ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

Facultad de Ingeniería en Electricidad y Computación

Desarrollo de Extensión de Chromium como herramienta de apoyo para el dictado de clases de Fundamentos de Programación en ESPOL.

PROYECTO INTEGRADOR

Previo la obtención del Título de:

Nombre de la titulación

Ingeniero en Computación

Presentado por:

Flores Meza Carlos Fernando

Torres Ramírez Iván Patricio

GUAYAQUIL - ECUADOR

Año: 2021

DEDICATORIA

El presente trabajo es dedicado a mi familia, a mis amigos que son familia. Un sentimiento especial de agradecimiento a mis tías Karla y Gabriela Meza y a mis abuelos, por su ayuda constante e indispensable a lo largo de mis años académicos. Sin todos ellos y la motivación que traen a mi vida no hubiera sido posible.

Carlos Flores Meza

Dedico el presente trabajo a mis padres, hermana y amigos. Margarita Hurtado y Jack Crimmins por el interés constante en el desarrollo de mi carrera profesional, y en memoria del arquitecto Antonio Valdez por desearme éxitos durante mi educación secundaria y terciaria.

Iván Torres Ramírez

AGRADECIMIENTOS

Mi más sincero agradecimiento para profesores y compañeros, fueron parte fundamental de mi camino universitario y de la ejecución de este proyecto. Proyecto que no hubiera sido posible sin el apoyo de mis amigos y compañeros: Nathaly Gavino, Óscar Ávila y Ricardo Villacis. Al PhD. Boris Vintimilla por su guía y ayuda durante el desarrollo de este proyecto. Al M.Sc. Rafael Bonilla por su tutoría y buen humor.

Carlos Flores Meza

Agradezco a todas las personas que me han ayudado a terminar mi educación universitaria. Al Ph.D. Boris Vintimilla por su interés constante en el desarrollo de este proyecto. Al M.Sc. Rafael Bonilla por acompañarme en procesos de investigación a lo largo de los años. Y a mis amigos cercanos: Richard Moreno, Gabriel Rodríguez y Giovanna Jara.

Iván Torres Ramírez

DECLARACIÓN EXPRESA

“Los derechos de titularidad y explotación, nos corresponde conforme al reglamento de propiedad intelectual de la institución; **Carlos Fernando Flores Meza** e **Iván Patricio Torres Ramírez** damos nuestro consentimiento para que la ESPOL realice la comunicación pública de la obra por cualquier medio con el fin de promover la consulta, difusión y uso público de la producción intelectual"

Carlos Flores

Iván Torres

EVALUADORES

Firmado electrónicamente por:
RAFAEL IGNACIO BONILLA ARMIJOS

Firmado electrónicamente por:
BORIS XAVIER VINTIMILLA BURGOS

Boris Vintimilla Burgos

PROFESOR DE LA MATERIA

Rafael Bonilla Armijos

PROFESOR TUTOR

RESUMEN

Los profesores de Fundamentos de Programación en ESPOL suelen usar herramientas en línea como Repl.it para el desarrollo de sus clases, sin embargo, no es suficiente para cumplir con las expectativas de aprendizaje de los estudiantes. Por lo tanto, el objetivo de este proyecto computacional es desarrollar una extensión de Chromium que acompañe a los estudiantes y profesores de Fundamentos de Programación en la ESPOL para mejorar la experiencia de aprendizaje dentro del ambiente de desarrollo integrado Repl.it.

Los requisitos del sistema se determinaron en base a las necesidades de las partes interesadas (docentes y estudiantes) y el tiempo disponible para completar la implementación del proyecto. Se establecieron seis módulos de desarrollo, dos actores para el uso del producto, y se optó por la metodología ágil SCRUM para llevar a cabo el proyecto.

Se desarrolló una extensión de Chromium, para ayudar a estudiantes y profesores del curso de Fundamentos de Programación, usando conocimientos básicos de desarrollo de aplicaciones web. La interfaz del estudiante usa dos contenedores para mostrar palabras reservadas y traducciones de excepciones usando *datalists*, lo que facilita el acceso a la información a los estudiantes y mejora la calidad del código generado por ellos. La interfaz del profesor posee varios contenedores enfocados para descargar los trabajos enviados por los estudiantes de un paralelo determinado y generar informes de plagio, facilitando el proceso de evaluación a los docentes y dando retroalimentación relevante a los estudiantes en menor tiempo.

Palabras Clave: Extensión de Chromium, Apoyo académico, Python, Palabras reservadas, Traducción.

ABSTRACT

The Fundamentos de Programación *teachers at* ESPOL *usually use online tools such as Repl.it in their classes, however, it is not enough to meet the student's learning expectations. Therefore, the objective of this computational project is to develop a Chromium extension that accompanies the students and professors of* Fundamentos de Programación *at* ESPOL *to improve the learning experience within the Repl.it integrated development environment.*

The system requirements were determined based on the needs of the interested parties (teachers and students) and the time available to complete the implementation of the project. Six development modules were established, two actors for the use of the product, and the agile SCRUM methodology was chosen to carry out the project.

A Chromium extension was developed, to help students and teachers of Fundamentos de Programación*, using basic knowledge of web application development. The student interface uses two containers to display reserved words and exceptions translations using datalists, facilitating information accessibility to students and improving the quality of the code generated by them. The teacher's interface has several containers focused on downloading the work sent by students from a specific classroom and generating plagiarism reports, thus facilitating the evaluation process for teachers, and giving relevant feedback to students in less time.*

Keywords: *Chromium Extension, Academic Support, Python, Reserved Words, Translation.*

ÍNDICE GENERAL

ABREVIATURAS

API	Interfaz de Programación de Aplicaciones (Application Programming Interface)
CMS	Sistema de Manejo de Contenido (Content Management System)
DRF	Django REST Framework
ENA	Estudiante No Anglófono
ESPOL	Escuela Superior Politécnica del Litoral
FIEC	Facultad de Ingeniería en Electricidad y Computación
FP	Fundamentos de Programación
GUI	Interfaz Gráfica de Usuario (Graphical User Interface)
JSON	Notación de Objetos en Javascript (JavaScript Object Notation)
MIP	Materia Introductoria de Programación
ORM	Mapeo Objeto Relacional (Object Relational Mapping)

ÍNDICE DE FIGURAS

ÍNDICE DE ILUSTRACIONES

CAPÍTULO 1

1. INTRODUCCIÓN

En este capítulo se describe la necesidad del proyecto y la estructura de este. Se va a desarrollar una extensión de Chromium para acompañar a los estudiantes y profesores de Fundamentos de Programación (FP) en la Escuela Superior Politécnica del Litoral (ESPOL) para mejorar su experiencia de aprendizaje mientras se usa el ambiente de desarrollo integrado Repl.it. A continuación, se presenta la descripción del problema, su justificación, objetivos generales y específicos, módulos del proyecto, y marco teórico.

1.1 Descripción del problema

La industria tecnológica se encuentra en constante crecimiento y así mismo la necesidad de trabajadores con habilidades básicas de programación. La enseñanza de los principios básicos de programación no solo se enfoca en áreas relacionas a las ciencias computacionales sino también en cualquier área en dónde las habilidades de creación de código y resolución de problemas sean un requisito [1]. La metodología de enseñanza usada para materias de programación suele variar dependiendo de la residencia o calidad de la institución de educación, pero generalmente se práctica el uso de clases en formato de conferencia para transmitir los conocimientos a los estudiantes. Previamente se han implementado formas para ayudar a los estudiantes en el aprendizaje de técnicas de programación cómo herramientas de narración, programación visual, o modelos de flujo [2]. La disponibilidad de herramientas para los estudiantes en el aprendizaje ayuda significativamente a los profesores en el proceso de enseñanza, pero también se necesitan herramientas que permitan ayudar a los docentes mientras interactúan con los alumnos.

Existen ambientes de desarrollo integrado en línea que permiten a los estudiantes programar sin la necesidad de instalar herramientas de programación en sus computadores, y al mismo tiempo permiten a los profesores revisar el contenido desarrollado por sus estudiantes. Aunque los profesores tienen acceso

al trabajo realizado por los estudiantes, no existe el suficiente apoyo en las herramientas para analizar el progreso de los estudiantes en la tarea y proveer una retroalimentación adecuada. Debido a la variedad de estudiantes y casos de estudiantes con necesidades especiales de aprendizaje, los profesores suelen tener la disposición de ayudar a los individuos, pero no los medios para hacerlos [3].

Los profesores suelen usar las herramientas disponibles en línea para la enseñanza, pero suele ser no suficiente para los conocimientos esperados por parte de los estudiantes. Varios docentes tratan de crear métodos innovadores e interactivos en el progreso de las clases. Sin embargo, los métodos para evaluar el progreso de los estudiantes suelen ser en forma de calificaciones, por lo que se suele evaluar los conocimientos retenidos por los estudiantes más no su aprendizaje en el curso [4]. En consecuencia, se esperan herramientas en línea que permitan no solo al estudiante trabajar mejor y aprender sobre el curso, sino también herramientas que ayuden a los profesores en la calificación y evaluación de los conocimientos adquiridos por sus alumnos y a entender mejor su proceso de resolución de problemas.

1.2 Justificación del problema

El desarrollo u optimización de herramientas de aprendizaje y enseñanza para los principios básicos de programación necesita beneficiar a los estudiantes y docentes. Herramientas de programación como Repl.it han demostrado ser útiles en la enseñanza de conocimientos básicos de programación, permitiendo que alumnos con presunciones sobre la programación no tengan dificultades o disgustos en aprender a programar dentro de herramientas en línea [5] .En ESPOL se usa la herramienta mencionada como apoyo en la instrucción de FP, curso sobre los conocimientos básicos de programación y clase que suele ser impartida a estudiantes de los primeros semestres en la institución de educación superior.

Aunque la herramienta en línea Repl.it es usada frecuentemente por los profesores y estudiantes de FP en ESPOL, se considera que esta herramienta no tiene todas las cualidades necesarias para proveer una evaluación adecuada

sobre los conocimientos del curso. Se ha encontrado que varios profesores de la materia han desarrollado sus propias extensiones para el navegador web que permitan ayudar a la calificación de los trabajos de sus estudiantes que hayan sido enviados por Repl.it. Sin embargo, estas extensiones suelen ser personales por profesor y aunque ayudan en sus evaluaciones se necesita estandarizar la forma de evaluación de los docentes mientras se ayuda también a los estudiantes en el desarrollo del curso.

1.3 Objetivos

1.3.1 Objetivo General

Desarrollar una extensión de Chromium que acompañe a los estudiantes y profesores de Fundamentos de Programación en la ESPOL para mejorar la experiencia de aprendizaje dentro del ambiente de desarrollo integrado Repl.it.

1.3.2 Objetivos Específicos

1. Identificar las necesidades y procesos insuficientes en la evaluación de lecciones y tareas al momento de usar Repl.it.
2. Diseñar funcionalidades suplementarias a Repl.it para retroalimentar a los estudiantes mientras programan en el ambiente de desarrollo.
3. Implementar sistema de descarga y comparación de marcas de tiempo de trabajos enviados por estudiantes mediante el uso de Repl.it.
4. Incorporar un sistema anti-plagio al módulo de análisis del código escrito por los estudiantes en Repl.it.

1.4 Descripción de los Módulos del Proyecto

En esta sección se describe cada módulo a desarrollar en este proyecto. Los módulos fueron determinados en base a las necesidades del proyecto para satisfacer a los actores y el tiempo existente para terminar la implementación del proyecto. Los módulos se presentan a continuación en la Figura 1.1.

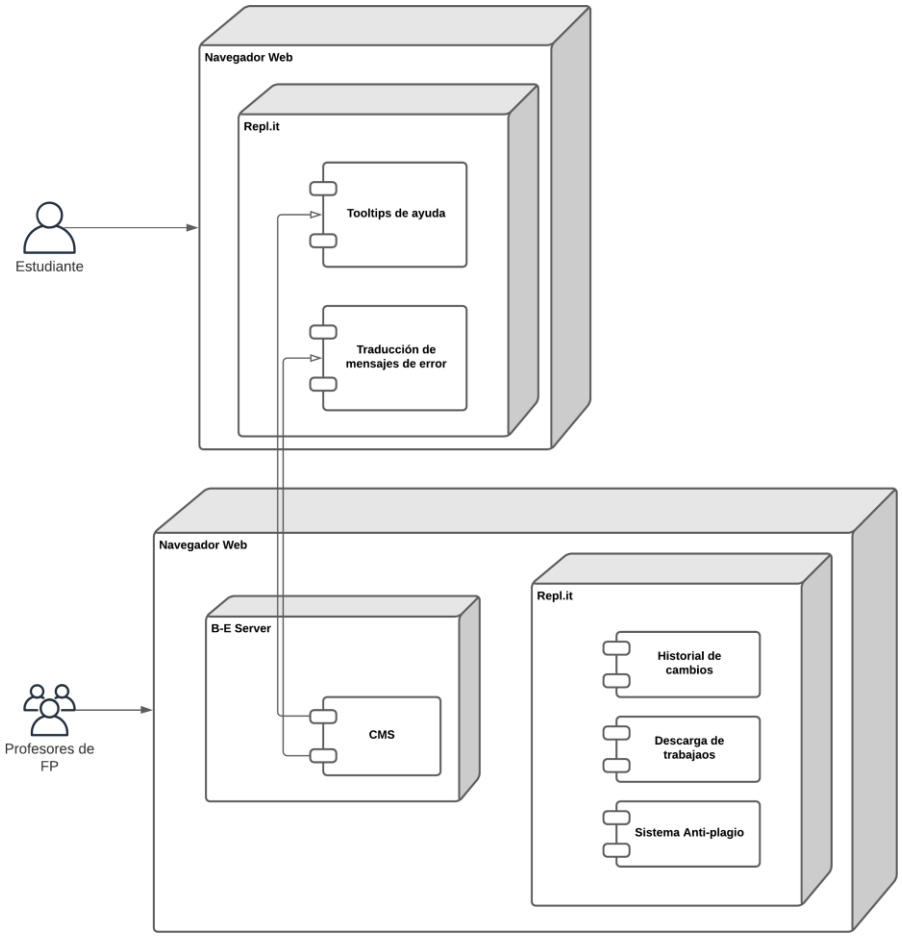


Figura 1.1 Diagrama de los componentes del proyecto

De esta forma, este proyecto plantea el desarrollo de una extensión de navegador web para el apoyo en el progreso del curso de FP mientras se usa la herramienta Repl.it. Se plantea que la extensión de Chromium a desarrollar ayude a los estudiantes por medio de la presentación de *tooltips* y traducción de excepciones en consolas. También que permita descargar y evaluar el progreso de los estudiantes mientras programan por parte de los docentes.

1.4.1 Módulo 1: Sistema de Manejo de Contenido

Este módulo es el centro de información del proyecto, se usa un sistema de manejo de contenido (*CMS*) para editar y agregar traducciones, ejemplos y consejos que los profesores consideren importantes para que luego sean consumidos por los *tooltips* y traducciones de la extensión del lado de los estudiantes. Esto permite editar de manera escalable y en tiempo real el contenido que se da a los estudiantes. Adicionalmente se puede usar el registro de peticiones para entender las necesidades de los alumnos y brindar esa información como apoyo a los profesores.

1.4.2 Módulo 2: Tooltips de ayuda

En este módulo se comienza a implementar las funcionalidades enfocadas en los estudiantes. La extensión por desarrollar es usada por estudiantes cursando FP, por lo que los *tooltips* de ayuda a diseñar contienen información sobre el lenguaje de programación Python. Se espera que, cuando un estudiante escriba funciones básicas, la extensión de Chromium despliegue un *tooltip* junto a la función que explique su utilidad y dé un ejemplo relevante de su uso. El contenido de los *tooltips* es predefinido por los desarrolladores y profesores involucrados para las funciones más comunes, autogenerado para los demás, y controlado mediante el módulo *CMS*.

1.4.3 Módulo 3: Traducción de excepciones

Este módulo se enfoca en la traducción de inglés a español de las excepciones mostradas en la consola de Repl.it, agregando valor en base al contexto. Se espera que, cuando un estudiante escriba código de forma que se produzca una excepción, la extensión de Chromium despliegue una ventana emergente con la traducción de la consola junto a la excepción original y comentarios relevantes controlados por los profesores de FP. Este módulo se piensa desarrollar usando herramientas externas de traducción junto a cambios diseñados por los desarrolladores para presentar más información sobre la excepción a estudiantes sin experiencia en el manejo de excepciones.

1.4.4 Módulo 4: Descarga automática de los trabajos de los estudiantes

En este módulo se comienza a implementar las funcionalidades enfocadas a los profesores. La herramienta Repl.it permite la descarga del código escrito por un determinado estudiante, sin embargo, esta acción tiene que repetirse para cada estudiante por parte del profesor. Se espera que, usando un botón en la interfaz de profesores de la extensión de Chromium, se puedan descargar todos los trabajos de los estudiantes inscritos en un curso dentro de la herramienta Repl.it. Se descarga una carpeta comprimida con carpetas individuales cuyo contenido es el código creado por el estudiante con su respectivo nombre, y las marcas de tiempo generadas en el desarrollo del código.

1.4.5 Módulo 5: Historial de cambios

Este módulo se enfoca en la generación de comparaciones entre marcas de tiempo del desarrollo del código por parte del estudiante. Se espera que, cuando el profesor entre a la sección de marcas de tiempo dentro de la herramienta Repl.it, se generen informes más detallados sobre los cambios generados en el código entre dos marcas de tiempo determinadas. Actualmente se pueden ver marcas de tiempo con los cambios del código realizados entre la marca y el final de la actividad, en la herramienta Repl.it, cuando los profesores ven las propiedades del deber enviado por un estudiante. La extensión añade más información sobre los cambios generados y permite a los profesores escoger las marcas de tiempo para realizar una comparación en el código creado.

1.4.6 Módulo 6: Sistema anti-plagio

El módulo final de este proyecto se enfoca en incorporar un sistema anti-plagio para conocer el porcentaje de plagio existente en los trabajos enviados por los estudiantes. Se espera que, usando un botón en la interfaz de profesores de la extensión de Chromium, se muestre el porcentaje de plagio de cada trabajo enviado por los estudiantes. El porcentaje mostrado tiene distintos colores dependiendo el valor del porcentaje para facilidad de lectura al profesor. Este módulo se piensa desarrollar junto a herramientas externas para análisis de código y detección de plagio.

1.5 Marco teórico

En la ESPOL, FP es una materia transversal, es decir, la cursan todos los estudiantes sin importar la carrera a la que pertenecen. FP se enseña usando Python, lenguaje de programación que posee en su librería estándar funciones cuyas acciones tienen una relación estrecha con el significado de sus nombres en el idioma inglés [6]. Estas dos particularidades llevan a que tanto profesores como estudiantes necesiten ayuda para aumentar la eficiencia de enseñar una materia en la que la mayoría de los estudiantes tienen huecos de conocimiento en las bases de la materia, en el lenguaje inglés o en ambos [7] [8].

Adicional a lo mencionado, la pandemia causada por la COVID-19 agregó problemas de logística, por lo que se recurrió a usar Repl.it, un ambiente integrado de desarrollo que permite la cooperación en tiempo real para el desarrollo de código. Esto abre puertas a la integración de soluciones a los problemas mencionados que no eran posibles con metodologías previas [9]. Específicamente ahora se puede brindar apoyo lingüístico a los estudiantes, darles fragmentos de código basados en el pensum y otorgar a los profesores soporte al calificar.

Previamente se han realizado estudios sobre el enseñar a programar [10], tanto desde el punto de vista de los profesores como de los estudiantes, sin embargo, no se ha estudiado a profundidad los efectos de la implementación de ayudas lingüísticas para estudiantes no anglófono (ENA) o de herramientas de apoyo a docentes para la calificación de trabajos.

1.5.1 Clases Introductorias de Programación

La relación entre conocimientos previos sobre resolución de problemas [10] y conocimientos matemáticos con el desempeño de estudiantes de cursos introductorios de programación (MIPs) está bien establecida en la investigación previa, relaciones que no existen en varias otras asignaturas [7]. Así mismo, existen estudios que recalcan la importancia de mantener el interés y la motivación de los estudiantes de cursos introductorios [11] [12] [13] ya que son factores determinantes en el éxito sostenido a lo largo de la materia [14].

Los profesores de MIPs como FP también enfrentan retos. Uno de los principales retos es entender la dificultad de ejercicios de programación, especialmente porque esto es algo fuertemente dictado por el contexto y varía para cada grupo particular de estudiantes [15] [16]. Consecuentemente, los profesores tienen que extraer información sobre el proceso de resolución de problemas de sus estudiantes al momento de calificar trabajos para ajustar su metodología a las necesidades de sus alumnos [17] y a los vacíos de conocimiento que los estudiantes presenten.

Si bien la literatura previa cubre las diferentes metodologías usadas por profesores [18], al momento del desarrollo de este estudio los autores no encontraron fuentes que estudien el efecto de implementar herramientas de ayuda didáctica sobre el desempeño de estudiantes de MIPs. Sin embargo, existen estudios que muestran los efectos del uso de herramientas web en MIPs [9] y presentan resultados que indican que el usarlas en conjunto con herramientas de ayuda didáctica sería beneficioso para los estudiantes de MIPs.

1.5.2 Estudiantes no anglófonos y ayudas lingüísticas

Habiendo establecido los retos de enseñar a programar, dado el contexto del estudio debemos entender el efecto que tiene el no ser un estudiante anglófono. De acuerdo con Horton y Craig en [14] el no hablar de manera adecuada el idioma inglés es un factor determinante en el éxito de los estudiantes de MIPs y [4]puede significar una diferencia promedio de 14.79% en los resultados de sus evaluaciones. Observación relevante pues Ecuador posee el índice de nivel de inglés más bajo de Latinoamérica [19] y FP es una materia dictada usando Python que es un lenguaje que tiene una relación estrecha con el lenguaje inglés [6].

Entre los ENAs el 34% identifica la lectura de material didáctico (libros guía, tutoriales, foros y documentación) como una barrera de aprendizaje, particularmente por la necesidad de recurrir a material escrito en inglés o por confusión al consultar material en sus lenguas nativas en los que tecnicismos se pierden en la traducción [20]. Otra fuente de información al practicar o resolver

ejercicios en una MIP son los mensajes de error. Investigación previa muestra que el dar mensaje de error más relevantes y representativos puede reducir la cantidad de errores cometidos al programar por un 29.06% en estudiantes anglófonos [21] de esto podemos inferir que el efecto sería similar o mayor en ENAs. El objetivo de brindar mensajes relevantes es el ayudar a los estudiantes a identificar la relación que existe entre su código y el error, reduciendo de esa forma la frecuencia de errores de sintaxis que son los más comunes en programadores novatos [22].

Esto lleva a que se hayan buscado previamente ayudas para ENAs, entre estas las más comunes son las ayudas lingüísticas, herramientas que buscan superar la barrera del lenguaje mediante el uso de dialectos locales [23]. Programas que se centran en ayudar a ENAs con la barrera del lenguaje también han tenido éxito en otros campos de la ingeniería [24] esto indica que un factor principal en la enseñanza de MIPs es el apoyo lingüístico, sin embargo, el solo traducir el contenido del curso no es suficiente para brindar una ayuda adecuada [20].

1.5.3 Antecedentes a la solución propuesta

La solución propuesta consiste en usar una extensión de Chromium para el apoyo didáctico y lingüístico, esto se basa en el uso previo de extensiones para superar barreras lingüísticas [25] e investigación previa sobre los efectos de brindar mensajes de retroalimentación relevantes y de calidad [21] junto a charlas con profesores que ya disponen de una extensión que al momento del desarrollo del estudio no satisface sus necesidades y confirman que los retos presentados en [10] representan la realidad del contexto del estudio. Finalmente, la herramienta para el sistema anti-plagio es seleccionada en base a la revisión de la literatura previa [26] e incorporada en la extensión de Chromium.

CAPÍTULO 2

2. METODOLOGÍA

2.1 Recolección de información

Actualmente se usa Repl.it como herramienta en línea para la enseñanza de FP en ESPOL debido a las distintas funcionalidades que posee, pero se considera que las mismas no son suficientes en base a las necesidades de los estudiantes y profesores. Los requerimientos del sistema en forma de historias de usuario han adquirido popularidad en el desarrollo ágil de *software* debido al aumento en productividad y calidad de entregables que se muestra por los desarrolladores [27]. Para obtener información sobre las historias de usuario del sistema, se realizó una encuesta a profesores que estén impartiendo el curso de FP en el segundo término académico del 2021 en ESPOL.

La encuesta se divide en seis preguntas para conocer sobre las dificultades y necesidades de enseñanza en el curso de FP. La primera y segunda pregunta tratan la importancia y dificultad que tienen los retos que enfrentan los profesores de FP en su experiencia dentro de la clase. Se usa una escala de Likert que va desde “Muy Importante” a “No Muy Importante” para medir indicadores como la comunicación con estudiantes, retroalimentación y el uso de herramientas para la enseñanza del curso. La tercera pregunta se enfoca en las dificultades que los estudiantes tienen que afrontar y con qué frecuencia los profesores ven estas dificultades en sus clases. Se usa una escala de Likert que va desde “Siempre” a “Nunca” para medir contenido como estructuras de control, habilidad lingüística y resolución de problemas. La cuarta pregunta se usa para conocer la cantidad de profesores que usan Repl.it, permitiendo saber la importancia de la herramienta en línea y la falta de funcionalidades en la misma. La quinta pregunta es una escala lineal del uno al diez para medir la satisfacción de los profesores con la herramienta y la pregunta final es una escala de Likert para determinar la relevancia de los cambios que estamos proponiendo en la extensión desarrollada.

La encuesta del Apéndice A permite tener información suficiente sobre las fallas de la herramienta Repl.it y las necesidades de profesores y estudiantes en el curso de FP.

Esta información es usada para la generación de requerimientos en la sección Análisis de Información.

2.2 Análisis de información

En base a la información obtenida de la encuesta en Recolección de Información, reuniones con el cliente y experiencia personal de los desarrolladores (como ex alumnos de FP) se generaron las historias de usuario. Las historias de usuario creadas para el desarrollo de la extensión de Chromium se enfocan en la interacción entre estudiantes y profesores, pero se dividen en base a los módulos de organización del proyecto. Se generaron diecinueve historias de usuario para todos los módulos del proyecto. Los módulos junto a sus respectivas historias de usuario se detallan a continuación.

Módulo 1: Sistema de manejo de contenido

1. Como profesor/administrador quiero editar los ejemplos, traducciones y consejos guardados para ayudar a los estudiantes en la generación de código.
2. Como desarrollador quiero desacoplar la base de información del *front-end* de la extensión cuando se hagan cambios para asegurar el funcionamiento correcto de la extensión.
3. Como profesor quiero controlar el contenido que los estudiantes reciben en sus componentes cuando usan Repl.it junto a la extensión de ayuda para brindar retroalimentación frecuentemente.
4. Como profesor quiero usar una interfaz gráfica de usuario (*GUI)* para realizar los cambios a los contenidos que ven los estudiantes en la extensión.

Módulo 2: Tooltips de ayuda

1. Como estudiante quiero visualizar *tooltips* de ayuda cuando escribo código para conocer información relevante de la función.
2. Como estudiante quiero visualizar ejemplos de uso de la función cuando escribo una función.
3. Como administrador quiero editar *tooltips* de ayuda para actualizar la información relacionada a la función en base a retroalimentación de los usuarios.
4. Como profesor quiero editar *tooltips* de ayuda para actualizar la información relacionada a la función en base al contenido que se va a ver en el curso.

Módulo 3: Traducción de excepciones

1. Como profesor quisiera saber cuántas veces los estudiantes reciben cada excepción.
2. Como estudiante quiero visualizar el texto original y traducido cuando se genera una excepción para entender el contenido de la excepción.
3. Como estudiante quiero recibir consejos en Repl.it cuando se genera una excepción para entender cómo solucionar el error en el código.
4. Como profesor quiero editar el contenido de los mensajes de excepciones traducidas cuando un estudiante genera una excepción para brindar una retroalimentación adecuada.

Módulo 4: Descarga automática de los trabajos de los estudiantes

1. Como profesor quiero presionar un botón de descargas cuando me encuentre en la ventana de trabajos enviados en Repl.it para descargar todos los trabajos enviados por los estudiantes.
2. Como profesor quiero almacenar los trabajos enviados en una carpeta comprimida cuando presiono el botón de descargar trabajos para evaluar los trabajos sin necesidad de conexión a internet.

Módulo 5: Historial de cambios

1. Como profesor quiero visualizar los cambios realizados en el código por un estudiante cuando envía un trabajo en Repl.it para conocer sobre las dificultades del estudiante al programar.
2. Como profesor quiero almacenar el historial de cambios de los trabajos enviados en una carpeta comprimida cuando presiono el botón de descargar trabajos para evaluar los trabajos sin necesidad de conexión a internet.

Módulo 6: Sistema anti-plagio

1. Como profesor quiero visualizar el porcentaje de plagio cuando me encuentre en la ventana de trabajos enviados en Repl.it para conocer sobre los estudiantes que realizaron la actividad deshonesta.

2. Como profesor quiero visualizar el contenido plagiado cuando abra el trabajo del estudiante para conocer las razones de plagio del estudiante.
3. Como profesor quiero conocer si el plagio fue a otro estudiante o a internet cuando abra el trabajo del estudiante para determinar la sanción adecuada.

Usando las historias de usuario creadas en base a la información de la encuesta, se puede diseñar alternativas de soluciones que cumplan los requerimientos del sistema y permitan la mejora en la enseñanza de los cursos de FP en ESPOL.

2.3 Alternativas de soluciones

Las historias de usuario permiten conocer los requerimientos del sistema, por lo que cada módulo tiene que cumplir sus respectivas historias mientras se soluciona el problema. Las alternativas de solución se dividen por los módulos del proyecto teniendo en cuenta las opciones de implementaciones y sus consideraciones. Esto se puede ver en la Figura 2.1.

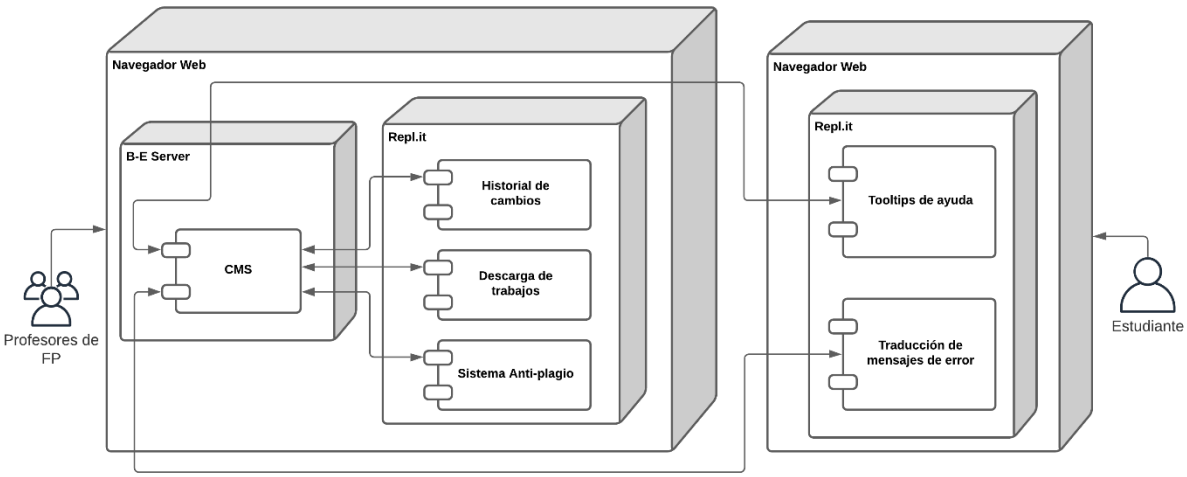


Figura 2.1 Diagrama de componentes para alternativas de solución

2.3.1 Módulo 1: Sistema de manejo de contenido

En el módulo uno se consideró como opciones de implementación a Django, Laravel y Node junto a Express, debido a que el *CMS* tiene que ser desarrollado internamente, poseer *GUI* y ser alojado en la nube.

Laravel tiene una comunidad de soporte muy activa y basta, optimización de *queries* a la base datos por defecto y los *queries* retornan notaciones de objetos en Javascript (*JSONs)* que no necesitan ser serializados o parseados para enviarlos como respuesta a una petición web de la extensión de Chromium. Por el otro lado, esta opción funciona con el lenguaje de programación PHP y no tiene un sistema de *templating* integrado, haciendo que se tengan que usar soluciones de terceros y aumentando el tiempo de trabajo de los desarrolladores.

Como segunda alternativa se tenía el uso de Node con Express debido a su alto rendimiento para interfaces de programación de aplicaciones (*APIs*), documentación completa, y que permite integración con bases No-SQL si se necesitara hacer cambios drásticos al sistema. Así mismo, esto se ve limitado debido al uso singular de hilos y programación asíncrona, aumentando la complejidad en el cambio de código y limitando el rendimiento en base al alojamiento en la nube.

Finalmente, se considera Django debido a que ya posee una página de administración estándar que permite un acceso sencillo a la *data* mediante su mapeo objeto relacional (*ORM)*, usa el lenguaje de programación Python que se enseña en FP y por tanto asegura la mantenibilidad a futuro, sus modelos de *ORM* son fácilmente editables y su capacidad de generar migraciones a partir de los modelos lo vuelven agnóstico a la solución de base de datos. Por el otro lado, no tiene manejo de *APIs* por defecto por lo que se tendrían que usar librerías externas, y la serialización a *JSON* es lenta, sin embargo, ya que no es un caso de uso en el que las operaciones a tiempo real sean críticas, esto último no se considera.

.

2.3.2 Módulo 2: Tooltips de ayuda

Es el primer módulo cuya interfaz se encuentra del lado del cliente web y es parte de una extensión de Chromium, esto presenta limitantes en cuanto a su desarrollo. Si bien existen varias formas de crear estas extensiones, los desarrolladores optaron por usar JavaScript puro para el funcionamiento y HTML+CSS3 para las vistas.

En cuanto al funcionamiento, Google presta una documentación para el desarrollo de extensiones, donde se encuentran dos funcionalidades que se consideraron para el

desarrollo de la solución del módulo: *Commands* y *Content Scripts* [28]. *Commands* hace referencia a la capacidad de ejecutar acciones mediante interacciones del usuario y *Content Scripts* se usa para realizar acciones en base al entorno específico del sitio web en cualquier momento. Alternativamente, se pueden usar contenedores que pidan la información bajo demanda usando Javascript puro.

2.3.3 Módulo 3: Traducción de excepciones

Este módulo presentaba retos similares a los del módulo dos, con la diferenciación importante de la *GUI*, ya que debe suceder sobre la consola de Repl.it. Para esto se planteó usar un *scrapper* (programa que analiza el HTML de una página y extrae información relevante) junto a un modal que se detonan con la funcionalidad de *ContentScripts* o usar un modal junto a la funcionalidad de *Commands*. El primero no requeriría interacción del usuario y el segundo funcionaría de manera similar a la solución con *Commands* del módulo anterior, de la misma manera, las peticiones bajo demanda son una opción.

2.3.4 Módulo 4: Descarga automática de los trabajos de los estudiantes

Este módulo tiene la particularidad de ser sencillo en su ejecución, por lo que la única consideración que se tuvo para su implementación es un botón en la *GUI* que detone su funcionalidad. En cuanto a la funcionalidad, la única posibilidad que se encontró fue el cambiar el comportamiento por defecto del botón “Enviar” que ven los estudiantes en Repl.it para que además envíe el trabajo al CMS y luego se sirven a los profesores bajo demanda.

2.3.5 Módulo 5: Historial de cambios

Para este módulo, se tuvieron requerimientos específicos, por lo que, para obtener estas funcionalidades, se plantearon entonces las alternativas de extender la funcionalidad del módulo cuatro y de manera remota generar un archivo de diferencia para cada marca de tiempo o aplicar en tiempo real los cambios con un *scrapper* que mande el texto al *CMS* y reciba de respuesta el texto comparado.

2.3.6 Módulo 6: Sistema anti-plagio

Para determinar la herramienta a usar para este módulo se consultó la investigación previa presentada en [29], la cual hace un análisis extenso de las diferentes herramientas, librerías y algoritmos que se pudieron usar.

2.4 Plan de implementación

2.4.1 Proceso SCRUM

Se empleó la metodología ágil por medio de la herramienta Jira de Atlassian para el desarrollo del proyecto, siendo dividido en seis *sprints* a lo largo de seis semanas. La metodología ágil se ha popularizado debido al cambio de enfoques jerárquicos a enfoques colaborativos para la administración de proyectos complejos [30], permitiendo que compañías enfocadas al desarrollo de *software* usen un *framework* específico para la resolución de problemas y creación de entregables.

Para la administración de este proyecto informático se utilizó Jira para la realización de los *sprints*, manejo de requerimientos, y realización de tickets. Jira es un sistema de seguimiento de problemas usado en procesos iterativos de entregables, y debido a sus características personalizables, también permite administrar las actividades a realizar por medio de la creación de tickets asignados en los *sprints* del proyecto [31].

Los estados de los tickets permiten conocer el desarrollo de las actividades y suelen ser los mismos para los proyectos manejados en Jira. Por el otro lado, en el desarrollo de este proyecto se realizaron cambios en los estados, como se muestra en la Figura 2.2, para facilitar el manejo de la herramienta sin experiencia alguna de la misma.

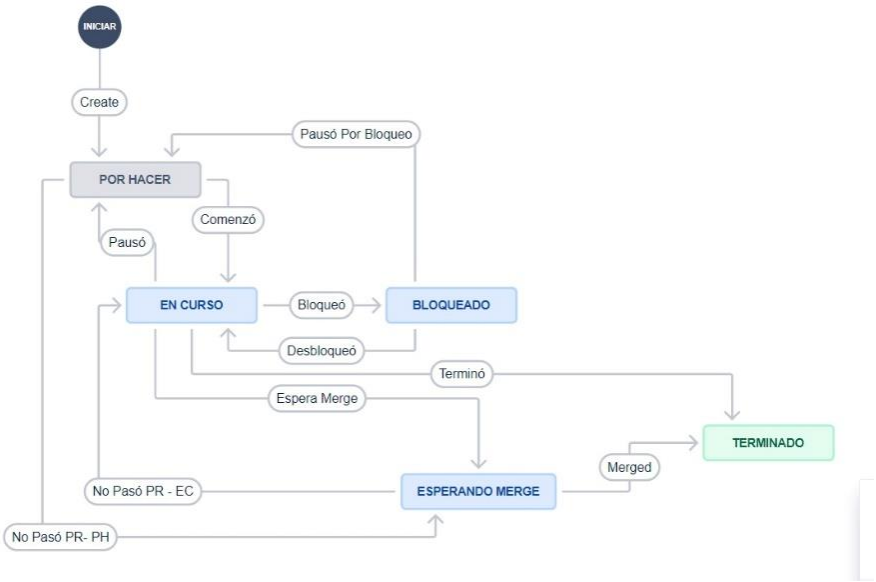


Figura 2.2 Cambios de estados de los tickets

Se decidió realizar el proyecto por medio de seis *sprints* debido a que el proyecto se encuentra dividido en seis módulos, con una duración total de seis semanas. Cada módulo se realizó por medio de un *sprint*, y el tiempo de diferencia entre la finalización de los *sprints* y la entrega del proyecto se usó para corrección de errores y optimización.

2.4.2 Propuesta de solución

En base a investigación previa se seleccionaron las herramientas que brindaban la mejor mantenibilidad, facilidad de desarrollo y cumplían con los requerimientos del proyecto. Estas se detallan a continuación:

1. Módulo 1:

   Se optó por usar Django principalmente por usar el lenguaje de programación Python, que es el único que garantiza una fácil mantenibilidad a futuro en el contexto del proyecto. Además, no tenía desventajas importantes en comparación a las otras alternativas.

2. Módulo 2:

   Se comparó la posibilidad de usar Content Scripts para detonar los tooltips de manera dinámica según lo que el usuario escribe, sin embargo, eso crea

una dependencia con el HTML de Repl.it sobre el que no se tiene control. Y ya que el usar *Commands* no brinda funcionamiento automatizado, se optó por usar Javascript puro para pedir la información bajo demanda. Se muestra una barra de búsqueda en la que el usuario puede ingresar una palabra reservada y obtener información y ejemplos del *CMS*. Esta implementación también facilita la recolección de data para los reportes del lado de los profesores.

3. Módulo 3:

   Por razones similares a las del Módulos 2, se optó por usar Javascript puro para que el estudiante pida la información bajo demanda desde una barra de búsqueda en la vista de la extensión.

4. Módulo 4:

   Para este módulo, un solo botón en la interfaz de los profesores les permite la descarga de los documentos de todos los estudiantes para una tarea o evaluación específica para un para un paralelo determinado.

5. Módulo 5:

   Durante la investigación para la herramienta a usar en el módulo 6, se encontró la librería '*difflib'* para Python. Esta es usada para determinar los cambios de una versión del documento a otra. Se usa un *scrapper* para conseguir el texto de cada marca de tiempo y se envía para procesamiento al *CMS* bajo demanda. Esto genera un archivo de cambios que luego es mostrado a los profesores en su *GUI* de Repl.it. Los profesores pueden escoger entre dos modos de funcionamiento que se implementaron mediante el uso de una barra en la *GUI*. Estos modos permiten comparar dos marcas de tiempo arbitrarias o una marca de tiempo con su inmediato anterior.

6. Módulo 6:

   Luego de comparar los resultados de [29], se eligió la herramienta MOSS. La misma será usada desde la interfaz de los profesores para determinar

el valor de similitud entre dos trabajos. Posteriormente se usa ese valor para mostrar una puntuación junto al trabajo de los estudiantes.

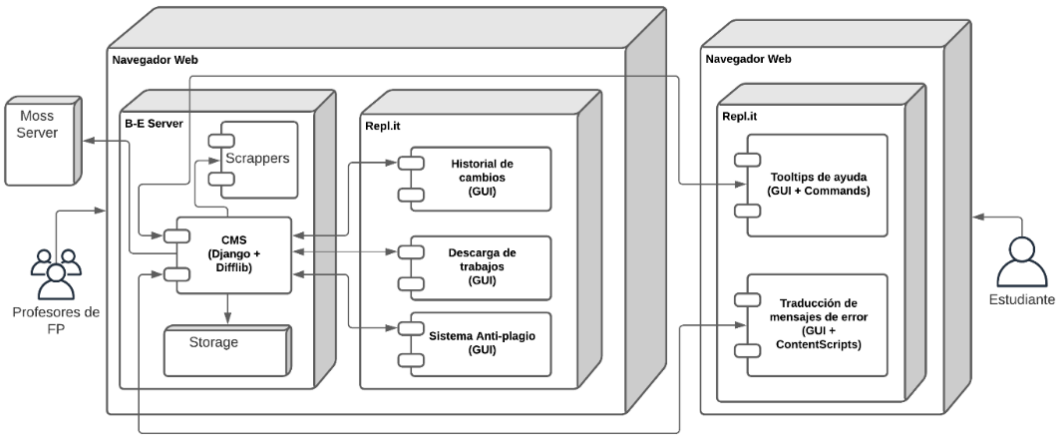


Figura 2.3 Diagrama de componentes para propuesta de solución

Uniendo todos los componentes con sus respectivas herramientas de solución se obtiene lo mostrado en la Figura 2.3. En el siguiente capítulo se explica a detalle la implementación de cada componente junto a sus respectivas herramientas.

CAPÍTULO 3

3. RESULTADOS

Se realizó una extensión de Chromium para ayudar a estudiantes y profesores del curso de FP usando conocimientos sobre desarrollo de aplicaciones web y usando la metodología de desarrollo de software ágil SCRUM. El proyecto de software desarrollado usó seis módulos para organizar las actividades a realizar en consideración a las necesidades de los profesores y estudiantes; por ende, el desenvolvimiento de los módulos se presenta en el funcionamiento de las interfaces de usuario creadas para cada actor involucrado.

3.1 Página de administración de Django

El primer módulo del proyecto consistió en crear un *CMS* por medio de *Django Administration* para administrar el contenido de la extensión de Chromium, almacenar trabajos enviados por estudiantes y generar informes de plagio. Para acceder a la página de administración de Django se tiene que pasar por una página de inicio de sesión (Ilustración 3.1) y usar credenciales que inicialmente se dejan definidas para que un usuario pueda entrar, y luego este usuario pueda crear más usuarios con los permisos necesarios.

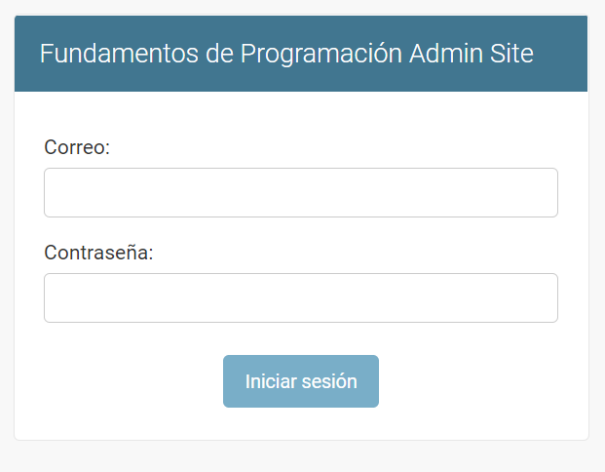


Ilustración 3.1 Inicio de sesión de la administración de Django

Los usuarios que usan permiso de edición son los administradores y los profesores. Los administradores serán los individuos (que pueden ser profesores)

encargados del mantenimiento de contenido en el *CMS*, así como la creación de usuarios con permisos en específico o el uso del token de identificación dentro de los módulos “Autentificación y autorización” y “Token de Autenticación” (Ilustración 3.2).

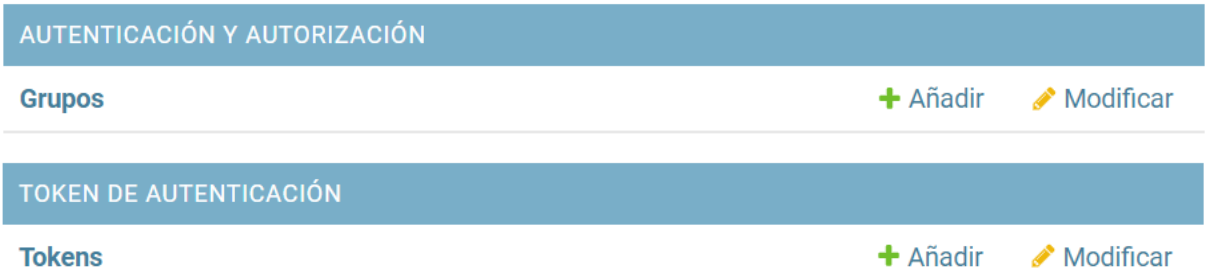


Ilustración 3.2 Módulos menores de *Django Administration*

En el módulo CMS se encuentra la información necesaria para el funcionamiento de la extensión y que va a ser actualizada al inicio de cada semestre (Ilustración 3.3). La tabla “Excepciones” y “Tooltips” son las tablas encargadas del almacenamiento de las excepciones que puedan aparecer por consola y las palabras reservadas de Python respectivamente. Las tablas “Ejemplo de Tooltips” y “Ejemplos de Excepciones” se encargan de guardar los ejemplos correspondientes a las excepciones y funciones escogidas por el estudiante en su respectiva interfaz dentro de la extensión de Chromium.

Ilustración 3.3 Tablas de administración del CMS

El almacenamiento de usuarios para estudiantes y profesores se encuentran en las tablas con los respectivos nombres, y estás tablas son actualizadas por administradores al inicio de cada semestre luego que ya se tienen los paralelos definidos con los

estudiantes integrados, esto sucede mediante el uso de la funcionalidad de "*loaddata*" del módulo *management* de Django y un archivo *JSON* con un formato acordado con los desarrolladores. Al final del módulo CMS se encuentra la tabla donde se crean los trabajos enviados por los profesores a los estudiantes, de forma que las soluciones enviadas por los estudiantes puedan ser guardadas sin dificultad. El mantenimiento del *CMS* cada semestre por parte de administradores y profesores asegura calidad en la información presentada a los estudiantes, permitiendo que la extensión siga estando en constante desarrollo mientras que los actores involucrados con la extensión se benefician.

3.2 Inicio de sesión de la extensión

En la página web de inicio de la herramienta Repl.it no existe información adecuada dentro del código fuente para diferenciar entre el usuario de profesor y estudiante, por lo que dentro de la extensión de Chromium se integró una pantalla de inició de sesión para diferenciar a los usuarios (ver Ilustración 3.4). Las credenciales de los profesores se generan de manera aleatoria cuando un administrador crea la cuenta. La contraseña de los profesores se caduca inmediatamente y cada profesor la cambia a una que solo ellos conocen en el primer inicio de sesión. Las cuentas de los estudiantes son creadas mediante la funcionalidad de `loaddata` y sus contraseñas se comportan de la misma manera.

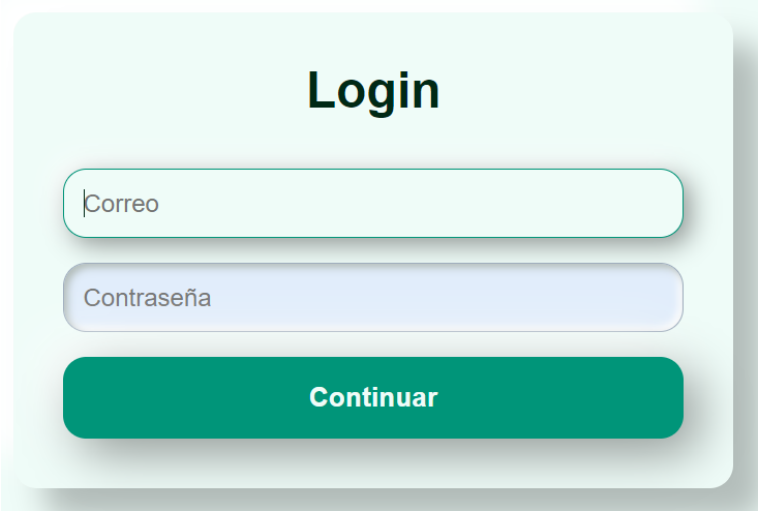


Ilustración 3.4 Inicio de sesión de la extensión de Chromium

3.3 Interfaz de estudiante

La interfaz del estudiante se presenta cuando las credenciales de inicio de sesión concuerdan con un estudiante registrado en cualquiera de los paralelos existentes dentro del *CMS*. Esta interfaz muestra dos contenedores para presentar la información relacionada al estudiante dentro del *CMS*, y un contenedor para cerrar la sesión del estudiante (ver Ilustración 3.5). Las áreas para presentar información consisten de un contenedor para la búsqueda de palabras reservadas de Python y otro para la búsqueda de excepciones comunes por los estudiantes.

Ilustración 3.5 Interfaz de estudiante en la extensión de Chromium

En los contenedores de búsqueda de la interfaz para estudiantes se muestra un elemento HTML llamado *datalist* (ver Ilustración 3.6). Este es muy similar al elemento *drop-down list* también usando en HTML, con la diferencia que tiene un buscador de elementos incluido. Al dar clic en el buscador del primer contenedor aparece una lista de las palabras reservadas de Python guardadas en el *CMS*, de manera que el estudiante puede buscar las funciones que necesite en caso de no conocer su uso o desee visualizar un ejemplo.

Aunque la información presentada tenga similitudes con información mostrada en buscadores de internet, el contenido de la extensión es escrito por profesores que poseen los conocimientos adecuados para que los estudiantes puedan aprender con mayor facilidad.

Ilustración 3.6 *Datalist* del contenedor "Buscar palabras reservadas de Python"

El segundo contenedor usa un *datalist* parecido al primer contenedor, con la diferencia que este muestra las excepciones comunes encontradas al programar en Python y describe posibles soluciones (ver Ilustración 3.7). Las excepciones mostradas en la consola de Repl.it se encuentran en inglés, mientras que el buscador de este contenedor permite la busqueda en inglés o español. La extensión no solo provee una traducción adecuada de la excepción, sino una posible solución propuesta por los profesores.

Ilustración 3.7 *Datalist* del contenedor "Buscar excepciones comunes"

El uso de estas dos herramientas de la interfaz del estudiante es opcional y las búsquedas por demanda. Permiten ayudar al estudiante con sus dificultades o deficiencia de inglés al programar. La información presentada en los *datalist* es redactada por los profesores de la materia que se involucran en el uso y desarrollo de la extensión. De esta manera, los datos brindados son relevantes para los estudiantes cursando la materia.

Al final de la interfaz del estudiante se encuentra un contenedor con el botón para cerrar la sesión y regresar a la interfaz inicial.

3.4 Interfaz de profesor

La interfaz del profesor se presenta cuando las credenciales de inicio de sesión concuerdan con un profesor registrado en el *CMS* (ver Ilustración 3.8). Esta interfaz posee varios contenedores que trabajan juntos para presentar las funcionalidades de los profesores. El primer contenedor posee un *drop-down list* donde se escoge el paralelo que el resto de los contenedores va a utilizar. La extensión intenta predecir el paralelo

en base al nombre del equipo de Repl.it, de no ser posible, el profesor debe escoger manualmente de su lista de paralelos asignados.

Ilustración 3.8 Interfaz de profesor en la extensión de Chromium

El segundo contenedor posee el botón descargar tareas que permite al profesor descargar todas las soluciones enviadas por los estudiantes del paralelo escogido previamente. Este botón se encarga de facilitar el proceso de descargar los trabajos de los estudiantes que se presenten en pantalla, guardando todos los archivos de código fuente en una carpeta comprimida (ver Ilustración 3.9). Las tareas se descargan del *CMS* donde se guardan las tareas enviadas por los estudiantes usando Repl.it mientras tienen la extensión de Chromium activa.

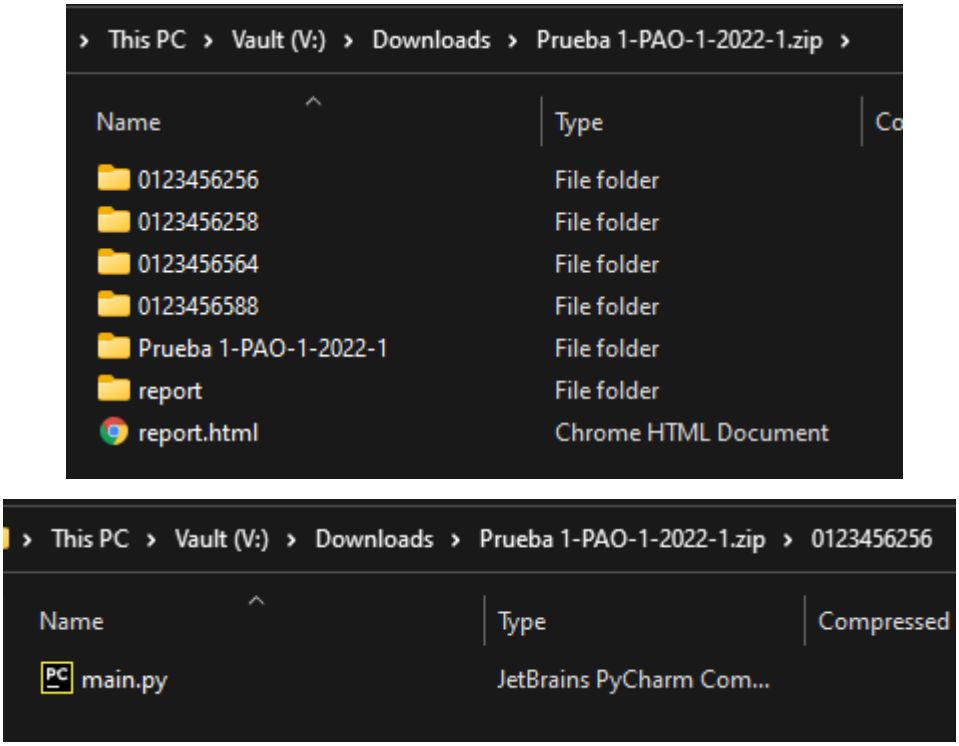


Ilustración 3.9 Trabajos descargados en la carpeta comprimida

El tercer y cuarto contenedor de la interfaz de profesor se relacionan a las actividades realizadas para detectar plagio en los trabajos enviados por los estudiantes. Al presionar el botón para calcular el porcentaje de plagio se generan los informes de MOSS (ver Ilustración 3.10) dentro del *CMS* respecto al paralelo escogido previamente. Al tener los informes de MOSS generados en el *CMS*, el cuarto contenedor con el botón “Descargar informe anti-plagio” adquiere su funcionalidad y los informes anti-plagio se guardan en un archivo comprimido.

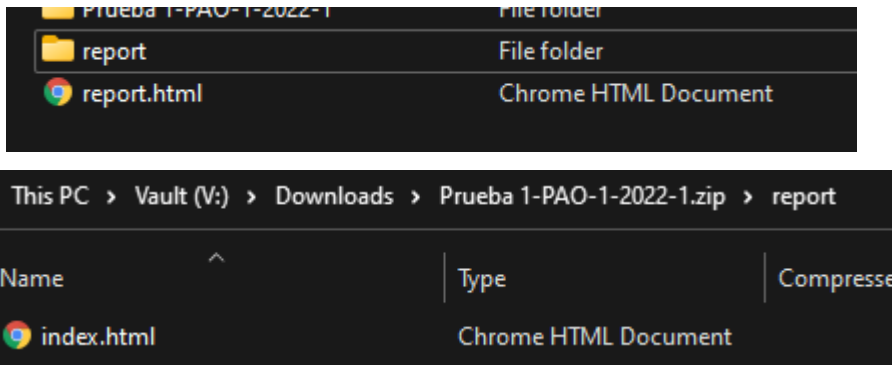


Ilustración 3.10 Informes anti-plagio descargados en la carpeta comprimida

El quinto contenedor muestra un *toggle* que permite visualizar u ocultar una columna con el porcentaje de plagio que posee cada trabajo enviado dentro de Repl.it (ver Ilustración 3.11). Este texto posee distintos colores: se usa el color rojo para trabajos con plagio superior al 66%, color naranja para trabajos con plagio entre 33% y 66%, y color verde para trabajos con plagio inferior a 33%.

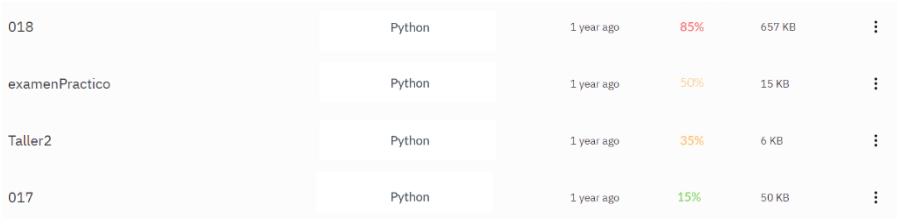


Ilustración 3.11 Columna de porcentaje de plagio

El sexto contenedor de la interfaz de profesor posee el *toggle* “Mostrar historial de cambios”, que solo funciona en la página “Historial de Repl.it” (ver Ilustración 3.12). Esta funcionalidad permite visualizar, con diferentes colores intuitivos, las líneas de código que han sido eliminadas o agregadas al documento y cuando fue hecho el cambio.

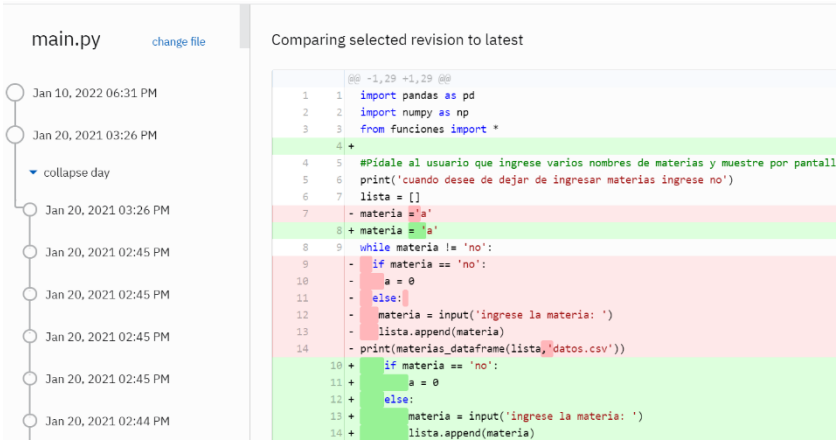


Ilustración 3.12 Comparación de marcas de tiempo

Teniendo una interfaz para cada usuario, con las funcionalidades mencionadas en este capítulo, se cumplen los requerimientos definidos al principio del desarrollo del proyecto y se puede calcular el costo de creación del mismo.

3.5 Análisis de Costo

El desarrollo de este proyecto integrador no posee costo alguno debido a que fue realizado por estudiantes de la carrera de Ingeniería en Computación como requisito

para graduarse, y enfocado en ayudar a profesores y estudiantes de FP en el dictado de la clase. En caso de que el proyecto fuese realizado de forma independiente a la institución educativa ESPOL, se presentan los posibles costos que tendría la realización de la extensión de Chromium en base a las horas de trabajo por los involucrados. A diferencia de la metodología de esta investigación, las horas empleadas en el desarrollo de software no son divididas en los módulos del proyecto si no en las diferentes actividades realizadas (ver Figura 3.1).

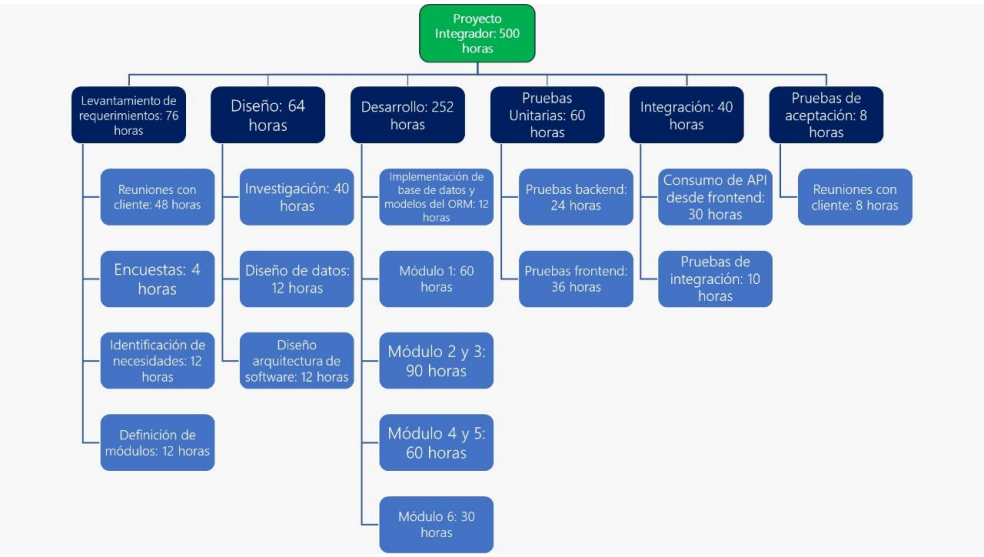


Figura 3.1 Horas de trabajo usadas para proyecto integrador

En total, se usaron 500 horas para el desarrollo del proyecto integrador. Considerando que el salario mensual promedio de un ingeniero en computación es alrededor de USD 1,500 [32] y que se trabajan 8 horas por 5 días a la semana, la ganancia promedio por hora sería de USD 9.38. Teniendo el valor de horas de trabajo para el proyecto integrador y la ganancia promedio por hora de un ingeniero en computación en Ecuador, se obtiene que el costo total del proyecto es de USD 5,060 (ver Figura 3.2).

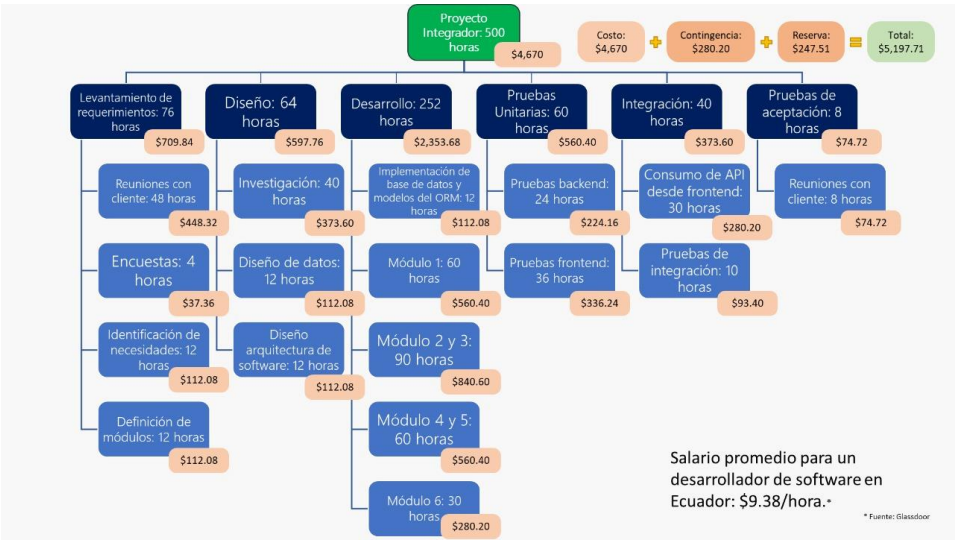


Figura 3.2 Costos por hora de trabajo en proyecto integrador

En cuanto a los costos incurridos durante el desarrollo, los mismos son inexistentes ya que se tomó provecho del nivel gratuito de ElasticBeanstalk de Amazon Web Services para el *hosting* del servidor.

3.6 Casos de Uso

En la extensión de Chromium existen tres usuarios involucrados: estudiante, profesor y administrador (ver Figura 3.3). El usuario estudiante no está vinculado directamente con el resto de los usuarios del sistema, pero la información que visualiza en su interfaz depende del mantenimiento por parte del administrador y el contenido creado por el profesor.

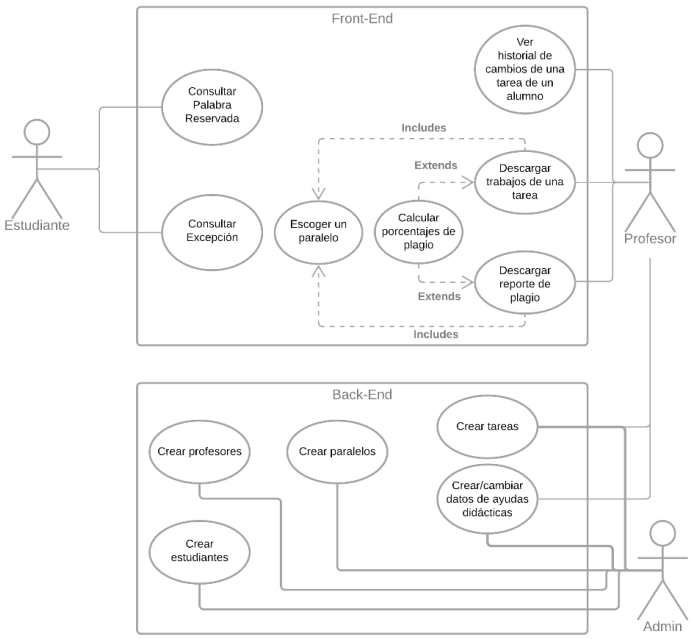


Figura 3.3 Diagrama de casos de uso

Los administradores tienen varias acciones para el mantenimiento del *CMS*: crear tareas, editar datos de ayuda didáctica, crear paralelos, crear profesores y crear estudiantes. Todas las acciones suelen ser realizadas al inicio del semestre como parte de rutina de mantenimiento de la extensión de Chromium, a excepción de la edición de datos de ayuda didáctica debido que algún profesor puede requerir cambios emergentes en los textos visualizados por los estudiantes de ser necesario o la creación de tareas para actualizar las tareas planificadas por el profesor.

Los estudiantes tienen dos acciones: consultar palabra reservada y consultar excepción. Cuando los estudiantes buscan una palabra, por medio de cualquiera de los dos contenedores de su interfaz, se realiza una de sus posibles acciones y se muestra información relevante que previamente ha sido editada por el profesor.

Finalmente, los profesores tienen tres acciones principales: mostrar historial de cambio, descargar trabajos de una tarea y descargar reporte de plagio. La descarga de tareas y reportes de plagio está vinculada al calculo del porcentaje de plagio y a la selección de paralelo, por lo que las descargas se realizan solo si se cumplieron las acciones previamente mencionadas.

CAPÍTULO 4

4. CONCLUSIONES Y RECOMENDACIONES

4.1 Conclusiones

- El uso de una herramienta de recolección de datos (Microsoft Forms) nos permitió identificar los puntos a fortalecer con la funcionalidad de la extensión y que esta funcionalidad adicional sea relevante tanto para profesores como para estudiantes en consideración a los requerimientos definidos en el producto de *software*.

- La creación de una extensión de Chromium es similar a la creación de una aplicación web, con la diferencia que se necesita un archivo *manifest* para que el empaquetador de extensiones de Google Chrome pueda leer el código fuente. Por esta misma razón, el mantenimiento de la extensión solo necesita personas con conocimientos básicos en desarrollo *web* para cualquier cambio que sea necesario referente a las interfaces de los usuarios.

- Debido que Django es conocido en el área de desarrollo web y su página predeterminada de administración cumple con las necesidades del sistema, vemos que fue la decisión correcta para el *framework* de desarrollo. Es también un ambiente en el que, en conjunto con Django REST Framework (DRF), se puede usar las credenciales brindadas solo a las personas encargadas del mantenimiento de la extensión junto a la página predeterminada de Django para la administración de información del *CMS* y las credenciales manejadas por los desarrolladores mediante la funcionalidad de autenticación de DRF para alimentar la información al *frontend* de la extensión.

- El encontrar una manera de suplementar la funcionalidad de Repl.it fue complicado debido a la confusa disposición del HTML de Repl.it y su interacción con Javascript. Por esta razón, el brindar información a los estudiantes bajo demanda fue la mejor alternativa para asegurar el funcionamiento a largo plazo, independencia de Repl.it y alta consistencia.

- Para la integración de las herramientas de la extensión dentro de las páginas web de Repl.it se necesitó de *Scrappers*. Sin el uso de estos, todas las funcionalidades de la extensión tendrían que haberse implementado dentro de la ventana de la extensión y no tendrían interacción significativa con la herramienta en línea Repl.it. Por esta razón, la facilidad que Python brinda para desarrollar *Scrappers* en comparación con otros lenguajes fue una de las razones para usarlo como el lenguaje de desarrollo para el *backend* de la extensión.

- Debido que los profesores de FP han trabajado previamente con MOSS, nos brindó la oportunidad de incorporar una herramienta sin dificultades de uso. El que tenga un cliente de Python nos dio una manera sencilla de integrarlo al proyecto, por lo que la funcionalidad del módulo seis y el objetivo cuatro se desarrolló sin mayor complicación.

- Finalmente, la extensión de Chromium desarrollada en esta investigación cumplió el objetivo general y las necesidades del cliente como fue detallado en el capítulo uno. Se realizó una interfaz para los usuarios involucrados y se cumplieron los requerimientos establecidos con las funcionalidades existentes, ayudando a los estudiantes a programar con mayor facilidad y a los profesores facilitar el proceso de evaluación de los trabajos enviados por los estudiantes.

4.2 Recomendaciones

- Los trabajos de los estudiantes y los informes de plagio descargados por el profesor se guardan con sus nombres correspondientes en base a los nombres asignados en Repl.it, por lo que la organización de los archivos corresponde a la forma en que los estudiantes nombren sus trabajos individuales. Se recomienda que los profesores que usen la extensión den un formato para nombrar las tareas enviadas de manera que el docente pueda identificar y calificar cada trabajo descargado sin dificultades. Este formato se puede extender a incluir el nombre de los equipos y de los paralelos, de esta forma más partes de la herramienta se podrían automatizar.

- Los reportes de plagio son generados por el sistema para detectar plagio MOSS, por lo que la velocidad para obtener los informes depende de los recursos en la nube que use Stanford para mantener el sistema. Debido a la pandemia de COVID-19 se ha aumentado la demanda de sistemas de detección de plagio, por lo que MOSS presenta fallas o demoras constantes, se recomienda que las personas encargadas del mantenimiento de la extensión busquen alternativas de sistemas de detección de plagio, o en su defecto que se establezcan protocolos para el uso de las funcionalidades que involucren la conexión a MOSS, esto requeriría trabajo adicional fuera del enfoque del proyecto pero puede ser investigación futura.

- No se necesitan conocimientos avanzados sobre desarrollo web para el mantenimiento o actualizaciones de funciones en la extensión de Chromium, por lo que se recomienda que el futuro desenvolvimiento de la extensión se asigne como un proyecto de prácticas comunitarias. Este proyecto se puede asignar a un grupo de estudiantes de la carrera Ingeniería en Computación que estén cursando o han cursado la materia Desarrollo de Aplicaciones Web y Móviles. Para el mantenimiento del *backend* se recomienda haber cursado la materia de Análisis de Algoritmos.

- En adición a la recomendación anterior, se recomienda que para desarrollos futuros un ejercicio de alto interés académico sería realizar un *port* del *frontend* de la extensión a un compilador de Javascript y analizar la diferencia en rendimiento. La recomendación personal de los autores es un compilador llamado Svelte.

BIBLIOGRAFÍA

[1] H.-W. Jung, «Revisiting to the necessity of programming Knowledge for Non-Computer Major Undergraduates,» *The Journal of the Convergence on Culture Technology,* vol. 6, nº 1, pp. 185-190, 2020.

[2] K. Powers, P. Gross, S. Cooper, M. McNally, K. J. Goldman, V. Proulx y M. Carlisle, «Tools for teaching introductory programming: what works?,» *Proceedings of the 37th SIGCSE technical symposium on Computer science education,* p. 560–561, 2006.

[3] A. Stefik y R. E. Ladner, «Introduction to AccessCS10K and Accessible Tools for Teaching Programming,» *Proceedings of the 46th ACM Technical Symposium on Computer Science Education,* p. 518–519, 2015.

[4] T. Jenkins, «TEACHING PROGRAMMING – A JOURNEY FROM TEACHER TO MOTIVATOR,» de *The 2nd Annual Conference of the LSTN Center for Information and Computer Science*, Leeds, 2001.

[5] I. Škorić, T. Orehovački y M. Ivašić-Kos, «Exploring the Acceptance of the Web-Based Coding Tool in an Introductory Programming Course: A Pilot Study,» de *International Conference on Human Interaction and Emerging Technologies*, 2020.

[6] N. Lazebna, «ENGLISH-LANGUAGE BASIS OF PYTHON PROGRAMMING LANGUAGE,» *Research Bulletin. Series: Philological Sciences,* pp. 371-376, 2021.

[7] A. J. Gomes y A. J. Mendes, «A Study on Student Performance in First Year CS Courses,» de *Proceedings of the Fifteenth Annual Conference on Innovation and Technology in Computer Science Education*, Bilkent, 2010.

[8] M. Ateeq, H. Habib, A. Umer y M. Rehman, «C++ or Python? Which One to Begin with: A Learner's Perspective,» de *2014 International Conference on Teaching and Learning in Computing and Engineering*, 2014.

[9] M. P. Uysal, «Improving first computer programming experiences: The case of adapting a web-supported and well-structured problem-solving method to a traditional course,» *Contemporary Educational Technology,* vol. 5, pp. 198-217, 2014.

[10] R. P. Medeiros, G. L. Ramalho y T. P. Falcão, «A Systematic Literature Review on Teaching and Learning Introductory Programming in Higher Education,» *IEEE Transactions on Education,* vol. 62, pp. 77-90, 2019.

[11] R. I. Bonilla, E. Lozano y R. Granda, «Pyweekend: Not Your Typical Hackathon,» de *2019 IEEE Global Engineering Education Conference (EDUCON)*, 2019.

[12] R. I. Bonilla, R. Granda y E. Lozano, «Effects of a Hackathon on the Motivation and Grades of CS1 Students,» de *2020 IEEE Global Engineering Education Conference (EDUCON)*, 2020.

[13] B. Haratano, «Enhancing the student engagement in an introductory programming: A holistic approach in improving the student grade in the informatics Department of the University of Surabaya,» de *International Conference on Soft Computing, Intelligence Systems, and Information Technology*, 2015.

[14] D. Horton y M. Craig, «Drop, Fail, Pass, Continue: Persistence in CS1 and Beyond in Traditional and Inverted Delivery,» de *Proceedings of the 46th ACM Technical Symposium on Computer Science Education*, 2015.

[15] O. Seppälä, P. Ihantola, E. Isohanni, J. Sorva y A. Vihavainen, «Do We Know How Difficult the Rainfall Problem Is?,» de *Proceedings of the 15th Koli Calling Conference on Computing Education Research*, 2015.

[16] S. D. D'Souza, J. Sheard, J. Harland, A. Carbone y M.-J. Laakso, «Can Computing Academics Assess the Difficulty of Programming Examination Questions?,» de *Proceedings of the 12th Koli Calling International Conference on Computing Education Research*, 2012.

[17] A. Gomes y A. Mendes, «A teacher's view about introductory programming teaching and learning: Difficulties, strategies and motivations,» de *2014 IEEE Frontiers in Education Conference (FIE) Proceedings*, 2014.

[18] T. Koulouri, S. Lauria y R. D. Macredie, «Teaching Introductory Programming: A Quantitative Evaluation of Different Approaches,» *ACM Trans. Comput. Educ.,* vol. 14, p. 28, febrero 2015.

[19] EF, «Índice del EF English Proficiency,» EF, 2020. [En línea]. Available: https://www.ef.com.ec/epi/. [Último acceso: 24 octubre 2021].

[20] P. J. Guo, «Non-Native English Speakers Learning Computer Programming: Barriers, Desires, and Design Opportunities,» de *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, 2018.

[21] B. A. Becker, G. Glanville, R. Iwashima, C. McDonnell, K. Goslin y C. Mooney, «Effective compiler error message enhancement for novice programming students,» *Computer Science Education,* vol. 26, pp. 148-175, 2016.

[22] D. McCall y M. Kölling, «Meaningful categorisation of novice programmer errors,» de *2014 IEEE Frontiers in Education Conference (FIE) Proceedings*, 2014.

[23] N. Ramakrishnan, *Breaking the language barrier in programming,* Bussiness Line, 2019.

[24] A. Bezrukov y J. Ziyatdinova, «Internationalizing engineering education: A language learning approach,» de *2014 International Conference on Interactive Collaborative Learning (ICL)*, 2014.

[25] R. Berton, A. Kolasinska, O. Gaggi, C. E. Palazzi y G. Quadrio, «A Chrome Extension to Help People with Dyslexia,» de *Proceedings of the International Conference on Advanced Visual Interfaces*, 2020.

[26] D. Heres y J. Hage, «A Quantitative Comparison of Program Plagiarism Detection Tools,» de *Proceedings of the 6th Computer Science Education Research Conference*, 2017.

[27] G. Lucassen, F. Dalpiaz y J. M. van der Wer, «The Use and Effectiveness of User Stories in Practice,» de *Requirements Engineering: Foundation for Software Quality*, Utrecht, 2016.

[28] Google, *Extension development overview,* Google, 2018.

[29] D. Heres y J. Hage, «A Quantitative Comparison of Program Plagiarism Detection Tools,» de *Proceedings of the 6th Computer Science Education Research Conference*, Helsinki, 2017.

[30] D. J. Fernandez y J. D. Fernandez, «Agile Project Management —Agilism versus Traditional Approaches,» *Journal of Computer Information Systems ,* vol. 49, nº 2, pp. 10-17, 2008.

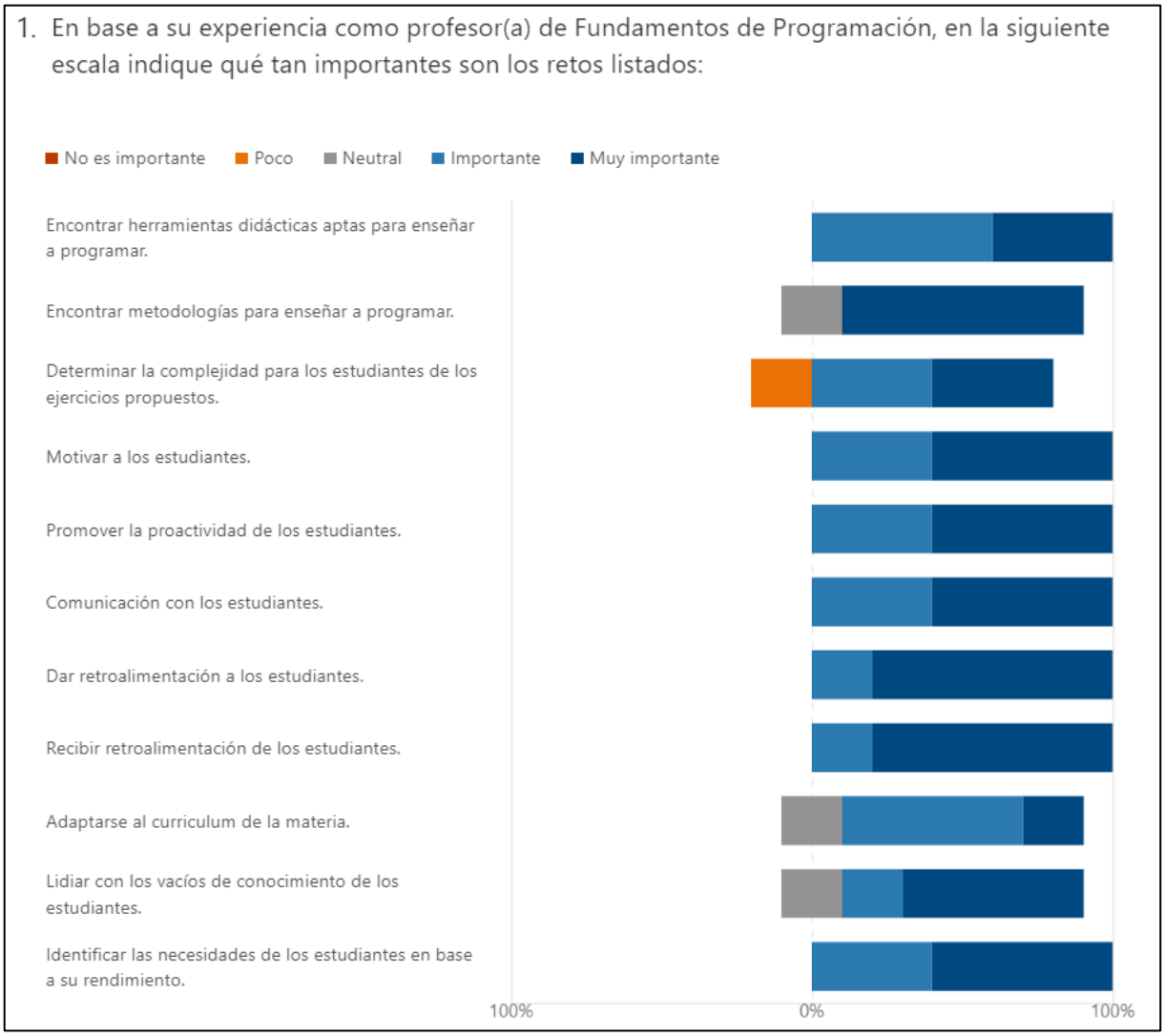
[31] U. A. J. For, «Fisher, John; D. Koning; A. P. Ludwigsen,» de *Lawrence Livermore National Lab.(LLNL)*, Livermore, CA (United States), 2013.

[32] «Glassdoor,» [En línea]. Available: https://www.glassdoor.com.mx/Sueldos/quito-software-developer-sueldo-SRCH_IL.0,5_IM1362_KO6,24.htm. [Último acceso: 25 01 2022].

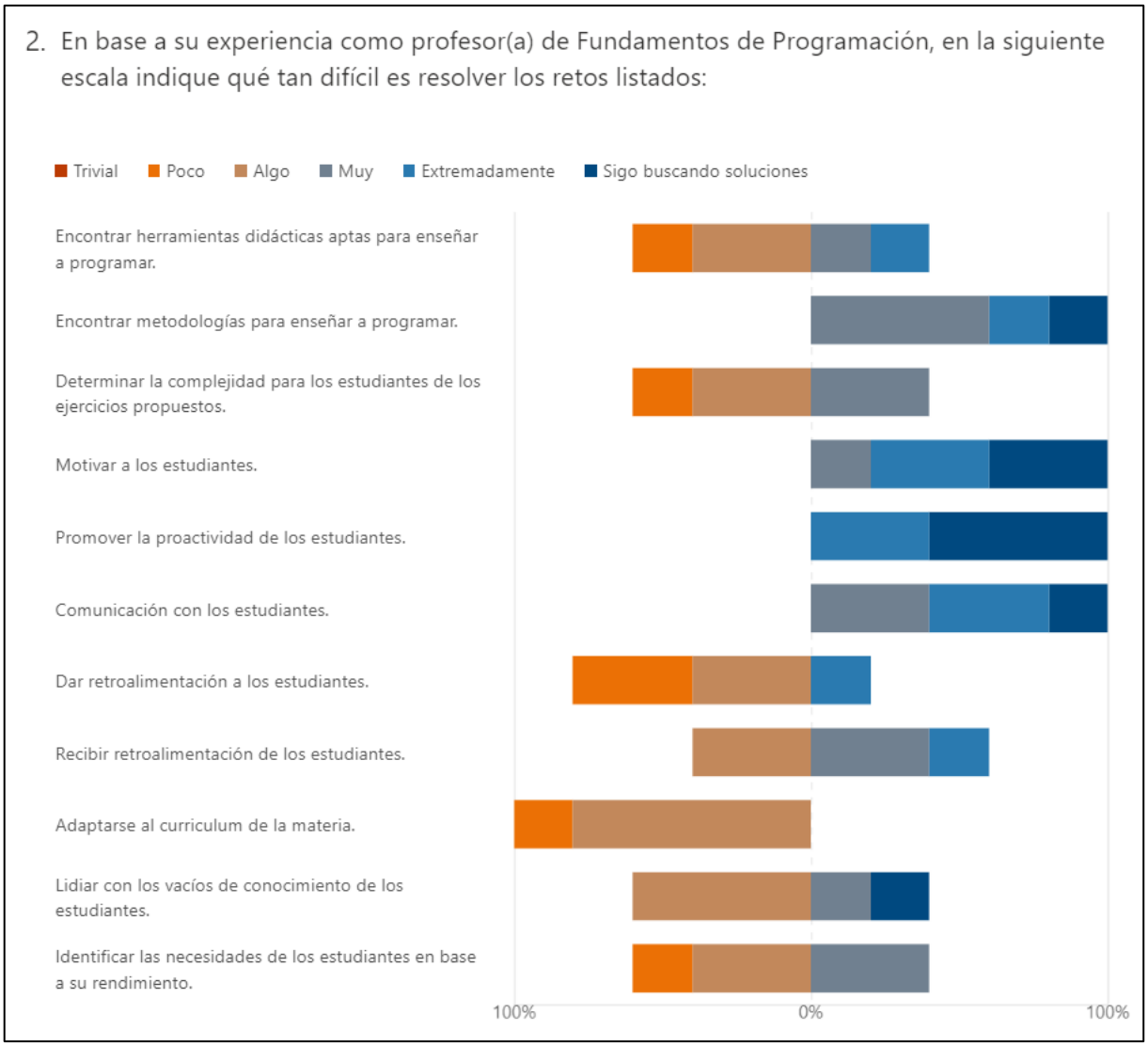
APÉNDICES

APÉNDICE A. Encuesta para busqueda de requerimientos

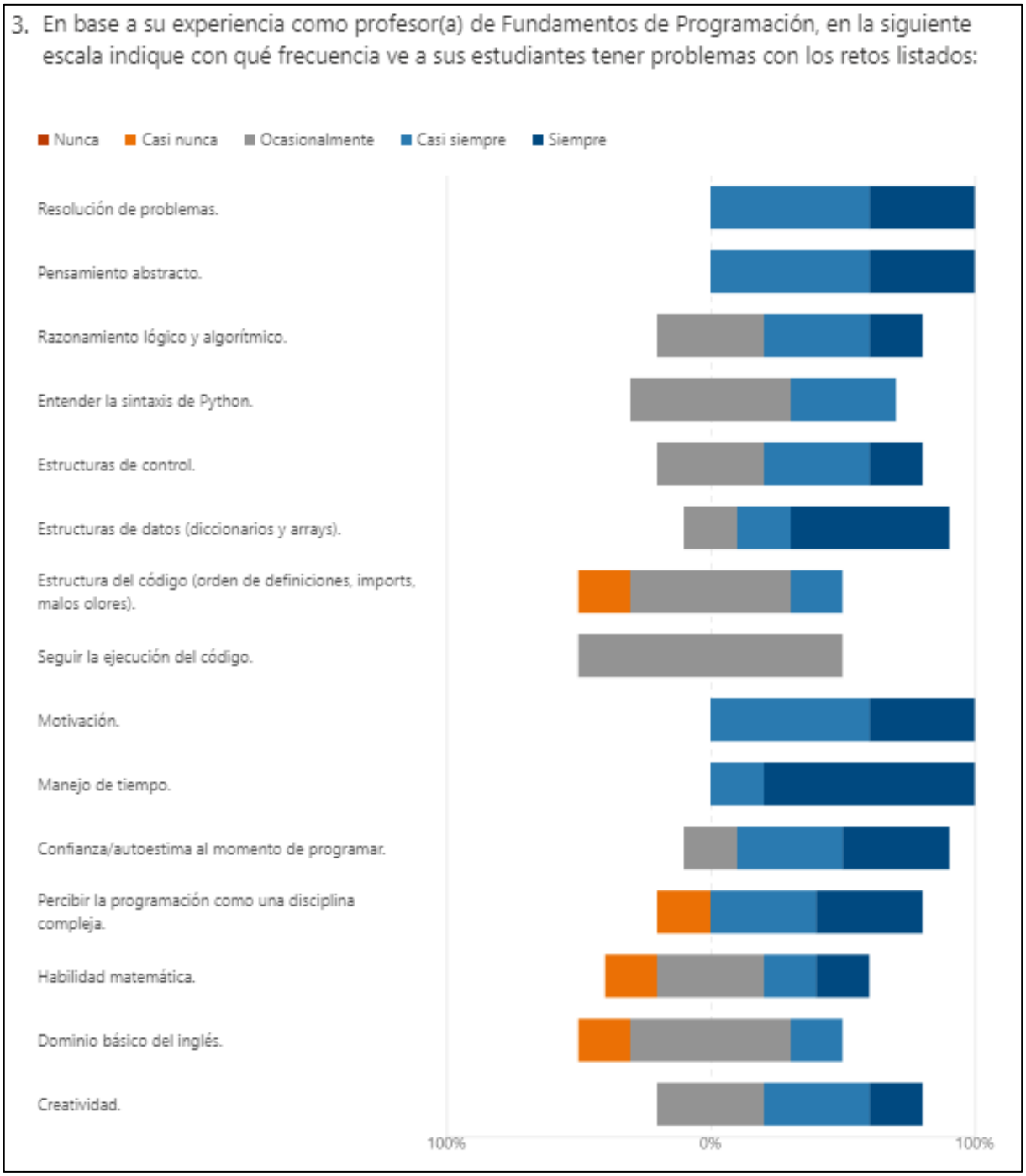
A continuación, se presenta el cuestionario para profesores de fundamentos de programación para la búsqueda de los requerimientos del sistema desarrollado en esta investigación.



Pregunta 1

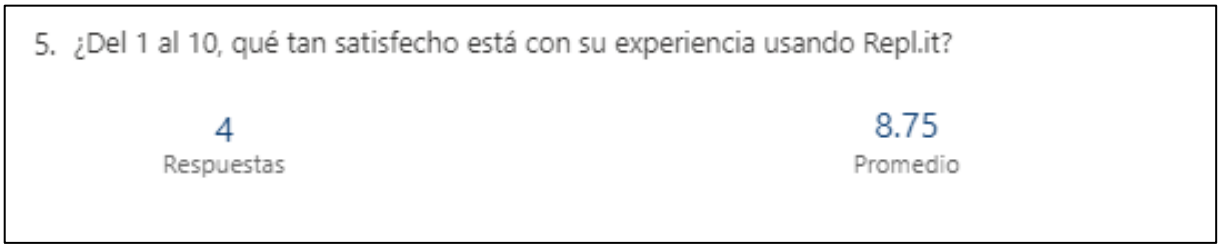

2. En base a su experiencia como profesor(a) de Fundamentos de Programación, en la siguiente escala indique qué tan difícil es resolver los retos listados:
Trivial
Poco
Algo
Muy
Extremadamente
Sigo buscando soluciones
Encontrar herramientas didácticas aptas para enseñar a programar.
Encontrar metodologías para enseñar a programar.
Determinar la complejidad para los estudiantes de los ejercicios propuestos.
Motivar a los estudiantes.
Promover la proactividad de los estudiantes.
Comunicación con los estudiantes.
Dar retroalimentación a los estudiantes.
Recibir retroalimentación de los estudiantes.
Adaptarse al curriculum de la materia.
Lidiar con los vacíos de conocimiento de los estudiantes.
Identificar las necesidades de los estudiantes en base a su rendimiento.
100%
0%
100%

Pregunta 2

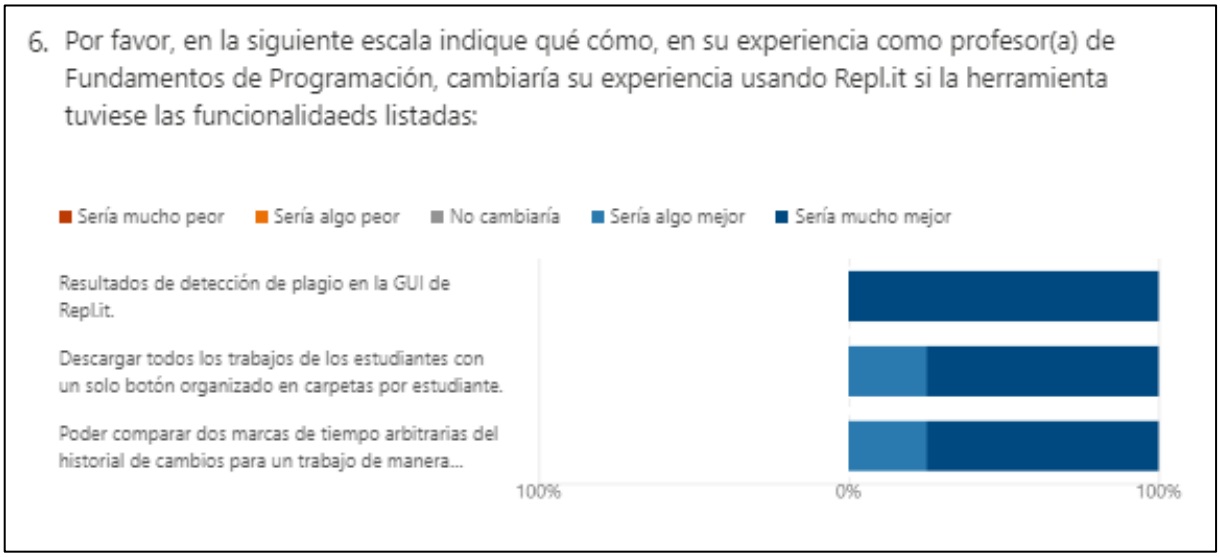

3. En base a su experiencia como profesor(a) de Fundamentos de Programación, en la siguiente escala indique con qué frecuencia ve a sus estudiantes tener problemas con los retos listados:
Nunca
Casi nunca
Ocasionalmente
Casi siempre
Siempre
Resolución de problemas.
Pensamiento abstracto.
Razonamiento lógico y algorítmico.
Entender la sintaxis de Python.
Estructuras de control.
Estructuras de datos (diccionarios y arrays).
Estructura del código (orden de definiciones, imports, malos olores).
Seguir la ejecución del código.
Motivación.
Manejo de tiempo.
Confianza/autoestima al momento de programar.
Percibir la programación como una disciplina compleja.
Habilidad matemática.
Dominio básico del inglés.
Creatividad.
100%
0%
100%

Pregunta 3

Pregunta 4



Pregunta 5



Pregunta 6