

UTEC

Universidad Tecnológica

**PROYECTO FIN DE CARRERA
TECNÓLOGO INFORMÁTICO**

AUTORES

Carlos Balbiani
Joaquin Suárez
Julio Arrieta

TUTOR

Nicolas Escobar

CONTENIDO

RESUMEN	3
1. INTRODUCCIÓN	4
2. OBJETIVOS PLANTEADOS Y RESULTADOS ESPERADOS	4
3. CONSIDERACIONES GENERALES	5
3.1. MARCO TEÓRICO	5
3.1.1. EL PRODUCTO O SERVICIO	5
3.1.2. EL MERCADO	5
3.1.3. LOS CLIENTES	5
3.1.4. LA COMPETENCIA	6
3.2. MARCO LEGAL	6
3.3. TECNOLOGÍAS	6
3.3.3. ARQUITECTURAS	7
3.3.3.1. MODELO CLIENTE – SERVIDOR	7
3.3.3.3. CAPA DE PRESENTACIÓN	7
3.3.3.4. CAPA DE NEGOCIO	7
3.3.3.5. CAPA DE DATOS	8
3.3.4. PYTHON	8
3.4. HERRAMIENTAS RELACIONADAS	8
3.4.1. URUGUAY CONCURSA (NACIONAL)	9
3.4.1.1. DISEÑO WEB	10
3.4.1.2. LA SOLUCIÓN PLANTEADA	10
3.4.1.3. FORTALEZAS	10
3.4.1.4. DEBILIDADES	10
3.4.1.5. APLICACIÓN MOBILE	10
3.4.2. BUSCO JOB (INTERNACIONAL)	10
3.4.2.1. DISEÑO WEB	11
3.4.2.2. LA SOLUCIÓN PLANTEADA	11
3.4.2.3. FORTALEZA	11
3.4.2.4. DEBILIDADES	12
3.4.2.5. APLICACIÓN MOBILE	12
3.4.3. TWAGO (INTERNACIONAL)	12
3.4.3.1. DISEÑO WEB	13
3.4.3.2. LA SOLUCIÓN PLANTEADA	13
3.4.3.3. FORTALEZAS	13
3.4.3.4. DEBILIDADES	13
3.4.3.5. APLICACIÓN MOBILE	13
3.5. CONCLUSIONES	13
4. ANÁLISIS DEL PROBLEMA	14
4.1. REQUERIMIENTOS DEL SISTEMA	14
4.2. VISTA DEL MODELO DE CASOS DE USO	14

4.3. CASOS DE USO RELEVANTES A LA ARQUITECTURA	15
4.3.1. BUSCAR OFERTA	15
4.3.2. VER ESTADÍSTICA	15
4.3.3. CALCULADORA	15
4.3.4. POSTULARSE A UN TRABAJO	15
4.3.5. ASIGNAR FECHA DE ENTREVISTA	15
4.3.6. CALIFICAR ENTREVISTA	15
4.3.7. ADMINISTRAR ARAÑAS	15
4.4. DOMINIO DEL PROBLEMA	16
4.5. ARQUITECTURA DEL SISTEMA	17
4.5.1. TRAZABILIDAD DESDE EL MODELO DE CASOS DE USO AL MODELO DE DISEÑO	17
4.6. VISTA DEL MODELO DE DISEÑO	18
4.7. DESCOMPOSICIÓN EN SUBSISTEMAS	18
4.7.1. SCRAPING	18
4.7.2. PERSISTENCIA	18
4.7.3. SPLASH	18
VISTA DEL MODELO DE DISTRIBUCIÓN	19
5. IMPLEMENTACIÓN	20
5.1. METODOLOGÍAS DE TRABAJO	20
5.2. ENTORNO DE DESARROLLO	21
5.3. ENTORNO DE EJECUCIÓN	21
5.4. TECNOLOGÍAS APLICADAS	21
5.4.1. CAPA DE PERSISTENCIA	22
5.5. PROBLEMAS ENCONTRADOS	22
5.6. HORAS DEDICADAS	23
5.7. LA HERRAMIENTA DESARROLLADA	23
6. CONCLUSIONES	24
6.1. TRABAJOS A FUTURO	28
7. REFERENCIAS	29

RESUMEN

En el marco de la asignatura Proyecto final se presentará a continuación un resumen para mayor entendimiento del mismo y a lo largo del documento se darán más detalles al respecto, Uruguay se caracteriza por ser un país en constante crecimiento por tanto encontramos muchas ofertas laborales, pero a nuestro criterio hace falta una herramienta accesible a todo el mundo, que permita buscar ofertas laborales de manera más eficiente.

Se propone analizar la manera de realizar un sistema de gestión para una empresa de búsqueda de trabajo, con el fin de llevar a cabo una gestión eficiente y a su vez captar clientes de forma rápida, es decir, antes de que puedan ser captados por la competencia. El objetivo planteado es estudiar y aplicar en la práctica las tecnologías Django y Scrapy para el desarrollo de una plataforma web.

Para resolver esta problemática, se planteó un sistema informático que permitirá acceder desde una plataforma web a las diferentes propuestas laborales, esto con un fuerte enfoque hacia los clientes, quienes podrán realizar sus postulaciones, completando sus curriculum, como los administradores quienes tendrán la posibilidad de dar sus opiniones sobre los mismos entre muchas otras elecciones.

1. INTRODUCCIÓN

Python¹ es un lenguaje de programación de código abierto, orientado a objetos, muy simple y fácil de entender. Tiene una sintaxis sencilla que cuenta con una vasta biblioteca de herramientas, que hacen de Python un lenguaje de programación único.

En la actualidad, Python goza de una gran popularidad en diferentes áreas como Ciencia e Ingeniería, Análisis de datos y Machine Learning², Desarrollo web, Administración de sistemas y DevOps³.

Todas estas características hacen de Python el lenguaje ideal para seleccionarlo en el proyecto de final de carrera, en donde los estudiantes ya cuentan con los conocimientos necesarios para explorar y utilizar nuevas tecnologías por sí solos.

2. OBJETIVOS PLANTEADOS Y RESULTADOS ESPERADOS

El objetivo del proyecto es estudiar diferentes tecnologías basadas en Python que puedan ser utilizadas para hacer Web Scraping⁴. El término Web scraping o raspado web, es una técnica utilizada mediante programas de software para extraer información de sitios web.

Lo que conseguimos con el Web Scraping es extraer información de diferentes sitios web y almacenarla de forma estructurada para poder aprovecharla.

Como punto de partida se deberá hacer un estudio del estado del arte de las principales tecnologías Python relacionadas con el Web Scraping y la extracción de información.

En base a este estudio, se pide desarrollar una Plataforma Online que simula un Buscador de empleos a nivel regional e internacional, donde los usuarios puedan tanto publicar ofertas de empleo como buscar las mismas.

Los servicios a ofrecer deberán ser "scrapeados" de la web (en caso de no disponer de un sitio legal para eso, se deberá simular uno de ellos para poder abastecer la información).

En cuanto a los requerimientos funcionales la plataforma permitirá la gestión de los perfiles de usuarios, gestión de base de datos de los curriculums de solicitantes, desarrollar algoritmos de búsqueda para ofrecer a los solicitantes las mejores opciones.

En cuanto a los requerimientos no funcionales, se espera que la aplicación sea distribuida, con un backend desarrollado en Python, y front end de libre elección al igual que la base de datos que puede ser relacional o no relacional.

3. CONSIDERACIONES GENERALES

Esta sección, formaliza e intenta expresar de forma clara la investigación realizada sobre las aplicaciones web, así como tecnologías aplicadas para cada desarrollo en lo que refiere a para administración y uso de la plataforma para buscar trabajo. Puede ver más acerca del estado del arte en el anexo 1.

3.1. MARCO TEÓRICO

La necesidad de entender el funcionamiento de la plataforma para la búsqueda de empleos, requiere conocer en primera instancia el producto o servicio que ofrecen, el mercado al cual se enfrentan, el tipo de cliente al que va dirigido dicho servicio, la competencia y las normativas legales a las cuales se tiene que ajustar.

3.1.1. EL PRODUCTO O SERVICIO

El servicio brindado por la plataforma es una clara opción para aquellos que buscan trabajos o por también quienes estén interesados en informarse de las ofertas y salarios que rondan, esta opción se diferencia del resto por su practicidad donde se podrá encontrar ofertas de diferentes sitios, la plataforma estará en constante actualización con las más recientes novedades de los sitios scrapeados.

3.1.2. EL MERCADO

El mercado en el que se sitúan las empresas de búsqueda de trabajo no está muy regulado y no cuenta con exigencias muy específicas a la hora de montar el negocio, muchas de las empresas que se encuentran en este mercado, han tenido cambios sustanciales a lo largo de los años, por lo cual para poder explotar nuevas posibilidades se debe estar en constante cambio.

3.1.3. LOS CLIENTES

Dentro de los posibles tipos de cliente se pueden destacar los siguientes:

Personas

Desempleadas en búsqueda de empleo o personas curiosas acerca de las novedades y salarios que cambian diariamente.

Empresas

Las empresas muchas veces tienen la necesidad de ofrecer nuevos puestos de trabajos muchas veces se deciden por contratar este tipo de servicios dado su practicidad, alcance y efectividad.

Postulantes

Esta última categorización refiere a aquellos clientes que desean postularse a las ofertas de las empresas que se anuncian en nuestro sistema, los cuales utilizan este servicios para hallar trabajo, un claro ejemplo es una persona que desea buscar trabajo para aumentar sus ingresos.

3.1.4. LA COMPETENCIA

Es conveniente analizar y conocer algunas agencias que operan a nivel nacional e internacional, tanto de sus formas de operar, sus fuentes de información, las facilidades dadas tanto a usuarios interesados en postularse como aquellos administradores buscando la ligereza de calificar las entrevistas previamente realizadas.

3.2. MARCO LEGAL

El data scraping o raspado de datos es una práctica que sigue levantando cierto revuelo, ya que desde algunos sectores se la considera poco ética. Al final, en muchos casos se utiliza para obtener datos de otras páginas webs para replicarlos en una nueva mediante el uso de una API, lo cual en algún caso podría dar lugar a copia o duplicidad de información.

Asimismo, estos bots pueden ser diseñados para navegar de forma automática por una web, incluso crear cuentas falsas, de ahí que en muchas webs veas el típico captcha para confirmar que no eres un bot.

Por otro lado, la extracción automática de información puede crearle problemas a las páginas web analizadas, sobre todo si el rastreo se realiza de forma recurrente, por tanto una web que está siendo continuamente visitada por los crawlers, podría verse afectada y perjudicada por estas visitas de “baja calidad” y perder así posicionamiento. Más información sobre la ley⁵ estarán disponibles en las referencias.

3.3. TECNOLOGÍAS

Para poder comprender el problema desde el punto de vista técnico, es importante introducir algunos conceptos clave. Estos se detallan a continuación.

3.3.1. SCRAPY

Web scraping o raspado web, es una técnica utilizada mediante programas de software para extraer información de sitios web, para esto utilizaremos la herramienta Scrapy⁶ la cual es un framework colaborativo de código libre que corre en Python orientado al Web Scraping, utilizado en sitios web para rastrear y extraer datos estructurados de sus páginas.

Se puede utilizar para una amplia gama de propósitos, desde Minería de Datos⁷ hasta monitoreo y pruebas automatizadas. Además es la herramienta más ampliamente utilizada para este propósito. Scrapy fue desarrollado inicialmente en una compañía de *e-commerce* londinense llamada Mydeco y a continuación fue desarrollada y mantenida por empleados de Mydeco e Insophia (una consultoría web basada en Montevideo, Uruguay).

3.3.2. DJANGO

Django⁸ es un framework de aplicaciones web gratuito y de código abierto (Open Source⁹) escrito en Python. La meta fundamental de Django es facilitar la creación de sitios web complejos. Django pone énfasis en el reuso, la conectividad y extensibilidad de componentes, el desarrollo rápido y el principio No te repitas (DRY, del inglés Don't Repeat Yourself), Python es usado en todas las partes del framework, incluso en configuraciones, archivos y en los modelos de datos.

3.3.3. ARQUITECTURAS

El concepto de arquitectura de software representa estructuralmente un diseño de alto nivel del sistema y tiene como objetivo ayudar a satisfacer los atributos de calidad y a su vez servir como guía para el desarrollo. Es por este motivo que idealmente se crea en etapas tempranas del desarrollo. A continuación, se presentan los modelos de arquitectura más conocidos.

3.3.3.1. MODELO CLIENTE – SERVIDOR

En las aplicaciones web, generalmente el usuario final, o en este caso cliente, inicia una comunicación cuando desea acceder o interactuar con el servicio. El cliente en este caso entra en contacto con el servidor, quien le brinda la información deseada, o eventualmente reacciona ante la petición del cliente.

3.3.3.2. ARQUITECTURA EN CAPAS

La arquitectura en capas propone separar la lógica del negocio del diseño de la interfaz de usuario. De esta forma se facilita el mantenimiento del sistema, y el desarrollo inicial del mismo, permitiendo abstraerse de la solución y concentrarse en una capa puntual.

3.3.3.3. CAPA DE PRESENTACIÓN

Es la cara visible de la aplicación, la que permite al usuario hacerse de la información solicitada, o interactuar con el servicio ofrecido. En ella el usuario ingresa datos que posteriormente son enviados a la siguiente capa, a la vez que recibe los resultados obtenidos de la misma, la capa de negocio.

3.3.3.4. CAPA DE NEGOCIO

Cada negocio o escenario plantea y define sus propias reglas, a las cuales se les denomina reglas de negocio. Estas reglas son el núcleo de la aplicación o sistema y son las que definen las funcionalidades que se ofrecen en el mismo. De ahí el nombre de esta capa. Para sustentar estas reglas, es aquí donde el sistema ejecuta, procesa y resuelve las peticiones de cada cliente, devolviendo datos en caso de ser necesario, para poder realizar estas acciones, es necesario interactuar con la siguiente capa de la arquitectura, la capa de datos.

3.3.3.5. CAPA DE DATOS

La capa de acceso a datos, o simplemente capa de datos, es la encargada de procesar las peticiones de datos, y devolverlas a la capa anterior, así como actualizar o agregar información al juego de datos. Por este motivo es la encargada de comunicarse con los datos mismos, los que se suelen persistir, ya sea en sistemas de archivos, base de datos, o algún otro modo de almacenamiento.

3.3.4. PYTHON

Python es un lenguaje de programación interpretado cuya filosofía hace hincapié en la legibilidad de su código. Se trata de un lenguaje de programación multiparadigma, ya que soporta orientación a objetos, programación imperativa y, en menor medida, programación funcional. Es un lenguaje interpretado, dinámico y multiplataforma, es un lenguaje de programación multiparadigma, esto significa que más que forzar a los programadores a adoptar un estilo particular de programación, permite varios estilos, por ejemplo, programación orientada a objetos, programación imperativa y programación funcional.

3.3.5. SPLASH

Splash¹⁰ es un servicio de renderizado de JavaScript¹¹, este es un navegador web ligero con una API HTTP¹², implementado en Python 3 usando Twisted¹³.

3.3.6. DOCKER

Docker¹⁴ es un proyecto de código abierto que automatiza el despliegue de aplicaciones dentro de contenedores de software, proporcionando una capa adicional de abstracción y automatización de virtualización de aplicaciones en múltiples sistemas operativos.

Docker utiliza características de aislamiento de recursos del Kernel Linux¹⁵, tales como cgroups y espacios de nombres (namespaces) para permitir que "contenedores" independientes se ejecuten dentro de una sola instancia de Linux, evitando la sobrecarga de iniciar y mantener máquinas virtuales.

3.4. HERRAMIENTAS RELACIONADAS

Además de realizar un análisis de la realidad del objeto de estudio, y de entender conceptos útiles para llevar a cabo una solución al problema planteado, es necesario conocer y analizar otras soluciones pertenecientes al mismo rubro, tanto a nivel nacional como internacional, con el objetivo de detectar fortalezas y debilidades en cada una.

3.4.1. URUGUAY CONCURSA (NACIONAL)

Uruguay Concurso es una dependencia del estado.

Tiene como objetivo principal satisfacer la demanda de empleados para el sector público, ofreciendo atención remota desde su página web.



Imagen 1 - Página de inicio de Uruguay Concurso.



Imagen 2 - Página de oferta laboral.

3.4.1.1. DISEÑO WEB

La Imagen 1 muestra la página principal de Uruguay concursa donde se puede ver información de contacto. un resumen de los llamados que posee la página e información de interés.

En la imagen 2 se muestra el detalle de uno de los llamados.

3.4.1.2. LA SOLUCIÓN PLANTEADA

A nivel nacional Uruguay Concurza ofrece los servicios requeridos en el rubro de búsqueda de trabajo. Sin embargo, destacamos que carece de servicios en línea para que sus clientes puedan realizar los trámites en forma remota. Si bien su sitio web está planteado a nivel informativo, carece de información básica como, por ejemplo, cantidad de horas, rango de salarios, empresa que solicita, consultas en línea, entre otros. Además de Uruguay Concurza se evaluaron otras compañías internacionales como, por ejemplo, BuscoJob, Twago, entre otras.

3.4.1.3. FORTALEZAS

- Consta con un gran número de ofertas.
- Diseño simple.
- Variedad y categorización de ofertas laborales.
- Atención telefónica.
- Información de otros servicios.
- Página web responsive.

3.4.1.4. DEBILIDADES

- No cuenta con funcionalidad de búsqueda de trabajo privado (crítico).
- No cuenta con funcionalidad de búsqueda de trabajo internacional (alto).
- No cuenta con información de posibles salarios por horas, o por mes (medio).
- Poca información de la empresa que solicita (alto).

3.4.1.5. APLICACIÓN MOBILE

No posee aplicación Mobile.

3.4.2. BUSCO JOB (INTERNACIONAL)

BuscoJob es una empresa internacional.

Tiene como objetivo principal satisfacer la demanda de empleados para el sector público y privado en el ámbito internacional, ofreciendo atención remota desde su página web.



Imagen 3 - Página de inicio buscoJob.



Imagen 4 - Página de oferta laboral.

3.4.2.1. DISEÑO WEB

La Imagen I muestra la página principal de BuscoJob donde se puede ver información de contacto. un resumen de los llamados que posee la página e información de interés.

En la imagen II se muestra el detalle de uno de los llamados.

3.4.2.2. LA SOLUCIÓN PLANTEADA

A nivel internacional BuscoJob ofrece los servicios requeridos en el rubro de búsqueda de trabajo. Sin embargo, destacamos que carece de servicios en línea para que sus clientes puedan realizar los trámites en forma remota.

3.4.2.3. FORTALEZA

- Consta con un gran número de ofertas.
- Diseño simple.
- Variedad y categorización de ofertas laborales.
- Atención telefónica.
- Información de otros servicios.
- Página web responsive.

3.4.2.4. DEBILIDADES

- No cuenta con información de posibles salarios por hr, o por mes (medio)
- Poca información de la empresa que solicita (alto).

3.4.2.5. APLICACIÓN MOBILE

No posee aplicación Mobile.

3.4.3. TWAGO (INTERNACIONAL)

Twago es una empresa internacional.

Tiene como objetivo principal satisfacer la demanda de empleados para el sector público y privado en el ámbito internacional, ofreciendo atención remota desde su página web.



Imagen 5 - Página de inicio de twago.

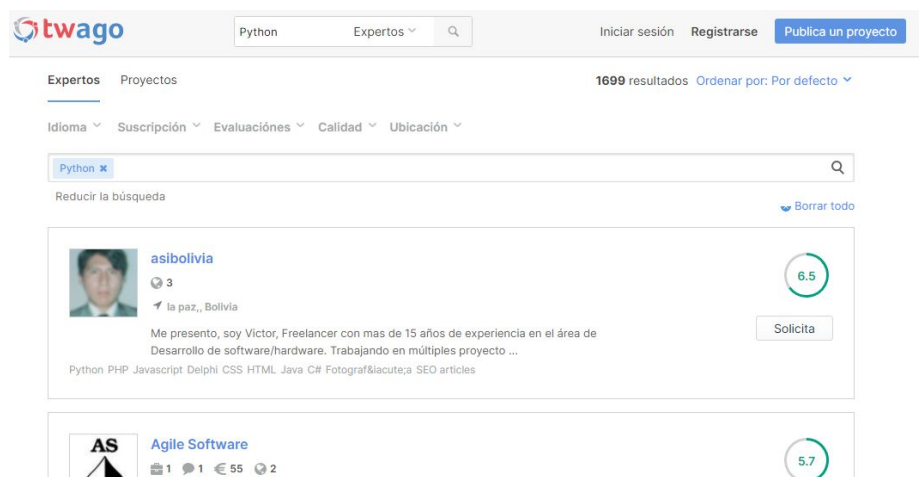


Imagen 6 - Página de oferta laboral.

3.4.3.1. DISEÑO WEB

La Imagen 1 muestra la página principal de Twago donde se puede ver información de contacto, un resumen de los llamados que posee la página e información de interés.

En la imagen 2 se muestra el detalle de uno de los llamados.

3.4.3.2. LA SOLUCIÓN PLANTEADA

A nivel internacional Twago ofrece los servicios requeridos en el rubro de búsqueda de trabajo, sin embargo, destacamos que carece de servicios en línea para que sus clientes puedan realizar los trámites en forma remota.

3.4.3.3. FORTALEZAS

- Consta con un gran número de ofertas.
- Diseño simple.
- Variedad y categorización de ofertas laborales.
- Atención telefónica.
- Información de otros servicios.
- Página web responsive.

3.4.3.4. DEBILIDADES

- No cuenta con información de posibles salarios por hr, o por mes (medio)
- Poca información de la empresa que solicita (alto).

3.4.3.5. APLICACIÓN MOBILE

No posee aplicación Mobile.

4. ANÁLISIS DEL PROBLEMA

Se presenta la necesidad de tener una plataforma para gestionar ofertas de trabajos, scrapeo de las mismas, postulantes, curriculums y perfiles de los mismos, postulaciones.

Por lo cual se nos ha encomendado la creación de una aplicación, dotada de un eficaz motor de búsquedas, con consulta de disponibilidad de ofertas laborales en base a los criterios de necesidad aplicados, facilitando el proceso de postulación, búsqueda y evaluación de posibles empleos, para sus usuarios. Esta aplicación cuenta con una sección para empleados y clientes de la compañía, que permite visualizar las ofertas laborales con la que cuenta la empresa, las postulaciones a las mismas, creación de currículum, un módulo de scrapeo para obtener información de otros sitios, un sector de administración de base de datos y arañas.

4.1. REQUERIMIENTOS DEL SISTEMA

Esta sección tiene el propósito de oficiar como medio de comunicación entre el cliente y los desarrolladores, sirviendo como base contractual en la que se especificarán los requerimientos funcionales y no funcionales a implementar.

El objetivo es disponer de un sistema que permita gestionar y administrar los diferentes aspectos que componen la operación de una empresa de búsqueda de trabajo. Dentro de la aplicación se cuenta con un módulo web que permite a los usuarios con los privilegios necesarios administrar y gestionar los elementos de la aplicación (Ofertas, postulantes, curriculums, calificaciones, entrevistas). Por otra parte, a los usuarios de tipo se les permite realizar búsquedas en base a palabras clave, país, categoría y subcategoría, postularse, ver estadísticas, subir curriculum, el uso de la calculadora de posibles salarios. Puede ver más acerca de este tema en el anexo 2.

4.2. VISTA DEL MODELO DE CASOS DE USO

A continuación, se muestra el diagrama de todos los casos de uso, que representa la relación entre los actores y las funcionalidades del sistema. Luego se brindan detalles sobre los casos de uso relevantes a la arquitectura.

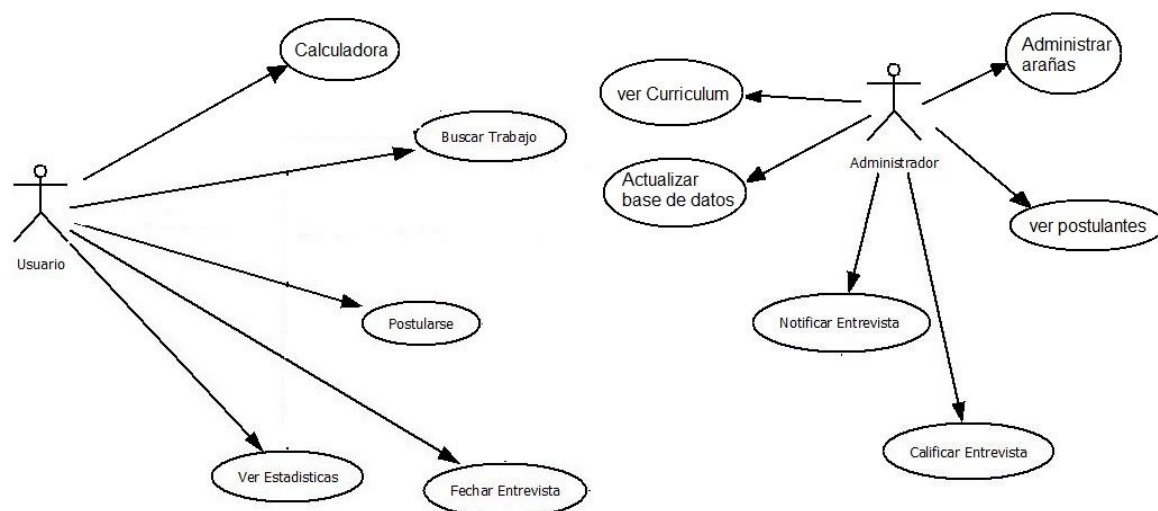


Imagen 7 - Diagrama de casos de uso.

4.3. CASOS DE USO RELEVANTES A LA ARQUITECTURA

Los casos de uso que se expondrán en esta sección se consideran relevantes dado que afectan a la mayor cantidad de capas de la aplicación. Estos casos fueron los primeros en ser modelados e implementados.

4.3.1. BUSCAR OFERTA

El usuario e invitado están habilitados para buscar ofertas laborales, Ingresa algún identificador de una oferta laboral que busca con la posibilidad de agregar filtros.

4.3.2. VER ESTADÍSTICA

Un usuario o invitado podrá seleccionar entre diferentes tipos de estadísticas basadas en las ofertas laborales, estas se visualizarán tanto en forma de tabla o gráfico.

4.3.3. CALCULADORA

Los usuarios podrán seleccionar entre un basta cantidad de categorías, las cuales a través de datos extraídos de diferentes páginas calcular los máximos y mínimos de los posibles salarios que podría cobrar según esas categorías de habilidades.

4.3.4. POSTULARSE A UN TRABAJO

Un usuario registrado podrá postularse a una oferta de trabajo que le interese.

4.3.5. ASIGNAR FECHA DE ENTREVISTA

El administrador establecerá determinadas fechas para realizar la entrevista.

4.3.6. CALIFICAR ENTREVISTA

El Administrador tendrá la posibilidad de calificar una entrevista, la cual se le otorgara cierto puntaje.

4.3.7. ADMINISTRAR ARAÑAS

El Administrador podrá gestionar las arañas por medio de una página, a través de la cual podrá seleccionar la cantidad de registros que quiere descargar así como de las páginas que desea scrapear.

4.4. DOMINIO DEL PROBLEMA

En esta sección se presenta el modelo de domino, en el que se representan las distintas entidades con sus atributos y relaciones.

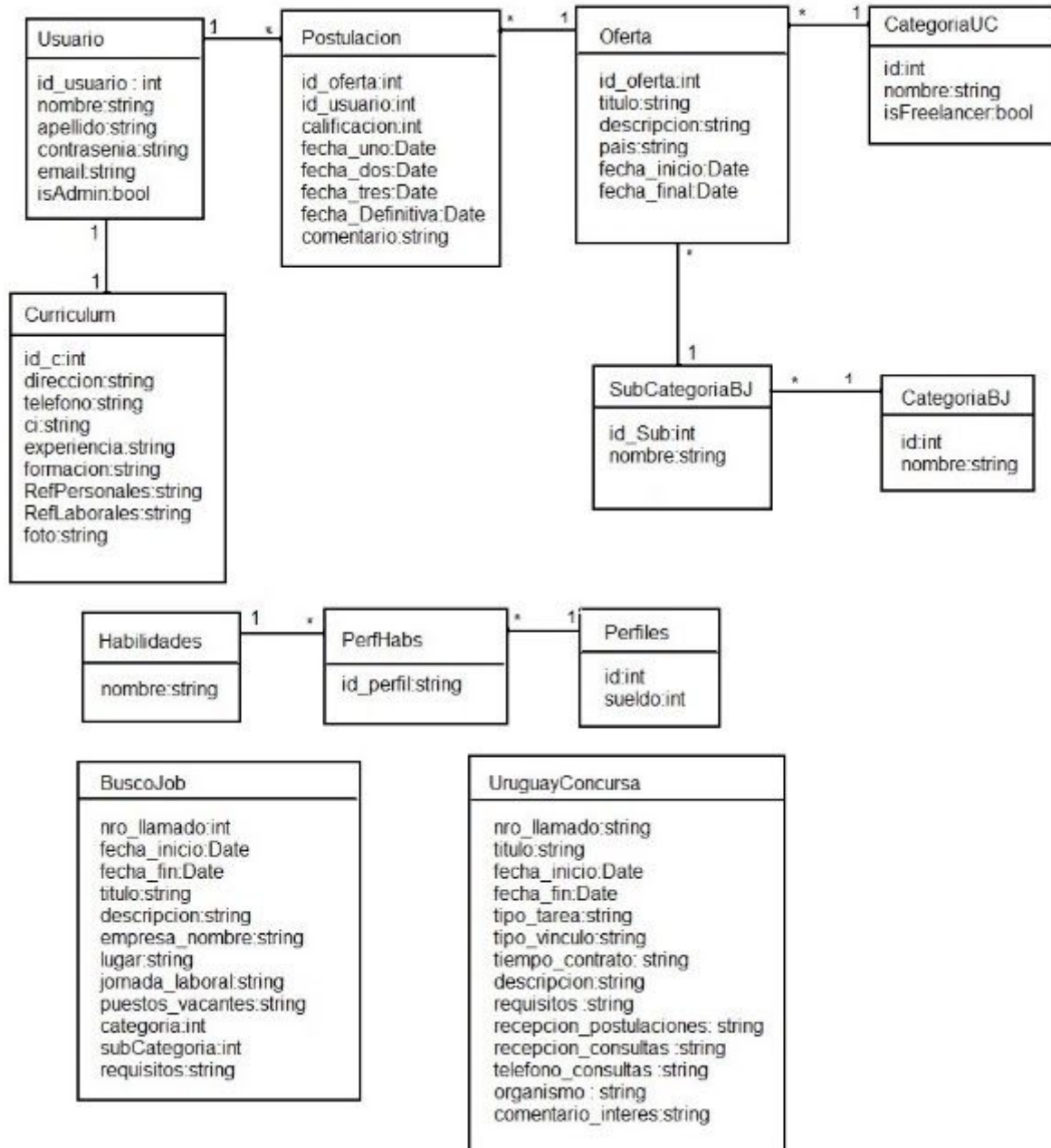


Imagen 8 - Dominio del problema.

4.5. ARQUITECTURA DEL SISTEMA

En esta sección se informa acerca de la arquitectura del sistema, donde se describe la misma utilizando el (Modelo de vistas de arquitectura 4+1)¹⁶ diseñado por Phillippe Kruchten. Este modelo describe cuatro vistas: vista lógica, vista de desarrollo, vista de proceso y vista física. Para poder hacer una trazabilidad a todos los componentes, se unifican las cuatro vistas mencionadas anteriormente a través de la vista de escenarios, la cual se corresponde con instancias de casos de uso. Puede ver más acerca de este tema en el anexo 3.

4.5.1. TRAZABILIDAD DESDE EL MODELO DE CASOS DE USO AL MODELO DE DISEÑO

La Vista Lógica se utiliza para marcar la trazabilidad entre el Modelo de Casos de Uso y el Modelo de Diseño, identificando los objetos y subsistemas de diseño que intervienen en el caso de uso y sus relaciones. Para este fin se utiliza el Diagrama de Paquetes:

Para este fin se utiliza el Diagrama de Paquetes:

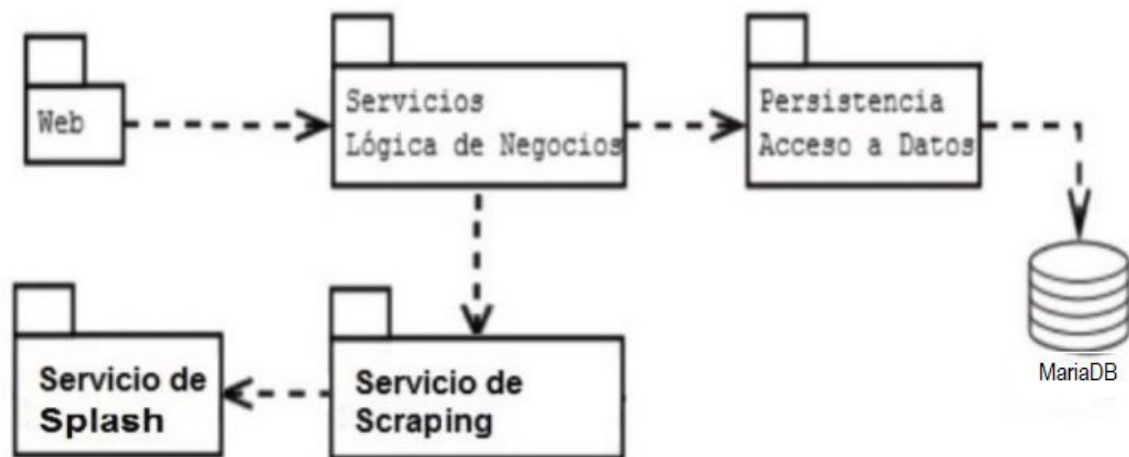


Imagen 9 - Diagrama de Paquetes.

4.6. VISTA DEL MODELO DE DISEÑO

El modelo de diseño de software es realmente un proceso de muchos pasos pero que se clasifican dentro de uno mismo. En general, la actividad del diseño se refiere al establecimiento de las estructuras de datos, la arquitectura general del software, representaciones de interfaz y algoritmos. Para más información sobre consultar el Anexo V (Modelo de Diseño).

4.7. DESCOMPOSICIÓN EN SUBSISTEMAS

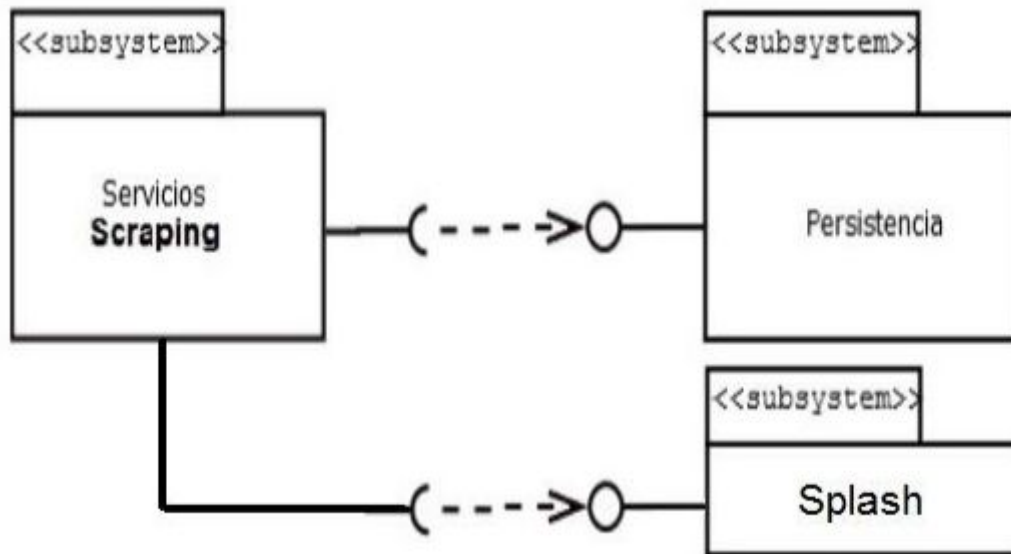


Imagen 10 - Descomposición en subsistemas..

4.7.1. SCRAPING

Este subsistema provee las funcionalidades de scraping, trayendo los recursos que serán persistidos por otros subsistemas.

4.7.2. PERSISTENCIA

Este subsistema se encarga del acceso a datos persistentes necesarios para que la aplicación mantenga su información en el tiempo. Ofrece interfaces al subsistema de Servicios (los datos son consumidos por la Lógica de Negocio).

4.7.3. SPLASH

Este subsistema permite obtener todo el código HTML de una sitio web, que normalmente no es accesible debido al renderizado por javascript.

4.8. TRAZABILIDAD DESDE EL MODELO DE DISEÑO AL MODELO DE IMPLEMENTACIÓN

VISTA DEL MODELO DE DISTRIBUCIÓN

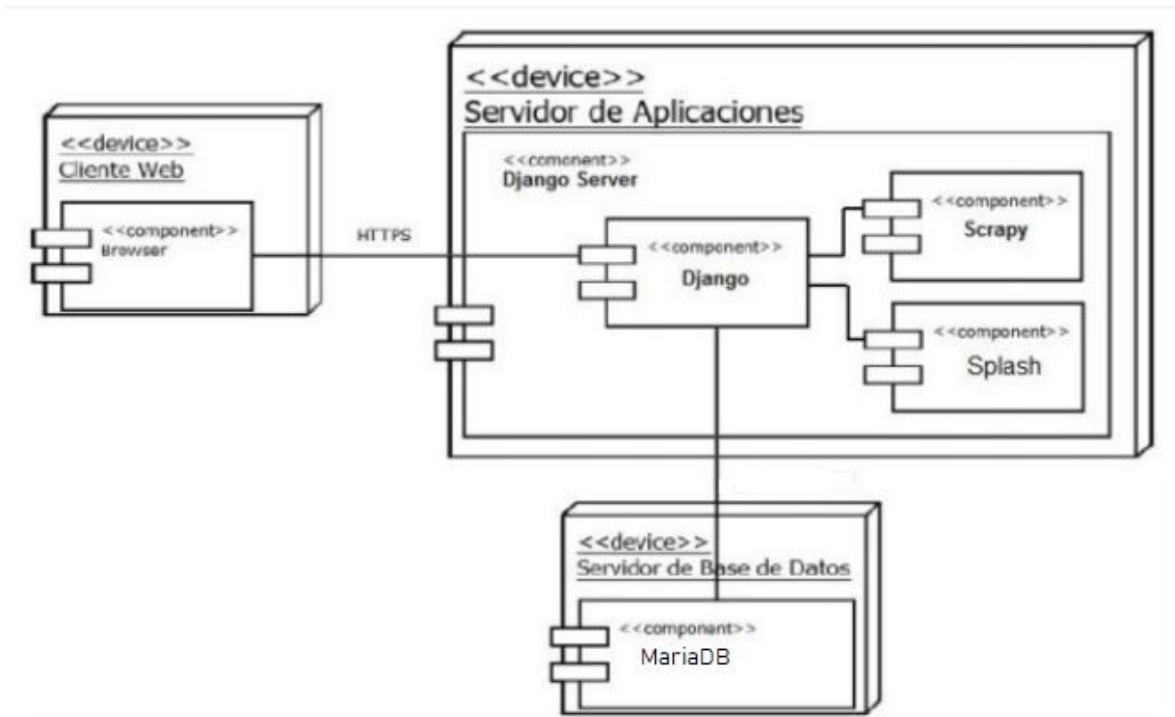


Imagen 11 - Vista del modelo de distribución.

5. IMPLEMENTACIÓN

En esta sección se brindan detalles sobre la metodología de trabajo utilizada, los entornos utilizados (desarrollo y ejecución), tecnologías aplicadas y problemas encontrados durante el transcurso del proyecto. También se detalla el proceso de implementación de las distintas funcionalidades de la aplicación, aplicando lo estipulado en las etapas de análisis y diseño.

5.1. METODOLOGÍAS DE TRABAJO

Para el desarrollo del proyecto se definieron roles desde el inicio del mismo en base a los conocimientos teórico-prácticos previos de los integrantes del equipo. Una vez definidos los roles, se estableció una planificación de trabajo, donde se definieron y asignaron las tareas (“Estado del arte y pruebas de concepto”, “Análisis, diseño e implementación de casos de uso críticos”, “Implementación”, “Testing y documentación final”) y se realizó una estimación de esfuerzo por tarea. Luego se pautaron reuniones semanales con los miembros del equipo para evaluar los avances obtenidos, comparar los tiempos reales con los estimados y de ser necesario modificar el cronograma inicial. También se definieron reuniones semanales con el tutor del equipo para realizar relevamientos de los avances obtenidos.

Los roles definidos fueron los siguientes:

Project Manager:

Quién sería el encargado de gestionar la planificación y el desarrollo del proyecto y velar por el cumplimiento de las tareas definidas. También sería quien se encargue de documentar los avances y generar la documentación correspondiente recopilando la información brindada por el resto de los miembros del equipo

Web Developer:

Quién sería el encargado de estudiar y evaluar las tecnologías web vigentes para su posterior aplicación. También sería quien proponga los distintos diseños de las vistas utilizadas en la aplicación web.

Backend Developer:

Quién tuviese este rol, sería el encargado de estudiar las posibles tecnologías aplicables en la capa de persistencia y negocio de la aplicación. Debido a la complejidad de las responsabilidades, se decidió que este rol sea asumido por dos integrantes del equipo.

Cliente:

Si bien este no sería un rol definido en la planificación, fue asumido por la tutora del equipo, quien realizó devoluciones en base a los avances que se presentaban en cada reunión de monitoreo.

Cabe aclarar que dado a la envergadura del proyecto, los roles se fueron rotando y todos los integrantes realizaron acciones correspondientes a los diferentes roles.

5.2. ENTORNO DE DESARROLLO

Se creó en primera instancia un entorno virtual, utilizando python para facilitar la instalación local de las librerías necesarias, configurar el entorno de desarrollo y luego replicarlo por todos los miembros del equipo.

Se instaló y se creó un nuevo proyecto Django, para el desarrollo del MTV (Model Template, View) lo que es similar a un MVC, esto se explica más adelante, desarrollar el backend y el frontend web a este MTV. Se agregó al entorno virtual, la librería Scrapy para poder scrapear, también se agregó Pillow¹⁷ una librería de python para cargar las imágenes al currículum al framework de Django.

Se instaló un contenedor Docker para poder correr un servidor que contiene Splash (programa que funciona como navegador para poder interpretar Javascript, dado que las arañas por sí solas no lo interpretan), y se configuró la araña (programa que obtiene los datos de las páginas), ésta en particular, Twago, dado que las páginas internacionales, tienen más medidas anti-scraping.

Para la base SQL se utilizó MariaDB.

5.3. ENTORNO DE EJECUCIÓN

Para la ejecución de la aplicación se requiere python-3.8.6rc1-amd64, MariaDB para gestionar la base de datos, se debe iniciar el servidor de Django y crear una base de datos "uruguayjob", también cargar los datos de prueba "datosDePrueba.sql" para facilitar las tareas de administración.

5.4. TECNOLOGÍAS APLICADAS

Esta sección brinda detalles de las tecnologías aplicadas para resolver las distintas funcionalidades de la aplicación. Se utilizó una implementación por capas, con una arquitectura cliente – servidor.

El objetivo primordial de este estilo es la separación de la lógica de negocios de la lógica de diseño y su ventaja principal es que el desarrollo se puede llevar a cabo en varios niveles y, en caso de requerir algún cambio, solo se ataca el nivel requerido sin necesidad de afectar otras capas.

La plataforma de programación utilizada en este proyecto fue Django y como servidor de aplicaciones se buscó uno de tipo open source que tuviese la categoría de MTV es decir un MVC. La principal diferencia entre MVC y MTV es que en un MVC, tenemos que escribir todo el código específico del control. Pero en un MTV, la parte del controlador está a cargo del marco en sí. El patrón MVC clásico funciona administrando el estado de una aplicación.

5.4.1. CAPA DE PERSISTENCIA

Para persistir los datos se optó por utilizar exclusivamente una sola base de datos, siendo SQL para la cual se eligió MariaDB.

MariaDB es un sistema de gestión de bases de datos. Se deriva de MySQL, una de las bases de datos más importantes que ha existido en el mercado, utilizada para manejar grandes cantidades de información.

En la base de datos SQL se persisten los datos cuyo volumen de crecimiento es más acotado, concretamente se persisten datos de las ofertas de las diferentes fuentes, los datos de usuarios tales como sus postulaciones y su currículum, así como datos de la parte del administrador como postulaciones a determinadas ofertas, opiniones dadas sobre las mismas.

5.4.2. CAPA DE NEGOCIO

Para diseñar la capa de negocio se utilizó Django en su versión 3.1.3, con Github repositorio y manejador de dependencias.

Esta capa cuenta con el acceso a la capa de persistencia utilizando las librerías de mysql-client, para el mapeo de las bases relacionales MariaDB.

La capa de negocio consume los servicios de Scrapy¹⁸, las arañas están alojadas en un servidor Scrapy, el servidor Scrapy implementa una API REST, recibe peticiones desde el administrador para iniciar las arañas, pararlas o preguntar acerca de su estado.

Splash es un navegador que se maneja específicamente por líneas de comando, recibe scripts en lenguaje LUA¹⁹, para poder interpretar Javascript, dado que Scrapy no es capaz de interpretar Javascript, de aquí la necesidad de usar un navegador, en nuestro caso, para utilizar Splash hizo falta el uso de un contenedor Docker que ya viene preconfigurado para correr el navegador Splash, dicho contenedor implementa un servicio web que permite el uso del navegador para hacer pruebas, como para su utilización en general.

Finalmente hace falta utilizar IP dinámica en la red Tor²⁰ para evitar el baneo en los sitios

5.4.3. CAPA DE PRESENTACIÓN

Se utilizó un diseño responsive para que permita el uso de la aplicación en dispositivos que trabajan con distintas resoluciones, lo cual es totalmente necesario en plataformas de este tipo.

5.5. PROBLEMAS ENCONTRADOS

Las dificultades a las que nos enfrentamos fueron nuevas tecnologías tanto Scrapy como Django, que si bien resultaron siendo totalmente prácticas y útiles una vez entendidas, su curva de aprendizaje fue bastante alta.

Al tratarse de una aplicación cuya información pertenece a múltiples sitios scrapeados, el manejo de grandes cantidades de datos no homogéneos, fue un desafío persistente.

Para la realización del scraping se tuvo que poseer un constante seguimiento de los permisos y formatos de las páginas a scrapear, ya que estas están en constante cambio, revisión y rediseño.

5.6. HORAS DEDICADAS

Según la planificación inicial se estimaron 365 horas en total, el tiempo real de trabajo en el proyecto tuvo una desviación del 66% en base a lo estimado. En total se trabajaron 565 horas.

Horas previstas y horas reales

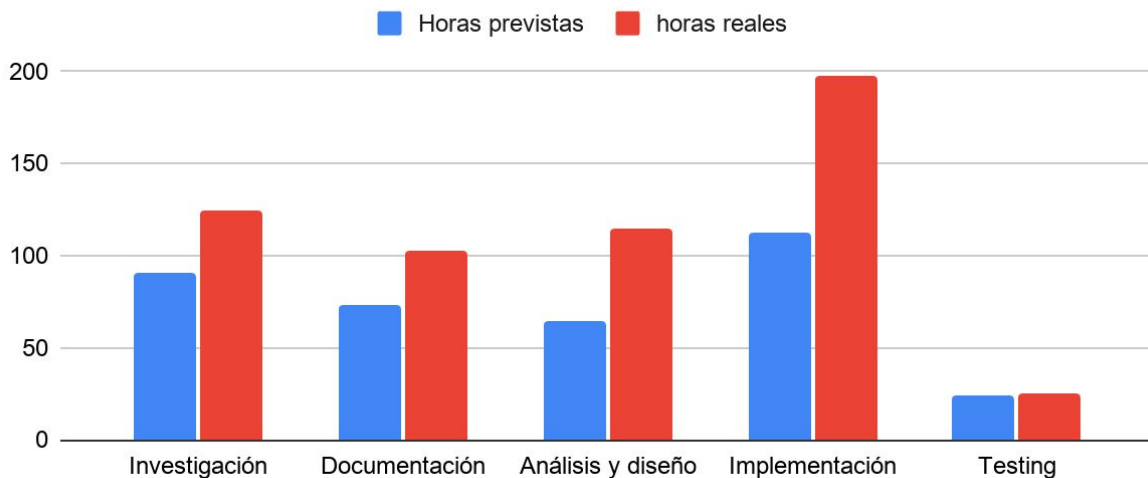


Imagen 12 - Comparación por etapa de horas previstas con horas reales.

5.7. LA HERRAMIENTA DESARROLLADA

La plataforma lograda, bajo el nombre de “UyJob”, está compuesta por una plataforma web, encargada de recibir tanto usuarios interesados en la búsqueda de propuestas laborales como dueños y empresas en busca de ofrecer trabajos.

En dicha aplicación se puede encontrar un backoffice orientado a administradores y empleados de la empresa y un frontoffice destinado a los usuarios. En el backoffice se centraliza la administración donde su funcionalidad principal entre otras es poder calificar las entrevistas.


Mientras que el frontoffice ofrece a los clientes la posibilidad de realizar búsquedas según sus parámetros de necesidad, obtener resultados de forma eficiente y poder concretar su postulación a la oferta laboral deseada.

A continuación se ilustran imágenes de la interfaz de la plataforma.



Imagen 13 - Página de inicio de la plataforma web “UyJob”.

Imagen 14 - Ingreso de Currículum (Rol usuario).



[Calculadora](#)
[Entrevistas](#)
[Estadísticas](#)
[Cerrar Sesión](#)

Resultados de la búsqueda

Guardia para Locales Comerciales HUMANWARE selecciona Guardia para locales comerciales de importante empresa del Rubro Inmobiliario e...	Uruguay
Guardia de Seguridad Importante empresa selecciona Guardia de Seguridad para local comercial en el Centro de Montevideo,...	Uruguay
VIGILANTE PARA EDIFICIO CON EXPERIENCIA 15 A 23 HORAS buscamos un vigilante para edificio en zona pocitos , de 15 horas a 23 , con experiencia,, No somos ...	Uruguay
ASISTENTE ADMINISTRATIVO PARA EMPRESA DE SEGURIDAD Se solicita Asistente administrativo/operativo para empresa de seguridad (masculino o femenino), La...	Uruguay


Detalles del llamado

Guardia para Locales Comerciales
 HUMANWARE selecciona Guardia para locales comerciales de importante empresa del Rubro Inmobiliario en Punta del Este., Requisitos: , • Entre 25-40 años, • Experiencia en el cargo de al menos 1 año, • Buena comunicación, trato cordial., Dentro de sus principales tareas se encuentran: • Manejo de la seguridad del edificio y locales como prioridad en las decisiones tomadas a diario, respetando los procedimientos en caso de robo o episodio de inseguridad, • Mantener siempre buena disposición, trato cordial., • Verificar que los visitantes estacionen correctamente en los lugares destinados a tal fin, comunicando y/o corrigiendo en caso de ser necesario., • Realizar rondas de recorridos por el perímetro verificando que esté todo correcto, • Control de ingreso y egreso de personas y vehículos a los accesos y estacionamiento linderos al edificio., • Apoyo en el mantenimiento del orden y limpieza de los espacios asignados., • Control y rápida respuesta ante situaciones de riesgo o emergencia, • Permanencia en el puesto de trabajo atento a las observaciones y órdenes., fin.

Valido desde Sept. 15, 2020 hasta Nov. 14, 2020

[Postularme](#)
[Cerrar](#)

Imagen 15 - Resultado de oferta laboral (Rol usuario).



[Entrevistas](#)
[Estadísticas](#)
[Ver Perfil](#)
[Cerrar Sesión](#)

Calculadora de Habilidades

Ingrese habilidad

Puede seleccionar varias habilidades

Lista de habilidades:

UYU

USD

EUR

Jornalero

Mensual

Posible sueldo: Diseño y medios;

Desde 150, hasta 700 UYU/hr

Consultar

Imagen 16 - Calculadora de salarios posibles por habilidades (Rol usuario).



Imagen 17 - Grafías (Rol usuario).

Imagen 18 - Control de arañas (Rol administrador).

Imagen 19 - Gestión de entrevistas(Rol administrador).

Imagen 20 - Entrevistas pendientes (Rol usuario).



Imagen 21 - Calificar entrevista (Rol administrador).

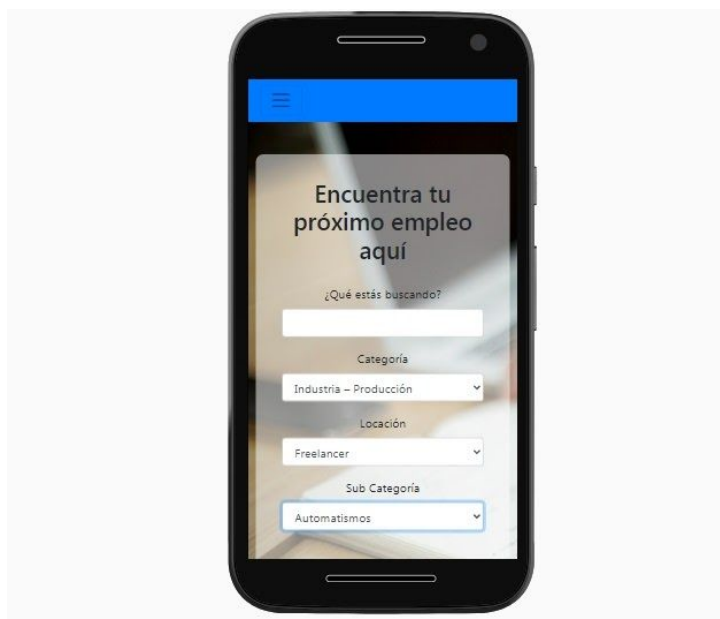


Imagen 22 - Búsqueda, vista desde SmartPhone.

6. CONCLUSIONES

Para poder llevar a cabo este proyecto se tuvo que estudiar la esencia de las páginas que brindan servicios de búsqueda de empleo, el mercado en el que operan, y los posibles clientes a captar. Al realizar dicha investigación se analizó el enfoque y las prestaciones de empresas nacionales e internacionales y se detectó que en Uruguay la gran mayoría de las empresas de este rubro tienen un deber en cuanto a las cantidades y diferentes fuentes donde brindan su información.

En Uruguay, al día de hoy no existe una empresa nacional del rubro búsqueda de trabajo que permita a sus clientes realizar búsquedas en el área pública como privada en su plataforma web. Para poder llevar a cabo un desarrollo que cumpliera con las necesidades requeridas, se plantearon 10 casos de uso para la plataforma web. El 100% de los casos fueron abordados y completados con éxito, en base a las opciones brindadas a nivel local, se considera que se ha obtenido una plataforma eficaz, amigable y competente dentro del plano mencionado, asimismo, debido a los tiempos acotados, se decidió por cubrir solamente los requerimientos solicitados inicialmente por el cliente dejando de lado algunas ideas que surgían durante el desarrollo del proyecto.

Se considera que la aplicación aún tiene mucho margen de crecimiento, si se la compara con alguna otra existente dentro del plano internacional.

A continuación, se presenta el apartado “trabajos a futuro” donde se plantean posibles mejoras que ayudarían a esta aplicación a competir y ser una excelente opción en el plano internacional.

6.1. TRABAJOS A FUTURO

Mejoras en la interfaz de usuario, como por ejemplo, que la palabra clave que uno busca aparezca resaltada en los resultados, que se produzca un autocompletado mediante ajax.

Garantizar que la imagen del curriculum quede bien, así sea evitando que el usuario ingrese una imagen con una resolución errónea o reajustando la misma.

Que las gráficas cambien mediante botones para determinar la moneda.

También podría ser posible que el curriculum se descargue como “.pdf”.

Respecto al multiplataforma, hacer una aplicación específica para smart devices.

Sobre los resultados obtenidos, se podría hacer scraping a un mayor número de páginas internacionales para obtener más datos.

Se podría implementar un módulo que sirva como deploy para las arañas utilizando websocket para evadir la limitación del Scrapy para disponer de una opción más personalizable.

Implementar inteligencia artificial para ordenar y clasificar mejor la información teniendo en consideración el contexto y la semántica de la misma.

Utilización de notificaciones que avisen a los usuarios de que la página está en mantenimiento.

7. REFERENCIAS

- 1 Python. Recuperado el 2 de septiembre del 2020, de <https://www.python.org/>
- 2 Machine Learning. Recuperado el 8 de diciembre del 2020, de <https://cleverdata.io/que-es-machine-learning-big-data/>
- 3 DevOps. Recuperado el 9 del diciembre del 2020, de <https://www.paradigmadigital.com/techbiz/que-es-devops-y-sobre-todo-que-no-es-devops/>
- 4 Web Scraping. Recuperado el 8 de diciembre del 2020, de <https://aukera.es/blog/web-scraping/>
- 5 Ley Protección de Datos Personales y garantía de los derechos digitales. Recuperado el 16 de noviembre del 2020, de <https://www.boe.es/buscar/doc.php?id=BOE-A-2018-16673>
- 6 Scrapy. Recuperado el 20 de septiembre del 2020, de <https://scrapy.org/>
- 7 Minería de Datos. Recuperado el 10 de diciembre del 2020, de https://es.wikipedia.org/wiki/Miner%C3%ADa_de_datos
- 8 Django. Recuperado el 10 de septiembre del 2020, de <https://www.djangoproject.com/>
- 9 Open Source. Recuperado 20 de septiembre del 2020, de <https://openexpoeurope.com/es/open-source-puede-ayudarte/>
- 10 Splash. Recuperado el 27 de septiembre del 2020, de <https://splash.readthedocs.io/en/stable/>
- 11 JavaScript. Recuperado el 1 de diciembre del 2020, de <https://es.wikipedia.org/wiki/JavaScript>
- 12 API Http. Recuperado 20 de septiembre del 2020, de <https://www.redhat.com/es/topics/api/what-are-application-programming-interfaces>
- 13 Twisted. Recuperado el 11 de diciembre del 2020, de <https://pypi.org/project/Twisted/>
- 14 Docker. Recuperado el 4 de noviembre del 2020, de <https://www.docker.com/>
- 15 Kernel Linux. Recuperado el 5 diciembre del 2020, de <https://www.kernel.org/>
- 16 Modelo de vistas 4+1. Recuperado el 12 de diciembre del 2020, de https://es.wikipedia.org/wiki/Modelo_de_Vistas_de_Arquitectura_4%2B1
- 17 Pillow. Recuperado el 20 de octubre del 2020, de <https://pypi.org/project/Pillow/>
- 18 Scrapyd. Recuperado el 6 de octubre del 2020, de <https://scrapyd.readthedocs.io/en/stable/>
- 19 LUA. Recuperado el 27 de septiembre del 2020, de <https://www.lua.org/>
- 20 TOR. Recuperado el 5 de diciembre del 2020, de <https://www.torproject.org/es/>

Anexos

Anexo 1 – Estado del arte del Web Scraping

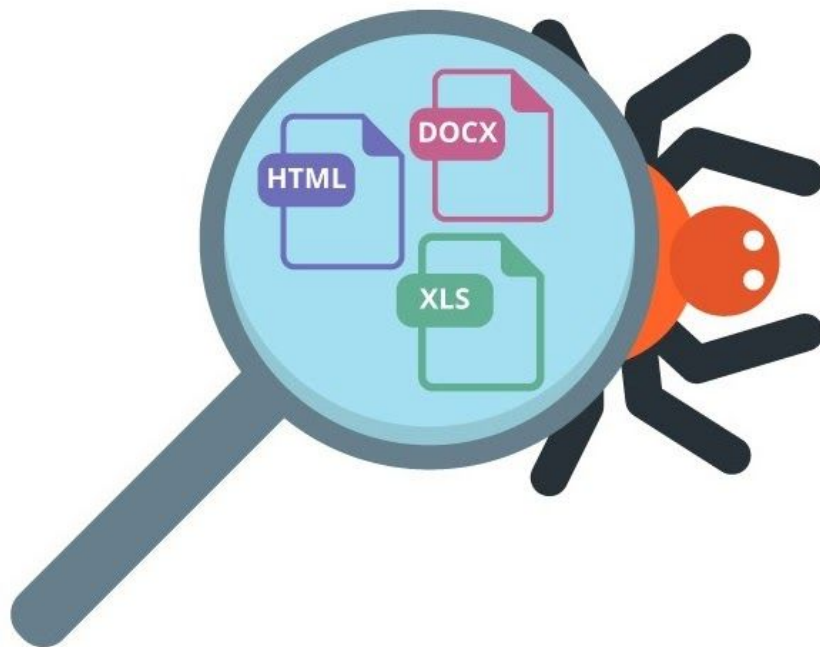
Anexo 2 – Requerimientos del Sistema

Anexo 3 – Arquitectura del Sistema

Anexo 1 - Estado del arte del Web Scraping

Estudio del estado del arte de las principales tecnologías relacionadas con el Web Scraping y la extracción de información

Integrantes:
Julio Arrieta
Joaquín Suárez
Carlos Balbiani



Índice

1. Definición de Web Scraping:
2. Algunas de las tecnologías más representativas.
3. Criterios de evaluación.
 - 3.1. Capacidad de crawling personalizado:
 - 3.2. Calidad de la Documentación:
 - 3.3. De pago:
 - 3.4. Interfaz gráfica:
 - 3.5. Machine learning.
4. Evaluación particular.
 - 4.1. Import.io.
 - 4.1.1. Contextualización.
 - 4.1.2. Evaluación.
 - 4.2. Hunter.
 - 4.2.1. Contextualización.
 - 4.2.2. Evaluación.
 - 4.3. WebHose.io.
 - 4.3.1. Contextualización.
 - 4.3.2. Evaluación.
 - 4.4. Octoparse.
 - 4.4.1. Contextualización.
 - 4.4.2. Evaluación.
 - 4.5. BeautifulSoup.
 - 4.5.1. Contextualización.
 - 4.5.2. Evaluación.
 - 4.6. Jsoup.
 - 4.6.1. Contextualización.
 - 4.6.2. Evaluación.
 - 4.7. Scrapy.
 - 4.7.1. Contextualización.
 - 4.7.2. Evaluación.
5. Conclusión.
6. Referencias.

1. Definición de Web Scraping.

El Web Scraping se trata de la recolección web o la extracción de datos web, se traduce como “raspado de datos” que se utiliza para extraer datos de sitios web.

Está automatizado porque utiliza bots para obtener la información del sitio web.

Es un análisis programático de una página web para descargar información de ella.

El scraping de datos implica localizar los datos y luego extraerlos. No copia y pega, sino que, obtiene directamente los datos de manera precisa. No se limita a la web, los datos se pueden obtener prácticamente desde cualquier lugar donde se almacenan. Puede ser Internet u otra fuente de datos.

Ejemplo de web scraping: El web scraping implicaría obtener la información de una página web específica, por ejemplo, obtener la información de precios de determinados productos de las páginas de Amazon.

2. Algunas de las tecnologías más representativas.

Estas son algunas tecnologías disponibles, aunque existen muchas más, nos centraremos solo en estas, con el fin de no extender demasiado este documento.

Import.io, Hunter.io, Webhose.io, Octoparse, BeautifulSoup, Jsoup, Scrapy.

3. Criterios de evaluación.

Vamos a evaluar las siguientes tecnologías mediante diversos criterios y realizaremos una comparación entre las mismas, para determinar, la tecnologías que más se adapte a nuestras necesidades.

3.1. Capacidad de crawling personalizado:

El término de crawler viene de la forma en la cual se desplaza una araña. Básicamente es un bot de internet que navega en forma sistemática en la World Wide Web, generalmente con el propósito de indexar la web. El crawling personalizado implica, que el usuario puede, de forma más o menos sencilla, elaborar un crawler con las características específicas deseadas. No entran aquí servicios de oferta de crawlers predefinidos.

3.2. Calidad de la Documentación:

Que tan buena, bien explicada y ejemplificada es la documentación de esta tecnología e incluso su comunidad y repositorios públicos gratuitos, esta característica permite reducir la curva de aprendizaje y puesta en marcha de sistemas para la recolección de datos usando cada herramienta. Debido a la importancia de este último criterio, no podemos simplemente observar si dispone de ella o no, debemos considerar fuertemente su calidad.

3.3. De pago:

La aplicación requiere una habilitación monetaria previa para su uso, sin contar con las demos o simplemente es gratis, esta característica será vital, dado el carácter académico del proyecto.

3.4. Interfaz gráfica:

No evaluaremos la experiencia de usuario con la interfaz gráfica, sino, simplemente, si dispone de ella o no. Cuanto más intuitiva sea una herramienta, más limitadas serán sus opciones de uso. Por otro lado, aquellas que no ofrecen ninguna clase de facilidad al usuario no técnico, posibilitan, mayor escalabilidad, mayores casos de uso y mayor potencia.

3.5. Machine learning.

Algunas tecnologías brindan técnicas de machine learning e incluso Inteligencia Artificial lo cual posibilita el mejoramiento de los algoritmos de búsqueda. Sin Embargo estas técnicas requieren muchos recursos, lo cual encarece el servicio de web scraping, de modo que evaluaremos, a una tecnología simplemente por el hecho de disponer de esta característica o no.

4. Evaluación particular.

4.1. Import.io.



4.1.1. Contextualización.

Import.io, es una plataforma que facilita la conversión de información semiestructurada en páginas web en datos estructurados, que se pueden utilizar para cualquier cosa, desde la toma de decisiones comerciales hasta la integración con aplicaciones y otras plataformas. Ofrecemos recuperación de datos en tiempo real a través de nuestras API de transmisión y basadas en JSON REST, e integración con muchos lenguajes de programación y herramientas de análisis de datos comunes, es una de las más utilizadas para web scraping.

4.1.2. Evaluación.

Dispone de crawling personalizable, tiene una documentación de muy buena calidad, pero desafortunadamente no es gratuita y no solo eso, es bastante costosa, pero una versión gratuita, pero solo dura 48 horas, lo que la convierte en mala candidata para realizar investigaciones y pruebas de concepto. Dispone de una interfaz gráfica de usuario amigable y fácil de utilizar. También permite utilizar algoritmos de Machine Learning para automatizar la recolección de datos.

4.2. Hunter.



4.2.1. Contextualización.

Hunter.io, se trata de una herramienta de extracción de datos, pero centrada en correos electrónicos, es decir, un buscador de emails en dominios personalizados. Útil si quieres contactar con alguna compañía, proporciona datos en línea para crear conexiones entre profesionales. Elabora una lista de todas las direcciones de correo electrónico de una empresa o un dominio disponible públicamente en la web y permite a sus usuarios encontrar direcciones de correo electrónico específicas desde el nombre, el apellido y el sitio web de la empresa.

4.2.2. Evaluación.

No dispone de un crawling personalizable, sin embargo tiene muy buena documentación. tiene una versión de pago y otra gratuita pero es limitada. no dispone de una interfaz gráfica y no proporciona herramientas para machine learning.

4.3. WebHose.io.



4.3.1. Contextualización.

Webhose.io, para los perfiles más técnicos, webhose se convierte en una muy buena alternativa. Tienen sus propios scrapers de datos y un servicio de estructuración que facilitan el scrapeo de sitios, además posibilita obtener datos estructurados obtenidos de sitios no estructurados conectándose a una API, a la que puedes hacer 1000 peticiones al mes gratuitas. Así, es fácil obtener los datos en formatos estandarizados y estructurados como XML o JSON.

4.3.2. Evaluación.

Al igual que Hunter.io, no dispone de un crawling personalizable, sin embargo tiene muy buena documentación. tiene una versión de pago y otra gratuita pero es limitada, no dispone de una interfaz gráfica y no proporciona herramientas para machine learning.

4.4. Octoparse.



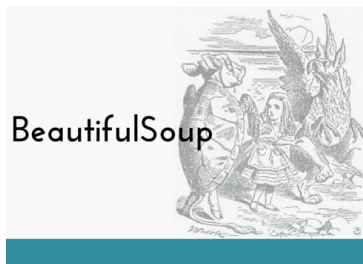
4.4.1. Contextualización.

Octoparse, es un moderno software de extracción visual de datos web. Tanto a los usuarios experimentados como a los que no tienen experiencia les resultaría fácil, una buena herramienta para gente que busca extraer datos de sitios web sin tener que codificar nada.

4.4.2. Evaluación.

No dispone de un crawling personalizable y su documentación no es muy buena. Sin embargo es gratuita, y posee una muy amigable interfaz gráfica. No dispone de herramientas que posibiliten el uso de machine learning.

4.5. BeautifulSoup.



4.5.1. Contextualización.

Beautiful Soup, una librería de Python lo cual de por sí ofrece muchas ventajas, funciona con su analizador de favoritos, para proporcionar formas idiomáticas de navegar, buscar y modificar el árbol de análisis. Por lo general, ahorra a los programadores horas o días de trabajo, sin embargo el usuario debe encargarse de muchas tareas y estás limitado a sistemas locales, si lo instalas en arquitecturas cloud, pierdes la ventaja de la gratuidad.

4.5.2. Evaluación.

Dispone de un crawling muy personalizable, pero su documentación no es bastante mala. Sin embargo es gratuita, no posee una interfaz gráfica y no dispone de herramientas que posibiliten el uso de machine learning.

4.6. Jsoup.



4.6.1. Contextualización.

Jsoup es una librería Java que proporciona operaciones para trabajar con HTML. Permite extraer y manipular datos, que podrán ser utilizados convenientemente para nuestras necesidades.

Con Jsoup podemos construir desde parseadores básicos de HTML para analizar y procesar páginas estáticas hasta herramientas de análisis recursivo de sitios completos (crawlers o spiders). No obstante, Jsoup está más pensado para análisis de páginas estáticas que para un crawler complejo. Si lo que queremos es recopilar diferentes tipos de datos de un sitio completo independientemente de sus URLs, puede ser más adecuado utilizar Crawler.

4.6.2. Evaluación.

Dispone de un crawling muy personalizable, con una muy buena documentación.

Es gratuita, no posee una interfaz gráfica y no dispone de herramientas que posibiliten el uso de machine learning.

4.7. Scrapy.



4.7.1. Contextualización.

Scrapy es una plataforma colaborativa de código libre que corre en Python orientada al web scraping, utilizado en sitios web para rastrear y extraer datos estructurados de sus páginas. Se puede utilizar para una amplia gama de propósitos, desde minería de datos hasta monitoreo y pruebas automatizadas.

Además es la herramienta más ampliamente utilizada para este propósito.

4.7.2. Evaluación.

Al igual que Jsoup dispone de un crawling muy personalizable, con una muy buena documentación. Es gratuita, no posee una interfaz gráfica y no dispone de herramientas que posibiliten el uso de machine learning.

5. Conclusión.

Dado que el proyecto que desarrollaremos es con fines académicos, y no disponemos de un gran presupuesto, las opciones pagas quedan descartadas (import.io), incluso las opciones con versiones gratuitas pero limitadas (Hunter.io y webhouse), dado que disponemos de opciones completamente gratuitas. Hemos descartado las opciones con una mala documentación por motivos obvios (Octoparse y BeautifulSoup). Finalmente debimos decidir entre Jsoup y Scrapy, pero Scrapy funciona con python y dado que se nos ha impuesto ese lenguaje, esta última opción es la más adecuada a nuestro propósito.

6. Fuentes utilizadas:

1. Import.io : <https://www.import.io/>
2. Hunter.io : <https://hunter.io/>
3. Webhose.io : <https://webhose.io/>
4. Octoparse : <https://www.octoparse.com/>
5. BeautifulSoup : <https://www.crummy.com/software/BeautifulSoup/>
6. Jsoup : <https://jsoup.org/>
7. Scrapy : <https://scrapy.org/>
8. herramientas de web scraping:
<https://papelesdeinteligencia.com/herramientas-de-web-scraping/>

Anexo 2 – Requerimientos del Sistema



Integrantes:
Julio Arrieta
Joaquín Suárez
Carlos Balbiani

1. REQUERIMIENTOS DEL SISTEMA

1.1. Propósito

El propósito de este documento es servir como medio de comunicación entre clientes y desarrolladores. Aquí deben tomarse en cuenta tanto las necesidades de los clientes y usuarios como los requisitos que debe cumplir el sistema a desarrollar para satisfacer dichas necesidades.

Tiene un carácter contractual, por lo que cualquier cambio que se desee realizar una vez se haya acordado la primera línea base, debe aplicarse siguiendo el procedimiento de Control de Cambios establecido, este documento se utiliza entonces, para recoger la especificación de requisitos funcionales y no funcionales del sistema a desarrollar y sirve como base contractual a los desarrolladores del sistema.

1.2. Alcance

El objetivo de este proyecto, es permitir buscar y acceder a diferentes ofertas laborales, nacionales e internacionales.

El objetivo es disponer de un sistema que permita a un conjunto de usuarios, buscar ofertas laborales. Dentro de la aplicación se cuenta con un módulo web que permite a los usuarios registrarse e introducir información personal que servirá como currículum, desde la misma podrán realizar búsquedas en la bolsa laboral mediante determinados criterios, podrán además postularse a las ofertas laborales que deseen, de las cuales posteriormente podrían recibir notificaciones de entrevistas. Por otra parte, los usuarios de tipo “administrador” podrá designar entrevistas a los postulantes, así como fijar un conjunto de posibles fechas para las mismas, y calificar mediante un sistema de puntos, una vez el postulante realice la entrevista, el administrador designará como realizada la misma.

Además el sistema contará con una opción de “superusuario”, que tendrá todos los privilegios sobre la base de datos.

2. Descripción general

2.1. Perspectiva del producto

2.1.1. Interfaces de usuario

El sistema contará con una única interfaz web de usuario la interfaz Web contará con una UI atractiva que permitirá a los usuarios una cómoda visualización en cualquier dispositivo.

Tendrá también una fuerte inclinación a la experiencia de usuario, permitiendo así que todas las interacciones de los usuarios sean en forma eficiente y ágil.

2.1.2. Interfaces con software

Plataforma Web

La plataforma web será accesible desde cualquier navegador actual con soporte para HTML5. De todas formas se detallan los requerimientos mínimos:

Google Chrome 48

Internet Explorer 10

Mozilla Firefox 41

2.2. Funciones del producto

2.2.1. Aplicación web

FrontOffice

Registro de Usuario.

Subir curriculum.

Iniciar sesión.

Buscar oferta laboral.

Ver detalle de oferta.

Ver estadísticas.

Postularse a un trabajo.

Entrevistas pendientes.

BackOffice

Ver ofertas activas.

Ver postulantes.

Ver curriculum de postulante.

Entrevistar.

Calificar entrevista.

2.3. Características de los usuarios

El sistema cuenta con los siguientes roles predefinidos:

2.3.1. Invitados

Son aquellos usuarios que ingresan de manera anónima al sitio y que pueden buscar ofertas laborales y ver estadísticas.

2.3.2. Usuarios

Son aquellos usuarios que se registran a través de la aplicación web y pueden buscar ofertas laborales, ver estadísticas, ingresar curriculum, postularse, seleccionar fecha de entrevista.

2.3.3. Administradores

Son aquellos usuarios que inician sesión a través de la aplicación web y pueden ver ofertas laborales ver postulantes de las mismas, asignar entrevistas y calificarlas.

3. Modelo de Casos de Uso

3.1. Registro de Usuario

3.1.1. Descripción

El usuario se registra ingresando una dirección de correo electrónico y contraseña.

3.1.2. Pre-condiciones

El usuario no debe estar registrado.

El correo debe existir.

3.1.3. Post-condiciones

Se creó un usuario en el sistema con los datos ingresados.

3.2. Subir curriculum.

3.2.1. Descripción

El usuario sube su currículum en formato pdf.

3.2.2. Pre-condiciones

El usuario debe estar registrado en el sistema.

El archivo debe tener un formato válido.

3.2.3. Post-condiciones

Se ingresó el curriculum del usuario en el sistema.

3.3. Iniciar sesión.

3.3.1. Descripción

El usuario y administrador ingresa su correo y contraseña, el sistema valida los datos e ingresa

a la vista correspondiente.

3.3.2. Pre-condiciones

Debe existir un usuario en el sistema con mail y contraseña dado.

3.3.3. Post-condiciones

El usuario quedó logueado en el sistema.

3.4. Buscar oferta laboral.

3.4.1. Descripción

El usuario e invitado están habilitados para buscar ofertas laborales, Ingresa algún identificador de una oferta laboral que busca con la posibilidad de agregar filtros.

3.4.2. Pre-condiciones

El actor deberá estar en la página de búsqueda para buscar la oferta laboral.

3.4.3. Post-condiciones

Se despliega en pantalla una lista de ofertas laborales activas.

3.5. Ver detalle de oferta.

3.5.1. Descripción

El usuario e invitado, podrán ver detalles de la oferta.

3.5.2. Pre-condiciones

El actor debe haber seleccionado una oferta.

3.5.3. Post-condiciones

El sistema despliega los detalles de la oferta seleccionada.

3.6. Ver estadísticas.

3.6.1. Descripción

Un usuario o invitado podrá seleccionar entre diferentes tipos de estadísticas basadas en las ofertas laborales, estas se visualizarán tanto en forma de tabla o gráfico.

3.6.2. Pre-condiciones

El actor deberá estar en la página ver estadísticas.

3.6.3. Post-condiciones

El sistema exhibe las estadísticas, al usuario.

3.7. Postularse a un trabajo.

3.7.1. Descripción

Un usuario registrado podrá postularse a una oferta de trabajo que le interese.

3.7.2. Pre-condiciones

La oferta debe estar vigente. El usuario no debe estar postulado para la oferta laboral en cuestión.

3.7.3. Post-condiciones

El sistema agrega al usuario a la lista de postulantes para la oferta laboral actual.

3.8. Entrevistas pendientes.

3.8.1. Descripción

El usuario ingresa en la página de entrevistas pendientes, donde designa la fecha de aquellas a las que desee postularse.

3.8.2. Pre-condiciones

El usuario debe estar registrado en el sistema, debe haberse postulado a las ofertas laborales para las cuales se lo está entrevistando.

3.8.3. Post-condiciones

El sistema lista las entrevistas pendientes.

3.9. Ver ofertas activas.

3.9.1. Descripción

El administrador verá una pantalla donde se mostrarán todas las ofertas activas

3.9.2. Pre-condiciones

El administrador debe tener sesión iniciada

3.9.3. Post-condiciones

El administrador será capaz de ver todas las ofertas activas de los diferentes trabajos

3.10. Ver postulantes.

3.10.1. Descripción

El Administrador verá una pantalla donde se encuentran todos los usuarios postulados a las ofertas aún activas, para una oferta en particular.

3.10.2. Pre-condiciones

El administrador debe tener la sesión iniciada

3.10.3. Post-condiciones

Podrá ver una lista de todos los postulantes a esa oferta activa en particular

3.11. Ver curriculum de postulante.

3.11.1. Descripción

Se desplegará una pantalla donde se verá el currículum del postulante a detalle

3.11.2. Pre-condiciones

El usuario debe estar registrado

El llamado debe estar aún válido

3.11.3. Post-condiciones

El administrador será capaz de ver a detalle cada curriculum

3.12. Asignar fecha de entrevista.

3.12.1. Descripción

El administrador establecerá determinadas fechas para realizar la entrevista

3.12.2. Pre-condiciones

La postulación debe existir y aún permanecer vigente

3.12.3. Post-condiciones

Se le enviará al usuario las fechas dispuestas por el administrador

3.13. Calificar entrevista.

3.13.1. Descripción

El Administrador tendrá la posibilidad de calificar una entrevista, la cual se le otorgara cierto puntaje

3.13.2. Pre-condiciones

Debe existir una entrevista previa ,donde usuario y administrador se contactaron

3.13.3. Post-condiciones

Se creará una calificación correspondiente a esa postulación

4.1. Restricciones de diseño

4.1.1. Lenguajes e IDE utilizados

Se utiliza Python como lenguaje para este desarrollo. Para esto mismo se utilizará la versión Python 3.8 con el IDE Visual Studio Code, el cual permite un desarrollo eficaz y veloz.

5. Requerimientos específicos

5.1. Usabilidad

La aplicación tendrá un fuerte enfoque hacia el usuario, simplificando lo más posible el manejo de la misma, sin importar el dispositivo en el cual se esté utilizando.

Una de las principales metas de esta aplicación es permitir a los usuarios clientes obtener resultados específicos a la hora de rentar un vehículo, con efectividad, eficiencia y satisfacción. Y a los usuarios de la empresa obtener mayor eficacia en la gestión de la misma. El sistema también ayudará a minimizar el riesgo de error (mediante validaciones y mensajes de error).

5.2. Amigabilidad

De la mano con la usabilidad, se destaca la amigabilidad del sistema. Esto hace referencia al hecho de que un usuario encuentre el sistema fácil de usar, por lo cual esta aplicación tendrá una interfaz de usuario rica pero sencilla además de ser visualmente atractiva.

5.3. Performance

Otra de las características más importantes de la aplicación consistirá en la eficiencia al realizar las tareas solicitadas por los usuarios. La idea será por consiguiente, utilizar los recursos computacionales de la forma más económica posible, para así aumentar la productividad de los usuarios disminuyendo los tiempos de respuesta de la aplicación.

Anexo 3 – Arquitectura del Sistema

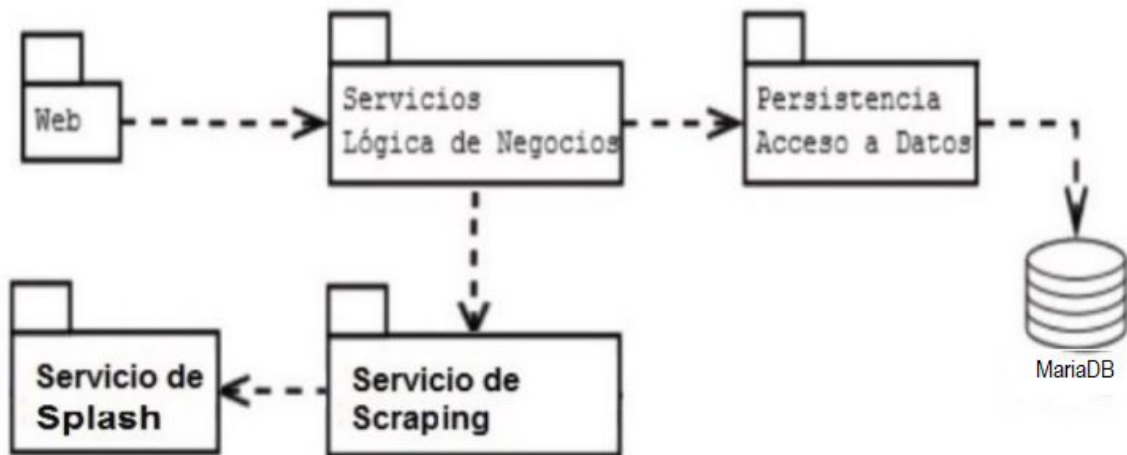


Integrantes:
Julio Arrieta
Joaquín Suárez
Carlos Balbiani

1. Trazabilidad Modelo de Casos de Uso al Modelo de Diseño.

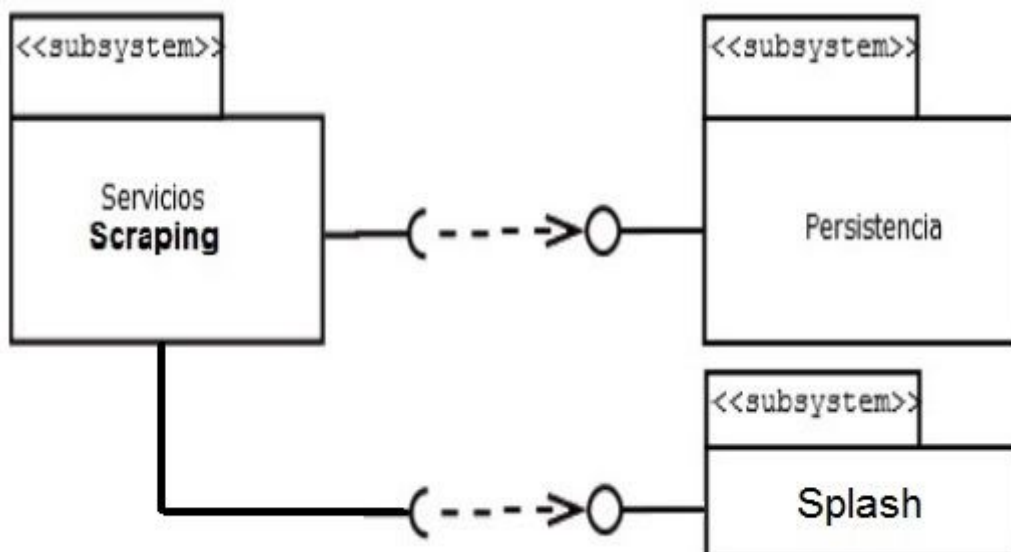
La Vista Lógica se utiliza para marcar la trazabilidad entre el Modelo de Casos de Uso y el Modelo de Diseño, identificando los objetos y subsistemas de diseño que intervienen en el caso de uso y sus relaciones.

Para este fin se utiliza el Diagrama de Paquetes:



2. Vista del Modelo de Diseño

2.2. Descomposición en Subsistemas



2.2.1. Servicios Scraping

Este subsistema provee las funcionalidades de scraping, trayendo los recursos que serán persistentes por otros subsistemas.

2.2.2. Persistencia

Este subsistema se encarga del acceso a datos persistentes necesarios para que la aplicación mantenga su información en el tiempo. Ofrece interfaces al subsistema de Servicios (los datos son consumidos por la Lógica de Negocio).

3. Vista del Modelo de Distribución

