SISTEMA DE INFORMACIÓN PARA GENERACIÓN DE HOJAS DE VIDA DE INVESTIGADORES

EILEN KARIME CHAPARRO JAIMES
JENNY KATHERIN SALAZAR DUARTE

UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE INGENIERÍAS FÍSICO-MECÁNICAS
ESCUELA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA
BUCARAMANGA
2014

SISTEMA DE INFORMACIÓN PARA GENERACIÓN DE HOJAS DE VIDA DE INVESTIGADORES

EILEN KARIME CHAPARRO JAIMES
JENNY KATHERIN SALAZAR DUARTE

TRABAJO DE GRADO PARA OPTAR EL TÍTULO DE
INGENIERO DE SISTEMAS

DIRECTOR
Ph.D SONIA CRISTINA GAMBOA SARMIENTO
DOCTORA EN EDUCACIÓN

CODIRECTOR
M.Sc. CARLOS HUMBERTO CARREÑO DÍAZ
MAGÍSTER EN INGENIERÍA DE SISTEMAS E INFORMÁTICA

UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE INGENIERÍAS FÍSICO-MECÁNICAS
ESCUELA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA
BUCARAMANGA
2014

DEDICATORIA

"…A Dios que ha hecho realidad este gran sueño, a él que me ha llenado de tantas bendiciones a él, la gloria de mis triunfos…

"…A mi madre Tulia Salazar Duarte que ha sido el apoyo más grande en mi vida y que siempre estuvo en los momentos que caí y me dio las fuerzas de salir adelante, Gracias madre mía…

"…A mi novio Julián Ángel y a mis hermanas que me apoyaron y se alegran de este gran paso que estoy dando en mi vida como si fuera un logro de ellos...

"…A mis amigos que siempre estuvieron a mi lado…

Jenny Salazar

AGRADECIMIENTOS

Agradecimientos especiales

Al ingeniero Carlos Humberto Carreño Díaz por su orientación intelectual y su apoyo en el desarrollo de este proyecto.

A la doctora Sonia Gamboa por colocar la confianza que puso en nuestras manos para el desarrollo de este proyecto.

CONTENIDO

LISTA DE TABLAS

Pág.

LISTA DE FIGURAS

Pág.

LISTA DE ANEXOS

RESUMEN

TITULO: SISTEMA DE INFORMACIÓN PARA GENERACIÓN DE HOJAS DE VIDA DE INVESTIGADORES.

AUTORES: EILEN KARIME CHAPARRO JAIME, JENNY KATHERIN SALAZAR DUARTE[**]

PALABRAS CLAVES: Sistema, información, web, hojas de vida, scrapy

DESCRIPCIÓN:

Los programas académicos de la Escuela de Ingeniería de Sistemas e Informática de la Universidad Industrial de Santander, requieren mantener registro de las actividades y productos de investigación de los grupos que los soportan, al no existir un medio que recopile la información y la represente de forma que contribuya a realizar procesos de autoevaluación del programa de posgrado, y a la actualización de la información de los investigadores.

Por estas razones se requiere diseñar algoritmos que puedan extraer la información registrada por los investigadores en alguna comunidad virtual, de manera que se tenga una base de datos de investigadores completa y actualizada para apoyar procesos de autoevaluación de los programas.

En el presente proyecto se extrajo información de dos sitios web el CvLac y el ResearchGate mediante la técnica de Scrapy, la cual consiste en diseñar e implementar un algoritmo que extraiga información registrada por los investigadores en alguna comunidad virtual para utilizarla en el sistema y brindar un software más eficiente y rápido a los usuarios en el momento de llenar su hoja de vida o editarla; este es un framework de rastreo web, de código abierto y escrito en PYTHON utilizada para rastrear los sitios mencionados anteriormente raspando la información de forma estructurada y organizada aplicando filtros que para que la información extraída y almacenada que pueda ser utilizada en el sistema y ofrecer un software más eficiente y rápido que supla las necesidades de los usuarios en el momento de llenar su hoja de vida.

[*] Trabajo de grado

[**] Facultad físico-mecánicas, escuela ingeniería de sistemas, director Ph.D Sonia cristina Gamboa

ABSTRACT

TITLE: INFORMATION SYSTEM FOR GENERATION TO CURRICULUM VITAE RESEARCH.

AUTHORS:EILEN KARIME CHAPARRO JAIMES, JENNY KATHERIN SALAZAR DUARTE[**]

KEY WORDS: System, information, web, Curriculum vitae, scrapy, mongodb

DESCRIPTION:

The academic programs of the School of Systems and Computer Engineering of the Universidad Industrial de Santander, are required to maintain records of activities and research products of the groups that support them, the absence of a means to collect information and represent, in contribute to make the program self-evaluation processes graduate and updating of information for researchers.

For these reasons it is necessary to design algorithms that can extract the information recorded by the researchers in a virtual community, so that you have a complete database researchers and updated to support self-evaluation processes of programs.

In the present project information in two websites and the CvLAC ResearchGate extracted by Scrapy technique, which is to design and implement an algorithm that extracts information recorded by researchers in a virtual community for use in the system and provide more efficient and faster to users at the time to fill your resume or edit software; This is a framework of web crawling, open source and written in Python used to track the sites listed above scraping the information in a structured and organized by applying filters to the extracted and stored information that can be used in the system and provide most efficient and fast software that meets the needs of users at the time to fill your resume.

[*] Bachelor thesis

[**] Faculty of Physical-Mechanical Engineering, Systems engineering school, Director Ph.D Sonia Cristina Gamboa

INTRODUCCIÓN

Las comunidades virtuales surgen como una herramienta útil que facilitan la comunicación, el compartir, el intercambio de información, y la interacción entre las personas las cuales pertenecen a un grupo o comunidad con intereses y objetivos en común.

Las comunidades virtuales científicas han tenido un gran impacto en la socialización de la información relacionada con la producción de conocimiento, en busca de unificar la información de estas comunidades se propone en este proyecto el desarrollo de un sistema de información para la generación de hojas de vida de investigadores de la escuela de ingeniería de sistemas, con la finalidad de disponer de bases de datos de investigadores centralizadas y actualizada facilitando los programas de mejoramiento continuo de la escuela.

La solución que se plantea es diseñar e implementar un algoritmo que extraiga información registrada por los investigadores en alguna comunidad virtual mediante las técnicas de Scrapy para utilizar esta información en el sistema y brindar un software más eficiente y rápido a los usuarios en el momento de llenar su hoja de vida o editarla.

1. PLANTEAMIENTO Y JUSTIFICACIÓN DEL PROBLEMA

Desde hace unos años el uso de las tecnologías de la información y la comunicación, y en especial de la web 2.0, ha impactado los entornos sociales, en busca de compartir la información de una forma completa. A partir del impacto que ha causado en las comunidades científicas, hoy en día estas comunidades llevan a cabo actividades propias de la investigación haciendo uso de estas herramientas, entre otras, para socializar la información relacionada con la producción de conocimiento.

"La misión de la Universidad Industrial de Santander sustenta su trabajo en las cualidades humanas de las personas que la integran, en la capacidad laboral de sus empleados, en la excelencia académica de sus profesores y en el compromiso de la comunidad universitaria con los propósitos institucionales y la construcción de una cultura de vida"[1]. Es por esto que los programas académicos de la Escuela de Ingeniería de Sistemas e Informática de la Universidad Industrial de Santander, requieren mantener registro de las actividades y productos de investigación de los grupos que los soportan, al no existir un medio que recopile la información y la represente de forma que contribuya a procesos de autoevaluación del programa de posgrado, y a la actualización de la información de los investigadores.

Por estas razones se requiere diseñar algoritmos que puedan extraer la información registrada por los investigadores en alguna comunidad virtual, de manera que se tenga una base de datos de investigadores completa y actualizada para realizar permanentemente procesos de autoevaluación de los programas. En suma, se busca representar las relaciones existentes entre investigadores, su

[1]Universidad Industrial de Santander, http://www.uis.edu.co/webUIS/es/acercaUis/index.html

producción intelectual y pertenencia a grupos de investigación de acuerdo a la información disponible en el sistema de información propuesto.

El sistema de información desarrollado contribuye a la integración y actualización de la información de los investigadores de la escuela de ingeniería de sistemas de la UIS, utilizando la implementación de la metodología del desarrollo basado en componentes. El proyecto agiliza la actualización de la información, la búsqueda de las actividades y productos de investigación ayudando a mejorar los procesos de autoevaluación del programa de posgrado de ingeniería de sistemas de la UIS.

2. OBJETIVOS

2.1. OBJETIVO GENERAL

Diseñar e implementar un sistema de información que represente y genere hojas de vida de investigadores.

2.2. OBJETIVOS ESPECÍFICOS

- Analizar los requerimientos necesarios para implementar el sistema de información, elaborando el documento con las especificaciones de requisitos y sus correspondientes casos de uso.

- Establecer una estructura de datos o de conocimiento a partir de la cual se pueda representar el sistema.

- Diseñar e implementar un sistema de información que permita realizar funcionalidades como:

    - Búsqueda de información de los investigadores.
    - Actualizar información de los investigadores.
    - Representar y generar las hojas de vida de los investigadores.

- Desarrollar para el sistema de información diseñado los siguientes módulos:

    - Extraer de internet los datos para actualizar las hojas de vida de los investigadores.

- Habilitar hojas de vida para ser verificadas y editadas por usuarios.
- Representar relaciones entre investigadores registrados.
- Consultar hojas de vida de investigadores.

- Validar la funcionalidad y utilidad del componente implantado en el portal web mediante la realización de pruebas.

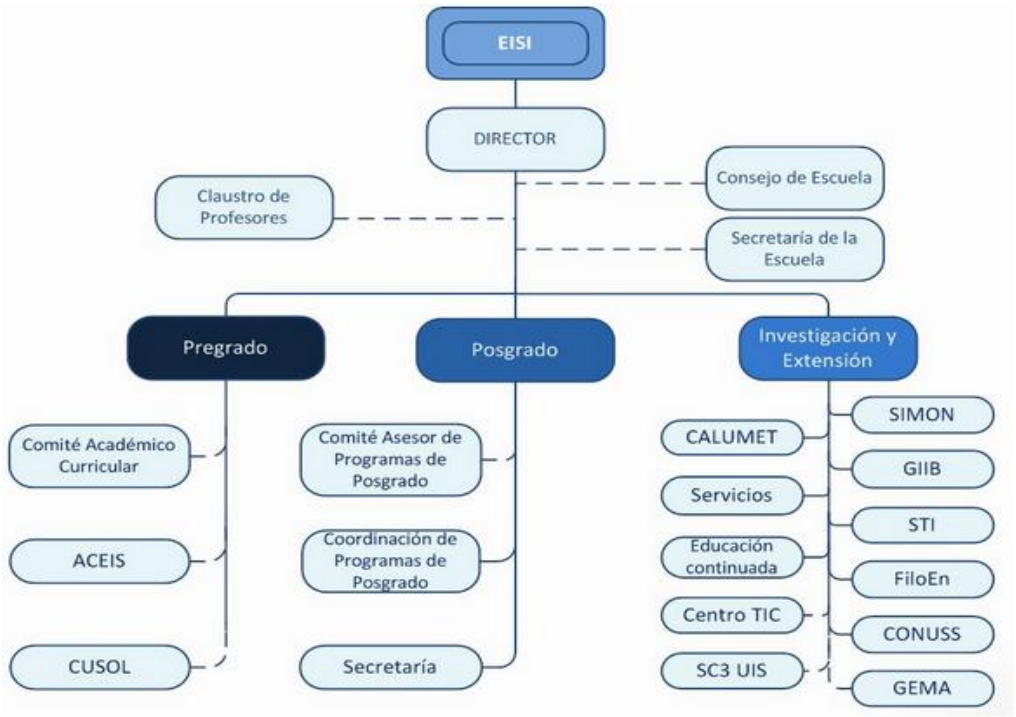
3. MARCO TEÓRICO

3.1. ESCUELA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA

El programa de Ingeniería de Sistemas e Informática de la Facultad de Ingenierías Físico-mecánicas de la UIS ofrece a la comunidad académica los siguientes programas:

- Pregrado: Ingeniería de Sistemas
- Posgrado: Maestría en Ingeniería de Sistemas e Informática
- Educación no formal: Diplomado en Ingeniería Web

Figura 1. Estructura organizacional EISI



La escuela de ingeniería de sistemas (EISI) se ha caracterizado por formar profesionales de alta calidad y por contribuir al progreso de la región aportando al análisis y solución de los problemas que la afectan a través de las investigaciones y los proyectos de grado que aquí se realizan.[2]

3.2. MAESTRÍA EN INGENIERÍA DE SISTEMAS E INFORMÁTICA

La Maestría de Ingeniería de Sistemas e Informática es una maestría de investigación que está estructurada de forma tal que los estudiantes puedan desarrollarse en cualquiera de las áreas que se abordan desde los grupos de investigación: modelos y simulación, ingeniería biomédica, gestión y optimización de sistemas, tecnologías de la información, ingeniería de software, ingeniería telemática y sistemas inteligentes o inteligencia artificial y sistemas de conocimiento experto.[3]

La Escuela de Ingeniería de Sistema e Informática cuenta actualmente con 8 grupos de investigación, 17 profesores planta y 3 catedra.

3.3. AUTOEVALUACIÓN

En los últimos años existe un especial interés de las universidades públicas por diagnosticar el estado en el que se encuentran sus programas educativos, la preocupación de los interesados por ingresar a programas de calidad, la necesidad de demostrar su nivel académico e identificar el estado actual de sus estudiantes, egresados e investigadores. Teniendo en cuenta lo anterior existe la

[2] ESCUELA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA UIS. [en línea] Disponible en: http://cormoran.uis.edu.co/eisi/eisi.jsp?IdServicio=S86
[3] ESCUELA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA UIS [en línea] Disponible en: http://cormoran.uis.edu.co/eisi/eisi.jsp?IdServicio=S78

exigencia de una constante autoevaluación en el aprendizaje de los estudiantes de posgrado siguiendo los principios de una evaluación formativa para convertirla en una herramienta de retroalimentación del aprendizaje.

La Universidad Industrial de Santander ha demostrado ser una de las mejores universidades del país, con una formación de alto nivel y calidad generando ventajas permanentes para un desarrollo sostenible, en la escuela de ingeniería de sistemas e informática y en especial en los posgrados la autoevaluación es un requerimiento que se extiende cada vez más; es por esto que un sistema de información que genere hojas de investigadores aportaría ventajas a la hora de apoyar los procesos de autoevaluación en el programa académico, teniendo información actualizada y veraz de los investigadores lo cual serviría para medir su nivel de producción intelectual ya que en este sistema de información estaría registrada su formación académica y cada uno de los aportes que hayan realizado, sería una forma de medir el nivel de los estudiantes de posgrados y así seguir con la búsqueda inherente de la acreditación.

4. TECNOLOGÍAS INVOLUCRADAS EN EL DESARROLLO DEL PROYECTO

4.1. WEB SCRAPY

En la extracción de datos se obtiene información relevante de los investigadores de la EISI, donde puede contener datos generales, experiencia, proyectos, artículos, trabajos, etc. Se consideró para extraer la información: Web Scrapy.

Esta técnica permite el rastreo, análisis y extracción de datos de un sitio web, extrae información que puede ser utilizada en muchas aplicaciones como la minería de datos, procesamiento de la información o de archivo.[4]

Se utiliza Scrapy para encontrar la información de los investigadores de la EISI registrados en CvLAC y ResearchGate como: datos generales, experiencia, libros, artículos, proyectos, etc.

La información que se recoge mediante esta técnica se genera a partir de la url del investigador para analizar el código html en busca de información que se encuentra en sus etiquetas.

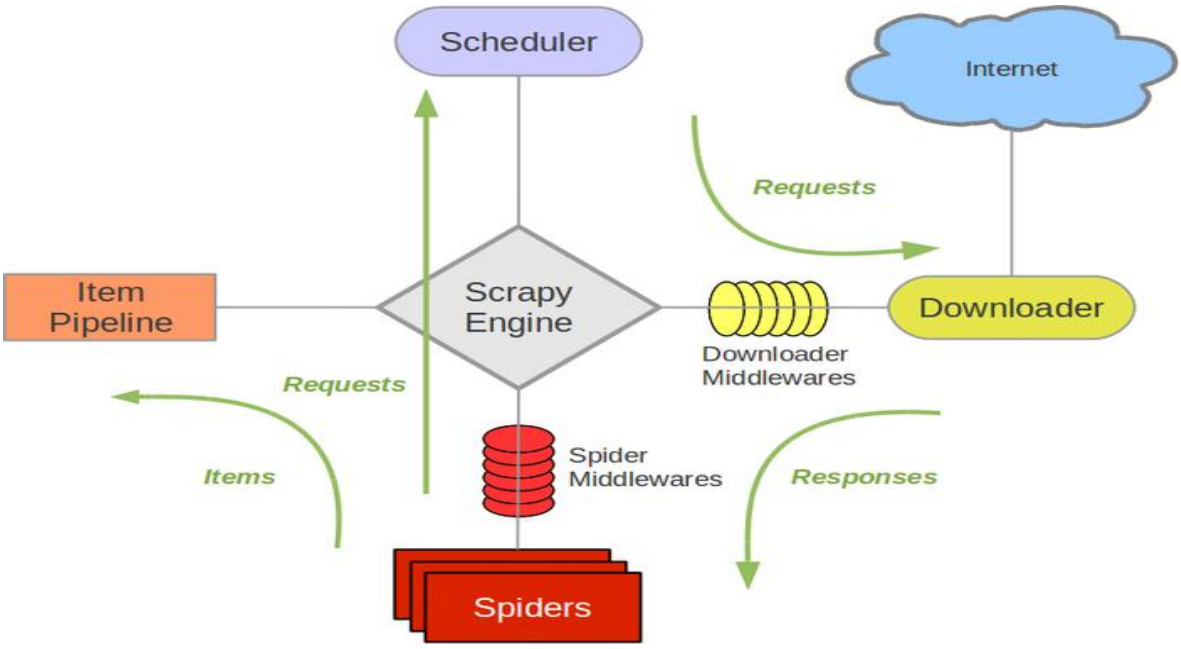
En la figura 2 se observa la arquitectura Scrapy. A continuación se definirán cada uno de sus componentes.

- *Scrapy Engine*: controla el flujo de datos entre todos los componentes del sistema.
- *Scheduler*: el *schedule*r recibe peticiones desde el motor y los pone en cola hasta que el motor las pide.

[4] http://scrapy.org/

- *Downloader*: Es el responsable de buscar las páginas web e interactuar con el motor y los spiders.
- *Spiders*: Los spider son las clases que definen como se raspo un sitio determinado, la forma de rastreo, en otras palabras define el comportamiento para el rastreo y análisis de la página.
- *Pipeline*: procesa los artículos una vez se han extraído por los Spiders. Sus tareas típicas son la limpieza, la validación y la persistencia (almacenamiento en una base de datos)
- *Downloader Middelware*: procesan solicitudes cuando pasan desde el motor a la *Downloader* y las respuestas que pasa desde el *Downloader* al motor.
- Spider *Middelware*: procesan las peticiones y las respuestas entre el motor y el Spider.

Figura 2. Arquitectura Scrapy



4.2. PYTHON

Python fue creado a finales de los ochenta por Guido van Rossum en el Centro para las Matemáticas y la Informática CWI. Es un lenguaje de alto nivel, multiparadigma que permite programación orientada a objetos, programación imperativa y programación funcional. Python usa tipado dinámico y conteo de referencias para la administración de memoria[5]

Figura 3. Estructura de un intérprete

Es un lenguaje interpretado incluye un modo interactivo en el cual se escriben las instrucciones, pueden ser introducidas una a una, pudiéndose ver el resultado de su evaluación inmediatamente.

La librería estándar de Python es muy amplia. Permite hacer varias cosas que involucran: expresiones regulares, generación de documentos, evaluación de unidades, pruebas, procesos, bases de datos, navegadores web, CGI, ftp, correo electrónico, XML, XML-RPC, HTML, archivos WAV, criptografía, y también otras funciones dependientes del Sistema.

[5] http://www.greenteapress.com/thinkpython/html/thinkpython002.html#toc4

4.3. JAVASCRIPT

Es un lenguaje de programación interpretado, dialecto del estándar *ECMAScript.* Es orientado a objetos, prototipos, imperativo, débilmente tipado y dinámico. Sus aplicaciones incluyen implementos en un navegador web permitiendo mejoras en la interfaz del usuario y páginas web dinámicas, además se utiliza en aplicaciones externas a la web como PDF y Widgets.

Su sintaxis es similar al C, pero además toma nombres y convenciones del lenguaje de programación Java, pero son dos lenguajes de programación que no tienen la misma semántica y propósitos.

4.4. JAVA

Es un lenguaje de programación desarrollado por James Gosling de Sun Microsystems y fue publicado en 1995. Es un lenguaje compilado e interpretado, y esto es lo que permite que pueda ser ejecutado en varios entornos y realizar cualquier tipo de programa. En la actualidad es un lenguaje muy extendido y cada vez cobra más importancia tanto en el ámbito de Internet como en la informática en general. Su sintaxis es muy parecida a la de C y C++, JAVA es orientado a objetos cumpliendo los tres paradigmas: encapsulamiento, herencia y polimorfismo. Las aplicaciones pueden ejecutarse en cualquier entorno virtual java, sin importar la arquitectura de la máquina.

4.4.1. Librerías Java utilizadas

4.4.1.1. iText: iText: es una librería que permite crear y manipular archivos PDF. Se pueden generar documentos e informes, añadir marcadores, números de página, marca de agua y otras características de los PDF existentes.

A partir de una base de datos o de un archivo XML se puede crear un PDF utilizando la clase *PDFWrite*. A continuación se describe tres maneras de generar el PDF:

- Usando objetos como *Chunk, Phrase, Paragraph, List.* Estos objetos son bloques de construcción básicos de iText.
- Usando *PDFContentByte*, que es una clase que utiliza una serie de métodos que asigna imágenes de Adobe, como dibujo de arcos, de círculos, de rectángulos, etc.
- Usando PDFGraphics2D, que es una implementación para iText de la clase Graphics2D de Java.

4.4.1.2. iReport Designer: iReport Designer es un diseñador virtual de informes, los cuales serán ejecutados por JasperReports. El JasperServer, proporciona una interfaz basada en web para gestionar, programar y ejecutar los informes, un repositorio para almacenar todos los recursos del informe como imágenes, fuentes de datos etc.

Se debe crear un archivo Jasper Reports Extensible Markup Language JRXML, el cual tendrá la definición del diseño del informe; antes de ser ejecutado este archivo debe elaborarse en un objeto binario que será un archivo de Jasper, y así podrá ser ejecutado por la biblioteca JasperReports.

Para ser ejecutado se debe pasar un archivo de Jasper y una fuente de datos para JasperReports, algunos de ellos son una consulta SQL, un archivo XML, un archivo Comma Separated Values CSV, una consulta Hibernate Query Language HQL, una colección de JavaBeans, etc; en caso de no tener una fuente de datos, se permite escribir un propio origen de datos personalizado. Con esto ya es posible generar el informe en el formato que se desee.

Para realizar el diseño de los informes puede hacerse desde cero, o también existen plantillas prediseñadas.

- Report Designer es donde se diseña visualmente el informe: arrastre, posicionamiento, alineación y cambio del tamaño de los elementos del informe.
- Report Inspector muestra la estructura completa del informe, que se compone de muchos objetos (por ejemplo, campos, parámetros y variables), bandas (que son las secciones del documento) y elementos (como campos de texto, imágenes o gráficos).
- Elements Palette contiene los elementos que se pueden arrastrar dentro de una banda para mostrar los datos.
- Property Sheet se utiliza para configurar las propiedades del componente seleccionado en el informe (por ejemplo, un campo, elemento, banda, grupo, u otros).

4.5. JAVASERVER FACES

Es un framework de interfaces de usuario del lado de servidor para aplicaciones web basada en Java EE y en el patrón MVC (Modelo Vista Controlador).
JSF incluye:

- Un conjunto de APIs para: representar componentes de una interfaz de usuario y administrar su estado, manejar eventos, validar entrada, definir un esquema de navegación de las páginas y dar soporte para internacionalización y accesibilidad.
- Una librería de etiquetas JSP personalizadas para dibujar componentes UI dentro de una página JSP.
- Un modelo de eventos en el lado del servidor.

- Administración de estados.
- Beans administrados.

JSF es muy flexible, nos permite personalizar los componentes como la recarga de la vista de las páginas, y elaborar interfaces de usuario como nos convenga. La tecnología JavaServer Faces permite construir aplicaciones web con una separación entre el comportamiento y la presentación.

4.6. EJB

Los Enterprise JavaBeans (EJB) es un modelo de componentes, del lado del servidor mientras que JSF es un modelo de componentes de la capa de presentación. Los EJB está basado en componentes lo que permite que sean flexibles y sobre todo reutilizables. El funcionamiento de los componentes EJB se basa en el trabajo del contenedor EJB. El contenedor EJB es un programa Java que corre en el servidor.

JSF y EJB funcionan mejor juntos los creadores de ambos modelos proporcionan puntos de extensión estándar para permitir la integración con otros marcos.

4.7. MONGODB

Es un sistema de base de datos NoSQL, el cual es orientado a documentos bajo el concepto de código abierto. Es un sistema que permite guardar los datos en documentos tipo JSON con un esquema dinámico, realizando la integración de los datos en aplicaciones fáciles y rápidas.

NoSQL es una clase de sistemas de gestión de bases de datos que difieren del modelo clásico del sistema de gestión de bases de datos relacionales en aspectos importantes, el más destacado es que no usan SQL como el principal lenguaje de consultas. Los datos no tienen una definición de atributos fija, cada registro puede contener una información de diferente forma, se pueden almacenar los atributos importantes en cada uno de ellos. No soportan operaciones JOIN, ni garantizan completamente ACID (atomicidad, coherencia, aislamiento y durabilidad), escalan bien horizontalmente, aumentando el rendimiento del sistema simplemente añadiendo más nodos, sin necesidad de realizar ninguna otra operación más que indicar al sistema cuáles son los nodos disponibles, pueden manejar enormes cantidades de datos y no generan cuellos de botella.

Las bases de datos NoSQL se clasifican según su forma de almacenar los datos, y comprenden categorías como clave-valor, las implementaciones de *BigTable*, bases de datos documentales, y Bases de datos orientadas a grafos.

A menudo las bases de datos NoSQL están altamente optimizadas para las operaciones recuperar y agregar, y normalmente no ofrecen mucho más que la funcionalidad que almacenar los registros. La pérdida de flexibilidad en tiempo de ejecución, comparado con los sistemas SQL clásicos, se ve compensada por ganancias significativas en escalabilidad y rendimiento cuando se trata con ciertos modelos de datos.

- Consistencia Eventual: No implementa mecanismos rígidos de consistencia, esto se conoce como BASE (*Basically Available Soft-state Eventual Consistency*, o coherencia eventual flexible básicamente disponible).
- Estructura Distribuida: Generalmente se asignan los datos por medio de tablas de hash distribuidas.
- Escalabilidad Horizontal: Permite aumentar el rendimiento del sistema simplemente añadiendo más nodos, sin necesidad en muchos casos de

realizar otra operación más que mostrar al sistema cuales nodos son los disponibles.

- Estructura Dinámica: Los datos no tiene una definición de atributos fija, cada registro puede contener una información con diferente forma pudiendo almacenar solo los atributos que interesan en cada uno de ellos. También se puede almacenar estructuras de datos complejas en un solo documento.
- Tolerancia a fallos y Redundancia.
- No genera cuellos de botella Aumentando la claridad y el rendimiento.

4.8. JAVASCRIPT INFOVIS TOOLKIT

JavaScript Infovis Toolkit ofrece herramientas para visualizar los datos de forma interactiva en la web. Estas herramientas implementan funciones avanzadas de visualización de la información como *hypertree TreeMaps SpaceTree* etc con animaciones avanzadas con RGraph.

En este proyecto se utilizó *hypertree* que nos permite hacer árboles jerárquicos y visualizar de una manera dinámica la información de los grupos de investigación con sus líneas de investigación, y sus investigadores, estos árboles de nodos que a su vez son padres de otros nodos.

5. METODOLOGÍA

Existen diferentes metodologías para el desarrollo de software, el presente trabajo de investigación se basará en la metodología de desarrollo de software *Rational Unified Process* (RUP). El Proceso Unificado es un proceso con un enfoque iterativo en el cual se propone la comprensión incremental del problema a través de una serie de refinamientos sucesivos y un crecimiento incremental de una solución a través de varios ciclos. Su objetivo es permitir la producción de software de la mayor calidad que satisfaga las necesidades de los usuarios finales, dentro de planificaciones y presupuestos predecibles. Como parte del enfoque iterativo se encuentra la flexibilidad para acomodarse a nuevos requisitos o a cambios tácticos en los objetivos del negocio. También permite que el proyecto identifique y resuelva los riesgos rápidamente.

El RUP es un proceso basado en los modelos de Cascada y por Componentes, el cual presenta las siguientes características: Es dirigido por los casos de uso, es centrado en la arquitectura, iterativo e incremental (Booch, Rumbaugh y Jacobson, 2000)

El desarrollo bajo el RUP está centrado en la arquitectura. El proceso se centra en establecer al principio una arquitectura software que guía el desarrollo del sistema. Con ello se facilita el desarrollo en paralelo, se minimiza la repetición de trabajos y se incrementa la probabilidad de reutilización de componentes y el mantenimiento posterior del sistema. Este diseño arquitectónico sirve como una sólida base sobre la cual se puede planificar y manejar el desarrollo de software basado en componentes. Las actividades de desarrollo bajo el RUP están dirigidas principalmente por los casos de uso.

5.1. REQUERIMIENTOS

En esta etapa se pretende hacer un reconocimiento de la problemática presente en la Facultad de Fisicomecánicas (específicamente en el departamento de posgrado de ingeniería de sistemas.), conociendo la necesidad de tener actualizada la información de los investigadores, implica tener acceso a las hojas de vida de los investigadores que están registradas en algún sitio virtual, luego analizar la información que realmente va ser útil para los procesos de autoevaluación del programa de posgrado; y así realizar una planificación y estimación completa sobre los recursos necesarios para el proyecto.

5.2. DISEÑO

En esta etapa se va a diseñar el modelo de datos y las interfaces con las que los usuarios del sistema van a interactuar. Para el diseño de la base de datos se elaborarán prototipos bases y se depurarán hasta obtener la versión final normalizada. También se documentará el sistema con los diagramas más importantes de la especificación del lenguaje unificado de modelado (UML)

5.3. IMPLEMENTACIÓN

Después de haber realizado un minucioso análisis y diseño, se procede a desarrollar la programación de la aplicación web, implementando los diferentes módulos planteados en las fases anteriores. Gracias a la metodología planteada se pueden hacer pequeños cambios en los requerimientos del sistema en esta etapa si es necesario.

5.4. PRUEBAS

Esta última etapa nos permite mediante procesos comprobar la calidad del software y el cumplimiento de los requerimientos. Estas pruebas se realizan para identificar posibles fallos de implementación, calidad o usabilidad de un programa.

Se elaboran los diferentes documentos que soporten el sistema realizado, y se hará en paralelo junto con la fase de implementación para tener una mejor organización y calidad en el manejo del sistema. La documentación en proyectos como este es sumamente importante porque de ésta depende en gran medida el éxito y uso masivo del sistema.

6. ANÁLISIS Y DISEÑO DEL SISTEMA DE INFORMACIÓN

Para el diseño e implementación del sistema de información para generación de hojas de vida de investigadores se realizó la toma de requerimientos y especificación de requisitos para los módulos a implementar en el sistema de información, se planteó un diseño inicial de la estructura del sistema de información que generara hojas de vida de investigadores.

6.1. ESPECIFICACIÓN DE REQUERIMIENTOS

6.1.1. Requisitos Funcionales: Especificaciones destinadas a cubrir los siguientes aspectos:

- El sistema debe permitir crear usuario y guardar su información en la base de datos.
- El sistema debe permitir actualizar y editar las hojas de vida por los usuarios.
- El sistema puede ser consultado por el investigador y administrador del sistema que este registrado y autenticado.
- El sistema debe permitir al usuario extraer de manera automatizada la información del CvLac o del ResearchGate.
- El sistema debe permitir que el usuario avale la información que fue actualizada por el sistema.
- El sistema debe permitir al usuario visualizar las relaciones entre los investigadores.

6.1.2. Requisitos no Funcionales. Especificaciones destinadas a los aspectos de calidad del sistema:

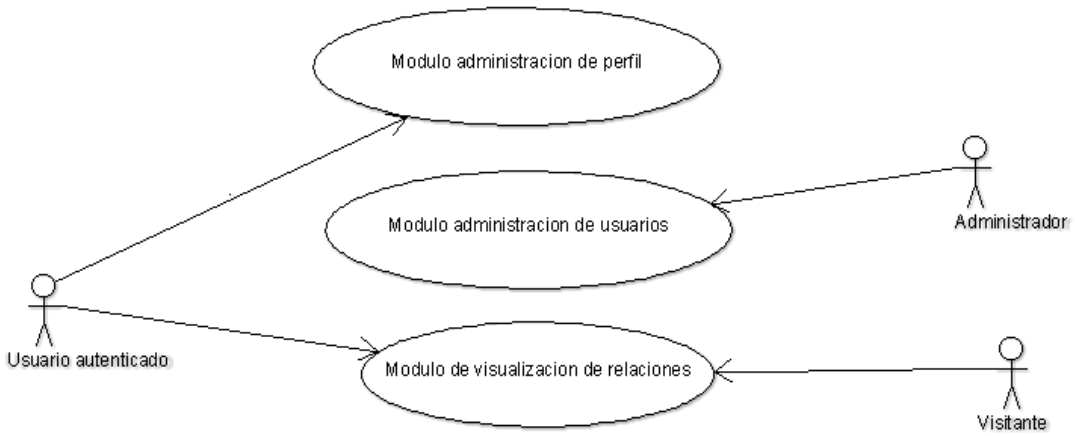
- El sistema debe visualizarse y funcionar correctamente en cualquier navegador.
- El sistema no debe tardar más de 30 segundos en mostrar los resultados requeridos por el usuario.
- El sistema debe ser consistente al momento de filtrar la información que actualiza y se guarda en la base de datos, no debe haber información duplicada.
- El sistema debe regir las reglas de seguridad de información de los usuarios.

6.2. CASOS DE USOS

Los casos de uso describe la interacción entre el usuario y el sistema.

6.2.1. Identificación de actores

Figura 4. Identificación de actores



- **Descripción del actor administrador**.

Actor:	Administrador
Casos de Uso asociados:	Autenticación, gestión información por usuarios.
Descripción:	El administrador puede gestionar la información de los usuarios seleccionando de qué página desea extraer del CvLac o del ResearchGate.
Tipo	Primario

- **Descripción del actor investigador.**

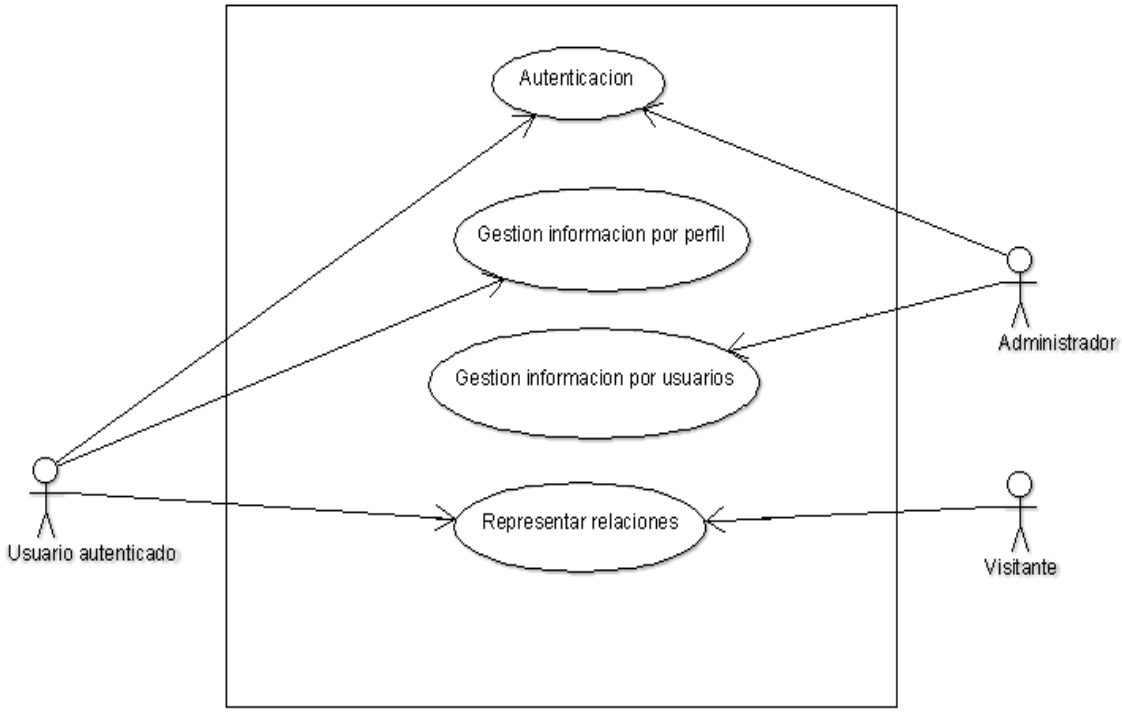
Actor:	Usuario autenticado
Casos de Uso asociados:	Autenticación, gestión información por perfil, representar relaciones.
Descripción:	El usuario autenticado puede gestionar la información de su perfil seleccionando de qué página desea extraer la información si del CvLac o del ResearchGate, puede editarla, consultar su hoja de vida y visualizar la relación entre los investigadores.
Tipo	Primario

- **Descripción del actor investigador.**

Actor:	Visitante
Casos de Uso asociados:	Representar relaciones.
Descripción:	El visitante puede visualizar la relación entre los investigadores.
Tipo	Primario

6.2.2. Diagrama modelo casos de usos

Figura 5. Diagrama modelo casos de uso



6.2.2.1. Descripción Diagrama modelo casos de usos

Caso de uso:	Autenticación
Actores:	Usuario autenticado, Administrador
Propósito:	Permite al investigador o administrador ingresar y tener permisos.
Resumen:	Este caso de uso está diseñado para ser utilizado para usuarios con permisos de administrador.
Precondición:	• El usuario debe autenticarse.
Flujo principal:	El caso de uso se inicia cuando se muestra dos cajas de texto donde se debe ingresar los datos (usuario y contraseña),

	además de un botón de inicio de sesión que envía los datos para la autenticación.
Sub-flujo	**SF-1** Datos Se muestra dos cajas de texto para el ingreso de datos del usuario. En la primera caja se ingresa el nombre del usuario y en la segunda se ingresa la contraseña.
Excepciones	**EX-1** *Debe especificar usuario,* el campo usuario se encuentra vacío debe ingresar su nombre de usuario en el campo. **EX-2** *Debe especificar contraseña,* el campo contraseña se encuentra vacío debe ingresar su contraseña en el campo **EX-3** *Verifique su usuario y contraseña,* usuario o contraseña incorrectos, se solicita al usuario verificar e ingresar sus datos correctamente.

Caso de uso:	Gestión Información por perfil
Actores:	Usuario autenticado.
Propósito:	Permite al usuario, consultar, editar, extraer información de su hoja de vida y generar PDF de su hoja de vida
Resumen:	En este caso de uso se permite extraer información del investigador que debe estar registrado en las páginas del CvLac o del ResearchGate, se puede editar consultar y generar PDF de su hoja de vida.
Precondición:	• El usuario ingresa el link donde está su hoja de vida en el CvLac o del ResearchGate. • El usuario da clic en actualizar y accede a la plataforma donde puede generar el PDF de su hoja de vida, consultar, o editarla.

Flujo principal:	El caso de uso se inicia cuando el usuario se ha autenticado correctamente. Seguido ingresa el tipo de usuario que puede ser docente o estudiante, después ingresa el link del CvLac o ResearchGate donde se va a extraer su información. Cuando ingresa puede ver una ventana con información sugerida en las pestañas de estudios, distinciones y experiencia profesional. En la última pestaña se puede ver un resumen de la hoja de vida y la opción de generar el PDF.
Sub-flujo	**SF-1** Se muestra un cuadro de texto donde el usuario ingresa el link de donde se extraerá la información.
Excepciones	**EX-1** *Se debe ingresar una de las links de donde se desea extraer*.

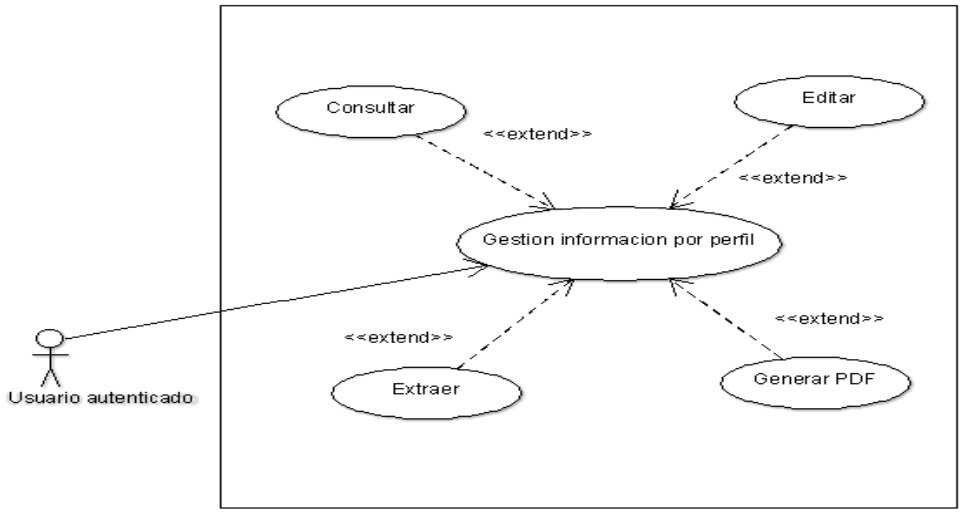
Caso de uso:	Gestión Información por usuarios.
Actores:	Administrador.
Propósito:	Permite al administrador, consultar, editar, extraer información de cualquier usuario y generar PDF de la hoja de vida que desee.
Resumen:	En este caso de uso se permite extraer información de los investigadores registradas en las páginas del CvLac o del ResearchGate,
Precondición:	• El administrador ingresa el link donde está la hoja de vida que desea extraer ya sea del CvLac o del ResearchGate. • El administrador da clic en actualizar y accede a la plataforma donde puede generar, consultar, o editar su hoja de vida.
Flujo principal:	El caso de uso se inicia cuando el usuario se ha autenticado correctamente. Seguido ingresa el tipo de usuario que puede ser

	docente o estudiante, después ingresa el link del CvLac o ResearchGate donde se va a extraer su información. Cuando ingresa puede ver una ventana con información sugerida en las pestañas de estudios, distinciones y experiencia profesional. En la última pestaña s ver un resumen de la hoja de vida y la opción de generar el PDF.
Sub-flujo	**SF-1** Se muestra un cuadro de texto donde el usuario ingresa el link de donde se extraerá la información.
Excepciones	**EX-1** *Se debe ingresar una de las links de donde se desea extraer*.

Caso de uso:	Representar Relaciones
Actores:	Usuario autenticado, visitante
Propósito:	Permite al usuario representar relaciones entre los investigadores.
Resumen:	En este caso de uso se permite representar relaciones entre los investigadores de Ingeniería de Sistemas UIS; a partir del grupo de investigación y la línea de investigación que pertenece.
Precondición:	• El usuario debe dar clic en representar relaciones para poder visualizarlas.
Flujo principal:	El caso de uso se inicia cuando el usuario da clic en el botón ver relaciones que está en el portal de autoevaluación.
Sub-flujo	**SF-1** Se muestra un botón representar relaciones donde se le puede dar clic para visualizar las relaciones en forma de grafo de los grupos de investigación, de las líneas de investigación y de los investigadores.
Excepciones	**E-1** .*El usuario debe dar clic en el botón representar relaciones* para poder visualizar el grafo.

6.2.3. Diagrama de caso de uso – Gestión información por perfil

Figura 6. Diagrama de caso de uso - Gestión información por perfil



6.2.3.1. Descripción del Diagrama de caso de uso – Gestión información por perfil

Caso de uso:	Consultar
Actores:	Usuario autenticado
Propósito:	Permite consultar la información deseada del usuario.
Resumen:	Este caso de uso permite consultar la información de los investigadores que han agregado la hoja de vida.
Precondición:	• El usuario debe tener registrada la hoja de vida en el sitio.
Flujo principal:	El caso de uso inicia cuando ingresa el link del CvLac del usuario y enseguida se da clic en el botón actualizar.
Sub-flujo	**SF-1** Se muestra un cuadro donde el usuario debe ingresar el link del CvLac del investigador.

	SF-2 El usuario da clic en el botón actualizar enseguida puede consultar su información personal, idiomas, estudios, distinciones, experiencia profesional y un resumen de su hoja de vida
Excepciones	**E-1** Debe dar clic en actualizar para visualizar la información.

Caso de uso:	Editar
Actores:	Usuario autenticado
Propósito:	Permite editar la información deseada de los usuarios.
Resumen:	En este caso de uso permite editar la información de los usuarios que está registrada en el sitio.
Precondición:	• El usuario debe tener registrada la hoja de vida en el sitio.
Flujo principal:	El caso de uso inicia cuando ingresa el link del CvLac del usuario y enseguida se da clic en el botón actualizar.
Sub-flujo	**SF-1** Se muestra un cuadro donde el usuario debe ingresar el link del CvLac del investigador. **SF-2** El usuario da clic en el botón actualizar para poder acceder a la plataforma y editar la información que desee ya sea información personal, idiomas, estudios, distinciones, experiencia profesional.
Excepciones	**E-1** Debe dar clic en actualizar para visualizar su información y poder editar los diferentes campos.

Caso de uso:	Extraer
Actores:	Usuario autenticado
Propósito:	Permite extraer la información deseada de los usuarios.
Resumen:	En este caso de uso permite extraer información de los usuarios que está registrada en el sitio web de Colciencias-CvLac o del

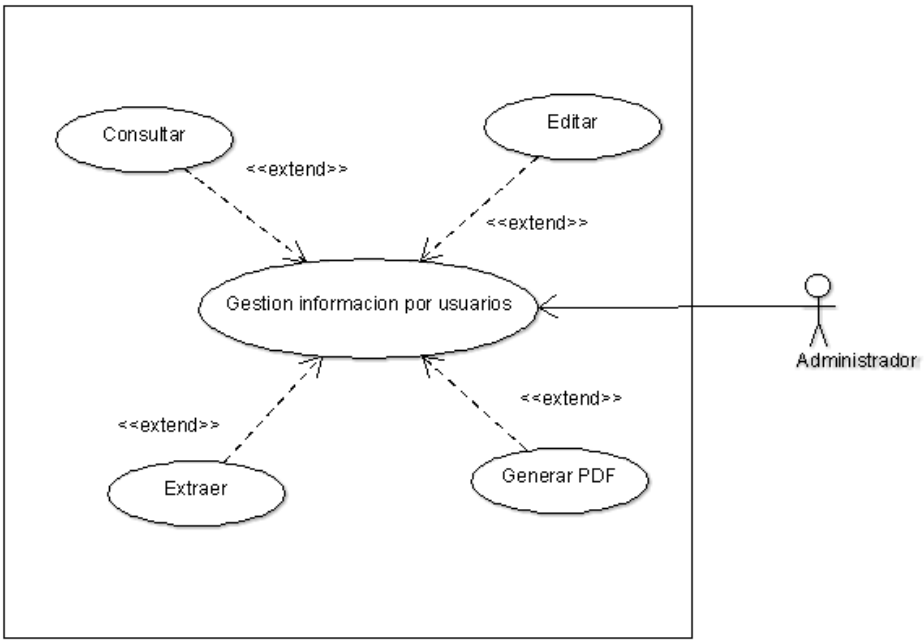
	ResearchGate.
Precondición:	• El usuario debe tener registrada la hoja de vida en el CvLac o en ResearchGate.
Flujo principal:	El caso de uso inicia cuando ingresa el link del CvLac o del ResearchGate del usuario y enseguida se da clic en el botón actualizar, enseguida en estudios, distinciones y experiencia profesional aparece una ventana de información sugerida con la información que se extrajo de alguna de las páginas y que nos servirá para llenar los campos.
Sub-flujo	**SF-1** Se muestra un cuadro donde el usuario debe ingresar el link del CvLac del investigador. **SF-2** El usuario da clic en el botón actualizar para extraer la información. **SF-3** En estudios, distinciones y experiencia profesional el usuario puede visualizar la información extraída en una ventana de información sugerida.
Excepciones	**E-1** Debe dar clic en actualizar para visualizar la información extraída. **E-2** El usuario debe tener su hoja de vida registrada en el CvLac o en ResearchGate.

Caso de uso:	Generar PDF
Actores:	Usuario autenticado
Propósito:	Permite generar PDF de su hoja de vida.
Resumen:	En este caso de uso permite generar el PDF de su hoja de vida.
Precondición:	• El usuario debe agregar su información a la hoja de vida.
Flujo principal:	El caso de uso inicia cuando ingresa el link del CvLac del usuario

	y enseguida se da clic en el botón actualizar, enseguida puede ver el resumen de su hoja de vida si ya la tiene registrada y generar el PDF; de lo contrario agrega la información antes de generar el PDF.
Sub-flujo	**SF-1** Se muestra un cuadro donde el usuario debe ingresar el link del CvLac del investigador. **SF-2** El usuario da clic en el botón actualizar para extraer la información. **SF-3** el usuario da clic en resumen para visualizar el resumen de su hoja de vida. **SF-4** el usuario da clic en PDF para generar el PDF de su hoja de vida.
Excepciones	**E-1** Debe tener la hoja de vida registrada en la plataforma.

6.2.4. Diagrama de caso de uso – Gestión información usuarios

Figura 7. Diagrama de caso de uso - Gestión información por usuarios



6.2.4.1. Descripción del Diagrama de caso de uso – Gestión información usuarios

Caso de uso:	Consultar
Actores:	Administrador
Propósito:	Permite consultar la información deseada de cualquier usuario.
Resumen:	Este caso de uso permite consultar la información de los usuarios que han agregado la hoja de vida.
Precondición:	• El usuario debe tener registrada la hoja de vida en el sitio.
Flujo principal:	El caso de uso inicia cuando ingresa el link del CvLac o del ResearchGate del usuario y enseguida se da clic en el botón actualizar.
Sub-flujo	**SF-1** Se muestra un cuadro donde el usuario debe ingresar el link del CvLac del investigador. **SF-2** El usuario da clic en el botón actualizar enseguida puede consultar la información personal, idiomas, estudios, distinciones, experiencia profesional y un resumen de la hoja de vida.
Excepciones	**E-1** Debe dar clic en actualizar para visualizar la información.

Caso de uso:	Editar
Actores:	Administrador
Propósito:	Permite editar la información deseada de los usuarios.
Resumen:	En este caso de uso permite editar la información de los usuarios que está registrada en el sitio.
Precondición:	• El usuario debe tener registrada la hoja de vida en el sitio.
Flujo principal:	El caso de uso inicia cuando ingresa el link del CvLac o del ResearchGate del usuario y enseguida se da clic en el botón

	actualizar.
Sub-flujo	**SF-1** Se muestra un cuadro donde el usuario debe ingresar el link del CvLac del investigador. **SF-2** El usuario da clic en el botón actualizar para poder acceder a la plataforma y editar la información que desee ya sea información personal, idiomas, estudios, distinciones, experiencia profesional.
Excepciones	**E-1** Debe dar clic en actualizar para visualizar su información y poder editar los diferentes campos.

Caso de uso:	Extraer
Actores:	Administrador
Propósito:	Permite extraer la información deseada de los usuarios.
Resumen:	En este caso de uso permite extraer información de los usuarios que está registrada en el sitio web de Colciencias-CvLac o del ResearchGate.
Precondición:	• El usuario debe tener registrada la hoja de vida en el CvLac o en ResearchGate.
Flujo principal:	El caso de uso inicia cuando ingresa el link del CvLac o del ResearchGate del usuario y enseguida se da clic en el botón actualizar, enseguida en estudios, distinciones y experiencia profesional aparece una ventana de información sugerida con la información que se extrajo de alguna de las páginas y que nos servirá para llenar los campos.
Sub-flujo	**SF-1** Se muestra un cuadro donde el usuario debe ingresar el link del CvLac del investigador. **SF-2** El usuario da clic en el botón actualizar para extraer la información. **SF-3** En estudios, distinciones y experiencia profesional el

	usuario puede visualizar la información extraída en una ventana de información sugerida.
Excepciones	**E-1** Debe dar clic en actualizar para visualizar la información extraída. **E-2** el usuario debe tener su hoja de vida registrada en el CvLac o en ResearchGate.

Caso de uso:	Generar PDF
Actores:	Administrador
Propósito:	Permite generar PDF de la hoja de vida de los usuarios.
Resumen:	En este caso de uso permite generar el PDF de la hoja de vida de los usuarios.
Precondición:	• El administrador debe agregar su información a la hoja de vida.
Flujo principal:	El caso de uso inicia cuando ingresa el link del CvLac del usuario y enseguida se da clic en el botón actualizar, enseguida puede ver el resumen de su hoja de vida si ya la tiene registrada y generar el PDF; de lo contrario agrega la información antes de generar el PDF.
Sub-flujo	**SF-1** Se muestra un cuadro donde el usuario debe ingresar el link del CvLac del usuario. **SF-2** El usuario da clic en el botón actualizar para extraer la información. **SF-3** el usuario da clic en resumen para visualizar el resumen de su hoja de vida. **SF-4** el usuario da clic en PDF para generar el PDF de su hoja de vida.
Excepciones	**E-1** Debe tener la hoja de vida registrada en la plataforma.

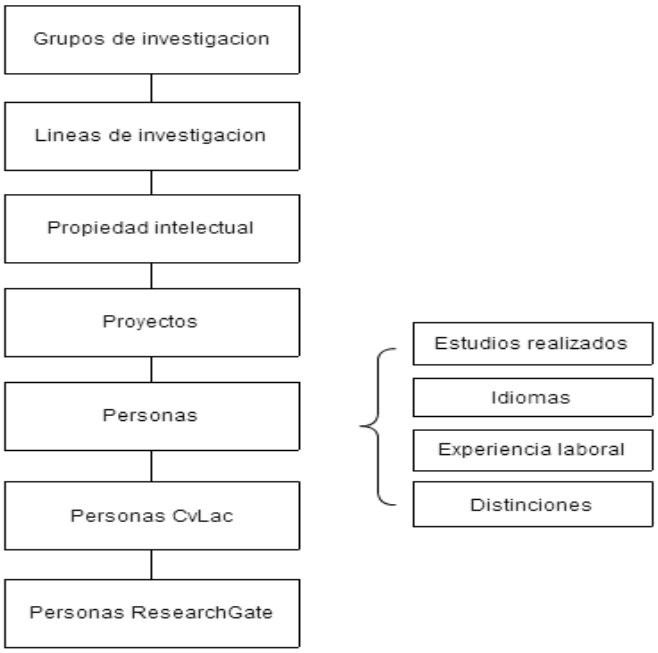
7. DISEÑO COMPONENTES

7.1. CARACTERIZACIÓN Y DISEÑO DE ENTIDADES

La base de datos que se utilizó en este proyecto fue MongoDB para la elaboración de la base de datos se determinaron las siguientes entidades iniciales como líneas de investigación, grupos de investigación, personas, personas CvLac, personas ResearchGate y propiedad Intelectual. Personas tiene colecciones embebidas: Estudios realizados, idiomas, experiencia laboral, distinciones.

Se representó la relación mediante el siguiente modelo conceptual ya que la base de datos es no relacional; no se puede representar a través de diagramas entidad relación.

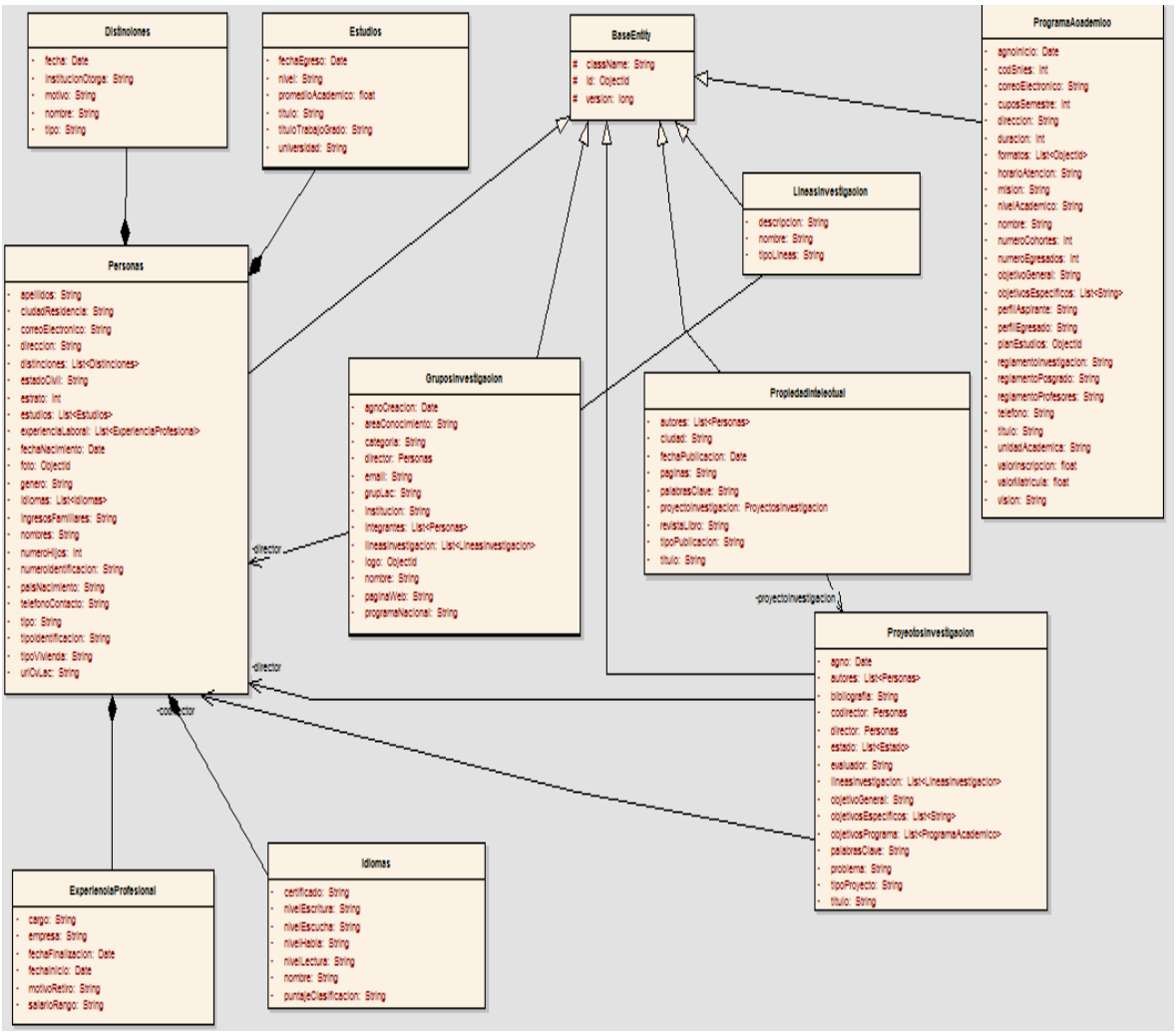
Figura 8. Modelo conceptual de la base de datos



7.2. DIAGRAMA DE CLASES

Después de elaborar el modelo conceptual se creó el diagrama de clases de la base de datos de mongoDB. Para la creación de las clases de dicho modelo se utilizó Morphia, este framework toma las entidades (@Entity) como personas, propiedad intelectual, personal ResearGate y personal CvLac y las transforma en una clase con atributos similar a los que tiene en las propiedades en la base de datos. Se agregó una entidad *BaseEntity* que nos permitiera heredar todas las propiedades y utilizar el concepto de herencia y polimorfismo.

Figura 9. Diagrama de Clases



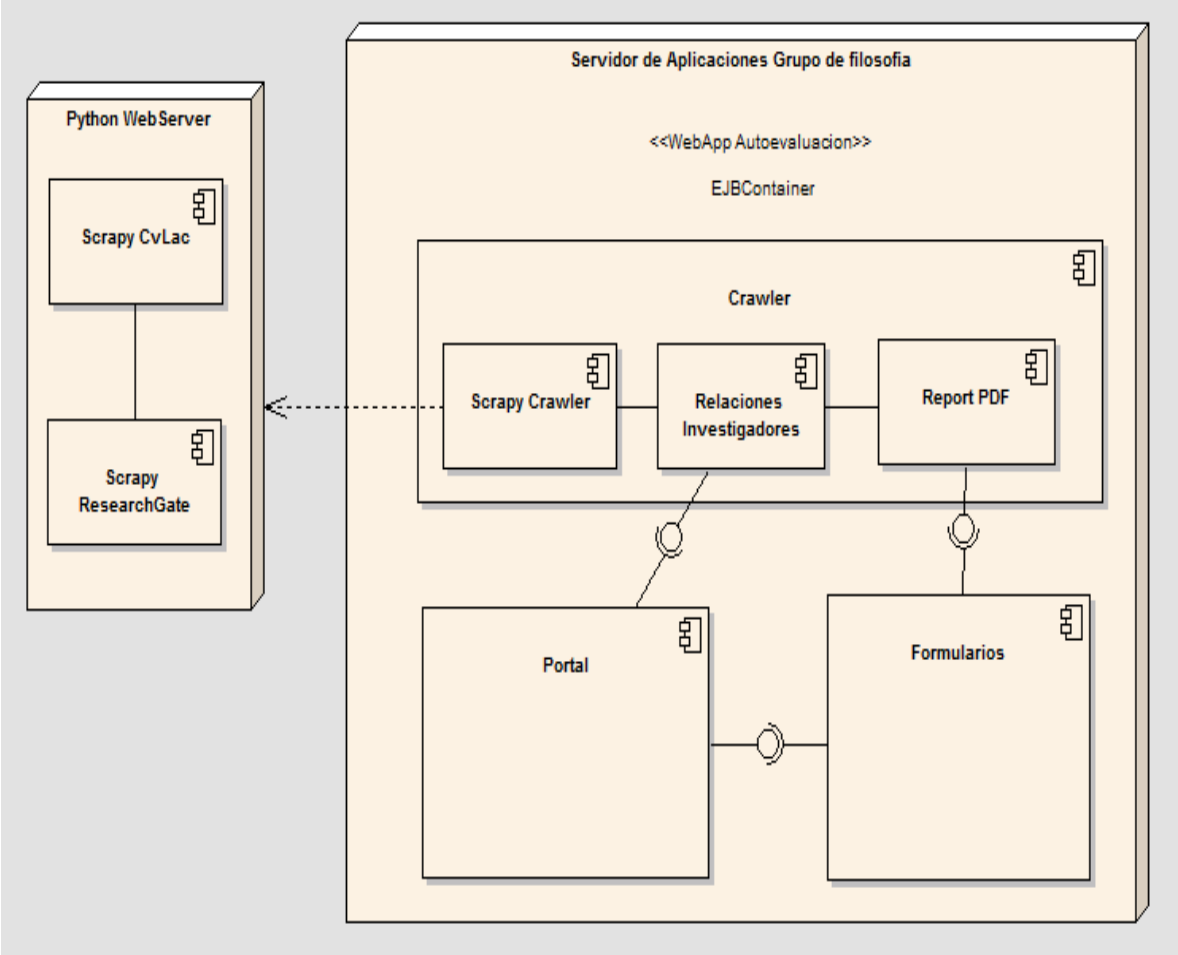
7.3. DISEÑO DE COMPONENTES DEL SISTEMA

Un diagrama de componentes permite visualizar con más facilidad la estructura general del sistema, A continuación se muestra el sistema de autoevaluación dividido en componentes y las dependencias entre ellos. Este sistema está basado en una interfaz orientada a la web. Se puede apreciar principalmente dos tipos de nodos. En este proyecto se desarrolló el componente Crawler y el nodo Python WebServer.

El nodo Python WebServer contiene dos componentes: Scrapy Cvlac y Scrapy ResearchGate. En este nodo se hacen los procesos de extracción de datos.

El módulo Crawler tiene tres componentes: Scrapy Crawler, Relaciones Investigadores y Report PDF. Este módulo se comunica con Python WebServer con el portal y formularios. El modulo Relaciones Investigadores lleva los procesos para representar las relaciones entre los investigadores y mostrarlas en el portal y el módulo Report PDF se encarga de generar el PDF de la información que se aceptó en los formularios.

Figura 10. Diagrama de Componentes

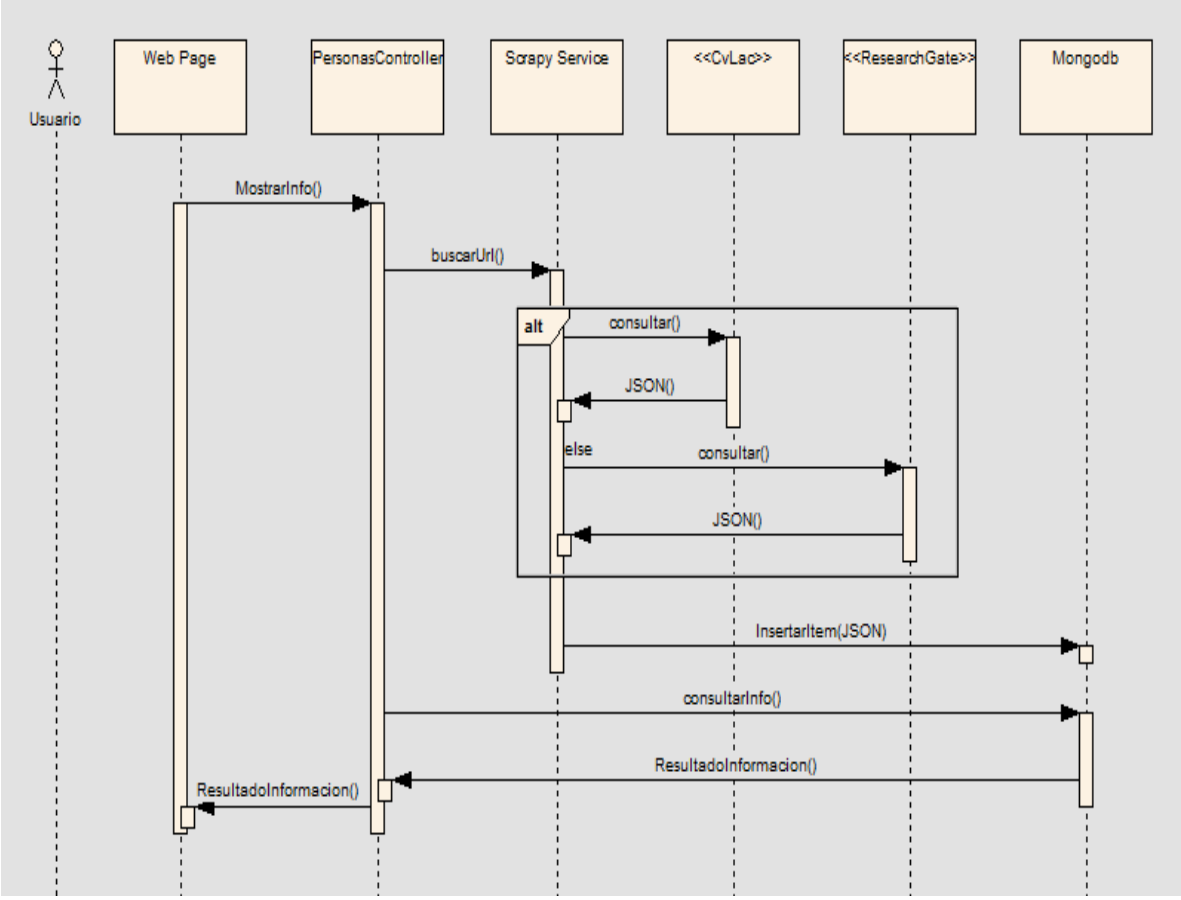


7.4. DIAGRAMA DE SECUENCIAS

Con el siguiente diagrama de secuencias, se representan detalladamente mediante un esquema conceptual el comportamiento del sistema. A continuación se muestra como Web Page interacciona con PersonasController toma la URL y ejecuta el proceso buscarUrl; en Scrapy Server se llama un archivo.py para extraer la información del CvLac o del ReserarchGate, estos retornan los datos extraídos en un JSON que a su vez son guardado en la base de datos MongoDB.

Mediante PersonasController consultamos la información en la base de datos para después verlos en Wep Page.

Figura 11. Diagrama de secuencia



7.5. TÉCNICAS DE PROGRAMACIÓN Y PATRONES UTILIZADOS

Una de las técnicas utilizadas fue el modelo vista controlador MVC, es un patrón de arquitectura software se compone de tres componentes que son el modelo, la vista y el controlador.

Modelo: contiene toda la información almacenada en la base de datos junto con la reglas de negocio encargada de alterar la información según acciones del usuario.
Vista: es la presentación del modelo en una forma ordenada y presentable para el usuario.

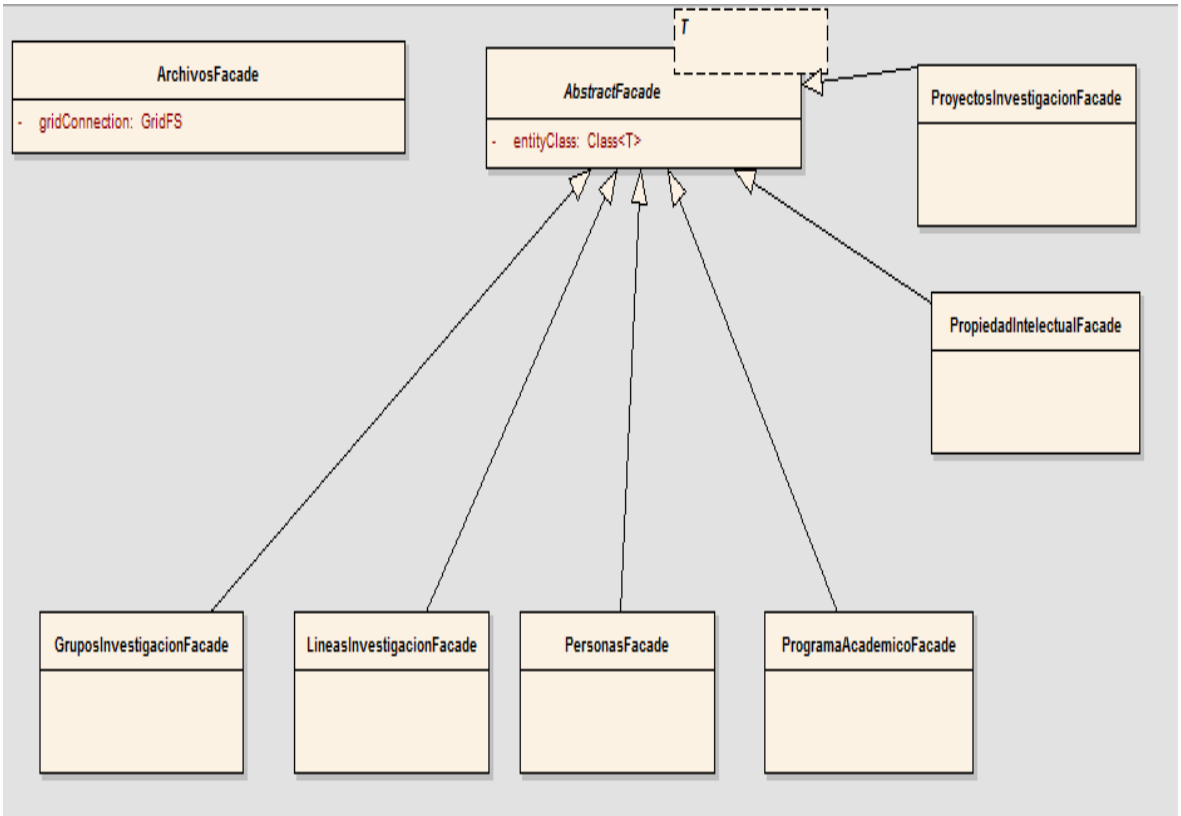
Controlador: código que obtiene datos dinámicamente y son generados para la vista al ser requeridos por el usuario.

El MVC funciona así: el usuario realiza una acción, el controlador interpreta la entrada del usuario, generando un mensaje de acción al modelo que le retorna una respuesta, seguido de esto el controlador envía a la vista, la vista toma los datos y los deja listos para entregar de nuevo.

7.5.1. Patrón Facade. Proporciona una interfaz unificada para un conjunto de interfaces de un sistema. Define una interfaz de alto nivel que hace que el subsistema sea más fácil de usar.

En la figura se puede observar el diagrama de clases del patrón Facade utilizado en el desarrollo del proyecto. En donde se aprecia la generalización hacia la clase AbstractFacade<T> por los cual los subsistemas como PersonasFacade, ArchivosFacade, GruposInvestigacionFacade, LineasdeInvestigacionFacade, PropiedadIntelectualFacade entre otros heredan las propiedades de esta clase lo que hace que reduzca las comunicaciones y dependencias de ciertos componentes

Figura 12. Diagrama de clases del patrón Facade



7.5.2. Patrón de diseño Front Controller. Es un patrón de diseño que se basa en usar un controlador como punto inicial para la gestión de las peticiones. El controlador gestiona estas peticiones, y realiza algunas funciones como: comprobación de restricciones de seguridad, manejo de errores, mapear y delegación de las peticiones a otros componentes de la aplicación que se encargarán de generar la vista adecuada para el usuario.

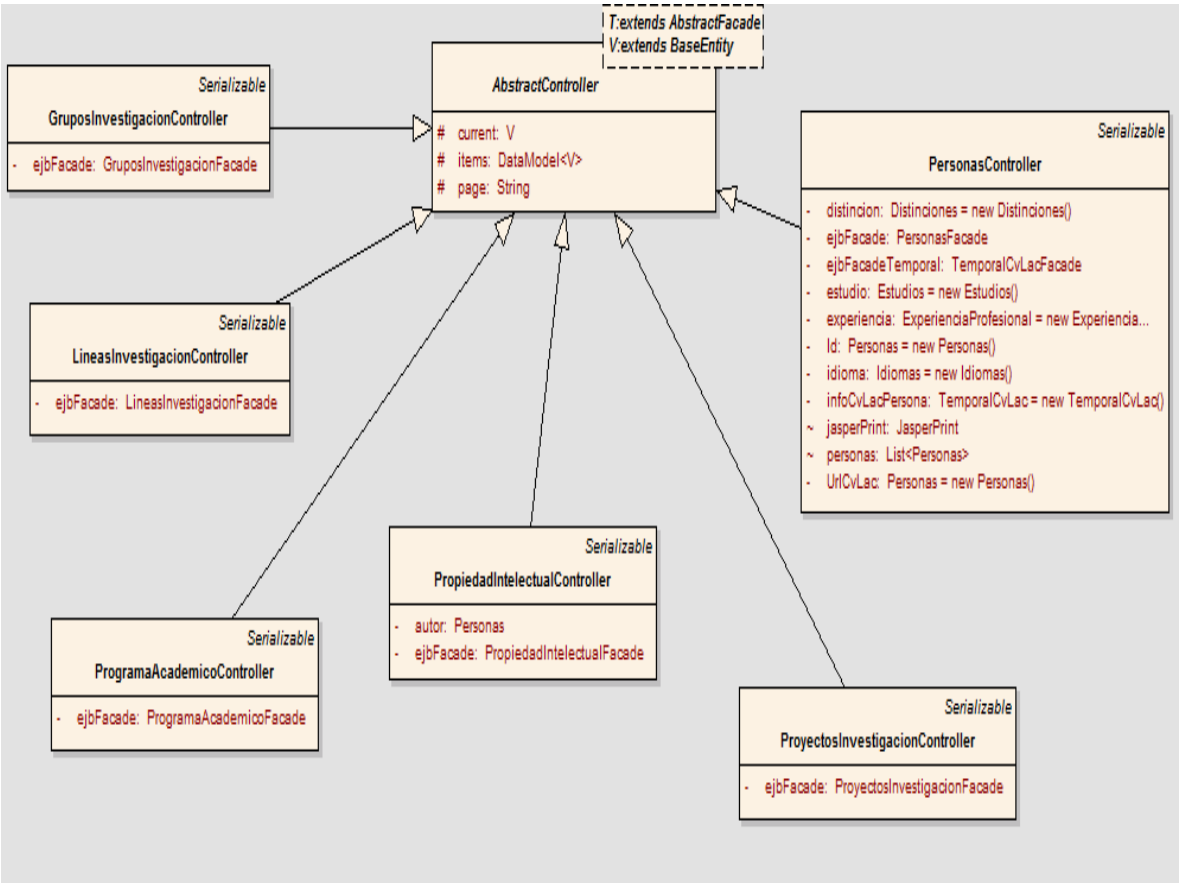
Se divide en 2 diferentes objetos el *Front Controller* y el *Dispatcher*. En ese caso, El *Front Controller* acepta todos los requerimientos de un cliente y realiza la autenticación, y el *Dispatcher* direcciona los requerimientos a manejadores apropiada. Alguna de las ventajas:

- Se centraliza en un único punto la gestión de las peticiones.

- Se aumenta la reusabilidad de código.
- Se mejora la gestión de la seguridad.

Se observa en el diagrama de clases del patrón *Front Controller* sus respectivas generalizaciones y sus diferentes controladores

Figura 13. Diagrama de clases del patrón Front Controller



8. CODIFICACIÓN DEL COMPONENTE

En este capítulo se presenta una descripción del desarrollo del proyecto, el proceso y avance de su ejecución.

8.1. ANÁLISIS DE LAS PÁGINAS USADAS PARA LA EXTRACCIÓN CON SCRAPY

La información consignada en CvLAC y en ResearchGate es individual y debe ser diligenciada personalmente por cada investigador quien realiza la actualización a través de una clave única de acceso al sistema.[6]

Los investigadores que utilizan la plataforma la página del ResearchGate pueden crear un perfil personal y unirse a grupos de su interés. Otra aplicación importante es que la plataforma funciona como una bolsa de trabajo internacional para la comunidad científica

8.1.1. Análisis de CvLac. Para la extracción de la información se usó varias hojas de vida de personas que están registradas. En las hojas de vida en la parte superior aparecen los módulos que se registra en este sitio, las cuales son datos generales, producción bibliográfica, producción técnica y más información sobre estas opciones se despliega la información a extraer. Una vez ubicados en el menú que despliega la información se inspecciona elemento por elemento; Los módulos se dividen en tablas y dentro estas tablas se encuentran la información a extraer las cuales están en celdas.

[6] http://www.colciencias.gov.co/glosario

- El módulo información Académica está ubicada en tablas, la información de los diferentes estudios se dividen por filas y su contenido en columnas.

Figura 14. Análisis CvLac

```
<tr>
    <td><li> </li></td>
    <td><strong>Doctorado</strong>
        Universidad Pedagógica Nacional - U.P.N.<br />
        Doctorado En Educación<br />
        Juliode2006 - Diciembrede 2011<br />
    Formación y argumentación. Automatización de procesos argumentativos</td>
</tr>

<tr>
    <td><li> </li></td>
    <td><strong>Maestría/Magister</strong>
        Universidad Industrial de Santander - UIS<br />
        Maestría En Informática<br />
        Juliode1999 - Octubrede 2003<br />
    Diseño de un entorno virtual para favorecer la independencia cognoscitiva de los estudiantes de educación superior</td>
</tr>

<tr>
    <td><li> </li></td>
    <td><strong>Pregrado/Universitario</strong>
        Universidad Industrial de Santander - UIS<br />
        Ingeniería de Sistemas<br />
        Febrerode1992 - Octubrede 1998<br />
    Planeación estratégica de sistemas para laboratorio de genética UIS</td>
</tr>

</table>
```

A partir del análisis anterior de la página del CvLac se diseñó la siguiente colección para el almacenamiento de la información.

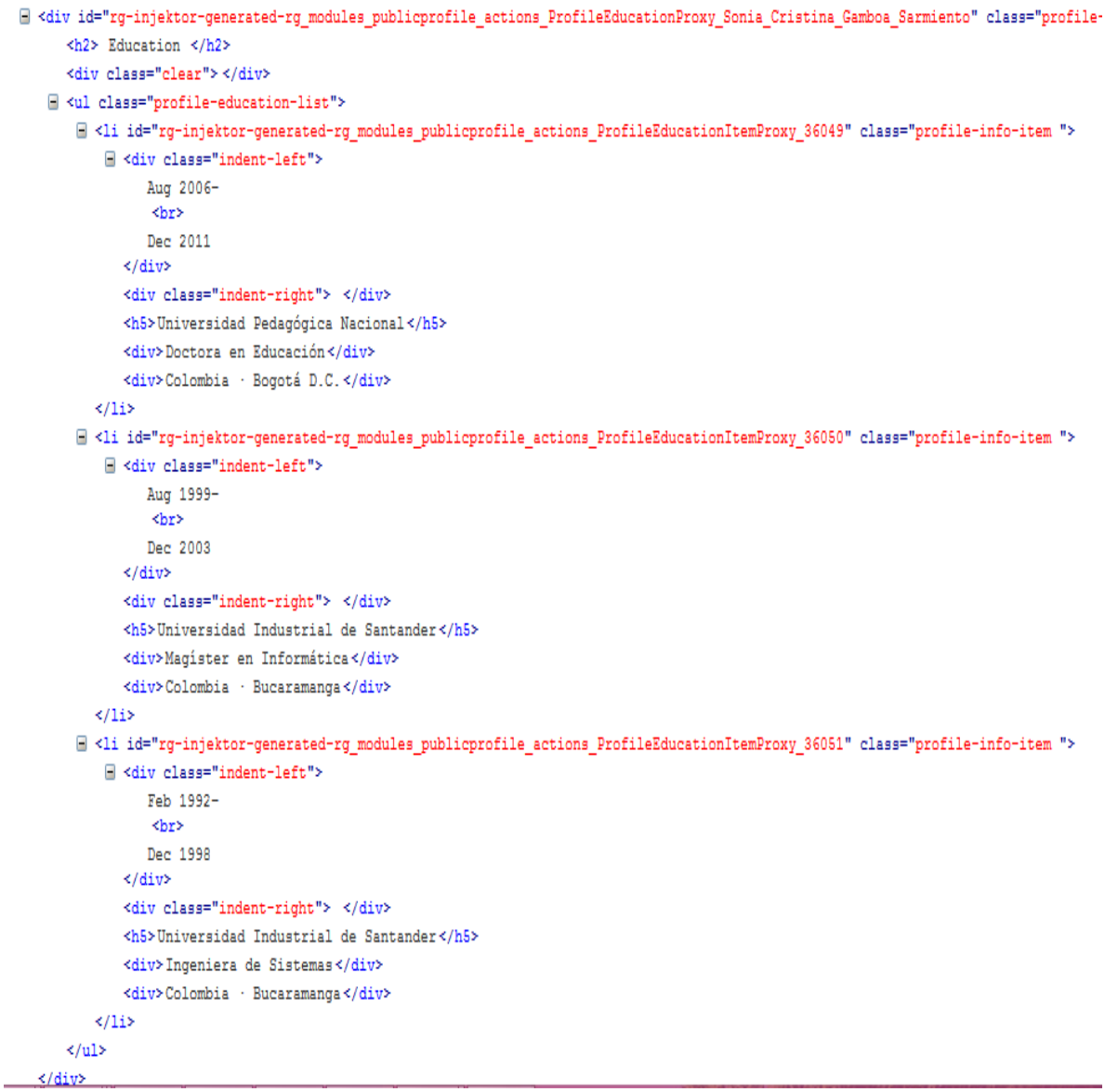
Tabla 1. Personas CvLac

Personal CvLac	Tipo	Descripción
Nombre	string	Nombre investigador
Identificación	string	Identificación investigador
url	string	url del CvLac investigador
formacion_academica	string	Formación académica del investigador
formacion_complementaria	string	Otros cursos del investigador
experiencia_profesional	string	Experiencia del investigador
idiomas	string	Idiomas que habla el investigador
lineas_investigacion	string	Líneas de investigación donde ha profundizado
reconocimientos	string	reconocimientos
cursos	string	Cursos realizado
trabajos_dirigidos	string	Que trabajos a dirigido
jurado_comite_evaluacion	string	Si ha sido jurado del comité de evaluación
participacion_comite_evaluacion	String	Si ha participado del comité de evaluación
par_evaluador	String	Par evaluador
eventos_cientificos	String	Eventos científicos que ha asistido
articulos	string	Artículos realizados
libros	String	Libros realizados
capitulos_de_libros	String	Capítulos de libros realizados
trabajos_eventos	String	Trabajos en eventos realizados
prototipos	string	prototipos
otra_produccion	String	Otros trabajos
software	String	Software realizados
produccion_tecnologica	String	Producción tecnológica
informes_de_investigacion	String	Artículos de investigación
trabajos_tecnicos	String	Trabajos técnicos realizados
normas	String	normar
mas_trabajos	String	Otros trabajos
proyectos	String	Proyectos realizados

8.1.2. Análisis ResearchGate. Los módulos del ResearchGate están divididos por información personal, formación académica, intereses y publicaciones. Los diferentes módulos se encuentra separadas por divisiones ( ) y cada módulo tiene una identificación (id) que lo identifica.

- Para extraer la información de la educación se ubica dentro de la etiqueta principal <ul> e ir raspando las etiquetas que se despliegan de esta.

Figura 15. Análisis ResearchGate



```
<div id="rg-injektor-generated-rg_modules_publicprofile_actions_ProfileEducationProxy_Sonia_Cristina_Gamboa_Sarmiento" class="profile-
    <h2> Education </h2>
    <div class="clear"> </div>
    <ul class="profile-education-list">
      <li id="rg-injektor-generated-rg_modules_publicprofile_actions_ProfileEducationItemProxy_36049" class="profile-info-item ">
        <div class="indent-left">
            Aug 2006-
            <br>
            Dec 2011
        </div>
        <div class="indent-right"> </div>
        <h5>Universidad Pedagógica Nacional</h5>
        <div>Doctora en Educación</div>
        <div>Colombia · Bogotá D.C.</div>
      </li>
      <li id="rg-injektor-generated-rg_modules_publicprofile_actions_ProfileEducationItemProxy_36050" class="profile-info-item ">
        <div class="indent-left">
            Aug 1999-
            <br>
            Dec 2003
        </div>
        <div class="indent-right"> </div>
        <h5>Universidad Industrial de Santander</h5>
        <div>Magíster en Informática</div>
        <div>Colombia · Bucaramanga</div>
      </li>
      <li id="rg-injektor-generated-rg_modules_publicprofile_actions_ProfileEducationItemProxy_36051" class="profile-info-item ">
        <div class="indent-left">
            Feb 1992-
            <br>
            Dec 1998
        </div>
        <div class="indent-right"> </div>
        <h5>Universidad Industrial de Santander</h5>
        <div>Ingeniera de Sistemas</div>
        <div>Colombia · Bucaramanga</div>
      </li>
    </ul>
  </div>
```

A partir del análisis anterior de la página del ResearchGate se diseñó la siguiente colección para el almacenamiento de la información extraída.

Tabla 2. Personas ResearchGate

Personal ResearchGate	Tipo	Descripción
Nombre	string	Nombre investigador
nivel_de_estudio	string	Doctorado, maestría
topics	string	Temas que maneja
experiencia_profesional	string	experiencia
estudios	string	Estudios realizados
idiomas	string	Idiomas que habla.
63embrecías_cientificas	string	Reconocimientos científicos
habilidades	string	Habilidades
premios	string	Premios
otros_intereses	String	Otros intereses
publicacion	String	Publicaciones realizadas

8.2. EXTRACCIÓN DE DATOS DE SITIOS WEB

A partir del análisis anteriormente realizado se procedió a realizar la extracción de la información contenida en dichos sitios. Se observó un patrón en la URL de cada uno de los perfiles como se presenta a continuación:

CvLac:
http://190.216.132.131:8081/cvlac/visualizador/generarCurriculoCv.do?cod_rh=CODIGO_INVESTIGADOR

ResearchGate:
http://www.researchgate.net/profile/NOMBRE_DEL_INVESTIGADOR

Para realizar el raspado de las páginas anteriores se tuvo en cuenta los siguientes pasos:

1. Creación del proyecto Scrapy.
2. Definición de los elementos a extraer.

3. Escribir la *araña* que rastrea el sitio web y extraer los ítems que desean almacenar.
4. Escribir un archivo llamado pipelines donde se especifique cómo y dónde guardar la información raspada.

8.2.1. Creación del proyecto

- Se escoge en que directorio almacenar el código
- Se abre la consola del sistema y se ejecuta lo siguiente NombreProyecto startproject scrapy

Este creara un directorio con el siguiente contenido

- NombreProyecto /
- scrapy.cfg
- NombreProyecto /
- __init__.py
- items.py
- pipelines.py
- settings.py
- spiders /
- __init__.py
- ...

Scrapy.cfg = Archivo de configuración del proyecto
NombreProyecto / = Modulo pitón del proyecto.
NombreProyecto / ítems.py = Archivos de ítem del proyecto.
NombreProyecto / pipelines.py = Archivo de tuberías del proyecto, se especifica donde se guarda la información raspada.
NombreProyecto / settings.py = Archivo de configuración.
NombreProyecto / spiders / = Lugar donde se ubica la araña (Rastreo de datos).

8.2.2. Definición de los elementos a extraer. Los ítems almacenan la información que se raspa, esto es útil para que solo guarde los datos que se desean obtener y evitar errores tipográficos.

Se crea una clase donde se declara (Item) y se definen los atributos como (Field) objetos.

Figura 16. Items.py

items.py

```
from scrapy.item import Item, Field

class CvItem(Item):
    # define the fields for your item here like:
    #name = Field()
    name = Field()
    name_cita = Field()
    nacionalidad = Field()
    for_aca = Field()
    for_com = Field()
    exp_prfesional = Field()
    areas_actuacion = Field()
    idiomas = Field()
    lineas_inv = Field()
    reconocimientos = Field()
    cursos = Field()
    trabajos_dirigidos = Field()
    jurado_comite_evaluacion = Field()
    participacion_comite_evaluacion = Field()
    par_evaluador = Field()
    eventos_cientificos = Field()
    articulos = Field()
    libros = Field()
    cap_libros = Field()
    tra_eventos = Field()
    doc_tra = Field()
```

8.2.3. Escribir el spider para el rastreo de datos. Un spider o *araña* es una clases que escribe el usuario para raspar la información esto depende de cada usuario como desee rapar u obtener los datos deseados, la extracción de datos se desarrolla por etapas, lo primero es acceder al sitio web y recorrer el HTML mediante el sistema que el usuario utilice para recolectar la información, pasarla por una tubería (archivo pipelines) dependiendo de los módulos que se deseen usar.

Primero se define una lista de URLs a descargar, la forma de cómo se debe seguir los enlaces y el análisis de las páginas. Después se definen los principales atributos obligatorios que son los siguientes:

- **Name:** Identifica la *araña* (Debe ser único).
- **Start_urls:** Se ubican la dirección o direcciones URL donde la *araña* empieza a rastrear.
- **Parse()** : Es un método de la *araña*, se llama con el request de cada dirección URL, la respuesta pasa al método como único argumento. Analiza los datos de la respuesta y la extracción, procesa la respuesta y la devolución.

8.2.2. Escribir pipelines.py

Figura 17. Info.py

```
from scrapy.item import Item, Field

class CvItem(Item):
    # define the fields for your item here like:
    #name = Field()
    name = Field()
    name_cita = Field()
    nacionalidad = Field()
    for_aca = Field()
    for_com = Field()
    exp_prfesional = Field()
    areas_actuacion = Field()
    idiomas = Field()
    lineas_inv = Field()
    reconocimientos = Field()
    cursos = Field()
    trabajos_dirigidos = Field()
    jurado_comite_evaluacion = Field()
    participacion_comite_evaluacion = Field()
    par_evaluador = Field()
    eventos_cientificos = Field()
    articulos = Field()
    libros = Field()
    cap_libros = Field()
    tra_eventos = Field()
    doc_tra = Field()
```

Figura 18. pipelines.py

pipelines.py

```python
import json
import codecs

class JsonWithEncodingPipeline(object):

    def __init__(self):
        self.file = codecs.open('datos1.json', 'w', encoding='utf-8')


    def process_item(self, item, spider):
        line = json.dumps(dict(item), ensure_ascii=False) + "\n"
        self.file.write(line)
        return item

    def spider_closed(self, spider):
        self.file.close()
```

Cuando se obtienen los datos en el método parser(), el objetivo es devolverlo dentro de un ítem que se encuentra en el fichero llamado ítems.py Scrapy ofrece un sencillo sistema para almacenar los datos raspados para su modificaciones. Pipelines es el encargado de procesar la información de guardarla o modificarla para el fin que se desea utilizar. En este caso la información será almacenada en un archivo .JSON llamado scraped_data_utf8.json

Figura 19. Archivo JSON


```
{"cargo": ["  Director SC3UIS Unit, Professor/Researcher   "], "name": ["Carlos Jaime Barrios Hernandez"],
"actual_tra": [" Industrial University of Santa... "], "habilidades": [" Distributed Systems ",
" Cloud Computing ", " GPU Programming ", " Concurrency ", " High Performance Computing ", " Modeling ", " Simulation ",
" Supercomputing ", " Scientific Computing ", " Grid Computing ", " Advanced Computing ", " Green Computing ",
" Parallel and Distributed Computing ", " Numerical Analysis ", " Computer Architecture ", " Scalable Architectures ",
" Network Architecture ", " E-Science ", " Computer Science ", " System Dynamics Modeling ", " Computer Science Education ",
" Informatics ", " Applied Mathematics ", " Systems Engineering ", " NVIDIA CUDA "], "journal": [" ", " ", "  IJCSI Journal   "],
"otros_intereses": ["  ", " ", "  I'm PhD in Computer Science of the University of Nice - Sophia Antipolis, in France. Nowadays,
I'm responsible of High Performance and Scientific Computing Center at Universidad Industrial de Santander, in Colombia..",
"Also, I'm co-founder member of the Latin American Conference on High Performance Computing,
co-founder of the Supercomputing Camp School and I participe in different projects between Latinamerica and Europe research teams
```

8.2.3. Modificaciones en Scrapy. Inicialmente se creó un spider que tenía las urls a raspar estáticas y guardaba la información en un archivo .JSON que este archivo fuera leído desde un servicio web que llamara la información extraída y de inmediato se guardara en la base de datos, al observar que esta no era la mejor solución se buscó una alternativa más efectiva en la cual se llegó a la conclusión que desde el spider abriera un archivo .txt que contiene las urls de los usuarios registrados y guardara directo a la base de datos.

Por esta razón se hicieron modificaciones en el Scrapy y para lograr la limpieza de código y poder extraer de una forma más ordenada. Las modificaciones realizadas se muestran a continuación

- *Spider*: Leer archivo .txt que contiene las urls de los investigadores una vez obtenida la información deseada es filtrada antes de pasar a la base de datos almacenándola en los contenedores.

Figura 20. Spider_cvlac.py

```
# encoding=utf8

from scrapy.selector import Selector
from scrapy.contrib.linkextractors.sgml import SgmlLinkExtractor
from scrapy.contrib.spiders import CrawlSpider, Rule
from scrapy.spider import Spider
from cv.items import CvItem


class CvlacSpider(Spider):
    def __init__(self, url):
        self.start_urls = [url]

    name = 'cvlac'
    allowed_domains = ['190.216.132.131']


    def parse(self, response):

        sel = Selector(response)

        i = CvItem()
        i['nombres'] = {}
        i['paisNacimiento'] = {}
        i['estudios'] = []
        i['experienciaProfesional'] = []
        i['idiomas'] = []
        i['lineasDeInvestigation'] = []
        i['distinciones'] = []
        i['eventosCientificos'] = []
        i['articulos'] = []
        i['libros'] = []
```

- *Pipelines*: En el archivo pipelines se encuentra la configuración de cómo se guardaran los datos en la base de datos, tomando los contenedores y llevándolo a la base de datos.

Figura 21. pipelines.py

- *Settings*: Se especifica la base de datos y la colección donde se almacena la información extraída.

Figura 22. Settings.py

settings.py

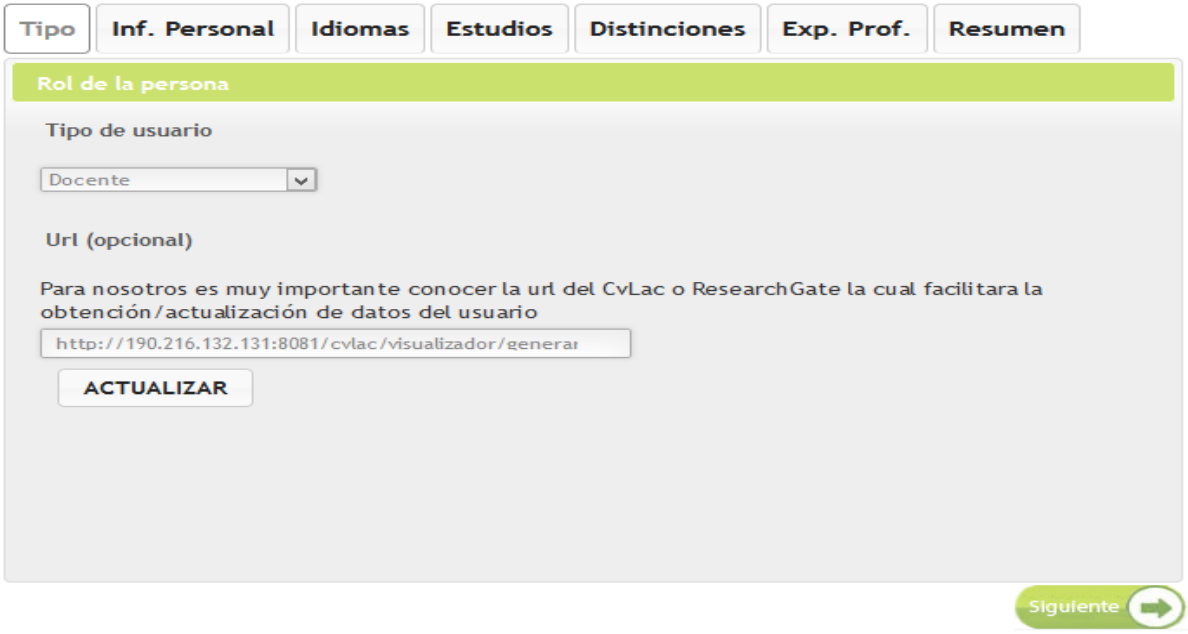
```
1
2   BOT_NAME = 'cvlac_scraping'
3
4
5
6   SPIDER_MODULES = ['cvlac_scraping.spiders']
7   NEWSPIDER_MODULE = 'cvlac_scraping.spiders'
8   ITEM_PIPELINES = ['cvlac_scraping.pipelines.MongoDBPipeline',
9   ]
10
11  MONGODB_URI = 'mongodb://localhost:27017'
12  MONGODB_DATABASE = 'usuarios'
13  MONGODB_COLLECTION = 'my_items'
14
15  RETRY_TIMES = 0
16
```

8.3. DESARROLLO DE LA INTERFAZ DE PRESENTACIÓN DE RESULTADOS

En la interfaz gráfica donde el usuario digita su información personal, hay una opción para escribir la URL y un botón actualizar, al darle clic este llama al scrapy donde raspa la información de la URL que el usuario digito y guarda la información raspada en una colección de la base datos llamada proyecto.

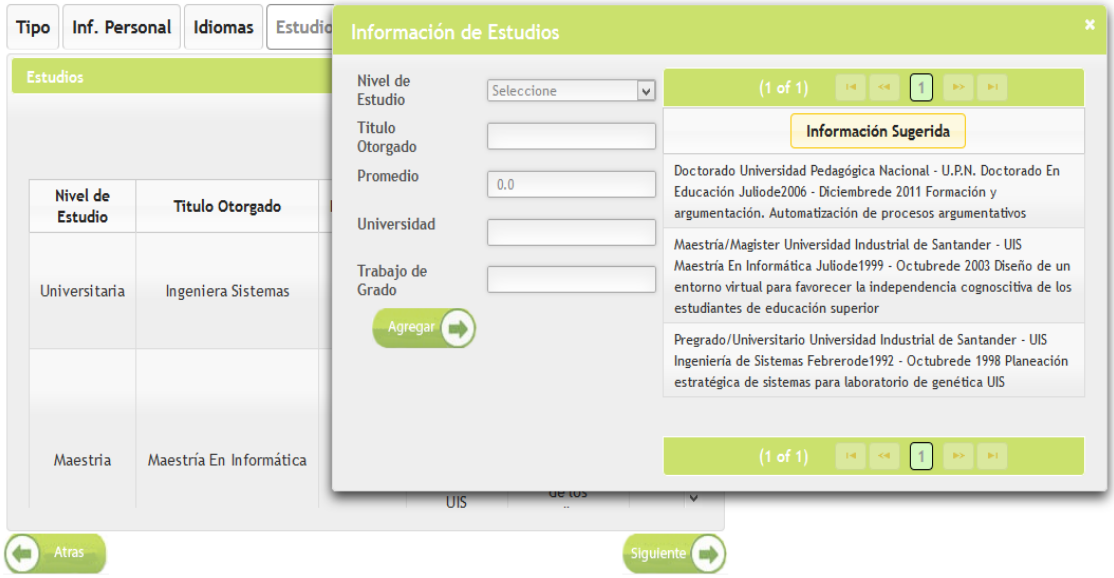
Mediante esta interfaz el componente desarrollado se comunica e integra con el componente formularios que es el proyecto creado por Oscar Duitama y Jhonatan Arias.

Figura 23. Interfaz final



Al momento que el usuario está registrando su información al lado derecho hay un botón de información sugerida al momento que el de clic se mostrara los datos que se extrajo de la url que registro anteriormente.

Figura 24. Interfaz estudios



Al finalizar el registro de datos se encuentra la opción resumen donde muestra toda la información que fue registrada, encontrándose con una opción de descargar un documento en formato PDF con esa información.

Figura 25. Interfaz resumen

8.4. GENERACIÓN DE REPORTES

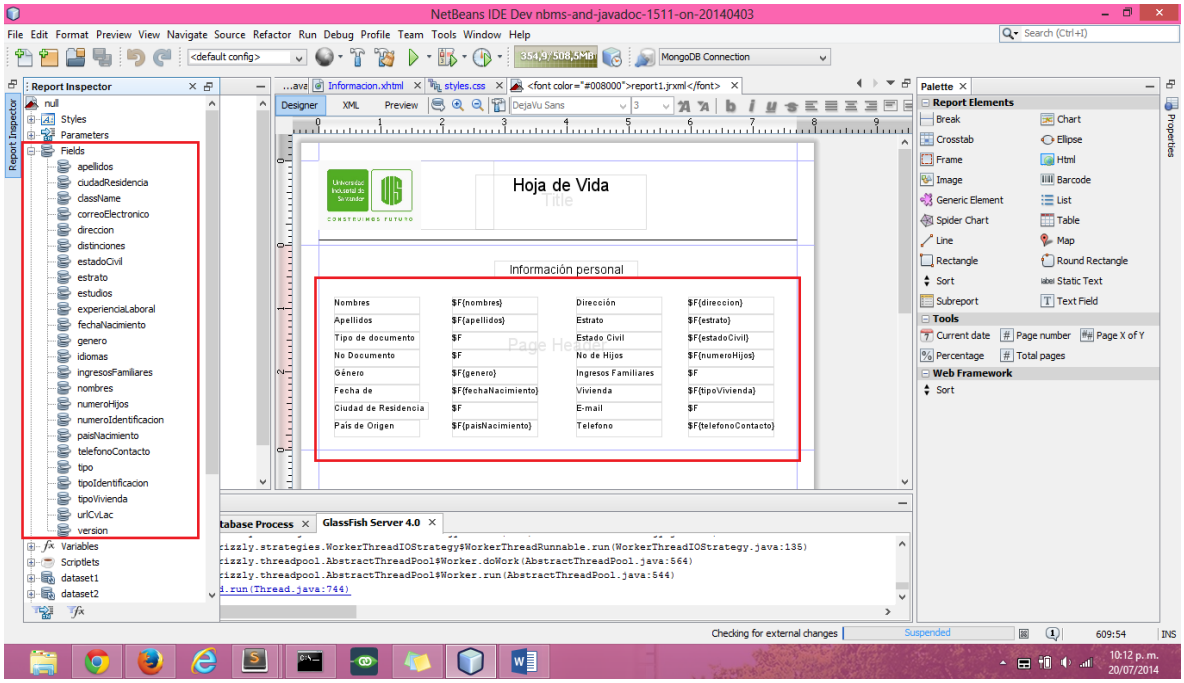
JasperResport (iReport) es un motor de informes de código abierto. Está escrito en JAVA capaz de usar cualquier fuente de datos y generar documentos que se pueden exportar en varios formatos, HTML, PDF, EXCEL, WORD, OPT, CSV, DOC.

Para generar reporte PDF de las hojas de vida de los usuarios registrados, se debe crear un método que recupere los datos de la base de datos, a continuación se utiliza una herramienta de iReport que genera una plantilla JRXML. Este es un archivo xml que describe el diseño este se debe compilar para generar un archivo

Jasper en su formato binario. El proceso que se utilizó para compilarlo fue desde la opción compilar de iReport.

Se utilizó las clases *JRBeanCollectionDataSource* y Estos métodos reciben como parámetro el objeto del diseño del informe. Para exportar el archivo PDF se usó *JasperEXportManager*

Figura 26. Generación de reportes



Report Inspector muestra la estructura completa del informe, que se compone de (campos, parámetros y variables). Los archivos *fields* deben llamarse tal cual como aparece en la base de datos.

Report Designer es donde se diseña visualmente el informe.

8.5. DESCRIPCIÓN DEL COMPONENTE DE RELACIONES ENTRE INVESTIGADORES.

Para la realización del componente de relaciones entre investigadores se genero un gráfico implementando HyperTree de JavaScript InfoVis Toolkit (a partir de ahora llamado “JIT”). Se trabajara con tecnologías open source como PHP y JavaScript. Y los datos se tomaran de una base de datos en MongoDB , de la cual se deberá representar gráficamente la información correspondiente a los Grupos de Investigación, que líneas de investigación siguen y que personas integran cada grupo.

8.5.1. Análisis

- HyperTree JIT

A partir del análisis de la documentación de JIT se observó que trabaja con objetos JavaScript aprovechando al máximo el encapsulamiento, donde solamente se debe pasar la información que se desea mostrar en el formato que el objeto *HyperTree* requiere, así como también se pyede pasar algunos datos de configuración.

La información la recibe en un objeto JavaScript armado de la siguiente manera:

```
Obj = {
        Id: “1”,
        name: “nodo 1”,
        children: [{
                id: “01”,
                name: “nodo 01”,
                children: [{
                        id: “001”,
                        name: “nodo 001”,
                        children: []
                }, {
                        Id: “002”,
```

```
                    name: “nodo 002”,
                    children: []
                }
        }, {
                Id: “02”,
                name: “nodo 02”,
                children: [{
                    id: “002”,
                    name: “nodo 003”,
                    children: []
                }, {
                    Id: “003”,
                    name: “node 003”,
                    children: []
                }
        }
}
```

Este es un árbol simple con un nodo principal “nodo 1” del que se desprenden dos nodos de 2do grado “nodo 01” y “nodo 02”, a su vez de estos dos son padres de otros nodos de 3er grado: “nodo 001”, “nodo 002” y “nodo 003”.

El “nodo 002” tiene la particularidad de estar repetido en los dos nodos de 2do grado, es así como el objeto *HyperTree* se entera de que este nodo es hijo compartido de los de un grado superior.

Este es el formato de entrada de *HyperTree*, es decir, se debe garantizar un objeto con esas características a partir de la base de datos, ya sea del lado del servidor como del lado del cliente.

Como se va a trabajar con objetos JSON, y este es el estándar para el proceso de transmisión de datos AJAX, se eligió convenientemente moldear los datos con PHP del lado del servidor y mediante una petición GET pasar esos datos JSON al cliente ya formateados.

- Análisis del origen de datos

En cuanto a la base de datos, se dará uso a tres colecciones: gruposinvestigacion, lineasinvestigacion y personas. Con las siguientes relaciones:

gruposinvestigacion1, varias ---------- 1, varias lineasinvestigacion
gruposinvestigacion1, varias ---------- 1, varias personas

A partir de esto se ha visto conveniente determinar que los nodos generados a partir de gruposinvestigacion corresponden a un grado superior con respecto a líneasinvestigacion y personas. Es decir que cada nodo de gruposinvestigacion tendrá como nodos hijos a los nodos generados a partir de las otras dos colecciones.

8.5.2. Proceso

- **API (server)**

Aquí se generará el código en PHP que le dará al cliente acceso a la base de datos, aquí es donde se formateará un JSON para pasarlo como respuesta a un GET del cliente.

Se generó un archivo llamado server.php y el proceso de codificación es el siguiente: Se conecta con la base de datos del proyecto de autoevaluación, en caso de fallar la conexión se genera una excepción de tipo MongoC*onnectionException* y se emite un mensaje que informará del error.

```
try{
$mongodb = new Mongo("mongodb://localhost:27017");
$gruposInvCollection = $mongodb->proyecto->gruposInvestigacion;
$lineasInvCollection = $mongodb->proyecto->lineasInvestigacion;
$personasCollection = $mongodb->proyecto->personas;
}
catch(MongoConnectionException $e){
die("failed to connect to MongoDB ". $e->getMessage());
}
```

Para poder generar el objeto JSON en PHP se trabaja con arrays que después pueden ser transformados a string mediante json_encode y quedaran con el formato JSON. Por lo tanto se generará un array con el nodo principal del cual se desprenderán todos los demás.

```
$dataInfoVis = array(
  'id' => 'grupos',
  'name' => 'Grupos',
  'children' => array(),
);
```

Posteriormente se analizó la colección de gruposinvestigacion, donde por cada grupo se generará otro objeto array con todos los datos del mismo en el array children del nodo principal.

```
$dataInfoVis["children"][$grupoInvCursorIndex] = array(
  'id' => 'grupo_' . $grupoInvCursorIndex,
  'name' => $grupoInv['nombre'],
  'children' => array()
);
```

A su vez en este array o nodo también debe de haber otro array children que se irá llenando recorriendo primeramente las relaciones en gruposinvestigacion con personas (integrantes) y luego con lineasinvestigacion

```
for ($i=0; $i < count($grupoInv["integrantes"]); $i++) {
  …..
}

for ($i=0; $i < count($grupoInv["lineasInvestigacion"]); $i++) {
  …..
}
```

A su vez por cada integrante y por cada línea de investigación hay que crear un objeto array como children del correspondiente nodo. Como son las últimas ramas del árbol en este caso el array children de estos nodos estará vacío.

```
$dataInfoVis["children"][$grupoInvCursorIndex]["children"][$childrenIndex] = array(
  '_id' => $auxLineaInv["_id"],
  'id' => 'linea_' . count($cacheLineasInv),
  'name' => $auxLineaInv["nombre"],
  'children' => array()
);
```

Además agregando un array “data” a cada nodo se pueden especificar diferentes aspectos visuales para el objeto *HyperTree* tales como: $type, que le asignara la

forma geométrica al nodo (cuadrado, circulo, estrella), $dim, que le asignara un tamaño, $color, que permite cambiar el color del nodo, y algunas otras propiedades. Y además algunos datos que se requieran del lado del cliente respecto del nodo.

A este código nos corresponde agregarle todas las validaciones correspondientes dependiendo del caso y además un sistema de cache para poder evitar recorrer toda una colección de datos cuando el nodo este duplicado, es decir ya sea hijo (es decir que este en el array children) de otro nodo.

Para finalizar se manda la respuesta mediante un echo, primeramente transformado a string mediante json_encode().

```
echo json_encode($dataInfoVis);
```

- Cliente

Del lado del cliente se tendrá que hacer una petición AJAX al server que responderá con el string JSON. Para esto se utilzó JQuery que simplificará el código que ira añadido en el HTML utilizando los CDNs de google.

Y además en el archivo HTML (index.html) se debe añadir un div contenedor del grafico generado por el objeto *HyperTree*, en este caso denominado “infovis” como id para poder identificarlo luego. Mediante JavaScript lo primero se realiza una petición AJAX que va a traer todos los datos. Y en la respuesta se llama a una función “init” que mediante los datos de respuesta será la encargada de iniciar todo el mecanismo.

```
function getData() {
$.get("server.php",function (response) {
init(response);
})
};
```

Dentro de la función “init” lo primero que corresponde hacer es generar el objeto JavaScript a partir del string de respuesta mediante JSON.parse().

```
var json = JSON.parse(jsonString);
```

Luego se crea el objeto *HyperTree* al que le envían un objeto JavaScript con todos los datos de configuración. En este objeto se debe informar cual va a ser el contenedor HTML (el div con id “infovis*”), width* y *height*, información respecto a los nodos y a las relaciones en general que se aplicaran en todo el grafico y que podrá ser sobre escrita o no por los datos que se envíen en cada nodo como color, forma, y demás.

En este objeto también puede manejar eventos que se van disparando a medida que el objeto *HyperTree* esté trabajando tales como *onCreateLabel* donde se muestra el nombre del nodo y *onPlaceLabel* donde cambiará el estilo de los nombres según que nodo este seleccionado.

```
var ht = new $jit.Hypertree({
injectInto: 'infovis',
width: w,
height: h,
Node: {
overridable: true,
dim: 12,
color: "#900"
},
Edge: {
overridable: true,
lineWidth: 3,
color: "#666"
},
onCreateLabel: function(domElement, node){
domElement.innerHTML = node.name;
},
onPlaceLabel: function(domElement, node){
var style = domElement.style;
style.display = '';
style.cursor = 'pointer';
if (node._depth <= 1) {
style.fontSize = "0.8em";
style.color = "#111";
```

```
} else if(node._depth == 2){
 style.fontSize = "0.7em";
style.color = "#333";
} else {
style.display = 'none';
}
var left = parseInt(style.left);
var w = domElement.offsetWidth;
style.left = (left - w / 2) + 'px';
},
});
```

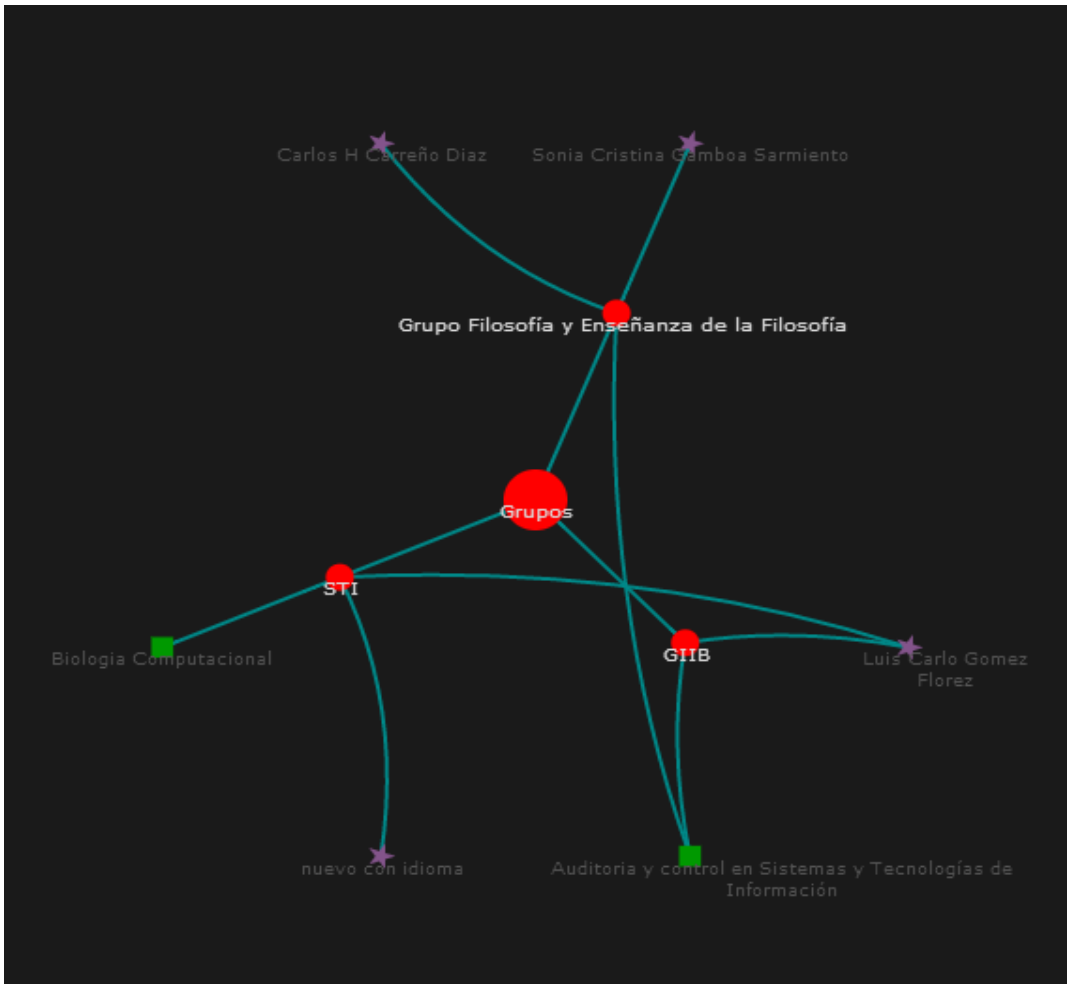
Este nuevo objeto instanciado llamado “ht” es alimentando con los datos utilizando un método propio de JIT loadJSON().

```
ht.loadJSON(json);
```

Además se debe agregar que agregar las correspondientes validaciones y funcionalidades Cross browser necesarias.

8.5.3. Resultados. Los resultados se presentan en la siguiente imagen donde se puede observar un árbol con un nodo principal llamado grupos donde se desprenden 3 nodos de segundo grado que son los grupos de investigación y a su vez de ellos son padres de los nodos de tercer grado investigadores y líneas de investigación. Las líneas de investigación están representadas con un cuadro verde y los investigadores con una estrella morada.

Figura 27. Relaciones entre los investigadores



9. PRUEBAS

Se realizaron pruebas de funcionalidad y pruebas de usabilidad con el fin de encontrar falencias y poder mejorar los componentes.

9.1. PRUEBAS DE FUNCIONALIDAD

Para las pruebas de funcionalidad se utilizó el método de la caja negra. Las pruebas de caja negra se llevan a cabo sobre la interfaz del software, obviando el comportamiento interno y la estructura del programa.

Las pruebas de la caja negra pretenden demostrar que:

- Las funciones del software son operativas
- La entrada se acepta de forma correcta
- Se produce una salida correcta
- La integridad de la información externa se mantiene

Las pruebas de caja negra pretenden encontrar estos tipos de errores:

- Funciones incorrectas o ausentes
- Errores en la interfaz
- Errores en estructuras de datos o en accesos a bases de datos externas
- Errores de rendimiento
- Errores de inicialización y de terminación

Tabla 3. Prueba funcionalidad

Caso Prueba	Descripción	Valores entrada	Respuesta	Resultado
URL correcta	Se ingresa un valor correcto de la url registrada en CvLac o ResearchGate	URL: http://190.216.132.131:8081/cvlac/visualizador/generarCurriculoCv.do?cod_rh=0000380059	Inicio de sesión valido. Sonia Gamboa Consultar Editar Extraer hoja de vida. Generar PDF	OK
URL correcta	Se ingresa un valor correcto de la url registrada en CvLac o ResearchGate	URL: http://190.216.132.131:8081/cvlac/visualizador/generarCurriculoCv.do?cod_rh=0000162701	Inicio de sesión valido. Carlos Barrios Consultar Editar Extraer hoja de vida. Generar PDF	OK
URL incorrecta	Se ingresa un valor no valido	URL: http://190.216.132.131:8081/cvlac/visualizador/generarCurriculoCv.do?cod_rh=0000	URL parcialmente correcta	OK
URL incorrecta	Se ingresa un valor no valido	URL: http://190.216.132.131:8081/cvlac/generarCurriculoCv.do?cod_rh=0000162701	URL parcialmente correcta r	OK
URL incorrecta	Se ingresa un valor no valido	URL: (NULO)	Debe pegar su URL	OK
URL incorrecta	Se ingresa un valor no valido	URL: https://co.linkedin.com/pub/sonia-cristina-gamboa-sarmiento/20/8/345	URL incorrecta	OK

9.2. PRUEBAS DE USABILIDAD

Se realizó un cuestionario en los usuarios pada medir de manera practica la usabilidad a partir de las tareas reales de los usuarios.

Este test nos permitirá verificar los posibles problemas de usabilidad, y encontrar posibles soluciones para los problemas encontrados.

9.2.1. Metodología

1 Seleccionar los usuarios

2 Preparación de cuestionario a utilizar

3 Desarrollar las tareas a ejecutar durante la prueba.

4 Prueba piloto

5 Inicio del Test definitivo a los participantes.

6 Análisis de los datos.

Seleccionar los usuarios: los participantes de la prueba fueron investigadores con sus hojas de vida registradas en el CvLac.

Preparación de cuestionario a utilizar: la metodología que se utilizó en este test se centró en realizar una capacitación a los investigadores acerca de cómo navegar en el software. Se le entrego un cuestionario para desarrollar después de finalizar las tareas.

Tareas a ejecutar:

- Ingrese su URL registrada en CvLAC
- Ingrese su tipo de usuario
- Navegue por el sitio web
- Busque la información sugerida en estudios y utilícela para llenar los campos
- Genere un pdf de su hoja de vida después de ingresarla.

Análisis y resultados de datos

Tabla 4. Resultados usuarios vs tareas

usuario	tareas	identidad	contenido	navegación	búsqueda	Utilidad
Investigador1	A	Mp	Mp	Mp	Mp	Mp
Investigador2	A	Mp	Mp	Mp	Mp	Mp
Investigador3	A	p	p	Mp	Mp	P

Percepción cualitativa de los comentarios:

Tabla 5. Percepción cualitativa de los comentarios

Percepción cualitativa del comentario	Descripción
A	Aprobado
R	Reprobado
Muy positiva (Mp)	El usuario en todas las preguntas dio un parte excelente del sitio
Positiva (P)	El usuario en todas las preguntas dio un parte buena del sitio
Normal (NOR)	El usuario en la mayoría dio un parte bueno del sitio.
Negativa (N)	El usuario en la mayoría dio un parte desfavorable del sitio

Se puede concluir que para los usuarios cuestionados se cumple positivamente con las medidas planteadas, así mismo se comprobó que no hay ningún inconveniente con las tareas asignadas.

10. CONCLUSIONES

Se identificó la importación de la extracción de datos de los sitios web CvLac y ResearchGate para la unificación de la información, eficiencia y rapidez del sistema.

El uso de las herramientas y patrones seleccionados agilizaron el proceso de desarrollo del sistema, mostrándonos así su flexibilidad y facilidad de codificación.

Con el desarrollo de este sistema se ve un sistema de hojas de vida mejorado, más rápido y eficiente. El usuario puede utilizar esta información extraída para agilizar el proceso de llenar o editar su hoja de vida.

La metodología de desarrollo RUP fue la adecuada para este proyecto ya que es un proceso iterativo se facilita hacer pruebas y cambios pudiendo asi garantizar la calidad y eficiencia del software.

El sistema de información es una solución que llena las expectativas de la escuela de ingeniería de sistemas de la Universidad Industrial de Santander ya que ayuda en los procesos de autoevaluación y está acorde con las necesidades de la comunidad.

11. RECOMENDACIONES

Se recomienda en otras versiones del software poder unificar la información extraída del ResearchGate y del CvLac con el fin de no especificar de qué sitio web extraer y poder utilizar la información de estos dos sitios web.

Se recomienda extraer información de más sitios web con el fin de acceder a más información ya que todos los investigadores no tienen sus hojas de vida en los mismos sitios.

BIBLIOGRAFÍA

- Arquitectura JAVA, EJB, Dirección URL: http://www.arquitecturajava.com/introduccion-a-ejb-3-1-i/ (consulta 02-02-14)
- Arquitectura Scrapy, Dirección URL: http://doc.scrapy.org/en/latest/topics/architecture.html; (consulta 06-06-14)
- Colciencias, glosario, Dirección URL: http://www.colciencias.gov.co/glosario (consulta 02-02-14)
- Escuela de ingeniería de sistemas, Dirección URL: http://cormoran.uis.edu.co/eisi/eisi.jsp?IdServicio=S86; (consulta 06-06-14)
- Front Controller, Dirección URL: http://programacion.net/articulo/catalogo_de_patrones_de_diseno_j2ee_i_-_capa_de_presentacion_240/4 (consulta 10-07'14)
- Introducción a la tecnología EJB, Dirección URL: http://www.jtech.ua.es/j2ee/2003-2004/abierto-j2ee-2003-2004/ejb/sesion01-apuntes.htm (consulta 02-02-14)
- Ireport Designer, Dirección URL: http://community.jaspersoft.com/project/ireport-designer, (consulta 10-07-14)
- Javascript Infovis toolkit, Dirección URL: http://philogb.github.io/jit/,_(consulta 02-07-14)
- Lenguaje Python, Estructura interprete Python, Dirección URL: http://www.greenteapress.com/thinkpython/html/thinkpython002.html#toc4; (consulta 10-06-14)
- Metodología RUP, Dirección URL: http://mtdologiarup.blogspot.com/ (consulta 02-10-13)

- Metodología RUP, Dirección URL: http://www.usmp.edu.pe/publicaciones/boletin/fia/info49/articulos/RUP%20vs.%20XP.pdf (consulta 02-10-13)
- MongoDB, Dirección URL: http://www.mongodb.org/, (consulta 10-10-13)
- Patrón FACADE Dirección URL: http://patronestructuralesyalgomas.blogspot.com/2009/05/patron-facade.html (consulta 10-07-14)
- Pruebas de la caja negra, Dirección URL: http://www.globetesting.com/2012/08/pruebas-de-caja-negra/ (consulta 20-07-14)
- Scrapy, Dirección URL: http://scrapy.org; (consulta 02-02-14)
- Universidad Industrial Santander, Dirección URL: http://www.uis.edu.co/webUIS/es/acercaUis/index.html; (consulta 06-06-14)

ANEXOS

ANEXO A. FORMATO ENTREVISTA PRUEBA DE USABILIDAD

Entrevistadores:

Karime chaparro

Katherine Salazar

Preguntas:

- 1 a 2 identidad
- 3 a 5 contenido
- 6 a 8–_navegación
- 9 a 10 *búsqueda*
- 11 a 14 utilidad

1. ¿Con la información que se ofrece en la pantalla, es posible saber a qué organización corresponde? ¿Cómo lo sabe?
2. ¿Hay algún elemento grafico o color que le haya ayudado a entender a qué organización pertenece el sitio?
3. ¿Hacia qué tipo de audiencia cree usted que está dirigido este sitio? ¿Porque?
4. ¿Le parece adecuada la selección de contenidos en las secciones de la hoja de vida o usted echo de menos otras áreas de información que le habría gustado ver en la hoja de vida?
5. ¿Los textos usados son suficientemente descriptivos de lo que se ofrece en la página?
6. ¿Existen elementos dentro de la página que le permitan saber exactamente donde se encuentra usted dentro de este sitio.
7. ¿Cómo vuelve a editar alguna sección que ya paso? ¿Fue fácil?
8. ¿Se ha sentido perdido dentro del sitio? ¿Si lo ha sentido en que área fue? ¿Si no lo ha sentido, que elemento lo ayudo a orientarse?
9. ¿Es fácil buscar la información de su hoja de vida?
10. ¿Es fácil buscar donde está la información en cada sección?
11. ¿Al mandar datos mediante un formulario el web le avisa si los recibió correctamente o no?
12. ¿Tras una primera mirada, le queda claro el objetivo del sitio?
13. ¿Cree que los contenidos y servicios que se ofrecen en este sitio son de utilidad?
14. ¿Qué es lo que más te llamo la atención positivamente o negativamente de la utilidad que ofrece el sitio web?