



UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

DESARROLLO DE UN SISTEMA DE EVALUACIÓN DE POSTULACIONES AL
MAGÍSTER EN TECNOLOGÍAS DE LA INFORMACIÓN

MEMORIA PARA OPTAR AL TÍTULO DE INGENIERO CIVIL EN COMPUTACIÓN

NICOLÁS SALAS VEGA

PROFESOR GUÍA:
SERGIO OCHOA DE LORENZI
PROFESOR GUÍA 2:
DANIEL PEROVICH GEROSA

MIEMBROS DE LA COMISIÓN:
JUAN ÁLVAREZ RUBIO
CLAUDIO GUTIÉRREZ GALLARDO

SANTIAGO DE CHILE
2021

Resumen

El Programa de Magíster en Tecnologías de la Información (MTI), que imparte el Departamento de Ciencias de la Computación de la Universidad de Chile, recibe de forma online postulaciones de ingreso al programa a través de la plataforma UCampus. Estas postulaciones deben ser descargadas por el personal de apoyo, y luego evaluadas por miembros del comité académico del programa. Estos últimos emiten una recomendación de aceptación o rechazo para cada postulación. El coordinador del programa analiza esas recomendaciones y emite una resolución al respecto.

Actualmente, el proceso de evaluación de cada postulación se lleva a cabo a través del intercambio de correos electrónicos, coordinados en gran medida por el personal de apoyo al programa. Estas personas se encargan de: (1) enviar los antecedentes a los miembros del Comité Académico, y solicitar su recomendación de aceptación o rechazo, (2) recolectar las recomendaciones, y (3) enviárselas al coordinador para que emita una resolución. Esta forma de procesar las evaluaciones dificulta extraer estadísticas o mantener la historia de las evaluaciones. Sin datos fidedignos acerca del funcionamiento de este proceso, se vuelve difícil poder mejorarlo. Hoy en día el MTI representa un canal importante de transferencia de conocimiento desde el Departamento hacia la industria, por lo tanto, todos sus procesos deben funcionar lo mejor posible.

El objetivo de este trabajo de memoria es desarrollar un sistema que permita automatizar gran parte del procesamiento de las postulaciones al MTI, aliviando la carga de trabajo sobre las personas involucradas, mejorando la gestión de éste y la capacidad de medir distintos aspectos del proceso.

Para alcanzar el objetivo planteado se desarrolló un sistema web que gestiona las evaluaciones de las postulaciones. Éste extrae los datos desde UCampus utilizando un scraper¹, almacena las postulaciones en su base de datos, e implementa un workflow que automatiza gran parte del proceso antes descrito. La solución desarrollada fue evaluada con postulaciones de prueba, donde dos miembros del comité académico ejecutaron el proceso completo. Durante las pruebas, estas personas asumieron los distintos roles involucrados en el proceso y ejecutaron las tareas correspondientes.

Como resultado de las pruebas se comprobó que el sistema soporta correctamente el proceso de evaluación de postulaciones, y sólo se indicaron mejoras a la usabilidad de las interfaces del mismo. Estas mejoras serán abordadas como parte del trabajo a futuro de este proyecto.

¹ Un *scraper* es un programa que visita sitios web y extrae datos desde éstos de forma automatizada.

Agradecimientos

A mis padres, quienes siempre me apoyaron contra todas las inclemencias de la vida.

A mis amigos, quienes me vieron crecer mientras estuve en la Universidad.

A mis empleadores, quienes hicieron posible el término de esta etapa.

A mis profesores guía, quienes me aceptaron como estudiante y guiaron este trabajo.

Finalmente, a mi hermano, quien estuvo conmigo durante las etapas más duras de este proceso.

Tabla de Contenido

1. Introducción	1
1.1. <i>Problema</i>	1
1.2. <i>Objetivos</i>	2
1.3. <i>Estructura del Documento</i>	3
2. Análisis del Problema	4
2.1. <i>Participantes en el Proceso de Evaluación de Postulaciones</i>	4
2.2. <i>Descripción general del Proceso</i>	5
2.3. <i>Obtención de Datos desde la Plataforma UCampus</i>	6
2.4. <i>Análisis de Alternativas de Interacción con UCampus</i>	7
2.4.1. <i>Posible Extensión de la API de UCampus</i>	7
2.4.2. <i>Extracción de Datos de UCampus Usando Scrapers</i>	8
2.4.3. <i>Carga Manual de Datos de Postulaciones</i>	9
2.5. <i>Formulario de Postulación al MTI</i>	10
3. Descripción de la Solución	13
3.1. <i>Descripción del Proceso Automatizado</i>	13
3.2. <i>Principales Requisitos de la Solución</i>	13
3.3. <i>Perfiles de Usuarios</i>	14
3.3.1 <i>Postulante</i>	14
3.3.2 <i>Miembro del Comité Académico del Programa</i>	14
3.3.3 <i>Coordinador de Magíster / Presidente del Comité Académico</i>	15
3.3.4 <i>Asistente / Funcionario del PEC</i>	15
4. Diseño del Sistema	16
4.1. <i>Modelo de Dominio</i>	16
4.2. <i>Arquitectura del Sistema</i>	16
4.3. <i>Modelo de Datos</i>	20
5. Implementación de la Solución	22
5.1. <i>Tecnologías Escogidas para la Implementación</i>	22

5.1.1 Back-end	23
5.1.2 Scraper	26
5.1.3 Deployment	28
5.1.4 Front-end	28
<i>5.2. Interfaces de Usuario</i>	29
5.2.1. Interfaces de Usuario para el Asistente	30
5.2.2. Interfaces de usuario para los miembros del Comité Académico	32
5.2.3. Interfaces de Usuario para el Coordinador	34
6. Evaluación de la Solución	37
6.1. <i>Proceso de Evaluación</i>	37
6.2. <i>Resultados</i>	38
7. Conclusiones y Trabajo a Futuro	40
Bibliografía	43
Anexo A: Definiciones, Siglas y Abreviaturas	45
Anexo B: Formulario de Evaluación de Postulaciones	48

1. Introducción

El Departamento de Ciencias de la Computación de la Universidad de Chile ofrece distintos programas de magíster. Uno de ellos es el Magíster en Tecnologías de la Información (en adelante MTI). Uno de los objetivos de este programa es formar especialistas con conocimientos de aspectos aplicados respecto al uso, gestión y adopción de Tecnologías de la Información y Comunicaciones.

Teniendo en cuenta el perfil de los profesionales que busca formar este programa, y la naturaleza misma del magíster, resulta razonable pensar que el proceso de postulación al mismo, así como el procesamiento de dichas postulaciones, se haga usando herramientas computacionales.

Hoy en día, la postulación a este programa se hace a través de la plataforma UCampus². En esa plataforma, el postulante debe ingresar sus datos personales y adjuntar un conjunto de documentos relevantes para respaldar su postulación. Entre esos documentos están las cartas de recomendación, el currículum del postulante, su certificado de título y su carta motivacional.

Una vez enviada la postulación a través de UCampus, los documentos quedan disponibles para que cada programa (en este caso, el MTI) haga con ellos lo que estime conveniente. UCampus no ofrece la funcionalidad para apoyar el procesamiento de las postulaciones, por lo que cada programa debe hacerlo por fuera de esta plataforma.

1.1. Problema

El proceso de evaluación de postulaciones al MTI se realiza en forma manual. Para ello, un funcionario del Programa de Educación Continua (PEC) del DCC inicia el proceso descargando manualmente cada uno de los archivos subidos a UCampus por el postulante. Esa información luego es enviada a los miembros del comité académico del programa a través de un correo electrónico. Los miembros de ese comité evalúan las postulaciones.

² UCampus es un Centro Tecnológico que desarrolla plataformas de apoyo a la gestión, automatizando los procesos de Instituciones de Educación Superior, en particular el de la Universidad de Chile. Ucampus.cl.

Cuando la información de una postulación está completa, los miembros del Comité Académico emiten un juicio individual y justificado respecto a la admisión o no de cada postulante al programa. Esta información es enviada vía correo electrónico al funcionario del PEC, quien reúne todas las opiniones de los miembros del Comité Académico acerca de una postulación, y luego se las envía al coordinador del programa para que decida la admisión o rechazo del postulante. En esa instancia, el coordinador revisa manualmente todas las evaluaciones y emite un juicio formal, el cual debe ser registrado en la plataforma UCampus, e informado al postulante.

Este flujo de trabajo (workflow) es completamente informal, por lo que los tiempos de procesamiento de una postulación varían mucho, dependiendo de la carga de trabajo que tengan los funcionarios del PEC y el resto de los participantes. Dado que hoy en día las evaluaciones de las postulaciones se realizan de forma manual, el proceso se vuelve lento, costoso y propenso a errores. Además, el flujo de trabajo asociado a este proceso es muy difícil de monitorear, medir y mejorar. Otro aspecto importante es que este proceso manual no es escalable, por lo que se requieren sistemas de apoyo al procesamiento de estas postulaciones.

Tomando en cuenta lo anterior, en esta memoria se diseñó e implementó un sistema Web que:

1. Agiliza el proceso de evaluación de postulaciones al MTI.
2. Facilita la recopilación y almacenamiento de datos relevantes, tanto del postulante como de la evaluación de su postulación.

El sistema no busca extender la funcionalidad de UCampus, ni la validez de la postulación, por lo tanto, en la memoria se asume que los datos que provienen de dicha plataforma son correctos.

1.2. Objetivos

El objetivo general de esta memoria fue desarrollar un sistema Web que automatiza parte del flujo de trabajo asociado al procesamiento de postulaciones al Magíster en Tecnologías de la Información; programa que es impartido por la Escuela de Postgrado de la FCFM³ a través del DCC⁴. Los objetivos específicos definidos para alcanzar el objetivo general fueron los siguientes:

³ Facultad de Ciencias Físicas y Matemáticas de la Universidad de Chile.

⁴ Departamento de Ciencias de la Computación.

1. Desarrollar software que extraiga automáticamente la información de las postulaciones al programa, desde la plataforma UCampus.
2. Desarrollar un sistema que automatice el flujo de trabajo (workflow) de evaluación de las postulaciones.
3. Desarrollar interfaces que faciliten la toma de decisiones del Coordinador del Programa, respecto a la aceptación o rechazo de los postulantes, y a la comunicación de éste con la Escuela de Postgrado.

1.3. Estructura del Documento

En este capítulo se introdujo brevemente el problema abordado, el contexto en el que se encuentra y los objetivos de este trabajo.

El Capítulo 2 describe cuál es la situación actual del proceso que se busca optimizar. Vale decir, entendiendo que el proceso de postulaciones actualmente se realiza, lo que interesa es entender a cabalidad cómo se hace actualmente: actores, fuentes de datos, estructura de los mismos y formularios relevantes.

En el Capítulo 3 se define el alcance de la solución propuesta para apoyar el proceso de evaluaciones. Esto es, definir explícitamente el proceso que se apoyará y cuáles usuarios serán los que participarán de la solución.

Después, el Capítulo 4 introduce el diseño en términos técnicos de la solución. En él se describe cuál es el modelo de dominio del sistema, su arquitectura y el modelo de datos.

Luego, en el Capítulo 5 se discute la implementación del sistema justificando técnicamente las decisiones de implementación tomadas, ya sean lenguajes de programación o tecnologías relevantes. También se muestra el resultado del sistema, describiendo sus interfaces principales para cada actor.

El Capítulo 6 muestra una evaluación del sistema, mostrando cuáles fueron los casos de prueba en el sistema y luego exponiendo los resultados obtenidos.

Finalmente, el Capítulo 7 muestra las conclusiones del trabajo, resumiendo brevemente cuáles eran los objetivos del mismo, la solución desarrollada, sus resultados, el trabajo futuro y lecciones aprendidas.

2. Análisis del Problema

A continuación, se indican los roles de los participantes en el proceso de evaluación de postulaciones. Luego se describe y discute dicho proceso. Finalmente, se analizan las alternativas para extraer la información de las postulaciones desde UCampus, y de esa manera poder alimentar el sistema desarrollado en esta memoria.

2.1. Participantes en el Proceso de Evaluación de Postulaciones

En la siguiente tabla se muestran los actores relevantes (roles o perfiles de usuario), quienes participan en el proceso de evaluación de una postulación al MTI. Además, se presenta una breve descripción de la labor que realiza cada uno en dicho proceso.

Tabla 1. Roles participantes en el proceso de postulación al MTI.

Rol	Descripción
Postulante	Genera una postulación a través de UCampus. Una vez entregada la postulación completa, éste espera por el resultado (aceptación o rechazo) de la misma.
Miembro del Comité Académico del Programa	Evalúa las postulaciones, emite un voto y entrega su opinión al Coordinador de Magíster en forma de comentarios. Estos comentarios justifican la opinión del miembro correspondiente respecto a la aceptación o rechazo de un postulante.
Coordinador de Magíster / Presidente del Comité Académico	Resuelve una postulación. La decisión es tomada en base a las evaluaciones realizadas por los miembros del Comité Académico.
Asistente / Funcionario del PEC	Se encarga de revisar la validez de los documentos adjuntados a la postulación, y deriva la información al Coordinador del Magíster y a los miembros del Comité Académico del Programa cuando corresponde. Los asistentes (funcionarios) son quienes llevan a cabo la mayor parte del workflow manual involucrado en el procesamiento de las postulaciones.

2.2. Descripción general del Proceso

Tomando en cuenta el rol de cada actor, a continuación se resume el proceso que sigue una postulación hasta lograr un veredicto de aceptación o rechazo.

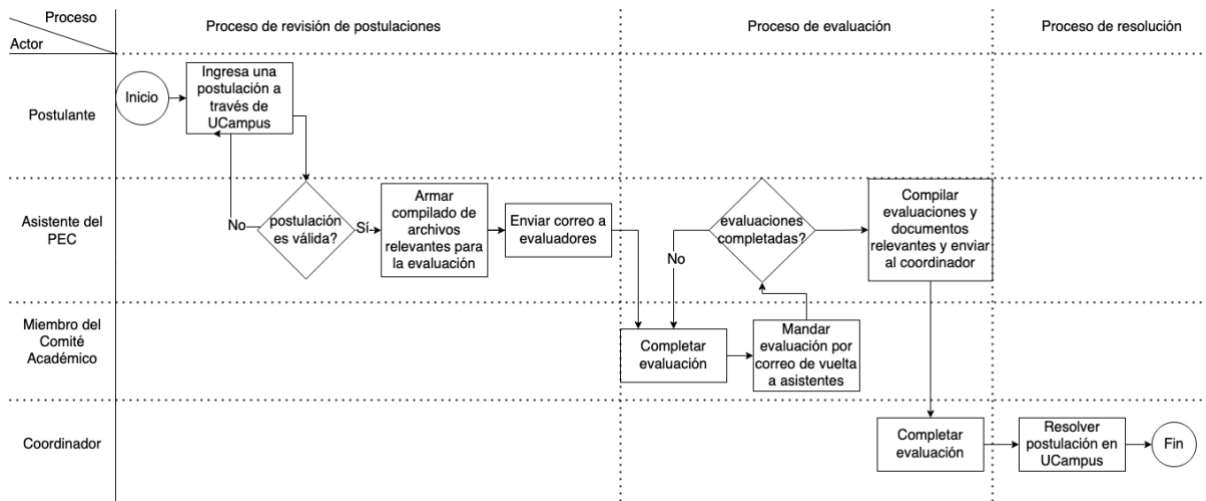


Figura 1. Workflow del proceso de evaluación de postulaciones.

Tal como se puede ver en la Figura 1, el proceso involucra los siguientes 7 pasos:

1. Un postulante ingresa una postulación a través de UCampus.
2. Los asistentes revisan la validez y completitud de la postulación. En caso de ser válida y estar completa, la información asociada a ésta se envía al Coordinador del Programa y a los Miembros del Comité Académico vía correo electrónico.
3. Los Miembros del Comité Académico evalúan cada postulación y emiten un juicio individual acerca de la aceptación o rechazo del postulante al programa, teniendo en cuenta las exigencias establecidas en la normativa vigente del MTI.
4. Cada miembro del Comité Académico envía por correo sus evaluaciones al asistente, quien las recolecta y envía al Coordinador del Programa el conjunto de todas las evaluaciones recibidas.
5. El Coordinador del Programa realiza su evaluación, revisa todas las opiniones de los miembros del Comité Académico y emite un veredicto acerca de la aceptación o rechazo del postulante.
6. El Coordinador del Programa registra la decisión en UCampus.

En el paso 2, la postulación puede ser devuelta al postulante para que corrija errores en caso de que existan, o para que éste envíe los documentos faltantes. Después de una revisión del proceso con el Coordinador del Programa y miembros del Comité Académico, se determinó que el diseño del mismo era apropiado. Las falencias del mismo radican principalmente en la artesanidad e informalidad con la que se realizan las actividades que son parte del mismo. Por lo tanto, la automatización de algunas de esas actividades parece ser suficiente para mejorar considerablemente la calidad y predictibilidad del proceso en términos de su duración.

2.3. Obtención de Datos desde la Plataforma UCampus

Respecto de la obtención de datos de las postulaciones (desde UCampus), a pesar de que exista una API publicada por su equipo desarrollador, ésta no incluye end-points que permiten obtener los datos de las postulaciones al magíster, lo cual va en contra del primer propósito: “permitir agilizar el proceso de postulaciones”.

Para resolver este problema, una solución que no involucra trabajo extra de los usuarios es hacer *scraping* de la página de postulaciones de UCampus. En ella se muestra una tabla con todas las postulaciones y un link de detalle a cada una de ellas, desde donde se puede obtener la información de todas las postulaciones abiertas del programa. Teniendo esta información, ya sea vía *scraping* o via una extensión a la API de UCampus, es posible construir un proceso autónomo que recupere la información de las postulaciones, sin que se requiera coordinación manual del proceso por parte de los Asistentes o del Coordinador del Programa.

Para agilizar el procesamiento de las postulaciones, el sistema enviará avisos a los usuarios relevantes cuando se requiere que tomen alguna acción, evitando así que estos tengan que estar pendientes de ingresar al sistema para ver si tienen alguna asignación sobre la cual responder. Un estudio realizado por la empresa Microsoft constató que este envío de avisos concientiza a las personas de que hay cosas pendientes, y aunque distrae, es preferido por su valor en la concientización [1].

Con respecto a la facilidad de recopilación y almacenamiento de los datos relevantes de los postulantes, usar una base de datos para dicho propósito es mejor que mantener los datos en las cuentas de correo de los participantes en el proceso. Si bien algunos datos están disponibles en UCampus (como los resultados de las postulaciones, por ejemplo), las opiniones de los Miembros del Comité Académico, entre otros datos, quedan en la cadena de correos asociada a una postulación. Usando una base de datos, estos datos

serán luego fáciles de acceder, por ejemplo, para realizar un análisis de la efectividad y eficiencia del procesamiento de las postulaciones.

El desarrollo de este sistema cuenta con los siguientes desafíos:

1. UCampus no entrega información estructurada respecto a postulaciones al magíster, y se pretende que la alimentación de datos sea realizada de forma automática.
2. El sistema debe automatizar el flujo de trabajo, invitando a los participantes (roles involucrados en el proceso) a completar sus tareas lo más rápido posible, manteniéndolos informados acerca de tareas que eventualmente ellos tengan pendientes de realizar.
3. El sistema debe garantizar que cada actor del proceso pueda hacer sólo lo que su rol permite, y nada más.

2.4. Análisis de Alternativas de Interacción con UCampus

En esta sección se describen las tareas realizadas en la búsqueda de una solución a la extracción automática de información desde UCampus. En particular se indican los desafíos asociados a los problemas que hay que resolver, las alternativas para abordarlos y las propuestas de solución.

2.4.1. Posible Extensión de la API de UCampus

Una de las primeras tareas realizadas fue comunicarse con el área de desarrollo de UCampus, para explorar la factibilidad de implementar un nuevo endpoint en su API, con el objetivo de permitir la extracción de los datos de las postulaciones desde una aplicación externa; en este caso, nuestro sistema de evaluación. Si bien la respuesta fue pronta y con buena disposición, no se dió curso a la solicitud por las siguientes razones:

- El formulario de ingreso de datos y el flujo de una postulación están en el sistema llamado “Workflow”, el cual está a cargo del ADI (Área de Infotecnologías)⁵, y por lo tanto, poco puede ayudar el Centro UCampus a la intervención de dicho sistema. Además, no hay una conexión directa entre el sistema de ADI y la plataforma UCampus, por lo que se requiere realizar bastante trabajo para poder contar con información de postulaciones a través de la API de UCampus.
- La respuesta desde ADI fue que los datos se encuentran encapsulados dentro de herramientas internas del sistema “Workflow”, lo cual dificulta la extracción de datos limpios desde su base de datos.

⁵ <http://ingenieria.uchile.cl/facultad/estructura/95223/area-de-infotecnologias>

En resumen, y aunque aún no está completamente cerrada la posibilidad de obtener los datos desde la fuente, se hace necesario explorar otras opciones, las cuales se detallan a continuación.

2.4.2. Extracción de Datos de UCampus Usando Scrapers

Scraping es una técnica que se usa para la extracción de datos no estructurados, ya sea a partir de imágenes, PDFs, o páginas web, entre otros. Al no poder obtener directamente los datos de postulaciones a través de una API (o una interfaz de acceso a datos similar), usar un scraper es una buena alternativa para recuperar dicha información. En este caso, los datos de las postulaciones se encuentran accesibles a través de una interfaz Web de la plataforma UCampus, siempre que se cuente con los permisos de acceso correspondientes. Este es el canal oficial tanto para postular como para gestionar las postulaciones.

El uso de scrapers no está exento de desafíos. Algunos de los más conocidos son los siguientes:

- Si el sitio web que representa la fuente de datos cambia su interfaz de usuario, entonces se debe adaptar el scraper para extraer información del nuevo *layout* de la página.
- Algunos sitios web (y este es el caso de UCampus) usan Javascript en su interfaz (versiones modernas). Esto hace que no sea trivial la extracción de los datos, pues el scraper debe esperar por eventos asíncronos para poder extraer datos. Las librerías tradicionales de scraping [6, 7] no cuentan con esa capacidad.

A pesar de lo planteado anteriormente, usar un scraper es realmente la única alternativa que existe hoy para poder extraer los datos de las postulaciones de forma automática. Con respecto a las alternativas para hacer el scraping, se analizaron las siguientes librerías, dado que están dentro de las más reconocidas:

- *Puppeteer* [3]: Librería Node.js que permite controlar una instancia de Chrome a través del protocolo DevTools.
- *Selenium WebDriver* [4]: Esta es una API con implementación en múltiples lenguajes, orientada al testing, que permite manejar un browser.

Las diferencias entre estas librerías radica principalmente en que Puppeteer controla instancias de Chrome, mientras que Selenium es una implementación para un navegador genérico. De acuerdo a las pruebas de concepto que se realizaron y dejaron disponibles en un repositorio [8], ambas cumplen el propósito de extraer datos desde UCampus sin

problemas. En el repositorio se muestran los resultados de extraer ciertos datos desde UCampus con la cuenta del autor, de acuerdo a la siguiente secuencia de pasos:

1. Ingresar a UCampus con usuario y contraseña.
2. Hacer clic en la pestaña *Tareas Iniciadas*.
3. Extraer links desde una tabla. Cada caso particular (postulación) corresponde a una “Solicitud de Inscripción con Excepción (SIEs)”.
4. Visitar todos los links disponibles y extraer sus datos.

Con el objetivo de mantener la estabilidad del scraper, en el sentido de que éste soporte la mayor cantidad de alternativas posibles para abordar los eventuales cambios en la interfaz de usuario, ambas opciones presentan APIs igualmente capaces. Por un lado, Selenium mantiene su última fecha de liberación (para Python) el 1 de noviembre de 2018, mientras que Puppeteer lanzó su última versión el 26 de febrero de 2021.

Aún cuando parezca que Selenium es una librería desactualizada, realmente no lo es. La fecha de liberación indicada anteriormente corresponde a la API de la librería. Sin embargo, Selenium funciona en base a *drivers*, que controlan el motor de un navegador, que puede ser Chromium o Firefox. La última versión para *chromedriver* es ChromeDriver 89.0.4389.23 [17], que corresponde a la versión actual de Chromium.

Para Puppeteer, no hay ningún driver requerido, sólo basta instalar la librería, por lo que se concluye que ambas herramientas son igualmente capaces para extraer datos de postulaciones desde UCampus.

Debido a la fragilidad que puede llegar a tener un scraper, y tomando en cuenta la cantidad de postulantes al MTI, resulta válido también explorar la posibilidad de que los datos sean ingresados manualmente al sistema por parte de sus usuarios; particularmente, por los funcionarios del PEC.

2.4.3. Carga Manual de Datos de Postulaciones

Si bien lo ideal es que los datos sean ingresados de forma automática al sistema, no hay que perder de vista que el objetivo principal de este trabajo es: *desarrollar un sistema Web que automatice el flujo de trabajo asociado al procesamiento de postulaciones al Magíster en Tecnologías de la Información*. Con eso en cuenta, la carga manual de datos se percibe más bien como un despropósito en este escenario.

Por otra parte, los scrapers anteriormente descritos fueron probados con resultados satisfactorios que, en una primera aproximación, demostraron que es posible extraer datos no estructurados desde UCampus, y confiar en los resultados de dicho proceso.

2.5. Formulario de Postulación al MTI

A continuación se muestra un formulario típico de postulación al MTI, implementado sobre la plataforma UCampus. Se han censurado los datos sensibles del postulante.

Postulación al Programa Magíster en Tecnologías de la Información (2021 Otoño)

Antecedentes Personales

Nombre	Sexo
RUT/Pasaporte	Nacionalidad
Fecha de Nacimiento	Email
Teléfono	Postulando a una beca
Dirección	

Antecedentes Académicos

Título/Grado	Universidad	País	Inicio	Termino

Solicitudes de Cartas de Recomendación (1)

N°	Nombre	Cargo
1		

Cartas de Recomendación Recibidas (1)

Las recomendaciones se reciben como archivos adjuntos en el email fcfm.escpostgrado+fcfm_postgrado+816266@ucampus.cl

N° Carta	Remitente	Fecha de Recepción
1		

Archivos Recibidos (5)

N°	Tipo de Archivo	Copia Digital
1	Certificado de Título o Grado	
2	Curriculum Vitae	
3	Carta Motivacional	
4	Certificado de Notas	
5	Programa de Estudio	

Figura 2. Formulario de postulación al Magíster en TI.

Como se puede ver en la Figura 2, una postulación completa consta de los siguientes antecedentes:

- Datos personales del postulante.
- Los antecedentes académicos del mismo: Título o grado, Universidad, País de la Universidad, Año de inicio y término de sus estudios.
- Solicitudes de cartas de recomendación, hechas por el postulante. Éstas se muestran por el nombre de cada persona a la que se le solicitó una carta de recomendación, su correo electrónico y su cargo.
- Cartas de recomendación recibidas. Éstas muestran el archivo correspondiente a la carta de recomendación, el nombre de la persona que lo envió, su email y la fecha de recepción.
- Los 5 archivos mandatorios de cada postulación: Certificado de título o grado, currículum vitae, carta motivacional, certificado de notas y el programa de estudio.

Algunas consideraciones a destacar del formato de los datos:

- La fecha de nacimiento de un alumno sólo está presente para usuarios antiguos de UCampus. Usualmente ese campo sólo muestra la edad.
- La nacionalidad, de la misma forma, no suele estar presente en la postulación.
- La postulación puede estar a medio hacer. Vale decir, puede ser que la postulación no contenga las cartas de recomendación o documentos esenciales, y aún así el sistema permite que la misma sea procesada como una postulación.

3. Descripción de la Solución

En este capítulo se presenta la descripción del sistema desarrollado para dar solución al problema planteado.

3.1. Descripción del Proceso Automatizado

Habiendo entendido cómo funciona el proceso actual, se puede concebir cómo automatizar el proceso de postulaciones al MTI. En efecto, el proceso parte cuando un postulante realiza su postulación en UCampus; ésta se mantiene ahí hasta que es resuelta en la misma plataforma. Sin embargo, el proceso interno de resolución del programa MTI no está actualmente tomado en cuenta por UCampus.

Para automatizarlo, se debe partir por extraer los datos desde UCampus. Este objetivo se logra haciendo un scraper que recorra el sitio, extraiga los datos de postulaciones y los guarde en una base de datos. Teniendo estos datos, se puede construir un sistema que asista en el workflow de postulaciones.

Luego, los procesos de validación, evaluación y resolución de la postulación de cada postulante se realizan actualmente a través de una seguidilla de correos y recopilación manual de información. Para automatizar este proceso se desarrolló un sistema que administra el workflow interno de resolución de una postulación según los requisitos del MTI.

El alcance del sistema desarrollado termina con la evaluación del coordinador, quien decide si se acepta o rechaza la postulación. Para terminar el workflow requerido por la Escuela de Postgrado, es necesario que dicho proceso se haga a través de la plataforma UCampus. Por esa razón, el sistema desarrollado redirige al coordinador a la postulación en UCampus, para que sea resuelta según el proceso definido por la Escuela.

3.2. Principales Requisitos de la Solución

A continuación se presentan los principales requisitos de la solución, los cuales sirvieron de guía para el desarrollo de la misma.

- Crear un scraper que extraiga datos de postulaciones, tanto nuevas como ya existentes, desde la plataforma UCampus.
- Mantener al scraper corriendo periódicamente, de modo que agregue las postulaciones nuevas y actualice las postulaciones que tienen datos nuevos.
- Crear una interfaz de usuario que permita validar las postulaciones e iniciar el proceso de evaluación de éstas.

- Crear una interfaz de usuario que permita evaluar las postulaciones por parte de los miembros del Comité Académico del programa y de su Coordinador.
- Crear una interfaz de usuario que permita marcar la resolución de cada postulación. Se debe redirigir a UCampus para que la postulación siga su proceso normal en dicha plataforma.
- Desarrollar un módulo de alertas por correo, que notifique a los participantes respecto a etapas del workflow que tienen tareas que resolver por parte de ellos.

3.3. Perfiles de Usuarios

En base a los actores del proceso presentados en la Sección 2.1, se describe a continuación en qué parte del proceso se involucra cada rol.

3.3.1 Postulante

El postulante no participa activamente del sistema desarrollado. La razón de esto es porque el objetivo del sistema es automatizar el proceso de evaluación de las postulaciones, aunque la postulación misma se realiza en UCampus. Este proceso de evaluación es un workflow externo, que se lleva a cabo a través de UCampus. Por lo tanto, el postulante sólo recibe el resultado de su postulación por parte de la Escuela.

3.3.2 Miembro del Comité Académico del Programa

Los miembros del Comité Académico son los encargados de evaluar las postulaciones. Cuando una postulación es válida (es decir, es correcta y está completa), los miembros del Comité dan su opinión acerca del potencial del postulante, su pregrado, su experiencia laboral, la carta motivacional enviada, las cartas de recomendación recibidas y otras observaciones adicionales. Con eso, cada miembro emite una recomendación de aceptación o rechazo del postulante en una escala de 4 puntos: aceptar, posible aceptación, posible rechazo, o rechazo. En el Anexo B se presenta el formulario (en Word) actualmente usado para evaluar estas postulaciones.

Típicamente una postulación tendrá 4 evaluaciones, una por cada miembro del Comité, pero no necesariamente en todos los casos. En aquellos casos donde hay 3 miembros de acuerdo respecto a la aceptación o rechazo de un candidato, el Coordinador del programa puede realizar la resolución sin esperar a que arribe la última evaluación. Por otra parte, cuando una vez que todos los miembros del Comité han emitido sus recomendaciones sobre un postulante, la postulación pasa automáticamente a evaluación por parte del Coordinador del programa.

3.3.3 Coordinador de Magíster / Presidente del Comité Académico

Cuando todos los miembros del Comité emiten su recomendación, el Coordinador hará lo mismo teniendo en cuenta las recomendaciones hechas anteriormente. Esto se hace para que quede constancia de la opinión del Coordinador.

En este paso, el Coordinador decide si se acepta o rechaza la postulación. Esta resolución debe hacerse en la plataforma UCampus, pues ese es el conducto formal para la Escuela de Postgrado. Debido a eso, el sistema de evaluación desarrollado abre una ventana nueva con la resolución de la postulación, para facilitarle la labor al coordinador.

3.3.4 Asistente / Funcionario del PEC

Las postulaciones en UCampus son incrementales. Esto es, pueden estar almacenadas en forma incompleta, hasta que eventualmente el postulante rellena todos los datos necesarios para su evaluación, o bien después de un tiempo sin completarse, el programa decide darle de baja.

Los asistentes o funcionarios del PEC cumplen dos funciones. La primera, es validar los datos de cada postulación. Vale decir, a pesar de que una postulación tenga todos los datos necesarios, el sistema no puede verificar automáticamente que los documentos contengan la información correcta. Este trabajo lo hacen los asistentes.

La segunda función que cumplen es monitorear las evaluaciones. Es decir, si una evaluación lleva demasiado tiempo sin completar, los asistentes les recuerdan a los miembros del Comité Académico que deben realizar la evaluación.

4. Diseño del Sistema

En este capítulo se presenta el diseño del sistema, que incluye el modelo de dominio abordado, así como la arquitectura y el modelo de datos de la solución.

4.1. Modelo de Dominio

La Figura 3 muestra el modelo de dominio abordado, el cual ha sido especificado utilizando UML. Allí se puede ver que la postulación es el componente central del modelo. Ésta mantiene su estado, el postulante al cual está asociada, la documentación adjunta, y todas las evaluaciones asociadas a ella con su correspondiente evaluador. Uno de estos evaluadores es el Coordinador del Programa, el cual emite una resolución acerca de la aceptación o rechazo de la postulación al MTI.

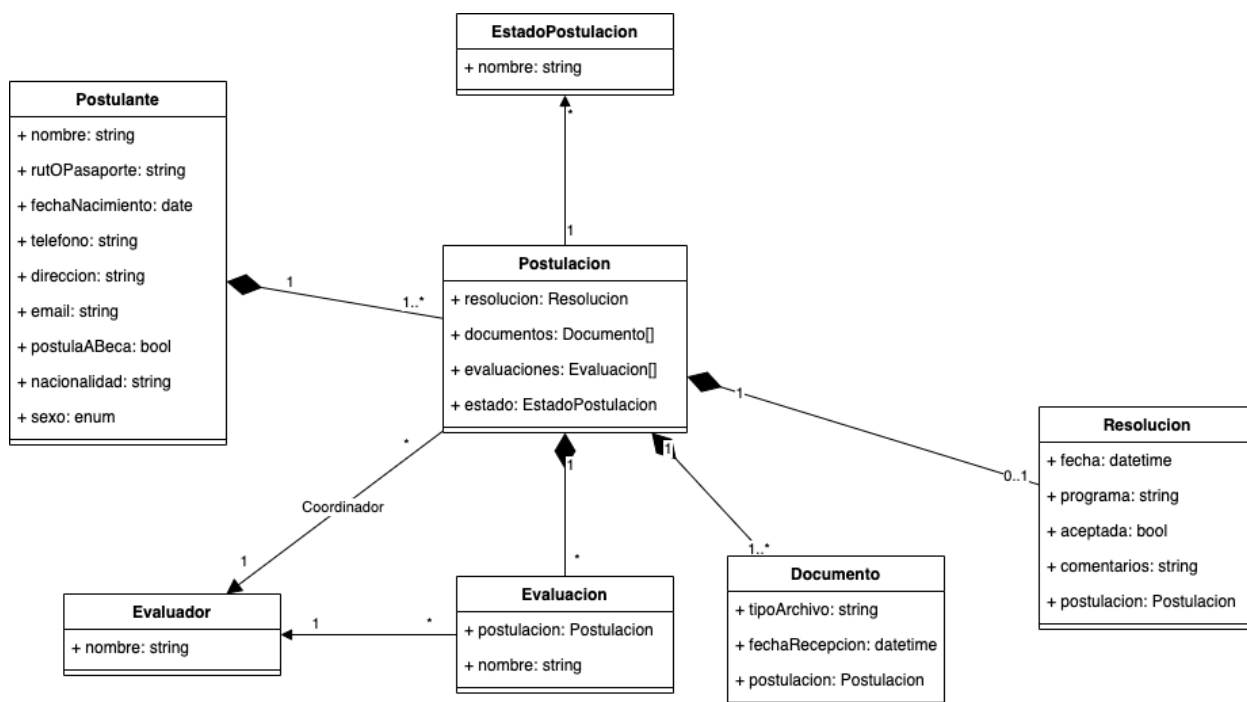


Figura 3. Modelo de dominio del sistema.

4.2. Arquitectura del Sistema

La Figura 4 muestra el modelo de contexto de la aplicación, el cual representa el nivel de abstracción más alto en la representación de arquitecturas según el modelo C4 [9]. Como se puede ver, los usuarios interactúan con el sistema de evaluación de postulaciones al MTI, el cual extrae los datos desde la aplicación web de UCampus.

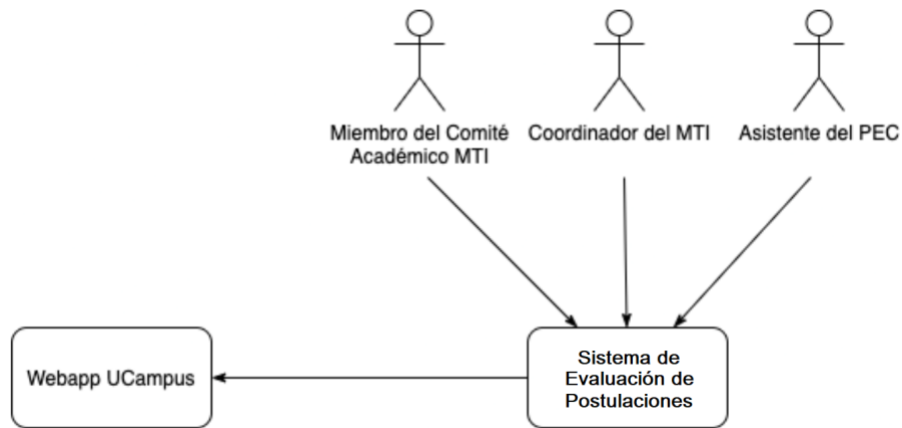


Figura 4. Arquitectura general de la aplicación (Modelo de Contexto).

La Figura 5 muestra cómo está estructurado el sistema en términos de sus macro-componentes y el repositorio de datos. Particularmente, los usuarios del sistema interactúan con una aplicación front-end, la cual contiene todas las interfaces de usuario del sistema. Ésta no contiene la lógica de negocio, pues su rol es presentar la interfaz de usuario, pedir datos al back-end, y mostrarlos al usuario.

Por otra parte, el back-end del sistema es el encargado de mantener la lógica de la aplicación. En él se describen todos los procesos de negocio, y es el encargado de la comunicación entre la interfaz de usuario y la base de datos. El back-end además se comunica con la aplicación web de UCampus y extrae los datos desde la página Web de postulaciones. Este componente es uno de los puntos más críticos, pues es el que mantiene toda la lógica del sistema, y por lo tanto, es el que ayuda a optimizar el proceso de postulaciones.

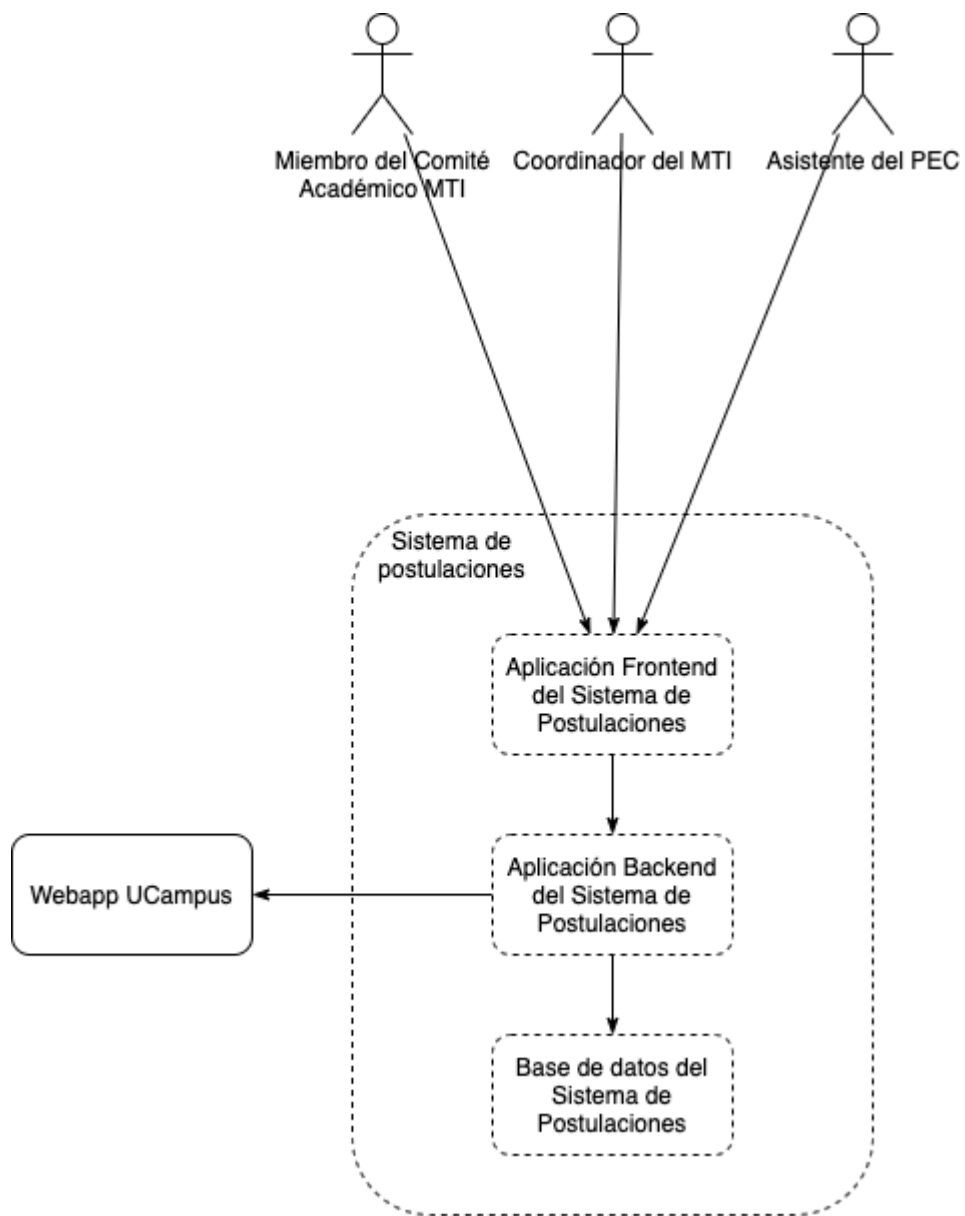


Figura 5. Descomposición en componentes del Sistema de Evaluación de Postulaciones.

Finalmente, la Figura 6 muestra la descomposición del back-end en sus principales componentes. Éste se divide en 4 componentes, los cuales se describen a continuación.

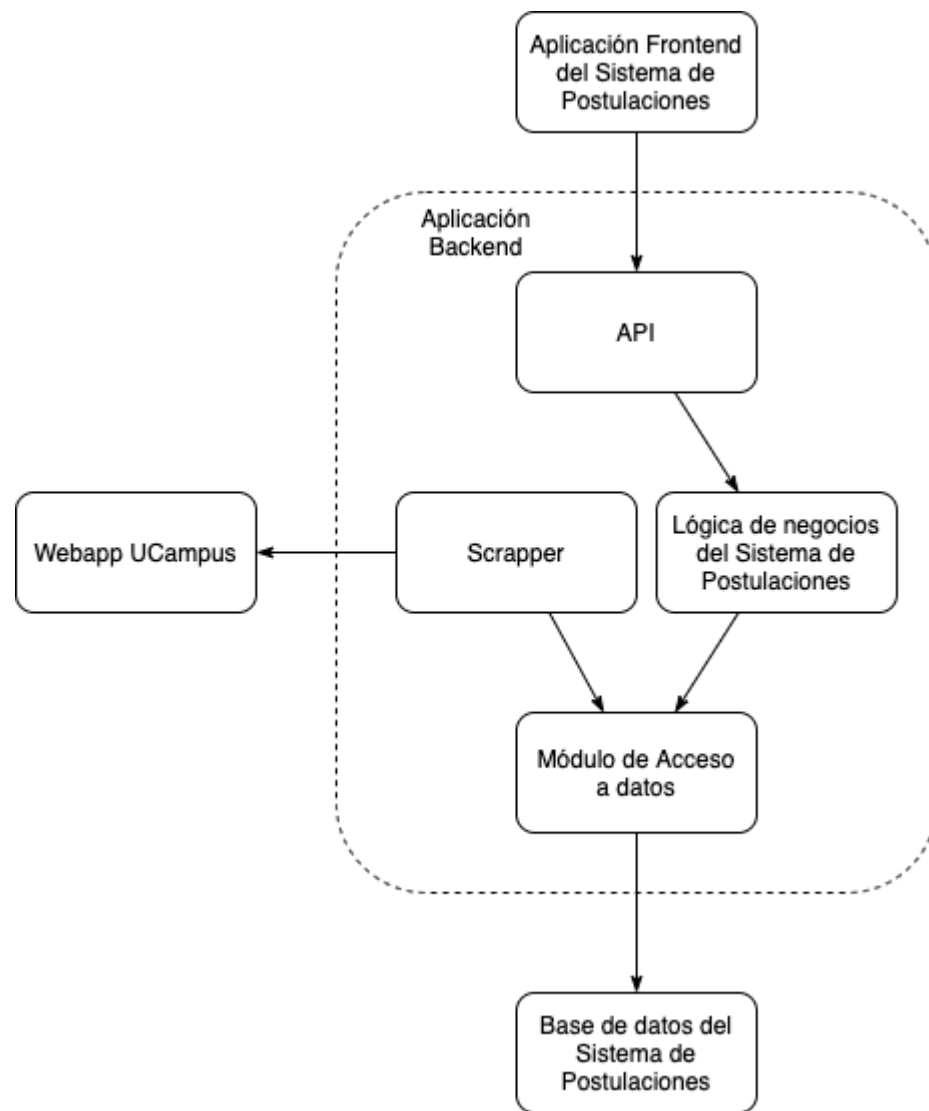


Figura 6. Descomposición en componentes del back-end del sistema (sólo se muestran los componentes relevantes).

La **API** es un módulo HTTP encargado de disponibilizar los datos de la aplicación para que puedan ser consumidos por la aplicación front-end. Este módulo no contiene ninguna lógica de negocios; sólo sirve como medio de transporte de datos desde y hacia la lógica de la aplicación. Por supuesto, este módulo también recibe datos desde el front-end, y en dicho caso, es el encargado de transportar esos datos hacia la capa de lógica de la aplicación.

Luego, el módulo de **lógica de negocios** es el encargado de implementar el proceso y las interacciones de sus actores. En este módulo se puede encontrar la lógica que procesa las postulaciones nuevas, crea las evaluaciones y recibe los inputs de los actores del sistema.

El módulo de **acceso a datos** es el encargado de comunicarse con la base de datos y persistir los datos que le llegan desde la lógica de negocio o desde el scraper. Por último, el **scraper** es el módulo encargado de consumir datos desde la página Web de UCampus, y enviarlos para que se almacenen en la base de datos. Para ello, los datos que procesa el scraper son entregados como input a la capa de acceso a datos. Vale la pena mencionar que los diagramas antes presentados seguirán siendo válidos en el caso de que UCampus entregue los datos de postulaciones vía una API para ser consumida por el sistema desarrollado en esta memoria. La única cosa que cambiaría sería el uso de la API, como reemplazo al actual uso del scraper.

Lo anterior porque la arquitectura presentada permite independizar la forma de extraer datos, ya sea que esto se haga a través de una API o a través de scrapers. El proceso de extracción de datos está encapsulado un único módulo del sistema, lo cual permite reemplazarlo en caso de que a futuro aparezcan mejores opciones para realizar la alimentación de datos.

4.3. Modelo de Datos

A continuación se muestra el modelo de datos del sistema, el cual adhiere al modelo de dominio mostrado en la Figura 3.

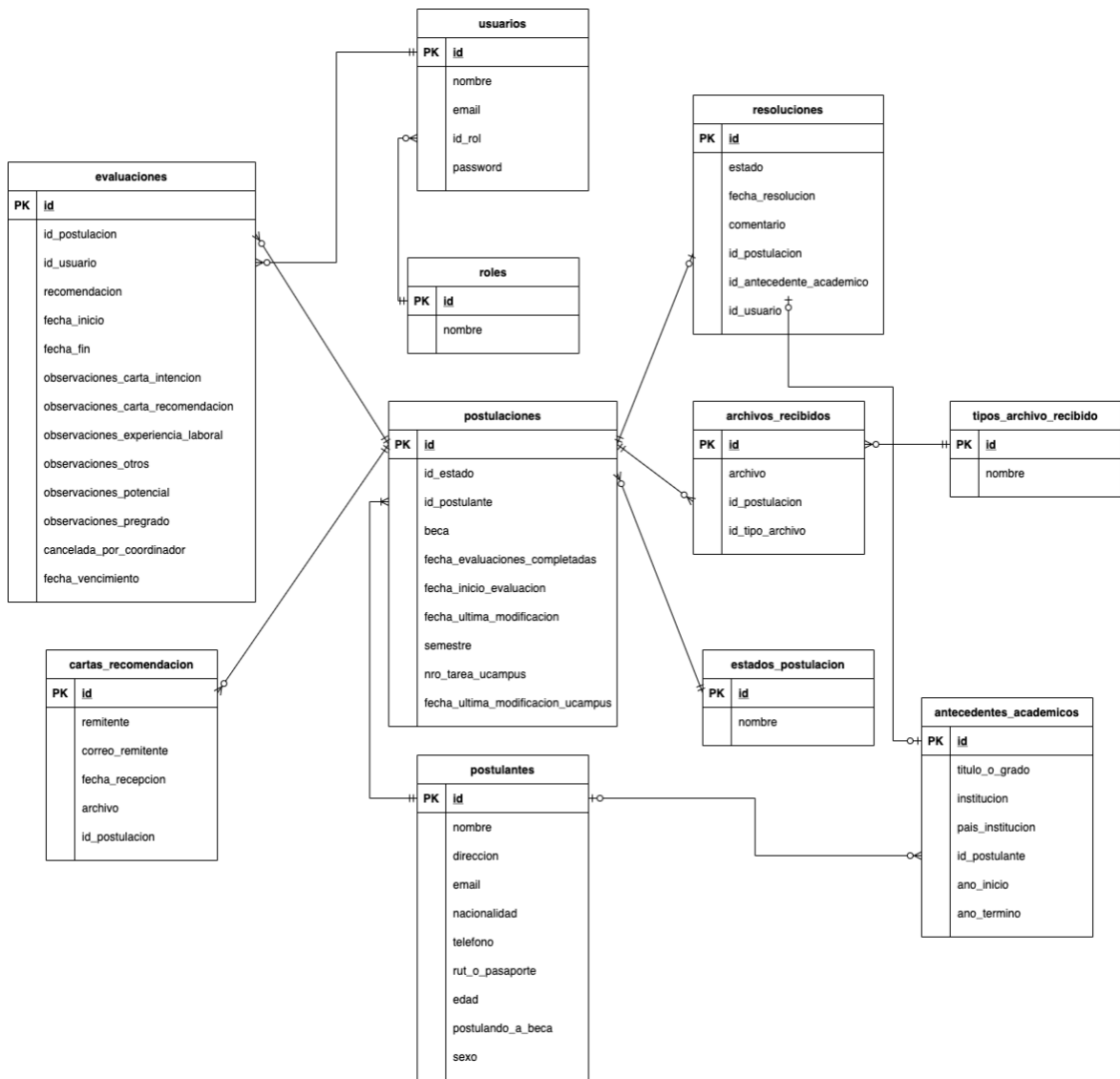


Figura 7. Modelo de datos del sistema.

En cada entidad de datos se puede ver los atributos de las mismas, cuyos nombres son casi auto-explicativos. Tal como se ha mencionado antes, las postulaciones son el centro neurálgico del sistema. Éstas pertenecen a un postulante y tienen documentación asociada (cartas_recomendación y archivos_recibidos, en la Figura 7). Además tienen evaluaciones, un estado, y eventualmente resoluciones acerca de ellas, tal como se indica en la figura anterior.

5. Implementación de la Solución

La implementación de la solución cuenta con distintos módulos que usan distintas tecnologías y potencialmente distintos lenguajes. Primero, para los módulos del sistema se detallan las tecnologías escogidas, y luego se presentan las interfaces creadas para los distintos actores (perfiles de usuario) del proceso.

5.1. Tecnologías Escogidas para la Implementación

En este apartado se describen los componentes críticos del sistema implementado, y cuáles fueron las tecnologías usadas para tal propósito en cada caso. Antes, vale la pena explicar los requerimientos de implementación de este sistema. Para ello, hay que tener muy claro qué capacidades tiene actualmente la aplicación, cuál es su perspectiva de evolución a futuro, y cómo será la operación de la misma. Estos tres puntos son claves para tomar cualquier decisión con respecto a las tecnologías que se escojan para implementar el sistema.

Primero que todo, este sistema muestra que se puede obtener datos desde UCampus y usarlos para automatizar *de forma interna*⁶ el proceso de evaluación de las postulaciones. Luego, el futuro de este sistema puede tomar dos rumbos: 1) los datos se extraerán desde una API que exponga UCampus, o 2) la plataforma UCampus estandariza el formulario de postulación a todos los programas de la Facultad (hoy en día dicho formulario no está estandarizado).

La segunda alternativa es poco viable en el corto plazo, pues requiere que todos los programas de la Facultad se pongan de acuerdo acerca de qué información debe aportar el postulante a la hora de postular a uno de ellos. Esto parece ser difícil de alcanzar, pues muchos de estos programas tienen naturaleza distinta (algunos tienen un perfil profesional y otros científico), y distintos niveles de exigencia (por ejemplo, para magíster y doctorado).

Considerando estos aspectos, el sistema de evaluación de postulaciones debe estar preparado para el futuro de varias formas:

- Debe ser extensible, en el sentido de que debe permitir cambiar el método de extracción de datos de las postulaciones.
- Debe ser altamente mantenible, pues hay componentes cuya actual implementación depende de componentes o condiciones externas, que podrían

⁶ Es decir, fuera de UCampus

cambiar en un futuro inmediato. A pesar de eso el sistema debería poder adaptarse en forma rápida para seguir brindando el servicio.

- La operación de este sistema debe presentar la menor cantidad de impedimentos, y bloqueos técnicos posibles, tanto para los desarrolladores como para los administradores de sistemas.

A continuación se explican las tecnologías que se utilizaron para implementar los macro-componentes de este sistema.

5.1.1 Back-end

El back-end es la piedra angular de esta aplicación. Dentro de él, funcionan el scraper, la conexión a la base de datos y la disponibilización de los datos de forma segura. Para implementar el back-end, la primera restricción que presenta es que existen dos maneras viables para extraer los datos de las postulaciones desde UCampus; cada una con su propio entorno de desarrollo. Una de ellas usa Node.js, y otra está en Python.

Para tomar esta decisión, la Stack Overflow Developer Survey 2020 [10] reveló que Python es el tercer lenguaje más amado por desarrolladores, sólo superado por TypeScript y Rust. Python es además el lenguaje que más desarrolladores buscan aprender.

A pesar de lo antes mencionado, sigue estando la opción de usar Puppeteer [3] con TypeScript para implementar el back-end. Sin embargo, teniendo en cuenta el perfil de los potenciales desarrolladores que tendrá el sistema (memoristas del DCC), el uso de Python parece ser la opción más apropiada para implementar el back-end del sistema. Python es un lenguaje que cualquier estudiante de una Universidad chilena conoce, al contrario de TypeScript, que si bien es un lenguaje con mucha popularidad en la industria, no lo es tanto en el mundo académico.

Por las razones mencionadas anteriormente, el lenguaje escogido para escribir el back-end es Python. Dado eso, queda por decidir cómo conectarse a la base de datos, y cómo exponer los datos de las postulaciones de forma segura.

Base de datos

La encuesta a desarrolladores mencionada anteriormente (de Stack Overflow) mostró que PostgreSQL es la base de datos persistente más amada por estas personas [10]. Le siguen, en orden, Elastic Search y MongoDB. Por otro lado, PostgreSQL es la segunda base de datos que los desarrolladores buscan aprender, sólo precedida por MongoDB.

La base de datos rankeada como #1 en la encuesta es Redis [12]; sin embargo, ésta no es una opción puesto que es una base de datos en memoria.

Con respecto a bases de datos SQL (PostgreSQL) y NoSQL (Elastic Search y MongoDB), la naturaleza estructurada y relacional de los datos existentes hacen que inclinarse por PostgreSQL sea una buena idea. Otra razón para ello es que potenciales desarrolladores de este sistema muy probablemente no estarán muy familiarizados con bases de datos NoSQL, pues son menos abordadas en cursos de bases de datos a nivel de pregrado. Por las razones anteriormente expuestas, se decidió que los datos de este sistema se almacenen en una base de datos PostgreSQL.

Servidor HTTP

Para el servidor HTTP se deben tomar en cuenta las siguiente consideraciones:

- Rendimiento (performance) del servidor, medido en cantidad de respuestas por segundo.
- Curva de aprendizaje para nuevos desarrolladores.

Para evaluar el rendimiento potencial del servidor se analizó la información disponible en the benchmarker [11], el cual consta de una serie de repositorios dedicados a encontrar *benchmarks* de servidores HTTP en frameworks web. Para cada framework se mide la cantidad de respuestas por segundo, según la cantidad de *threads* concurrentes que intentan acceder al servidor. En la Tabla 2 se muestra un resumen de los resultados obtenidos por los frameworks más conocidos.

Tabla 2. Rendimiento de frameworks de Python [11]

#	Lenguaje	Framework	Req/s (64)	Req/s (128)	Req/s (256)
58	python (3.9)	falcon (2.0)	74.256,04	81.538,76	82.897,69
88	python (3.9)	pyramid (2.0)	51.298,30	56.850,32	57.128,55
100	python (3.9)	asgineer (0.8)	44.745,54	51.318,30	52.105,43
104	python (3.9)	bottle (0.12)	39.690,41	42.590,65	44.329,21
107	python (3.9)	emmett (2.2)	35.983,95	41.270,86	42.295,95
115	python (3.9)	apidaora (0.28)	34.119,60	38.263,87	38.707,73
116	python (3.9)	hug (2.6)	34.040,91	35.909,39	53.828,77

125	python (3.9)	<u>blacksheep (1.0)</u>	28.887,22	33.204,44	34.356,25
129	python (3.9)	<u>index.py (0.16)</u>	27.273,81	29.282,48	30.258,63
131	python (3.9)	<u>starlette (0.14)</u>	26.693,66	31.709,52	31.302,99
134	python (3.9)	<u>responder (2.0)</u>	26.092,65	31.092,58	32.175,53
135	python (3.9)	<u>clastic (19.9)</u>	25.651,88	28.936,99	28.781,05
136	python (3.9)	<u>sanic (21.3)</u>	25.446,08	28.833,98	29.194,50
145	python (3.9)	<u>molten (1.0)</u>	18.055,01	21.858,74	21.906,60
146	python (3.9)	<u>aiohttp (3.7)</u>	17.837,07	23.359,98	24.128,26
149	python (3.9)	<u>fastapi (0.63)</u>	16.701,72	22.402,72	22.042,29
164	python (3.9)	<u>flask (1.1)</u>	12.901,82	16.427,23	16.521,94
176	python (3.9)	<u>cherrypy (18.6)</u>	9.328,09	9.396,33	8.832,55
177	python (3.9)	<u>guillotina (6.2)</u>	9.152,33	8.843,65	8.742,31
181	python (3.9)	<u>quart (0.14)</u>	7.571,28	7.501,08	6.888,88
183	python (3.9)	<u>tonberry (0.2)</u>	7.363,12	6.948,61	6.344,40
191	python (3.9)	<u>django (3.1)</u>	5.918,73	6.572,13	6.150,29
193	python (3.9)	<u>tornado (6.1)</u>	5.722,03	5.728,67	5.624,07
210	python (3.9)	<u>masonite (3.0)</u>	2.485,30	2.477,63	2.477,29
217	python (3.9)	<u>cyclone (1.3)</u>	1.597,57	1.591,42	1.577,40
218	python (3.9)	<u>klein (20.6)</u>	1.501,11	1.538,24	1.513,71
220	python (3.9)	<u>nameko (2.13)</u>	1.213,25	1.164,90	1.142,60
222	python (3.9)	<u>django-ninja (0.11)</u>	1.065,37	1.478,17	1.496,42

La Tabla 2 muestra los resultados para herramientas python que sirven para implementar el back-end del sistema. Las columnas de resultados (que parten con *Req/s*) indican cuántas peticiones por segundo puede responder el servidor en promedio cuando hay una cierta cantidad de usuarios haciendo peticiones de forma concurrente (el número entre los paréntesis).

En todo caso, hay que tener en cuenta que muchas de estas herramientas suelen ser experimentales, y otras son tan minimales que no es viable su uso para sistemas como el reportado en esta memoria.

Tomando en cuenta la popularidad de los paquetes, en término de cantidad de estrellas en GitHub, los contendores son:

- FastAPI (29.2k estrellas)
- Django (56.5k estrellas)
- Flask (54.4k estrellas)

FastAPI es el más nuevo de los tres y cuenta con soporte para asyncio [14], con tutoriales cortos donde se explica la totalidad de sus capacidades. Django es el más popular, con una gran cantidad de capacidades (tan grande de hecho, que no se hacen necesarias todas ellas). Finalmente, Flask es prácticamente FastAPI, pero sin soporte para asyncio, ni inyección de dependencias.

Considerando lo anterior, se decidió utilizar usar FastAPI tomando en cuenta diversos aspectos:

- Su gran popularidad pese a su poco tiempo de vida, comparado con Flask y Django.
- Soporta asyncio en forma nativa. El uso de asyncio hace que pueda manejar una mayor cantidad de peticiones por segundo.
- La barrera de entrada para usar este framework es mucho más pequeña que Django y es comparable a Flask. Por lo tanto, para el futuro de este proyecto, los nuevos desarrolladores no tendrán que entender la filosofía de un framework completo, como Django, sino que simplemente les bastará entender cómo funcionan los módulos de Python.

5.1.2 Scraper

Con respecto al scraper, ya se introdujo anteriormente las alternativas evaluadas, éstas eran: Puppeteer [3] y Selenium [4]. Al tomar en cuenta que Selenium tiene bindings para Python, se hace evidente usarlo en vez de Puppeteer. De nuevo, la mantenibilidad juega un rol importante en esta decisión.

El flujo que realiza el scraper para extraer datos de postulaciones es el siguiente:

1. Entra a UCampus con usuario y contraseña, provistos como configuración del sistema.
2. Ingresa directamente a la URL: <https://ucampus.uchile.cl/m/workflow/tareas?tipo=resolver>
3. Busca la tabla que tenga como cabecera el título "Postulación Postgrado"
4. Por cada uno de los postulantes:

- a. Hace clic en el link “Resolver”
 - b. Recolecta los datos relevantes de postulaciones: antecedentes personales, antecedentes académicos, cartas de recomendación y archivos recibidos.
 - c. Guarda los datos de cada postulante en la base de datos interna del sistema de postulaciones, descargando archivos cuando es necesario.
 - d. Vuelve a la página que contiene la tabla con los postulantes y hace clic en el candado (éste queda bloqueado al hacer clic en “Resolver”) sólo si es que se encuentra bloqueado.
5. Termina el proceso.

Para encontrar cada dato de forma precisa, se utiliza una combinación de XPath con selectores CSS. A modo de ejemplo, para encontrar la tabla de Antecedentes Académicos de un postulante, primero se realiza la siguiente consulta XPath:

```
//h2[contains(., 'Antecedentes Académicos')]/following-sibling::div
```

Ello entrega un elemento *div*, que contiene la tabla de antecedentes académicos. Para ello, busca un elemento *h2* que contenga el texto “Antecedentes Académicos” y selecciona el siguiente nodo hermano en el DOM que sea un elemento *div*. Luego, con selectores CSS se puede encontrar los antecedentes académicos para iterar sobre ellos. Las filas de la tabla se encuentran con el siguiente selector CSS:

```
tbody tr
```

Dicha consulta entrega todas las filas de la tabla que no corresponden a la cabecera de ésta. Es decir, *se omite la primera fila de la tabla*. Con estas filas, basta iterar sobre cada celda usando el selector CSS *td*. Este proceso es ilustrado en la Figura 8.

Antecedentes Académicos

Título/Grado	Universidad	País	Inicio	Termino
[Faded text]	[Faded text]	[Faded text]	[Faded text]	[Faded text]
[Faded text]	[Faded text]	[Faded text]	[Faded text]	[Faded text]
[Faded text]	[Faded text]	[Faded text]	[Faded text]	[Faded text]

Figura 8. Selección de datos relevantes de postulaciones por el scraper.

La Figura 8 muestra en color morado lo que encuentra la consulta XPath, y en celeste se muestra las filas encontradas por el selector CSS.

5.1.3 Deployment

Para decidir cómo hacer el deployment de la aplicación, se tuvo en cuenta que el sistema involucra, entre otros:

1. Un módulo de scraping, que requiere ejecutables en el sistema operativo host.
2. Una API.
3. Ejecución continua del módulo de scraping.

La instalación de un sistema con tales características no es trivial, pues no sólo se necesita de las dependencias naturales de Python, sino además componentes del sistema operativo para que funcione, en particular, para que Selenium funcione, se requiere que el binario de *chromedriver* [17] esté instalado en el sistema operativo.

Para automatizar esa tarea y hacer el deployment de esta aplicación un proceso confiable, se utilizó Docker [13], herramienta que permite abstraer tales restricciones y hace del deployment un proceso con bajo riesgo, pues no importa el ambiente en el que se ejecuta el proyecto. En ese sentido, sólo basta que funcione en Docker.

5.1.4 Front-end

Con respecto al front-end, los dos lenguajes más populares para implementar un sitio web con sus interfaces son JavaScript y TypeScript. Existen otras opciones que simplemente se descartan por baja popularidad, y por lo tanto, una potencial baja capacidad para darle mantenimiento al sistema.

El Developer Survey 2020 de Stack Overflow [11] reveló que, de los frameworks web más apreciados por los desarrolladores, React [15] está en la posición 2, mientras que Vue.js [16] está en la posición 3; ambos superados por ASP.NET Core, que se descarta usar en este proyecto simplemente porque requiere que el servidor sea escrito en C#, lenguaje que no es opción para este proyecto.

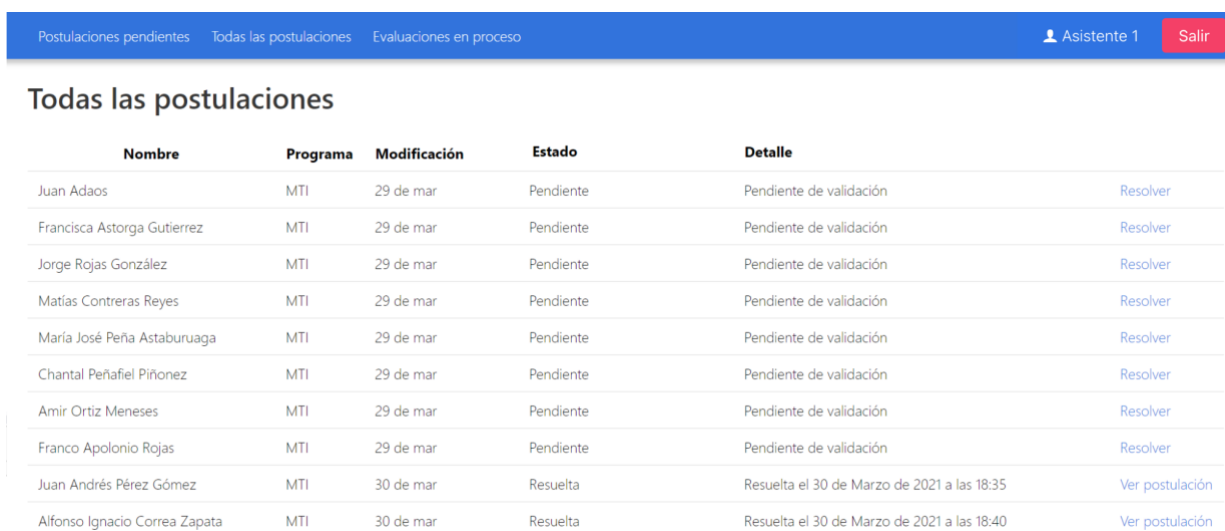
React [15] es el framework JavaScript más popular de front-end en el mercado, y es elegido para el desarrollo de esta aplicación precisamente por eso. Como en este proyecto se prioriza la mantenibilidad, React es probablemente la mejor opción. Vue [16] es el segundo framework más popular. Sin embargo, al momento del comienzo de la implementación de este sistema, Vue se encontraba en transición entre la versión 2 a la 3. Por lo tanto, elegirlo habría implicado trabajar con APIs inestables, o bien con APIs

obsoletas. Por las razones anteriormente expuestas, React es la opción elegida para este proyecto.

5.2. Interfaces de Usuario

En esta sección se presentan las principales interfaces de usuario del sistema. El propósito de esto es ilustrar cómo se ayuda a los diferentes tipos de usuario, a cumplir con sus tareas. A continuación se muestran las principales interfaces para cada usuario, es decir, aquellas que son críticas para el cumplimiento de las funciones asignadas a cada rol.

En primer lugar se muestra la interfaz principal del sistema, a la cual se accede luego del proceso de autenticación del usuario (Figura 9). En este caso el usuario está logueado con el rol de asistente. Es importante aclarar que los datos mostrados en estas interfaces son ficticios.



The screenshot shows the main interface of the system. At the top, there is a blue navigation bar with three menu items: "Postulaciones pendientes", "Todas las postulaciones", and "Evaluaciones en proceso". On the right side of the bar, there is a user profile icon labeled "Asistente 1" and a red "Salir" button. Below the navigation bar, the title "Todas las postulaciones" is displayed. The main content area contains a table with the following data:

Nombre	Programa	Modificación	Estado	Detalle
Juan Adaos	MTI	29 de mar	Pendiente	Pendiente de validación Resolver
Francisca Astorga Gutierrez	MTI	29 de mar	Pendiente	Pendiente de validación Resolver
Jorge Rojas González	MTI	29 de mar	Pendiente	Pendiente de validación Resolver
Matías Contreras Reyes	MTI	29 de mar	Pendiente	Pendiente de validación Resolver
María José Peña Astaburuaga	MTI	29 de mar	Pendiente	Pendiente de validación Resolver
Chantal Peñafiel Piñonez	MTI	29 de mar	Pendiente	Pendiente de validación Resolver
Amir Ortiz Meneses	MTI	29 de mar	Pendiente	Pendiente de validación Resolver
Franco Apolonio Rojas	MTI	29 de mar	Pendiente	Pendiente de validación Resolver
Juan Andrés Pérez Gómez	MTI	30 de mar	Resuelta	Resuelta el 30 de Marzo de 2021 a las 18:35 Ver postulación
Alfonso Ignacio Correa Zapata	MTI	30 de mar	Resuelta	Resuelta el 30 de Marzo de 2021 a las 18:40 Ver postulación

Figura 9. Interfaz principal del sistema.

El menú principal tiene tres opciones: postulaciones pendientes, todas las postulaciones y postulaciones en proceso. La semántica de cada categoría varía dependiendo del perfil de usuario que accede a la información. En el caso de la Figura 9, la información que se muestra corresponde a “todas las postulaciones”, la cual incluye las “pendientes de evaluación”, las “en proceso” y las “resueltas”. A continuación se explican las principales interfaces para cada rol.

5.2.1. Interfaces de Usuario para el Asistente

En el caso del usuario asistente, las postulaciones pendientes corresponden a aquellas que aún no han sido mandadas a evaluar (Figura 10), probablemente porque aún no tienen todos los documentos que se requiere para poder procesarlas. Para verificar esto, el asistente debe clicar en el link “Resolver” de la postulación que quiere revisar. Una vez hecho eso, accede a la postulación.

Nombre	Programa	Estado	Detalle
Juan Adaos	MTI	Pendiente	Pendiente de validación Resolver
Francisca Astorga Gutierrez	MTI	Pendiente	Pendiente de validación Resolver
Jorge Rojas González	MTI	Pendiente	Pendiente de validación Resolver
Matías Contreras Reyes	MTI	Pendiente	Pendiente de validación Resolver
María José Peña Astaburuaga	MTI	Pendiente	Pendiente de validación Resolver
Chantal Peñafiel Piñonez	MTI	Pendiente	Pendiente de validación Resolver
Amir Ortiz Meneses	MTI	Pendiente	Pendiente de validación Resolver
Franco Apolonio Rojas	MTI	Pendiente	Pendiente de validación Resolver

Figura 10. Interfaz de postulaciones pendientes.

La Figura 11 muestra cómo se ven los datos de una postulación para un usuario asistente. A la izquierda se muestran los datos personales del postulante. En la sección media de la interfaz se presentan los antecedentes académicos del postulante, y a la derecha se encuentran los documentos requeridos por el programa para que sea válida una postulación. Los documentos que han sido subidos correctamente contienen un link; éste sirve para descargar dicho documento y verificar que corresponde a lo requerido (por ejemplo, es un Currículum Vitae). Los documentos que aparecen en rojo son aquellos que están faltantes.

Postulaciones pendientes Todas las postulaciones Evaluaciones en proceso Asistente 1 Salir

Matías Contreras Reyes
 999999989
 mcontreras@ejemplo.com

Dirección
 Los Aromos 1701, Maipú, Chile

RUT/Pasaporte
 5173906-k

Nacionalidad
 No hay datos

Edad
 54

Postulando a Beca
 No

Antecedentes académicos

Grado de prueba
 Grado de prueba 2015 - 2017
 Chile

Documentos

Certificado de Título o Grado

Curriculum Vitae

Carta Motivacional

Certificado de Notas

Programa de Estudio

Carta de recomendación 1

Carta de recomendación 2

Esta postulación está pendiente.

Mandar a evaluación

[Volver](#)

Figura 11. Formulario de antecedentes de la postulación.

Aunque la postulación no tenga todos los datos que considera el formulario, ésta se puede mandar a evaluar si tiene lo mínimo necesario. Para ello, al hacer click en el botón “Mandar a Evaluación” (Figura 11), el sistema le informa al usuario el estado de completitud de la postulación, y le pide confirmar la acción antes indicada (Figura 12).

Por favor confirme

Esta postulación tiene los siguientes datos incompletos:

- Dos cartas de recomendación
- Carta Motivacional
- Programa de Estudio

Por favor confirme que desea continuar

Validar la postulación No validar

Figura 12. Confirmación al mandar una postulación a evaluación.

La opción “evaluaciones en proceso” para este perfil de usuario, al igual que para el perfil coordinador, muestra las postulaciones que han sido enviadas a evaluar y que aún no

cuentan con una resolución por parte del coordinador. La opción “todas las evaluaciones” muestra lo ya indicado en la Figura 9.

5.2.2. Interfaces de usuario para los miembros del Comité Académico

Las interfaces de los usuarios miembros del Comité Académico son similares a la de los usuarios asistentes en términos de estructura e información mostrada. Sin embargo, los miembros del Comité Académico sólo pueden ver la información de las postulaciones asignadas a ellos, mientras que los asistentes pueden ver todas las postulaciones.

Otra diferencia radica en la semántica de la opción de menú “evaluaciones pendientes”, que en el caso de los miembros del Comité Académico corresponde a las postulaciones asignadas a ellos, pero sobre las cuales aún no emiten un juicio de aceptación o rechazo.

Una tercera diferencia corresponde a que este perfil de usuario puede evaluar una postulación, y para ello debe completar el formulario que se muestra en la Figura 13. Los campos del formulario completo se muestran en el Anexo B.

Postulaciones pendientes Todas las postulaciones Evaluaciones en proceso Evaluador 3 Salir

Juvenal Herrera González
 999999994
 jherrera@ejemplo.com

Dirección
 Los Aromos 1701, Maipú, Chile

RUT/Pasaporte
 4-3

Nacionalidad
 No hay datos

Edad
 22

Postulando a Beca
 No

Antecedentes académicos

Grado de prueba
 Grado de prueba 2015 - 2017
 Chile

Documentos

- Certificado de Título o Grado
- Curriculum Vitae
- Carta Motivacional
- Certificado de Notas
- Programa de Estudio
- Carta de recomendación 1
- Carta de recomendación 2

Evaluación

Recomendación

Aceptar

Posible Aceptación

Posible Rechazo

Rechazar

Observaciones respecto del potencial

Observaciones respecto del potencial

Observaciones sobre pregrado

Observaciones sobre pregrado

... Otros campos omitidos

Otras observaciones

Otras observaciones

Guardar Terminar evaluación

[Volver](#)

Figura 13. Formulario de evaluación de una postulación.

Una cuarta diferencia entre las interfaces de los miembros del Comité Académico y los asistentes consiste en que los primeros, una vez que han emitido su opinión, pueden acceder a las opiniones emitidas por otros miembros del Comité Académico. Los asistentes no tienen acceso a los comentarios de los los miembros del Comité.

5.2.3. Interfaces de Usuario para el Coordinador

Al igual que en los casos anteriores, la opción de menú “Postulaciones pendientes” muestra todas aquellas postulaciones sobre las cuales el Coordinador tiene algo pendiente (una acción que tomar). En la Figura 14 se muestran postulaciones en estado “pendiente de validación”, las cuales corresponden a aquellas que aún no se han mandado a evaluar. También se muestran las que están “en evaluación por el comité académico”, y finalmente, aquellas que están esperando una resolución por parte del usuario Coordinador.

Nombre	Programa	Estado	Detalle	
Matías Contreras Reyes	MTI	Pendiente	Pendiente de validación	Resolver
Amir Ortiz Meneses	MTI	Pendiente	Pendiente de validación	Resolver
María José Peña Astaburuaga	MTI	Esperando Resolución	Evaluaciones terminadas el 29 de Marzo de 2021 a las 7:28	Resolver
Juvenal Herrera González	MTI	En evaluación por el comité académico	Estado de las postulaciones Evaluador 1 Evaluador 2 Evaluador 3 Evaluador 4	Resolver
Gabriela Alejandra Carrasco Martínez	MTI	En evaluación por el comité académico	Estado de las postulaciones Evaluador 1 Evaluador 2 Evaluador 3 Evaluador 4	Resolver
Chantal Peñafiel Piñonez	MTI	En evaluación por el comité académico	Estado de las postulaciones Evaluador 1 Evaluador 2 Evaluador 3 Evaluador 4	Resolver

Figura 14. Postulaciones pendientes para un usuario coordinador

Una vez que una postulación cuenta con la opinión de tres o más miembros del Comité Académico, el Coordinador puede acceder y resolver sobre la aceptación o rechazo del postulante. En la Figura 14, la cuarta postulación se encuentra en ese estado. Para tomar una decisión el Coordinador selecciona “Resolver” y accede al formulario que se muestra en la Figura 15 (vista parcial del formulario real).

Postulaciones pendientes Todas las postulaciones Evaluaciones en proceso Coordinador Salir

Juana Fernández Correa
 999999995
 jfernandez@ejemplo.com

Dirección
 Los Paltos 1701, Maipú, Chile

RUT/Pasaporte
 3-5

Nacionalidad
 No hay datos

Edad
 25

Postulando a Beca
 No

Antecedentes académicos

Grado de prueba
 Grado de prueba 2015 - 2017
 Chile

Documentos

- Certificado de Título o Grado
- Curriculum Vitae
- Carta Motivacional
- Certificado de Notas
- Programa de Estudio
- Carta de recomendación 1
- Carta de recomendación 2

Resolver la postulación

Usted deberá resolver esta postulación en UCampus

Cuando usted haga click sobre los botones de resolución, se copiarán los nombres de los evaluadores al portapapeles.

Evaluaciones

▼ Evaluador 1

Evaluación enviada el martes, 30 de Marzo de 2021 19:17
 El evaluador recomendó Aceptar

▼ Evaluador 3

Evaluación enviada el martes, 30 de Marzo de 2021 19:19
 El evaluador recomendó Aceptar

▼ Evaluador 4

Evaluación enviada el martes, 30 de Marzo de 2021 19:22
 El evaluador recomendó Aceptar

Figura 15. Formulario de resolución de postulaciones.

Para terminar, una vez que el Coordinador emite su evaluación sobre el postulante, la postulación avanza al estado “Esperando Resolución”. En este estado, el Coordinador debe resolver si acepta o rechaza la postulación. Para ello, la Figura 15 muestra dos botones en el lado izquierdo de la interfaz, uno para aceptar y otro para rechazar la postulación. Al lado derecho en dicha interfaz se muestran las evaluaciones y recomendaciones de cada miembro del Comité académico.

Sea cual sea la decisión del Coordinador, en el momento en que se presiona cualquiera de los botones, se termina el proceso interno de evaluación de una postulación, y ésta debe resolverse en UCampus. Esto es porque a pesar de que el sistema marque la postulación como resuelta, UCampus no expone una forma de resolver postulaciones, y por dicha razón, se debe resolver la postulación dentro de UCampus.

Mientras tanto, el sistema internamente marca la postulación como resuelta. Cuando una postulación ya está resuelta internamente en el sistema de evaluación de postulaciones, el Coordinador verá la interfaz mostrada en la Figura 16, la que le permite (en caso de ser necesario) devolver la postulación al estado “Esperando Resolución”. Esto se hace clickeando el botón “Volver a evaluación”. La razón de esta interfaz es para poder mitigar posibles equivocaciones del Coordinador, al momento de presionar los botones que indican la aceptación o rechazo de una postulación.

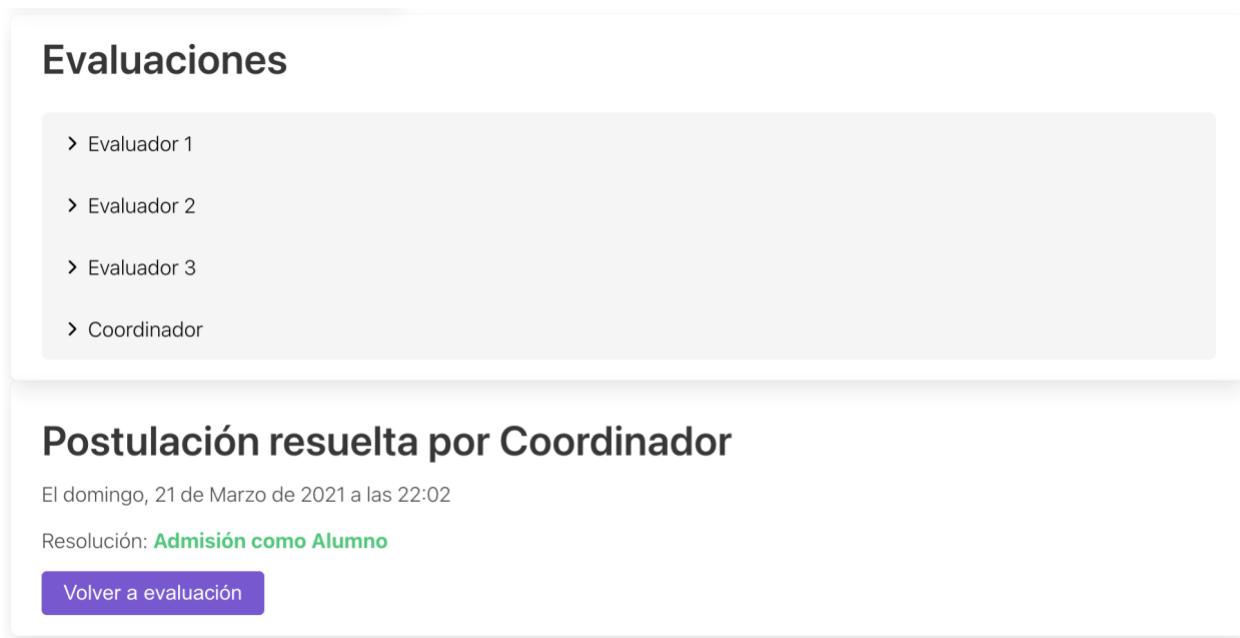


Figura 16. Interfaz para el coordinador de una postulación resuelta.

6. Evaluación de la Solución

A continuación se describe el proceso de evaluación realizado, los resultados obtenidos y los aspectos de mejora de la aplicación.

6.1. Proceso de Evaluación

El proceso de evaluación del software se enfocó principalmente en determinar la correctitud de operaciones realizadas por el sistema, y la adhesión de éste al flujo de trabajo definido para cada uno de los participantes. Para las pruebas el software se dejó disponible en la siguiente URL: <https://postulaciones.netlify.app/>. Se ingresaron 17 postulaciones al sistema, todas ellas con información ficticia, por un tema de privacidad.

Además se definieron tres usuarios con perfil de *asistente*, cuatro con perfil de *evaluador* (miembros del Comité Académico del programa) y un *Coordinador*. La evaluación fue hecha por los actuales coordinadores del Programa, quienes asumieron temporalmente los distintos roles (perfiles de usuario) para llevar adelante el proceso completo. Es importante destacar que estas personas han participado en la evaluación de postulaciones durante al menos los últimos 10 años.

Los distintos usuarios se autenticaron en la aplicación, y realizaron su labor habitual utilizando el nuevo sistema. Particularmente, durante esta evaluación los asistentes procesaron 12 de las 14 postulaciones disponibles. Los evaluadores revisaron y emitieron un juicio sobre 11 de ellas, y el Coordinador decidió la aceptación o rechazo de sobre 6 postulaciones. Esto significa que se realizó el flujo de trabajo completo para 6 postulaciones, involucrando a todos los actores en el proceso. Además, se realizó una parte del flujo de trabajo para otras 6 postulaciones.

El estado en que quedaron las postulaciones se muestra en la Figura 17, donde el estado "*pendiente*" significa que el asistente aún no ha revisado la postulación, y por lo tanto, no es posible evaluarla. El estado "*resuelta*" indica que los evaluadores emitieron su opinión, y el Coordinador resolvió en base a ellas. El estado "*en evaluación por el comité académico*" significa que la postulación está siendo evaluada, y que al menos un miembro de dicho comité no ha emitido su opinión respecto a la aceptación o rechazo de la misma. Finalmente, el estado "*esperando resolución*" significa que todos los miembros del Comité Académico han emitido su opinión, y sólo resta que el Coordinador decida y registre la decisión en UCampus.

Todas las postulaciones

Nombre	Programa	Modificación	Estado	Detalle
Matías Contreras Reyes	MTI	29 de mar	Pendiente	Pendiente de validación Resolver
Amir Ortiz Meneses	MTI	29 de mar	Pendiente	Pendiente de validación Resolver
Juan Andrés Pérez Gómez	MTI	30 de mar	Resuelta	Resuelta el 30 de Marzo de 2021 a las 18:35 Ver postulación
Alfonso Ignacio Correa Zapata	MTI	30 de mar	Resuelta	Resuelta el 30 de Marzo de 2021 a las 18:40 Ver postulación
Juvenal Herrera González	MTI	29 de mar	En evaluación por el comité académico	Estado de las postulaciones Evaluador 1 Evaluador 2 Evaluador 3 Evaluador 4 Resolver
Gabriela Alejandra Carrasco Martínez	MTI	29 de mar	En evaluación por el comité académico	Estado de las postulaciones Evaluador 1 Evaluador 2 Evaluador 3 Evaluador 4 Resolver
Chantal Peñafiel Piñonez	MTI	29 de mar	En evaluación por el comité académico	Estado de las postulaciones Evaluador 1 Evaluador 2 Evaluador 3 Evaluador 4 Resolver
Jorge Rojas González	MTI	29 de mar	En evaluación por el comité académico	Estado de las postulaciones Evaluador 1 Evaluador 2 Evaluador 3 Evaluador 4 Resolver
Francisca Astorga Gutierrez	MTI	29 de mar	En evaluación por el comité académico	Estado de las postulaciones Evaluador 1 Evaluador 2 Evaluador 3 Evaluador 4 Resolver
Juana Fernández Correa	MTI	6 de abr	Resuelta	Resuelta el 6 de Abril de 2021 a las 6:55 Ver postulación
Andrea Isidora Yáñez Oporto	MTI	6 de abr	Resuelta	Resuelta el 6 de Abril de 2021 a las 6:57 Ver postulación
Juan Adaos	MTI	6 de abr	Resuelta	Resuelta el 6 de Abril de 2021 a las 6:58 Ver postulación
Franco Apolonio Rojas	MTI	6 de abr	Resuelta	Resuelta el 6 de Abril de 2021 a las 7:00 Ver postulación
María José Peña Astaburuaga	MTI	29 de mar	Esperando Resolución	Evaluaciones terminadas el 29 de Marzo de 2021 a las 7:28 Resolver

Figura 17. Estado de las postulaciones una vez realizado el proceso de evaluación de las mismas.

6.2. Resultados

En primer lugar es importante destacar que la aplicación logró dar soporte al workflow completo del proceso de evaluación, considerando los roles de los participantes, y sin ningún error de lógica. Tampoco se detectó inestabilidad del sistema, y su usabilidad fue

calificada como “buena” por parte de los usuarios. En ese sentido se puede afirmar que se alcanzaron los objetivos definidos en esta memoria, y que la aplicación resultante está en condiciones de probarse en marcha blanca con postulaciones reales.

Por otra parte, los usuarios hicieron varios comentarios orientados a mejorar las interfaces del sistema, en pos de facilitar el trabajo de cada rol. También observaron varios aspectos que restringen la información a la que pueden acceder algunos roles, respecto de las postulaciones. A continuación se indican los más importantes:

- Que las tablas que muestran las postulaciones en los diferentes tracks permitan ordenar dichas postulaciones por columna, para así hacer más flexible y simple la identificación de postulaciones en diferentes estados, o bien identificar postulaciones en particular.
- Que los asistentes no puedan ver las evaluaciones hechas por los miembros del comité académico. Esto incluye la resolución propuesta y los comentarios realizados para cada uno de los ítems de evaluación. En resumen, los asistentes sólo podrán ver que la evaluación fue hecha por un evaluador un cierto día y hora.
- Un evaluador solo debe poder ver las postulaciones en las que participó como evaluador (independientemente de si realizó la evaluación, o se la cancelaron). Este tipo de usuario no debe poder ver las postulaciones en las que no participó.
- Reemplazar las actuales tres opciones de menú, por las que se indican a continuación, solo por una cuestión de facilidad de comprensión y acceso a las postulaciones por parte de los diferentes usuarios:
 - *Postulaciones abiertas*. Mostrar todas las postulaciones que aún tienen tareas pendientes de ser realizadas, ordenadas de más antiguas a más nuevas por fecha de postulación.
 - *Postulaciones cerradas*. Mostrar todas las postulaciones que ya fueron resueltas (ya sea aceptadas o rechazadas), ordenadas de más nueva a más antiguas por fecha de postulación.
 - *Evaluaciones pendientes*. Mostrar todas las postulaciones que el usuario tiene asignadas para evaluar, ordenadas de más antiguas a más nuevas por fecha límite para hacer la evaluación.
 - *Evaluaciones realizadas*. Mostrar todas las postulaciones que el usuario ya evaluó, ordenadas de más nueva a más antiguas por fecha en que fue realizada la evaluación.

Es importante destacar que estos ajustes al sistema plantean reorganizar o limitar el acceso a los servicios que ya están implementados. Por lo tanto, su incorporación no significa un gran desafío para el proyecto, y son fácilmente realizables.

7. Conclusiones y Trabajo a Futuro

Este trabajo se enmarca en el contexto de la evaluación de postulaciones de candidatos a ingresar al programa de Magíster de Tecnologías de la Información, que imparte la Escuela de Postgrado a través del DCC. Este proceso de evaluación, hasta antes de este trabajo, se realizaba de forma no automatizada; particularmente, era un proceso manual apoyado por correo electrónico.

Entendiendo el contexto general en el que se enmarca este trabajo, el objetivo general de esta memoria fue desarrollar un sistema Web que automatice parte del flujo de trabajo asociado al procesamiento de las postulaciones al programa de Magíster en TI. Para alcanzar ese objetivo, se definieron tres objetivos específicos:

- Desarrollar un servicio que extraiga datos de postulaciones desde UCampus.
- Desarrollar un sistema que asista a los participantes del proceso de evaluación en realizar su labor y automatizar el flujo de evaluación de cada postulación.
- Desarrollar un servicio que facilite la toma de decisiones del Coordinador del programa, respecto a la aceptación o rechazo de las postulaciones.

La solución desarrollada para lograr dichos objetivos fue una aplicación web que consta de dos componentes: front-end y back-end. El front-end es la interfaz con la que interactúan los participantes del proceso, cumpliendo el objetivo de asistir a éstos con las tareas que les competen. Por su parte, el back-end es el servicio que (1) extrae datos de postulaciones desde UCampus, (2) almacena datos tanto de cada postulación como de su historial de evaluaciones y su resolución y (3) disponibiliza los datos de una forma segura para el consumo de éstos por parte del front-end.

La solución fue evaluada por los actuales coordinadores del Programa, quienes han participado en la evaluación de postulaciones durante al menos los últimos 10 años, asumiendo distintos roles en dicho proceso. Durante la evaluación estas personas asumieron temporalmente distintos roles (perfiles de usuario) para poder realizar el flujo de trabajo completo.

Como resultado de la evaluación se comprobó que el software obtenido es estable y permite realizar las tareas asignadas a cada rol. También se pudo ver que el software también permite la interacción con la plataforma UCampus, con las limitaciones mencionadas en el Capítulo 2. Estas limitaciones se deben a restricciones impuestas por UCampus para cualquier software que quiera interactuar con dicha plataforma. A pesar de ello, la funcionalidad implementada para la interacción con UCampus reduce

considerablemente la carga de trabajo y las chances de cometer errores por parte de los usuarios asistentes y del Coordinador.

En resumen, se obtuvo un software funcional que puede ser puesto en producción (en marcha blanca), para procesar las postulaciones del MTI. Con algunos ajustes menores es posible que este software pueda ser utilizado para apoyar la evaluación de las postulaciones a otros programas del DCC, como por ejemplo, al Magíster en Ciencias y al Doctorado en Computación.

Como parte del trabajo a futuro, hay dos líneas principales. La primera (y más de corto plazo) es abordar los comentarios hechos por los evaluadores durante la evaluación del sistema. La segunda línea (de mediano plazo) es revisar este software con los coordinadores de otros programas del DCC, a fin de identificar funcionalidad extra que estos pudieran requerir. Luego, en base a ello, generar un único sistema que (aunque tenga variantes) sirva para apoyar a todos los programas.

La realización de este trabajo deja varias lecciones aprendidas, algunas de ellas como reflexión personal del autor de este trabajo y otras que deben ser tomadas en cuenta al momento de continuar con este trabajo.

Primero, un gran acierto de este trabajo fue que, luego de haber planteado la necesidad del sistema, se evaluara la porción del trabajo que involucraba el mayor riesgo: el scraper. En efecto, la primera tarea realizada fue evaluar la factibilidad de poder extraer datos desde UCampus, tarea que fue demostrada posible [8] en los inicios de este trabajo. Contar con el scraper resuelto resolvió ansiedad respecto a la factibilidad del proyecto y permitió continuar con el resto de los componentes teniendo la certeza de que el proyecto es posible. Naturalmente que esta reflexión no aplica sólo para este trabajo, sino para cualquier otro. La disminución del riesgo es imprescindible para que un proyecto sea exitoso.

Segundo, cuando las dependencias de un proyecto se vuelven complejas, necesitando de instalaciones sobre el sistema operativo para poder funcionar, la capa de abstracción que ofrece Docker es casi indispensable. Cualquier desarrollador que tenga que tomar este proyecto y extenderlo, puede hacerlo usando docker sin necesidad de instalar ningún driver de Selenium, y, más aún, ni siquiera necesita saber que existe o cómo se instala.

Finalmente, el scraping, pese a hacer posible la extracción de datos no estructurados o semi-estructurados desde cualquiera sea la fuente, no está exento de riesgos. Durante el desarrollo de este trabajo, el scraping falló múltiples veces al hacer *parsing* de campos

de fecha. Este es un problema -al menos para el parsing de UCampus-, porque las fechas están en formato legible para humanos, pero difícil de convertir a un objeto fecha de cualquier lenguaje de programación. Por ejemplo, algunos formatos encontrados fueron: *Ayer, a las 21:05 hrs., 2 de diciembre a las 21:05 hrs y 02/12/2021 a las 21:05 hrs.* Todos ellos son posibles y potencialmente referencian la misma fecha, pero **no es posible adivinar** de antemano cuáles son los formatos posibles.

En conclusión, se desarrolló un software que puede recolectar datos de postulaciones para el Magíster en Tecnologías de la Información y se proporcionó interfaces para ayudar a la resolución de estas postulaciones. Las decisiones técnicas tomaron en cuenta en todo momento la mantenibilidad de este proyecto, pues, al no ser posible predecir las decisiones de cambio en las interfaces de los lugares desde donde se hace scraping, la mejor opción es dar la capacidad de arreglar más rápidamente los errores antes que adivinarlos. En la misma línea, se usó Docker para minimizar el tiempo usado en correr el proyecto para desarrollo o para ponerlo en producción.

Bibliografía

- [1] Shamsi T. Iqbal y Eric J. Horvitz, 2010. Notifications and awareness: a field study of alert usage and preferences. En: 2010 ACM conference on Computer supported cooperative work. pp 27–30. URL: <https://doi.org/10.1145/1718918.1718926>.
- [2] Elements of Reusable Object-Oriented Software. 1994. Por Erich Gamma “et al”. Addison Wesley. pp 305-313.
- [3] Chrome DevTools team. 2020. Puppeteer v5.2.0 [en línea] <https://pptr.dev/>, [consulta: 29-03-2021].
- [4] Software Freedom Conservancy. 2021. Selenium WebDriver downloads [en línea] <https://www.selenium.dev/downloads/>, [consulta: 29-03-2021].
- [5] Curso CC5401: Ingeniería de Software II. 2013. Documento Histórico del Proyecto: Sistema de Recepción Postulaciones Magíster TI. Semestre Primavera 2013.
- [6] Crummy. 2004. Beautiful Soup: We called him tortoise because he taught us. [en línea] <https://www.crummy.com/software/BeautifulSoup/>, [consulta: 06-03-2021].
- [7] Scrapinghub. 2008. Scrapy | A Fast and Powerful Scraping and Web Crawling Framework [en línea] <https://scrapy.org/>, [consulta: 06-03-2021]
- [8] Nicolás Salas V. 2020, Scraping examples [en línea] <https://github.com/anachronic/scraping-examples>, [consulta: 10-04-2021].
- [9] Simon Brown. 2021. The C4 model for visualising software architecture, [en línea] <https://c4model.com/>, [consulta: 06-03-2021].
- [10] Stack Overflow. 2021. Stack Overflow Developer Survey, [en línea] <https://insights.stackoverflow.com/survey/2020> [consulta: 31-03-2021].
- [11] The Benchmark. 2021. Which is the fastest? [en línea] <https://github.com/the-benchmark/web-frameworks>, [consulta: 31-03-2021].
- [12] Redis Labs. 2021. Redis. [en línea] <https://redis.io/>, [consulta: 31-03-2021].

[13] Docker. 2021. Empowering App Development for Developers | Docker [en línea] <https://www.docker.com/>. [consulta: 31-03-2021].

[14] Python Software Foundation. 2021. Asyncio - Asynchronous I/O - Python 3.9.2 documentation. [en línea] <https://docs.python.org/3/library/asyncio.html>, [consulta: 01-04-2021].

[15] Facebook Inc. 2021. React - A JavaScript library for building user interfaces. [en línea] <https://reactjs.org/>, [consulta: 01-04-2021].

[16] Evan You. 2021. Vue.js. [en línea] URL: <https://vuejs.org/>, [consulta: 01-04-2021].

[17] Google. 2021. ChromeDriver - WebDriver for Chrome, [en línea] <https://chromedriver.chromium.org/>, [consulta: 10-04-2021].

Anexo A: Definiciones, Siglas y Abreviaturas

Las siguientes definiciones fueron sacadas en su mayoría de [5].

Comisión: Comisión de Magíster en Tecnologías de la Información.

Coordinador: Coordinador Académico del Magíster en Tecnologías de la Información.

Datos de postulación (DP): Consiste en todos los datos requeridos para postular. Estos se exponen a continuación.

a. Documentos Específicos de Postulación (DEP): Corresponden a:

- Curriculum Vitae.
- Certificado de título profesional o grado académico universitario equivalente por cada título o grado a presentar.
- Certificado de notas obtenidas en la institución que le otorgó el título profesional o grado académico universitario por cada título o grado a presentar.
- Dos cartas de recomendación.
- Carta personal de presentación del postulante donde explicita sus áreas de interés y objetivos para postular al programa.
- Documento adicional optativo.

b. **Datos Financieros de la Postulación (DFP):** Corresponden a los datos de postulación referentes al financiamiento del Magíster. Por cada atributo, una sola de las siguientes alternativas puede considerarse:

- Financiación: Particular o Empresa.
- Beca: Con solicitud de beca en trámite, con beca otorgada, o ninguna de las anteriores.

c. Datos Laborales de Postulación (DLP): Corresponden a:

- Empresa donde trabaja actualmente.
- Teléfono de la empresa.
- Dirección postal de la empresa.
- Cargo desempeñado.

d. Datos Personales de la Postulación (DPP): Corresponden a:

- Nombre del postulante.
- Nacionalidad.
- País de residencia.
- Dirección postal.
- RUT o pasaporte en caso de ser extranjero.
- E-mail.
- Teléfono de contacto.
- Licenciatura (especificar nombre del grado académico e institución que otorgó el grado, país y año de obtención).

- Magíster (idem anterior).
- Doctorado (idem anterior).
- Título profesional o grado universitario equivalente e institución donde finalizó tales estudios (idem anterior).

Para los últimos cuatro atributos se considera que el postulante puede ingresar información de más de un grado académico o título profesional si correspondiera, por ejemplo, en el caso de tener dos títulos profesionales.

e. Estado de una postulación: Corresponde al estado en que se encuentra una postulación dentro del proceso de resolución por parte del Departamento. Estos son los siguientes:

- Pendiente. La postulación se encuentra en el inicio del proceso de resolución.
- Validada o válida. Los DP de una postulación se consideran válidos para en base a ellos emitir una resolución.
- Invalidada o inválida. Se considera que una postulación tiene datos inválidos, por ejemplo, hay un documento en blanco, la carta de recomendación está duplicada, etc.
- En evaluación por el Coordinador. La postulación ha sido derivada al Coordinador para que genere una resolución.
- En evaluación por el Comité Académico. La postulación está siendo evaluada por los Miembros del Comité Académico.
- Resuelta. El Coordinador dio su veredicto sobre la resolución de la postulación.

f. Marca de una postulación: Corresponde al acto de señalar que se ha realizado alguna actividad particular con una postulación. Estas marcas (indicadores o flags) pueden ser:

- Notificada a la Escuela. Se ha notificado a la Escuela de Postgrado que se ha recibido una nueva postulación válida.
- Resolución notificada a la Escuela. Se ha notificado a la Escuela de Postgrado la resolución de la postulación.

MTI: Magíster en Tecnologías de Información.

Postulación: Acto de entregar a la Escuela de Postgrado, los DP por parte de un postulante a través del sistema de postulaciones de UCampus.

Postulación completa: Postulación en que todos los DP están completos, considerando las siguientes excepciones:

- En DPP, magíster y doctorado son datos opcionales.
- En DPP, entre licenciatura y título profesional, es mandatorio que al menos uno de ellos esté completo.
- Todos los DLP son opcionales.
- En DEP, el documento adicional optativo es, lógicamente, optativo.

Por lo tanto, toda postulación que no incluya los datos opcionales, y que cumpla la restricción de tener licenciatura o título profesional completo, se considerará completa también.

Resolución: Corresponde a dar un veredicto final sobre la postulación. El único tipo de usuario con esta capacidad es el Coordinador, quien puede emitir uno de los siguientes dictámenes:

- Rechazar solicitud de admisión.
- Admitir como alumno regular.
- Admitir como alumno condicional. En este caso se debe especificar los cursos de nivelación a tomar y el plazo para completar la nivelación.

Adicionalmente, el Coordinador puede incluir observaciones y/o sugerencias en la resolución.

Trabajar sobre una postulación: Corresponde al acto de cambiar el estado de una postulación o de marcar una postulación.

Anexo B: Formulario de Evaluación de Postulaciones

Magíster en Tecnologías de la Información
Formulario de evaluación de postulación

Postulante	_____	_____	Procedencia
Evaluador	_____	_____	Fecha

Recomendación	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
	Aceptar	Posible aceptación	Posible rechazo	Rechazar

Observaciones	Potencial:
	Pregrado:
	Experiencia laboral:
	Carta de intención:

Cartas de recomendación:

Otros antecedentes: