* es el archivo eje, el cual se encarga de realizar:
   a) Carga de archivos que conforman el núcleo del Framework.
   b) Se llama al archivo *autoload.php*, el cual carga todas las librerías colocadas en la carpeta libs. Se instanciarán si se tratan de clases.
   c) Se carga el archivo *config.php*, el cual contiene las credenciales de la base de datos, clave hash para la encriptación de contraseñas.
   d) Se manda a llamar *"getInstancia"* de la clase registro, el cual usa el patrón de diseño *"Singleton"*, con la finalidad de alojar una instancia del controlador y otra instancia de la conexión de la base de datos, evitando así la generación de múltiples instancias y la posible saturación de conexiones a la base de datos.
   e) Instancia la clase *BD* y realiza la conexión a la base de datos.
   f) Llamado al archivo *Boostrap.php*, el cual analiza la URI y manda a llamar al controlador y método correspondiente; en caso de no existir el controlador manda a llamar al Generator.

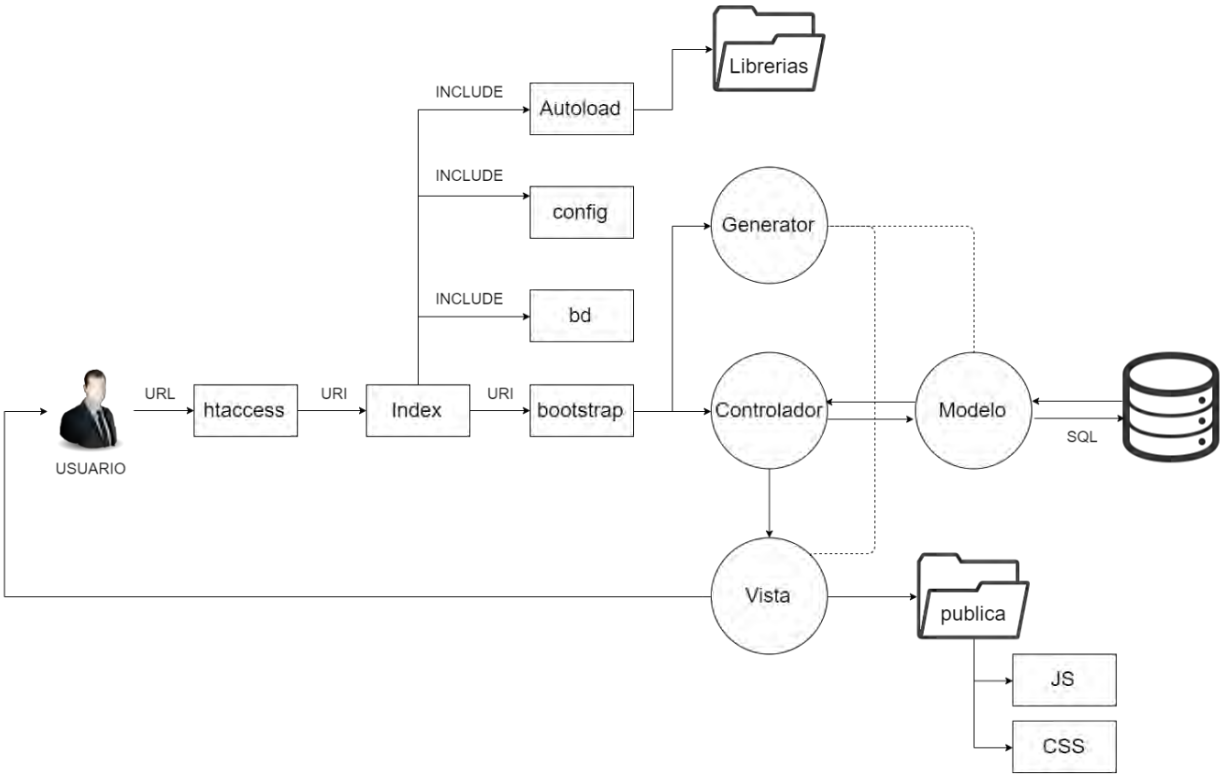

Ilustración 2: Diagrama arquitectónico de operación del Framework. Fuente: Elaboración propia.

3.2.2 Diagrama de organización de archivos

Este diagrama es una representación gráfica de cómo se visualiza la estructura y la organización de los archivos en el Framework. Esta se realizó para poder mostrar la jerarquía y la relación entre los diferentes tipos de archivos del Framework.

En la ilustración 3 se muestra el diagrama de cómo están organizados los archivos.

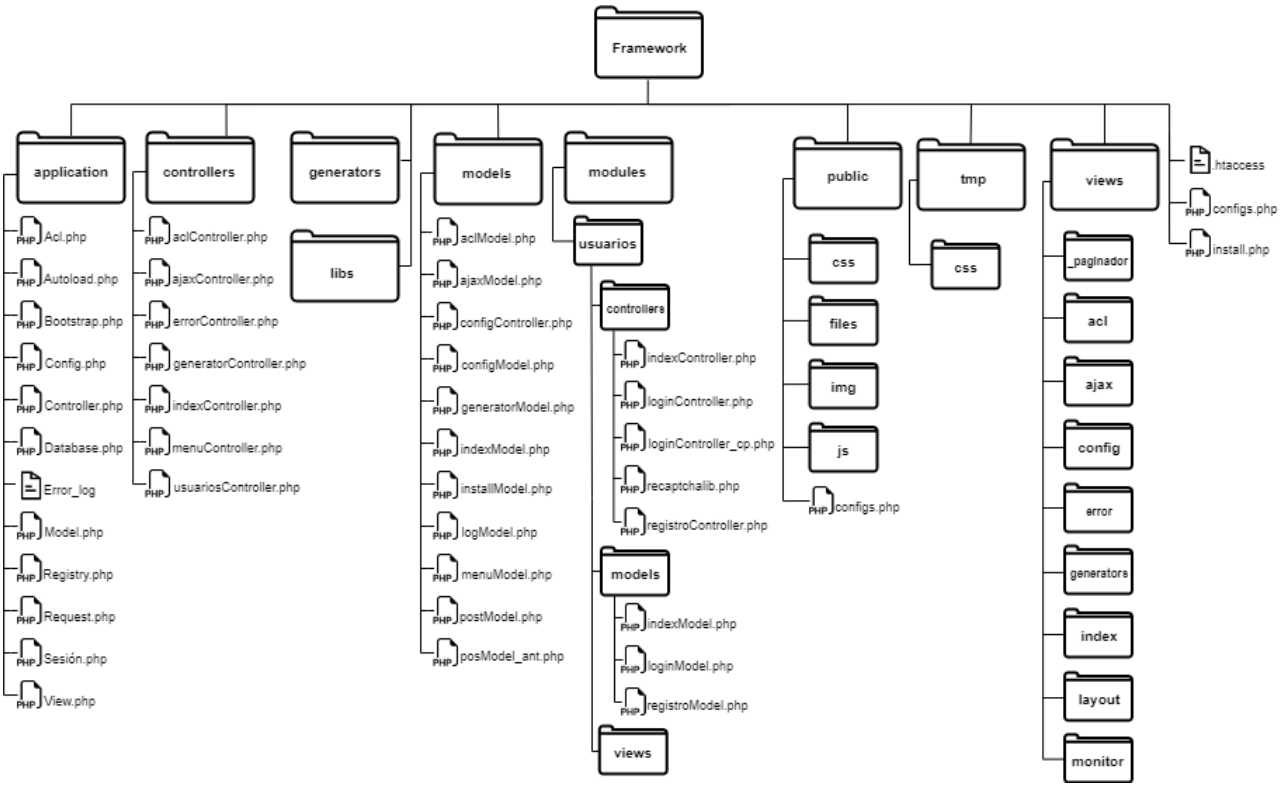

Ilustración 3: Diagrama de organización de archivos. Fuente: Elaboración propia.

Carpeta Application

En la carpeta *application* se encuentran archivos PHP, en los cuales se encuentran las configuraciones básicas del Framework, a continuación, se listarán cada uno de ellos y se dará una breve descripción:

- **Acl.php:** forma parte de la seguridad de la aplicación, como son los permisos y los roles de los usuarios.
- **Autoload.php:** configuraciones del llamado de clases y librerías.
- **Bootstrap.php:** se encarga de identificar los controladores, modelos y vistas, para después **mandarlos llamar.**

- **Config.php:** configuración del acceso a la base de datos, PATHS.
- **Controller.php:** configuración de los controladores.
- **Database.php:** configuración PDO para la base de datos.
- **Error_log:** configuraciones del manejo de errores.
- **Hash:** configuraciones de encriptación.
- **Model.php:** configuraciones de los modelos.
- **Registry.php:** funciones para el registro de la aplicación.
- **Request.php:** en el archivo request se encuentran definidas la reglas de las URL, la forma en la que se interpretan los datos ingresados, para que se tomen como controlador, método o argumentos. En las funciones se dice que el primer dato ingresado en la url después de PATH se tome como el controlador, el segundo como método y los siguientes como argumentos. Request procesa la petición del Bootstrap, que es responsable de las llamadas a los controladores y a los métodos.
- **Session.php:** configuraciones cuando un usuario inicia sesión.
- **View.php:** configuraciones de las vistas.

Carpeta Controllers

Contiene los controladores de la aplicación y sus componentes. Cabe mencionar que dentro de esta carpeta se encuentra el archivo PHP *indexController.php* el cual es el controlador por defecto de la aplicación.

Carpeta Libs

Las clases o librerías de terceros deberían ser ubicadas aquí. Dentro de la carpeta *libs* se encuentra la subcarpeta *smarty*, el cual es un motor de plantillas de PHP, su función es facilitar la separación de la parte lógica del contenido de la aplicación.

Carpeta Models

Esta carpeta contiene los modelos de la aplicación, comportamientos y orígenes de datos.

Carpeta TMP

Dentro de esta carpeta se encuentra la subcarpeta template, en el cual se encuentra el diseño principal de la aplicación.

Carpeta Views

Las vistas de la aplicación son ubicadas aquí, así como el *layout* de la aplicación. Dentro de la carpeta *views* se encuentra la subcarpeta *layout* en el cual se encuentran las carpetas *css, img, js* y el archivo *template.tpl*, en este último encuentra el diseño de la aplicación que no son cambiantes, tal como el *header, footer*, el menú, entre otros. Cabe mencionar que el *layout* puede ser modificado al gusto del usuario, esto se puede lograr modificando los archivos *css, img* y *js* de ser necesario.

Archivo .htacces

Gracias a este archivo podremos configurar nuestro servidor web para hacerlo algo más seguro, pero, además, también podremos realizar redirecciones, crear mensajes de error personalizados, restringir el acceso a carpetas, evitar el listado de directorios de nuestro servidor o permitir el uso de nuestro dominio sin usar las famosas '*www*'. No tiene porqué existir un solo archivo *.htaccess*, sino que puede existir un *.htaccess* para cada directorio si lo consideras necesario.

Archivo index.php

En este archivo se encuentran algunas configuraciones, tal como la instancia de la base de datos y el precargado del archivo de configuración *config.php*

3.2.3 Diagrama UML de componentes del framework

El diagrama UML representa de forma visual la estructura, así como la forma en que está conformado el framework.

A continuación, se da muestra del diagrama UML, donde se pueden observar las clases, atributos y métodos principales que conforman al framework y su relación.

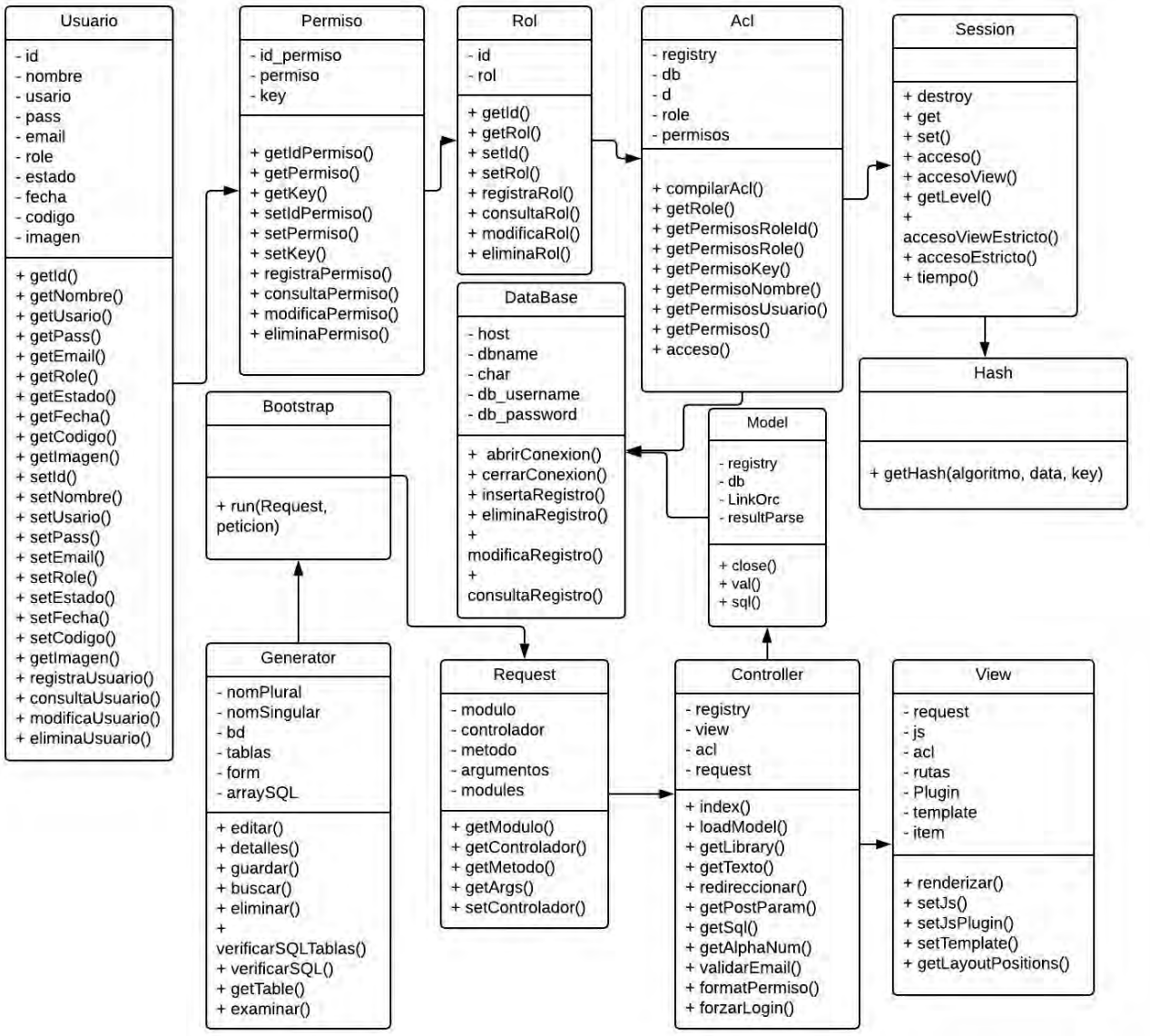

Ilustración 4: Diagrama UML de los componentes del framework. Fuente: Elaboración propia.

3.2.4 Diagrama Entidad-Relación "Usuarios"

Este diagrama Entidad-Relación ayuda a visualizar las relaciones de las tablas de la base de datos del framework. Este diagrama se realizó para mostrar la estructura específica de la tabla de "Usuarios" y la forma en cómo se relaciona con las demás tablas. Esto nos permitirá tener una idea de interconexión entre los permisos a los usuarios, los roles y los permisos de los roles.

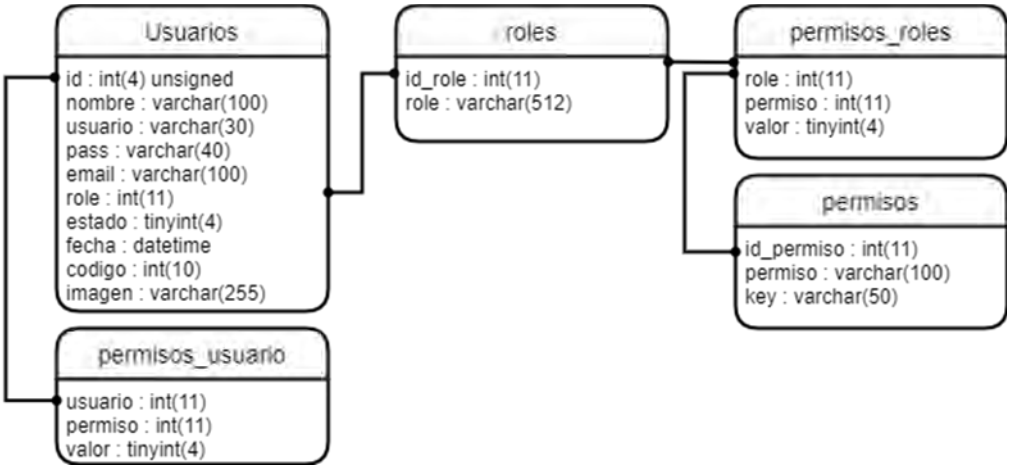

Ilustración 5: Diagrama Entidad-Relación. Fuente: Elaboración propia.

3.2.5 Basado en MCV (Modelo – Vista – Controlador)

Las aplicaciones desarrolladas con este framework siguen el patrón de diseño de software MVC(Modelo-Vista-Controlador). Este patrón separa la aplicación en tres partes:

- **Modelo:** generalmente se encarga de los datos, guardando, consultando o eliminando la información de la base de datos.
- **Vista:** es la representación visual de los datos, todo lo que tenga que ver con la interfaz gráfica. Ni el modelo ni el controlador se preocupan de cómo se verán los datos, esa responsabilidad es únicamente de la vista.
- **Controlador:** maneja y enruta las peticiones hechas por el usuario, se encarga de solicitar los datos al modelo y de comunicárselos a la vista.

En los siguientes ejemplos, se describen en forma de escenarios cada uno estos. Se hace uso de diagramas para facilitar la explicación:

Escenario 1

En este escenario interactúa solo el Controlador. Se da cuando, por ejemplo, el usuario solicita un archivo para su descarga, dicho archivo puede ser un archivo PDF, documento, una imagen, etcétera.

Ilustración 6: Ejemplo del Controlador. Fuente: Elaboración propia.

1. El usuario realiza una petición para descarga de archivo PDF.
2. El controlador captura la petición del usuario.
3. El controlador realiza la petición del archivo al servidor.
4. El servidor encuentra el archivo y lo retorna al controlador.
5. El controlador recibe la información y la envía al usuario.

Escenario 2

Para este escenario es necesario la interacción del Controlador y el Modelo. Este se presenta cuando un usuario guarda datos usando AJAX, y recibe una respuesta en JSON, por lo que no necesita una vista previa para mostrar dicha respuesta.

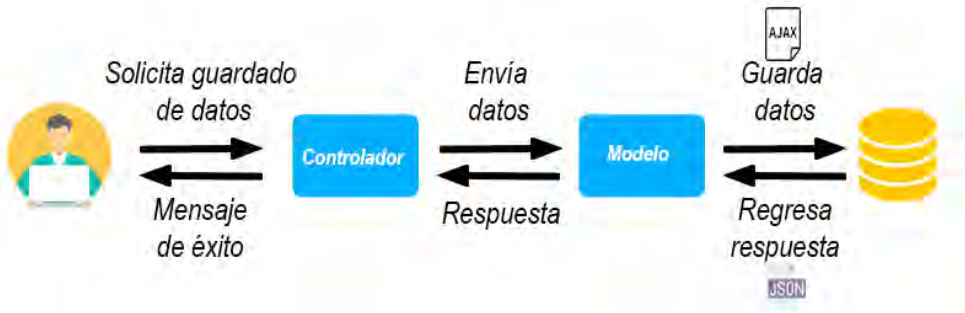

Ilustración 7: Ejemplo de interacción Controlador-Modelo. Fuente: Elaboración propia.

1. El usuario realiza una petición para guardado de información.
2. El controlador captura la petición del usuario.
3. El controlador llama al modelo.
4. El modelo interactúa con la base de datos, guarda la información mediante AJAX y retorna la información en JSON al controlador.
5. El controlador recibe la información y envía mensaje de éxito al usuario.

Escenario 3

En este escenario interactúan el Modelo, la Vista y el Controlador. Este se da cuando el usuario solicita una información y necesita recibirla de forma visual en un archivo con la extensión***.tpl***.

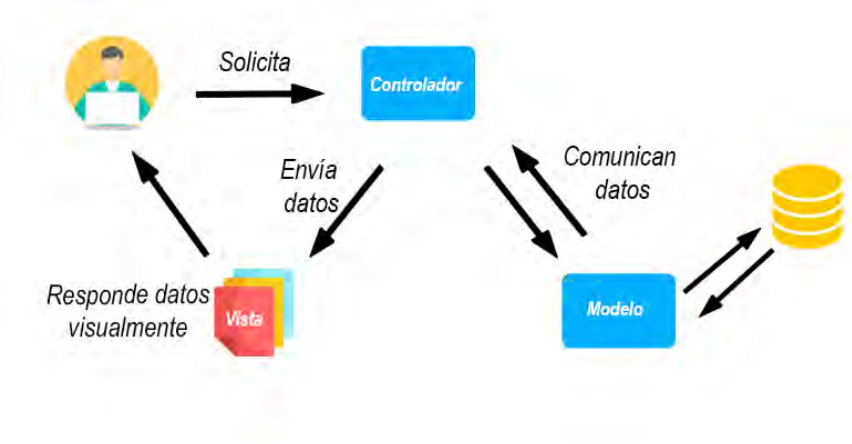

Ilustración 8: Ejemplo de interacción de Modelo-Vista-Controlador. Fuente: Elaboración propia.

1. El usuario realiza una petición.
2. El controlador captura la petición del usuario.
3. El controlador llama al modelo.
4. El modelo interactúa con la base de datos, y retorna la información al controlador.
5. El controlador recibe la información y la envía a la vista.
6. La vista procesa la información recibida y la entrega de una manera visualmente entendible al usuario.

Beneficios del MVC

- Las diferentes partes de la aplicación se separan en módulos independientes. Esto permite desarrollar las distintas partes de forma independiente, además separar las funciones de la aplicación en modelos, vistas y controladores hace que la aplicación sea muy ligera.
- Facilita el manejo de errores, es decir, permite hacer cambios en una parte de la aplicación sin que las demás se vean afectadas.
- Permite que el sistema sea escalable si es requerido, por lo que el sistema podrá adaptarse y crecer. Cuando se cuenta con una base sólida y estructurada, es más sencillo añadir nuevas funcionalidades.

3.2.6 Diagrama de navegación

El Framework trabaja con las llamadas "rutas amigables", en la ilustración 9 se encuentran señaladas dichas partes y lo que representan.

Ilustración 9: Ejemplo URL. Fuente: Elaboración propia.

Visualizando la ilustración 9, se puede decir que la URL del Framework se conforma por:

- **Path:** es la ruta raíz del proyecto, cuando se accede a ella normalmente siempre direcciona hacia el index del proyecto.
- **Controlador:** el segundo componente que compone la URL es el nombre del controlador que se manda llamar, en este caso es el controlador "alumnos".
- **Método del controlador:** el tercer componente es el método del controlador, en este caso es el método "listar".

- **Parámetro del controlador:** el cuarto componente es el parámetro del controlador, en este caso está representado con el número 1, cabe mencionar que puede ser más de un parámetro.

3.2.7 Diseño de las interfaces gráficas

Las interfaces graficas de este Framework son una parte fundamental para el desarrollo de los sistemas informáticos, ya que es el puente entre los usuarios que harán uso de este y la funcionalidad del sistema. De acuerdo con las partes interesadas, el diseño de las interfaces se encuentra orientado al usuario, puesto que se busca que éstas sean fáciles de usar lo que permitirá mejorar la experiencia del usuario, así como optimizar la usabilidad del Framework o sistema desarrollado en éste.

Ilustración 10: Interfaz de la pantalla Inicio de Sesión. Fuente: Elaboración propia.

La pantalla de inicio de sesión es una parte importante y fundamental para el ingreso al Framework, ya que ésta permite que el ingreso al Framework se realice de manera segura. El proceso de creación de la pantalla de inicio de sesión fue sencillo, logrando obtener un diseño básico, tal y como se visualiza en la ilustración 10, ya que el administrador o programados que haga uso del Framework, tendrá las facilidades para realizar modificaciones, permitiendo adecuar el diseño de acuerdo con sus necesidades.

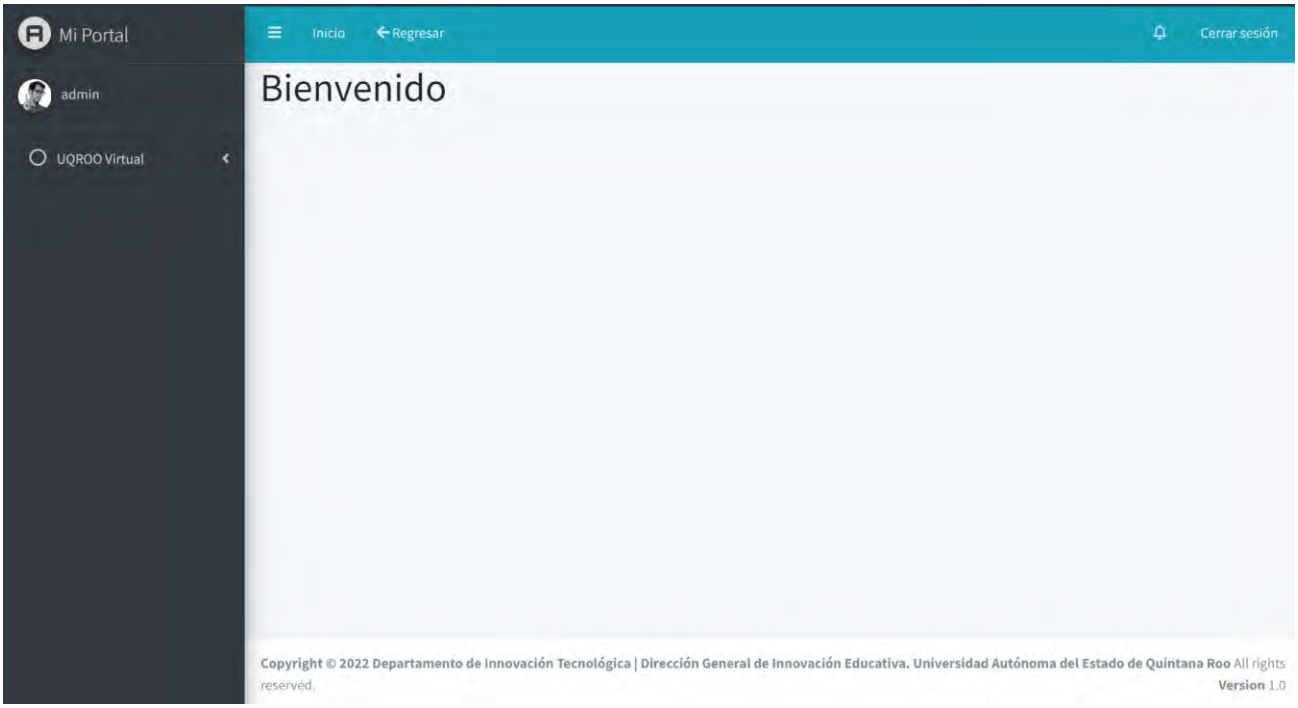

Ilustración 11: Interfaz de la pantalla de inicio. Fuente: Elaboración propia.

La pantalla de inicio como editor o administrador se muestra en la ilustración 11, donde se visualiza una pantalla de "Bienvenida" al ingreso del Framework. Esta pantalla es la plantilla base que tiene por defecto el Framework, la cual puede ser modificada, dependiendo de las necesidades que se tengan, por el administrador del sistema que se desarrolle en éste Framework.

En la ilustración 12 se puede visualizar la interfaz de la creación de la tabla "Ciudades" con la herramienta "Generator", mostrando el diseño implementado de esta herramienta y la forma en cómo está puede ser usada.

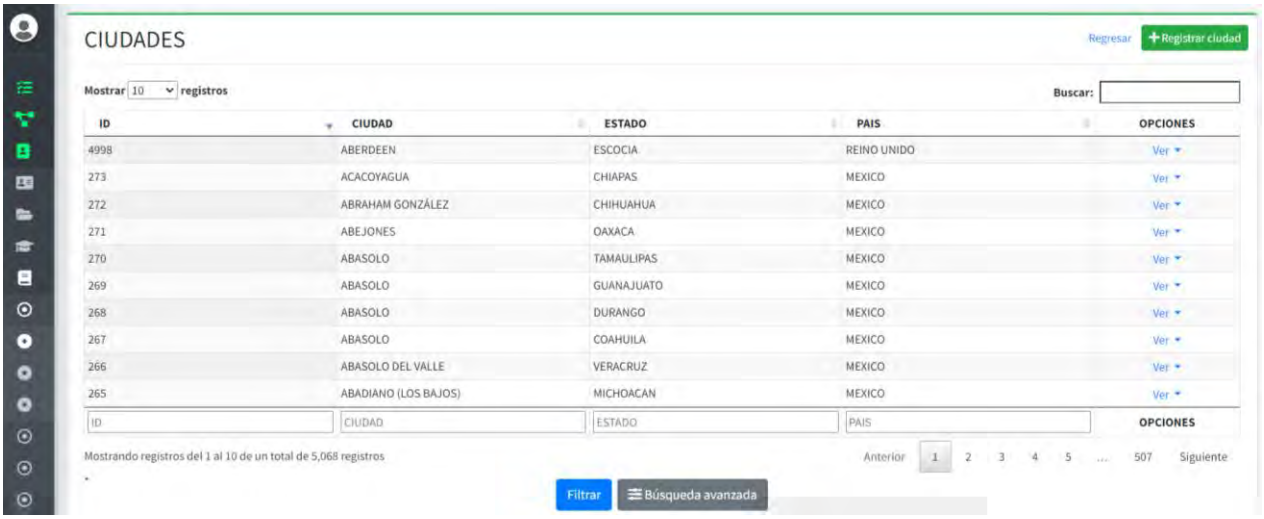

Ilustración 12: Interfaz de la herramienta Generator. Fuente: Elaboración propia.

De acuerdo con todo lo anterior, el diseño de las interfaces gráficas presentado a lo largo de esta sección pretende mejorar la experiencia del usuario, optimizando su usabilidad y, sobre todo, lograr la satisfacción general del usuario. Por lo consiguiente, se espera que la interacción de los usuarios con el sistema se desarrolle sin problemas y de manera fluida, lo que permitirá mejorar la productividad.

3.3 Fase de Codificación

El código de este sistema está basado en la Programación Orientada a Objetos (POO); como se explicó en el Capítulo 2, este utiliza objetos para resolver problemas, generar programas y aplicaciones informáticas.

A continuación, se presenta un ejemplo de la estructura del código en este Framework. Se observa cómo se muestra el Modelo – Vista - Controlador (MVC), que permite la reutilización de código y la separación de conceptos, de manera que se facilite el desarrollo de aplicaciones y posteriormente, su mantenimiento.

aclModel.php

```php
<?php

class aclModel extends Model

{
    public function __construct()
    {
        parent::__construct();
    }
    public function guardar_seccion_rol($post)
    {
        $idRole = $post["idrole"];

        $role = $this->_db->query("DELETE FROM secciones_role_edicion WHERE fk_id_role = {$idRole}");

        $role = $this->_db->query("DELETE FROM secciones_role_prev WHERE fk_id_role = {$idRole}");

        if(isset($post["secceditar"])){

            foreach($post["secceditar"] AS $i){

                $pag = $this->_db->prepare("INSERT INTO secciones_role_edicion VALUES(:idrole,:idseccion)");

                 $res = $pag->execute(

                            array(

                               ':idrole' =>$idRole,

                               ':idseccion' => $i

                            ));
            }
        }
```

Ilustración 13: Creación de un Modelo. Fuente: Elaboración propia.

En la ilustración 13, se muestra el ejemplo de un modelo con la clase *aclModel*, este ayuda a mantener los enlaces entre el Modelo y sus Vistas, y notifica a las Vistas cuando hay algún cambio en el Modelo. Las vistas están creadas por archivos *.tpl,* los cuales incluyen mediante líneas de códigos en HTML lo que se va mostrar en la pantalla.

Un controlador es parte fundamental de este framework, ya que actúa como un intermediario entre la Vista y el Modelo. En la ilustración 14, se muestra un ejemplo de creación de un controlador denominado con la clase *aclController*, el cual recibe los eventos de entrada (cambios en los campos de texto, etc), gestionando el flujo de información entre el Modelo y la Vista.

aclController.php

```php
<?php

class aclController extends Controller
{
    private $_aclm;

    public function __construct()
    {
        parent::__construct();
        $this->forzarLogin();
        $this->_aclm = $this->loadModel('acl');

        $this->_acl->acceso('add_permissions');
    }

    public function index()
    {
        $this->_view->assign('titulo', 'Listas de acceso');
        $this->_view->renderizar('index', 'acl');
    }

    public function roles()
    {
        $this->_view->assign('titulo', 'Administracion de roles');
        $roles = $this->_aclm->getRoles();
        if(is_array($roles))
            $roles = array_map(array($this, 'convierteArrayIndicesMinusculas'), $roles);

        $this->_view->assign('roles', $roles);
        $this->_view->renderizar('roles', 'acl');
    }

```

Ilustración 14: Creación de un Controlador. Fuente: Elaboración propia.

3.3.1. Desarrollo de propuesta de instalación

Para el instalador se realizó un formulario, en el cual se solicitan los datos para creación de los archivos que permiten el arranque y uso del framework.

En la ilustración 15, se puede visualizar el formulario para la instalación del framework, donde en el primer apartado se rellena los datos para la creación de la base de datos.

Ilustración 15: Pantalla de instalación del Framework. Fuente: Elaboración propia.

Los datos solicitados son los siguientes:

1. **Nombre de la base de datos**: Nombre para la base de datos del framework.
2. **Nombre de usuario de la base de datos:** En este punto se debe añadir un nombre de usuario, el cual se guardará en la base de datos y con este se tendrá el acceso como administrador del sistema.
3. **Contraseña:** Incluir una contraseña para el acceso a la base de datos.
4. **Prefijo:** Código que se añade a la base de datos y acceder de manera más rápida.

5. **Base de datos a utilizar**: en este punto se elegirá la tecnología de la base de datos de desea usar, actualmente las opciones son MySQL u Oracle.

En el apartado de "Creación de usuario para iniciar sesión" se va a crear el usuario administrador para el acceso a todo el sistema cuando este ya se encuentre instalado y listo de usar.

1. **Nombre de usuario administrador:** en este punto se deberá incluir un usuario administrador, con el cual se tendrá acceso al sistema para modificaciones y mantenerlo.
2. **Contraseña:** ingreso de una contraseña para el administrador que incluye una configuración de encriptación MD5.
3. **Correo:** un correo electrónico para la administración del sistema, así como para tener un modo de recuperación de contraseña.

La parte de codificación en este archivo incluye las instrucciones para la creación de las bases de datos durante el proceso de instalación, creando las tablas necesarias para el funcionamiento correcto de esta. Esto será posible siempre y cuando se llene de la manera correcta el formulario para poder realizar su creación, lo que permitirá un adecuado funcionamiento sin problema alguno.

3.3.2 Implementación de herramientas y uso del módulo "Generator"

Componentes del módulo "Generator"

Se entiende por componente a los elementos que puede tener un formulario para representar un conjunto de datos resultante después de la edición de los campos de usuario para acceder a un servicio de información. A continuación, se describen ejemplos de los componentes que forman parte del Generator:

- **CRUD:** En este Framework se implementó CRUD, que se utiliza para realizar la generación de formularios y administración de la base de datos. Este crea formularios para las tablas de la base de datos, así como agiliza y facilita el arranque de está.

- **URL de archivo con ventana de navegación:** se utiliza para campos de entrada que deben contener una dirección URL. En función de soporte de los navegadores, el campo URL se puede validar de forma automática cuando se presente.
- **Subida de archivo:** define un campo de selección de archivo y un botón "Examinar" para la carga de archivos.
- **Subida de imagen con miniatura:** función de inserta imagen y crear una imagen con una miniatura uniforme al tamaño.
- **Listas (select) dependientes:** consiste en el uso de dos o más listas cuyos sus valores se van cargando, dependiendo de la opción elegida en la lista anterior.
- **Búsqueda:** se utiliza para campos de búsqueda (un campo de búsqueda se comporta como un campo de texto normal).
- **Tablas:** para mostrar contenido en su sitio web. Consta de una o varias filas, cada una de las cuales consta, a su vez, de una o más celdas.
- **Datalist:** elemento especifica una lista de instrucciones para localizar nuevo archivo de datos, leer, traducir y crear un nuevo archivo activo.
- **Radio:** Los botones de radio permiten al usuario seleccionar sólo una de un número limitado de opciones.
- **Datetime:** especifica un campo de entrada de fecha y hora, sin zona horaria.
- **Week:** permite al usuario seleccionar una semana y el año.
- **Number:** define un campo numérico de entrada.
- **Range:** define un control para la introducción de un número cuyo valor exacto no es importante (como un control deslizante). Rango por defecto es de 0 a 100.
- **Email:** se utiliza para campos de entrada que deben contener una dirección de correo electrónico.
- **File:** define un campo de selección de archivo y un botón "Examinar" para la carga de archivos.

Funcionalidades del módulo "Generator"

La creación de catálogos son una parte importante para el funcionamiento de los sistemas informáticos. Por ese motivo, se implementó el módulo "Generator", el cual incorpora diversas funciones para agilizar y facilitar la creación de los catálogos en el sistema. A continuación, se describen las funcionalidades de "Generator".

- **Funcionalidades de duplicar registro:** Se puede duplicar un registro de una tabla, así como editar y eliminar, todo esto añadiendo un botón con la respectiva función.
- **Búsqueda con filtros avanzados:** Para la búsqueda de registros en la base de datos, se hace uso de jQuery, Ajax, PHP y la base de datos ya sea MySQL u Oracle.
- **Ordenar visualización de registros:** Mediante consultas, se puede seleccionar el orden en el cual se requiere visualizar las consultas y registros las tablas con su respectiva información que se encuentra almacenada en la base de datos.
- **Botón de activar y desactivar:** Un botón con una opción el cual funciona para activar o desactivar páginas de información o de registro.
- **Paginación AJAX:** La paginación para la visualización de resultados al momento de realizar una consulta de información a la base de datos. Mediante jQuery, Ajax y PHP se visualizan los registros e información de una base de datos.
- **Cambiar orden de registros (arrastrar y soltar):** Función que mediante el uso de JavaScript, paginación Ajax, jQuery, se pueden ordenar los registros de una base de datos.
- **Tablas para el catálogo:** Lo primero que se realiza es listar por medio del array *"$tablas"* el conjunto de tablas que intervienen para realizar el catálogo. El contenido del array es el siguiente:
    - **nom:** nombre de la tabla.
    - **id:** nombre de la llave primaria de la tabla.
    - **getId:** sql para obtener el ID siguiente, para insertar un registro nuevo; si el id es *autoincrement*, no se agrega.

- **Formulario para inserción y edición:** Se establecen los campos que tendrá el formulario para insertar un nuevo registro o actualizar uno existente. Algunas propiedades son obligatorias y otras opcionales, a continuación, se listan:
    - **campo** (Obligatorio): nombre del campo de la tabla, que representa.
    - **tipo** (Obligatorio): tipo de campo, en este caso es del tipo "*text*".
    - **label** (Opcional): texto que llevará la etiqueta del campo de texto.
    - **holder (**Opcional): Texto que va dentro del campo, el cual ayuda al usuario a saber qué información escribir.
    - **pattern (**Opcional): expresión regular que regula el contenido de la información que se teclea en el campo de texto. Limita al usuario a escribir determinados tipos de caracteres, ya sea números o letras. Ejemplo: *('pattern'=>'[A-Z]{1,512}')*
    - **tabla** (Opcional): se indica la tabla al que pertenece el campo, de acuerdo con el listado de tablas definido con anterioridad; cuando no se especifica se toma la primera tabla declarada. Ejemplo: *('tabla' =>'d').*
    - **col** (Opcional): Se especifica el tamaño de las columnas con las clases css que Bootstrap define para cada tamaño de pantalla. Por ejemplo *.col-md-XX*, donde *XX* es el tamaño de la columna, que podrá tomar valores entre 1 y 12.
    - **required** (Opcional): Especificamos si el campo es obligatorio para que el formulario sea enviado.

- **Menú de opciones en el catálogo: (CRUD)** El menú de opciones del catálogo es una lista de opciones que se despliega al seleccionar el botón *Ver*, que se encuentra en la columna *Opciones.*

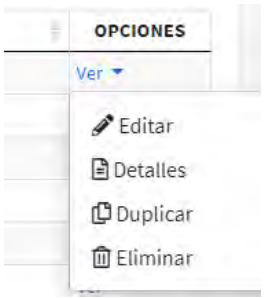

Ilustración 16: Menú de opciones. Fuente: Elaboración propia.

Por defecto este menú cuenta con las opciones de *Editar, Detalles*, *Duplicar* y *Eliminar*, pero también es posible añadir más opciones por medio de la opción *btnOpciones* en un array *$bd.* Cada opción en el menú es un array con las siguientes propiedades:

- **label:** texto que se va a visualizar al momento de desplegar el menú.
- **class:** propiedad para agregar una o más clases a la opción.
- **href:** propiedad donde indicamos el URL o función en JavaScript a la que queremos dirigirnos al darle clic a la opción.
- **target:** propiedad que indica donde se abrirá la URL indicada en *href*, los valores pueden ser *_blank* (abre el URL en una nueva ventana o pestaña) o *_self* (abre el URL en la misma ventana o pestaña).
- **mostrar_si:** propiedad para definir la condición que se usará para mostrar la opción al usuario; si no se cumple no se mostrará.

- **Establecer valores dentro de las opciones:** Para establecer un valor en las opciones es necesario escribir el nombre de la columna como aparece en la consulta sql que se definió en la opción *sqlDeplegar* en el array *$bd*. Ejemplo:

```
'eliminar' => array(
      'mostrar_si' => '[STATUS] != "ACTIVO" ',
),
```

Ilustración 17: Consulta SQL para establecer valores. Fuente: Elaboración propia.

El nombre de la columna *ESTATUS* debe estar entre corchetes para que pueda ser remplazada por su valor en la consulta sql.

Módulos adicionales del Framework:

En este framework, se añadieron nuevas funciones para que el desarrollador o el usuario pueda hacer uso de esta herramienta en el Framework. Unos ejemplos de estos módulos adiciónales se presenta a continuación:

- **Importador del Generator:** A partir de una tabla existente o por medio de un formulario de creación, se podrá generar el conjunto de "*arrays"* que conforman al generator para importar la información.
- **Exportador del Generator:** A partir del del conjunto de *"arrays"* que conforman al Generator se podrá exportar creando el esquema de Controlador, Modelo y vista, para su edición manual posterior.
- **Instalador:** Formulario para ingresar la configuración inicial (credenciales de acceso a base de datos, usuario y contraseña del administrado) y la creación de las tablas en base de datos.
- **ConsultorSQL:** Herramienta para la ejecución de consultas SQL con instrucciones más ágiles de escribir, sin establecer las especificaciones SQL de una base de datos en particular.

3.3.3 Desarrollo del módulo "Roles de usuario"

El desarrollo del módulo "Roles de Usuario" hace referencia al proceso de crear una funcionalidad dentro del framework la cual permita la administración y asignación de roles a los usuarios. Durante el desarrollo de este módulo se diseñó e implementó una interfaz de usuario que sea de fácil uso, que le permita al administrador del sistema la creación, edición y modificación de roles, así como la asignación de ciertos roles a aquellos usuarios que ya se encuentren registrados dentro del sistema.

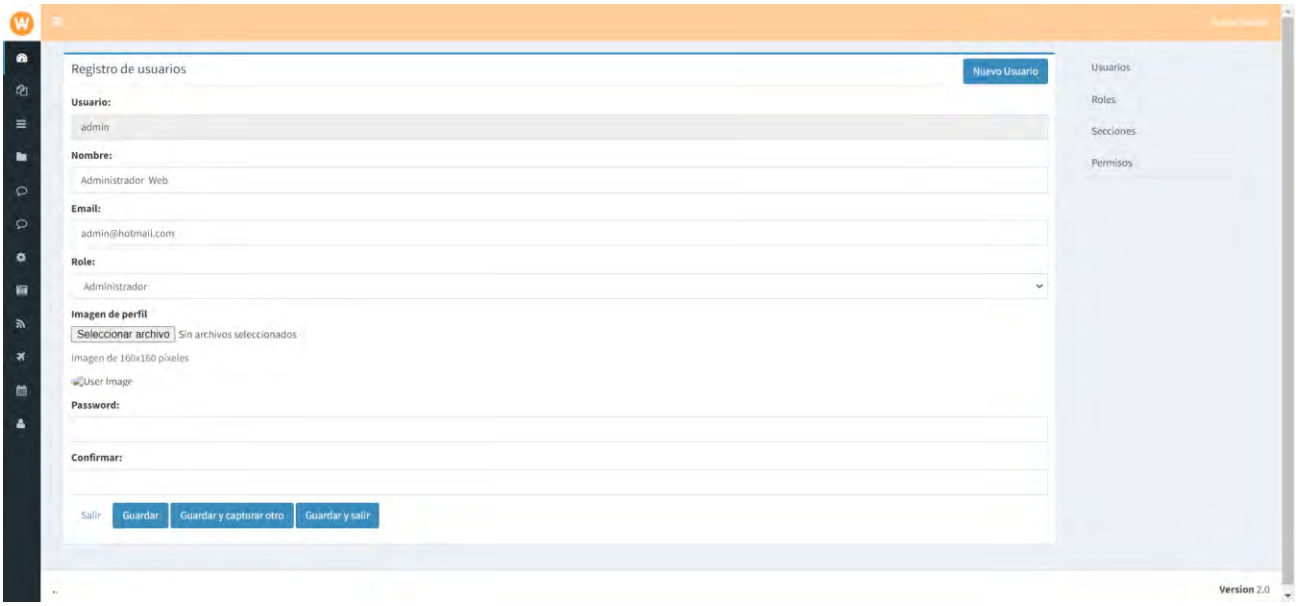

Ilustración 18: Interfaz registro de usuarios. Fuente: Elaboración propia.

En la ilustración 18 se observa la ventana "registro de usuarios", perteneciente al panel de roles de usuario. En esta ventana se muestra un formulario en el cual se realiza el registro de los nuevos usuarios en el sistema, dándoles un nombre, un correo electrónico, la asignación de un rol, una imagen de perfil, así como una contraseña para el inicio de sesión.

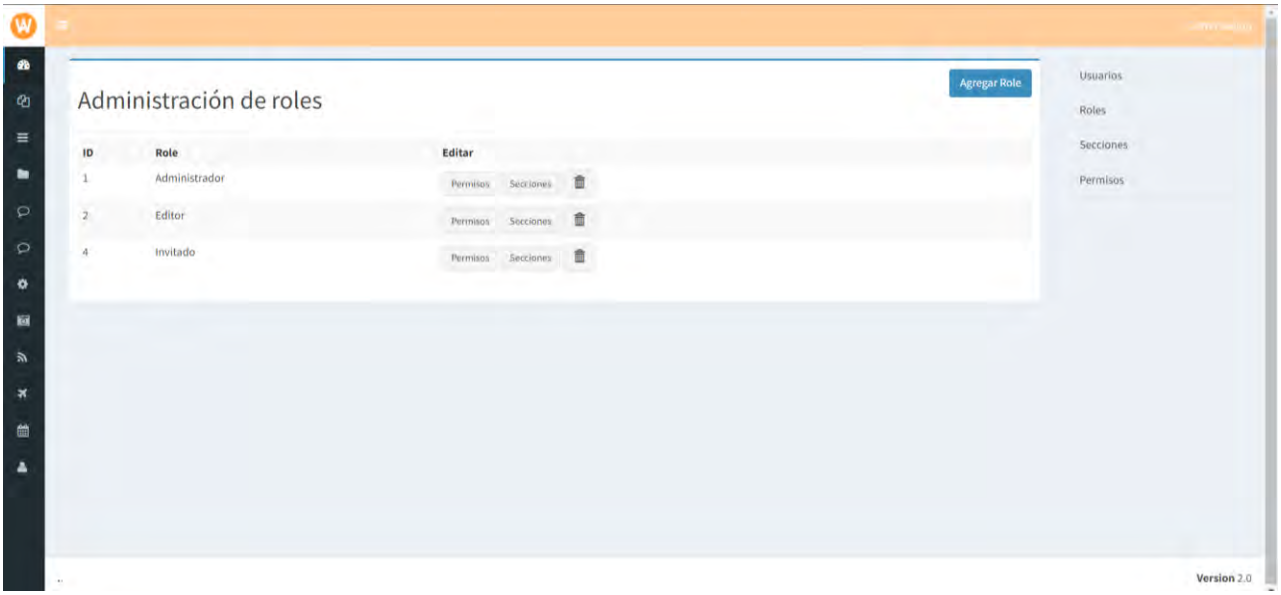

Ilustración 19: Interfaz administración de roles. Fuente: Elaboración propia.

La administración de permisos en el framework se refiere a la parte de asignar y gestionar los privilegios de acceso a los usuarios o roles. En la ilustración 19 se aprecia como el administrador del sistema ahora puede establecer la configuración de los roles de cada uno de los usuarios y, de igual manera, asignarlos como administrador, editor o invitado.

Ilustración 20: Interfaz administración de permisos. Fuente: Elaboración propia.

El administrador del Framework es el que se encarga de la identificación de usuarios y roles, así como definir los **permisos** que se pueden otorgar, como lectura, escritura, eliminación, creación o ejecución de ciertas acciones en el sistema. Este realiza la asignación de los permisos apropiados para cada rol, además de que gestiona su mantenimiento y actualización, como agregar, modificar o revocar permisos de acuerdo con las necesidades del momento. De igual manera, este mantiene un control de acceso para garantizar que los usuarios solo puedan acceder a ciertas características del sistema, dependiendo de sus roles asignados y los permisos que tengan cada uno de estos. En la ilustración 20 se muestra una tabla en la cual se enlistan los permisos que el administrador puede asignar a cada rol que se encuentre agregado en el sistema.

Los roles de usuario son una herramienta importante en la administración de permisos, ya que estos facilitan la gestión, mejoran la seguridad y brindan flexibilidad en el control de accesos. Al asignar permisos de manera apropiada y establecer roles adecuados, se mantiene un equilibrio entre la seguridad y la eficiencia en el uso del Framework y de los sistemas desarrollados en este.

3.3.4 Estabilidad y seguridad en el Framework

Para garantizar la estabilidad de los sistemas, se hace uso del patrón de diseño ***Singleton***, este logra limitar el número de estancias posibles de una clase de nuestro sistema, y proporciona un acceso global al mismo.

Un uso y ejemplo implementado de esto se pude visualizar en la ilustración 21, donde se puede apreciar que el método constructor es privado, lo que evita que pueda ser instanciada la clase (desde fuera), pero se encuentra un método estático ***"getInstancia"*** que puede ser evocado sin necesidad de instanciar la clase, en este método se hace la instancia de la misma clase una sola vez.

```php
<?php
class Registry
{
    private static $_instancia;
    private $_data;

    //no se puede instanciar la clase
    private function __construct() {}

    //singleton
    public static function getInstancia()
    {
        if(!self::$_instancia instanceof self){
            self::$_instancia = new Registry();
        }
        return self::$_instancia;
    }

    public function __set($name, $value)
    {
        $this->_data[$name] = $value;
    }

    public function __get($name)
    {
        if(isset($this->_data[$name]))
        {
            return $this->_data[$name];
        }
        return false;
    }
}
?>
```

Ilustración 21: Implementación de Singleton. Fuente: Elaboración propia.

Control de errores

El sistema debe llevar un control de errores y su registro en el sistema. Para esto, se creó un archivo para el registro y manejo de errores que se generan durante el uso de framework. El archivo se denominó *"error_log"*. En este archivo, localizado en la carpeta *application*, se mantendrá la información sobre los errores generado; el usuario podrá tener un control y darle un orden para resolver los errores generados.

error_log

```
[27-Aug-2016 17:34:25 Etc/GMT] PHP Warning:  include(): http:// wrapper is disabled in the server configuration by allow_url_include=0 in /home/pasealwm/public_html/paqueteanimes.com/admin/application/tmp.php on line 3
[27-Aug-2016 17:34:25 Etc/GMT] PHP Warning:  include(http://www.paqueteanimes.com/template/lte/config.php): failed to open stream: no suitable wrapper could be found in /home/pasealwm/public_html/paqueteanimes.com/admin/application/tmp.php on line 3
[27-Aug-2016 17:34:25 Etc/GMT] PHP Warning:  include(): Failed opening 'http://www.paqueteanimes.com/template/lte/config.php' for inclusion (include_path='.:/opt/alt/php54/usr/share/pear:/opt/alt/php54/usr/share/php') in /home/pasealwm/public_html/paqueteanimes.com/admin/application/tmp.php on line 3
[27-Aug-2016 17:34:25 Etc/GMT] PHP Notice:  Undefined variable: config in /home/pasealwm/public_html/paqueteanimes.com/admin/application/tmp.php on line 7
[27-Aug-2016 17:34:38 Etc/GMT] PHP Warning:  include(): http:// wrapper is disabled in the server configuration by allow_url_include=0 in /home/pasealwm/public_html/paqueteanimes.com/admin/application/tmp.php on line 3
[27-Aug-2016 17:34:38 Etc/GMT] PHP Warning:  include(http://www.paqueteanimes.com/template/lte/config.php): failed to open stream: no suitable wrapper could be found in /home/pasealwm/public_html/paqueteanimes.com/admin/application/tmp.php on line 3
[27-Aug-2016 17:34:38 Etc/GMT] PHP Warning:  include(): Failed opening 'http://www.paqueteanimes.com/template/lte/config.php' for inclusion (include_path='.:/opt/alt/php54/usr/share/pear:/opt/alt/php54/usr/share/php') in /home/pasealwm/public_html/paqueteanimes.com/admin/application/tmp.php on line 3
[27-Aug-2016 17:34:38 Etc/GMT] PHP Notice:  Undefined variable: config in /home/pasealwm/public_html/paqueteanimes.com/admin/application/tmp.php on line 7
```

Ilustración 22: Archivo "error_log". Fuente: Elaboración propia.

Garantizar seguridad en el sistema

La seguridad en los sistemas es importante para el desarrollo y uso de este framework, ya que esta permite mantener un control en riesgos y ataques que día a día pueden sufrir los sistemas como lo desarrollados en empresas o instituciones de mayor importancia y relevancia. En este apartado, el enfoque principal no se trata de tener una buena seguridad, puesto que todo sistema siempre es propenso a las amenazas y vulnerabilidades, por lo que siempre hay que estar identificando y actualizando. Más que nada, unos de los aspectos generales para garantizar la seguridad en este framework es la implementación de algunas herramientas que ayuden a tener un control en los sistemas desarrollados. Por lo tanto, en esta sección se abordará dos ejemplos: 1) un control de seguridad, de los que fueron implementados en este framework, siendo esta una pantalla de inicio de sesión, en la cual se incorporó un CAPTCHA; y 2) la forma en cómo se introduce una inyección en el mismo formulario y cómo se evita la introducción en código malicioso dentro de esta mismo.

- **Pantalla de inicio de sesión:** Para obtener acceso al sistema, los usuarios deberán ingresar desde una interfaz un usuario y contraseña, este proceso de identificación es conocido como identificación de usuario o acceso del usuario al sistema. En la ilustración 23 se muestra un ejemplo de un formulario para el inicio de sesión en el sistema:

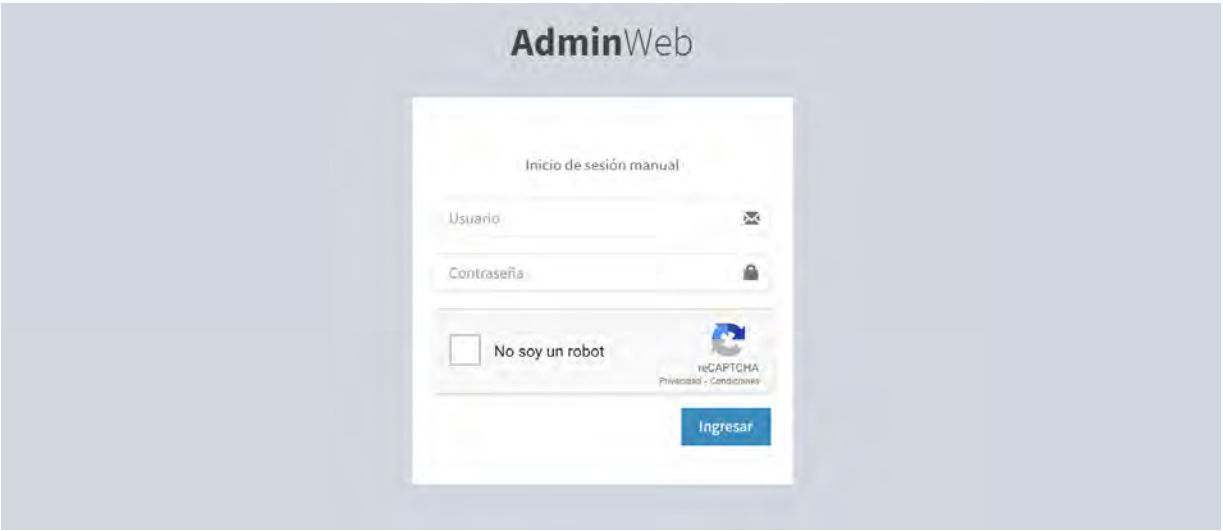

Ilustración 23: Pantalla de inicio de sesión con CAPTCHA. Fuente: Elaboración propia.

- **CAPTCHA:** Un CAPTCHA o *Completely Automated Public Turing test to tell Computers and Humans Apartsus*, que traducido al español dice: Prueba de Turing pública y automática para diferenciar a máquinas y humanos. En otras palabras, esta prueba se trata de para poder determinar si al momento de iniciar sesión, este es de un humano o no.

  La prueba de esta consiste en una pregunta que necesite una respuesta, donde pueden ser diferentes tipos de ejercicios por resolver. Los ejercicios pueden ser variados, desde seleccionar diversos tipos de imágenes o ingresar un texto que está incluido en una imagen. En la ilustración 24, se muestra un ejemplo de CAPTCHA al momento de realizar un inicio de sesión en el sistema.

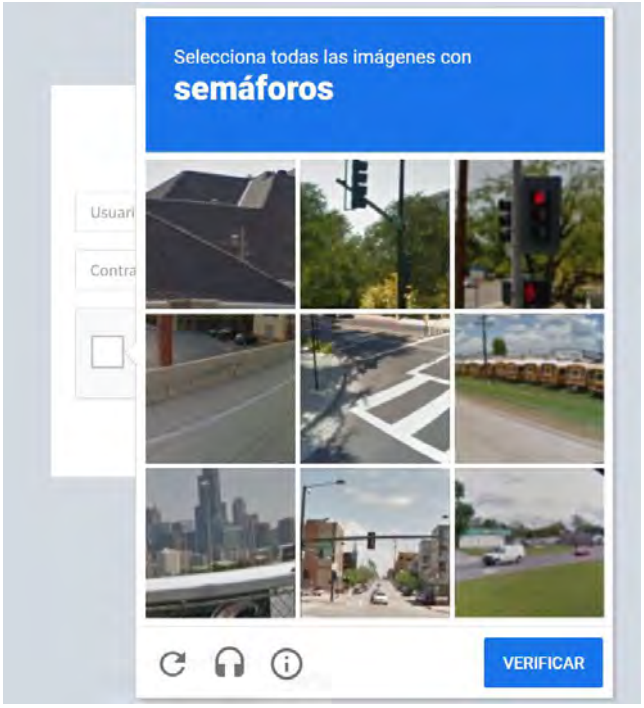

Ilustración 24: Uso de CAPTCHA. Fuente: Elaboración propia.

Los captchas sirven para limitar el acceso a determinado tipo de registros, algunos ejemplos de su utilización e implementados para este sistema son:

- Proteger el registro al sistema.
- Proteger el acceso a las cuestas de correo electrónico.
- Prevenir ataques de diccionario.

- Prevenir votaciones masivas.
- Evitar buscadores y bots.
- Evitar spam en general.

Inyecciones SQL: Una inyección SQL es una ejecución de sentencias SQL en una consulta mediante la manipulación de la entrada de datos de una aplicación. Con la inyección SQL se puede conseguir validaciones de entrada, extracción, modificación de dato se incluso el compromiso total del servidor. Un ejemplo de una inyección SQL, se puede visualizar en la ilustración 25, donde al ingresar en el formulario de inicio de sesión el usuario y la contraseña, esta manda una sentencia SQL y verifica si los datos son correctos y poder iniciar sesión.

```
<div style="display: nonex;">

        <p class="login-box-msg">Inicio de sesi&oacute;n manual</p>
        <form name="form1" method="post" action="">
            <input type="hidden" value="1" name="enviar" />
            <div class="form-group has-feedback">
                <input id="usuario" name="usuario" value="{$datos.usuario|default:""}" type="text" class="form-control" placeholder="Usuario">
                <span class="glyphicon glyphicon-envelope form-control-feedback"></span>
            </div>
            <div class="form-group has-feedback">
                <input id="pass" name="pass" type="password" class="form-control" placeholder="Contrase&ntilde;a">
                <span class="glyphicon glyphicon-lock form-control-feedback"></span>
            </div>

          {if $captcha eq true}
           <div class="g-recaptcha" data-sitekey="{$clave_sitio}"></div>
           {/if}
            <div class="row" style="margin-top:10px;">
                <div class="col-xs-8">
                </div>
                <div class="col-xs-4">
                    <button type="submit" class="btn btn-primary btn-block btn-flat">Ingresar</button>
                </div>
                <!-- /.col -->
            </div>
        </form>
</div>
```

Ilustración 25: Inyección SQL. Fuente: Elaboración propia.

En la ilustración 25. se puede observar un ejemplo de cómo se introduce una inyección SQL, la cual se puede evitar utilizando consultas parametrizadas. Éstas consultas son una técnica aplicada en la programación de las bases de datos para realizar la ejecución de consultas SQL de manera eficiente y segura, las cuales proporcionan seguridad y evitan la inyección de SQL.

3.4 Fase de Pruebas

3.4.1. Elaboración del plan de pruebas

Después de todo lo desarrollado y documentado con anterioridad, en este apartado se enlistan las pruebas que serán necesarias para lograr verificar las cosas que funcionan y las que no. Esto permitirá comprobar el estado en el que se encuentra el Framework y las posibles áreas de mejora.

Pruebas unitarias

Tabla 1: Pruebas unitarias del Framework

No.	Prueba	Ejecución
1	Pantalla de instalación del Framework.	Satisfactorio
2	Ejecución de la base de datos con MySQL.	Satisfactorio
3	Ejecución de la base de datos con Oracle.	Satisfactorio
4	Pantalla de inicio de sesión.	Satisfactorio
5	Visualización de la pantalla de inicio.	Satisfactorio
6	Creación de un controlador.	Satisfactorio
7	Creación de una vista.	Satisfactorio
8	Creación de un modelo.	Satisfactorio
9	Uso del Generador. (Crear, editar, borrar, desplegar) (individual).	Satisfactorio
10	Llamado a un método de un controlador.	Satisfactorio
11	Creación de un catálogo.	Satisfactorio
12	Uso de AJAX para extraer información en la vista.	Satisfactorio

Fuente: Elaboración propia

Pruebas del sistema

Tabla 2: Pruebas del sistema

No.	Prueba	Ejecución
1	Correcto funcionamiento en diferentes entornos de S.O.	Satisfactorio
2	Velocidad, capacidad y eficiencia.	Satisfactorio
3	Rendimiento y la capacidad de escalabilidad.	Satisfactorio
4	Rendimiento y la eficiencia de las consultas y operaciones en la base de datos.	Satisfactorio
5	Interoperabilidad entre bases de datos.	Satisfactorio
6	Uso en diferentes versiones de MySQL.	Insatisfactorio
	Observación: • Características y funciones pueden variar según la versión. • Problemas de incompatibilidad con algunas bibliotecas.	
7	Uso en diferentes versiones de ORACLE.	Insatisfactorio
	Observación: • Características y funciones pueden variar según la versión. • Problemas de incompatibilidad con algunas bibliotecas.	
8	Usabilidad en diferentes navegadores web.	Satisfactorio

Fuente: Elaboración propia

Pruebas de interfaces

Tabla 3: Pruebas de interfaces

No.	Prueba	Ejecución
1	Pantalla de inicio de sesión.	Satisfactorio
2	Validación de formularios.	Satisfactorio
3	Navegación entre páginas.	Satisfactorio
4	Visualización de datos de una tabla.	Satisfactorio
5	Búsquedas.	Satisfactorio
6	Filtrado de resultados.	Satisfactorio
7	Capacidad de respuesta de la interfaz.	Satisfactorio

8	Visualización adecuada de diferentes dispositivos y tamaños de pantalla.	Insatisfactorio
	Observación: • Tamaños de pantalla no están adaptados para dispositivos móviles.	
9	Consistencia en el diseño y la navegación.	Satisfactorio
10	Accesibilidad de la interfaz.	Satisfactorio
11	Compatibilidad con navegadores web.	Satisfactorio
12	Deshacer y rehacer acciones.	Insatisfactorio
	Observación: • No se encuentra contemplado, se notificó para futuras implementaciones.	

Fuente: Elaboración propia

3.4.2. Ejecución de las pruebas

Una vez enlistadas las pruebas unitarias y de sistemas, se procede a evaluar el funcionamiento del Framework. En la tabla 4, se observa un resumen de las pruebas realizadas, así como el estado general de las mismas. En el anexo 1, se pueden encontrar las pruebas de forma detallada, en las cuales se evalúa el procedimiento para la ejecución de cada prueba y se analiza el resultado obtenido en cada una de estas, para que, de igual forma, se presenten las observaciones correspondientes.

Tabla 4: Lista de la ejecución de las pruebas

No.	Prueba	Estado de la prueba
1	Archivo de instalación del Framework	Exitoso
2	Uso de la herramienta Generator	Exitoso
3	Administración de roles de usuarios	Exitoso
4	Seguridad del Framework	Exitoso
5	Conexión del Framework con la base de datos (MySQL u Oracle)	Exitoso

Fuente: Elaboración propia

3.5 Documentación

La documentación para el uso de este framework se encuentra conformada por un manual de usuario, en el cual se explica la creación de un modelo, una vista y un controlador; estos son creados directamente de la edición de los archivos de la carpeta del Framework. Este manual tiene como objetivo, no solamente mostrar la forma en cómo modificar el Framework de manera gráfica, sino que, de igual manera, lograr realizar cambios directamente desde el código fuente de éste.

Seguidamente, en el apartado del condicionamiento del entorno, se especifica la licencia que utiliza este Framework, así como los requisitos básicos para su instalación, el cual puede usar dos tipos de servidores web, ya sea uno local o uno que se encuentre en la nube. Además, se describe su instalación y cómo realizar la configuración del entorno para poder hacer uso de la base de datos Oracle en el Framework.

3.5.1. Manual del usuario

3.5.1.1 Creación de un modelo

Para crear un modelo se debe crear un archivo PHP dentro de la carpeta ***models*** de la aplicación. Como se muestra en la siguiente ilustración 26:

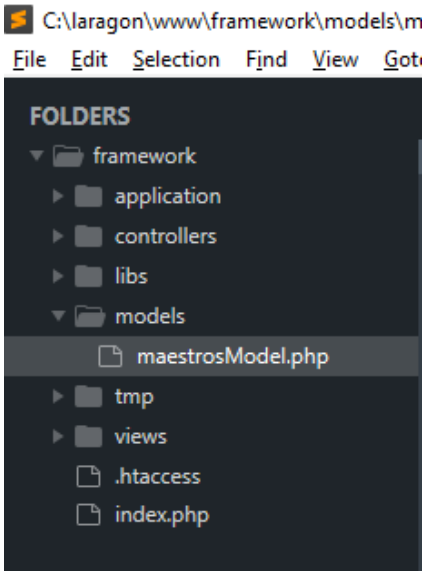

Ilustración 26: Carpeta models. Fuente: Elaboración propia.

Dentro del archivo PHP, se debe escribir el nombre del modelo seguido de la para ***Model***. Tal como se muestra en el siguiente ejemplo del modelo denominado ***maestrosModel***. Es importante tener en cuenta que todos los modelos deberán extender de la clase padre ***Model***.

```php
<?php

class maestrosModel extends Model{

    public function __construct(); {
        parent::__construct();
    }
    function mostrarMaestros(){
        $sql = "SELECT * FROM maestros";
        $post = $this->_db->query($sql);
        return $post->fetchall();
    }
    function obtieneMaestros($id){
        $sql = "SELECT * FROM maestros WHERE id='$id'";
        $post = $this->_db->query($sql);
        return $post->fetch();
    }
    function actualizaMaestros($p){
        $sql = "UPDATE maestros SET nombre='".$p["nombre"]."' WHERE id='".$p["id"]."'";
        $post = $this->_db->query($sql);
    }

?>
```

Ilustración 27: Ejemplo creación de un modelo. Fuente: Elaboración propia.

3.5.1.2 Creación de una vista

Para crear una vista, se tiene que crear una carpeta dentro de la carpeta ***views*** con el mismo nombre que tiene el controlador al que se le desea crear dicha vista.

Por ejemplo, si se tiene un controlador llamado ***maestrosController***, dentro de la carpeta ***views*** se tendrá que crear una carpeta llamada maestros y dentro de la dicha carpeta crear un archivo con el nombre deseado, con la ***extensión.tpl***. En la ilustración 28, se muestra un ejemplo de este:

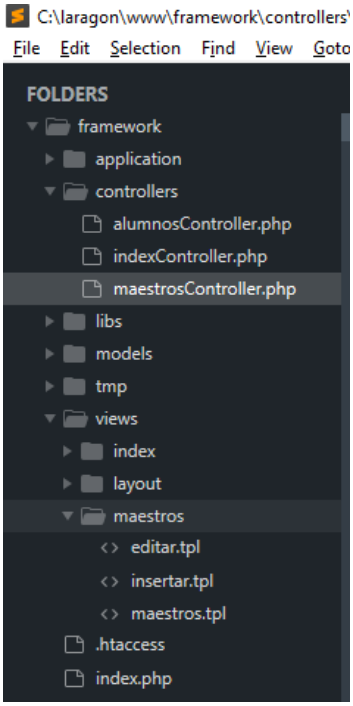

Ilustración 28: Carpeta views. Fuente: Elaboración propia.

Cabe mencionar que la vista deberá ser llamada desde el controlador con el mismo nombre que se le asignó. En la ilustración 29 se puede visualizar un ejemplo:

```php
<?php

class maestrosController extends Controller{

    private $_maestros;

    public function __construct() {
        parent::__construct();
        $this->_maestros = $this->loadModel('maestros');
    }
    public function index(){
        $res = $this->_maestros->mostrarMaestros();
        $this->_view->assign("maestros",$res);
        $this->_view->renderizar('maestros');
    }
    public function insertar(){
        $this->_view->renderizar('insertar');

    }
    public function editar($id){
        $res = $this->_maestros->obtieneMaestros($id);
        $this->_view->assign("info",$res);
        $this->_view->renderizar('editar');
    }
}

?>
```

Ilustración 29: Ejemplo creación de una vista. Fuente: Elaboración propia.

3.5.1.3 Creación de un controlador

Para crear un controlador se debe crear un archivo PHP dentro de la carpera ***controllers*** de la aplicación, tal y como se muestra en la siguiente ilustración:

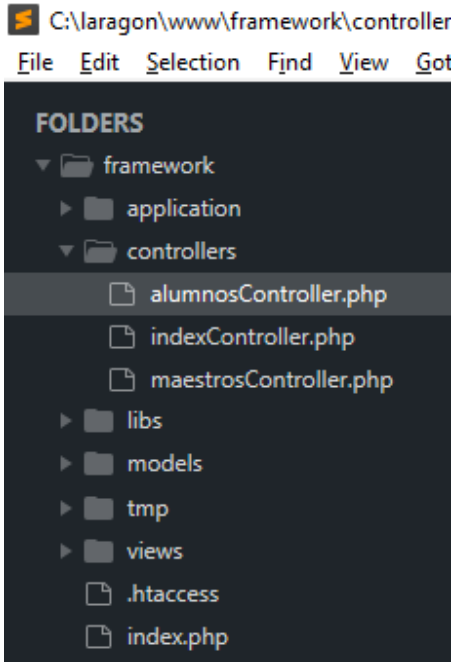

Ilustración 30: Carpeta controllers. Fuente: Elaboración propia.

Dentro del archivo PHP, se debe escribir el nombre del controlador seguido de la para ***Controller***. Tal como se muestra en el siguiente ejemplo del controlador denominado ***alumnosController***. Es importante tener en cuenta que todos los controladores deberán extender de la clase padre ***Controller***.

```php
<?php

class alumnosController extends Controller{

    public function index(){
    }

    public function listar(){
        echo "alumnos";
    }
}

?>
```

Ilustración 31: Ejemplo creación de un controlador. Fuente: Elaboración propia.

Se puede apreciar que dentro del controlador se encuentra declarada la función ***index***, esto es porque en el archivo ***Controller.php***, que se encuentra en la carpeta ***application***, hay una clase ***Controller*** en la que se encuentra declarada una función abstracta ***index***, por lo que cuando creamos nuevos controladores y, debido a que extienden de la clase padre ***Controller***, se tiene que declarar esta función ***index***. A continuación, se muestra el contenido del archivo PHP ***Controller***.

```php
<?php

abstract class Controller
{
    private $_registry;
    protected $_view;
    protected $_acl;
    protected $_request;

    public function __construct()
    {
        $this->_registry = Registry::getInstancia();
        $this->_acl = $this->_registry->_acl;
        $this->_request = $this->_registry->_request;
        $this->_view = new View($this->_request, $this->_acl);
    }

    abstract public function index();
}

?>
```

Ilustración 32: Contenido archivo Controller.php. Fuente: Elaboración propia.

3.5.2 Condicionamiento del entorno

3.5.2.1 Instalación del Framework

Licencia:

Este framework está licenciado bajo la Licencia MIT. Esto significa que es libre para modificar, distribuir y republicar el código fuente con la condición de que las notas de copyright queden intactas. También es libre para incorporar este framework en cualquier aplicación comercial o de código cerrado.

Requisitos del servidor:

Para utilizar este Framework se deben cumplir con los siguientes requisitos:

- Servidor web HTTP. Por ejemplo: Apache.
- PHP 5.2 hasta 7.2

Hasta la fecha, el Framework se ha implementado en dos sistemas de bases de datos:

- MySQL (de la versión 5.2 a la 5.4)
- ORACLE (de la versión 9 a la 11)

Instalación del Framework de forma manual:

Para el uso de este framework, es necesario tener los archivos correspondientes y la base de datos que se conecta a está, por lo cual los pasos a seguir son los siguientes:

1. Descargar los archivos RAR desde la dirección oficial de descarga.
2. Extraer los archivos que contiene el archivo RAR.
3. Descargar la Base de datos desde la dirección oficial de descarga.
4. Copiar el contenido de la carpeta extraída y copiarlo en la carpeta del proyecto (ya sea en un servidor Web, servidor apache o un servidor local).

Después de instalar el framework en el directorio elegido, la estructura de los archivos debe ser la como se muestra en la ilustración 33 (archivos del framework).

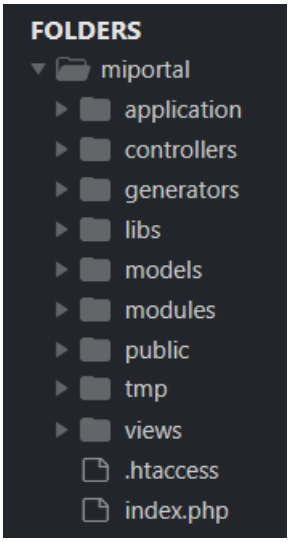

Ilustración 33: Carpetas del Framework. Fuente: Elaboración propia.

3.5.2.2 Configuración del framework

Para que la aplicación funcione correctamente se tienen que realizar algunas configuraciones al archivo ***config*** del framework. A continuación, se explicará cómo realizar dichas configuraciones.

Abrir el archivo ***Config.php*** que se localiza dentro la carpeta ***application*** como se muestra en la ilustración 33.

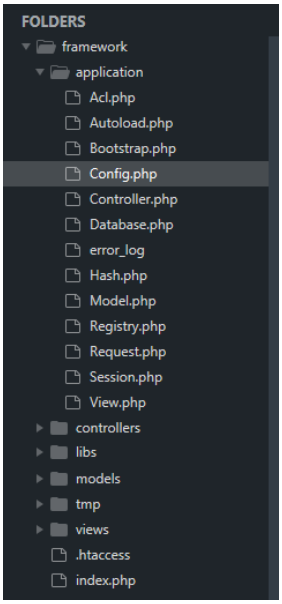

Ilustración 34: Carpeta Config.php del Framework. Fuente: Elaboración propia.

El contenido del archivo es el siguiente:

```php
<?php

//Acceso a Base de Datos
define('DB_HOST', 'localhost');
define('DB_USER', 'root');
define('DB_PASS', '');
define('DB_NAME', 'uqroo');
define('DB_CHAR', 'utf8');


//PATH'S
 //URL base de la aplicación
define('BASE_URL', 'http://localhost/framework/');


//Otras opciones
//En caso de que no se escriba controlador en la url
//se manda a llamar al controlador indexController.php
define('DEFAULT_CONTROLLER', 'index');
define('DEFAULT_LAYOUT', 'default'); //twb


define('SESSION_TIME', 120); define('HASH_KEY', '4f6a6d832be79'); ?>
```

Ilustración 4: Contenido del archive Config.php. Fuente: Elaboración propia.

Las configuraciones que se deben cambiar son las siguiente:

- El acceso a la base de datos (nombre del host, nombre, usuario, contraseña y codificación:

```php
<?php

//Acceso a Base de Datos
define('DB_HOST', 'localhost');
define('DB_USER', 'root');
define('DB_PASS', '');
define('DB_NAME', 'uqroo');
define('DB_CHAR', 'utf8');
```

Ilustración 35: Modificación de acceso a la base de datos. Fuente: Elaboración propia.

- La URL para el acceso a la aplicación:

```php
//PATH'S
 //URL base del administrador
define('BASE_URL', 'http://localhost/framework/');
```

Ilustración 36: URL de acceso. Fuente: Elaboración propia.

Configuraciones opcionales para cambiar:

- El controlador por defecto que se manda a llamar al momento de acceder a la aplicación.

```
//Otras opciones
//En caso de que no se escriba controlador en la url
//se manda a llamar al controlador indexController.php
define('DEFAULT_CONTROLLER', 'index');
```

Ilustración 37: Controlador de acceso. Fuente: Elaboración propia.

- El layout de la aplicación, en caso de que se quiera cambiar el estilo del Framework.

```
define('DEFAULT_LAYOUT', 'default');
```

Ilustración 38: Layout. Fuente: Elaboración propia.

- El tiempo de la sesión.

```
define('SESSION_TIME', 120);
```

Ilustración 39: Tiempo de session. Fuente: Elaboración propia.

- El cifrado de la contraseña del usuario al ingresar a la aplicación.

```
define('HASH_KEY', '4f6a6d832be79');
```

Ilustración 40: Cifrado de contraseña. Fuente: Elaboración propia.

Después de concluir estás configuraciones que son necesarias para el uso del Framework, este puede ser utilizado a hora sin problema alguno.

3.5.2.3 Uso de Oracle para el Framework

Un sistema con la base de datos de Oracle es un sistema de gestión de base de datos tipo Objeto-Relacional. Este tipo de sistema mejora la gestión de grandes bases de datos y también aumenta el nivel de seguridad.

Para el uso de Oracle en este framework, se presenta a continuación los archivos necesarios para la instalación y configuración para su uso en un sistema operativo Windows:

Archivos necesarios

Tabla 5: Archivos para el uso de una base de datos Oracle

Archivos descargados e instalados	Descripción
Oracle SQLDeveloper[1]	Descarga e instalación de Oracle SQLDeveloper para el entorno de Windows.
Oracle installclient[2]	La conexión de la Base de datos usada para la prueba usa una versión de 11_2. Este proporciona los archivos necesarios del lado del cliente de Oracle Database para crear y ejecutar aplicaciones OCI.
oci8	Archivo de funciones de reenvío con las que se vinculan las aplicaciones.

Fuente: Oracle (2022)

La conexión de la Base de datos (instantclient) usada para la prueba usa una versión de 11_2, lo cual se tuvo adaptar y descargar la versión correspondiente para su uso. Este archivo se descargó en la página oficial del fabricante y es necesario para crear y ejecutar los archivos OCI.

Variables de entorno

Las variables de entorno almacenan valores que servirán para que ciertos programas las utilicen o bien para configurar ciertos parámetros del entorno de trabajo.

Las variables de entorno tienen un valor asignado por el sistema operativo en el inicio de la sesión hasta el cierre. Esta configuración es necesaria para la conexión de la Base de datos y su uso en el Framework.

La modificación en la variable de entorno en Windows se realiza a través de la variable de usuario PATH, la cual cuenta con rutas en las que el intérprete busca las órdenes a ejecutar cuando se especifican donde se encuentran. Una vez dentro de la variable PATH, se

[1] Puede consultarse en https://www.oracle.com/database/sqldeveloper/

[2] Puede consultarse en https://www.oracle.com/database/technologies/instant-client/winx64-64-downloads.html

procede a agregar la variable de entorno instantclient. En la ilustración 42 se puede visualizar el resultado de esta:

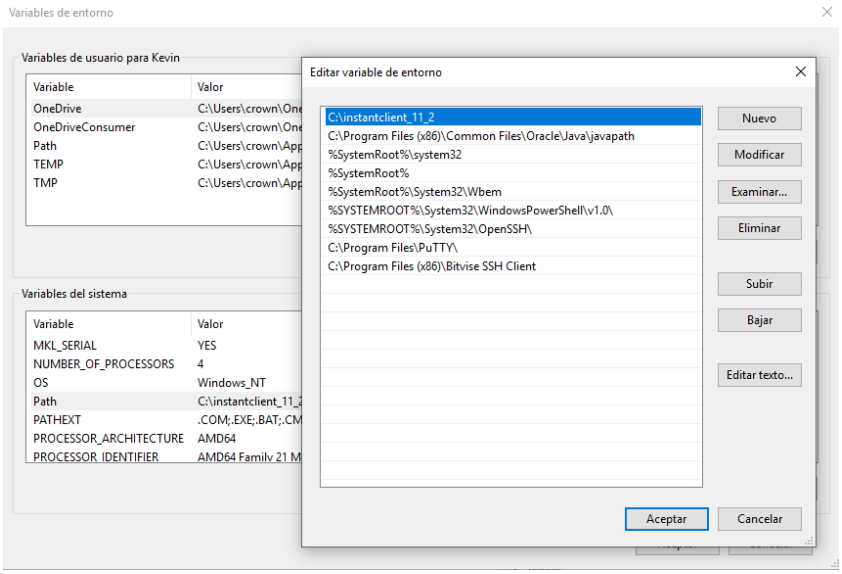

Ilustración 41: Modificación de las variables de entorno. Fuente: Elaboración propia.

Configuración de un Servidor Local para el uso de Oracle

El archivo oci8 es necesario para el uso de la conexión de una base de datos de Oracle. La función de conexión estándar es *oci_connect(),* el cual, crea una conexión a una base de datos de Oracle y devuelve un recurso utilizado por las llamadas subsiguientes a dicha base de datos. (Corporativo, 2022)

Para el uso de *oci_connect*, es necesario agregar los archivos *dll* con una versión compatible. En este caso, se hizo uso del servidor local XAMPP. El archivo necesario para agregar dentro de la configuración del Servidor Local es el siguiente:

php_oci8_11g.dll → C:\xampp\php\ext

Instalación y configuración de ORACLE SQLDEVELOPER

Los archivos necesarios para la instalación pueden ser descargados desde la página oficial del fabricante de Oracle. Una vez realizada la instalación del ORACLE SQLDEVELOPER, es necesario iniciar el programa para el manejo de las bases de datos de Oracle. Una vez dentro del entorno de Oracle, en la barra de opciones o tareas, se debe seleccionar Herramientas, para posteriormente elegir la opción de "Bases de Datos" y seleccionar los "Controladores JDBC de Terceros". Finalmente, se agrega la entrada. ***.C:\instantclient_11_2\ojdbc6.jar*** de manera que se obtiene el resultado de la ilustración 43:

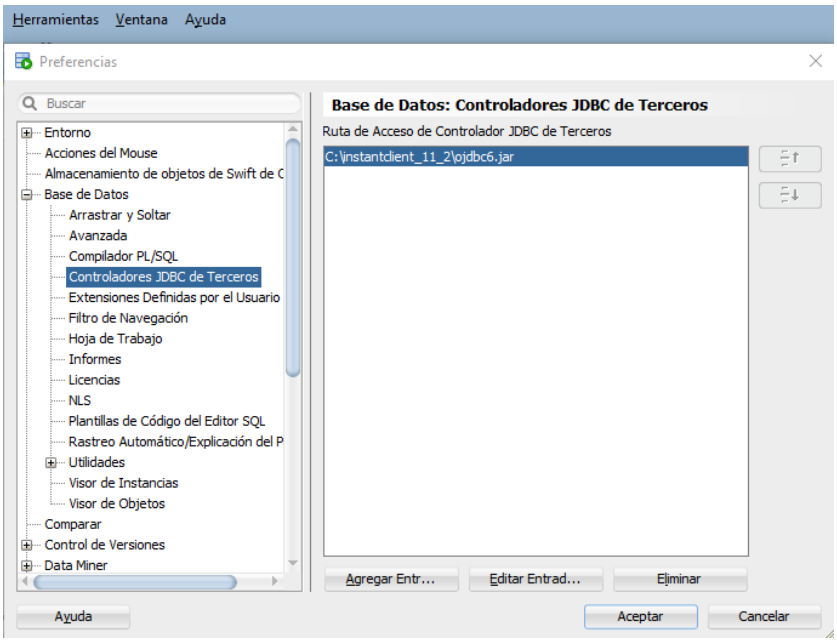

Ilustración 42: Configuración ORACLE SQLDEVELOPER. Fuente: Elaboración propia.

Capítulo 4 Conclusiones

Actualmente, es común observar el uso de sistemas informáticos para la administración de las diversas áreas que se encuentran en la Universidad Autónoma del Estado de Quintana Roo, ya que facilitan los procesos que se llevan a cabo para su operación.

En esta tesis se abordó la creación del framework artesanal "Bitly" que utiliza el patrón de diseño de software MVC (Modelo-Vista-Controlador); en el cual se tiene control total de todas las partes de la aplicación, permitiendo así modificar las funcionalidades, sin estar implementada por terceros. El mercado actual de frameworks presenta algunas limitantes, puesto que la mayoría de estos se vuelven obsoletos con el paso del tiempo ya sea porque el proceso de aprendizaje resulta tardado, o por la tardía incorporación de nuevas tecnologías para el desarrollo de software. De igual manera, algunos de los frameworks son inflexibles, ya que no permiten realizar nuevos desarrollos de sistemas, la incorporación de algunas librerías y las tecnologías incorporadas no se adaptan con facilidad a otros entornos. Por lo tanto, se decidió desarrollar e implementar herramientas en este framework, con el fin de brindar flexibilidad y que está pueda adaptarse a las necesidades que surjan en un futuro, actualizando el frameworks según las nuevas necesidades de los sistemas.

La creación de catálogos en los sistemas, denominados CRUD's (por sus siglas en inglés: Create, Read, Update, Delete) es una parte fundamental para su interacción en las bases de datos, ya que permite la programación rápida de sistema. Por este motivo, la creación del módulo "Generator" de este Framework pretende cubrir esa necesidad, lo que su mejora continua permitirá optimizar esfuerzo y tiempo en el desarrollo. De las mejoras propuestas implementadas en este módulo se encuentran: *el importador del generator, exportador del generator, instalador y ConsultorSQL*, así como la documentación las herramientas que ya incluía. Las herramientas implementadas en este framework a lo largo de este proyecto pueden ser utilizadas para la creación de diversos sistemas informáticos. La mejora podrá garantizar el desarrollo ágil y óptimo de los sistemas que lleguen a ser necesarios en la Universidad Autónoma del Estado de Quintana Roo.

La seguridad del framework es un tema muy importante, ya que se debe garantizar la protección del sistema y de los datos. En este framework se logró introducir una serie de características que ayudan a prevenir vulnerabilidades y ataques cibernéticos, como la implementación de un control contra las inyecciones SQL, un control en el manejo de errores, así como la implementación de una pantalla de inicio de sesión que incorpore un CAPTCHA. De esta manera es como se evitarán ataques automatizados, lo que permitirá la protección contra el acceso no autorizado, así como la mejora de la experiencia de usuario al filtrar registros falsos, spam y actividades indeseables.

Tomando en consideración la metodología ágil para desarrollo de sistemas informáticos (Scrum), se logró el éxito en cada una de las etapas del desarrollo del framework, siendo estas las fases de planificación, diseño, codificación y pruebas. Siguiendo las iteraciones propuestas en la metodología Scrum, fue posible entregar un framework funcional a los interesados en el proyecto, al finalizar cada ciclo de desarrollo. Todo lo anterior permitió el cumplimiento de cada uno de los objetivos establecidos y, por ende, la conclusión satisfactoria de todos los módulos planificados.

Entre las ventajas que ofrece el uso de este framework artesanal de código reducido y documentado, le permite al usuario incorporar las herramientas que considere pertinentes para asegurar el correcto funcionamiento de este, adaptándolo a las necesidades que se tengan en ese momento. Como una propuesta para la mejora futura del Framework, es la posible incorporación de una biblioteca de pruebas unitarias tales como JUnit, NUnit o PHPUnit. Esta herramienta permitirá la realización de pruebas automatizadas con el fin de asegurar la calidad, así como el correcto funcionamiento del código.

De acuerdo con las pruebas unitarias realizadas, para evaluar el funcionamiento básico de los componentes del framework, se observaron resultaron satisfactorios. No obstante, tanto en las pruebas de sistema como en la de interfaces, se detectó que, en algunos casos, los resultados no fueron satisfactorios. Sin embargo, esto no quiere decir que el funcionamiento del framework no sea el óptimo, sino más bien son áreas de oportunidad futuras en las cuales enfocar las mejoras.

La ejecución de las pruebas se realizó a partir de la instalación del archivo del framework la cual se desarrolló exitosamente, ya sea a través de una base de datos MySQL u Oracle, logrando ejecutarse correctamente y obteniendo la pantalla de inicio de sesión para ambos casos. Así también, se puso a prueba el uso de la herramienta Generator para la creación de un catálogo. El resultado fue satisfactorio puesto que se logró mostrar el catálogo con la información correcta, así como el respectivo mensaje de error en caso de que no existan las tablas necesarias en la base de datos.

En cuanto a la administración de roles de usuario, se logró el registro exitoso de nuevos usuarios, así como la edición y cambio de roles a los usuarios que ya se encontraran registrados en el sistema. De igual forma se comprobó que el usuario que no cuente con los permisos adecuados no podrá realizar modificaciones en los roles de usuario.

En el caso de la comprobación de la seguridad del Framework, se evaluó que el usuario que no cuente con el rol adecuado no tendrá permitido hacer modificaciones a las cuentas de los usuarios. Asimismo, se determinó que al visualizar las contraseñas éstas se encontraran encriptadas con funciones Hash. Otro resultado exitoso se obtuvo mediante el uso de las consultas parametrizadas, lo que permitió evitar la inyección SQL en el formulario de inicio de sesión.

Por otra parte, las pruebas de conexión del framework con las bases de datos MySQL y Oracle resultaron satisfactorias, ya que, para ambos casos, en los cuales se utilizaron más de 10 mil datos, no se detectaron problemas relacionados con la conexión, consulta, inserción, actualización, eliminación, rendimiento, seguridad y compatibilidad. Un complemento a las pruebas realizadas fue el patrón de diseño "Singleton", el cual se implementó en esté framework, que permitió limitar el número de estancias posibles de una clase dentro del sistema, así como proporcionar un acceso global al mismo.

Como conclusión general, Bitly ha demostrado un desempeño exitoso para cada una de las pruebas realizadas. Sin embargo, es posible realizar modificaciones a lo largo del tiempo, las cuales puedan ayudar a mejorar el funcionamiento de éste. Por lo tanto, realizando una comparación con un framework tradicional como Laravel, con el tiempo se recomienda

contemplar nuevas herramientas al framework Bitly, por ejemplo, la implementación de ORM (por sus siglas en inglés: Object Relational Mapping). Este es un modelo de programación que permite mapear las estructuras de bases de datos (MySQL, Oracle, SQL Server, etc.), la cual utiliza una sintaxis clara y legible para realizar consultas y manipular datos en la base de datos, haciendo que el código sea más fácil de entender y mantener. Además, facilita el manejo de relaciones entre tablas, proporciona un sistema de migraciones que permite controlar y permite modificar la base de datos de manera controlada.

De igual manera, otra mejora a implementar es el manejo de errores. Actualmente en el framework los errores no son visibles claramente, y el único control de estos son los mensajes de errores almacenados en el archivo "error_log" del sistema. Por lo consiguiente, se recomienda crear un controlador de errores, como errores 404 (página no encontrada), y un registro de errores que puedan ser almacenados directamente a la base de datos.

Otra herramienta para implementar es el "renombrar rutas o crear alias para las mismas", esto para mejorar la legibilidad, flexibilidad y escalabilidad del código, ayudando a la optimización del motor de búsqueda y enriqueciendo la experiencia de usuario. Asimismo, como otra propuesta, se podría adaptar la compatibilidad con el uso de otras bases de datos como SQLServer ya que, de esta manera, los desarrolladores y usuarios podrán elegir entre más variedad, la base de datos que más se adapte a sus necesidades y preferencias.

Con el transcurso del tiempo, se recomienda llevar a cabo un análisis del proyecto, con el fin de identificar y determinar las posibles áreas de mejora, de acuerdo con las necesidades que vayan surgiendo. Todo lo anterior permitirá realizar una serie de actualizaciones, tales como la incorporación e implementación de nuevas herramientas y funciones, que permitirán que el desarrollo de sistemas se realice de forma más rápida y eficiente.

Referencias

Acens Technologies. (Marzo de 2014). *Framework para el desarrollo ágil de aplicaciones.* Recuperado el 25 de Noviembre de 2019, de acens: https://www.acens.com/wp-content/images/2014/03/frameworks-white-paper-acens-.pdf

Alicante, U. d. (2006). *Repositorio Institucional de la Universidad de Ricalde.* Obtenido de https://rua.ua.es/dspace/bitstream/10045/4042/1/tema11.pdf

Camazón, J. N. (2011). *Aplicaciones web.* Editex.

Cobo, Á., Gómez, P., Pérez, D., & Rocha, R. (2005). *PHP y MySQL: Tecnología para el desarrollo de aplicaciones web.* Ediciones Díaz de Santos.

Corporativo. (2022). *Manejo de conexiones de OCI8 y agrupamiento de conexiones*. Obtenido de PHP: https://www.php.net/manual/es/oci8.connection.php

Delía, L. (2019). *Framework para el Desarrollo Ágil de Sistemas Web.* Obtenido de Universidad Nacional de La Plata: http://sedici.unlp.edu.ar/bitstream/handle/10915/4000/Tesis.%20Framework%20para%20el%20desarrollo%20%C3%A1gil%20de%20sistemas%20web.pdf-PDFA1b.pdf?sequence=2&isAllowed=y

Dimes, T. (2015). *JavaScript Una Guía de Aprendizaje para el Lenguaje de Programación JavaScript.* Babelcube Inc.

FPDF. (2019). *FPDF Library PDF generator*. Obtenido de FPDF: http://www.fpdf.org/

Hundermark, P. (2011). *Un Mejor Scrum. Un conjunto no oficial de consejos e ideas sobre cómo implementar Scrum.* Ciudad del Cabo: scrumsense.

Lafosse, J. (2010). *Struts 2: El framework de desarrollo de aplicaciones Java EE.* Barcelona: Ediciones ENI.

Lainez, J. C. (2016). *Implementación de un Sistema de AdministraciónWeb para la Indexación de la Revista Ciencias Pedagógicas e Innovación de la UPSE.* Obtenido de Universidad Estatal Península de Santa Elena: chrome-extension://oemmndcbldboiebfnladdacbdfmadadm/https://repositorio.upse.edu.ec/bitstream/46000/2501/1/UPSE-TIN-2016-0015.pdf

Lancker, L. V. (2014). *JQuery: el framework JavaScript de la Web 2.0.* Ediciones ENI.

Lemus, G. V., Ortíz, H. J., & Segura, J. L. (Abril de 2019). *APLICACIÓN WEB PARA ADMINISTRACIÓN DE EXÁMENES DIFERIDOS, REPORTES DEACTIVIDADES Y CORRECCIONES DE NOTAS.* Obtenido de Universidad Tecnológica de El Salvador: http://biblioteca.utec.edu.sv/siab/virtual/tesis/941001169.pdf

Mariño, S. I., & Alfonzo, P. L. (2014). *Implementación de SCRUM en el diseño del proyecto del Trabajo Final de Aplicación .* Obtenido de Redalyc: https://www.redalyc.org/pdf/849/84933912009.pdf

Mariscal, A. B. (2015). *UF2405 - Modelo de programación web y bases de datos.* Elearning S.L.

Markiewicz, M. E., & Lucena, C. J. (Octubre de 2000). *Understanding Object-Oriented Framework Engineering.* Obtenido de Semantic Scholar: https://pdfs.semanticscholar.org/0cad/f48a557024a53b06c5ed73d6ef965c80861f.pdf

Muñoz, V. J. (2013). *El nuevo PHP. Conceptos avanzados.* Bubok Publishing S. L.

Navarro Cadavid, A., Fernández Martínez, J. D., & Morales Vélez, J. (2013). *Revisión de metodologías ágiles para el desarrollo de software.* Obtenido de Redalyc: https://www.redalyc.org/pdf/4962/496250736004.pdf

Oracle. (2022). *Oracle*. Obtenido de SQL Developer: https://www.oracle.com/database/sqldeveloper/

Pérez, J. C. (2015). *PROGRAMACIÓN ORIENTADA A OBJETOS.* ESPAÑA: RA-MA.

PhpSpreadsheet. (2019). *PhpSpreadsheet documentation*. Obtenido de PhpSpreadsheet: https://phpspreadsheet.readthedocs.io/en/latest/

Ricardo J. Vargas Del Valle, J. P. (2006). *Di Mare.* Obtenido de http://www.di-mare.com/adolfo/cursos/2007-2/pp-3capas.pdf

Romero, Y. F., & González, Y. D. (Abril de 2012). *Patrón Modelo-Vista-Controlador.* Obtenido de Revista Telemática: http://revistatelematica.cujae.edu.cu/index.php/tele/article/download/15/10/0

Sanz, M. L., Fúquene, D. M., Pérez, Á. M., Velasco, M. P., Fuentes, J. U., & Mesa, J. M. (2015). *Programación web en el Entorno Cliente. (MF0491_3).* Madrid: Grupo Editorial RA-MA.

Sommerville, I. (2005). *Ingeniería del software.* Madrid: Pearson Education.

Uceda, O. C. (2013). *Desarrollo Web con PHP: Aprende PHP paso a paso.*

Vázquez, F. A. (2010). *Portal de Revistas Academicas.* Obtenido de http://revistas.urp.edu.pe/index.php/Paradigmas/article/view/1498/1384

Villalobos, G. M., Sánchez, G. D., & Gutiérrez, D. A. (Colombia de Abril de 2010). *DISEÑO DE FRAMEWORK WEB PARA EL DESARROLLO DINÁMICO DE APLICACIONES.* Obtenido de Redalyc: https://www.redalyc.org/pdf/849/84917316032.pdf

Anexos

Anexo 1. Ejecución de las pruebas

Tabla 6: Ejecución de la prueba archivo de instalación del Framework

Archivo de instalación del Framework	
Objetivo	**Requisitos**
Verificar que el archivo de instalación del Framework se ejecute sin problema y lograr cargar la pantalla de inicio de sesión.	- Capacidad de respuesta de ejecución. - Tiempo de carga aceptables. - Almacenamiento suficiente.
Proceso de ejecución	
Iniciar el archivo "instalador.php".	Verificar que los archivos estén completos dentro de la carpeta de instalación del Framework para que este pueda ser ejecutado de manera correcta.
Rellenar el formulario de instalación.	Comprobar que se pueda introducir información en el llenado de los formularios de instalación.
Instalación seleccionando y usando una base de datos MySQL.	Verificar que se pueda seleccionar usar una base de datos con MySQL en el formulario de instalación.
Instalación seleccionando y usando una base de datos Oracle.	Verificar que se pueda seleccionar usar una base de datos con Oracle en el formulario de instalación.
Verificar la conectividad de la base de datos. (MySQL u Oracle).	Comprobar que la conexión entre la base de datos y el Framework funcionan de manera correcta al ejecutar una consulta.
Iniciar sesión	Comprobar el inicio de sesión después de realizar la instalación del Framework, usando el usuario y contraseña ingresados anteriormente en el formulario de instalación.
Resultados obtenidos	
Resultado 1: Instalación del Framework usando una base de datos MySQL.	Se realizó la instalación de manera exitosa logrando administrar y verificar el funcionamiento de la base de datos con MySQL y así mismo, logrando iniciar la pantalla de inicio de sesión del Framework.
Resultado 2: Instalación del Framework usando una base de datos Oracle.	Se realizó la instalación de manera exitosa logrando administrar y verificar el funcionamiento de la base de datos con MySQL y así mismo, logrando iniciar la pantalla de inicio de sesión del Framework.
Estado de la prueba	Exitoso

Fuente: Elaboración propia

Tabla 7: Ejecución de la prueba uso de la herramienta Generator

Uso de la herramienta Generator	
Objetivo	**Requisitos**
Crear un catálogo mediante la herramienta Generator del Framework	- Ingresar el Framework de manera correcta. - Tiempo de carga aceptables.
Proceso de ejecución	
Ingresar al sistema y seleccionar el nombre del módulo.	Ingresar al Framework y dirigirse a la sección de paginación, en el apartado de "crear catálogo". Se desplegará un formulario para ingresar el nombre del catálogo a crear; al final del proceso el archivo creado se renombrará *"ejemploGenerator:"*
Tablas para el catálogo.	Verificar que se muestre la lista de tablas que intervienen para realizar el catálogo.
Listar el contenido de la tabla.	Comprobar que las tablas enlistadas puedan mostrar la información correspondiente en el catálogo.
Personalizar las columnas.	Verificar que en las columnas del catálogo que fueron seleccionadas se pueda modificar el estilo y/o formato de los datos que se van a mostrar.
Menú de opciones.	Comprobar que, en la lista de opciones, que se despliegan al seleccionar el botón "Ver", se puedan visualizar de manera correcta las funciones de "Editar", "Detalles", "Duplicar", "Eliminar", y que estas se ejecuten de manera correcta.
Visualización del catálogo.	Guardar los cambios de creación y edición de un nuevo catálogo y verificar que muestre los datos de las tablas de manera correcta.
Resultados obtenidos	
Resultado 1: Las tablas seleccionadas existen en la base de datos.	Se muestra el catálogo creado de manera correcta y los datos coinciden con las tablas.
Resultado 2: Las tablas seleccionadas no existen en la base de datos.	Se visualiza un mensaje de error ya que, al no existir las tablas correspondientes falla la creación del catálogo.
Estado de la prueba	Exitoso

Fuente: Elaboración propia

Tabla 8: Ejecución de la prueba administración de roles de usuarios

Administración de roles de usuarios	
Objetivo	**Requisitos**
Prueba para registrar nuevos usuarios y editar, cambiar los roles a los usuarios existentes en el sistema.	- Ingresar el Framework de manera correcta. - Tiempo de carga aceptables.
Proceso de ejecución	
Creación de roles	Verificar que se pueda crear un nuevo rol de usuario de manera correcta.
Modificación de roles	Verificar que se pueda modificar un rol existente, como actualizar el nombre del rol o los permisos asociados a este.
Eliminación de roles	Verificar que se pueda eliminar un rol de manera adecuada y comprobar que el rol y todas sus asociaciones, como los usuarios que lo tengan asignado, sean eliminados correctamente.
Asignación de roles a usuarios	Validar que se pueda asignar un rol existente a un usuario y que éste tenga los permisos adecuados después de asignarle un rol.
Revocación de roles a usuarios	Comprobar que se pueda revocar un rol a un usuario y verificar que los permisos y accesos sean retirados de manera correcta.
Consulta de roles y usuarios asociados	Verificar que se pueda obtener información sobre los roles que están creados en el sistema y sobre los usuarios que tienen asignado cada rol.
Restricciones y validaciones	Verificar que se puedan aplicar las restricciones y validaciones necesarias, al crear, modificar o eliminar roles de usuarios.
Prueba de seguridad	Verificar que los usuarios con los permisos adecuados puedan acceder y realizar acciones relacionadas con la administración de roles de usuarios.
Resultados obtenidos	
Resultado 1: El usuario puede realizar la administración de roles de usuarios.	Se logró administrar los roles de usuario, así como asignar, eliminar y editar los roles existentes, y crear nuevos.
Resultado 2: El usuario no puede realizar la administración de roles de usuarios.	No logró administrar los roles de usuarios, ya que no tenía los permisos adecuados y asignados para realizar modificaciones.
Estado de la prueba	Exitoso

Fuente: Elaboración propia

Tabla 9: Ejecución de la prueba de seguridad el Framework

Seguridad del Framework	
Objetivo	**Requisitos**
Pruebas que permitirán evaluar la seguridad del Framework	- Ingresar el Framework de manera correcta. - Tiempo de carga aceptables.
Proceso de ejecución	
Gestión de inicio de sesión	Comprobar que el inicio de sesión funcione correctamente, así como el cierre de sesión.
Autorización de acceso	Verificar los controles de acceso y permisos para usuarios y roles, asegurando que los usuarios tengan acceso a las funciones autorizadas.
Intento de inyección SQL	Evaluar la resistencia del sistema ante intentos de inyección de código, como SQL inyection o ataques de script.
Manipulación de sesiones	Verificar que el Framework proteja las sesiones de usuario y evite la manipulación de estas.
Gestión de contraseñas	Evaluar la seguridad de las contraseñas almacenadas en el Framework y verificar que se implementan políticas de contraseñas seguras y que éstas se almacenan de forma encriptada.
Patrón Singleton	Evaluar el control de instancias y verificar que el método ***"getInstancia"*** funciona y solo una instancia tenga acceso global al sistema.
Resultados obtenidos	
Resultado 1: Usuario con un rol editor intentar realizar funciones de administrador.	Al intentar realizar una modificación a las cuentas de los usuarios, el sistema no permitió realizar cambio alguno.
Resultado 2: Contraseñas encriptadas de los usuarios.	Al realizar la prueba para visualizar contraseñas de los usuarios, éstas están encriptadas con funciones Hash.
Resultado 3: Inyección SQL en el formulario de inicio de sesión.	El resultado fue satisfactorio, evitando la introducción de la sentencia SQL utilizando consultas parametrizadas.
Resultado 4: Intento de varias conexiones a la base de datos.	El resultado fue satisfactorio, logrado limitar el número de estancias posibles de una clase y proporcionar un acceso global al mismo.
Estado de la prueba	Exitoso

Fuente: Elaboración propia

Tabla 10: Ejecución de la prueba conexión del Framework con las bases de datos (MySQL u Oracle)

Conexión del Framework con las bases de datos (MySQL u Oracle)	
Objetivo	**Requisitos**
Realizar pruebas de conexión de las bases de datos MySQL y Oracle, usando tablas de más de 10 mil datos.	- Tiempo de carga aceptables. - Conexión estable con las bases de datos. - Bases de datos con tablas de 10 mil datos.
Proceso de ejecución	
Conexión	Comprobar que el sistema puede establecer conexiones exitosas con las bases de datos.
Consulta básica	Realizar una consulta simple para obtener resultados y verificar que se pueda ejecutar.
Inserción	Insertar datos en la base de datos y comprobar que se pueda agregar correctamente la información.
Actualización	Actualizar datos de las bases de datos y comprobar que el sistema puede modificarlos de manera correcta.
Eliminación	Eliminar datos de las bases de datos y verificar que la operación sea exitosa.
Rendimiento	Evaluar las operaciones de las bases de datos, medir los tiempos de respuesta y la capacidad de procesamiento.
Seguridad	Comprobar la autenticación y el cifrado; evaluar la seguridad y protección de la conexión de las bases de datos.
Compatibilidad	Evaluar que las consultas y comandos SQL se puedan ejecutar de manera correcta y sean compatibles.
Resultados obtenidos	
Resultado 1: Pruebas realizadas usando una base de datos MySQL.	Se logró realizar todas las pruebas usando una base de datos MySQL de más de 10 mil datos.
Resultado 2: Pruebas realizadas usando una base de datos Oracle.	Se logró realizar todas las pruebas usando una base de datos Oracle de más de 10 mil datos.
Estado de la prueba	Exitosa

Fuente: Elaboración propia