UNIVERSIDAD DE CARABOBO
Facultad Experimental de Ciencias y Tecnología
Licenciatura en Computación

Sistema de gestión de información de la Universidad de Carabobo para el ranking, utilizando arquitectura de Microservicios. Caso de estudio: información de estudiantes de pregrado y posgrado

Trabajo Especial de Grado presentado ante la ilustre Universidad
de Carabobo como credencial de mérito para optar al título de
Licenciado en Computación

Tutor Académico:
Ph.D. Desirée Delgado
Ph.D. Mirella Herrera

Autor:
Br. Juan Sánchez

Bárbula, diciembre de 2018

UNIVERSIDAD DE CARABOBO
Facultad Experimental de Ciencias y Tecnología
Departamento de Computación

FACYT

ACTA DE VEREDICTO DE TRABAJO ESPECIAL DE GRADO

Quienes suscribimos, profesores Desirée Delgado C.I.V- 7.352.958, Mirella Herrera, C.I. Nro. V-8.044.677, Francisca Grimón C.I. V-5.521.244 y Luis León C.I. V-14.999.453, integrantes del Jurado designado por el Consejo del Departamento de Computación de la Facultad Experimental de Ciencias y Tecnología (FaCyT) de la Universidad de Carabobo (UC), en su Reunión Ordinaria 21/2018 de fecha 27/11/2018, para considerar y evaluar el Trabajo Especial de Grado titulado **"Sistema de Gestión de Información de la Universidad de Carabobo para Rankings Universitarios, Utilizando Arquitectura de Microservicios. Caso de Estudio: Información de estudiantes de pregrado y posgrado"** presentado por el bachiller **Juan Sánchez C.I. V-24.569.851,** bajo la tutoría académica de las profesoras Desirée Delgado C.I.V- 7.352.958 y Mirella Herrera C.I. Nro. V-8.044.677, como requisito parcial para optar al título de Licenciado en Computación.

Una vez leído el Trabajo Especial de Grado se convocó a los miembros del jurado a la defensa pública el día 30 de enero de 2019, a la 8:30 a.m., en el Salón del Consejo de Facultad de la FACYT (Decanato). Finalizada la defensa pública del trabajo, los integrantes del Jurado emitimos por unanimidad el siguiente veredicto de **APROBADO** al trabajo sometido a nuestra consideración todo conforme a lo dispuesto en las Normas para la Elaboración, Presentación y Evaluación de Trabajo Especial de Grado de la Facultad Experimental de Ciencias y Tecnología de la Universidad de Carabobo.

En fe de todo lo cual levantamos y firmamos la presente acta de veredicto en Naguanagua, Estado Carabobo, a los treinta días del mes de enero de dos mil diecinueve.

Prof. DESIREE DELGADO
C.I. Nro. V-7.352.958
Tutor - Coordinador del Jurado
Dpto. de Computación FaCyT-UC

Prof. MIRELLA HERRERA C.
C.I. Nro. V- 8.044.677
Tutor
Dpto. de Computación- FaCyT-UC

Prof. FRANCISCA GRIMÓN
C.I. Nro. V-5.521.244
Miembro del Jurado
Dpto. de Computación FaCyT-UC

Prof. LUIS LEON
C.I. Nro. V-14.999.453
Miembro del Jurado
Dpto. de Computación FaCyT-UC

Dedicatoria

A los miembros de mi familia que me apoyaron durante todo el trayecto, en especial a mis padres y a mis amigos por su gran apoyo.

Agradecimientos

En primer lugar, agradezco inmensamente a mis tutores por haberme guiado y haber dedicado parte de su tiempo y esfuerzo. Siempre me ayudaron en todo momento, y por ello estaré siempre agradecido.

También quiero dar las gracias a mis padres, por sus enseñanzas, paciencia, motivación y apoyo que me brindaron en todo momento. Sin ustedes este camino no hubiera sido el mismo, y quizá no estaría donde estoy.

A todos mis profesores y compañeros que me acompañaron a lo largo de la carrera y formaron parte de esta etapa llena de aprendizajes.

Índice general

Sistema de gestión de información de la Universidad de Carabobo para el ranking, utilizando arquitectura de Microservicios. Caso de estudio: información de estudiantes de pregrado y posgrado

Resumen

La presente investigación se basará en el desarrollo de un sistema de gestión de información de la Universidad de Carabobo para el ranking, utilizando arquitectura de Microservicios. Caso de estudio: información de estudiantes de pregrado y posgrado. Para dicha investigación se utilizó la metodología de investigación-acción y como metodología de desarrollo, se utilizó la metodología Scrum. Como resultado del desarrollo del sistema de UC RANKING, se cumplieron todos los objetivos planteados. Además se obtuvo un software capaz de proporcionarle información al módulo de integración del sistema UC RANKING. Esto le permite al módulo de integración administrar la data de los estudiantes de pregrado y postgrado. Como recomendación es necesario hacer prueba de estrés del software desarrollado en este trabajo con la finalidad de conocer la magnitud que podría soportar el servidor.

Palabras clave: sistema, información, automatización, gestión, análisis, academia.

Autor:
Br. Juan Sánchez

Tutores:
Ph.D. Desirée Delgado
Ph.D. Mirella Herrera

Introducción

Desde su inicio, la Universidad como institución ha resultado pieza clave para la modernización de la sociedad. Tanto por su función docente que posibilita la difusión del conocimiento más avanzado a través de la formación de los estudiantes, como por su función investigadora que se centra en la generación del conocimiento abstracto que está en la base de la resolución de los problemas específicos de las empresas e instituciones. Este papel se ha vuelto más importante en el momento actual, cuando el conocimiento, en tanto que activo económico, ha alcanzado un valor estratégico para el desarrollo.

Por ello, el sector de la educación superior ha experimentado en las últimas décadas una demanda creciente, a la vez que una internacionalización progresiva. Ello ha hecho que la posibilidad de escoger entre una Universidad u otra, entre esta o aquella titulación, sea cada vez mayor, y que la inmensa mayoría de interesados (empresas, estudiantes,…) opten por la “mejor” de todas. Pero, ¿cuál es la mejor Universidad? En respuesta a esta pregunta, los rankings de Universidades se nos presentan no sólo como elementos decisorios, sino como procedimientos para valorar la calidad de estas instituciones. Teniendo en cuenta la importancia de la innovación universitaria, existe una necesidad apremiante de resultados estudios e iniciativas de mejora de la calidad en las empresas e instituciones de investigación.

La evaluación puede ser un estímulo poderoso para el mejoramiento institucional, pero si no se hace bien puede llevar también a decisiones equivocadas e injustas y a consecuencias negativas. La evaluación no es un fin en sí misma. Tiene sentido en la medida en que ofrece información que, por su calidad y pertinencia, constituya un elemento que ayude a mejorar, sea por la retroalimentación que ofrece para reorientar esfuerzos (evaluación formativa) sea

por su uso para asignar estímulos o sanciones, con la prudencia debida (evaluación sumativa). Estas consideraciones son especialmente importantes cuando a los resultados de la evaluación del desempeño o la calidad institucional se asocia decisiones fuertes, por ejemplo sobre financiamiento, como está ocurriendo con las evaluaciones que consisten en los llamados rankings de universidades.

Analizar la emergencia de los rankings y de su significado plantea desafíos de orden teórico y metodológico que comprenden los objetivos de los rankings, las definiciones de calidad que se adoptan, las unidades de análisis, las dimensiones y los indicadores que se eligen, las fuentes que se utilizan, los criterios de ponderación, la forma de organización de los resultados y las modalidades de difusión pública de sus resultados. Todos estos aspectos involucran opciones valorativas, conceptuales y metodológicas que es preciso examinar con atención.
Considerando que la visibilidad actúa como factor asociado a la producción y calidad del conocimiento científico en una comunidad académica institucional, se debe establecer una relación directa entre la producción científica y su difusión, y por consiguiente, el uso y el impacto en la comunidad internacional. Esta realidad, conlleva a las universidades a dar respuestas globales frente a una educación superior cada vez más universal e internacional, e impulsar la necesidad de medir el rendimiento de sus actividades.

Se determinó los factores que no ayudan en la divulgación en su totalidad de la producción científica de la Universidad de Carabobo. A partir de estos factores, se dio una propuesta para la mejora de la misma a través de un sistema que contribuya en la plena propagación de estas investigaciones, el criterio que se tomó para la presente investigación, fue esbozar una parte de dicho sistema que recolecta en todas las faculta desde la institución, la data asociada a los estudiantes de pregrado

y postgrado puesto que no se posee una organización que administre y organice esta información en conjunto.

Las actividades llevadas a cabo en esta investigación se describen en capítulos, a través de los cuales se distribuye la información de la siguiente manera:

Capítulo I: Presenta el planteamiento del problema, describe el objetivo general y los objetivos específicos, y la justificación de la investigación.

Capítulo II: Presenta el marco teórico, que incluye los antecedentes y referencias de proyectos relacionados con el Ranking de las universidades, como también una sección de bases teóricas donde se definen aquellos términos necesarios para comprender el contexto y el dominio de la investigación.

Capítulo III: Describe los aspectos metodológicos de la investigación, exponiendo el tipo de investigación llevada a cabo y la metodología de desarrollo de software seleccionada para llevar a cabo la implementación del sistema de Rankings.

Capítulo IV: En este capítulo se encuentra el proceso de desarrollo y los artefactos obtenidos de aplicar la metodología usada incluyendo los resultados, las conclusiones y recomendaciones de la investigación.

Capítulo I

El Problema

El Problema

1.1 Planteamiento del Problema

La educación es un derecho fundamental por ser inherente, inalienable y esencial a la persona humana, se ha convertido en un medio de realización de la actividad humana en todos los tiempos. El derecho a alcanzar el máximo desarrollo posible de las energías y características de la personalidad, de forma que toda persona pueda disfrutar de la vida personal y social de forma más integrada y plena posible. (Zambrano 2004). Ciertamente, la educación lleva implícito el conocimiento. Se educa para que la persona se prepare para el futuro; separar al ser humano de lo que la educación le aporta, es alejarlo del mundo social en el cual vive y de la naturaleza propia que constituye su existencia (Suying 2007).

La educación superior se le conoce a los estudios posteriores a la educación diversificada, donde el estudiantado escoge un área a fin en la cual desee continuar sus estudios con la aspiración de laborar en ese campo. La educación universitaria prepara a las personas para enfrentarse al mercado laboral, las especializa y capacita para el desempeño de sus quehaceres.

Para Manes (1997), la educación superior debe realizar un marketing educativo para desarrollar servicios de enseñanza que respondan a las demandas sociales, para garantizar el equilibrio entre los individuos y las organizaciones. Con esto, nos aclara el panorama que las universidades deben generar planes de estudios para las carreras acordes a las necesidades de la sociedad, que constantemente son cambiantes.

Las universidades tienen la tarea de formar ciudadanos plenamente capacitados en el campo laboral para la sociedad, por lo que también debe velar por la culminación de los estudios de sus estudiantes, para Bravo y Mejía (2010, p.1), “la educación superior enfrenta cambios asociados a su misión educativa en la sociedad. Estos cambios expresan una situación muy particular, la deserción estudiantil “, adicionalmente atribuye que las instituciones de educación superior deben de tener un equilibrio entre la cantidad de estudiantes que ingresan y los que desertan.

La importancia de la calidad a nivel de educación superior se evidencia con el desarrollo de organismos acreditadores nacionales e internacionales que realizan auditorías externas e internas de las instituciones y sus programas (Reid, 2009), por medio de estándares cuantitativos y cualitativos (Jones, 2003). Asimismo, en las últimas tres décadas se ha hecho evidente el surgimiento de rankings comparativos de instituciones educativas que establecen un conjunto de atributos que caracterizan a las universidades, comparándolas entre ellas y confiriendo autenticidad, legitimidad y estatus (Devinney, et al., 2008).

Los rankings universitarios tienen por objetivo realizar una jerarquización de las instituciones de educación superior basándose en parámetros e indicadores que intentan medir la calidad de la educación universitaria, el grado de investigación y otros aspectos de la actividad académica. (Ana, 2009). En sus concepciones iniciales los rankings fueron pensados como una herramienta para medir la efectividad de las instituciones de educación superior, su trabajo era considerar los avances internos de las instituciones; sin embargo, en la actualidad los calificadores de universidades y los promotores de este tipo de evaluación ponen poca atención a esta situación, sólo asumen que las instituciones ubicadas en un alto nivel en la

tabla son más productivas y tienen mejor calidad de docencia e investigación que las ubicadas en un nivel más bajo. Estos instrumentos se han convertido en simples listados de instituciones, donde se abre la competencia, primero por incluirse en la tabla y después por aparecer en mejores lugares.(Leyva,2012). La permanencia y credibilidad de la evaluación a través de los rankings depende de su transparencia, relevancia y la validación de los datos, pero se deben atender determinadas reglas que se encuentran expresadas en los denominados Principios de Berlín, los cuales fueron aprobados en mayo de 2006 por el *International Ranking Expert Group* (ireg). Según Altbach (2006): "Aplicar las normas y valores de las principales potencias académicas no medirá en forma exacta la calidad a nivel mundial, ni dará lugar a ranking mundiales de interés. En el competitivo y orientado hacia el mercado mundo académico del siglo XXI, los rankings son inevitables y probablemente necesarios. El desafío es asegurar que provean criterios exactos y relevantes y midan las cosas adecuadas.". Esta clasificación está enfocada para encontrar aquellas instituciones que conservan la mejor enseñanza a los estudiantes donde se hallan las mejores oportunidades para obtener un título universitario y la obtención de una mejor preparación para tener éxito en sus carreras.

Casi todas las clasificaciones internacionales se concentran en la medición de indicadores asociados a la circulación internacional de la producción de investigación como premios Nobel, artículos en revistas indexadas en Web of *Science* o *Scopus*, académicos altamente citados (HiCi), artículos en *Nature* y *Science* y citas por artículo, entre otros. Las actividades de formación de estudiantes, de extensión universitaria y difusión de la cultura, y la atención a diversas responsabilidades y compromisos con la sociedad, todas ellas funciones sustantivas de las universidades, están prácticamente ausentes de los rankings. La

indexación de revistas también presenta sesgos disciplinarios. Desde sus orígenes, el ISI, después WoS y el Scopus se orientaron fundamentalmente a la inclusión de revistas de ciencias biológicas y de la salud. Aunque ambos índices se han diversificado, la indexación y los sistemas de medición de impacto para las ciencias sociales y las humanidades son todavía incipiente. Algunos rankings (como THE y QS) utilizan métodos de reputación, a través de encuestas a muestras seleccionadas de académicos y empleadores potenciales de todo el mundo. La información sobre los criterios y la integración final de estas muestras seleccionadas es limitada.

La Universidad de Carabobo necesita de alguna forma estar sostenida por un buen proceso de divulgación de todos los proyectos científicos realizados en la misma, si lo anterior no se lleva a cabo, la mayoría de las investigaciones se quedan dentro de la institución, lo cual deja como resultado un impacto negativo en el indicador del ranking.

Aunque se está haciendo un trabajo a lo interno por potenciar las estructuras de investigación y propiciar el que hacer investigativo de la Universidad; los resultados de los indicadores bibliométricos de los ranking que miden la producción científica de la Institución, reflejan que los resultados de la investigación no están generando la difusión y el impacto esperado. En tal sentido, a mayor exigencia que establece la metodología del ranking, a los indicadores de producción académica, la Universidad de Carabobo se aleja de los lugares de vanguardia o simplemente no aparece en las tablas de clasificación. Por otro lado, si estos resultados se contrastan con los datos arrojados por motores de búsquedas y servicios de información científica como el *Web of Science y Scopus*, se podrá comprobar que la

visibilidad de la producción científica, mantiene las mismas tendencias y comportamiento.

En la actualidad la clasificación mundial de universidades QS, muestra que la UC va en un descenso en la posición en el ranking. Para el 2018 la institución está ubicada en la posición 171-180, es importante resaltar que a partir del año 2015, fue que comenzó a descender en este indicador, posiblemente sea por problemas financieros ya que hay otras universidades autónomas de Venezuela con el mismo descenso continuo.

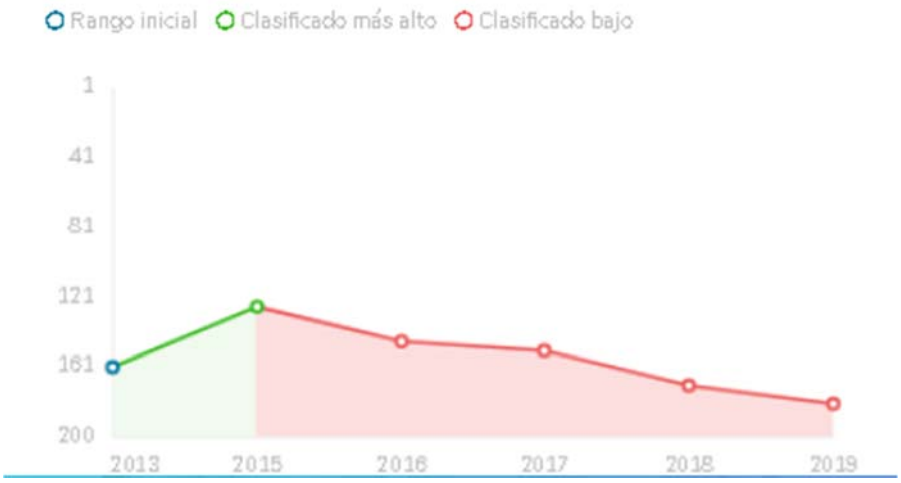

Figura 1: Descenso de la Universidad de Carabobo en el *rankings*

Fuente: *QS World University Rankings*, 2019.

Desafortunadamente en los años siguientes, los problemas financieros pueden llevar una caída drástica con respecto al indicador en el ranking de la institución, provocado por la fuga de talentos, la complejidad o imposibilidad de mantener ciertos procesos de la actividad científica o estructuras de la universidad, lo cual hace prácticamente imposible el incremento del posicionamiento de la institución.

La Universidad de Carabobo necesita tomar en cuenta a los estudiantes internacionales ya sea de pregrado o posgrado, debería de poseer una información verificable de ellos, es decir, saber cuánta de ella es real y cuanta es falsa, bien sea por manipulación voluntaria o por deficiencia en la recolección de los datos. Además se tiene que poseer una organización que verifique y administre de manera rigurosa todas las actividades de producción científicas llevadas a cabo en todas las facultades en conjunto, existe la posibilidad de que se realizaron artículos científicos en diferentes facultades de la institución y no se propagó de dicha facultad por la razón de no poseer una unidad superior; lo anterior puede llevar a la poca, desordenada e inexistente recolección de datos para el análisis en el rankings. Hay que tener en cuenta que "La accesibilidad es uno de los factores más importantes que determina la selección de una fuente de información para ser citada". Esta es una de las críticas más importantes que pueden dejar un impacto en el ranking para la institución. Para lograr el objetivo de posicionar la institución, es recomendable tener en cuenta un sistema que implemente la solución a los problemas antes mencionados.

1.2 Justificación

Los *rankings* universitarios son de suma importancia para la medición de la calidad de las instituciones, la Universidad de Carabobo actualmente se encuentra en un descenso continuo en el *rankings*, una de las posibles fallas que provoca este problema es porque debería de existir una mejor divulgación de la producción científica en conjunto de las facultades de la academia antes mencionada, es por esta razón que en la presente investigación tiene como aporte contribuir en el desarrollo de un nuevo sistema que ayude a una amplia divulgación de las investigaciones científicas, con el fin de obtener una merecida reputación a nivel nacional e internacional. Dicha contribución al nuevo sistema, es recolectar la información referida a los estudiantes de pregrado y posgrado de todas las facultades de dicha institución.

En efecto lo anterior podría hacer contribución a un mejor posicionamiento en el ranking, la data de los estudiantes puede contener los estudiantes extranjeros y venezolanos o los que han hecho maestrías por ejemplo. No cabe duda que este sistema dejara un impacto en los *rankings* de las universidades.

1.3 Objetivos de la Investigación

1.3.1 Objetivo General

Desarrollar un sistema de gestión de información de la Universidad de Carabobo para el ranking, utilizando arquitectura de Microservicios. Caso de estudio: información de estudiantes de pregrado y posgrado.

1.3.2 Objetivos Específicos

• Realizar una revisión bibliográfica en lo referente al desarrollo de sistemas web, almacenamiento de grandes volúmenes de datos, análisis de indicadores de ranking universitarios, entre otros

• Indagar sobre las estructuras del sistema de información de los estudiantes de pregrado y posgrado activos de cada una de las Facultades de la Universidad de Carabobo para analizar la gestión de los datos de estos sistemas.

• Implementar el módulo de gestión de información asociada a todos los estudiantes activos de la Universidad de Carabobo para el sistema UC RANKING.

• Realizar los manuales del sistema de gestión de estudiantes de la Universidad de Carabobo.

Capítulo II

Marco Teórico

2.1 Trabajos Relacionados

A continuación se presentan una serie de trabajos relacionados con el presente trabajo.

En referencia a Francisco Ponte (2013). Realizo un trabajo titulado: “Acercamiento a la Visibilidad de la Producción Científica de la Universidad de Carabobo. Propuesta de mejoras”. En esta investigación se planteó como objetivo: “Analizar la visibilidad de la producción científica de la Universidad de Carabobo, a través de los Rankings de Instituciones de Educación Superior, a fin de impulsar mejoras en la divulgación de los resultados de las actividades de investigación de la Institución”. La misma se enmarca dentro de la modalidad de Proyecto Factible. El estudio se fundamenta en una investigación de campo. Esta investigación deja un gran aporte a este proyecto puesto que realiza un estudio para dar propuestas de mejoras de una mayor visibilidad de la producción científica en la Universidad de Carabobo, esta mejoras fueron utilizadas para realizar una estructura de la información que se va a manejar en el sistema en este caso, la información de los estudiantes de pregrado y postgrado.

Una investigación similar es de José R. Alexander C. (2013). Titulado como “EL SISTEMA DE REGISTRO DE LA CARGA ACADÉMICA MEDIANTE ENTORNOS WEB. UNA PROPUESTA TECNOLÓGICA PARA LA GESTIÓN EN LA UNIVERSIDAD NACIONAL EXPERIMENTAL TÁCHIRA (VENEZUELA) “. Esta investigación planteo como objetivo “Diseñar un sistema de registro de la carga académica mediante entornos web como propuesta tecnológica para la gestión en la Universidad Nacional Experimental del Táchira.”. Esta tesis doctoral deja un importante aporte sobre los procesos de una carga académica en entornos web.

Otro trabajo similar fue Goyal M, Vohra R. (2012), el presente artículo titulado “Applications of Data Mining in Higher Education”, describe un enfoque para examinar el efecto del uso de técnicas de minería de datos en la educación superior para así mejorar la eficiencia de la misma. En dichas técnicas tales como: agrupación, árbol de decisión y asociación se aplican a procesos de educación superior, ayudando a mejorar el rendimiento de los estudiantes, su gestión del ciclo de vida, selección de curso, medir su tasa y la gestión de fondos de una institución. Este trabajo deja como aporte una mayor claridad sobre la minería de datos. Manejar toda una estructura de información de una institución, es sumamente complejo, por esta razón es necesario estar basado en una arquitectura que proporcione un excelente rendimiento para así manejar cualquier tipo de información de manera eficiente en una institución, sobre todo la información de estudiante, en donde el volumen, es inmenso.

Además un trabajo similar fue el de Elvia Gómez, Diana Jumbo y Adriana Rios(2012). Ellos realizaron un trabajo titulado: Desarrollo de un sistema automatizado para la inscripción y mantenimiento de información de alumnos vía internet, del colegio Primero de Mayo del cantón Yanzatza. En esta investigación se planteó como objetivo: Integración de la herramienta *e-learning MOODLE* al sistema de matriculación y mantenimiento de la información de alumnos vía internet del instituto tecnológico “Primero de Mayo”. La misma se enmarca dentro de la modalidad de Proyecto Factible. Este trabajo deja como contribución la estructura de la data que debe ser almacenada con respecto a un estudiante.

2.2 Fundamentos teóricos

La información de los estudiantes para ranking es de suma importancia, a partir de allí se puede llevar una estadística o estimación de los mismos, esto podría ayudar a determinar la popularidad de dicha institución y de que un estudiante termine su carrera exitosamente.

Para tener una popularidad se debe tomar en cuenta muchos factores como la calidad, el costo de vida, la diversidad cultural y el mercado laboral en la zona. Podemos partir de la premisa de que el costo de vida y el mercado laboral obtienen un puntaje muy bajo para este país sin nada que hacer al respecto, la diversidad cultural si se puede decir que obtiene un puntaje relativamente alto, la calidad es el punto más importante a resaltar, esta se mide en función de criterios, como el nivel de los estudiantes, el profesorado, los centros, actividad investigadora, publicaciones, entre otros.

2.2.1 Clasificación académica de universidades

Son listas ordenadas que clasifican a las universidades e instituciones de educación superior e investigación, de acuerdo a una metodología de tipo bibliométricos, que incluye criterios objetivos medibles y reproducibles, por ello el calificativo de "académica". El objetivo de estas listas es dar a conocer públicamente la calidad relativa de tales instituciones. Las listas clasificadoras son de dos tipos principales: globales o específicas. Las globales toman en cuenta al menos dos criterios y en general muchos de ellos a la vez. Las listas específicas se elaboran tomando en cuenta una sola categoría y están destinadas a valorar aquellos aspectos únicos en los que las instituciones individualmente pueden destacarse. Además de estos listados hay también otros que son producto de criterios subjetivos

a los que suele dárseles menos importancia pues carecen de rigor o seriedad ya que están basados fundamentalmente en sondeos de opinión, reflejando por ello, las experiencias personales y las opiniones subjetivas de los encuestados. (Susana y Alma, 2013)

2.2.2 Aplicación web

Se denomina aplicación web a toda aplicación de software con arquitectura cliente-servidor en el cual el cliente (o interfaz de usuario) es ejecutado en un navegador web (Nations, 2016).

Bajo la arquitectura mencionada, por lo general existen dos componentes fundamentales: El *frontend*, el cual conforma la interfaz de usuario que es ejecutada en el navegador web para presentar la información y permitir al usuario interactuar con la aplicación, y el *backend*, el cual se encarga de, en general, almacenar y recuperar información a través de una base de datos.

En la Figura 1 se puede apreciar el funcionamiento de las aplicaciones web. En primer lugar, el usuario realiza una solicitud al servidor web a través del internet con la ayuda del cliente o interfaz de usuario de la aplicación. Una vez que el servidor web recibe dicha solicitud, ésta es reenviada al servidor de aplicación web apropiado. El servidor de aplicación web realiza la tarea solicitada, como por ejemplo una consulta a la base de datos o el procesamiento de datos para entonces generar los resultados de la información solicitada. Luego, el servidor de la aplicación envía el resultado al servidor web con la información solicitada o los datos procesados y este último envía la respuesta al cliente con la información a visualizar en la pantalla del usuario.

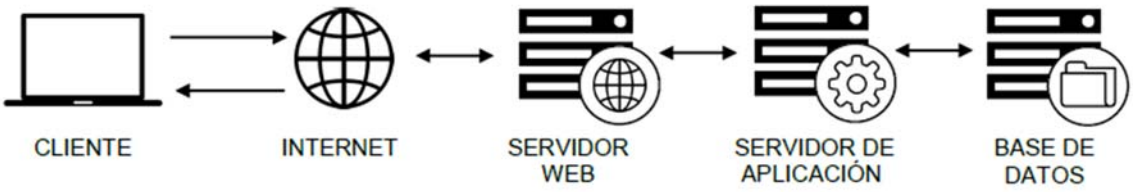

Figura 2: Funcionamientos de aplicaciones web.

Fuente: Khristopher Perdomo, 2017

De esta forma, los usuarios de la aplicación sólo necesitan de un navegador web moderno compatible con el sistema (como por ejemplo Google Chrome, Mozilla Firefox) y conexión a internet para hacer uso de ella.

2.2.3 Framework

En un artículo publicado por la Universidad de Sevilla (Gutiérrez, 2014), definen a un *framework* web como una aplicación genérica incompleta y configurable, con directrices arquitectónicas ofreciendo al desarrollador un conjunto de herramientas para agilitar el proceso de construir una aplicación web concreta, siempre teniendo en cuenta que es necesario adaptarlo para cada una de las aplicaciones a desarrollarse.

2.2.5 Interfaz de programación de aplicaciones

De acuerdo a Merino, M. (2014), interfaz de programación de aplicaciones o API (del inglés *Application Programming Interface*), es un conjunto de reglas (código) y especificaciones que las aplicaciones pueden interpretar para comunicarse entre ellas, sirviendo de interfaz entre programas o aplicaciones diferentes, de la misma manera en que la interfaz de usuario facilita la interacción

humano-software. Las API pueden servir para comunicarse con el sistema operativo, con bases de datos o con protocolos de comunicaciones.

2.2.6 Arquitectura de Microservicios

2.2.6.1 Definición

Microservicios, es otro término que ha revolucionado en la actualidad, describe un estilo de sistemas de software que nos resulta cada vez más atractivo. Muchos proyectos en los últimos años utilizan este enfoque, y los resultados producidos han sido positivos, tanto que se ha convertido en el estilo predeterminado para crear aplicaciones web.

Es un enfoque para desarrollar una sola aplicación como un conjunto de pequeños servicios, cada uno ejecutándose en su propio proceso y comunicándose con mecanismos livianos, a menudo una API de recursos HTTP. Estos servicios se basan en las capacidades empresariales y se pueden implementar de forma independiente mediante una maquinaria de implementación completamente automatizada. Existe un mínimo de administración centralizada de estos servicios, que puede escribirse en diferentes lenguajes de programación y utilizar diferentes tecnologías de almacenamiento de datos. (Fowler y Lewis, 2014).

Este estilo arquitectural fue acuñado por Martin Fowler en un taller de arquitectos de software como una descripción del nuevo campo que los participantes estaban explorando. No existe una definición en concreto para microservicio, sin embargo, una aproximación que la realiza (Newman, 2015) lo define como: “Pequeños servicios autónomos que trabajan juntos”.

Una arquitectura de microservicios promueve el desarrollo y despliegue de

aplicaciones compuestas por unidades independientes, autónomas, modulares y auto-contenidas, lo cual difiere de la forma tradicional o monolítico (Wolff E, 2016).

Una de las ventajas de utilizar microservicios es la capacidad de publicar una aplicación grande como un conjunto de pequeñas aplicaciones (microservicios) que se pueden desarrollar, desplegar, escalar, manejar y visualizar de forma independiente. Los microservicios permiten a las empresas gestionar las aplicaciones de código base grande usando una metodología más práctica donde las mejoras incrementales son ejecutadas por pequeños equipos en bases de código y despliegues independientes. La agilidad, reducción de costes y la escalabilidad granular, traen algunos retos de los sistemas distribuidos y las prácticas de gestión de los equipos de desarrollo que deben ser considerados. (Villamizar et al., 2015).

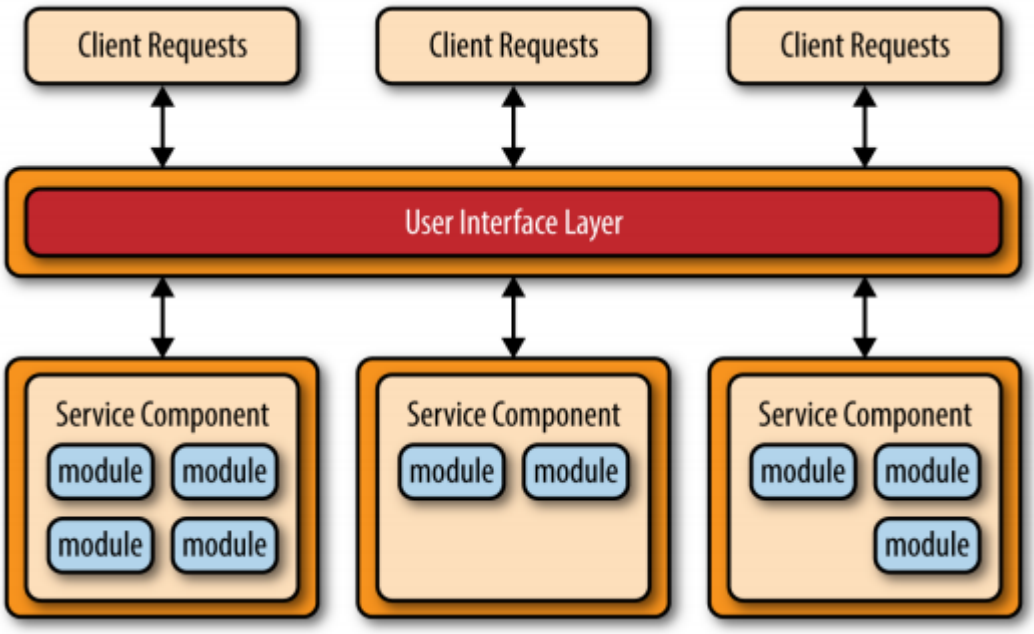

Figura 3. Patrón básico de arquitectura de Microservicios.

Fuente: Richards M, 2015.

2.2.6.2 Beneficios

Un enfoque de microservicios provee de una serie de beneficios tales como:

2.2.6.3 Intensa modularización

Básicamente, la arquitectura de microservicios consiste en dividir una aplicación o sistema en partes más pequeñas. La modularización facilita la automatización y proporciona un medio concreto de abstracción. Los servicios modularizados son desplegables de forma independiente y ayudan en la velocidad de la que se entrega del software (Nadareishvili, Mitra, McLarty, y Amundsen, 2016).

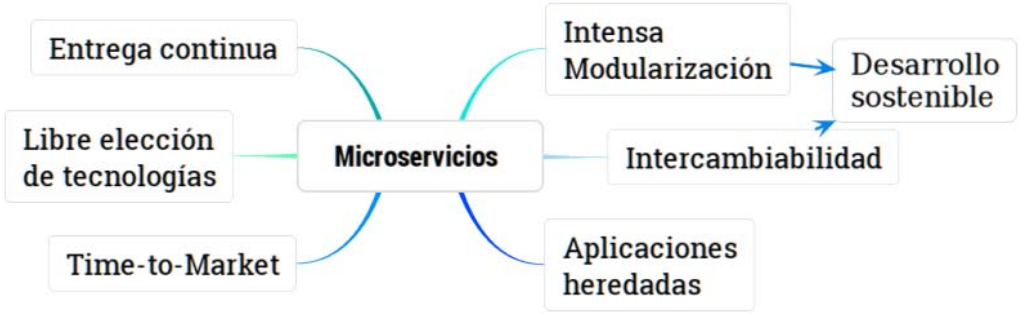

Figura 4. Beneficios de los microservicios.

Fuente: Wolff E, 2016

2.2.6.4 Intercambiabilidad

Los microservicios se pueden reemplazar más fácilmente que los módulos en un sistema monolítico. Si un nuevo servicio ofrece la misma interfaz, puede reemplazar el microservicio existente, el cual puede utilizar una base de código diferente e incluso diferentes tecnologías, siempre y cuando presente la misma interfaz, lo que en un sistema monolítico puede ser difícil o imposible de lograr. La

fácil sustitución de los microservicios reduce los costes de decisiones incorrectas, si un microservicio fue construido bajo la decisión de una tecnología o enfoque, este puede ser completamente reescrito si surge la necesidad. Esta necesidad de reemplazar el código en el futuro, frecuentemente se descuida durante el desarrollo de los sistemas de software (Wolff E, 2016).

2.2.6.5 Desarrollo sostenible

El desarrollo de software sostenible se debe gracias a la intensa modularización y la fácil sustitución o intercambiabilidad ya que los microservicios no están vinculados a una tecnología específica, lo que permite la utilización de nuevas tecnologías apropiadas para cada problema, evitando su caducidad o deterioro por la acción de su mantenimiento correctivo, evolutivo o adaptativo (dependiendo de qué se quiere perfeccionar).

2.2.6.6 Entrega continua

Lo que hace especial la modularización, es que cada microservicio puede ser desplegado de manera independiente. Concretamente, los microservicios pueden ser, por ejemplo: contenedores (Docker) en los que está instalado el software que constituye el microservicio que forma parte de un sistema de software general, siendo alternativamente, partes de una página HTML con llamadas JavaScript o pueden ofrecer servicios a través de REST, que pueden ser utilizados por otros microservicios u otros sistemas externos o clientes móviles. Estos microservicios pueden ofrecer una interfaz de usuario, o llamar a otros que implementan la lógica de negocio (Wolff E, 2017).

2.2.6.7 Libre elección de tecnologías

A diferencia de una aplicación monolítica en donde todos necesitan estar de acuerdo en un lenguaje de programación, *frameworks* o librerías, incluso de una versión específica de dichos *frameworks*, con un enfoque de microservicios la aplicación puede estar compuesta de servicios independientes que aprovechan diferentes *frameworks*, versiones de bibliotecas e incluso plataformas de sistema operativo completamente diferentes (Scholl, Swanson, y Fernandez, 2016).

2.2.6.8 Time-to-market

Los microservicios reducen el tiempo de lanzamiento al mercado (time-to-market) pudiendo permitir realizar cambios sólo en aquellos que requieren de un cambio, cada equipo a cargo del desarrollo de uno o varios microservicios puede desarrollar e implementar características sin la coordinación de tiempo con el resto de equipos, lo que permite trabajar en paralelo y poner en producción las nuevas funcionalidades en corto tiempo. (Wolff E, 2017).

2.2.6.9 Topologías

Según Mark Richards (2015), existen muchas maneras de implementar un patrón de arquitectura de microservicios, pero se destacan tres topologías principales que son las más comunes y populares: basada en API REST Transferencia de Estado Representacional (*Representational State Transfer*), basada en aplicaciones REST y la centralizada de mensajería.

- La topología basada en API REST es útil para sitios web que exponen servicios individuales pequeños y autónomos a través de algún tipo de API. Consta de componentes de servicio de grano fino (de ahí el nombre microservicios) que contienen uno o dos módulos que realizan funciones empresariales específicas independientes del resto de los servicios. En esta topología, estos componentes de servicio de grano fino se acceden normalmente mediante una interfaz basada en REST implementada a través de una capa de API basada en la Web desplegada separadamente. Algunos ejemplos de esta topología incluyen algunos de los servicios web REST basados en la nube como los usados por Yahoo, Google y Amazon.

- La topología basada en aplicaciones REST difiere del enfoque basado en API REST en que las solicitudes de cliente se reciben a través de pantallas de aplicaciones empresariales tradicionales basadas en web o en clientes pesados (*fat-client*) en lugar de una simple capa de API. La capa de interfaz de usuario de la aplicación se despliega como una aplicación web separada que accede de forma remota a componentes de servicio desplegados por separado (funcionalidad empresarial) a través de simples interfaces basadas en REST. Otra diferencia se encuentra en que los componentes de servicio tienden a ser más grandes, de grano más grueso y representan una pequeña porción de la aplicación empresarial general en lugar de granos finos, servicios de acción simple. Esta topología es común para las aplicaciones empresariales pequeñas y medianas que tienen un grado relativamente bajo de complejidad.

- La topología de mensajería centralizada, es similar a la basada en REST, excepto que en lugar de usar REST para acceso remoto, esta utiliza un

intermediario de mensajes centralizado ligero (por ejemplo, *ActiveMQ, HornetQ*, etc.). Es importante considerar que no se debe confundir con el patrón SOA o considerarlo "SOA-Lite". El agente de mensajes ligeros que se encuentra en esta topología no realiza ninguna orquestación, transformación o enrutamiento complejo, es sólo un transporte ligero para acceder a los componentes de servicio remotos. Esta topología se encuentra frecuentemente en aplicaciones de negocios más grandes o aplicaciones que requieren un control más sofisticado sobre la capa de transporte entre la interfaz de usuario y los componentes de servicio. Entre los beneficios que aporta frente a las anteriores es que son mecanismos avanzados de colas, mensajería asíncrona, monitoreo, manejo de errores y mejor balanceo de carga y escalabilidad. El único punto de fallo y los problemas de cuello de botella arquitectónico generalmente asociados con un intermediario centralizado se abordan a través de la agrupación de intermediarios y de la federación de intermediarios (dividir una única instancia de intermediario en múltiples instancias de intermediario para dividir la carga de procesamiento de mensajes en función de las áreas funcionales del sistema).

2.2.6.10 Aplicación monolítica

En una aplicación monolítica o bajo una arquitectura monolítica, toda la lógica se ejecuta en un único servidor de aplicaciones o un único programa que lo hace todo.

Las aplicaciones monolíticas típicas son grandes y construidas por múltiples equipos, requiriendo una orquestación cuidadosa del despliegue para cada cambio. También se consideran aplicaciones monolíticas cuando existen múltiples servicios

de API que proporcionan la lógica de negocio, toda la capa de presentación es una sola aplicación web grande, es decir la aplicación hace todo. (Stephens, 2015). En ambos casos, la arquitectura de microservicios puede proporcionar una alternativa.

Tabla 1. Arquitecturas monolíticas vs microservicios.

Categoría	Arquitectura Monolítica	Arquitectura de microservicios
Código	Una base de código única para toda la aplicación	Múltiples bases de código. Cada microservicio tiene su propia base de código.
Comprensibilidad	A menudo confuso y difícil de manejar.	Mayor facilidad de lectura y mucho más fácil de mantener.
Despliegue	Implementaciones complejas con ventanas de mantenimiento y paradas programadas.	Despliegue sencillo ya que cada microservicio se puede implementar de forma individual, con un tiempo de inactividad mínimo, si no es cero.
Idioma	Típicamente totalmente desarrollado en un lenguaje de programación.	Cada microservicio puede desarrollarse en un lenguaje de programación diferente.
Escalamiento	Requiere escalar la aplicación entera, aunque los cuellos de botella estén localizados.	Le permite escalar servicios con cuello de botella sin escalar la aplicación completa.

Fuente: Shahir Daya et al., 2015.

2.7 Plug-in

Según un artículo publicado por la Universidad del Sur de Florida (2011), un Plug-in es una pieza de software que actúa como un complemento para un navegador web o una aplicación.

Los Plug-ins pueden permitir que un navegador muestre contenido adicional para el que no fue diseñado originalmente o actuar como una funcionalidad adicional para un sistema.

Un ejemplo de un Plug-in es Macromedia Flash Player, el cual permite que el navegador web pueda visualizar animaciones utilizando el formato Flash.

2.8 Gestión de la Información

La gestión de información es todo lo que tiene que ver con obtener la información correcta, en la forma adecuada, para la persona indicada, al costo correcto, en el momento oportuno, en el lugar indicado para tomar la acción precisa. También se define como la coordinación eficiente y eficaz de la información procedente de fuentes internas y externas. Woodman y White, 1985.

2.9 Academic Ranking of World Universities (ARWU)

El Academic Ranking of World Universities (ARWU) cuya actualización se realiza anualmente, se publicó por primera vez en junio de 2003 por el Centro de Universidades de Clase Mundial y el Instituto de Educación Superior de la Universidad Shanghai Jiao Tong, China. A partir de 2009, el ARWU ha sido

publicado por la organización independiente Shanghai Ranking. El propósito inicial, se circunscribía a las principales universidades de China, pero su influencia traspasó fronteras; convirtiéndose en el sistema de clasificación internacional más ampliamente utilizado por las Instituciones de Educación Superior y de investigación. Francisco Ponte, 2013

En tal sentido, la gran influencia lograda está soportada en una metodología que emplea seis (06) indicadores objetivos, que incluyen: número de alumnos y el personal de ganadores Premios Nobel y Medallas Fields3; número de investigadores altamente citados seleccionadas por *Thomson Scientific*; número de artículos publicados en revistas de *Nature y Science*; número de artículos indexados en *Science Citation Index* y Social *Sciences Citation Index*, y el rendimiento per cápita con respecto al tamaño de la institución. Bajo criterios e indicadores, califican más de 12000 universidades cada año y las 500 mejores se incluyen en el ARWU y se publica en la web. Francisco Ponte, 2013

2.10 The World University Rankings (THE Ranking)

El *World University Rankings THE*, es una medición de universidades de todo el mundo que emplea trece (13) indicadores de rendimiento pertenecientes a las cuatro misiones de las universidades modernas y globales: investigación, docencia, transferencia del conocimiento e internacionalización. Estos indicadores se agrupan en cinco categorías: Enseñanza, Investigación, Citaciones, Ingresos de la industria y Panorama internacional. Este ranking es publicado por el diario británico *The Times* a través de *Times Higher Education*; en trabajo conjunto con la empresa *Quacquarelli Symonds* hasta el año 2009; y a partir del año 2010 crea una

metodología propia obteniendo la información de la base de datos de Thomson Reuters. Francisco Ponte, 2013

2.11 Ranking Mundial de Universidades en la Web (*Webometrics*)

El Ranking Mundial de Universidades en la Web es una iniciativa del Laboratorio de Cibermetría4, que pertenece a la Agencia Estatal Consejo Superior de Investigaciones Científicas (CSIC)5 de España. Proporciona una clasificación de instituciones de educación superior de todo el mundo desde el año 2004, con el objetivo de suministrar información sobre las IES, teniendo en cuenta su presencia e impacto en la Web.

La clasificación se construye a partir de datos publicados en la web, e indicadores de actividad e impacto (visibilidad web) se combinan en un indicador compuesto, que incluye no sólo las publicaciones formales (revistas electrónicas, repositorios), sino también la comunicación informal; teniendo en cuenta no sólo el impacto científico de las actividades de la universidad, sino también la importancia económica de la transferencia de tecnología a la industria, el compromiso con la comunidad e incluso la influencia política. Francisco Ponte, 2013.

2.12 QS World University Ranking (Ranking QS)

El *QS World University* Ranking es un *ranking* independiente y global, producido por la empresa *Quacquarelli Symonds QS*, que presenta una clasificación anual de las mejores 600 universidades del mundo y publicado desde el año 2004. Este ranking era un producto co-elaborado junto a *Times Higher Education*, una revista especializada en enseñanza superior, pero a partir del año 2009; ambas producen estudios por separado. Se caracteriza por utilizar la encuesta como una de las

fuentes de información más importantes. Los datos bibliométricos son suministrados por *Elsevier* a través de la herramienta *Scopus*.

El propósito de la clasificación ha sido reconocer las universidades como las organizaciones de múltiples facetas que son y para proporcionar una comparación global de su éxito frente a la misión teórica de permanecer o llegar a ser de clase mundial. De la versión mundial del *ranking QS World University Rankings*, se presentan por separado una medición para Asia (*QS Asia University Rankings*), y Latinoamérica (*QS Latin American University*). Francisco Ponte, 2013.

2.13 GIT

Es un control de versiones distribuido creado en el 2005 por los creadores de Linux. Dentro de las principales directrices que este sistema de control de versiones plantea se encuentran: la rapidez, posee un diseño sencillo, posee gran soporte para programación no lineal, es altamente distribuido y se puede utilizar para manejar proyectos grandes. Otra de las grandes ventajas de GIT es que no guarda copias de archivos por versiones sino que guarda el estado de los archivos, es decir, los cambios realizados sobre estos archivos en forma de enlaces, hecho que reduce el tamaño del sistema de control de versiones, ideal para proyectos de gran envergadura (Chacón, 2010).

Capítulo III

Marco Metodológico

3.1 Modalidad de la Investigación

En el siguiente proyecto se va a utilizar una metodología de tipo cualitativo. Para Rodríguez, Gil y García (1996) es difícil determinar cuáles son los métodos de investigación cualitativos. Estos autores explican que “método” es la forma característica de investigar, determinada por la intención sustantiva y el enfoque que la orienta. Los investigadores cualitativos estudian la realidad en su contexto natural, tal y como sucede, intentando sacar sentido de, o interpretar, los fenómenos de acuerdo con los significados que tienen para las personas implicadas. Este tipo de investigación implica la utilización y recogida de una gran variedad de materiales, como son la entrevista, también las experiencias personales, las historias de vida, las observaciones, los textos históricos, y por último, imágenes y sonidos (Rodríguez, Gil y García, 1996: 39). En su obra destacan las aportaciones recogidas por Taylor y Bogdan (1987), destacando que la metodología cualitativa es inductiva y con una perspectiva holística; además, los investigadores son sensibles a los cambios que generan y son comprensivos con las personas que investigan. En esta metodología, todas las perspectivas son valiosas y tenidas en cuenta, los métodos tienen un claro carácter humanista, además, añaden que se trata de un arte. En este proyecto se tomará el enfoque de **metodología de Investigación-acción** de tipo cualitativo.

Las "teorías" no se validan de forma independiente y luego se aplican a la práctica, sino que son validadas por la práctica misma. (Elliott, 1991, p. 69). La Investigación-acción, tiene en cuenta cuestiones de mejora y cambio social. Se trata de planificar una acción, actuar o llevarla a la práctica, observar la misma y por último, reflexionar sobre lo ocurrido para replantearse la acción inicial. Su fuente principal se encuentra en la teoría crítica. Por otro lado, Baskerville (1999), muestra

en forma de cinco fases que conforman un ciclo (Ver Figura 5), que se describen a continuación.

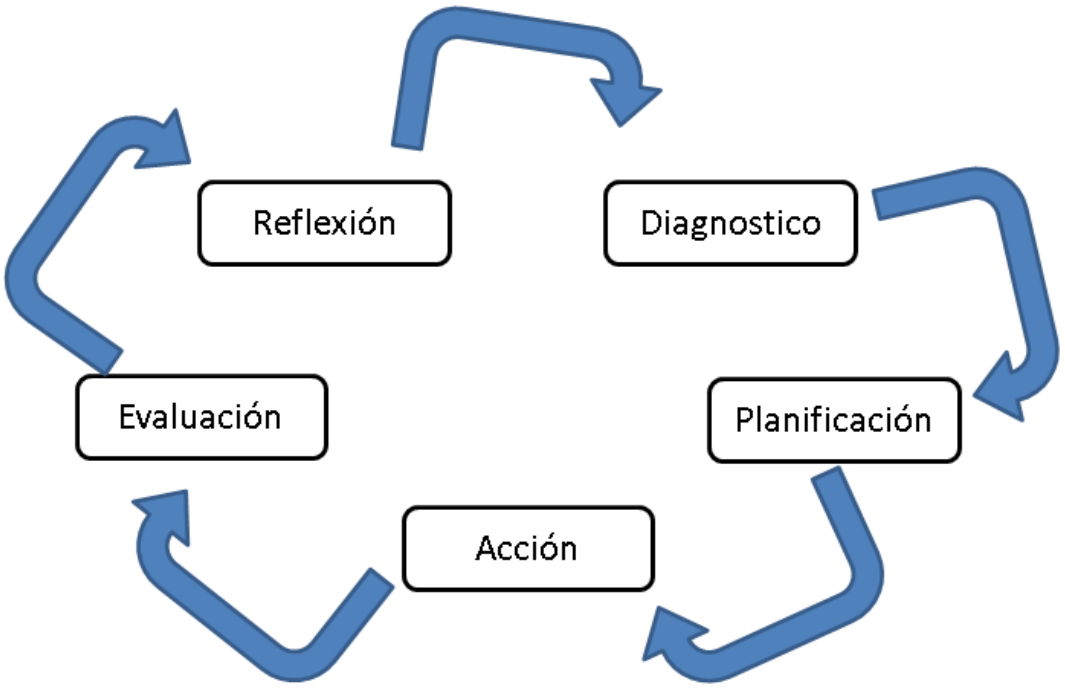

Figura 5: Ciclo de investigación-Acción.
Fuente: Baskerville, 1999.

Fase de diagnóstico: Se realiza el proceso de identificación de los problemas primarios de la investigación.

Fase de planificación: Se especifican las acciones que se llevarán a cabo para solucionar los problemas primarios.

Fase de acción: Se ejecutan las acciones planificadas en la fase anterior.

Fase de evaluación u observación: Se efectúa una evaluación de los resultados obtenidos, para observar, conocer y documentar los efectos de las acciones que fueron realizadas.

Fase de reflexión: Se toman los conocimientos adquiridos en la investigación-

acción. Si las acciones ejecutadas no fueron exitosas, los conocimientos pueden proporcionar la base para el diagnóstico de un nuevo ciclo de investigación-acción.

En la Tabla 2 se describen las fases de la Investigación-Acción instanciadas en la presente investigación.

Investigación-Acción

Tabla 2: Resumen de actividades a desarrollar en base a la metodología

Fase	Actividades
Diagnóstico	Se evalúa la problemática presente en la Universidad de Carabobo y se prepara la información para el análisis.
Planificación	Se define la arquitectura a utilizar, se realiza el diseño del sistema, y se escoge la metodología Scrum para el desarrollo del sistema.
Acción	Se implementa el sistema utilizando la metodología de desarrollo seleccionada.
Evaluación	Se analizan los resultados, se realizan pruebas y se elaboran manuales del sistema.
Reflexión	Se realizan las recomendaciones. Se especifican mejoras con respecto a los resultados.

Fuente: autor

3.2 Metodología de Desarrollo de Software

Scrum es una metodología ágil y flexible para gestionar el desarrollo de software, cuyo principal objetivo es maximizar el retorno de la inversión para su empresa (ROI). Se basa en construir primero la funcionalidad de mayor valor para el cliente y en los principios de inspección continua, adaptación, auto-gestión e innovación.

Otra definición es que es un proceso en el que se aplican de manera regular un conjunto de buenas prácticas para trabajar colaborativamente, en equipo, y obtener el mejor resultado posible de un proyecto. Estas prácticas se apoyan unas a otras y su selección tiene origen en un estudio de la manera de trabajar de equipos altamente productivo

En *Scrum* se realizan entregas parciales y regulares del producto final, priorizadas por el beneficio que aportan al receptor del proyecto. Por ello, *Scrum* está especialmente indicado para proyectos en entornos complejos, donde se necesita obtener resultados pronto, donde los requisitos son cambiantes o poco definidos, donde la innovación, la competitividad, la flexibilidad y la productividad son fundamentales.

Scrum también se utiliza para resolver situaciones en que no se está entregando al cliente lo que necesita, cuando las entregas se alargan demasiado, los costes se disparan o la calidad no es aceptable, cuando se necesita capacidad de reacción ante la competencia, cuando la moral de los equipos es baja y la rotación alta, cuando es necesario identificar y solucionar ineficiencias sistemáticamente o cuando se quiere trabajar utilizando un proceso especializado en el desarrollo de producto.

3.2.1 El proceso

En *Scrum* un proyecto se ejecuta en ciclos temporales cortos y de duración fija (iteraciones que normalmente son de 2 semanas, aunque en algunos equipos son de 3 y hasta 4 semanas, límite máximo de *feedback* de producto real y reflexión). Cada iteración tiene que proporcionar un resultado completo, un incremento de producto final que sea susceptible de ser entregado con el mínimo esfuerzo al cliente cuando lo solicite.

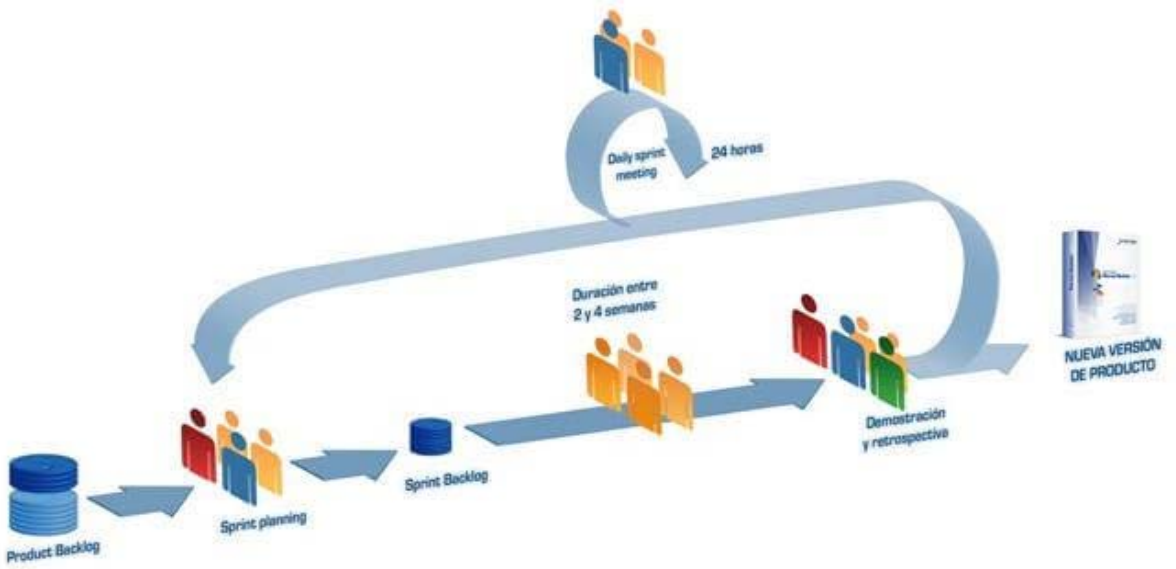

Figura 6: Proceso de la metodología *Scrum*

***Fuente:* Softeng, 2018.**

El proceso parte de la lista de objetivos/requisitos priorizada del producto, que actúa como plan del proyecto. En esta lista el cliente (*Product Owner*) prioriza los objetivos balanceando el valor que le aportan respecto a su coste (que el equipo

estima considerando la Definición de Hecho) y quedan repartidos en iteraciones y entregas.

Las actividades que se llevan a cabo en *Scrum* son las siguientes (los tiempos indicados son para iteraciones de 2 semanas):

Planificación de la iteración: El primer día de la iteración se realiza la reunión de planificación de la iteración. Tiene dos partes:

Selección de requisitos (2 horas). El cliente presenta al equipo la lista de requisitos priorizada del producto o proyecto. El equipo pregunta al cliente las dudas que surjan y selecciona los requisitos más prioritarios que prevé que podrá completar en la iteración, de manera que puedan ser entregados si el cliente lo solicita.

Planificación de la iteración (2 horas). El equipo elabora la lista de tareas de la iteración necesarias para desarrollar los requisitos seleccionados. La estimación de esfuerzo se hace de manera conjunta y los miembros del equipo se auto asignan las tareas, se auto organizan para trabajar incluso en parejas (o grupos mayores) con el fin de compartir conocimiento o para resolver juntos objetivos especialmente complejos.

3.2.2 Ejecución de la iteración

Cada día el equipo realiza una reunión de sincronización (15 minutos), normalmente delante de un tablero físico o pizarra (*Scrum Taskboard*). El equipo inspecciona el trabajo que el resto está realizando (dependencias entre tareas, progreso hacia el objetivo de la iteración, obstáculos que pueden impedir este objetivo) para poder hacer las adaptaciones necesarias que permitan cumplir con la

previsión de objetivos a mostrar al final de la iteración. En la reunión cada miembro del equipo responde a tres preguntas:

¿Qué he hecho desde la última reunión de sincronización para ayudar al equipo a cumplir su objetivo?

¿Qué voy a hacer a partir de este momento para ayudar al equipo a cumplir su objetivo?

¿Qué impedimentos tengo o voy a tener que nos impiden conseguir nuestro objetivo?

Durante la iteración el Facilitador (*Scrum Master*) se encarga de que el equipo pueda mantener el foco para cumplir con sus objetivos. Elimina los obstáculos que el equipo no puede resolver por sí mismo. Protege al equipo de interrupciones externas que puedan afectar el objetivo de la iteración o su productividad.

Durante la iteración, el cliente junto con el equipo refinan la lista de requisitos (para prepararlos para las siguientes iteraciones) y, si es necesario, cambiar o re planifican los objetivos del proyecto (10%-15% del tiempo de la iteración) con el objetivo de maximizar la utilidad de lo que se desarrolla y el retorno de inversión.

3.2.3 Inspección y adaptación

El último día de la iteración se realiza la reunión de revisión de la iteración. Tiene dos partes:

Revisión (demostración) (1,5 horas). El equipo presenta al cliente los requisitos completados en la iteración, en forma de incremento de producto

preparado para ser entregado con el mínimo esfuerzo. En función de los resultados mostrados y de los cambios que haya habido en el contexto del proyecto, el cliente realiza las adaptaciones necesarias de manera objetiva, ya desde la primera iteración, re planificando el proyecto.

Retrospectiva (1,5 horas). El equipo analiza cómo ha sido su manera de trabajar y cuáles son los problemas que podrían impedirle progresar adecuadamente, mejorando de manera continua su productividad. El Facilitador se encargará de eliminar o escalar los obstáculos identificados que estén más allá del ámbito de acción del equipo.

3.3 Artefactos Rup:

La metodología consta de una serie de artefactos los cuales serán nombrados a continuación:

- Casos de uso
- Especificación de requisitos
- Bpmn
- Modelo E-R
- Historias de uso

Capítulo IV

Resultados

El presente capítulo muestra los resultados obtenidos durante el desarrollo de la investigación.

Este trabajo se divide en 5 partes, en este capítulo se desarrolló la parte de recolección de información de los estudiantes, lo cual de alguna forma va a tener una comunicación con los módulos de integración y usuarios. En la fase de la metodología Investigación/acción específicamente el actuar se desarrollara toda la arquitectura general y específica del sistema.

4.1 Diagnóstico

Durante esta primera fase se realizó un estudio a la institución para así evaluar la problemática presentada. Se llevó a cabo un estudio de varios parámetros en donde se pudo constatar que para darle una merecida posición a la institución, se necesita de un sistema que proporcione una mejor divulgación acerca de la información particular o complementaria de los estudiantes.

4.2 Planificar

Se realizó un proceso de recolección de datos sobre los requisitos a través de múltiples reuniones con el cliente, para así exponer la problemática presentada que dio paso al desarrollo de esta investigación. Hubo cierta complejidad definir la necesidad por parte del cliente debido a la magnitud del proyecto.

4.3 Acción

Durante esta tercera fase se establecieron los objetivos de la investigación, general y específicos. Igualmente se realizó un estudio teórico y bibliográfico de los conceptos que soportan dicha investigación; especialmente sobre los sistemas de

Ranking de las instituciones. También se realiza un estudio de la arquitectura general y los procesos generales del sistema mismo. Asimismo, siendo necesario integrar el sistema a desarrollar con los sistemas existentes de los controles de estudios de la UC, se realizó un estudio de dichas arquitecturas del sistema.

A continuación se detallara todo lo que se llevado a cabo mediante la metodología de desarrollo *Scrum*. En la respectiva metodología se incluirán distintos artefactos ya mencionados anteriormente, para el diseño del módulo de este trabajo.

En la siguiente tabla se realizara un resumen de todos los Sprints realizados en esta investigación.

Tabla 3: Resumen de Sprints para el módulo de los estudiantes

Sprints	Actividades	Duración
1	Diseño de la arquitectura general del sistema UC RANKING	1 semana
2	Se define el diseño del módulo “Estudiantes” perteneciente al sistema UC RANKING	4 semanas
3	Implementación del módulo “Estudiantes” en el sistema UC RANKING.	3 semanas
4	Integración de todos los módulos que comprende el sistema UC RANKING	1 semana

Fuente: Autor

4.3.1 Sprint 1:

4.3.1.1 Diseño de la arquitectura general del sistema

Código: S01

Estimación: 1 semana.

En este sistema, llamado UC RANKING, pueden acceder sólo los usuarios autorizados de la Universidad de Carabobo, pudiendo cargar información por lotes referente a los estudiantes y docentes mediante la carga de archivos, además pueden agregar y modificar usuarios, también se pueden modificar parámetros del sistema, visualizar las acciones de los usuarios, generar reportes, entre otras funciones. Esto evidencia la necesidad absoluta de mantener una comunicación constante entre los involucrados. La arquitectura y el modelado de procesos general del sistema se muestran a continuación:

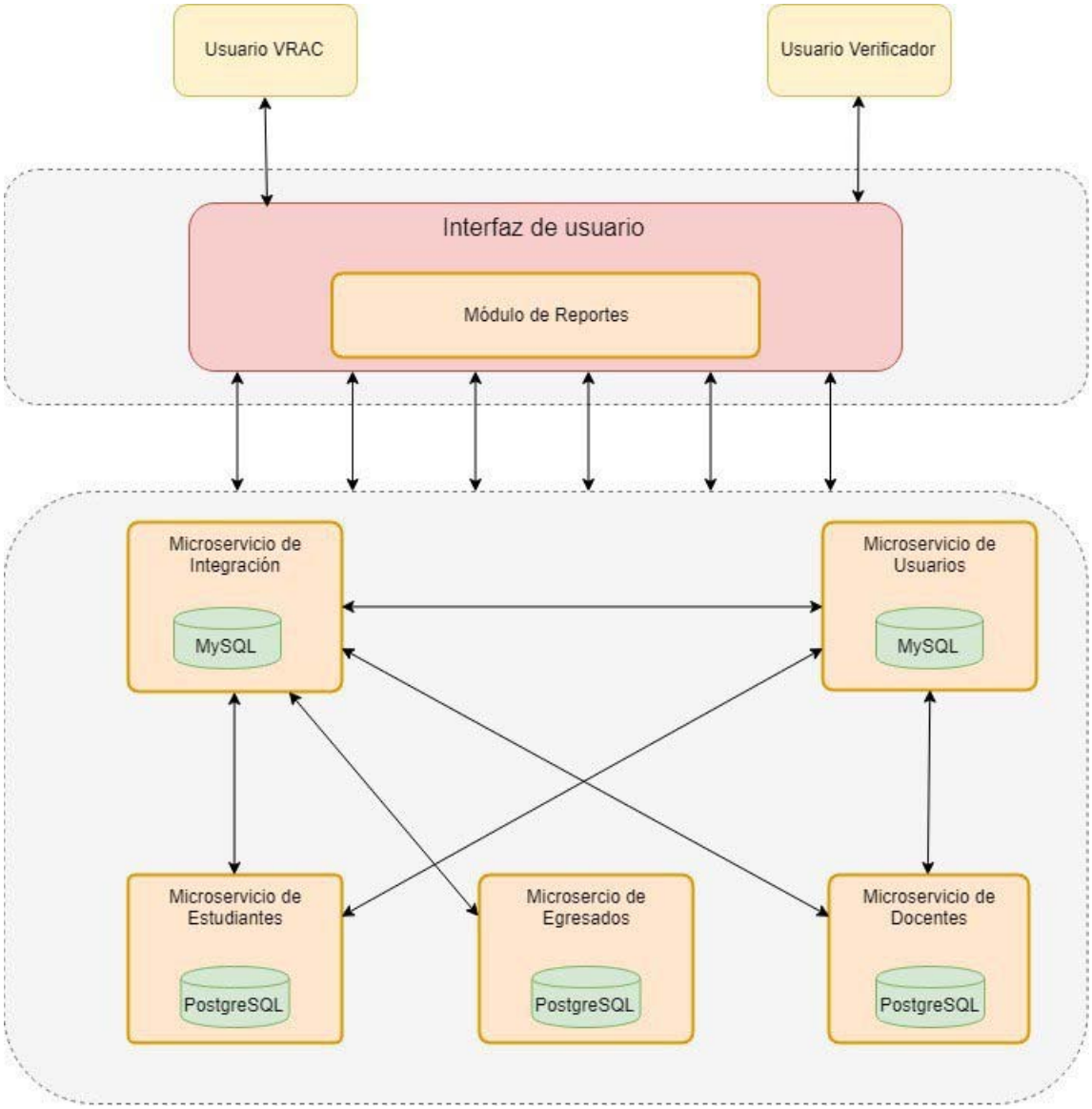

Figura 7: Arquitectura general del sistema

Fuente: Wilker Giovanni, 2019

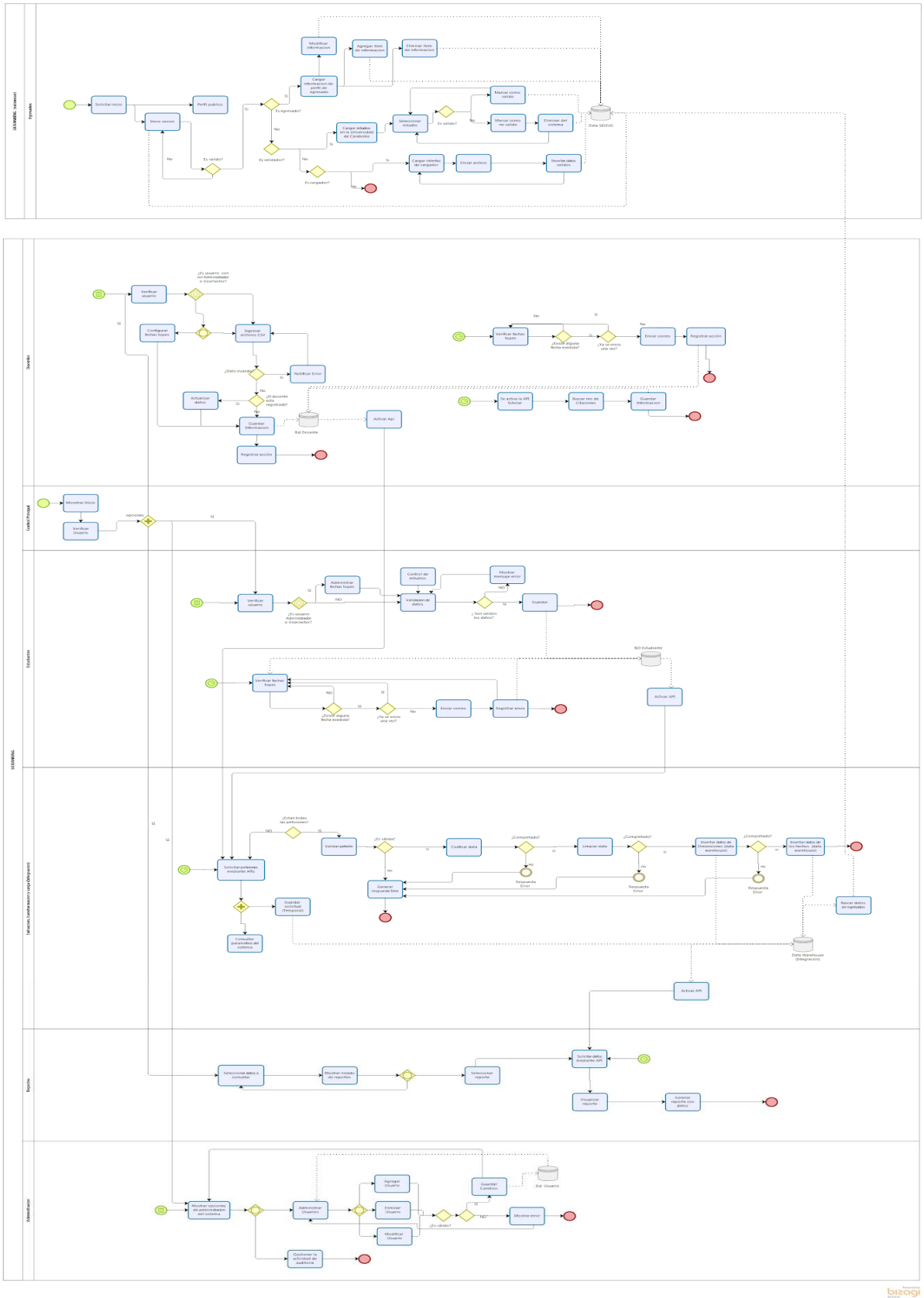

Figura 8: Modelado de procesos del sistema UC RANKING

Fuente: Autor

En la figura 8 se puede observar el modelado de procesos del sistema UC RANKING, la cual posee una extensión del sistema antes mencionado, dicha extensión es el módulo de los egresados, se decidió separar este módulo porque adicionalmente administra información de manera externa. El sistema UC RANKING consta de varios subprocesos, donde el primer subproceso, es el control principal, allí se ejecuta un proceso para iniciar el sistema, luego se tiene otro proceso para verificar el usuario, también se tiene el subproceso de administración, la cual es el encargado de verificar el estado de los microservicios de los módulos de los estudiantes, docente y egresados; también es utilizado para administrar usuarios con sus respectivos roles. En la sesión de procesos de negocios de este trabajo, se detalla el subproceso de los estudiantes y en los demás trabajos del sistema UC RANKING se detallan los subprocesos egresados, docentes, reportes e integración respectivamente.

4.3.2 Sprint 2:

Código: S02

Estimación: 4 semanas.

4.3.2.1 Diseño de la arquitectura del módulo de los estudiantes

A continuación se muestra una figura de la arquitectura del módulo del estudiante. El módulo se divide en dos secciones llamadas Frontend y Backend, los procesos externos a ellos son para uso de entrada y salida del módulo. El proceso de control de estudios se utiliza como entrada al *Frontend*, el proceso de

integración se utiliza para que el sistema de los estudiantes, le proporcione una salida mediante una llamada de una API alojada en el Backend.

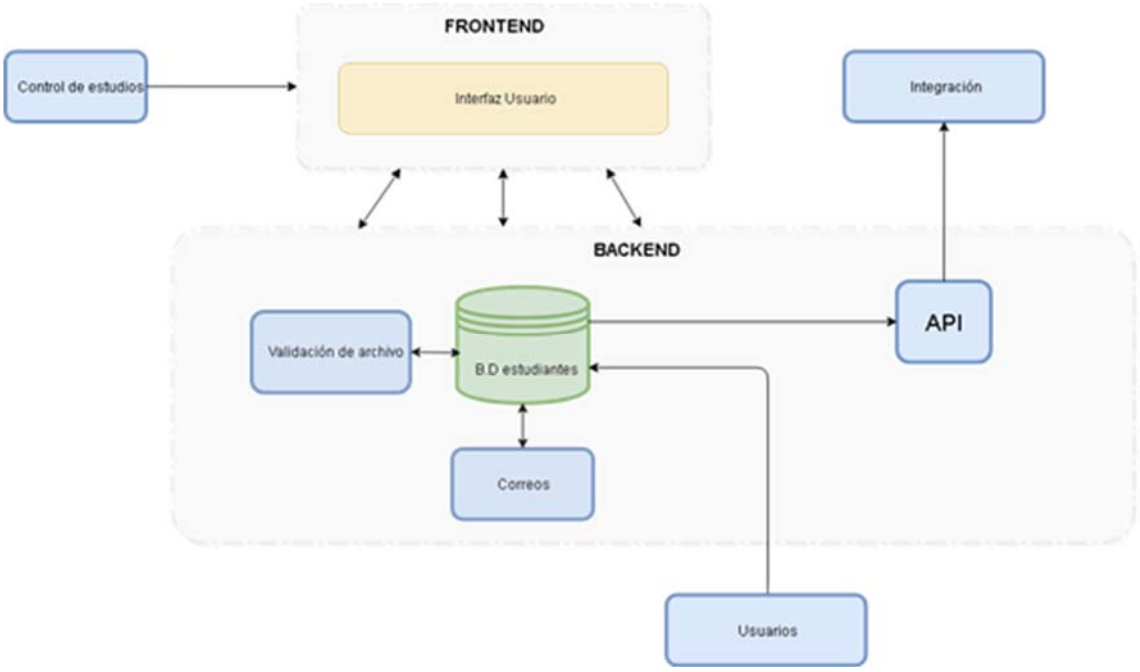

Figura 9: Arquitectura del módulo de los estudiantes

Fuente: Autor

4.3.2.2 Casos de Uso

Actores:

Analista de control de estudio: Debe subir el archivo asociado a la información de los estudiantes

Administrador: Verifica y supervisa los usuarios asignados

Vicerrector académico: Chequea que la información sea entregada a tiempo por parte del analista de control de estudios

Módulo de carga de archivo

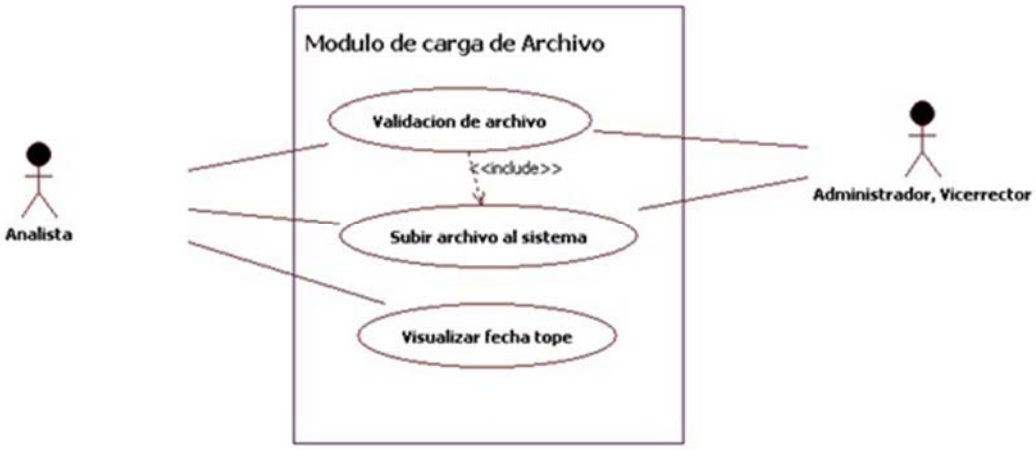

Figura 10: Diagrama de casos de uso para la carga de archivo

Fuente: Autor

Casos de uso involucrados:

Validación de archivo: El sistema debe verificar que el archivo subido esté correctamente llenado.

Subir el archivo al sistema: El actor sube el archivo al sistema y al subir se ejecuta el caso anterior.

Visualizar fecha tope: Para este caso solo el analista podrá ver la fecha tope, el administrador y el vicerrector podrán ver todas las fechas topes de todas las facultades en otro módulo.

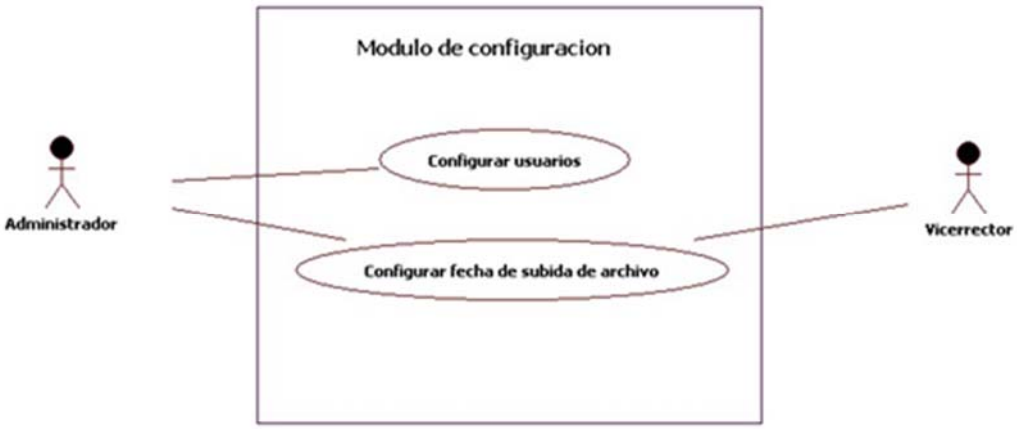

Figura 11: Diagrama de casos de uso para el módulo configuración.
Fuente: Autor.

Casos de uso involucrados:

Configurar usuarios: El administrador configura los usuarios del sistema

Configurar fechas de subida de archivos: El administrador y el vicerrector pueden administrar las fechas tope de la subida del archivo de cada facultad.

4.3.2.3 Procesos de negocio

A continuación se mostrará el modelado del proceso de negocios de módulo de los estudiantes. El Módulo inicia verificando el Usuario en donde partir de allí, se podrá tener distintos permisos como es el caso del vicerrector y el administrador que pueden ver y modificar todas las fechas límites de carga de archivo de todas las facultades de la UC. Tanto el administrador como el vicerrector podrán subir los archivos de cualquier facultad, el analista sólo podrá subir el archivo asociado a su facultad. Este modelo de procesos se realizó en Bizagi 3.2.7.

El sistema envía un correo de manera automática cuando detecta que hay una fecha tope de subida del archivo retrasada. El correo solo se envía una vez.

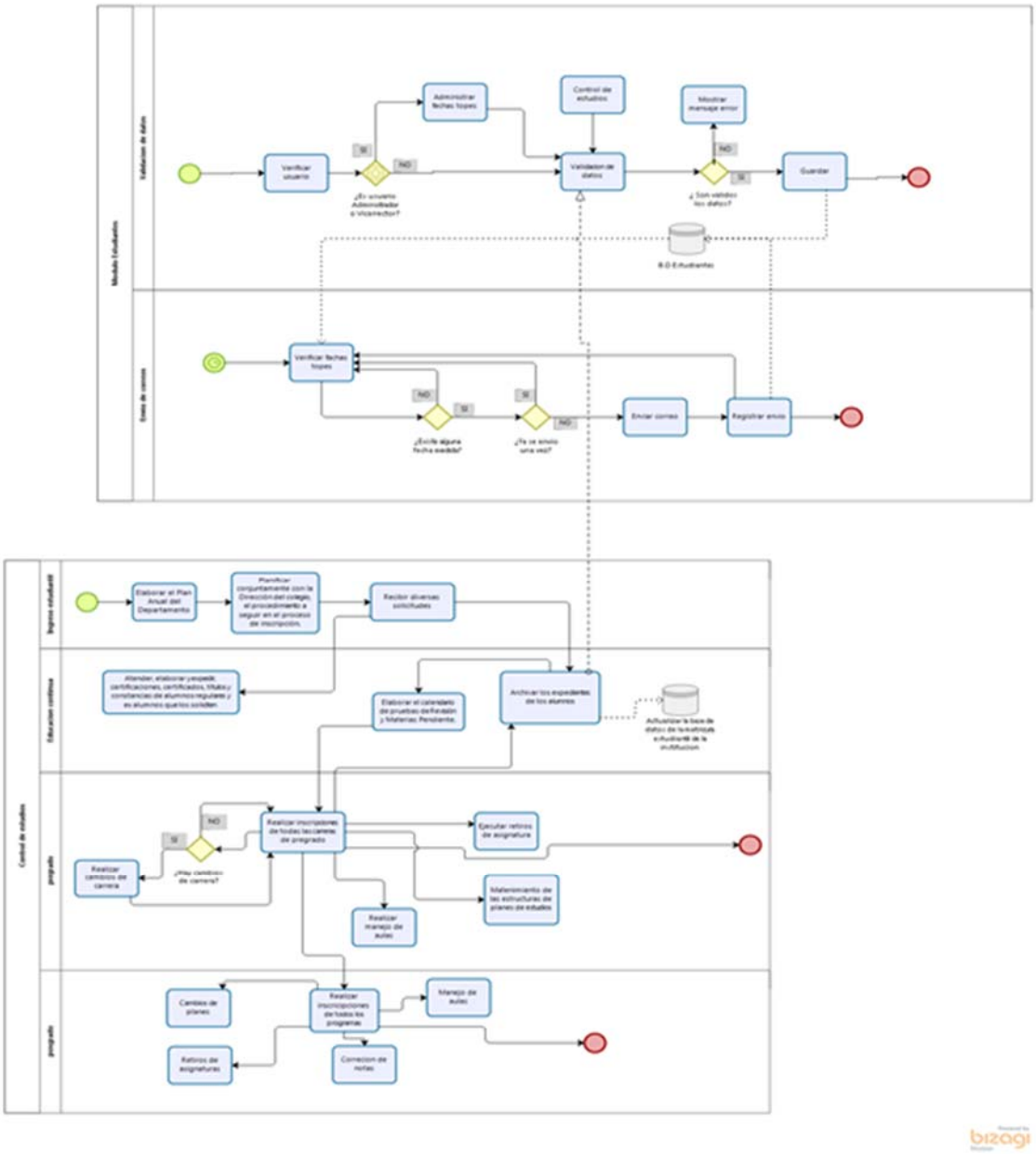

Figura 12: Modelado de procesos del módulo estudiante.

Fuente: Autor.

4.3.2.4 Requisitos funcionales

Tras la recolección y análisis de requisitos, se determinaron un conjunto de funcionalidades del sistema a desarrollar, requeridas para

dar solución al problema planteado por este proyecto, los cuales son presentados en la siguiente tabla

Tabla 4: Requisitos funcionales del sistema

Núm.	Requisito	Descripción	Dif.	Pri.
1	Creación de usuario	---El sistema debe permitir poder crear un usuario con un perfil ya definido en el sistema.	2	3
2	Modificación de datos del usuario	El sistema debe posibilitar al usuario el cambio de su contraseña de acceso al sistema desde un panel de mantenimiento de datos.	3	2
3	Carga de archivos	El sistema debe de permitir la carga de archivos asociada a la información al estudiante.	1	1
4	Realización de respaldo	En caso de posible pérdida de datos, es necesario respaldar la información de la base de datos del sistema de forma periódica.	1	1
5	Verificación de los datos	La información asociada a los estudiantes tiene que ser verificable	1	1

Fuente: Autor

4.3.2.5 Requisitos No funcionales

Además de las funcionalidades requeridas, en todo sistema existe un conjunto de requisitos que no son considerados funcionalidad del sistema, pero deben cumplirse con la finalidad de asegurar un producto de calidad, confiable, seguro, y cumplir además con características particulares.

Tabla 5: Requisitos funcionales del sistema

Núm.	Requisito	Descripción	Dif.	Pri.
1	Escalabilidad	El sistema debe poseer un diseño escalable, de manera que cambios o adiciones de nuevas funcionalidades puedan ser realizados de forma apropiada.	1	1
2	Documentación	El sistema debe tener la documentación necesaria para su uso y mantenimiento.	3	2
3	Compatibilidad	El sistema debe estar parame trizado para facilitar la comunicación con algún otro módulo. También debe ser compatible con la mayoría de los navegadores modernos existentes en la actualidad: *Google Chrome, Mozilla Firefox, Internet Explorer, Safari*, etc.	1	1
4	Validación	El sistema debe validar que el archivo ingresado, este de manera correcta	1	1
5	Usabilidad	La interfaz de usuario se diseñará de tal manera que facilite el uso de la misma y en caso de algún error del usuario, el sistema debe poder informar con un mensaje especificando el error	2	3
6	Modulación	El sistema debe de estar desarrollado para ser capaz de ser instalado como un módulo para otro sistema	1	1

Fuente: Autor

4.3.2.6 Base de datos

La base de datos del sistema contiene numerosas tablas ya que se registra una gran cantidad de información referente a los estudiantes, por lo que a continuación se nombran las tablas que contienen información relevante para

gestionar la data.

Cabe destacar que solo se hablara de las tablas de pregrado debido que las de postgrado tiene prácticamente la misma idea.

- **Estudiante**: Contiene toda la información acerca del estudiante

  Sus atributos son:

  - id: Id único autogenerado por la base de datos.
  - nacionalidad: Identifica si el estudiante es extranjero o Venezolano
  - tipo_estudio: Identifica si el estudiante es de pregrado o postgrado
  - cedula: Identificacion del estudiante
  - primer_nombre: Primer nombre del estudiante
  - segundo_nombre: Segundo nombre
  - primer_apellido: Primer apellido del estudiante
  - segundo_apellido: Segundo apellido
  - sexo: Sexo de estudiante(Solo se valida dos tipos de sexos)
  - discapacidad: Identifica algún problema físico asociado al estudiante.
  - direccion_actual : Lugar de donde reside actualmente el estudiante
  - telefono1: Telefono celular del estudiante.
  - telefono2: Telefono opcional del estudiante.
  - etnia : Pertenencia de algún pueblo indígena.
  - email: Correo del estudiante
  - edo_procedencia: Estado de donde proviene el estudiante
  - fecha_nacimiento: Fecha de nacimiento del estudiante
  - estatus: Estado del estudiante

- fecha_creacion: Fecha de creación de la tabla
- fecha_modificacion: Fecha modificación de la tabla

- **Carrera:** Contiene la información acerca de las carreras de la Universidad de Carabobo.
    - Id: Identificador de la tabla Carrera
    - Nombre: Nombre de la carrera
    - Id_facultad: identificador de la facultad
    - tipo_semestr_anno: Identifica si la carrera es por semestre o por año
    - status: Identifica el estado actual de la carrera
    - fecha_creacion: Fecha de creación de la tabla
    - fecha_modificacion: Fecha modificación de la tabla

- **Facultad:** Contiene la información acerca de las facultades de la Universidad de Carabobo.
    - Id: Identificador de la tabla Facultad
    - status: Identifica el estado actual de la facultad
    - Nombre: Nombre de la facultad
    - fecha_creacion: Fecha de creación de la tabla
    - fecha_modificacion: Fecha modificación de la tabla

- **fecha_tope_pregrado:** Contiene la información acerca de las fechas topes para las subidas de los archivos de cada facultad para la parte de pregrado de la Universidad de Carabobo.

- Id: Identificador de la tabla fecha_tope_pregrado
- Id_facultad: identificador de la facultad
- estatus: Identifica si ya fue enviado o no el correo
- fecha_tope: Fecha tope para subir el archivo
- fecha_creacion: Fecha de creación de la tabla
- fecha_modificacion: Fecha modificación de la tabla

- **estatus_estudiante:** Contiene la información acerca del estatus de los estudiantes de pregrado de la Universidad de Carabobo.
  - Id: Identificador de la tabla estatus_estudiante
  - codigo: identificador único asociado a la tabla estatus_estudiante
  - estatus: Identifica el estatus del estudiante
  - id_estudiante: Identifica el id asignado de un estudiante en la base de datos
  - fecha_creacion: Fecha de creación de la tabla
  - fecha_modificacion: Fecha modificación de la tabla

- **estudio_adicional :** Contiene la información acerca del estudio adicional de los estudiantes de pregrado de la Universidad de Carabobo.
  - Id: Identificador de la tabla estudio_adicional
  - codigo: identificador único asociado a la tabla estudio_adicional
  - descripcion: Contiene la descripción del estudio adicional del estudiante
  - id_estudiante: Identifica el id asignado de un estudiante en la base de datos

- fecha_creacion: Fecha de creación de la tabla
- fecha_modificacion: Fecha modificación de la tabla

- **estudiante_carrera:** se utiliza para relacionar la carrera y el estudiante de pregrado.
  - Id: Identificador de la tabla estudiante_carrera
  - id_estudiante: Identifica el id asignado de un estudiante en la base de datos
  - id_carrera: Identifica el id asignado de una carrera en la base de datos

A continuación se presenta el diagrama de entidad relación acerca de del módulo de los estudiantes:

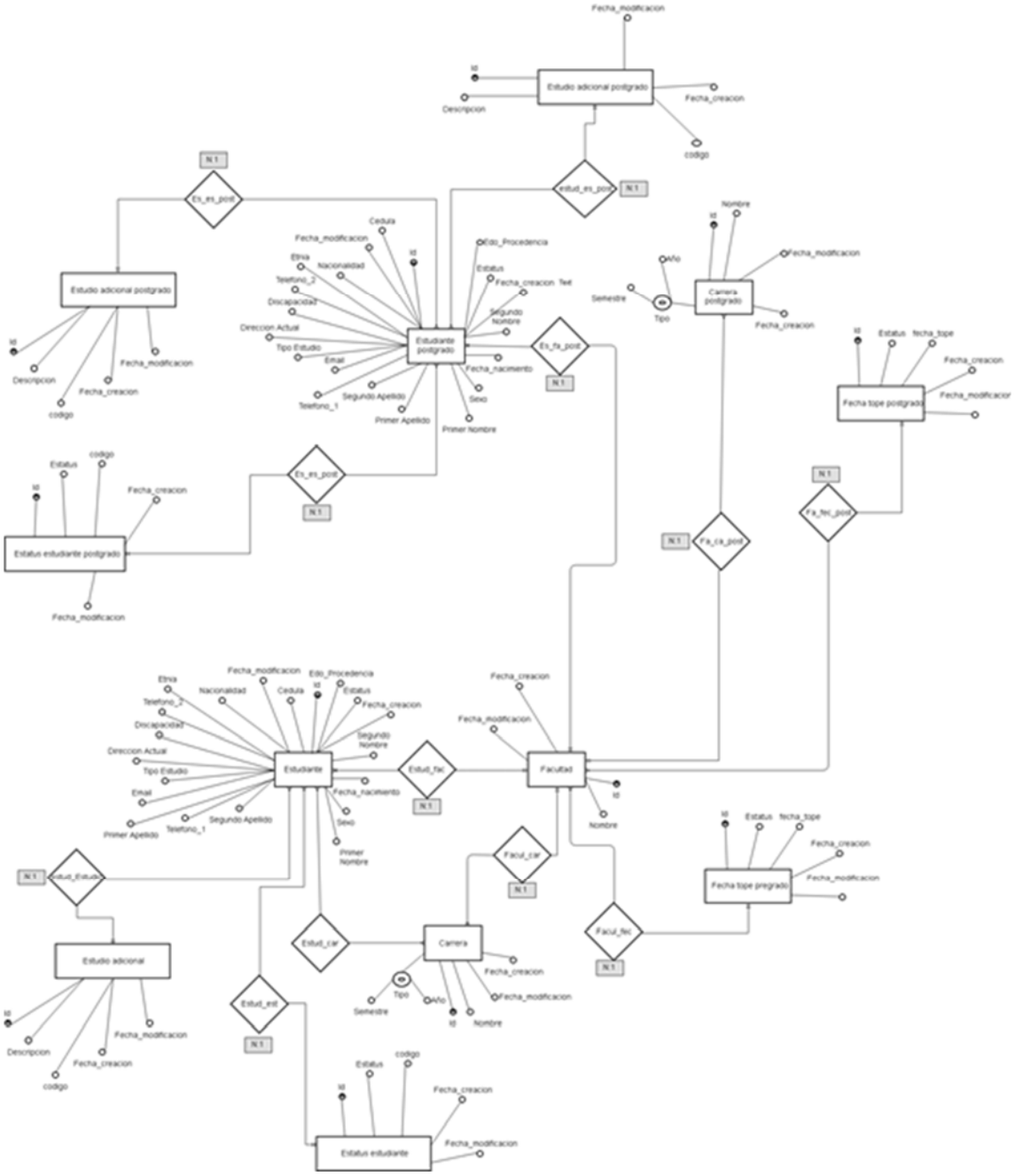

Figura 13: Diagrama de entidad relación.

Fuente: Autor.

El diagrama de la figura 13 se realizó con Draw versión 10.1.4.

4.3.3 Sprint 3:

Código: S03

Estimación: 3 semanas.

4.3.3.1 Funcionalidades del Módulo

A continuación se especificará las funcionalidades del módulo de los estudiantes que se realizó durante este Sprint. Cabe destacar que la el módulo de integración explica toda la parte de la administración de los usuario como por ejemplo, el cambio de contraseña asociado a un usuario en particular.

Tabla 6: Historia de usuario con el rol administrador.

Historia de Usuario		
ID: Hu1	**Usuario: Administrador y Vicerrector**	**Prioridad: 1**
Descripción	El usuario puede iniciar sesión	
Tareas	● Subir archivo para cualquier facultad. ● Visualizar y modificar todas las fechas topes de todas las facultades	
Observaciones	El único usuario que puede modificar las fechas topes y subir cualquier archivo para cualquier facultad, es el usuario con el rol de Administrador	

Fuente: Autor

Tabla 7: Historia de usuario con el rol analista.

Historia de Usuario		
ID: Hu2	**Usuario: Analista de control de estudios**	**Prioridad: 1**
Descripción	El usuario puede iniciar sesión	
Tareas	● Subir archivo asociado a su facultad. ● Puede visualizar la fecha límite de carga del archivo asociada a su facultad.	
Observaciones	El único usuario que puede visualizar la fecha limite asociado a su facultad y cargar el archivo de su respectiva facultad, es el usuario con el rol de Analista	

Fuente: Autor

4.3.3.2 Herramientas de software y hardware disponibles

Para la construcción del sistema, fue necesario preparar un entorno de desarrollo adecuado a las necesidades del proyecto, lo cual requirió una serie de herramientas indicadas a continuación:

4.3.3.3 Software instalado

· **Sistema Operativo**: Microsoft Windows 7 64 bits

· **Entorno de Desarrollo Integrado**: Visual Studio code

· **Software de servidor web**: Apache v2.2.27

· **Sistema de Gestor de Base de Datos**: Mysql 5.5.41 y Postgre 9.5

· **Lenguajes de Programación**: Python 3.5 y Nodejs 5.6.

· **Control de Versiones**: Se utilizó Git a partir del segundo Sprints de este trabajo.

4.3.3.4 Hardware utilizado

· **Procesador**: Intel® Celeron® 847 .caché de 2 M, 1,10 GHz

· **Memoria RAM**: 4 GB

· **Disco duro**: 320 GB

· **Tarjeta de video**: Intel(r) Hd Graphics Family

Asimismo, se establecieron unos requerimientos mínimos de hardware para el servidor mostrados en la siguiente tabla, tomando en cuenta que el sistema de gestión de solicitudes tendría poco tráfico y cantidades simultáneas de conexiones.

Tabla 8: Requerimientos de hardware para servidor web

Equipo/Requerimiento	Mínimo	Recomendado
Procesador	**DUAL CORE 2GHZ**	**Quad Core 2GHz**
Memoria	**4 GB**	**4+ GB**
Disco Duro	**10 GB**	**50+ GB**
Sistema Operativo	**Windows 7 / Linux 32 bit**	**Windows 7 64 bits/ Linux 64 bit**

Fuente: Autor

4.3.3.5 Fase de Elaboración

Para las tecnologías a usar, debido a que poseen licencia gratuita, los costos de la integración fueron nulos.

Debido a que el sistema a desarrollar es basado en web, teniendo en cuenta que la comunidad de desarrollo web cuenta con amplia documentación, el costo y tiempo invertido en el conocimiento de las herramientas fue menor.

4.3.3.6 Tecnologías utilizadas

- **Flask**: es un framework minimalista escrito en Python que permite crear aplicaciones web rápidamente y con un mínimo número de líneas de código.
- **Postgres**: Base de datos SQL, utilizada para almacenar y consultar los datos de interés.
- **JavaScript**: Lenguaje de programación ejecutado en el agente de usuario, utilizada para habilitar interacción con la interfaz gráfica.
- **JQuery**: Librería de JavaScript, utilizada para complementar funcionalidades de JavaScript.
- **HTML5 & CSS3**: Tecnología utilizada para diseñar la interfaz de usuario mostrada en el navegador.
- **Bootstrap**: Framework popular para HTML, CSS y JavaScript para diseñar páginas web responsivas.
- **Bizagi**: es un *Freeware* utilizado para diagramar, documentar y simular procesos usando la notación estándar BPMN.

4.3.3.7 Limitaciones

- El servidor debe de tener como mínimo 4 Gb de Ram con dos más

procesadores multinúcleo de al menos 2,0 Ghz

- La entrega de datos por parte de control de estudios debe de estar correcta y completa.
- Se tiene que tener instalado Python ver 3,5 o superior y Nodejs ver 5,6 o superior. Para la base de datos se necesita tener instalado Postgres en la versión 9.5.X y Mysql en cualquier versión

4.3.3.9 Pruebas

4.3.3.9.1 Prueba de volumen

En el siguiente cuadro se realizará una prueba de volumen para la carga de archivos. Solo se realizarán tres pruebas y a partir de allí. Se tomará un aproximado de cuánto tiempo puede durar a medida que se incrementan los registros. Es importante aclarar que esta prueba será llevada a cabo con un procesador Celeron Dual Core modelo 847 de 1,1Ghz, 4 Gb de Ram y un disco duro de 320 GB.

Tabla 9: Prueba de volumen para la carga de archivos

Cantidad de registros	Tiempo
20	4 seg. aprox
288	6 seg. aprox
650	8 seg. aprox
4600	30 seg. aprox
10000	1 minuto con 10 seg aprox

Fuente: Autor

4.3.3.9.2 Prueba de compatibilidad

Uno de los objetivos principales de realizar el desarrollo del sistema en base a un ambiente web, se debe al deseo de permitir la mayor cantidad de usuarios a utilizar la aplicación, independientemente de la plataforma de sus computadoras. Por ello, se realizó la tarea de probar de forma exhaustiva la compatibilidad con los agentes de usuario.

Tabla 10: Prueba de compatibilidad browser

Agente de usuario	Compatibilidad
Google Chrome	Sí
Firefox	Sí
Internet Explorer	No

Fuente: Autor

4.3.3.9.3 Prueba de usabilidad y accesibilidad

La usabilidad y accesibilidad es un factor importante de diferenciación en las aplicaciones de software. Las pruebas de usabilidad son de mucha importancia ya que los usuarios buscan mayor satisfacción respecto a facilidad de uso, navegabilidad, adaptabilidad, simplicidad y estética. Para la evaluación de la usabilidad y accesibilidad de la plataforma se realizó un estudio con las diferentes alternativas online gratuitas, tomando en cuenta los siguientes aspectos:

- Equilibrio entre diseño e información.
- Cuidado de los enlaces (sin enlaces rotos).

- Llamadas a la acción.
- Diseño de calidad y coherente.
- Legibilidad.
- Uso de espacios en blanco.
- Interfaz sencilla e intuitiva.
- Acceso al contenido en el menor número de clicks.
- Adaptación según los estándares de accesibilidad publicados por el World Wide Web Consortium (la W3C).

Entre las diferentes alternativas en el internet las gratuitas más populares por son:

- https://validator.w3.org/
- http://quirktools.com/screenfly/
- http://nibbler.silktide.com
- http://examinator.ws/

Seguidamente se concluyó que utilizar http://nibbler.silktide.com era la opción más adecuada ya que evaluaba la mayor cantidad de opciones de forma gratuita, tomando en cuenta los aspectos señalados anteriormente. A continuación se muestran unas series de imágenes donde se muestra un nivel de puntuación del 1.0 al 10.0 de la usabilidad del sistema UC RANKING:

Figura 14: Puntuación general del sistema UC RANKING.
Fuente: Autor

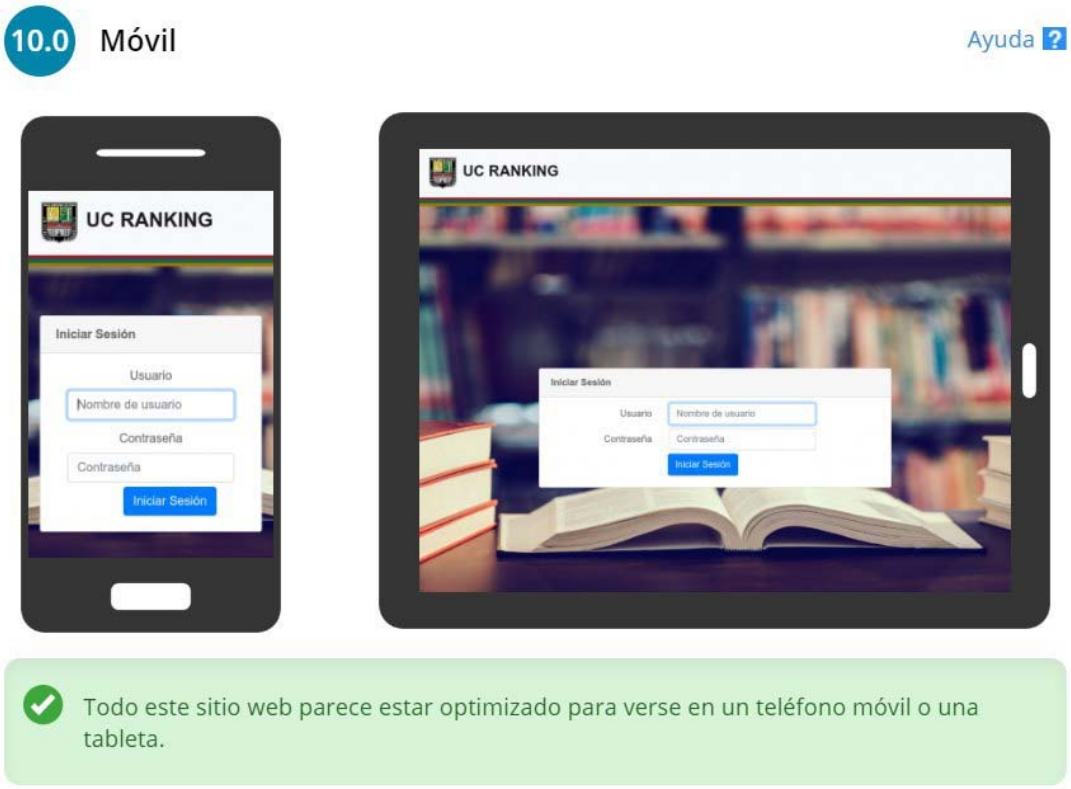

Figura 15: Puntuación Response del sistema UC RANKING.
Fuente: Autor

4.3.3.9.4 Prueba de rendimiento

El rendimiento se refiere a la efectividad de la plataforma y a la rapidez de respuesta que muestra cuando los usuarios solicitan alguna acción. En otras palabras,

podemos decir que el rendimiento está asociado íntimamente al tiempo que se demora en cargar los distintos elementos que conforman una vista concreta de la plataforma. Para la evaluación de rendimiento de la plataforma se realizó un estudio con las diferentes alternativas online gratuitas, tomando en cuenta los siguientes aspectos:

- Tamaño de la página.
- Peticiones de la página.
- Velocidad de la página.
- Caché del navegador.
- Re direccionamientos de la página.
- Compresión.
- Bloqueo de renderizado.

Entre las diferentes alternativas en el internet las gratuitas más populares por son:

- https://tools.pingdom.com/
- https://gtmetrix.com/
- https://www.dareboost.com/en/
- https://developers.google.com/speed/pagespeed/insights/
- http://examinator.ws/

Se seleccionó https://www.dareboost.com/en/ debido a la sencillez con la que muestra los resultados, así como para utilizar la herramienta, sin ningún tipo de registro. A continuación se muestra una figura donde se evalúa el rendimiento del sistema UC RANKING:

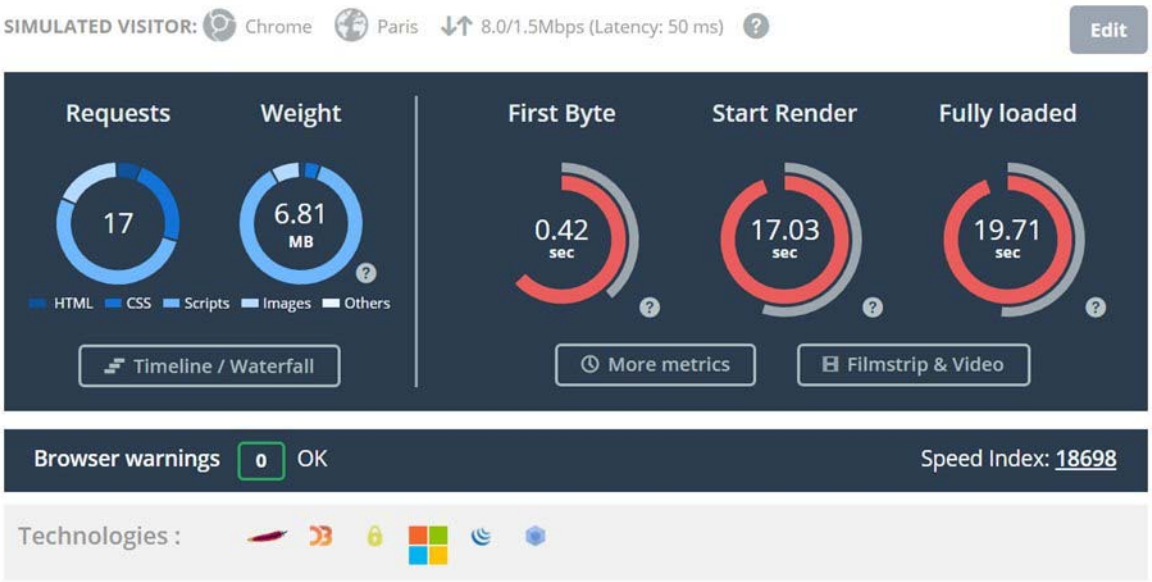

Figura 16: Rendimiento de sistema UC RANKING vía web.
Fuente: Autor

4.3.4 Sprint 4:

Código: S04

Estimación: 2 semanas.

4.3.4.1 Integración

A continuación se presenta una estandarización de ejemplo con la finalidad de hacer la comunicación con el módulo de integración.

También se presenta una estandarización de ejemplo de dos APIS, una para carga y otra para actualización, con la finalidad de hacer la comunicación con el módulo de integración, este es un microservicio que solo es utilizado por el modulo antes mencionado.

Carga:

ruta: /api/v1/estudiantes

Actualización:

ruta: /api/v1/estudiantes/<fecha_actualiazacion>


```
{
  "hechos-estudiante-carrera-facultad": {
    "items": [
      {
        "estudiante": "26011707",
        "carrera": "Computación",
        "facultad": "Ciencias y Tecnología"
      },
      {
        "estudiante": "22422883",
        "carrera": "Computación",
        "facultad": "Ciencias y Tecnología"
      },
      {
        "estudiante": "27855129",
        "carrera": "Química",
        "facultad": "Ciencias y Tecnología"
      },
      {
        "estudiante": "13381615",
        "carrera": "Física",
        "facultad": "Ciencias y Tecnología"
      },
      {
        "estudiante": "22345223",
        "carrera": "Matemática",
        "facultad": "Ciencias y Tecnología"
      },
      {
        "estudiante": "22345243",
        "carrera": "Matemática",
        "facultad": "Ciencias y Tecnología"
      }
    ]
  },
  "dim-facultad": {
    "items": [
      {
```

```
            "nombre": "Ingeniería"
        },
        {
            "nombre": "Ciencias Jurídicas y Políticas"
        },
        {
            "nombre": "Ciencias de la Salud"
        },
        {
            "nombre": "Ciencias Económicas y Sociales"
        },
        {
            "nombre": "Ciencias de la Educación"
        },
        {
            "nombre": "Odontología"
        },
        {
            "nombre": "Dirección General de Postgrado"
        },
        {
            "nombre": "Ciencias y Tecnología"
        }
    ]
},
"dim-carrera": {
    "items": [
        {
            "nombre": "Computación",
            "tipo": "Semestre"
        },
        {
            "nombre": "Química",
            "tipo": "Semestre"
        },
        {
            "nombre": "Física",
            "tipo": "Semestre"
        },
        {
```

```
            "nombre": "Matemática",
            "tipo": "Semestre"
        },
        {
            "nombre": "Biología",
            "tipo": "Semestre"
        }
    ]
},
"dim-estudiante": {
    "items": [
        {
            "cedula": "22422883",
            "nacionalidad": "Venezolano",
            "nombre": "Wilkel",
            "apellido": "Apellido",
            "sexo": "Masculino",
            "fecha_nacimiento": "1995-05-24",
            "telefono1": "0412-76558802",
            "telefono2": "0245-3351406",
            "correo": "wilkelgiovanni@gmail.com",
            "edo_procedencia": "Carabobo",
            "etnia": "NO PERTENEZCO A UN PUEBLO INDÍGENA",
            "discapacidad": "NO POSEO NINGUNA DISCAPACIDAD",
            "status": 1,
            "tipo": "1"
        },
        {
            "cedula": "27855129",
            "nacionalidad": "Venezolano",
            "nombre": "Ana",
            "apellido": "Sanchez",
            "sexo": "Femenino",
            "fecha_nacimiento": "1999-09-22",
            "telefono1": "0241-8481233",
            "telefono2": "0426-3437317",
            "correo": "anasanchez@gmail.com",
            "edo_procedencia": "Carabobo",
            "etnia": "NO PERTENEZCO A UN PUEBLO INDÍGENA",
            "discapacidad": "NO POSEO NINGUNA DISCAPACIDAD",
```

```
            "status": 1,
            "tipo": "1"
        },
        {
            "cedula": "26011707",
            "nacionalidad": "Venezolano",
            "nombre": "Alba",
            "apellido": "Silva",
            "sexo": "Femenino",
            "fecha_nacimiento": "1997-03-01",
            "telefono1": "0241-2051334",
            "telefono2": "0412-1308522",
            "correo": "andreadellepere_3@hotmail.com",
            "edo_procedencia": "Carabobo",
            "etnia": "YANOMAMI",
            "discapacidad": "SI POSEO DISCAPACIDAD",
            "status": 1,
            "tipo": "1"
        },
        {
            "cedula": "13381615",
            "nacionalidad": "Venezolano",
            "nombre": "Luis",
            "apellido": "Servita",
            "sexo": "Femenino",
            "fecha_nacimiento": "1976-07-07",
            "telefono1": "02418140120",
            "telefono2": "04265413615",
            "correo": "luisservita777@gmail.com",
            "edo_procedencia": "Carabobo",
            "etnia": "NO PERTENEZCO A UN PUEBLO INDÍGENA",
            "discapacidad": "NO POSEO NINGUNA DISCAPACIDAD",
            "status": 2,
            "tipo": "2"
        },
        {
            "cedula": "22345243",
            "nacionalidad": "Venezolano",
            "nombre": "Alejandro2",
            "apellido": "Giovanni2",
```

```
            "sexo": "Masculino",
            "fecha_nacimiento": "1995-05-24",
            "telefono1": "0215545",
            "telefono2": "155455515",
            "correo": "alejandro2@gmail.com",
            "edo_procedencia": "Carabobo",
            "etnia": "TIMOTO-CUICAS/TIMOTES",
            "discapacidad": "NO POSEO NINGUNA DISCAPACIDAD",
            "status": 1,
            "tipo": "2"
        },
        {
            "cedula": "22345223",
            "nacionalidad": "Venezolano",
            "nombre": "Alejandro",
            "apellido": "Giovanni",
            "sexo": "Masculino",
            "fecha_nacimiento": "1995-05-24",
            "telefono1": "0215545",
            "telefono2": "155455515",
            "correo": "alejandro@gmail.com",
            "edo_procedencia": "Carabobo",
            "etnia": "NO PERTENEZCO A UN PUEBLO INDÍGENA",
            "discapacidad": "SI POSEO DISCAPACIDAD",
            "status": 2,
            "tipo": "1"
        }
    ]
  }
}
```


Formato JSON para la comunicación con el módulo de integración.

4.3.4.2 Historias de usuario

Las siguientes son las historias de usuario que se elaboraron para el sistema, el orden de éstas no es relacionado con su prioridad.

1. El usuario puede iniciar sesión en el sistema.
2. El usuario puede administrar usuarios.
3. El usuario puede cargar archivos con información de docentes y estudiantes.
4. El usuario puede modificar las fechas límites de carga de archivos para docentes y estudiantes.
5. El usuario puede consultar las fechas de últimas actualizaciones de información docente y estudiantes.
6. El usuario puede configurar parámetros del sistema.
7. El usuario puede visualizar las facultades.
8. El usuario puede visualizar las carreras.
9. El usuario puede visualizar las acciones de los usuarios.
10. El usuario puede ver el monitoreo de microservicios.
11. El usuario puede visualizar los reportes.
12. El usuario puede descargar reportes.
13. El sistema debe buscar automáticamente las citaciones en Google scholar

4.3.4.3 Lista del producto (*Product backlog*):

A continuación se mostrara una tabla donde fue construida basada en las reuniones hechas con anterioridad, en el mismo se listan las tareas que se requieren para realizar funciones en el sistema, la descripción de las tareas están descritas en alto nivel y se detallaron a medida que se avanzó en el desarrollo y están descritas con más detalle en cada tabla de historia individualmente. La Estimación se refiere a una escala de 1 a 2 para el número de iteraciones o sprints

que se requieren en la construcción de la funcionalidad, mientras que la prioridad, que se basa en el valor de urgencia que tiene cada historia en el desarrollo para el proyecto, del 1 al 6 por número de iteración.

Tabla 11: Lista producto.

Nro.	Nombre	Tareas	P
HU1	El usuario puede iniciar sesión en el sistema.	Creación de módulo de autentificación: • Definición de usuarios. • Definición de roles. • Creación de vista para el inicio de sesión.	1
HU2	El usuario puede administrar usuarios.	Creación de vistas de: • Listado de usuarios creados en el sistema. • Creación de usuarios. • Modificación de usuarios. • Inactivación de usuarios.	1
HU3	El usuario puede cargar archivos con información de estudiantes.	• Creación del módulo • Creación de la vista de carga de archivos con información de estudiantes.	1
HU4	El usuario puede modificar las fechas límites de carga de archivos para estudiantes.	• Creación del módulo • Creación de la vista de fechas límites.	1
HU5	El usuario puede	• Creación del módulo	2

	consultar las fechas de últimas actualizaciones de información estudiantes.	• Creación de la vista de fechas de última actualización.	
HU6	El usuario puede cargar archivos con información de docentes.	• Creación del modulo Creación de la vista de carga de archivos con información de docentes.	1
HU7	El usuario puede modificar las fechas límites de carga de archivos para docentes.	• Creación del módulo • Creación de la vista de fechas límites.	1
HU8	El usuario puede consultar las fechas de últimas actualizaciones de información docente.	• Creación del módulo • Creación de la vista de fechas de última actualización.	2
HU9	El usuario puede visualizar las acciones de los usuarios.	• Creación de vistas donde se visualiza un listado de las acciones que realiza el usuario en el sistema. • Implementación de acción que permita al usuario poder descargar el listado de acciones. • Buscador de acciones de los usuarios. • Implementación de acción para limpiar el listado de acciones.	1
HU10	El usuario puede configurar parámetros	Creación de vistas donde el usuario pueda:	1

	del sistema.	• Visualizar el listado de parámetros Modificar los parámetros del sistema.	
HU11	El usuario puede visualizar las facultades.	Creación de vistas para los formularios de creación e inactivación de facultades.	2
HU12	El usuario puede visualizar las carreras.	Creación de vistas para los formularios de creación e inactivación de facultades.	2
HU13	El usuario puede ver el monitoreo de Microservicios.	• Creación de vistas la cual se aprecia un listado de actividades que realizan los Microservicios. • Activación y desactivación de tareas programadas. • Implementación de acción que permita al usuario poder descargar el listado de actividades. • Buscador de actividades de los usuarios. • Implementación de acción para limpiar el listado de actividades.	1
HU14	El usuario puede visualizar los reportes.	Creación del módulo de reportes: • Creación de vista del módulo. • Creación de interfaz para la visualización de los reportes.	1
HU15	El usuario puede descargar reportes.	• Creación de vista. • Implementación de acciones	1

		que permitan al usuario descargar reportes.	

Fuente: Autor

4.4 Evaluación

En el punto anterior de la fase acción de la metodología Investigación/acción se pudo observar los resultados obtenidos de la metodología de desarrollo llamada Scrum. Cabe destacar que en la metodología de desarrollo se aplicó en cuatro sprint, el sprint más complejo fue el segundo.

4.5 Reflexión

Para cerrar el ciclo metodológico Investigación-Acción, se presentan los resultados obtenidos.

4.5.1 Manual de Programador

Con la finalidad de documentar todos los aspectos de desarrollo del sistema y permitir mejor entendimiento a otros desarrolladores al momento de contribuir con el proyecto, se desarrolló un manual de programador incluido como anexo en este documento.

4.5.2 Manual de Usuarios

Teniendo en cuenta que el sistema cuenta con diferentes interfaces para distintos usuarios, se desarrolló un manual de usuarios anexo a este documento,

donde se explica el flujo de interacción y funcionalidades en el sistema para cada uno de los usuarios.

4.5.3 Manual de Instalación

En el anexo c de este documento, se explica detalladamente la instalación del sistema UC RANKING. Debe de seguir rigurosamente los pasos explicado en este manual para tener ejecución del software sin fallas.

4.5.4 Resultados

Tras haber desarrollado el sistema de UC RANKING, se cumplieron todos los objetivos planteados:

• Se realizó una investigación previa referente a los sistemas de ranking, con la finalidad de establecer una base teórica indicados en el capítulo II.

• Se generó un modelado del proceso de negocios como primera fase de la metodología de desarrollo utilizada.

• Se logró implementar la base de datos sobre el cual se basa el sistema.

• Se implementaron vistas para cada tipo de usuario existente en el sistema, a través de la cual pueden interactuar de manera fluida e intuitiva con el sistema.

• Se integró exitosamente la comunicación con el sistema de los controles de estudios de la UC, con la finalidad de obtener información de los estudiantes.

• Se llevaron a cabo pruebas de desempeño del software, permitiendo así determinar la calidad del software desarrollado de forma exitosa.

• Se redactaron manuales de usuarios, programador e instalación.

4.5.5 Conclusiones

Tras culminar con el proceso de desarrollo, se obtuvo un software capaz de proporcionarle información al módulo de integración del sistema UC RANKING. Esto le permite al módulo de integración administrar la data de los estudiantes de pregrado y postgrado mediante un data warehouse. Asimismo el módulo implementado en esta sección puede enviar correos de manera automática cuando exista una fecha tope de subida de archivo retrasada.

Actualmente existe una lucha entre las universidades por estar mejor posicionadas en el Ranking, este módulo realizado en esta tesis, proporciona una contribución al sistema UC RANKING, lo cual este sistema realizará una mejor divulgación de la información básica o particular del estudiante de la Universidad de Carabobo, lo cual contribuirá en un mejor posicionamiento en el ranking de las universidades.

Los experimentos con datos reales que se propusieron realizar, no se realizaron por limitaciones de tiempo. Los cuales se consideran valiosos para el punto de vista de los instructores sobre la utilidad de la herramienta y su eficacia.

Para cerrar, este trabajo le proporciona un microservicio al módulo de integración, donde se le entrega mediante una API, la información completa de todas las facultades cuando este se ha llamado por el módulo de integración.

4.5.6 Recomendaciones

A continuación se enumeran una serie de recomendaciones cuya implementación son vitales para mejorar la operatividad del sistema implementado en este trabajo:

Realizar pruebas de rendimiento para verificar si es necesario realizar algún cambio en la sesión del código de validación del archivo.

Como recomendación es necesario hacer pruebas de estrés del software desarrollado en esta tesis con la finalidad de conocer la magnitud que podría soportar el servidor.

Para la instalación del software se recomienda instalar todo lo descrito en este documento, incluyendo la versión de cada uno de estos, para así evitar problemas de compatibilidad.

Referencias bibliográficas:

- Marlo M. Vernon, E. Andrew Balas, Shaher Momani. (2018)¿Las clasificaciones universitarias son útiles para mejorar la investigación? Georgia, Estados Unidos de América.
- Juan Briceño R (2014). Reflexiones sobre la educación superior en Venezuela. Maracay, Venezuela.
- Abraham Toro y Luzmila Marcano. (2007). Calidad y Educación Superior Venezolana. Valencia, Venezuela.
- Francisco López (2008). Tendencias de la educación superior en el mundo y en América Latina y el Caribe. Paris, Francia.
- Amalia Yoguez (2009). ¿Cómo se evalúan las universidades de clase mundial? Ciudad de México, México.
- Felipe Martínez (2011). Los rankings de universidades: una visión crítica. Ciudad de México, México.
- Santos López (2012). Los rankings universitarios. Bases teóricas, metodología y su impacto en la educación superior global. Ciudad de México, México.
- Mario Albornoz y Laura Osorio (2017). Uso público de la información: el caso de los rankings de universidades. Buenos Aires, Argentina.
- James Lewis y Martin Fowler (2014). Microservicios. Una definición de este nuevo término arquitectónico. Chicago, E.E.U.U.
- Colmenares, A. M. (2012). Investigación-acción participativa: una metodología integradora del conocimiento y la acción.
- Hernández, Fernández, & Baptista. (2010). Metodología de la

Investigación. IBM. (2016). What is cloud storage? Obtenido de IBM Cloud: www.ibm.com/cloud-computing/learn-more/what-is-cloud-storage/

- Elvia Gómez, Diana Jumbo y Adriana Ríos (2012). : Desarrollo de un sistema automatizado para la inscripción y mantenimiento de información de alumnos vía internet, del colegio Primero de Mayo del cantón Yanzatza. Ecuador.
- Molina, J., Loja, M., Zea, M., & Loaiza, E. (2016). Evaluación de los Frameworks en el Desarrollo de Aplicaciones Web con Python.
- Nations, D. (2016). What is a Web Application? Obtenido de Lifewire: https://www.lifewire.com/what-is-a-web-application-3486637
- Velásquez, C., & Cárdiz, A. (2011). Metodos Cualitativos Investigación Acción. Barquisimeto: UNIVERSIDAD YACAMBÚ.
- Glenford J. Myers, John Wiley & Sons. The Art of Software Testing, Second Edition, Inc., 2004
- Softeng. Metodología Scrum. Madrid, España. Obtenido de softeng : https://www.softeng.es/es-es/empresa/metodologias-de-trabajo/metodologia-scrum.html
- Xavier Albaladejo. (2008). Qué es Scrum? Obtenido de proyectosagiles : https://proyectosagiles.org/que-es-scrum/
- Saisana M, d'Hombres B, Saltelli A (2011) Rickety numbers: Volatility of university rankings and policy implications. Research Policy.
- Taylor P, Braddock R (2007) International University Ranking Systems and the Idea of University Excellence. Journal of Higher Education Policy and Management.
- Moed HF (2017) A critical comparative analysis of five world university

rankings. Scientometrics.

- Woodman y White, (1985). Gestión de la Información. London.

  Arquitectura de software. Obtenido de Ecured:

  https://www.ecured.cu/Arquitectura_de_software
- Khristopher Perdomo (2017). Implementación de un sistema de gestión de solicitudes de préstamos personales para el Instituto de Previsión Social del Personal Docente y de Investigación de la Universidad de Carabobo. Valencia, Venezuela.
- Susana y Alma, 2013. Rankings de las mejores 100 universidades del mundo. México.
- Wilker Giovanni, 2019. SISTEMA DE GESTIÓN DE INFORMACIÓN DE LA UNIVERSIDAD DE CARABOBO PARA RANKINGS UNIVERSITARIOS, UTILIZANDO ARQUITECTURA DE MICROSERVICIOS. CASO DE ESTUDIO: INTEGRACIÓN DE LA INFORMACIÓN. Valencia. Venezuela.

Anexo

Anexo A. Manual de programador

Tecnologías utilizadas

El desarrollo del *backend* de este sistema se realizó utilizando el lenguaje de programación Python en su versión 3.5 a través del *Framework* Flask version 1.0.2.

Por otro lado, el desarrollo del *frontend* se realizó utilizando el lenguaje de programación JavaScript, a través del Framework Vuejs versión 2.5.2, el cual es un proyecto derivado del Framework Vuejs para JavaScript, agregándole un conjunto de servicios y directivas basadas enteramente en *Material Design*. Adicionalmente, se hizo uso de múltiples Plugins y librerías para incrementar robustez y disminuir el tiempo de desarrollo, los cuales se describen a continuación:

Plug-ins y librerías

A continuación se listan los *plugin* y librerías utilizadas para el desarrollo del backend y el *frontend*, con una breve descripción de cada una.

Para el *backend*, se utilizaron las siguientes librerías:

- **Flask, make_response, request** son para dar respuestas y hacer peticiones desde o al *frondend*.
- **flask_cors** es una extensión de Flask para el manejo de recursos compartidos de origen cruzado (CORS), que hace posible el AJAX de

origen cruzado

- **flask_restful** es una extensión para Flask que agrega soporte para la creación rápida de API REST. Es una abstracción ligera que funciona con sus ORM / bibliotecas existentes. Flask-RESTful fomenta las mejores prácticas con una configuración mínima.
- **os** es un módulo que proporciona una forma portátil de utilizar la funcionalidad dependiente del sistema operativo.
- **werkzeug.utils** es una completa biblioteca de aplicaciones web WSGI (interfaz de la puerta de enlace del servidor web)
- **simplejson** es utilizado para llevar el control del formato de salida
- **csv, operator** son utilizados para el manejo de los archivos de entrada que solo acepta el formato csv
- **sys** es un módulo que proporciona acceso a algunas variables utilizadas o mantenidas por el intérprete ya funciones que interactúan fuertemente con el intérprete. Siempre está disponible.
- **types** es un módulo que define nombres para algunos tipos de objetos que utiliza el intérprete estándar de Python, pero no para los tipos definidos por varios módulos de extensión
- **psycopg2** es un adaptador PostgreSQL para el lenguaje de programación Python. Es un contenedor para libpq.
- **requests** es una biblioteca HTTP con licencia Apache2
- **re** es una librería para expresiones regulares. Con esta librería se valida el correo
- **datetime** es una librería para el manejo de la fecha.

Para el *frontend*, se utilizaron las siguientes librerías y *plugin*:

- **ajv-keywords: "^ 3.2.0**': Es basado en la *typeof* operación de JavaScript.
- **axios: "^0.18.0'**: Cliente HTTP basado en la promesa para el navegador
- **jquery: "^3.3.1'**: Biblioteca de JavaScript para operaciones DOM
- **jspdf: "^1.4.1'**: Creación de documentos PDF desde JavaScript
- **npm: "^6.4.1'** :gestor de paquetes, el cual hará más fáciles nuestras vidas al momento de trabajar con NodeJs.
- **plotly.js: "^1.41.3'**: Envoltorio simple node.js para la API plot.ly
- **popper.js: "^1.14.4'**: Automatización en tiempo real de navegador cruzado
- **to: "^0.2.9'** :Carga / conversión entre formatos xml, json, yaml
- **update: "^0.7.4'***: Update* es un nuevo marco de desarrollo de código abierto y CLI para automatizar actualizaciones de cualquier tipo en proyectos de código.
- **vue: "^2.5.2'**: Capa de vista reactiva orientada a componentes para interfaces web modernas
- **vue-axios: "^2.1.3'**: Pequeña envoltura para integrar axios a Vuejs
- **vue-localstorage: "^0.6.2'**: Vue.js localStorage *plugin* con soporte de tipos
- **vue-router: '^3.0.1'**: Se integra profundamente con el núcleo de Vue.js para facilitar la creación de aplicaciones de una sola página con Vue.js
- **vuex: '^3.0.1'**: Gestión estatal para vue.js

- **xlsx: "^0.14.0"**: Analizador y escritor para varios formatos de hoja de cálculo

Distribución de archivos fuentes

De manera tradicional el *frontend* y el *backend* están separados en distintos directorios. A continuación se mostrará dos imágenes para la estructura de archivos, una para el *frontend* y la otra para el *backend* respectivamente.

Figura 17: Estructura de archivo para el *frontend.*
Fuente: Autor.

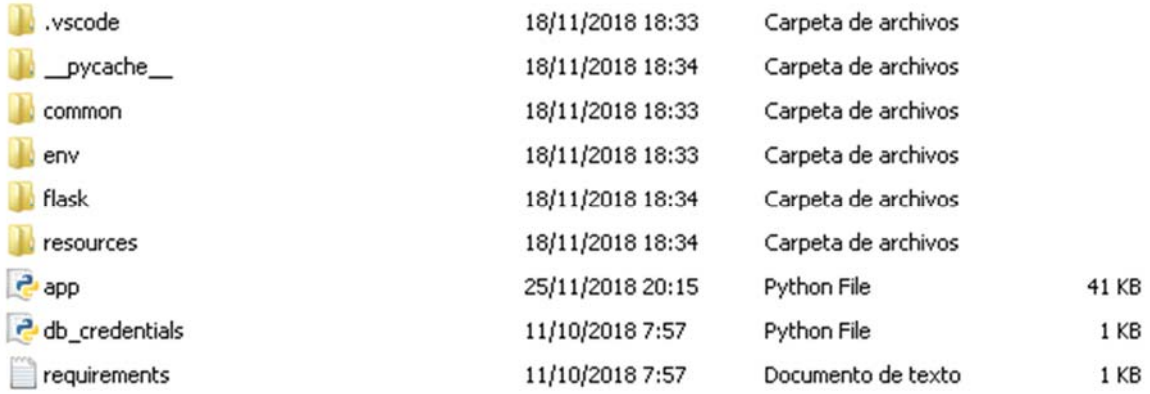

Figura 18: Estructura de archivo para el backend.

Fuente: Autor.

Cabe mencionar que para el backend hay dos estructuras de archivos iguales, lo cual uno es para la carga de archivo y el otro es para el envío de correos. No se quiso un incluir el envío de correos en la estructura de archivos de carga, por la razón de que es un proceso que se ejecuta en un intervalo de tiempo muy corto, lo cual podría estropear una llamada del frontend en un instante determinado

Para el ***frontend*** esta estructura se crear con la instalación del *webpack*, en este capítulo no se hablara de este punto. Lo que sí es importante mencionar en este capítulo, es hablar del componente “students” ubicado en la src “src\components”, allí se encuentra el archivo llamado “Students.vue”, que es el encargado de controlar y enviar las peticiones al *backend* de este módulo. Ese archivo se divide en tres secciones, uno para el código html, otro para el código de Javascript y el otro para el código Css. En el código Html se utiliza un framework llamado Bootstrap allí se define ciertos condicionales para validar el tipo de usuario ingresado en el sistema. Cabe destacar que el módulo de integración de este sistema UC RANKING explica el manejo de los usuarios. En el código en Javascript del archivo antes

mencionado, se realizan las peticiones al servidor y dependiendo de las respuestas, se muestra un resultado. El código Css es para darle una mejor forma a la vista.

Para el ***Backend*** el app.py es uno de los archivo más importante de todos, allí está todo el contenido para hacer la validación del archivo ingresado. A continuación se mostrará una foto de las validaciones que se hacen en las columnas del archivo:

```
    if (reg.get("a_ci")):
        if not (reg['a_ci'].isdigit()):
            resul.append("En la columna a_ci, fila '{}' hay un campo malo".format(cont))
    else:
        if not ("No se encuentra la columna: a_ci\n" in resul ):
            resul.append("No se encuentra la columna: a_ci\n")

    if (reg.get("primer_nombre")):
        if not (type(reg['primer_nombre'])==str):
            resul.append("En la columna primer_nombre, fila '{}' hay un campo malo".format(cont))
    else:
        if not ("No se encuentra la columna: primer_nombre" in resul ):
            resul.append("No se encuentra la columna: primer_nombre")

    if (reg.get("segundo_nombre")):
        if(reg['segundo_nombre']==""):
            reg['segundo_nombre']= "NO EXITE"
        if not (type(reg['segundo_nombre'])==str):
            resul.append("En la columna segundo_nombre , fila '{}' hay un campo malo".format(cont))
    else:
        if not ("No se encuentra la columna: segundo_nombre\n" in resul ):
            resul.append("No se encuentra la columna: segundo_nombre\n")

    if (reg.get("primer_apellido")):
        if not (type(reg['primer_apellido'])==str):
            resul.append("En la columna primer_apellido, fila '{}' hay un campo malo".format(cont))
    else:
        if not ("No se encuentra la columna: primer_apellido\n" in resul ):
            resul.append("No se encuentra la columna: primer_apellido\n")
```

Figura 19: Fragmento de validación del archivo ingresado por control de estudios

Fuente: Autor

También se mostrará una foto del vector llamado orden_columna que hay que modificar a la hora de querer hacer una modificación o agregar columnas del archivo Csv a ser leído:

Figura 20: Fragmento del vector para validar las columnas del archivo.

Fuente: Autor.

Estos son los puntos más importantes para el *backend* de la parte de subida del archivo. Para el *backend* de la parte de envío de correo, solo es de importancia mostrar la sección en donde se puede modificar el mensaje que se enviará de forma automática.

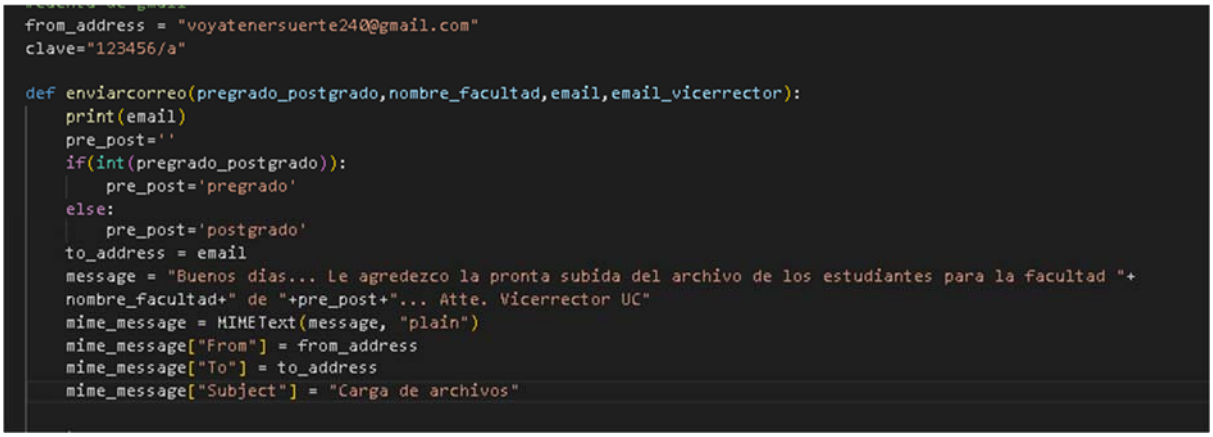

Figura 21: Fragmento del vector para la configuración de correo.

Fuente: Autor.

Anexo B. Manual de Usuario

A continuación se mostrará una imagen de la vista para subir el archivo para el caso del analista de control de estudios:

Figura 22: Vista para el analista.
Fuente: Autor.

Solo se tiene que pulsar el botón de seleccionar archivo, allí se le abrirá otra ventana donde podrá seleccionar el archivo en formato csv, en el panel 'RANKING' debe de seleccionar si es de pregrado o postgrado, y luego procede a pulsar el botón de 'enviar'. Para el caso de que sea de postgrado y el archivo contenga carrera de pregrado, el sistema automáticamente lo detectará y le dará un error. Si hay un error el sistema le proporcionará automáticamente un archivo con el error indicado, de donde podrá descargarlo y revisarlo. También puede observar la fecha tope que le fue asignada para enviar el archivo. Si se pasa de esa fecha y no ha subido el archivo solicitado, el sistema le enviara un correo el cual fue registrado en la configuración de usuario.

También se mostrará una imagen de la vista para configuración de fechas para el caso del administrador y el vicerrector. Es importante decir que la vista de cargar

archivo es muy parecida a la vista del analista a diferencia que, pueden ver todas las fechas topes de todas las facultades en otra vista aparte.

Figura 23: Vista para la configuración de fechas del vicerrector o administrador. Fuente: Autor.

Anexo C. Manual de Instalación

Instalación del Sistema UC RANKING

Frontend

El sistema web está construido del lado del cliente sobre HTML, CSS y JavaScript, específicamente se utilizó el Framework Vue.js en su versión 2.5.2.

Para ello, se debe tener previamente instalado Node.js en su versión 10.13.0, a su vez se trabajó con la versión de Npm en su versión 6.4.1.

(Una vez hecho lo anterior, se realiza la acción de agregar el directorio del proyecto con todos los archivos fuente del sistema a la raíz del servidor (public_html) o a un subdirectorio).

Se debe acceder al directorio del proyecto

Ejemplo FRONTEND

Cd test-backend-apis

Luego, se procede a instalar las dependencias del proyecto, ejecutando desde la consola el comando la siguiente instrucción: npm install

(Después, se ejecuta desde la consola de comandos la instrucción npm run dev o npm run build)

Se debe configurar todas las rutas de cada módulo según los endpoint.

Backend

La aplicación web se desarrolló con Python 3.5 del lado del servidor, específicamente Framework Flask 1.0.2, con un manejador de Base de Datos Mysql.

Se debe tener instalado pip en la versión 10.0.1.

Configuración del virtualenv

1. Abrir terminal.
2. Instalar `virtualenv` con:

   > `pip install virtualenv`
3. Acceder al directorio del proyecto (raíz sugerida) por medio del terminal.
4. Crear el entorno virtual con:

   > `virtualenv <nombre_entorno>`

   Se sugiere llamarlo `env`, quedando el comando como:

   > `virtualenv env`

Activar el virtualenv de Python

> `$ .\env\Scripts\activate` \

> `(env)$ `

Luego de esto ya estamos dentro de nuestra virtualenv, en el caso de que ya no queramos trabajar en otro virtualenv y deseamos desactivar la actual aplicamos:

> `(env)$ deactivate` \

Configuración para el entorno de `Flask`

Instalar las extensiones o librerías necesarias. En este caso:

- `flask`.
- `flask-restful`.
- `flask-cors`.
- `simplejson`.

Se instalan haciendo uso del `pip` del nuevo entorno, esto se hará mediante un archivo o fichero en el que estarán todas las extensiones ("requirements.txt"):

> `(env)$ pip install -r ./requirements.txt`

De esta forma se estarán instalando todas las extensiones, la estructura de fichero donde están las extensiones sería básicamente:

requirements.txt

> `flask` \

> `flask-restful` \

> `flask-cors` \

> `simplejson` \

Proceso de inicio del servidor

1. Iniciar el servidor (corre por defecto en el puerto `5000`). Ejemplo:

> `(env) C:\Users\Juan\Desktop\flask-vue\backend-users>python app.py`

NOTA: Actualmente este servidor está configurado para únicamente para desarrollo. El puerto se toma por defecto (el que usa el Flask, en este caso es el 5000), si se quiere tener otro puerto, se le debe colocar en la clase principal (app.py), por ejemplo:

```
if __name__ == '__main__':
    app.run(debug=True, host='0.0.0.0', port=int('numero puerto'))
```

Y de esta forma se estará cambiando el puerto.

Configurar credenciales de base de datos

El siguiente paso es crear una base de datos desde PhpMyAdmin para luego exportar las tablas, el archivo con las tablas tiene por nombre db.sql, ya que la base de datos contará con un nombre, un host, un usuario y una contraseña, en el archivo de conexión a la base de datos (inc/protected/config.php) se le debe proporcionar esos valores correctamente para garantizar la conexión.

Cabe destacar que el proceso para la instalación del *backend*, es solo para el módulo de los usuarios en la siguiente parte se explicará el proceso de instalación para el backend de los estudiantes

Instalación del Sistema del módulo de los estudiantes

En la sesión anterior el *frontend* para este módulo, ya está instalado, en esta parte se explicara como instalar el backend del módulo de los estudiantes.

En primera instancia tiene que tener todos los archivos del módulo de los estudiantes copiados en un mismo directorio, con lo anterior ya se entiende que

tiene que tener instalado Python 3.5. Active el entorno virtual ya explicado en la sesión anterior y luego proceda a ejecutar igual el `(env)$ pip install -r ./requirements.txt` una vez instalado todas las librerías proceda a ejecutar el app.py por consola, tiene que estar parado en el directorio principal del módulo.

Ahora bien para la instalación de la base de datos solo tiene que instalar en el sistema Postgres en la versión 9.5.x. Luego tiene que correr el script proporcionado en la ruta principal del módulo (en donde está el app.py).