UNIVERSIDAD CENTRAL DE VENEZUELA

FACULTAD DE CIENCIAS

ESCUELA DE COMPUTACIÓN

Desarrollo de una aplicación web para el proceso de inscripción de los estudiantes de la Escuela de Estudios Políticos y Administrativos de la Universidad Central de Venezuela.

Autores:

Efraín U. Díaz P.

V-24.888.992

Keiber A. Prin C.

V-24.933.657

Tutor:
Profa. Vanessa Leguizamo
Cotutor:
Prof. Rafael Romero

Caracas, junio de 2019.

UNIVERSIDAD CENTRAL DE VENEZUELA
FACULTAD DE CIENCIAS
ESCUELA DE COMPUTACIÓN

ACTA

Quienes suscriben, miembros del jurado designado por el Consejo de la Escuela de Computación, para examinar el Trabajo Especial de Grado titulado "**Desarrollo de una aplicación web para el proceso de inscripción de los estudiantes de la Escuela de Estudios Políticos y Administrativos de la Universidad Central de Venezuela**" y presentado por los Bachilleres **Efraín U. Díaz P. (C.I. V-24.888.992)** y **Keiber A. Prin C. (C.I. V-24.933.657)**, a los fines de optar al título de **Licenciado en Computación**, dejamos constancia de lo siguiente:

Leído como fue dicho trabajo, por cada uno de los miembros del jurado, se fijó el día 28 de junio de 2019, a las 11:00 am horas, para que los autores lo defendieran en forma pública, lo que estos hicieron en la Sala 1 de la Escuela de Computación, mediante una presentación oral de su contenido, luego de lo cual respondieron a las preguntas formuladas. Finalizada la defensa pública del Trabajo Especial de Grado, el jurado decidió aprobar con la nota de 18 puntos.

En fe de lo cual se levanta la presente Acta, en Caracas el día 28 de junio de 2019.

Profa. Vanessa Leguizamo
(Tutora)

Profa. Maria Herrero
(Jurado)

Profa. Keyla Rivas
(Jurado)

Resumen

El trabajo propone el desarrollo de una aplicación web a través de la cual se manejan diversos aspectos relacionados al proceso de inscripción de los estudiantes de la Escuela de Estudios Políticos y Administrativos, además facilitar y gestionar de manera conveniente este fundamental proceso. Dado que el sistema fue diseñado para amoldarse a las necesidades y requerimientos de la Escuela de Estudios Políticos y Administrativos, éste integra las funcionalidades aunadas a estos requerimientos, centralizando y acoplandolas entre sí para brindar una experiencia de usuario fluida. Para el desarrollo de la aplicación se utilizó el proceso SCRUM, diseñando los sprints y luego ajustándolos a los tiempos requeridos conjuntamente con los stakeholders. Se liberó una versión preliminar que fue utilizada en el proceso de inscripción del semestre II-2018.

Palabras clave: Aplicación web, SCRUM, Sistema de Inscripción, UCV.

Tabla de Contenidos

Lista de figuras

Introducción

En la Universidad Central de Venezuela para dar inicio a un semestre en sus escuelas es necesario que un alumno realice el proceso de inscripción de materias a cursar por semestre, esto hace que sea posible cumplir con las normas establecidas de esta institución y poder evaluar en el transcurso de tiempo que dura un semestre las aptitudes académicas de un estudiante en las asignaturas escogidas, permitiendo conseguir los requisitos necesarios para avanzar en su carrera.

Es preciso que el personal administrativo de la escuela planee y coordine las diferentes actividades a realizar para poder llevar a cabo las inscripciones en un semestre, con la creación de las materias a ofertar, la asignación de profesores y horarios, etc. Estas materias son ofrecidas durante las inscripciones y los estudiantes las puedan registrar para poder dar inicio a su curso académico.

No todas las escuelas de la Universidad Central de Venezuela cuentan con el material o los recursos necesarios para realizar este proceso de la manera más eficiente posible, por ejemplo en alguna de ellas todo el proceso está disponible para ser realizado digitalmente mientras que en otras la automatización es solo parcial durante las inscripciones; esto último conlleva a diferentes problemas del cual es objeto y se plantea resolver en este trabajo de grado, beneficiándose de las tecnologías actuales disponibles a nivel de programación.

Con el objetivo de alcanzar ese fin, este documento propone el uso de una aplicación web a través de la cual se manejan diversos aspectos relacionados al proceso de inscripción de la Escuela de Estudios Políticos y Administrativos de la UCV, además de facilitar y gestionar de manera conveniente este fundamental proceso.

En el primer capítulo se detalla el problema que se ha resuelto junto con una solución propuesta para corregir los inconvenientes planteados, y un alcance que sirvió como guía de las funcionalidades que tiene la aplicación desarrollada. Luego en el capítulo 2 se definieron los conceptos imprescindibles tomados en cuenta para la realización de esta solución y entendimiento de la misma. En el capítulo 3 se explica el marco metodológico que fue usado durante el desarrollo de esta solución además de una justificación de esa elección. Ya en el capítulo 4 se implementó la solución y se explica paso a paso cómo fue llevada a cabo esta tarea para que finalmente, en el último apartado de este documento se presenten las conclusiones que dan cierre a este trabajo.

Capítulo 1

Problema de investigación

El presente capítulo introduce el problema del cual es objeto este trabajo, además de exponer los objetivos a alcanzar para lograr la resolución del problema, junto con una solución y justificación de la propuesta que fue desarrollada.

Planteamiento del problema

La Universidad Central de Venezuela (UCV), cuenta dentro de su jurisdicción con facultades, dichas facultades son encargadas de estructurar, impartir y difundir una determinada área del conocimiento, así como servir de núcleo de investigación. Dada la extensión y el alcance de sus competencias, cada facultad cuenta con escuelas que se centran en un ámbito más específico de su área de conocimiento.

Cada facultad cuenta con departamentos administrativos que soportan procesos fundamentales para mantener su operatividad, éstos ayudan a cumplir con el propósito y los objetivos de la facultad en la que se encuentran alojados. Así como hay procesos administrativos propios de cada facultad, hay procesos administrativos propios de cada escuela, independientes entre sí. El objetivo de este trabajo está centrado en uno de esos procesos administrativos de una escuela en particular, esta es la Escuela de Estudios Políticos y Administrativos perteneciente a la Facultad de Ciencias Jurídicas y Políticas. El proceso en el que se hizo énfasis es el de inscripción de estudiantes que se realiza cada semestre.

Mediante el proceso de inscripción se permite a los estudiantes inscribir las materias a cursar en el semestre próximo a iniciar, este proceso evalúa prelaciones (o dependencias) que deben ser cumplidas antes de poder cursar ciertas materias, choque de horarios que puedan ocurrir cuando un estudiante inscribe más de una materia con horarios que coincidan, límite de cupos, límite de créditos y asignación de aulas. Por tanto este proceso permite llevar el registro y control de los requerimientos académicos que cada estudiante necesita para inscribir alguna materia y dar constancia de las materias que cada estudiante inscribió.

Particularmente en la Escuela de Estudios Políticos y Administrativos, el proceso de inscripción ha sido voluble y errático a lo largo de los semestres anteriores debido a diversos problemas que se presentaban durante la realización de este proceso.

Estos problemas venían desde el choque de horarios, ya que no está permitido que un estudiante pueda inscribir materias en horarios que coincidan, una política propia de la Escuela en cuestión. Otro problema era el tope de créditos por semestre, ya que con el fin de evitar la sobrecarga académica de un estudiante, se establecieron límites mínimos y máximos de créditos permitidos por semestre, cada materia tiene una carga de créditos asociada y un estudiante no puede inscribir materias cuya suma de créditos exceda el límite máximo o sea inferior al límite mínimo, salvo ciertas ocasiones excepcionales, que se deciden directamente por el Consejo de Escuela. Otro de los inconvenientes fue el tope de estudiantes por aulas, para cada materia se tienen horarios asociados y para cada horario hay aulas asociadas con una capacidad determinada de estudiantes, no se debe permitir la inscripción a aulas cuya capacidad haya sido agotada.

El proceso de inscripción ha tenido cambios a lo largo de los semestres, y a continuación expondremos algunos de los casos más recientes.

Durante el semestre I-2017, se habilitaron aulas para la inscripción de materias, las cuales son agrupadas por el semestre al cual corresponden a ser dictadas durante la carrera. Cada estudiante debía ir a estas aulas a anotarse en las listas (hojas de papel) de cada materia que quería inscribir, y al final del proceso el estudiante se quedaba con un papel que servía de comprobante para ser sellado por un personal autorizado que indicaba la culminación de la inscripción. Al final si una lista se llenaba por encima del cupo disponible se descartaban las inscripciones excedentes.

Durante el semestre II-2017 a diferencia del semestre I-2017 los estudiantes no se inscribían por cuenta propia en cada aula, sino que cada aula solo servía como punto de información para horarios, cupos y profesores; y luego de haber revisado la información de las materias que quería inscribir en su semestre se dirigía a una estación final con personal administrativo encargado de realizar, usando una hoja de Excel, la inscripción del estudiante en las materias que él había indicado.

La inscripción de estudiantes ha sido llevada a cabo con herramientas que no promovían ni cubren completamente con los requisitos deseados por los miembros de la Escuela, haciendo los tiempos de cumplimiento elevados y un uso ineficiente de recursos. Dichas herramientas, se usaban como componentes aislados para cumplir en conjunto el objetivo de inscribir estudiantes, y aun así no proveían de ciertas funcionalidades que por ende debían ser trasladadas al personal administrativo de la Escuela y ser atacadas bajo procesos manuales y poco ágiles, como fue mencionado anteriormente. Los procesos manuales implicaban costos de transporte, así como tiempo al haber una disponibilidad de

atención limitada en la dirección de control de estudios, y a las limitaciones de velocidad que conlleva un proceso manual lo que se traduce en colas de espera.

Además dentro de las funcionalidades que las herramientas usadas sí cumplían, cuyos propósitos no son principalmente orientados a soportar un proceso de inscripción, éstas requerían de un alto grado de intervención manual por parte del personal administrativo para captar la información de inscripción de los estudiantes así como verificar la cantidad de cupos, horarios disponibles y evitar permitir a un estudiante inscribir materias con choque de horarios, entre otras.

Adicionalmente es importante contemplar que, producto de este proceso de inscripción, se tenga una nómina de estudiante con la cual se pueda constatar el estado de los estudiantes inscritos, sus materias inscritas y créditos inscritos, que actualmente son manejadas por un sistema llamado UXXI.

El sistema UXXI (Universitas XXI) se usa para manejar la información de los estudiantes, el cual se encarga de la gestión administrativa y académica de una parte de la comunidad estudiantil de la UCV, incluyendo la Escuela de Estudios Políticos y Administrativos. Esta herramienta posee un récord académico de cada estudiante, además de proporcionar documentos como el Kardex o ficha de grado. UXXI es usado por la Escuela de Estudios Políticos y Administrativos para cargar y mantener la información académica de sus estudiantes y llevar el control de procesos como el de inscripción, mas mediante ese sistema los estudiantes no realizan sus inscripciones como tal. La disponibilidad de este sistema es limitada dado que se encuentra alojada en un servidor inestable que con frecuencia se encuentra caído, esto mencionado por miembros de la

Escuela, lo cual complica su acceso y entorpece los procesos de la Escuela que dependen de dicho sistema.

Estos factores mencionados enmarcaban un proceso irregular en el tiempo y descentralizado, por ende, expuesto significativamente a fallas y errores que luego implicaban un esfuerzo adicional para ser solventados. En función de lo antes descrito, se propuso el siguiente objetivo general para este trabajo.

Objetivo General

Implementar una aplicación web para el proceso de inscripción de los estudiantes de la Escuela de Estudios Políticos y Administrativos de la Universidad Central de Venezuela.

Objetivos Específicos

1. Definir los requerimientos de la aplicación web que permita la inscripción de estudiantes de la Escuela de Estudios Políticos y Administrativos de la Universidad Central de Venezuela.
2. Definir e implementar el modelo de la base de datos que refleje y soporte el proceso de inscripción dentro de la aplicación.
3. Reflejar la estructura y el comportamiento de la aplicación mediante diagramas de modelado.
4. Investigar candidatos a herramientas de software para ser utilizadas en el desarrollo de la aplicación.
5. Desarrollar un módulo para la configuración del proceso de inscripción.

Justificación

Se justificó la solución propuesta en base a que, la existencia de un sistema web para gestionar y realizar este proceso provee de accesibilidad y concurrencia para que múltiples estudiantes gestionen su inscripción durante el periodo estipulado desde cualquier lugar con acceso a internet. Dado que el sistema fue diseñado para amoldarse a las necesidades y requerimientos de la Escuela de Estudios Políticos y Administrativos, éste integrará las funcionalidades aunadas a estos requerimientos, centralizando y acoplando ellas entre sí para brindar una experiencia de usuario fluida.

Inmerso en estos requerimientos, la portabilidad debe ser una cualidad intrínseca del sistema por lo cual el lenguaje y las herramientas de software fueron evaluadas y escogidas tomando eso en cuenta para iniciar un nuevo despliegue del mismo a demanda, según lo necesite la Escuela en caso de querer trasladar el hospedaje de la aplicación.

Al mantener toda la lógica encapsulada en una aplicación, además de implícitamente automatizar el proceso (lo cual es en gran parte lo que se necesita), se tiene un mayor control del flujo de información por lo que la realización de operaciones y validaciones en línea que requieran dar una respuesta al momento, se pueden dar tan pronto la información sea ingresada en el sistema, de lo cual hay escenarios en el proceso que requieren esta clase de prontitud.

Solución propuesta

Se propuso crear una aplicación web, que pueda ser accedida desde cualquier ubicación en tanto se disponga de conexión a internet y un navegador web. Los usuarios de la aplicación van a interactuar con ésta a través de un sitio web que les permite realizar las

diversas operaciones relacionadas con el proceso de inscripción estudiantil. Dichos usuarios comprenden tanto los estudiantes a inscribirse, como al personal de la escuela que usa el sistema para dar soporte al proceso de inscripción.

Se planteó reducir significativamente los tiempos de inscripción así como el esfuerzo producto de la automatización en el proceso y el soporte a la concurrencia de varios estudiantes inscribiéndose durante un mismo intervalo de tiempo, sumado a esto la aplicación posee bajo sus funcionalidad la de realizar las validaciones y manejar las restricciones de la escuela, dejando al personal administrativo de la escuela el manejo de casos excepcionales que se puedan presentar y la realización de cambios que correspondan ser realizados directamente por personal docente o administrativo.

Dentro de esta solución se permitió generar nóminas y reportes de estudiantes inscritos con el fin de proveer a los docentes la lista de estudiantes cursando su materia en un semestre dado.

Esta aplicación fue desarrollada bajo un lenguaje de programación web ampliamente usado para ser alojada en servidores web que se encuentren preparados para gestionar aplicaciones en dicho lenguaje, por lo que el proceso de configuración e instalación es bastante simple.

Alcance

El alcance de este proyecto constituye la elaboración de una aplicación web administrativa para un proceso de inscripción de estudiantes, que permita el registro y carga de estudiantes o materias dentro del sistema, además de inscribir y validar las inscripciones

de los estudiantes de acuerdo a las normativas actuales dentro de la Escuela de Estudios Políticos y Administrativos.

Para cumplir con dicha visión del producto se debe:

- Permitir la autenticación en el sistema y mantener estados de sesión.
- Mantener distinciones entre tipos de usuarios administradores y estudiantes.
- Permitir registrar estudiantes manualmente o por carga de archivo sólo si es un usuario administrador.
- Permitir crear horarios si se es usuario administrador.
- Permitir a usuario administradores modificar la inscripción de un estudiante
- Permitir añadir materias disponibles.
- Ver y listar estudiantes inscritos si se es un usuario administrador.
- Ver y listar estudiantes agrupados por materias inscritas si se es un usuario administrador.
- Ver perfil de estudiante que debe contener data relacionada a su inscripción y registro académico si se es un usuario administrador.
- Generar reportes de alumnos inscritos si se es un usuario administrador.
- Exportar listado de alumnos y data de inscripción a archivos con formato xlsx si se es un usuario administrador.
- Inscribir materias si se es un usuario estudiante.
- Validar choque de horarios entre materias e impedir a un estudiante inscribirlas el mismo semestre.

- Verificar capacidad de alumnos por aula e impedir a un estudiante inscribirse en materias con capacidad llena.
- Evaluar límites de créditos por semestre e impedir a un estudiante inscribir materias cuya suma de créditos supere el límite máximo o sea inferior al límite mínimo.
- Validar prelaciones e impedir a los estudiantes inscribir materias de las cuales no hayan cumplido su respectiva prelación.
- Generar comprobante de inscripción si se es un usuario estudiante.
- Enviar correo electrónico con lista de estudiantes inscritos a los profesores.
- Enviar correo electrónico al estudiante con su comprobante de inscripción.

Capítulo 2

Marco conceptual

El siguiente capítulo cumple con el objetivo de definir los conceptos necesarios para comprender la resolución de este trabajo. En gran medida hace referencia a conceptos relacionados al desarrollo de una aplicación web, comenzando por una definición de aplicación web, para luego explicar elementos relacionados al front-end y al back-end del sistema a realizar.

Aplicación web

En este documento el término aplicación es definido como cualquier programa de computación diseñado para desempeñar un conjunto coordinado de funciones, tareas o proceso orientados al beneficio de un usuario, o en algunos casos, para otro programa de aplicación. Basándonos en lo anterior una aplicación web es una aplicación basada en el intercambio de información, conocida como hipertexto, vía internet. Las aplicaciones web se basan en protocolos de internet para el intercambio de información, y entre sus lenguajes comunes se encuentran aquellos que son interpretados por navegadores web, sirviendo estos últimos como clientes de la aplicación, y aquellos lenguajes que se ejecutan en ordenadores remotos que sirven como servidores de la aplicación, lenguajes de los cuales hay una gama amplia de uso, por lo general tienden a soportar programación orientada a objetos y/o ser multiparadigmas.

Front-end de una aplicación web

En una aplicación web, el front-end es la parte de la aplicación con la que el usuario interactúa directamente, por lo cual se encuentra ubicada en el cliente, y por tanto sus interfaces, servicios y lógicas están dirigidos en base a ello, en el front-end recae la experiencia de usuario y usabilidad de la aplicación, así como la captación de datos ingresados por un usuario. El front-end permite al usuario acceder y solicitar los recursos y funcionalidades del sistema de información subyacente o back-end.

Herramientas a usar para el front-end

Dados los intereses y propósitos del front-end en una aplicación, las siguientes herramientas usadas en el presente trabajo son:

HTML

Hypertext Markup Language, es un lenguaje basado en etiquetas usado para crear y dar estructura al contenido de las páginas web, y una de las herramientas más básicas para hacerlo, determina el contenido de una página web y su estructura más no su funcionalidad. Los documentos HTML son interpretados por los navegadores web, y permiten proveer enlaces a otros documentos en la web e incorporar elementos de hipermedia.

CSS

Cascading Style Sheets, es un lenguaje diseñado para determinar la presentación del contenido de documentos HTML y XML, Permitiendo modificar el estilo gráfico de los

elementos de una interfaz web. Es el navegador web el encargado de interpretar estos estilos y desplegarlos en la interfaz gráfica de cara al usuario.

Bootstrap

Es un marco de trabajo de desarrollo front-end que define conjuntos de herramientas y estilos gráficos para ayudar a definir la representación visual de una interfaz web y agilizar su desarrollo. Bootstrap cuenta con un sistema de cuadrillas que facilita el posicionamiento de los elementos y la segmentación visual de las secciones de una interfaz, su estilos incorporan diseño sensible para que la organización y tamaño de los elementos de la interfaz cambien en base al tamaño de la pantalla del dispositivo en el que se esté mostrando y contiene elementos de diseño predefinidos como menús, formularios, botones, entre otros, con los que se ayuda a enriquecer la experiencia de usuario.

Javascript

Es un lenguaje de programación ligero, interpretado y orientado a objetos con funciones de primera clase. Su sintaxis está basada en los lenguajes de programación C++ y Java.

JavaScript puede funcionar como lenguaje procedimental y como lenguaje orientado a objetos. Los objetos se crean programáticamente añadiendo métodos y propiedades a lo que de otra forma serían objetos vacíos en tiempo de ejecución, en contraposición a las definiciones sintácticas de clases comunes en los lenguajes compilados como C++ y Java. Una vez se ha construido un objeto, puede usarse como modelo (o prototipo) para crear objetos similares. (MDN Web Docs, 2015)

Es usado comúnmente en el lado del cliente de una aplicación web, puede diseñar o programar como las páginas web se comportan ante la ocurrencia de un evento. Es muy conocido como lenguaje de scripting para páginas web, aunque también es usado en otros ambientes no basados en navegadores como Node.Js, Apache CouchDB y Adobe Acrobat.

Javascript también soporta programación funcional, las funciones son objetos, dándole a las funciones la capacidad de mantener código ejecutable y ser pasado a otro lugar como cualquier otro objeto.

AJAX

Hace referencia a "Javascript Asíncrono + XML", en sí misma no es una tecnología, se concibe como un término que describe una nueva forma de usar un conjunto existentes de tecnologías juntas, incluyendo HTML o XHTML, Cascading Style Sheets, Javascript, el Document Object Model, XML, y más importante, el objeto XMLHttpRequest.

El objeto XMLHttpRequest proporciona una forma fácil de obtener información de una URL sin tener que recargar la página completa. Una página web puede actualizar sólo una parte de la página sin interrumpir lo que el usuario está haciendo. (MDN Web Docs, 2015)

Cuando estas últimas tecnologías mencionadas son combinadas en el modelo de Ajax, las aplicaciones web pueden hacer actualizaciones rápidas e incrementales en la interfaz de usuario sin recargar completamente la página del navegador. Esto hace que la aplicación sea más rápida y más "amigable" a las acciones del usuario y se pueda procesar datos provenientes del servidor en el lado cliente de la aplicación.

Back-end de una aplicación web.

Se refiere a la parte de una aplicación (programas, subrutinas, servicios, entre otros) que no son accedidos directamente por el usuario y se encuentra ubicada del lado del servidor, en el desarrollo de back-end conciernen temas de seguridad, estructura y manejo de contenido, esto incluye manejo de base de datos y servidores. El back-end de una aplicación se encarga de proveer a front-end recursos y funcionalidades para que puedan ser presentadas al usuario, también aloja lógica de la aplicación y reglas de negocio.

Herramientas a usar para el back-end

Las herramientas a usar en el presente trabajo para realizar el back-end de la aplicación son:

Servidor HTTP Apache

Un servidor web es un término que puede ser empleado para referirse tanto a hardware como a software, en hardware es la computadora que almacena los archivos que componen un sitio web, en software, que es la acepción a la que se hace referencia en esta definición, es un programa encargado de recibir peticiones enviadas por un cliente (típicamente un navegador web), vía el protocolo de internet HTTP, una vez la petición es recibida, el servidor envía el recurso solicitado respondiendo a la solicitud usando nuevamente el protocolo HTTP. El servidor HTTP Apache es un servidor web de código abierto, desarrollado y mantenido por La Fundación de Software Apache, es rápido, confiable y seguro, altamente configurable y bastante completo en características.

PHP

Es un lenguaje de programación de propósito general del lado del servidor, diseñado para el desarrollo web de contenido dinámico, el código php puede ser incluido dentro de documentos HTML para manejar variables, estados, condiciones y lógica en general, o puede funcionar como código por sí solo. Este lenguaje es interpretado por un servidor web con módulo de procesamiento de php, y como resultado genera una página web resultante que es enviada al navegador para ser desplegada del lado del cliente como resultado de una petición HTTP.

CodeIgniter

Es un marco de trabajo (o framework) de PHP ligero y con poca carga de procesamiento, de código abierto, para el desarrollo rápido de aplicaciones web que se encuentra basado en el patrón de diseño MVC. Codeigniter cuenta con soporte en temas de seguridad, una documentación clara y completa; requiere poca configuración inicial, no requiere adherirse a estrictos estándares de código, provee de una estructura lógica y de una interfaz reusable para acceder a librerías implementadas para desempeñar tareas comunes. Fue construido para desarrolladores que necesitan un kit de herramientas simples y elegantes para crear aplicaciones web completas.

Base de datos

Una base de datos es un repositorio centralizado de datos lógicamente relacionados, que permite almacenar y organizar hechos o eventos y restituirlos a demanda de él, o los usuarios para producir información. (Ospina, 2009)

Los datos están distribuidos por campos, registros y archivos. Poseen sus índices que permiten hacer más fácil buscar información relevante.

SQL

SQL quiere decir en español Lenguaje de Consultas Estructurado, es un lenguaje estandarizado de programación para manejar bases de datos relacionales y ejecutar distintos tipos de operaciones sobre los datos que ellas guardan.

Aunque el lenguaje SQL se considere un lenguaje de consultas, contiene muchas otras capacidades además de la consulta en bases de datos. Incluye características para definir la estructura de los datos, para la modificación de los datos en la base de datos y para la especificación de restricciones de seguridad. (Silberschatz, 2002).

Base de datos relacional

Es una base de datos tabular en la que los datos son definidos de manera que se pueda acceder o reorganizar en un número diferente de formas. Las bases de datos relacionales se componen por un conjunto de tablas con datos que se ajustan a una categoría predefinida. Cada tabla tiene al menos una categoría de datos en una columna, y en cada fila hay una determinada instancia de datos para las categorías que son definidas en las columnas.

Una base de datos relacional posee la siguiente estructura:

- Tabla: Una tabla es una estructura lógica hechas de filas y columnas. Normalmente si se necesita recuperar información de una tabla, se tiene que ordenar los datos ya que las filas no poseen un orden fijo. En la intersección de cada columna con una fila hay un dato específico comúnmente llamado valor.
- Índices: Un índice es un conjunto ordenado de apuntadores a filas de una tabla.
- Llaves: Una llave es una o más columnas que son identificadas de ese modo. Ellas son usadas para establecer e identificar relaciones entre tablas y también para identificar de manera única un registro o fila dentro de una tabla.

Sistema manejador de base de datos.

Consiste en una colección de datos interrelacionados y un conjunto de programas para acceder a dichos datos. (Silberschatz, 2002)

En esencia es un sistema computarizado para la gestión de datos. Los usuarios de este sistema tienen la facilidad de usar distintos tipos de tareas, ya sea para la manipulación de los datos en la base de datos o la estructura. Proporciona una manera de guardar los datos de una base de datos de manera que sea tanto práctica como eficiente, normalmente basándose en un lenguaje que permita realizar esta tarea.

MySQL

Es un sistema manejador de bases de datos relacionales, basado en el lenguaje SQL para el manejo de las bases de datos. Además es una herramienta de Código Abierto que quiere decir que cualquiera puede usar o modificar el software.

MySQL se puede usar en distintas plataformas como Linux, Unix o Windows. Se puede usar para cualquier tipo de aplicaciones.

Patrón de diseño MVC

El Modelo-Vista-Controlador (MVC) es un paradigma de programación que separa una aplicación en tres componentes principales: el modelo, la vista y el controlador. Esto con el objetivo de simplificar y mejorar el proceso de desarrollo de software. A continuación se explica con más detalle cada una de las partes de este paradigma.

- Modelo: El modelo representa la parte de la aplicación que implementa la lógica para el manejo de los datos en la aplicación. El modelo normalmente almacena su estado de una manera persistente en una base de datos para luego acceder, agregar, modificar o eliminar datos sobre ese repositorio.
- Vista: La vista representa los módulos que se encargaran de instanciar la interfaz gráfica de usuario de una aplicación.
- Controlador: Es el encargado de gestionar y responder a la entrada de datos realizada por un usuario. Este elemento del paradigma captura la interacción del usuario dentro de una aplicación; normalmente se comunica con el modelo para

obtener datos, y luego se relaciona con la vista para mostrar los datos en una interfaz de usuario.

El paradigma MVC ayuda a crear aplicaciones que separan los diferentes “asuntos” de la aplicación (lógica de entrada, lógica de negocio y lógica de interfaz de usuario).

Debido a la relación entre los tres componentes principales de una aplicación MVC, se facilita la realización de un desarrollo en paralelo.

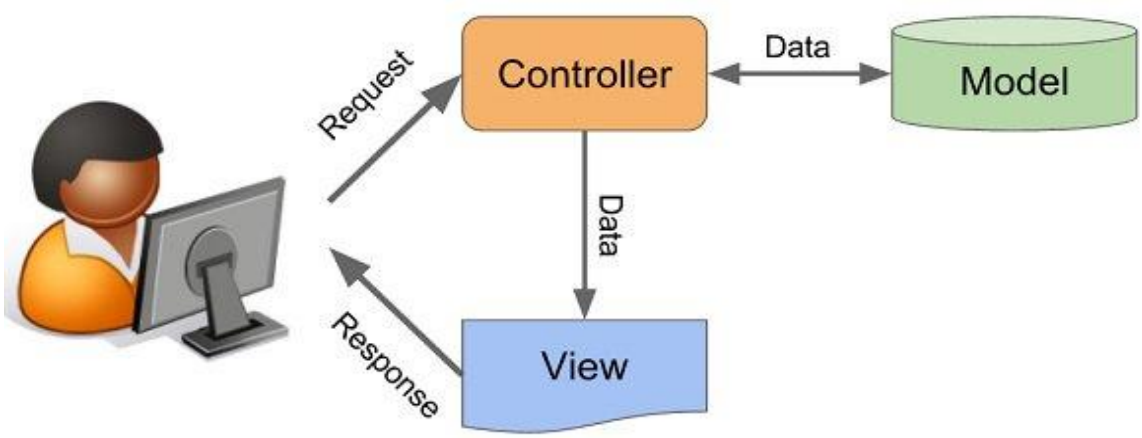

Figura 1 MVC

Capítulo 3

Marco Metodológico

En la siguiente sección se expone el marco metodológico a ser usado para el desarrollo de la aplicación web producto de este proyecto. Se conceptuliza de manera concisa cada componente que está incluido, incorporando una estimación sobre la duración de la realización del proyecto, además de justificar el marco metodológico seleccionado.

Metodología ágil.

Una metodología es un enfoque a la gestión de proyectos que puede entenderse como una gestión ágil de proyectos, y es típicamente usada en el desarrollo de software. Su visión contempla la flexibilidad, oponiéndose a un esquema rígido que no soporta la incidencia de cambios durante el desarrollo de un proyecto, por lo que ayuda al equipo involucrado a responder a lo impredecible que es construir software a través de cadencias de trabajo iterativas e incrementales, conocidas en algunos casos como Sprints. Mediante la cualidad iterativa se refina la visión del producto final después de los ciclos de desarrollo. Siendo que se concibe que la visión y el alcance de un producto pueden no estar completamente definidos al iniciar el proyecto, se permite que estos se vayan refinando junto con el producto, por lo que es una práctica común iniciar el desarrollo teniendo como objetivo conseguir un producto mínimo viable. La iterabilidad permite tomar cuales opciones son mejores para las subsiguientes versiones del producto, y que necesita ser descartado del desarrollo, a través de revisión y corrección continúa. La incrementalidad

permite construir y añadir partes del producto final hasta conseguirlo, cada incremento es una porción funcional del sistema potencialmente entregable,

La comunicación entre los equipos y sus miembros es bastante acentuada en las metodologías ágiles y se favorece a ésta frente a la documentación extensiva y los equipos con los que se cuentan son multidisciplinarios.

El desarrollo ágil ha encontrado supuestos claves en los que basan su marco de trabajo, estos señalan que es difícil predecir qué requisitos persistirán y cuales cambiarán, de la misma forma con las prioridades del cliente; es complicado predecir cuánto diseño es necesario antes de llegar a implementarlo por lo que se intercala con el desarrollo y se aprueba el diseño mientras se crea; y desde el punto de planificación no es previsible el análisis, el diseño y la implementación necesaria de un proyecto. Para responder a estos supuestos claves se debe tener una abundante retroalimentación con el cliente al involucrarlo en el proceso de desarrollo e ir continuamente entregando incrementos funcionales que puedan irse adaptando según el entorno y las necesidades cambien.

El documento que sentó las bases del movimiento ágil es el denominado Manifiesto Ágil para el desarrollo de software, en el que se definen los principios del software ágil y se menciona aquello que tiene más valor en el desarrollo de software, en este sentido, se valoran:

- Las interacciones individuales por encima de los procesos y herramientas.
- Software funcional por encima de documentación extensiva.
- Colaboración con el cliente por encima de negociación contractual.
- Respuesta al cambio por encima de apegarse a un plan.

Con esto se expresa que hay valor en todos los elementos, salvo que se indican cuales se valoran más.

Metodología de desarrollo

SCRUM es un marco de trabajo para desarrollar, entregar y mantener productos complejos, (Sutherland, 2017). Su objetivo es la gestión de proyectos, y posee un diseño que permite que proyectos en los que sus requisitos cambian rápidamente, puedan ser manejados bajo este marco de manera eficiente.

SCRUM no es un metodología de desarrollo de software per se, sino más bien un framework de metodologías ágiles que posee varios procedimientos o técnicas para alcanzar el desarrollo de un producto, y es el que se va a emplear en este trabajo.

A continuación se describen los diferentes elementos involucrados en la aplicación de este marco de trabajo.

El Equipo SCRUM

El equipo SCRUM posee tres roles necesarios para llevar a cabo el desarrollo de un proyecto, estos son el Dueño del Producto, El Equipo de Desarrollo y el Maestro SCRUM.

El Dueño del Producto

Es el responsable de representar los intereses de todos durante el desarrollo del producto y su resultado final. En un comienzo es el encargado de crear los requisitos generales del proyecto, esta lista de requisitos es llamada Product Backlog. El Dueño del

Producto usará el Product Backlog para asegurar que las funcionalidades más importantes sean realizadas con prioridad. Este rol será fungido por el co-tutor Rafael Romero, miembro de la Escuela de Estudios Políticos y Administrativos.

El Equipo de Desarrollo

Es un rol en que los miembros se auto gestionan y son multifuncionales. El Equipo es el encargado de desarrollar las funcionalidades del producto de acuerdo al Product Backlog, además de ser responsables de convertir los elementos de esta lista, en un Incremento de las funcionalidades dentro de una iteración. El equipo se encuentra conformado por los autores del presente trabajo Keiber Prin y Efraín Díaz.

Maestro SCRUM

Es el encargado de todo el proceso SCRUM. Enseña como es el proceso SCRUM, implementa SCRUM para que encaje en una organización y que entregue los resultados deseados. Además asegura de que todos sigan las prácticas y reglas de SCRUM. Este rol será fungido por la tutor del trabajo Vanessa Leguizamo.

Eventos SCRUM

Sprint

Un Sprint es el procedimiento de adaptación de las cambiantes variables del entorno (requerimientos, tiempo, recursos, conocimiento, tecnología). Son ciclos iterativos en los cuales se desarrolla o mejora una funcionalidad para producir nuevos incrementos.

Durante un Sprint el producto es diseñado, codificado y probado. Y su arquitectura y diseño evolucionan durante el desarrollo. (Peralta, 2003)

En el presente proyecto cada Sprint se toma como una iteración de 15 días. Este lapso de tiempo es durante el cual el equipo de trabajo desarrollará los requisitos acordados en el Product Backlog y los transforma en un Incremento.

Planificación de Sprint

Este evento corresponde a una reunión con la cual se comienza cada Sprint. La reunión está dividida en dos, durante un primer segmento el Dueño del Producto presenta los requisitos de alta prioridad que se encuentran en el Product Backlog al Equipo de Desarrollo. En conjunto con el Dueño del Producto, el Equipo de Desarrollo determina que tareas de los requisitos pueden ser alcanzadas en el siguiente Sprint. Durante la segunda parte de la reunión se discute cómo se realizarán las tareas seleccionadas de manera detallada en forma de plan en el Sprint Backlog.

SCRUM Diario

Es una reunión breve diaria en la cual miembros del Equipo de Desarrollo discuten el trabajo realizado durante el último día, qué planean hacer durante el siguiente día y qué problemas han habido. El propósito es coordinar el trabajo y el progreso de los miembros del equipo. En este proyecto no se contará con este evento, ya que no lo consideramos necesario para el desarrollo puesto que no se estará dedicando tiempo completo al desarrollo del proyecto.

Revisión de Sprint

Al finalizar un Sprint, una reunión es llevada a cabo con el objetivo de que el Equipo de Desarrollo muestre al co-tutor Rafael Romero los avances alcanzados en el desarrollo del proyecto; solo las funcionalidades completas son mostradas mediante un entregable para ser probado. Producto de la retroalimentación obtenida se va a determinar lo que se hará luego, promoviendo la colaboración y realizando posibles cambios, de ser necesario, al Product Backlog de acuerdo al nuevo estado del proyecto.

Artefactos SCRUM

Product Backlog

Es una lista de requisitos funcionales y no funcionales, que son desarrolladas y transformadas en funcionalidades. Está basada en prioridades y dividida en propuestas de lanzamientos. Además posee una estimación calculada en puntos de historia (esfuerzo en realizar una tarea), este último vinculado a un elemento de referencia; para la obtención de la estimación de una historia solo basta compararla com la historia de referencia para obtener su valor. La primera historia en el product backlog representa la historia de referencia.

Los cambios realizados en la Product Backlog reflejan los cambios en los requisitos del negocio y cambios tecnológicos, aunque estos cambios sólo pueden realizarse antes de comenzar un Sprint.

El product backlog es el siguiente.

Historia de Usuario	Estimación	Prioridad
Como un usuario administrador, quiero que se tenga un medio de autenticación en el sistema, para verificar la identidad de quienes acceden al mismo, proteger sus datos y restringir el acceso a las funcionalidades.	1	12
Como usuario administrador, quiero tener distinción en las funcionalidades y acciones a las que tengo acceso respecto a los usuarios estudiantes. Esto para evitar que se hagan acciones indeseadas por personal no autorizado.	2	13
Como usuario administrador, quiero poder crear usuarios con rol estudiante. Esto mediante un formulario o cargando un archivo en el sistema, para poder proveer acceso a los usuarios estudiantes al sistema.	1	14
Como usuario administrador, puedo crear horarios dentro de la aplicación, para poder asignarlos a las materias que se dicten en el momento de la inscripción.	1	10
Como usuario administrador quiero exportar los estudiantes inscritos junto con la información de su inscripción, para tener en un medio trasladable dicha información. Esto mediante la generación de un archivo en formato xlsx.	3	18
Como usuario administrador quiero exportar el listado de estudiantes inscritos de una materia en particular, para tener segmentado el conjunto de estudiantes y visualizar más granularmente la información. Esto mediante la generación de un archivo en formato xlsx.	3	19
Como usuario estudiante quiero inscribir materias del semestre en curso, para que mi inscripción sea validada y registrada en la Escuela. Esto mediante la selección de materias en un formulario en el sistema	2	3
Como usuario estudiante no puedo materias que presenten choque de horario, para que se cumpla la restricción de la Escuela al respecto.	3	4
Como usuario estudiante no puedo inscribir en una materia cuyos cupos hayan sido agotados.	1	5
Como un usuario administrador, yo puedo modificar la inscripción de un estudiante dentro del sistema.	2	8
Como un usuario administrador, yo puedo agregar las materias que	2	9

comprenden el pensum de la carrera que realizará la inscripción.		
Como un usuario administrador, yo puedo agregar las materias habilitadas para la inscripción junto con sus horarios para que los estudiantes las puedan elegir durante el momento de la inscripción.	3	11
Como un usuario administrador, yo puedo ver individualmente o listar los estudiantes inscritos dentro del sistema, esto con el fin de obtener un reporte de los estudiantes inscritos.	5	15
Como un usuario administrador, yo puedo generar la lista de los estudiantes agrupados por las materias que hayan inscrito, esto permitirá obtener la nómina de estudiantes correspondientes a cada sección.	3	16
Como un usuario administrador, puedo ver el perfil de un estudiante cargado dentro del sistema para conocer sus datos relacionados a su inscripción y registro académico, así como de su usuario, correo.	1	17
Como un usuario estudiante, quiero que al momento de intentar realizar la inscripción el sistema evalúe si las condiciones de límites de créditos se cumplen para poder realizar la inscripción de las materias seleccionadas.	3	6
Como un usuario estudiante, puedo generar un comprobante luego de haber culminado el proceso de inscripción, el cual contiene toda la información relacionada con mi inscripción.	5	7
Como usuario administrador, puedo seleccionar una materias habilitada para la inscripción y por medio de un formulario en el navegador colocar la dirección de correo electrónico de un profesor para luego enviar la nómina de estudiantes a esa dirección de correo.	5	20
Como usuario estudiante, yo quiero recibir un correo electrónico luego de culminar el proceso de inscripción, que contiene el comprobante de inscripción generado por el sistema.	2	21
Realizar la investigación de los requerimientos de software y hardware para el desarrollo del proyecto.	2	1
Definir e implementar el modelo de base de datos.	3	2
Implementar plantillas HTML y CSS para mejorar la interfaz de usuario.	2	22
Como usuario administrador quiero importar los datos de los estudiantes que realizaran la inscripciones usando la aplicación.	4	23
Implementar módulo de configuración para el envío de correos	1	24

Sprint Backlog

Define el trabajo o las tareas que el Equipo de Desarrollo selecciona para convertir las tareas del Product Backlog en un Incremento de la funcionalidad del proyecto para ese Sprint. Esta lista surge durante un Sprint, el Equipo de Desarrollo compila una lista inicial de esas tareas en la segunda parte de la Planificación de Sprint. Solo el Equipo de Desarrollo puede cambiar el contenido de ella. El Sprint Backlog hace visible todo el trabajo que será realizado en la iteración de manera detallada.

Cada tarea tiene sus responsables para llevarla a cabo y la cantidad de trabajo restante en cualquier momento durante el Sprint.

Incremento

El Incremento es la suma de todos los elementos del Product Backlog completados durante un Sprint y el valor de los incrementos de todos los Sprints anteriores. Al final de un Sprint el nuevo Incremento debe estar "Terminado", lo cual significa que está en condiciones de ser utilizado y que cumple la Definición de "Terminado" del Equipo SCRUM. (Sutherland, 2017)

Estos son desarrollados por el Equipo de Desarrollo durante cada Sprint.

Planificación estimada del proyecto

Se estimó realizar el en ocho (7) Sprints de dos (2) semanas cada uno, los siguientes serán llevados a cabo de la siguiente manera:

Sprint 0

Historia de Usuario	Estimación	Prioridad
Realizar la investigación de los requerimientos de software y hardware para el desarrollo del proyecto.	2	1

Sprint 1

Historia de Usuario	Estimación	Prioridad
Definir e implementar el modelo de base de datos.	3	2

Sprint 2

Historia de Usuario	Estimación	Prioridad
Como usuario estudiante quiero inscribir materias del semestre en curso, para que mi inscripción sea validada y registrada en la Escuela. Esto mediante la selección de materias en un formulario en el sistema	2	3
Como usuario estudiante no puedo materias que presenten choque de horario, para que se cumpla la restricción de la Escuela al respecto.	3	4
Como usuario estudiante no puedo inscribir en una materia cuyos cupos hayan sido agotados.	1	5
Como un usuario estudiante, quiero que al momento de intentar realizar la inscripción el sistema evalúe si las condiciones de límites de créditos se cumplen para poder realizar la inscripción de las materias seleccionadas.	3	6
Como un usuario estudiante, puedo generar un comprobante luego de haber culminado el proceso de inscripción, el cual contiene toda la información relacionada con mi inscripción.	5	7

Sprint 3

Historia de Usuario	Estimación	Prioridad
Como un usuario administrador, yo puedo modificar la inscripción de un estudiante dentro del sistema.	2	8
Como un usuario administrador, yo puedo agregar las materias que comprenden el pensum de la carrera que realizará la inscripción.	2	9
Como usuario administrador, puedo crear horarios dentro de la aplicación, para poder asignarlos a las materias que se dicten en el momento de la inscripción.	1	10
Como un usuario administrador, yo puedo agregar las materias habilitadas para la inscripción junto con sus horarios para que los estudiantes las puedan elegir durante el momento de la inscripción.	3	11
Como un usuario administrador, quiero que se tenga un medio de autenticación en el sistema, para verificar la identidad de quienes acceden al mismo, proteger sus datos y restringir el acceso a las funcionalidades.	1	12
Como usuario administrador, quiero tener distinción en las funcionalidades y acciones a las que tengo acceso respecto a los usuarios estudiantes. Esto para evitar que se hagan acciones indeseadas por personal no autorizado.	2	13
Como usuario administrador, quiero poder crear usuarios con rol estudiante. Esto mediante un formulario o cargando un archivo en el sistema, para poder proveer acceso a los usuarios estudiantes al sistema.	1	14
Como un usuario administrador, yo puedo ver individualmente o listar los estudiantes inscritos dentro del sistema, esto con el fin de obtener un reporte de los estudiantes inscritos.	5	15
Como un usuario administrador, yo puedo generar la lista de los estudiantes agrupados por las materias que hayan inscrito, esto permitirá obtener la nómina de estudiantes correspondientes a cada sección.	3	16
Como un usuario administrador, puedo ver el perfil de un estudiante cargado dentro del sistema para conocer sus datos relacionados a su inscripción y registro académico, así como de su usuario, correo.	1	17
Como usuario administrador quiero exportar los estudiantes inscritos junto con la información de su inscripción, para tener en un medio trasladable dicha información. Esto mediante la generación de un archivo en formato xlsx.	3	18
Como usuario administrador quiero exportar el listado de estudiantes	3	19

inscritos de una materia en particular, para tener segmentado el conjunto de estudiantes y visualizar más granularmente la información. Esto mediante la generación de un archivo en formato xlsx.		

Sprint 4

Historia de Usuario	Estimación	Prioridad
Como usuario administrador, puedo seleccionar una materias habilitada para la inscripción y por medio de un formulario en el navegador colocar la dirección de correo electrónico de un profesor para luego enviar la nómina de estudiantes a esa dirección de correo.	5	20
Como usuario estudiante, yo quiero recibir un correo electrónico luego de culminar el proceso de inscripción, que contiene el comprobante de inscripción generado por el sistema.	2	21

Sprint 5

Historia de Usuario	Estimación	Prioridad
Implementar plantillas HTML y CSS para mejorar la interfaz de usuario.	2	22

Sprint 6

Historia de Usuario	Estimación	Prioridad
Como usuario administrador quiero importar los datos de los estudiantes que realizaran la inscripciones usando la aplicación.	4	23
Implementar módulo de configuración para el envío de correos	1	24

Justificación de selección de metodología.

Las metodologías ágiles traen consigo un conjunto de características que son consideradas de alto valor para el proyecto. Tener al alcance una frecuente

retroalimentación del cliente a lo largo del proceso de desarrollo permitirá en primera instancia construir el producto correcto, es decir, aquel que cumpla con las necesidades del cliente, por lo cual tiene sentido que su inclusión sea clave, también permitirá evaluar los cambios, realizar correcciones lo antes posible en el desarrollo del proyecto y ahorrar esfuerzos en cambios de requerimientos y corrección de errores en etapas tardías del desarrollo en las cuales terminan siendo más costosos. Dado que en este proyecto se tiene acceso al cliente y disponibilidad de parte del mismo a ser partícipe para proveer de retroalimentación, las condiciones se dan para apoyar la metodología en este sentido, se prefiere de esta manera un enfoque ágil en lugar de emplear la metodología Cascada en la que por ejemplo hay poco o ningún espacio a realizar cambios una vez realizado el levantamiento inicial de requerimientos o usar RUP en el que la robustez de los artefactos, fases procesos no se ajustaría bien al alcance de un proyecto como este.

Mantener periodos cortos de tiempo definidos para desarrollar incrementos de funcionalidad como plantean las metodologías ágiles ayuda a mantener las metas concretas, alcanzables, medibles y revisables por el cliente, como resultado de una correcta definición de las tareas, lo que ayuda a definir y mantener un ritmo de progreso además de contar con indicadores que ayuden a medirlo, como por ejemplo el tiempo de completitud de las tareas de un Sprint y la cantidad de trabajo realizada respecto al Sprint anterior. Este manejo del tiempo se ajustaría al proyecto como una herramienta de autogestionar el trabajo de una manera simple y efectiva con la mínima cantidad de actividades y herramientas indispensables para planificar el desarrollo, ya que la documentación extensiva no es prioridad en los requerimientos del cliente y el uso de artefactos adicionales será empleado en caso de ser considerado necesario con el fin de priorizar el desarrollo del producto. La

realización de fases de mayor extensión que son contempladas en otras metodologías restaría a la interacción con el cliente.

Dentro de las metodologías ágiles se seleccionó el marco de trabajo SCRUM. En contraste con otras metodologías ágiles como XP (Extreme Programming) siendo otra candidata considerada, mantiene ciclos de desarrollo de duración similar, siendo SCRUM en la que estos pueden tener una duración mayor, sin embargo una diferencia puntual radica en que en SCRUM una vez la Planificación de Sprint ha sido completada y se ha asumido el compromiso de entregar un conjunto de los elementos definidos en el Product Backlog, estos deben mantenerse inalterados. XP es más flexible al respecto y permite intercambiar en una iteración una característica en la que el equipo aún no haya empezado a trabajar por otra de tamaño equivalente. Mantener invariantes las tareas a realizar durante un Sprint como en SCRUM se considera que permitirá al equipo concentrar el esfuerzo en el desarrollo de lo acordado una vez priorizada y seleccionada la carga de trabajo de un Sprint y evitar retomar actividades de planificación que dispersen el foco de cómo dirigir el esfuerzo, por lo cual el énfasis que se le dé a la planificación es de crucial importancia para un correcto desenvolvimiento del Sprint, algo que el equipo debe internalizar.

En XP, las funcionalidades a atacar en un iteración deben ser desarrolladas en estricto orden de prioridad. En SCRUM el Dueño del Producto prioriza los elementos de Product Backlog pero una vez las funcionalidades están en el Sprint Backlog el equipo tiene la libertad de seleccionar el orden en el que serán desarrolladas, lo que es preferido en pro de dar lugar al criterio del equipo en autogestionar qué orden se ajusta mejor a la dinámica de trabajo y asignación de tareas.

XP define prácticas de ingeniería bajo las que trabajar, en tanto que SCRUM no. Mientras que las prácticas definidas por XP pueden ser de alto aporte y útiles, no hay porque estar ligadas a ellas sino dejar al equipo explorar y descubrir el valor en ellas por sí mismo e incorporarlas si así se quiere ya que SCRUM deja la puerta abierta en este aspecto.

De esta manera quedan justificados los criterios bajos los cuales se seleccionó la elección de la metodología y su marco de trabajo.

Capítulo 4

Marco Aplicativo

El siguiente capítulo presenta las iteraciones llevadas a cabo para el desarrollo de la aplicación de la cual es objeto este proyecto, es importante destacar que la escuela de Estudios Políticos y Administrativos pidió usar la aplicación con las funcionalidades primordiales implementadas en el proceso de inscripción II-2018 a modo de prueba. Por tanto se tuvo que reorganizar las tareas a realizar con el objetivo de cumplir los nuevos marcos de tiempo establecidos y poder usar la aplicación con anterioridad en una versión beta. Además se agregó una iteración extra que se encarga de corregir los errores encontrados en la versión beta de la aplicación.

Iteración 0

Descripción: Realizar la investigación de los requerimientos de software y hardware para la implementación inicial del proyecto.

Fecha de inicio / Fecha fin: 27/08/2018 al 10/09/2018

Diseño:

En primer lugar fue necesario comunicarse con el Dueño del Producto representado por el profesor Rafael Romero para entender los requerimientos y las necesidades del proyecto. Estos datos fueron recolectados y luego analizados para poder escoger las herramientas que mejor se adaptan para resolver el problema; este análisis condujo al

planteamiento del uso de una aplicación web, junto con lenguajes de programación que permitan obtener un producto fácil de instalar en servidores de baja envergadura.

Entre estos lenguajes de programación es necesario tomar en cuenta que se tiene que permitir tener una base de datos para mantener un histórico de lo que ocurre en la aplicación debido al uso de ella, por eso el lenguaje web que se escogió se debía de comunicar fácilmente con una base de datos. El lenguaje de programación escogido fue PHP usando el framework CodeIgniter.

Esto genera una aplicación con una arquitectura similar a la que se muestra en la figura 2.

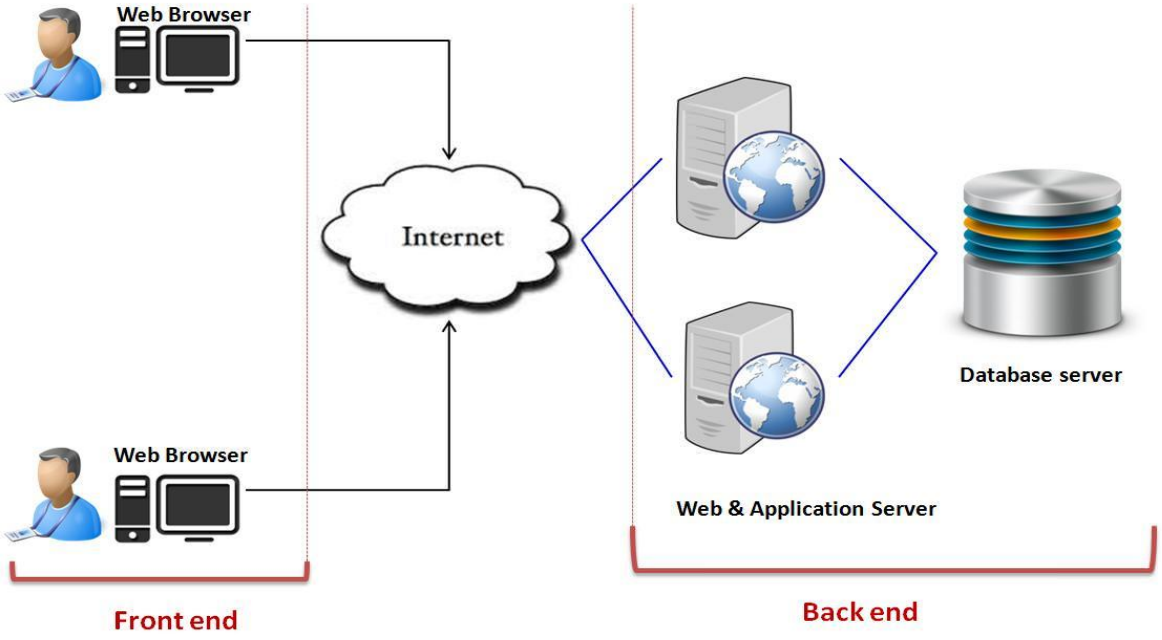

Figura 2 Arquitectura Front End - Back End

Codificación:

Para comprobar que el framework de programación escogido funcionara de acuerdo a los requerimientos del cliente, se realizó el despliegue de una aplicación a modo de prueba, esto basado en los pasos de instalación básicos recomendados en el sitio web de CodeIgniter https://codeigniter.com/user_guide/installation/index.html.

Para poder dar despliegue a una aplicación basada en este framework es necesario editar algunos campos en el archivo application/config/config.php, el más importante de ellos es la asignación de la URL base que usará la aplicación web (esto se muestra en la figura 3).

```
$config['base_url'] = '';
```

Figura 3: Configuración base_url

Luego debido a la necesidad de tener una repositorio de datos, fue necesario configurar las opciones de la base de datos para que pueda existir una comunicación entre la aplicación web y los datos de la aplicación, esta queda de la siguiente forma (figura 4)

```
$active_group = 'default';
$query_builder = true;

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

Figura 4 Configuración base de datos

Esto dio como resultado la instalación correcta del marco básico de la aplicación mostrando en pantalla la ventana inicial de CodeIgniter (figura 5).

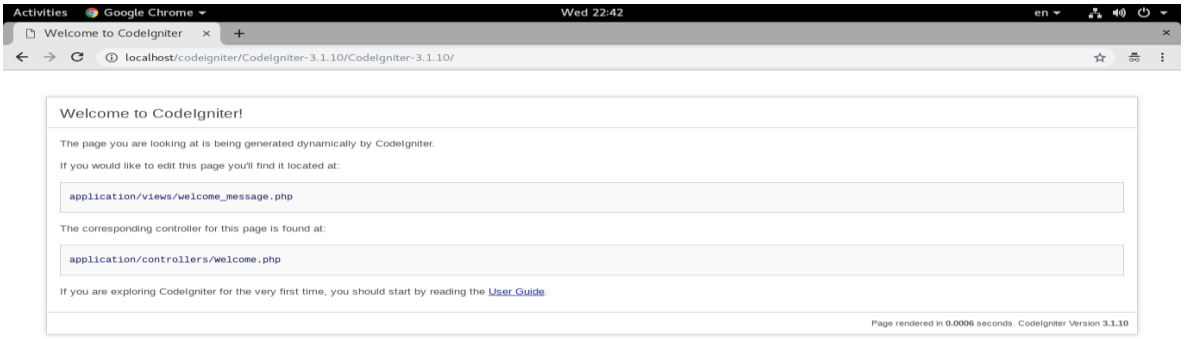

Figura 5: Pantalla inicial CodeIgniter

Pruebas:

Como prueba necesaria para la realización de esta iteración, se procedió a desplegar la aplicación en un ambiente real para conocer si el proceso de instalación era fácil de hacer en este tipo de ambiente y además percibir el desempeño de la aplicación. El resultado fue óptimo cumpliendo así los requerimientos iniciales del proyecto.

Iteración 1

Descripción: Definir e implementar el modelo de base de datos.

Fecha de inicio / Fecha fin: 10/09/2018 al 24/09/2018

Diseño:

Para la realización del modelo de datos se tomaron en cuenta diferentes aspectos relacionados no solo a la realización de las inscripciones sino como también al proceso previo de realización de esta.

Es necesario que estén disponibles en la aplicación las materias a ofertar durante el semestre para que los estudiantes las puedan seleccionar con la finalidad de cumplir con el proceso de inscripción. Se decidió crear una tabla “materias” que corresponde a un contenedor de las materias presentes en la Escuela de Estudios Políticos y Administrativos. Luego se creó una tabla con el nombre de “materias_disponibles”, esta posee las materias a ofertar durante un proceso de inscripción en particular, es decir, los estudiantes durante este proceso seleccionan las materias presentes en esta tabla para poder registrar su inscripción, “materias_disponibles” está basada en los datos existentes de la tabla “materias” además de incorporar campos que son de utilidad para la escuela, como por ejemplo la cantidad de cupos que indica el número máximo permitido de estudiantes por aula, el nombre del profesor y el correo electrónico de este último; sumado a eso posee un contador de los cupos ya ocupados de la materia disponible con el fin de reducir el procesamiento necesario para obtener esa información durante el uso de la aplicación en las inscripciones.

Convenientemente se planteó en el modelo de datos la creación de una tabla, del tipo muchos a muchos, llamada “materia_horario”, esta es el enlace de cada materia

disponible con horarios asignados en los cuales van a ser dictadas. Los horarios se plantearon para que sean guardados en una tabla con nombre “horarios” la cual además de poseer una hora de comienzo y una hora final posee el día de la semana al que pertenece este horario.

Una vez concretado los datos anteriores, la aplicación tiene que poder almacenar las materias disponibles inscritas por parte de los estudiantes. Para almacenar estos datos se creó una tabla llamada “cursa” que no es más que el contenedor de las inscripciones que cada estudiante realiza usando esta aplicación web.

Se creó una tabla “users” que representa a los usuarios que pueden acceder al sistema, estos corresponden tanto estudiantes de la escuela y adicional a ellos, el personal administrativo. Esta tabla posee características similares a una tabla usuarios de cualquier otro sistema. Con el fin de tener un control sobre los usuarios que pueden acceder al sistema se creó una tabla “groups” (relacionada a “user_groups”) y otra con el nombre de “login_attemps”, la primera con el fin de dar permisologías a los usuarios dentro de la aplicación, ya que un usuario estudiante no puede ver la misma información que vería un usuario del personal administrativo, y la segunda como una ayuda en cuestión de seguridad y rendimiento de la aplicación, que almacena los intentos de login de un usuario.

A continuación se presenta de manera gráfica el modelo planteado e implementado en la figura 6.

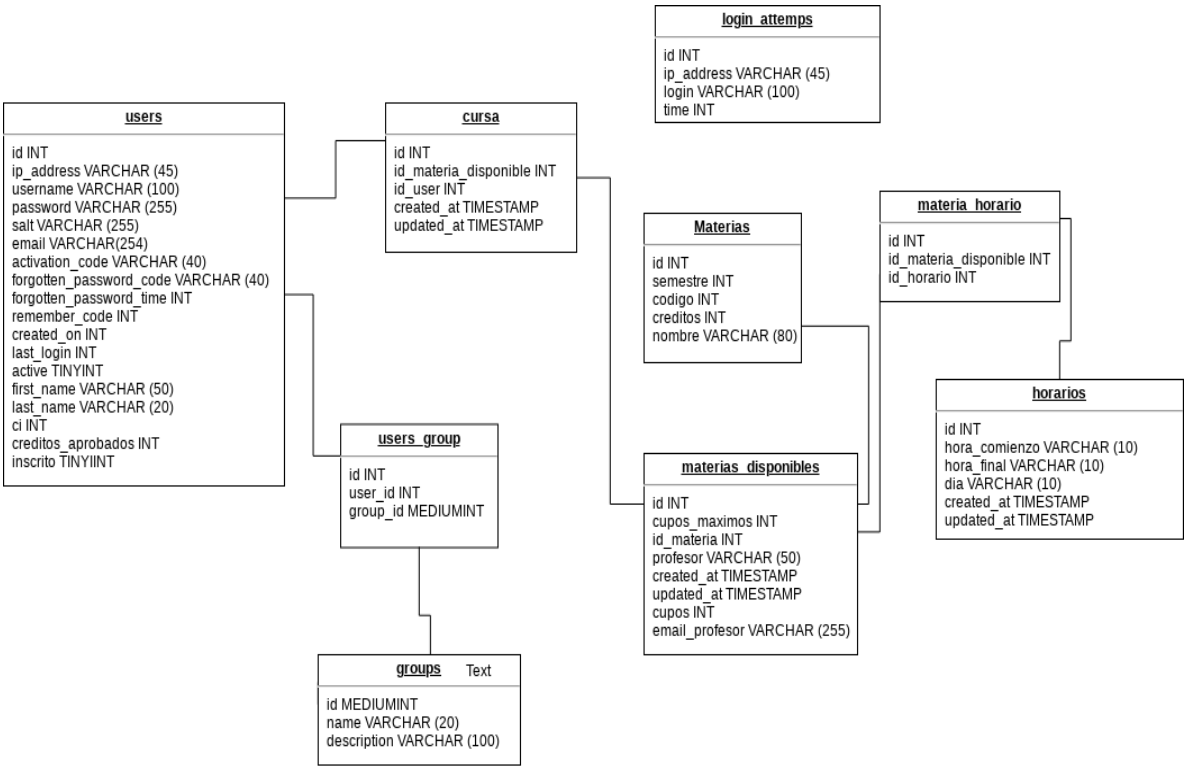

Figura 6: Modelo de datos

Codificación:

Para realizar la implementación se usó código sql para la creación de las tablas en el sistema, estas fueron ejecutadas en la interfaz de línea de comandos de MySQL. En la figura 7 se muestran las queries realizadas para la creación de las tablas “materias”, “materias_disponibles” y “cursa”, para la creación de las otras tablas se ejecutaron queries similares.

```
CREATE TABLE `materias` (
  `id` int(10) unsigned NOT NULL AUTO_INCREMENT,
  `semestre` int(11) NOT NULL,
  `codigo` int(11) DEFAULT NULL,
  `creditos` int(11) NOT NULL,
  `nombre` varchar(80) CHARACTER SET utf8 NOT NULL,
  `repite` tinyint(1) NOT NULL DEFAULT '0',
  PRIMARY KEY (`id`)
);

CREATE TABLE `materias_disponibles` (
  `id` int(10) unsigned NOT NULL AUTO_INCREMENT,
  `cupos_maximos` int(11) DEFAULT '0',
  `id_materia` int(11) NOT NULL,
  `profesor` varchar(50) COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `created_at` timestamp NULL DEFAULT NULL,
  `updated_at` timestamp NULL DEFAULT NULL,
  `cupos` int(11) NOT NULL DEFAULT '0',
  `email_profesor` varchar(255) COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  PRIMARY KEY (`id`)
);

CREATE TABLE `cursa` (
  `id` int(10) unsigned NOT NULL AUTO_INCREMENT,
  `id_materia_disponible` int(11) DEFAULT NULL,
  `id_user` int(11) NOT NULL,
  `created_at` timestamp NULL DEFAULT NULL,
  `updated_at` timestamp NULL DEFAULT NULL,
  PRIMARY KEY (`id`)
);
```

Figura 7: Queries para creación de materias, materias disponibles y cursa.

Pruebas:

No se realizaron pruebas.

Iteración 2

Descripción: Implementar las funcionalidades del sistema para un usuario estudiante.

Fecha de inicio / Fecha fin: 24/09/2018 al 08/10/2018

Diseño:

La primeras características implementadas para estos usuarios consistieron en la autenticación en el sistema y reconocimiento de su rol como estudiante. La autenticación se realizó usando una librería llamada Ion Auth basada en el framework CodeIgniter, la cual implementa la abstracción necesaria para el manejo de roles y sesiones dentro de la aplicación. Esta a su vez permite el registro de estudiantes dentro del sistema. Luego de realizar el login usando la cédula de identidad y un token que será dado por el personal administrativo, al estudiante se le presentan en pantalla dos funciones en forma de botones, una de ellas corresponde al cierre de la sesión actual y otra para el inicio del proceso de inscripción. Cada usuario registrado en el sistema está asociado a un rol en particular, en este proyecto existen dos de ellos, el rol de administrador y el de estudiante, una vez confirmado que el usuario que hizo login posee el rol de estudiante se le muestra y da acceso a realizar las acciones adecuadas.

Una vez autenticado en el sistema, el estudiante puede realizar la inscripción de las materias en el semestre a cursar presionando un botón que indica el inicio de este procedimiento. Esta operación fue manejada a través de un formulario el cual lista las materias agrupadas por semestre que están disponibles para ser inscritas, cada materia de este formulario muestra el nombre, la cantidad de créditos que ocupa, el profesor que la dicta y el horario asociado a ella. Se implementó el formulario con la intención de que el

estudiante pueda seleccionar múltiples elementos, luego la acción de hacer click en un botón al final del formulario termina la inscripción para finalizar este proceso.

Culminado el paso anterior, el estudiante puede generar un comprobante de inscripción, que indica las materias inscritas por un estudiante en particular, así que, una vez culminado el proceso de inscripción por parte de un estudiante, este debe de poder generar este reporte, se planteó la creación de un botón que genere ese documento. Además, posterior a haber culminado su proceso de inscripción, el estudiante podrás ver las materias inscritas en pantalla cada vez que se autentique en el sistema.

Los procesos mencionados anteriormente necesitan de restricciones para que su funcionamiento sea llevada a cabo de manera correcta, tal es el caso de restricciones de autenticación, estas manejadas por la librería seleccionada para el manejo de sesiones, a su vez están las restricciones necesarias durante el proceso de inscripción (que forman parte de los requisitos del proyecto), para esto se implementaron estas funcionalidades de acuerdo a:

1. Un estudiante no puede inscribir más de una vez la misma materia.
2. Un estudiante no puede inscribir materias en horarios que coincidan.
3. Un estudiante no podrá inscribirse si la suma de créditos es superior o inferior a los límites establecidos por Escuela de Estudios Políticos y Administrativos.
4. Una vez finalizado el proceso de inscripción el estudiante no puede inscribirse una vez más.

Codificación:

Para la implementación del diseño discutido, en primer lugar fue necesario modificar el código del método de login (proporcionado por Ion Auth) para acceder a la aplicación, ya que como se ha mencionado, se deben de mostrar opciones diferentes en pantalla dependiendo del rol del usuario que está ingresando al sistema. Esta modificación se aprecia en la figura 8.

```
/**
 * Log the user in
 */
public function login()
{

    if ($this->ion_auth->logged_in()) {
        if ($this->ion_auth->is_admin()) {
            redirect('/inscripciones', 'refresh');
        } else {
            redirect('/inscripcion', 'refresh');
        }
    }

    $this->data['title'] = $this->lang->line('login_heading');

    $this->form_validation->set_rules('identity', str_replace(':', '', $this->lang->line('login_identity_label')),
'required');
    $this->form_validation->set_rules('password', str_replace(':', '', $this->lang->line('login_password_label')),
'required');

    if ($this->form_validation->run() === true) {
        $remember = (bool) $this->input->post('remember');

        if ($this->ion_auth->login($this->input->post('identity'), $this->input->post('password'), $remember)) {
            if ($this->ion_auth->is_admin()) {
                redirect('/inscripciones', 'refresh');
            } else {
                redirect('/inscripcion', 'refresh');
            }
        } else {
            $this->session->set_flashdata('message', $this->ion_auth->errors());
            redirect('auth/login', 'refresh');
        }
    } else {

        $this->data['message'] = (validation_errors()) ? validation_errors() : $this->session->flashdata('message');

        $this->_render_page('auth/login', $this->data);
    }
}
```

Figura 8 Método login

A continuación correspondía dar paso a la realización del código que generaría el formulario con las materias disponibles para inscribir. Tomando en cuenta que un estudiante no se puede inscribir más de una vez, se agregó una condición que impediría que un estudiante que ya se ha inscrito lo haga de nuevo, el método que realiza esta tarea se puede ver en la figura 9.

```
/**
 * Genera el formulario con las materias a inscribir
 *
 * @param string $message, corresponde, de existir, al mensaje de error generado por el formulario.
 */
function new ($message = '') {

    for ($i = 1; $i < 11; $i++) {
        $data['materias_por_semestre'][$i] = array();
    }

    if ($this->ion_auth->user()->row()->inscrito != 0) {
        return show_error('Usted no puede accesar a esta página.');
    }

    $this->load->helper('form');

    $data['materias_disponibles'] = $this->MateriasDisponibles_model->get();

    foreach ($data['materias_disponibles'] as $key => $value) {

        if ($value['cupos'] >= $value['cupos_maximos']) {
            unset($data['materias_disponibles'][$key]);
            continue;
        }

        $data['materias_disponibles'][$key]['horario'] = $this->horarios_by_materia_disponible($value['id']);

        array_push($data['materias_por_semestre'][$value['semestre']], $data['materias_disponibles'][$key]);

    }

    $data['cantidad'] = count($data['materias_disponibles']);
    $data['message'] = $message;
    $this->load->view('layouts/header');
    $this->load->view('inscripciones/new', $data);
    $this->load->view('layouts/footer');
}
```

Figura 9 Método new para la generación de formulario de inscripción

Una vez el estudiante haya seleccionado las materias que desea inscribir en su semestre, él podrá hacer click en un botón que representaría la culminación de su

inscripción, este botón se encarga de manejar las restricciones mencionadas anteriormente y validar los datos que se están enviando desde el formulario, en la figura 10 se puede apreciar esa funcionalidad, la cual verifica los datos enviados por el formulario y de ser no válidos se muestra un mensaje de error, caso contrario se procede a hacer la inserción en la base de datos y a marcar al usuario como inscrito.

```php
/**
 * Realiza la inscripción del estudiante en el sistema.
 */
public function create()
{
    if (empty($this->input->post()) || $this->register_validation($this->input->post())) {
        $message = 'Inscripción inválida, por favor asegúrese de que haya seleccionado al menos una opción, de que la
cantidad de créditos no exceda 20 o sea inferior a 9, o de que no haya un solapamiento de horarios';
        $this->new($message);
    } else {
        foreach ($this->input->post() as $key => $value) {

            $array = array(
                'id_materia_disponible' => $value,
                'id_user' => $this->ion_auth->user()->row()->id,
                'created_at' => date('Y-m-d H:i:s', time()),
                'updated_at' => date('Y-m-d H:i:s', time()),
            );

            $this->Cursa_model->set($array);

            $this->MateriasDisponibles_model->add_cupo($value);
        }
        $this->Ion_auth_model->set_inscrito($this->ion_auth->user()->row()->id);

        $this->send_register_document();

        redirect(site_url('inscripcion'), 'refresh');
    }
}
```

Figura 10 Método create para el registro de una inscripción

Para la validación de las restricciones se ejecuta un método con el nombre de “register_validation” (figura 11) que se encarga de verificar que el límite de créditos no pase los márgenes establecidos, que no se repitan materias o que los horarios en algunas de ellas se solapen.

```php
/**
 * Valida el formulario de inscripción.
 *
 * @param array $post, corresponde a $this->input->post()
 */
public function register_validation($post)
{
    $horarios['Lunes'] =
    $horarios['Martes'] =
    $horarios['Miércoles'] =
    $horarios['Jueves'] =
    $horarios['Viernes'] =
    $horarios['Sábado'] =
    $horarios['Domingo'] = array();

    $lim = 0;
    $set = array();
    $repeated = false;
    foreach ($post as $key => $value) {

        $md = $this->MateriasDisponibles_model->get($value);
        $horariosMD = $this->MateriaHorario_model->get($value);

        $lim += $md['creditos'];

        if (!array_key_exists($md['codigo'], $set)) {
            $set[$md['codigo']] = '';
        } else {
            $repeated = true;
            break;
        }

        foreach ($horariosMD as $value) {
            array_push($horarios[$value['dia']], array($value['hora_comienzo'], $value['hora_final']));
        }

    }
    return $repeated || $lim > 20 || $lim < 9 || $this->schedules_overlap($horarios);
}
```

Figura 11 Método register_validation para la validación de una inscripción

Finalizado el proceso anterior se implementó el método constancia_download (figura 12) que generaría la constancia de inscripción descargable a cada estudiante, esto se hizo con ayuda de la librería Dompdf que permite la creación de archivos PDF basado en un marco HTML.

```php
/**
 * Genera la constancia de inscripción descargable del estudiante.
 */

public function constancia_download()
{
    $html = $this->constancia();
    $dompdf = new Dompdf();
    $dompdf->loadHtml($html);
    $dompdf->setPaper('A4', 'Portrait');
    $dompdf->render();
    $dompdf->stream("comprobante-" . $this->ion_auth->user()->row()->ci . ".pdf");

}
```

Figura 12: Método constacia_download para la generación del comprobante de inscripción

Pruebas:

Para poder realizar las pruebas de las distintas funcionalidades implementadas en la presente iteración se planteó el uso de pruebas de aceptación que abarcaran el trabajo realizado.

Nro	Descripción del caso de prueba	Resultado esperado	Resultado Obtenido	Motivo de la falla
1	No aceptar que un estudiante se pueda inscribir luego de ya haber realizado el proceso de inscripción.	No mostrar la funcionalidad de inscribir a un estudiante ya inscrito.	Luego del estudiante haber realizado el proceso de inscripción no puede acceder de	Luego de obtener los datos del estudiante no se validaba correctamente si el estudiante

			nuevo a realizar este proceso.	estaba inscrito o no
2	Aceptar inscripción válida.	Mostrar pantalla con botón para generar comprobante de inscripción.	Luego del estudiante haber finalizado el proceso de inscripción se le muestra en pantalla el botón para generar el comprobante de inscripción.	En la vista no se contaba con la variable de control para desplegar el botón de generar constancia
3	No aceptar inscripción con límites de créditos mayor fuera del rango permitido.	Un mensaje de error en pantalla indicando el problema.	Mensaje de error indicando los errores posibles con la inscripción del estudiante.	Inicialmente la validación de para el límite inferior de créditos no se estaba correctamente realizada, y permitía inscribir muy pocos créditos
4	No aceptar inscripción con horarios que se solapen.	Un mensaje de error en pantalla indicando el problema.	Mensaje de error indicando los errores posibles con la inscripción	Casos borde en los intervalos de las horas causaban que no

			del estudiante.	se pudieran inscribir horarios que no estaban solapados
5	No aceptar inscripción de materias iguales dictadas por distintos horarios o profesores.	Un mensaje de error en pantalla indicando el problema.	Mensaje de error indicando los errores posibles con la inscripción del estudiante.	La validación usaba el identificador de la materia disponible en lugar del de la materia
6	Aceptar generación de comprobante de inscripción	Descarga de archivo al computador del estudiante.	Descarga del comprobante de inscripción con las materias inscritas por el estudiante.	La librería para generar el comprobante en PDF no se encontraba configurada correctamente dentro del proyecto

Iteración 3

Descripción: Implementar las funcionalidades del sistema para un usuario administrador.

Fecha de inicio / Fecha fin: 08/10/2018 al 22/10/2018

Diseño:

Los usuarios administradores cuentan con la mayor parte de funcionalidades implementadas ya que es mediante este rol que se gestionan los elementos necesarios para que se dé el proceso de inscripción, a continuación se exponen dichas funcionalidades.

Como funcionalidad básica se tiene la autenticación en el sistema y reconocer al usuario como un administrador, similar a la realizada en iteraciones anteriores.

Respecto a la creación de estudiantes, se permite poblar la base de datos con los estudiantes que van a poder ingresar al sistema. Los campos requeridos para crear un estudiante son nombres, apellidos, cédula, email, créditos aprobados y token. La vía para crear un estudiante es mediante un formulario en la aplicación, este formulario permite manualmente cargar la información de un estudiante a la vez, luego los datos se validan en la lógica del controlador. El código usado para la creación manual de estudiantes es el siguiente (figura 13):

```php
public function create_user()
    {
        /** authentication validation */

        $tables = $this->config->item('tables', 'ion_auth');
        $identity_column = $this->config->item('identity', 'ion_auth');
        $this->data['identity_column'] = $identity_column;

        // validate form input
        $this->form_validation->set_rules('first_name',
            $this->lang->line('create_user_validation_fname_label'), 'trim|required');

        $this->form_validation->set_rules('last_name',
            $this->lang->line('create_user_validation_lname_label'), 'trim|required');

        .
        . // setting validation rules for the rest of the form fields
        .

        if ($this->form_validation->run() === true) {
            $email = strtolower($this->input->post('email'));
            $identity = $this->input->post('identity');
            $password = $this->input->post('password');

            $additional_data = array(
                'first_name' => $this->input->post('first_name'),
                'last_name' => $this->input->post('last_name'),
                'creditos_aprobados' => $this->input->post('creditos_aprobados'),
            );
        }
        if ($this->form_validation->run() === true && $this->ion_auth->register($identity, $password, $email,
$additional_data)) {
            // check to see if we are creating the user
            // redirect them back to the admin page
            $this->session->set_flashdata('message', $this->ion_auth->messages());
            redirect("estudiantes", 'refresh');
        } else {
            // display the create user form
            // set the flash data error message if there is one
            $this->data['message'] = (validation_errors() ? validation_errors() : ($this->ion_auth->errors() ?
$this->ion_auth->errors() : $this->session->flashdata('message')));

            $this->data['first_name'] = array(
                'name' => 'first_name',
                'id' => 'first_name',
                'type' => 'text',
                'value' => $this->form_validation->set_value('first_name'),
                'class' => 'form-control',
            );
          /** Create the rest of the form fields attributes' arrays */
          /** Render templates */
```

Figura 13 Método create_user para la creación de usuarios

Luego de tener creados los usuarios, estos se listan con sus datos básicos nombre, apellido, cédula y si está inscrito o no. Estos datos pueden ser editados en caso de ser necesario, a esta funcionalidad se le accede mediante un botón de editar estudiante que se muestra en una de las columnas de la tabla en la que se listan los estudiantes. Esta vista muestra los datos del estudiante a editar en un formulario que luego puede ser guardado, a

continuación se muestra el método (figura 14) del controlador “Auth.php” encargado de procesar la edición del usuario en la aplicación.

```php
/**
 * Edit a user
 *
 * @param int|string $id
 */
public function edit_user($id)
{
    $this->data['title'] = $this->lang->line('edit_user_heading');

    if (!$this->ion_auth->logged_in() ||
        (!$this->ion_auth->is_admin() && !($this->ion_auth->user()->row()->id == $id))) {
        redirect('auth', 'refresh');
    }

    $user = $this->ion_auth->user($id)->row();
    $groups = $this->ion_auth->groups()->result_array();
    $currentGroups = $this->ion_auth->get_users_groups($id)->result();

    $this->form_validation->set_rules('first_name',
        $this->lang->line('edit_user_validation_fname_label'), 'trim|required');

    $this->form_validation->set_rules('last_name',
        $this->lang->line('edit_user_validation_lname_label'), 'trim|required');
    $this->form_validation->set_rules('email', 'email', 'trim|required');

    if (isset($_POST) && !empty($_POST)) {

        if ($this->_valid_csrf_nonce() === false || $id != $this->input->post('id')) {
            show_error($this->lang->line('error_csrf'));
        }

        if ($this->input->post('password')) {
            $this->form_validation->set_rules('password', $this->lang->line('edit_user_validation_password_label'),
'required|min_length[' . $this->config->item('min_password_length', 'ion_auth') . ']|max_length[' . $this->config-
>item('max_password_length', 'ion_auth') . ']|matches[password_confirm]');
            $this->form_validation->set_rules('password_confirm', $this->lang-
>line('edit_user_validation_password_confirm_label'), 'required');
        }

        if ($this->form_validation->run() === true) {
            $data = array(
                'first_name' => $this->input->post('first_name'),
                'last_name' => $this->input->post('last_name'),
                'email' => $this->input->post('email'),
            );

            if ($this->input->post('password')) {
                $data['password'] = $this->input->post('password');
            }

            if ($this->ion_auth->update($user->id, $data)) {

                $this->session->set_flashdata('message', $this->ion_auth->messages());
                $this->redirectUser();

            } else {
                $this->session->set_flashdata('message', $this->ion_auth->errors());
                $this->redirectUser();

            }

        }
    }

    $this->data['csrf'] = $this->_get_csrf_nonce();

    $this->data['message'] = (validation_errors() ? validation_errors() : ($this->ion_auth->errors() ? $this->ion_auth-
>errors() : $this->session->flashdata('message')));

    $this->data['user'] = $user;
}
```

Figura 14: Método edit_user para la creación de usuarios

Los administradores pueden también realizar el proceso de inscripción para un estudiante, este botón se muestra en la fila de cada estudiante y al presionarlo se muestra la lista de materias disponibles por semestre para indicar cuáles se les quiere inscribir al estudiante, este es el método (figura 15) usado para inscribir un estudiantes, ubicado en el controlador Inscripciones.php.

```
public function create($id)
{
    if (empty($this->input->post())) {
        $this->session->set_flashdata('message',
                'Inscripción inválida, debes seleccionar al menos una materia.');
        redirect(current_url());
    } else {
        foreach ($this->input->post() as $key => $value) {
            $array = array(
                'id_materia_disponible' => $value,
                'id_user' => $id,
                'created_at' => date('Y-m-d H:i:s', time()),
                'updated_at' => date('Y-m-d H:i:s', time()),
            );
            $this->Cursa_model->set($array);
            $this->MateriasDisponibles_model->add_cupo($value);
        }
        $this->Ion_auth_model->set_inscrito($id);
        redirect(site_url('inscripciones'), 'refresh');
    }
}
```

Figura 15 Método create para inscribir estudiantes

En relación a la funcionalidad para exportar los estudiantes, los administradores pueden generar un archivo con formato .xlsx presionando un botón, ahí se lista la información básica de todos los estudiantes registrados en el sistema, indicando si estos realizaron su inscripción o no. Básicamente el controlador obtiene todos los registros de estudiantes de la base de datos, recopila y procesa los campos que van a ser usados en el archivo, arma una lista y esa lista se usa para finalmente generar el archivo con el uso de la librería PhpSpreadsheet (figura 16).

```
public function xls_estudiantes()
{

    $data = $this->ion_auth->users('members')->result();

    $list = array("Apellidos, Nombres, CI, Inscrito");

    foreach ($data as $key => $value) {

        $arr = array(
            $value->last_name,
            $value->first_name,
            $value->ci,
            ($value->inscrito == 1 ? 'Si' : 'No'));

        $string = implode(",", $arr);

        array_push($list, quitar_acentos($string));

    }

    create_xls($list, 'estudiantes');
}
```

Figura 16: Método xls_estudiantes

El método xls_estudiantes es encargado de recopilar los registros de los estudiantes, darles estructura y enviarlos a ser procesados para generar el archivo mediante la función

create_xls (figura 17) , la cual establece los parámetros y cabeceras (headers) para realizar la descarga del archivo, y toma los datos pasados a ella para llenar las celdas de las hojas.

```php
function create_xls($xls_data, $filename, $path = "php://output")
{
    if ($path = "php://output") {
        header('Content-Type: application/vnd.openxmlformats-officedocument.spreadsheetml.sheet');
        header('Content-Disposition: attachment;filename="' . $filename . '.xlsx"');
        header('Cache-Control: max-age=0'); // always download from the server, file is not cache
    }

    $spreadsheet = new Spreadsheet();
    foreach ($xls_data as $row_index => $value) {
        $row = explode(",", $value);
        foreach ($row as $column_index => $value) {
            $spreadsheet
                ->getActiveSheet()
                ->setCellValueByColumnAndRow($column_index + 1, $row_index + 1, $value);
        }
    }

    $writer = new Xlsx($spreadsheet);
    // start stream download or store in the server depending on path
    $writer->save($path, 'xls');

    // free spreadsheet usage
    $spreadsheet->disconnectWorksheets();
    unset($spreadsheet);
}
```

Figura 17: Método create_xls

Relacionado al listado de inscripciones, los administradores pueden ver las inscripciones que se han hecho en el sistema, en principio es la lista de estudiantes, mostrando aquellos que han sido inscritos, esto se hace en la función index del controlador Inscripciones.php (figura 18).

```php
/**
 * Muestra la vista de los estudiantes inscritos en el sistema
 */
public function index()
{
    $data['estudiantes'] = $this->Cursa_model->signed_up_students();

    $this->load->view('admin/layouts/header');
    $this->load->view('admin/estudiantes/inscripciones', $data);
    $this->load->view('admin/layouts/footer');
}
```

Figura 18 Método index para listar estudiantes inscritos

La función signed_up_students (figura 19) del modelo cursa, se encarga de hacer la búsqueda en la base de datos y retornar la información.

```php
public function signed_up_students()
{
    // Lists all the signed up students
    $this->db->select('c.id_user, u.ci, u.first_name, u.last_name, u.creditos_aprobados');
    $this->db->from($this->table . ' as c'); // table = 'cursa'
    $this->db->join('users as u', 'u.id = c.id_user', 'left');
    $this->db->group_by('c.id_user');
    $users = $this->db->get()->result_array();

    return $users;
}
```

Figura 19 Método signed_up_students para obtener los estudiantes inscritos

Adicionalmente, se puede acceder a las materias inscritas de cada estudiante, que se muestran en una vista aparte desde la cual, se puede también modificar la inscripción. Esto se hace en el método detail del controlador Inscripciones.php (figura 20)

```php
public function detail($student_id, $message = '')
{

    $data['message'] = $message;
    // key-value arrays to create the form inputs in the template
    $submit_eliminar = array(
        'name' => 'eliminar',
        'type' => 'submit',
        'value' => 'Eliminar',
        'class' => 'btn btn-danger',
    );
    $submit_agregar = array(
        'name' => 'agregar',
        'type' => 'submit',
        'value' => 'Agregar',
        'class' => 'btn btn-primary',
    );
    $data['submit_eliminar'] = $submit_eliminar;
    $data['submit_agregar'] = $submit_agregar;
    // Gets students signed in materias
    $data['student'] = $this->Cursa_model->get_estudiante_materias($student_id);

    if (isset($data['student'])) {
        foreach ($data['student'] as $key => $value) {
            // attaching the corresponding 'horario' to each 'materia'
            $data['student'][$key]['horario'] =
            $this->horarios_by_materia_disponible(
                $data['student'][$key]['id_materia_disponible']
            );

            // Retrieving the list of Materias Disponibles
            $md_for_student = $this->MateriasDisponibles_model->get();
            if (!empty($md_for_student)) {
                $materias_disponibles = array();
                foreach ($md_for_student as $temp_md) {
                    // format the text to be shown in the dropdown
                    $materias_disponibles[$temp_md['id']] =
                        "{$temp_md['nombre']} {$temp_md['codigo']} {$temp_md['profesor']}";
                }
                $data['materias_disponibles'] = $materias_disponibles;
            } else {
                $data['materias_disponibles'] = null;
            }
        }
    }
    $this->load->view('admin/layouts/header');
    $this->load->view('admin/inscripciones/edit', $data);
    $this->load->view('admin/layouts/footer');
}
```

Figura 20 Método detail, para traer la información de la inscripción de un estudiante

Cada materia inscrita que se lista tiene la opción de ser eliminada de la inscripción, que requiere básicamente eliminar un registro de la tabla cursa y realizar actualizaciones de cupos para dicha materia, lo cual se logra con el siguiente método (figura 21)

```
public function eliminar_inscripcion($id_user)
{
    if (!empty($id_user) && !empty($this->input->post('materia_d'))) {
        $this->Cursa_model->delete($id_user, $this->input->post('materia_d'));
        $this->MateriasDisponibles_model->substract_cupo($this->input->post('materia_d'));
        redirect('estudiantes/' . $id_user);
    }
}
```

Figura 21 Método eliminar_inscripcion para eliminar una materia de la inscripción de un estudiante

Si se quiere agregar una materia a la inscripción, hay un dropdown que tiene la lista de las materias disponibles no inscritas de ese estudiante y un botón para agregarla a la lista de materias inscritas del estudiantes, siendo mediante estas funciones de agregar materias y eliminar materia que se modifica la inscripción. Se implementó el siguiente método (figura 22) para agregar materias a la inscripción de un estudiante.

```
public function agregar_inscripcion($id_user)
{
    if (!empty($id_user) && !empty($this->input->post('materia'))) {
        $mat_added = $this->add($id_user, $this->input->post('materia'));
        if ($mat_added) {
            redirect('estudiantes/' . $id_user);
        } else {
            $this->detail($id_user, 'Lo sentimos la materia ya está inscrita');
        }
    }
}
```

Figura 22 Método agregar_inscripcion para agregar una materia a la inscripción de un estudiante

Respecto a las funcionalidades relacionada a las inscripciones también se pueden generar una lista de estudiante inscritos de una determinada materia, y se contará adicionalmente con una casilla de verificación que da la opción de adjuntar un correo al profesor de la materia con dicha nómina, la funcionalidad del envío del correo fue implementada en una iteración posterior.

Parte del proceso de inscripción involucra la gestión de las materias que dicta la Escuela y los horarios en los cuales se imparten clases. Los administradores pueden gestionar las materias en una sección propia para ello, tanto agregándolas como eliminándolas mediante formularios en la página web, la información asociada es la siguiente, código, nombre, créditos y el semestre al que pertenece. A nivel de implementación se compone de las operaciones CRUD (Create Read Update Delete) básicas.

Los horarios tienen su propia sección para ser manejados, y se gestionan agregándose o eliminándose mediante formularios en la aplicación. A nivel de implementación se compone de las operaciones CRUD (Create Read Update Delete) básicas.

La manera en que las materias y los horarios se relacionan es mediante el modelo de materias disponibles. Las materias disponibles tienen asociadas una materia, uno o más horarios, el número de cupos, cantidad de estudiantes inscritos y el profesor que la dicta. Se pueden modificar los datos de una materia disponible en un formulario, siendo uno de esos campos un dropdown en el que se selecciona la materia que se le quiere asociar. Los

horarios se modifican mediante otra opción la cual permite asociarlos y desvincularlos de la materia disponible, usando el id de la materia disponible, se puede apreciar en la figura 23.

```php
/**
 * Muestra los horarios de una materia disponible
 *
 * @param string|int $id, id de la materia disponible.
 */
public function horarios($id)
{
    $this->load->helper('form');

    $data['materia_horarios'] = $this->MateriaHorario_model->get($id);
    $data['horarios'] = $this->Horarios_model->get();
    $data['id'] = $id;

    $this->load->view('admin/layouts/header');
    $this->load->view('admin/materias_disponibles/horarios', $data);
    $this->load->view('admin/layouts/footer');
}
```

Figura 23 Método horarios para retornar horarios de una materia disponible

Pruebas:

Para poder realizar las pruebas de las distintas funcionalidades implementadas en la presente iteración se planteó el uso de pruebas de aceptación que abarcaran el trabajo realizado.

Nro	Descripción del caso de prueba	Resultado esperado	Resultado Obtenido	Motivo de la falla

1	Crear estudiante: lograr la creación de un estudiante manualmente mediante el formulario en la aplicación web.	Luego de introducir todos los campos en el formulario y guardar, se debe crear la fila en la base de datos y mostrarse en la lista de estudiantes.	Los datos del estudiante se guardaron en la base de datos y se muestran en la lista de estudiantes.	Falla de validaciones en las plantillas de campos requeridos hacía que se hiciera submit null de campos que en la BD son por restricción not null
2	Guardar datos de estudiante: exigir que se llenen todos los campos del formulario de estudiante para poder guardarse en la base de datos.	Para cada uno de los campos, si se deja uno sin llenar no se deben guardar los datos en la base de datos y se debe mostrar un mensaje indicando los campos faltantes.	De faltar al menos un campo en el formulario se valida al momento de enviar los campos que faltan, se notifica al usuario y se omite el guardado de datos a la base de datos.	No validar campos requeridos en la vista ocasionaba que se le mostrara un error del framework poco amigable al usuario
3	Editar estudiante: al acceder al formulario de un estudiante al cual se está editando, el	Al editar un estudiante el formulario viene se llena con los datos de éste por defecto.	Se muestra el formulario con los datos del estudiante por defecto.	La forma de la variable que contenía los campos del estudiante no coincidía con cómo estaba

	formulario debe venir lleno por defecto con los datos del estudiante.			siendo usada en la plantilla
4	Inscribir estudiante como administrador: por privilegios de administrador no implementar restricciones de créditos sobre la inscripción.	Al intentar inscribir a un estudiante con materias que sumen créditos fuera de los límites de la escuela, la inscripción debe realizarse.	La inscripción se guarda para casos en que se excedan los límites inferiores o superiores de créditos.	La validación de límite de créditos para estudiantes se estaba manteniendo para los administradores, se removió la validación para solucionar esto
5	Inscribir estudiante como admin: actualizar el número de cupos de las materias inscritas.	Luego de guardar la inscripción, las materias inscritas deben disminuir su cantidad de cupos disponibles, si se remueve una materia de una inscripción, los cupos disponibles deben aumentar.	Se actualizan correctamente los cupos de las materias.	La validación para incrementar los cupos cuando se elimina una materia de la inscripción del estudiante no se estaba realizando
6	Inscribir estudiante como	Luego de inscribir un estudiante, la opción	Se deshabilita la opción de inscribir a	

	admin: deshabilitar la opción de inscribir a aquellos estudiantes que ya han sido inscritos.	para inscribirlo debe estar deshabilitada.	los estudiantes ya inscritos.	
7	Exportar estudiantes: generar e iniciar descarga de un archivo con el listado de estudiantes.	Al usar la opción de exportar estudiantes, debe iniciar el proceso de descarga del archivo con el formato y los datos correctos.	El documento se descarga exitosamente con los datos y el formatos correcto.	Fallas en la configuración de la librería y la estructura de la información que se muestra en el archivo era incorrecta
8	Listado de inscripciones: mostrar solo los estudiantes inscritos.	Al ver la lista de inscripciones deben filtrarse de tal modo que no se muestren los estudiantes no inscritos.	En el listado de inscripciones se omiten los estudiantes no inscritos.	En algunos casos editar una inscripción hacia que el estudiante dejara de mostrarse en la tabla, se añadieron las validaciones correspondientes
9	Ver detalle de	Al ir al detalle de una	Se muestran	Agregar una

	inscripción.	inscripción se deben mostrar las materias disponibles y una opción para agregarlas a la inscripción. Se debe mostrar las lista de materias inscritas y una opción para eliminarlas de la inscripción.	correctamente las materias disponibles e inscritas con sus opciones para agregar y eliminar correspondientemente.	materia a una inscripción hacía que se mostrara duplicada en el detalle de inscripción, se validó para eliminar el duplicado extra
10	Guardar datos de materias: exigir que se llenen los campos obligatorios en el formulario para guardar la materia.	Para todos los campos requeridos si alguno falta por llenar, no se deben guardar los datos y se debe mostrar un mensaje al usuario indicando los campos faltantes.	De faltar campos requeridos, se muestra un mensaje de error al usuario y se omite el proceso de guardado.	
11	Guardar datos de horario: exigir que se llenen los campos obligatorios en el formulario para guardar el horario.	Para todos los campos requeridos si alguno falta por llenar, no se deben guardar los datos y se debe mostrar un mensaje al usuario indicando los campos faltantes.	De faltar campos requeridos, se muestra un mensaje de error al usuario y se omite el proceso de guardado.	El formato de horas para guardar los horarios no se estaba manejando correctamente y ocasionaba problemas al guardarlo

12	Guardar datos de materia disponible: exigir que se llenen los campos obligatorios en el formulario para guardar la materia disponible.	Para todos los campos requeridos si alguno falta por llenar, no se deben guardar los datos y se debe mostrar un mensaje al usuario indicando los campos faltantes.	De faltar campos requeridos, se muestra un mensaje de error al usuario y se omite el proceso de guardado.	El dropdown de materias estaba omitiendo algunas materias por una validación incorrecta
13	Ver horarios de materia disponible.	En la vista de ver horarios se deben mostrar los horarios y una opción para asociarlos a la materia disponible. Se debe mostrar las lista de horarios asociados y una opción para desvincularlos de la materia disponible.	Se muestran correctamente los horarios asociados y los disponibles con sus opciones para agregar y eliminar correspondientemente.	Una validación innecesaria ocasionaba que no se pudiera asociar más de un horario a una materia disponible

Iteración 4

Descripción: Corregir errores de versión beta.

Fecha de inicio / Fecha fin: 22/10/2018 al 05/11/2018

Diseño:

Esta iteración aunque no planteada en un primer momento, fue llevada a cabo ya que se consideró necesaria luego de haber usado la aplicación por primera vez en un ambiente de producción, esto durante el proceso de inscripción II-2018 de la Escuela de Estudios Políticos y Administrativos.

Durante este proceso la aplicación presentó un bug que la presente iteración plantea resolver. Este corresponde a una característica particular de la materia Seminario de esta escuela, un estudiante puede inscribir esta materia varias veces en un mismo semestre, esto debido a políticas propias de la escuela. En un primer lugar la aplicación solo estaba preparada para que en el momento de la inscripción un estudiante solo pueda inscribir una materia en particular (por ejemplo Matemática I) una sola vez, esto para evitar el problema de inscribir dos veces una misma materia en un semestre, se plantea agregar un nuevo campo a la tabla “materias” de la base de datos que indique si una materia en específico se puede inscribir varias veces, este valor se tomará en cuenta durante la validación de la inscripción para que este caso en específico pueda ser posible.

El personal de la Escuela de Estudios Políticos y Administrativo también pidió la inclusión de un mensaje de advertencia justo antes de que el estudiante termine el proceso de inscripción, esto para que el estudiante esté prevenido que confirmada su inscripción no hay vuelta atrás para cambiarla. El equipo de desarrollo decidió usar un mensaje de alerta

proporcinado por JavaScript que se muestre justo al hacer click en el boton de finalizar inscripcion y antes de enviar el formulario al servidor, para que los usuarios estén al tanto de esta información y decidan si continuar o no.

Codificación:

Para resolver el primer bug mencionando en la etapa de diseño se procedió a realizar una query en SQL para agregar el nuevo campo “repite” a la tabla “materias” (referir a figura 24). Este campo tendrá el valor de 1 para indiciar si es posible repetir la materia o 0 si no lo es.

```
ALTER TABLE materias ADD repite tinyint(1) NOT NULL DEFAULT 0
```

Figura 24: Agregar nuevo campo a tabla materias

Luego durante el momento de la inscripción, al ejecutar el método ‘register_validation’, se toma el valor repite por cada materia que el estudiante desee inscribir en un semestre y si es una materia que contenga ese campo en 1, se da por válida la materia seleccionada, se puede ver este cambio en la figura 25. Este proceso se repite para cada materia que el estudiante haya marcado para inscribir en el formulario de inscripción.

```php
if ($md['repite'] == 1 || !array_key_exists($md['codigo'], $set)) {
    $set[$md['codigo']] = '';
} else {
    $repeated = true;
    break;
}
```

Figura 25 Actualizar condición register_validation

Para agregar el mensaje de error se usó el componente confirm de JavaScript que genera una pequeña ventana antes de enviar un formulario, en esa ventana el usuario puede confirmar la solicitud del formulario o rechazarla (ejemplo en figura 26).

```php
form_open('inscripcion', 'onsubmit="return confirm(\'¿Estás seguro de que quieres inscribir las materias marcadas?\');"')
```

Figura 26 Ventana de confirmación

El resultado en el navegador se puede apreciar en la figura 27.

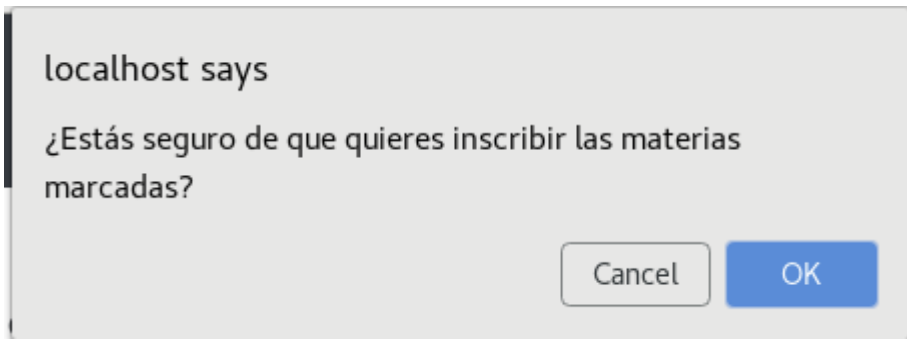

Figura 27 Ventana de confirmación

Pruebas:

Para poder realizar las pruebas de las distintas funcionalidades implementadas en la presente iteración se planteó el uso de pruebas de aceptación que abarcaran el trabajo realizado.

Número	Descripción del caso de prueba	Resultado esperado	Resultado Obtenido	Motivo de la falla
1	Aceptar inscripción de múltiples materias que tiene el campo repite en 1.	Cambio de pantalla al listado de materias inscritas por el estudiante.	Envío exitoso de formulario de inscripción al servidor y cambio de pantalla al listado de materias inscritas.	En la funcionalidad de administrador faltaba incorporar la validación
2	No aceptar inscripción de múltiples materias que tiene el campo repite en 0.	Volver al formulario de inscripción mostrando error en la solicitud de inscripción.	Carga del formulario de inscripción con error indicando el problema que ocurrió.	Para los estudiantes el error no estaba siendo manejado amigable para el usuario, se mostraba un

				error del framework
3	No aceptar envío del formulario de inscripción al hacer clic en cancelar en la ventana de confirmación de inscripción.	No hacer envío del formulario al servidor. Mostrar pantalla de materias a inscribir.	No se realiza el envío del formulario de inscripción.	

Iteración 5

Descripción: Implementar envío por correo electrónico.

Fecha de inicio / Fecha fin: 05/11/2018 al 19/11/2018

Diseño:

Otro de los requisitos establecidos para la realización de este proyecto es el envío por correo electrónico de información relevante al proceso de inscripción a los estudiantes y profesores. Se planteó la implementación de un módulo en el código de la aplicación que se encargue de realizar esta tarea, es importante permitir el envío de archivos adjuntos en esos correos electrónicos, por lo que tal módulo debe de estar preparado para ese cometido. El framework Codeigniter permite crear piezas de código independientes en la aplicación que pueden ser cargados a demanda en los controladores, estos tienen el nombre de helpers, es un archivo que posee una colección de funciones de una categoría particular.

Para la implementación de esta funcionalidad solo es necesario que exista una función que reciba como argumentos el asunto del correo electrónico, el mensaje, la dirección del archivo en el servidor a ser adjuntado, y el correo del destinatario.

Además para facilitar la codificación y evitar problemas en la recepción de los correos por parte de los destinatarios, se limitó la configuración de este módulo para correos emisores de Gmail.

Esta funcionalidad tiene que ser aplicada para el envío de comprobantes de inscripción a los estudiantes una vez culminada su inscripción y para el envío de las nóminas de estudiantes cursando una materia a los profesores que correspondan.

Codificación:

Para la implementación del diseño se creó un archivo encargado de realizar la tarea de este módulo dentro de la aplicación, este solo tendría una función de nombre “send_email” (figura 28) delegada de realizar el envío por correo electrónico y sería un helper del framework Codeigniter.

```php
<?php
/**
 *  Envía un correo con información sobre la inscripción del estudiante.
 *
 * @param string $subject, asunto del correo.
 * @param string $message, mensaje del correo.
 * @param string $attach, dirección del adjunto del correo.
 * @param string $to, destinatario del correo.
 */
function send_email($subject, $message, $attach, $to)
{

    $CI = &get_instance();
    $CI->load->library('email');
    $json_content = json_decode(file_get_contents("./config.json"), true);

    $config = array();
    $config['charset'] = 'utf-8';
    $config['useragent'] = 'Codeigniter';
    $config['protocol'] = "smtp";
    $config['mailtype'] = "html";
    $config['smtp_host'] = "ssl://smtp.gmail.com";
    $config['smtp_port'] = 465;
    $config['smtp_timeout'] = "5";
    $config['smtp_user'] = $json_content['email'];
    $config['smtp_pass'] = $json_content['email_contrasena'];
    $config['crlf'] = "\r\n";
    $config['newline'] = "\r\n";

    $config['wordwrap'] = true;

    $CI->email->initialize($config);

    $CI->email->from($json_content['email'], 'Escuela de Estudios Políticos y Administrativos');
    $CI->email->to($to);
    $CI->email->attach(site_url($attach));

    $CI->email->subject($subject);
    $CI->email->message($message);

    $CI->email->send();

}
```

Figura 28 Función send_email para el envío de correos electrónicos.

Una vez realizado lo anterior se hizo uso de esa función para el envío de un correo electrónico luego de que un estudiante terminase su proceso de inscripción, este tendría como archivo adjunto el documento de comprobante de inscripción. La función encargada de esta tarea en se puede observar en la figura 29.

```php
/**
 * Crea la constacia de inscripción y la envía por correo electrónico.
 */
public function send_register_document()
{
    $html = $this->constancia();
    $dompdf = new Dompdf();
    $dompdf->loadHtml($html);
    $dompdf->setPaper('A4', 'Portrait');
    $dompdf->render();
    $pdf_gen = $dompdf->output();
    $full_path = './assets/files/comprobante-' . $this->ion_auth->user()->row()->ci . '.pdf';
    if (!!file_put_contents($full_path, $pdf_gen)) {
        send_email('Comprobante de inscripcion', '', $full_path, $this->ion_auth->user()->row()->email);
    }
}
```

Figura 29 Método send_register_document para el envío de comprobante de inscripción por correo a los estudiantes.

Además se usó la función “send_email” para el envío de la nómina de estudiantes para los profesores de las materias o para cualquier otro correo electrónico que el usuario administrador escriba en un campo de texto, como se puede ver en la figura 30.

```php
if ($this->input->post('checkbox') || ($this->input->post('checkbox-profesor') && $md_data[0]['email_profesor'])) {
    $time = time();
    $path = './assets/files/' . $md_data[0]['codigo'] . '-estudiantes-' . $time . '.csv';
    create_xls($xls_data, $md_data[0]['codigo'] . ' ' . $md_data[0]['nombre'] . ' - estudiantes', $path);

    if ($this->input->post('email')) {
        send_email('Listado de estudiantes - ' . $md_data[0]['codigo'], '', $path, $this->input->post('email'));
    }

    if ($this->input->post('checkbox-profesor') && $md_data[0]['email_profesor']) {
        send_email('Listado de estudiantes - ' . $md_data[0]['codigo'], '', $path, $md_data[0]['email_profesor']);
    }
} else {
    create_xls($xls_data, $md_data[0]['codigo'] . ' ' . $md_data[0]['nombre'] . ' - estudiantes');
}
```

Figura 30 Extracto de código para el envío de correo electrónico a profesor u otro correo con nómina de estudiantes.

Pruebas:

Número	Descripción del caso de prueba	Resultado esperado	Resultado Obtenido	Motivo de la falla
1	Aceptar envío de correo electrónico con el comprobante de inscripción a los estudiantes que culminen el proceso de inscripción.	Enviar correo electrónico al estudiante con el comprobante de inscripción.	Envío exitoso con el archivo adjunto al correo del estudiante.	Una configuración errónea del SMTP ocasionaba que no se enviaran los correos desde el servidor
2	Aceptar envío de correo electrónico con nómina de estudiantes a profesores de las materias disponibles	Enviar correo electrónico al profesor con la nómina de estudiantes.	Envío exitoso con el archivo adjunto al correo del profesor.	En la búsqueda de profesores se usaba el id de la tabla materia en lugar de la tabla materias_disponibles, lo que ocasionaba un error en la BD

3	Aceptar envío de correo electrónico con nómina de estudiantes a cualquier correo eléctronico.	Enviar correo electrónico al correo con la nómina de estudiantes.	Envío exitoso con el archivo adjunto al correo.	Un error en el uso de las variables ocasionó que se enviara el correo a profesores de la materia en lugar del correo que se introdujo manualmente

Iteración 6

Descripción: Implementar plantillas HTML y CSS para mejorar la interfaz de usuario.

Fecha de inicio / Fecha fin: 19/11/2018 al 03/12/2018

Diseño:

Las interfaces fueron realizadas en conjunto con las funcionalidades correspondientes, las cuales estaremos mostrando, la primera siendo la vista de autenticación, cuenta con el logo de la Escuela en cuestión y un formulario con dos

campos, cédula y token, que sirven para autenticar a los usuarios tanto estudiantes como administradores (figura 31).

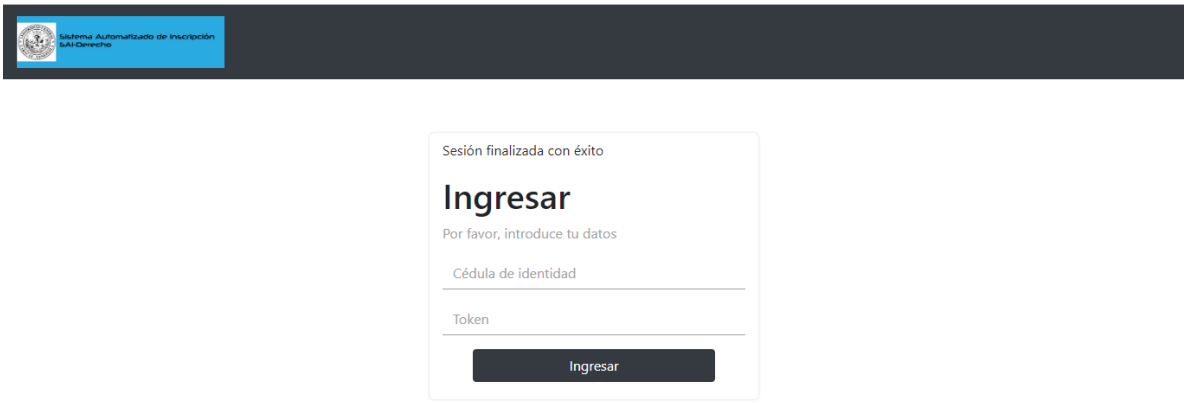

Figura 31 Interfaz de login

Las interfaces fueron construidas tomando en parte estilos de la biblioteca Bootstrap, y con otras definiciones de estilo propias, la mayoría de la información se muestra con tablas, que muestran las operaciones disponibles para cada registro según su información y los datos son cargados mediante formularios o archivos, las distintas secciones con las que cuenta la aplicación son mostradas en un menú en la parte superior de las páginas.

La vista de estudiantes del módulo de administrador muestra los campos apellidos, nombres, cédula de identidad, y si está inscrito o no, las opciones con las que cuenta son, editar estudiantes e inscribir estudiante, se puede ver en la figura 32.

Figura 32 Interfaz de listado de estudiantes

La vista de editar estudiantes cuenta con un formulario mediante el cual se envían los datos a modificar del estudiante (figura 33).

Figura 33 Vista de edición/creación de estudiante

La vista de inscribir estudiante como administrador cuenta con la lista de materias disponibles, estas se pueden seleccionar para luego guardar y realizar la inscripción del estudiante, figura (34).

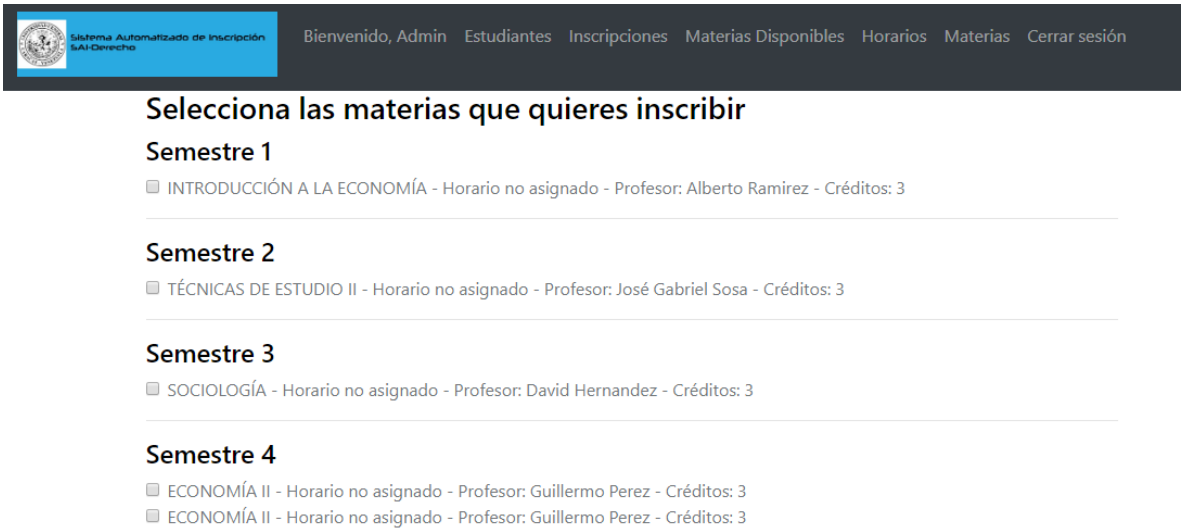

Figura 34 Vista de inscripción de estudiante como admin

En la sección de inscripciones tenemos en la vista principal la lista de estudiantes inscritos con información básica y la opción de editar sus inscripciones. Adicionalmente se encuentran las opciones para enviar la nómina de estudiantes general por correo a algún profesor, enviar la nómina de una materia en particular por correo, o exportar en un archivo, figura 35

Figura 35 Vista general de inscripciones

Con el botón de 'Ver materias inscritas' podemos ir a detalle de una inscripción como se ve en la figura 36.

Figura 36 Vista de detalle de una inscripción

En la sección de materias disponibles tenemos en la vista general el listado y las opciones para editar, crear, eliminar las materias disponibles así como sus horarios, figura 37.

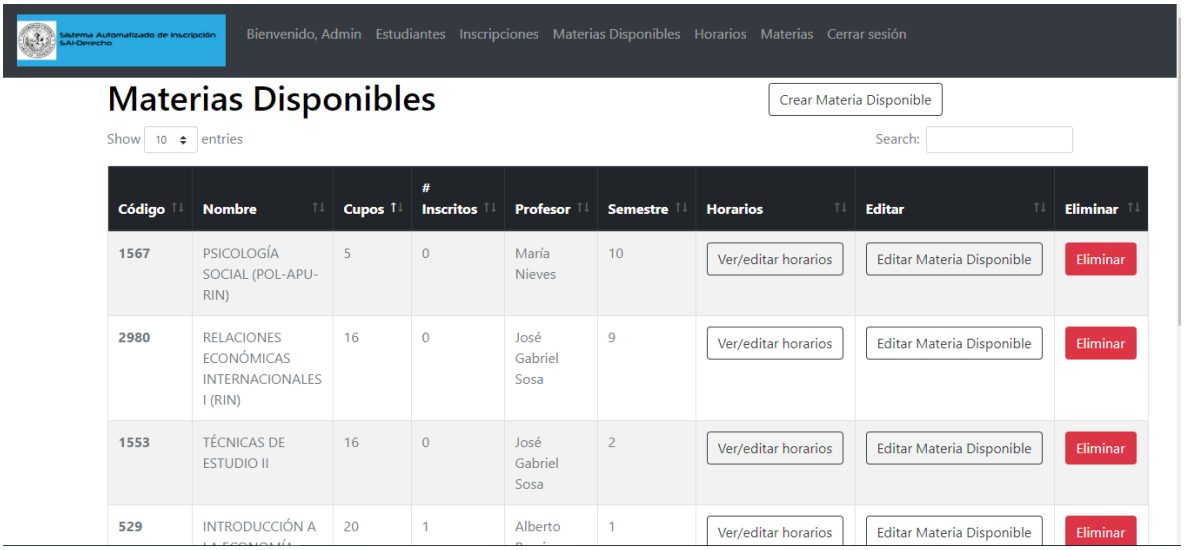

Figura 37 Vista general de materias disponibles

Tenemos acceso al formulario de los datos de las materias disponibles mediante los botones ‘Editar materia disponible’ y ‘Crear Materia Disponible’, figura 38.

Figura 38 Formulario de datos de materias disponibles

Y el accesos a los horarios de las materias disponible mediante el botón ‘Ver/editar horarios’, figura 39.

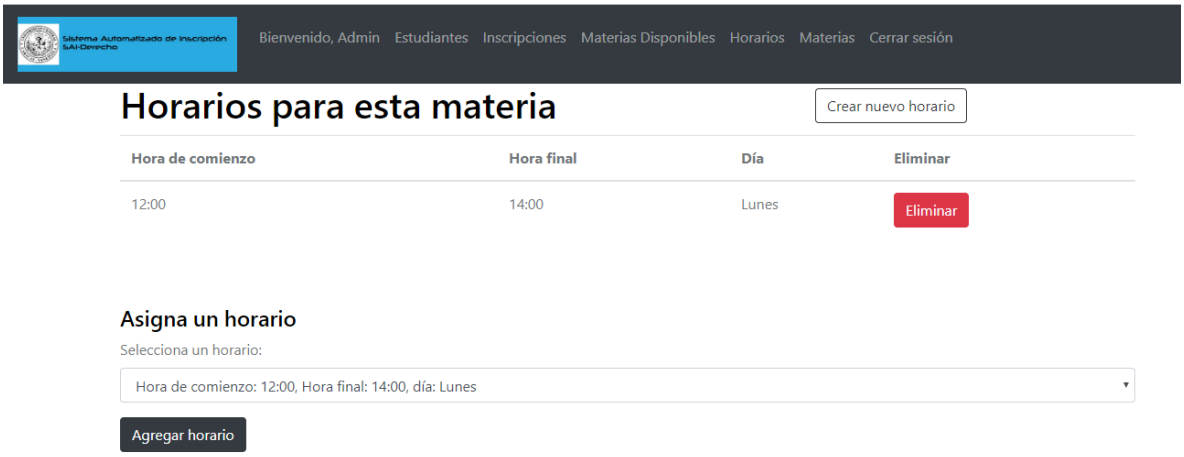

Figura 39 Vista de horarios de una materia disponible

Desde acá podemos ir directamente a la vista de creación de un horario mediante la opción ‘Crear nuevo horario’, figura 40.

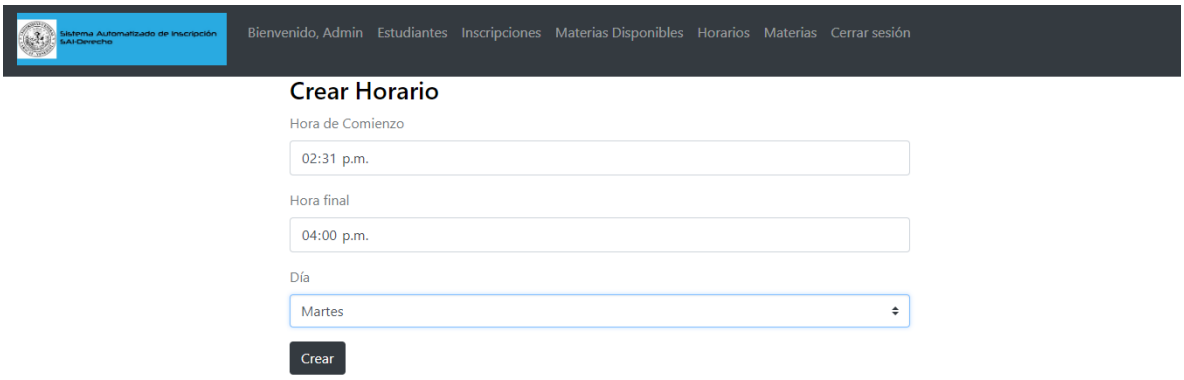

Figura 40 Vista de creación de un horario

En la vista general de horarios tenemos el listado, y las opciones de crear y eliminar de los mismos, figura 41.

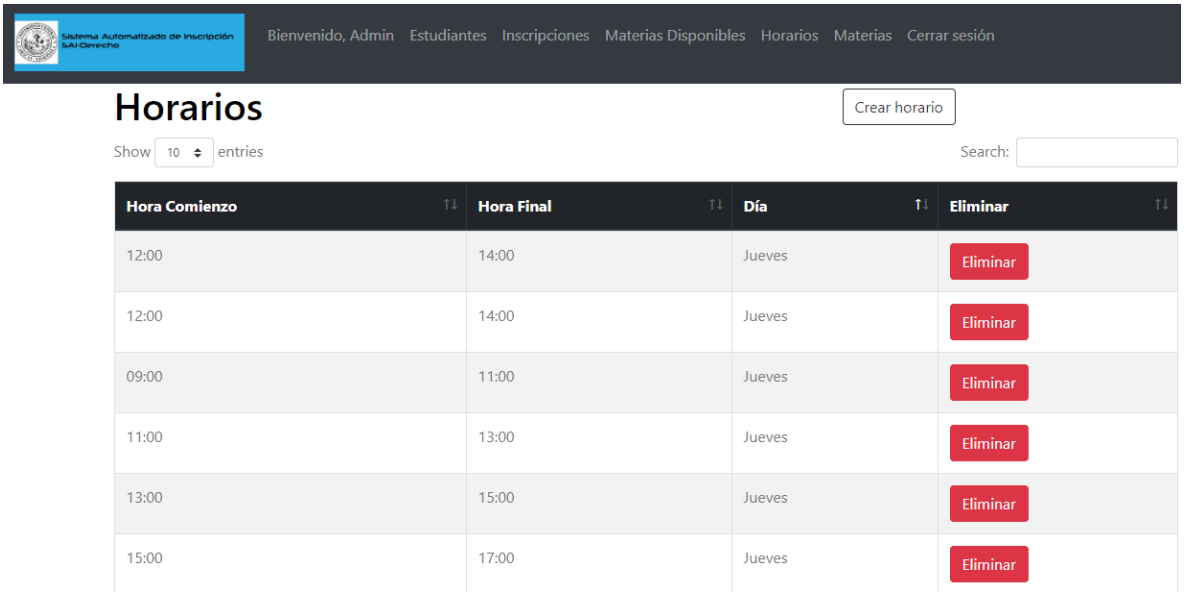

Figura 41 Vista general de Horarios

La sección de Materias en su vista general (figura 42) cuenta entre otros con un campo que indica si puede repetir en un semestre, es decir si dicha materia puede ser inscrita más de una vez en un semestre, que aplica mayormente para materias de los últimos semestres como los cursos monográficos que pueden ser inscritos más de uno en un semestre por políticas de la escuela.

Figura 42 Vista general de materias

Desde dicha vista podemos acceder al formulario de creación de materias (figura 43).

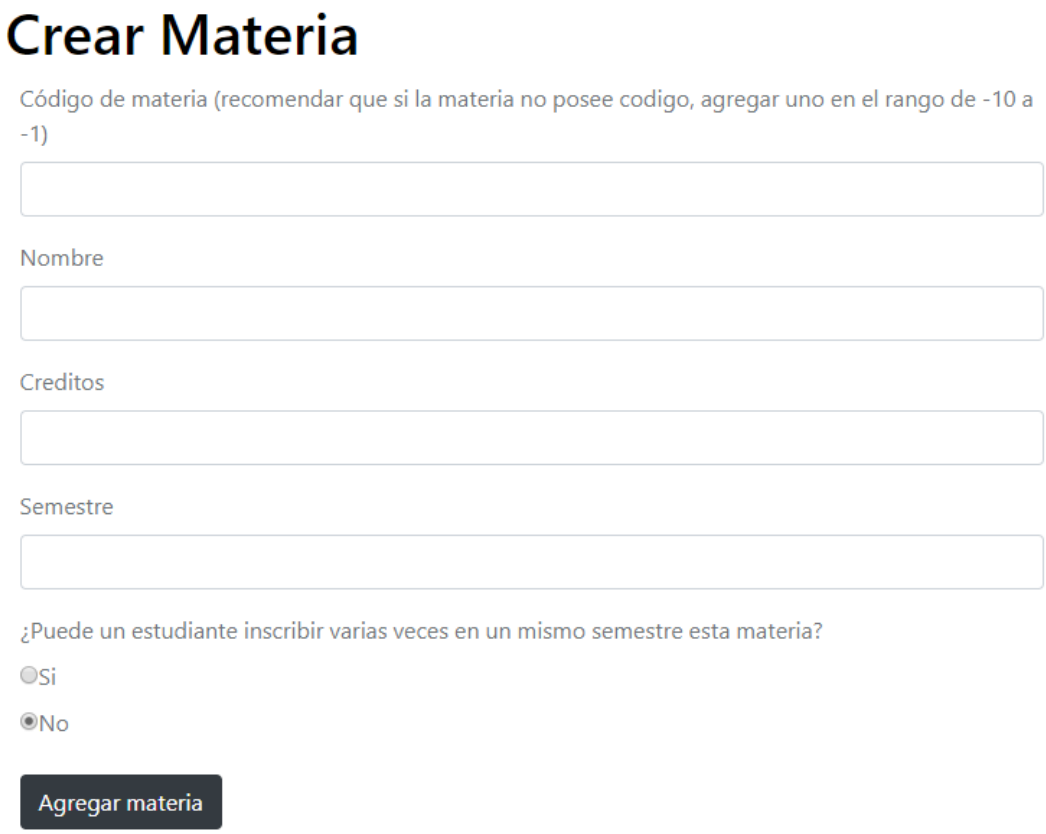

Figura 43 Vista de creación/edición de materias

Con esta se concluyen las interfaces desarrolladas en el sistema, cumpliéndose las funcionalidades esperadas.

Pruebas:

Se realizaron pruebas de flujo en la aplicación para garantizar la navegabilidad de la misma y que las funcionalidades sean accedidas desde los elementos de la interfaz.

Iteración 7

Descripción: Implementar módulo para el despliegue de la aplicación.

Fecha de inicio / Fecha fin: 03/12/2018 al 17/12/2018

Diseño:

Para dar despliegue de la aplicación en un ambiente de producción fue necesario implementar la carga "masiva" por archivo de los estudiantes que se inscribirán por semestre, esto para evitar el esfuerzo que conlleva crear los estudiantes de manera individual en la aplicación, estos usuarios contarían con el rol de estudiantes que solo les permitirán las inscripción y la generación del reporte de inscripción. Además se crearían con un token (contraseña) provisto por la Escuela de Estudios Políticos y Administrativos que le permitirían el ingreso a la aplicación. El archivo de entrada estaría en formato XLSX y con los datos organizado como se ve en la figura 44, en la primera fila el nombre de los campos, una segunda fila vacía y partir de la tercera los datos de los estudiantes. Para poder usar esta funcionalidad el administrador solo tendría que hacer click en un botón para iniciar la carga de un archivo que se encuentre en el computador.

	A	B	C	D	E	F	G	H
1	Marca temporal	Dirección de correo electrónico	Escriba su cédula	Primer Apellido	Segundo Apellido	Primer Nombre	Segundo Nombre	Token
2				#N/A	#N/A	#N/A	#N/A	#VALUE!
3	3/23/2018 10:47	julioandres72@gmail.com	20913999	HERNANDEZ	TAMAYO	JULIO	ANDRES	786651XTand
4	3/23/2018 10:56	genesysvillac@gmail.com	19967724	VILLAVICENCIO	CARRE±O	GENESYS		805969XC
5	3/23/2018 11:08	sergiog1999@hotmail.com	26902858	CABRERA	ROJAS	SERGIO	GABRIEL	196254XRgab
6	3/23/2018 11:42	possidente20@gmail.com	24861564	PONCE	POSSIDENTE	ALVARO	ANDRES	218977XPand
7	3/23/2018 11:42	naiviv.turmero@gmail.com	25545166	TURMERO	SINAPELLIDO	NAIVIV	SINNOMBRE	779857XSsin
8	3/23/2018 11:43	ariipalma@gmail.com	26739096	PALMA	VARGAS	ARIANNY	VIRGINIA	854201XVvir
9	3/23/2018 11:49	cristian221097@gmail.com	26045384	TORRES	PEÑA	CRISTIAN	JESUS	727383XPjes
10	3/23/2018 11:53	jordanysyramad@gmail.com	23631887	MARTINEZ	HIDALGO	JORDARY	SYRAMAD	143546XHsyr

Figura 44 Ejemplo de formato de datos para la carga masiva de estudiantes.

Además para poder enviar los correos dentro de la aplicación es necesario que se use una cuenta Gmail que sirva como emisor de los correos, por tanto, se creó un archivo en formato JSON con nombre “config.json” en el que se colocaría la información de ese correo para que pueda ser usado dentro del sistema. Este archivo además contaría con un campo que permitiría el ingreso del nombre del semestre que se va a cursar, esto para la personalización de las constancias de inscripción en cada semestre.

Codificación:

Para la implementación del diseño planteado se procedió a realizar un método que permitiera la carga de un archivo a la aplicación y que éste además sea interpretado en el formato xlsx, la librería PhpSpreadSheet fue utilizada para esta tarea. Luego de eso, la información de este archivo tiene que ser filtrada para poder ser usada en la aplicación, debido a que algunos estudiantes por ejemplo no poseen un segundo nombre o un segundo apellido, esos casos se manejan con condiciones en el código que impiden que ocurran errores en la creación de los estudiantes. Cada vez que sea leída la información de un estudiante se debe proceder al registro del mismo en la aplicación (figura 45).

```php
/**
 * Carga un archivo XLSX de los estudiantes en el sistema, el formato de ese archivo se puede ver en
general/formato_carga_estudiantes.xls
 */
public function load_estudiantes()
{
    $config['upload_path'] = './assets/files/';
    $config['allowed_types'] = 'xlsx';
    $config['file_name'] = 'out.xlsx';
    $config['overwrite'] = true;
    $this->load->library('upload', $config);
    $this->upload->initialize($config);

    if (!$this->upload->do_upload('userfile')) {
        echo 'Error subiendo el archivo (formato incorrecto o falla en la subida del archivo)';
        return;
    }
    $reader = new \PhpOffice\PhpSpreadsheet\Reader\Xlsx();
    $reader->setReadDataOnly(true);
    $reader->setReadFilter(new MyReadFilter());
    $spreadsheet = $reader->load("./assets/files/out.xlsx");
    $sheetData = $spreadsheet->getActiveSheet()->toArray(null, true, true, true);

    foreach ($sheetData as $key => $value) {

        if ($key == 1 || $key == 2) {
            continue;
        }

        if ($value['B'] == '' && $value['C'] == '') {
            break;
        }

        $f1_name = ($value['F'] == 'SINNOMBRE' || $value['F'] == '#N/A' ? '' : $value['F']);
        $f2_name = ($value['G'] == 'SINNOMBRE' || $value['G'] == '#N/A' ? '' : $value['G']);
        $f1_last_name = ($value['D'] == 'SINAPELLIDO' || $value['D'] == '#N/A' ? '' : $value['D']);
        $f2_last_name = ($value['E'] == 'SINAPELLIDO' || $value['E'] == '#N/A' ? '' : $value['E']);

        $email = strtolower($value['B']);
        $identity = $value['C'];
        $password = $value['H'];

        $additional_data = array(
            'first_name' => $f1_name . ' ' . $f2_name,
            'last_name' => $f1_last_name . ' ' . $f2_last_name,
            'ci' => $value['C'],
        );

        $this->ion_auth->register($identity, $password, $email, $additional_data);
    }
    redirect('estudiantes', 'refresh');
}
```

Figura 45 Método load_estudiantes para la carga masiva de estudiantes.

El archivo config queda planteado de la siguiente forma (figura 46).

```
{
    "semestre": "I-2019",
    "email": "email@gmail.com",
    "email_contrasena": "123456789a"
}
```

Figura 46 Archivo config.json con configuración inicial de la aplicación

Y su uso se puede apreciar en el ejemplo de envío de correos, se lee la información de ese archivo JSON para ser usado en donde corresponda (figura 47).

```
$json_content = json_decode(file_get_contents("./config.json"), true);

$config = array();
$config['charset'] = 'utf-8';
$config['useragent'] = 'Codeigniter';
$config['protocol'] = "smtp";
$config['mailtype'] = "html";
$config['smtp_host'] = "ssl://smtp.gmail.com";
$config['smtp_port'] = 465;
$config['smtp_timeout'] = "5";
$config['smtp_user'] = $json_content['email'];
$config['smtp_pass'] = $json_content['email_contrasena'];
$config['crlf'] = "\r\n";
$config['newline'] = "\r\n";
```

Figura 47 Ejemplo uso config.json

Pruebas:

Número	Descripción del caso de prueba	Resultado esperado	Resultado Obtenido	Motivo de la falla
1	Aceptar archivo con formato correcto y carga de datos.	Carga de datos en el sistema.	Carga exitosa de los datos de los estudiantes en la aplicación.	Un error en el procesamiento de la estructura de los datos ocasionaba que se cargaran datos en columnas que no correspondían
2	No aceptar archivo con formato incorrecto para carga de datos.	Mensaje de de error indicando el problema.	Mensaje de error indicando un problema con el archivo.	La validación no era suficientemente restrictiva solo para los archivos .xlsx y ocasionaba que otros tipos de archivos se leyeran cargando data corrupta

Conclusiones

El presente trabajo especial de grado enmarca el proceso de desarrollo de un software que será utilizado por la Escuela de Estudios Políticos y Administrativos para la realización de las inscripciones de estudiantes que se realizan cada semestre en esa escuela. Los objetivos planteados para la elaboración de este proyecto fueron llevados a cabo de manera satisfactoria, dando como resultado el estudio de los problemas existentes en esta escuela durante la realización de este proceso vital para el comienzo de un ciclo educativo, y además de una solución de software que permite solventar estos problemas de una manera eficaz. De manera conveniente (conveniencia dada por las circunstancias) tuvimos la oportunidad de usar esta solución en un ambiente de uso real antes de dar por completado su íntegra realización, esta situación nos permitió adquirir una mejor comprensión de cómo se beneficiaba la comunidad estudiantil con esta propuesta a diferencia de la manera en cómo se realizaba el proceso de inscripción en semestres pasados. Sumado a lo anterior están los beneficios inherentes a nivel de desarrollo de software que conllevan el uso de una aplicación aún en desarrollo en un ambiente de producción para encontrar bugs y realizar mejoras en la experiencia de usuario.

Inicialmente el desarrollo de esta aplicación fue planteado en base a sprints que creíamos que encajaban de buena manera con un proceso de elaboración que cumpliera los requisitos del proyecto, aunque luego el personal de la escuela nos solicitó acortar el tiempo

de desarrollo para poder usar la aplicación durante las inscripciones del semestre II-2018, esto si nosotros lo creíamos posible. Luego de haber analizado la situación optamos por acortar los tiempos del trabajo requerido para que esa solicitud fuera cumplida, además las condiciones estaban dadas para que fuese así; la metodología de desarrollo escogida podía soportar ese cambio y existía una buena disponibilidad por parte del personal de la escuela para ayudarnos a que se cumpliera esta labor de la mejor forma. Esto dio como resultado que diéramos prioridad a las historias de usuario fundamentales para que la aplicación pudiera salir a flote e ir resolviendo cualquier otra nueva característica o bug presente en el sistema a medida que creamos pertinente.

El uso de una metodología basada en Scrum nos dio pie para poder adaptarnos correctamente con los requisitos para la realización de este trabajo en todo momento, nos sirvió como un marco para la toma de decisiones y manejo de recursos que teníamos disponibles.

Esta aplicación comprende un conjunto de características que automatizan y simplifican la ejecución de las inscripciones en esta escuela, los estudiantes que hacen uso de esta herramienta para registrar su inscripción podrán ser favorecidos debido a que solo les llevará cuestión de minutos completar esta tarea, además tendrán disponible de manera personalizada un comprobante generado por el sistema que confirmará sus inscripciones, esto solventará los problemas mencionados en el Planteamiento del Problema de este documento. Para los administradores esta herramienta representa un mejor control sobre este proceso, podrán observar un reporte general de cómo las inscripciones de los

estudiantes fueron realizadas y tener disponible de forma simple la nómina de estudiantes por cada materia a ser dictada en el semestre. Los procedimientos esenciales para la inicialización del proceso de inscripción usando esta herramienta de software son bastante sencillos de realizar lo que facilita la conformación y configuración de las inscripciones en un semestre. Los administradores podrán cargar toda la información necesaria una vez se haya hecho el despliegue de la aplicación (carga de materias, horarios, etc.), a fin de que esta quede apta para las inscripciones.

Lo anterior generó como resultado un sistema robusto que dio fe, debido al uso de la versión beta en el semestre II-2018, para que pueda ser usado en un ambiente de producción sin inconveniente alguno. También debido a las tecnologías usadas podemos garantizar que el sistema se podrá instalar fácilmente en servidores en un futuro inclusive, y la aplicación podrá ser sencillamente modificada por un programador debido a que se usaron lenguajes de programación ampliamente usados y se trabajó siguiendo las directrices establecidas por el framework CodeIgniter.

Recomendaciones

A continuación presentamos algunas recomendaciones propuestas para continuar el incremento de características del sistema desarrollado en orden de mejorar su funcionamiento.

- Permitir la carga de archivos de datos en diversos formatos, actualmente el sistema solo permite la carga por archivos con formato XLSX.
- Crear restricciones para poder iniciar el proceso de inscripción, esto requerirá la implementación de una nueva característica que de acuerdo a un horario establecido habilitará o no la inscripción de estudiantes usando la aplicación.
- Crear un panel administrativo que permite a los usuarios de este rol observar mediante gráficos o tablas información que pueda ser relevante para la escuela con el objetivo de análisis sobre el proceso de inscripción.

Apéndice

Requisitos para la instalación de la aplicación

- El computador en el cual se va a instalar la aplicación debe de contar con un servidor Apache.
- El computador en el cual se va a instalar la base de datos debe de contar con el sistema de gestión de base de datos MySQL.

Pasos para la instalación

1. Obtener el archivo sis-inscripcion-ci.zip que contiene una carpeta contenedora de la aplicación y un archivo sis-inscripcion.sql para la configuración inicial de la base de datos.

2. Instalación de la base de datos:

La configuración inicial de la base de datos puede depender de acuerdo al lugar donde se instalará, normalmente hay dos alternativas:.

- Algunos servidores ya traen instalados un programa con nombre PhpMyAdmin, mediante el uso de esta herramienta se puede cargar el archivo .sql con solo seleccionar la pestaña de importar de este programa y luego seleccionar el archivo

sis-inscripcion.sql, esto cargará los datos iniciales a la base de datos. Además es común que también se encuentren disponibles los datos sobre la base de datos a usar, estos son hostname (el host donde se encuentra alojado la base de datos), user (el usuario para acceder a la base de datos), password (la contraseña de la base de datos) y database_name (el nombre de la base de datos), es recomendable anotar estos valores para usarlos luego.

- Mediante el uso del CLI de Mysql, ejecutar el comando mysql -u username -p sis-inscripcion < sis-inscripcion.sql

3. Instalación de la aplicación

- Abra el archivo sis-incripcion/application/config/config.php con un editor de texto y configure su URL base (Línea 26), en este campo deberá colocar entre comillas la url inicial del sitio donde estará alojada la aplicación (ejemplo: $config['base_url'] = 'http://derecho-noviembre.ml/';)
- Abra el archivo sis-incripcion/application/config/database.php con un editor de texto y establezca la configuración de su base de datos, solo hay que modificar estos cuatro valores: 'hostname' (el host donde se encuentra alojado la base de datos) , 'username' (el usuario para acceder a la base de datos), 'password' (la contraseña de la base de datos), 'database' (el nombre de la base de datos).
- Abra el archivo sis-incripcion/config.json con un editor de texto y modifique los valores de "semestre", "email", "email_contraseña" a los que correspondan. Ese

email es el que se usará como emisor de los correos (para que el correo pueda ser usado como emisor siga el paso número 1 de este sitio web https://www.hostinger.es/tutoriales/como-usar-el-servidor-smtp-gmail-gratuito/)

- Para instalar la aplicación basta con copiar y pegar el contenido de la carpeta sis-inscripcion al directorio html o htdocs del servidor.

Lista de referencias

Agile Methodology. (26 de Agosto de 2008). *Agile Methodology*. Obtenido de http://agilemethodology.org/?s=Agile+methodology&x=0&y=0

Apache HTTP Server Project. (s.f.). *Apache HTTP Server Project*. Obtenido de https://httpd.apache.org/

British Columbia Institute of Technology. (19 de Junio de 2017). *CodeIgniter*. Obtenido de https://www.codeigniter.es/user_guide/general/welcome.html

Cohn, M. (6 de Abril de 2007). *Mountain Goat Software*. Obtenido de Differences Between Scrum and Extreme Programming: https://www.mountaingoatsoftware.com/blog/differences-between-scrum-and-extreme-programming

Dingsøyr, T. (Junio de 2012). *Science Direct*. Obtenido de A decade of agile methodologies: Towards explaining agile software development: https://www.sciencedirect.com/science/article/pii/S0164121212000532

IBM. (s.f.). *IBM*. Obtenido de What is a database management system?: https://www.ibm.com/support/knowledgecenter/en/zosbasics/com.ibm.zos.zmiddbmg/zmiddle_46.htm

IBM. (s.f.). *IBM*. Obtenido de Model-View-Controller design pattern: https://www.ibm.com/support/knowledgecenter/en/SSZLC2_9.0.0/com.ibm.commerce.developer.doc/concepts/csdmvcdespat.htm

Kezmo. (20 de Marzo de 2017). *Kezmo*. Obtenido de ¿Qué son las metodologías ágiles y por qué debes implementarlas en tu organización?: https://blog.kezmo.com/qu%C3%A9-son-las-metodolog%C3%ADas-%C3%A1giles-y-por-qu%C3%A9-debes-implementarlas-en-tu-organizaci%C3%B3n-484a510e5b0

MDN Web Docs. (19 de Abril de 2018). *MDN Web Docs*. Obtenido de CSS: https://developer.mozilla.org/es/docs/Web/CSS

MDN Web Docs. (22 de Junio de 2018). *MDN Web Docs*. Obtenido de About JavaScript: https://developer.mozilla.org/en-US/docs/Web/JavaScript/About_JavaScript

MDN Web Docs. (28 de Enero de 2018). *MDN Web Docs*. Obtenido de Ajax: https://developer.mozilla.org/en-US/docs/Web/Guide/AJAX

Mitchell, B. (19 de Diciembre de 2017). *Lifewire*. Obtenido de Apache Web Server: https://www.lifewire.com/definition-of-apache-816509

Oracle Corporation. (s.f.). *MySQL*. Obtenido de Documentation: https://dev.mysql.com/doc/refman/5.7/en/what-is-mysql.html

Ospina, M. (2009). *Fundamentos y Conceptos Básicos de Bases de Datos.* Caracas.

Peck, J. (7 de Noviembre de 2013). *Lynda*. Obtenido de https://www.lynda.com/PHP-tutorials/What-CodeIgniter-why-should-I-use-it/126122/141742-4.html: https://www.lynda.com/PHP-tutorials/What-CodeIgniter-why-should-I-use-it/126122/141742-4.html

Peralta, A. (2003). *Facultad de Ingeniería - Ing. Bernard Wand-Polak.* Obtenido de Metodología SCRUM: https://fi.ort.edu.uy/innovaportal/file/2021/1/scrum.pdf

Pluralsight. (28 de Enero de 2015). *Pluralsight*. Obtenido de What's the Difference Between the Front-End and Back-End?: https://www.pluralsight.com/blog/film-games/whats-difference-front-end-back-end

Red Gráfica Latinoamérica. (s.f.). *Red Gráfica Latinoamérica*. Obtenido de El lenguaje de programación PHP: http://redgrafica.com/El-lenguaje-de-programacion-PHP

Silberschatz, A. (2002). *Fundamentos de Bases de Datos - Cuarta Edición.* Madrid: Concepción Fernández Madrid.

Sutherland, J. (Noviembre de 2017). *Scrumguides.* Obtenido de https://www.scrumguides.org/docs/scrumguide/v2017/2017-Scrum-Guide-Spanish-SouthAmerican.pdf

Technology Of Computing. (24 de Enero de 2017). *Technology Of Computing [Figura]*. Obtenido de Model-View-Controller Explained in C++: https://helloacm.com/model-view-controller-explained-in-c/

The Editors of Encyclopaedia Britannica. (s.f.). *Brittanica*. Obtenido de Client-server architecture: https://www.britannica.com/technology/client-server-architecture

Vignoni, D. (13 de Julio de 2011). *Wikipedia [Figura]*. Obtenido de https://en.wikipedia.org/wiki/Client%E2%80%93server_model#/media/File:Client-server-model.svg

Wong, K. (s.f.). *Coursera*. Obtenido de Software Architecture: https://es.coursera.org/learn/software-architecture