Escuela Superior Politécnica del Litoral

Facultad de Ingeniería en Electricidad y Computación

Desarrollo de un módulo de retroalimentación automática para el Aula Virtual utilizando inteligencia artificial

TECH-385

Proyecto Integrador

Previo la obtención del Título de:

Ingeniero en Ciencias de la Computación

Presentado por:

Freddy Alexander Gómez Urrego

Danilo Adrian Torres Vera

Guayaquil - Ecuador

Año: 2024

Dedicatoria

El presente proyecto se lo dedico a mi madre Mirian y a mi padre Fredi por apoyarme en toda mi carrera universitaria, motivarme a ser mejor y a mi novia Jesmy que me brindó su constante apoyo cuando más lo necesitaba.

Freddy Alexander Gómez Urrego

Dedicatoria

El presente proyecto lo dedico a mis padres Danilo y Patricia pilares fundamentales en mi carrera, mis hermanos Jonathan y Gabriela, espero ser ejemplo para ellos, mis abuelos, tíos que me han apoyado, a mis profesores que me han enriquecido con sus conocimientos en mi formación académica. A todos los amigos que me acompañaron desde un principio, por haber sido una guía y un apoyo constante en mi camino.

Danilo Adrián Torres Vera

Agradecimientos

Nuestro más sincero agradecimiento al profesor Ing. Eduardo Cruz por su guía y apoyo para el desarrollo de este proyecto y el profesor M.Sc. Ronald Criollo por su guía en la escritura del presente documento.

Declaración Expresa

Nosotros Freddy Alexander Gómez Urrego y Danilo Adrián Torres Vera acordamos y reconocemos que:

La titularidad de los derechos patrimoniales de autor (derechos de autor) del proyecto de graduación corresponderá al autor o autores, sin perjuicio de lo cual la ESPOL recibe en este acto una licencia gratuita de plazo indefinido para el uso no comercial y comercial de la obra con facultad de sublicenciar, incluyendo la autorización para su divulgación, así como para la creación y uso de obras derivadas. En el caso de usos comerciales se respetará el porcentaje de participación en beneficios que corresponda a favor del autor o autores.

La titularidad total y exclusiva sobre los derechos patrimoniales de patente de invención, modelo de utilidad, diseño industrial, secreto industrial, software o información no divulgada que corresponda o pueda corresponder respecto de cualquier investigación, desarrollo tecnológico o invención realizada por mí/nosotros durante el desarrollo del proyecto de graduación, pertenecerán de forma total, exclusiva e indivisible a la ESPOL, sin perjuicio del porcentaje que me/nos corresponda de los beneficios económicos que la ESPOL reciba por la explotación de mi/nuestra innovación, de ser el caso.

En los casos donde la Oficina de Transferencia de Resultados de Investigación (OTRI) de la ESPOL comunique al/los autor/es que existe una innovación potencialmente patentable sobre los resultados del proyecto de graduación, no se realizará publicación o divulgación alguna, sin la autorización expresa y previa de la ESPOL.

Guayaquil, 8 de Octubre del 2024.

Freddy Gómez

Danilo Torres

Evaluadores

Firmado electrónicamente por:
RONALD RAUL CRIOLLO BONILLA

M.Sc Ronald Raúl Criollo Bonilla

Profesor de Materia

Firmado electrónicamente por:
RONALD RAUL CRIOLLO BONILLA

M.Sc Ronald Raúl Criollo Bonilla

Tutor de proyecto

Resumen

El proyecto desarrolla un módulo de retroalimentación automática basado en inteligencia artificial para la plataforma educativa Aula Virtual de ESPOL, con el objetivo de mejorar la evaluación de tareas de programación en Python en la carrera de Ciencias de Datos e Inteligencia Artificial. La iniciativa responde a la creciente demanda de retroalimentación personalizada y oportuna, optimizando los recursos docentes mediante la integración de IA.

Para su desarrollo, se utilizaron estándares LTI para la integración con Canvas, aprendizaje automático para analizar sintaxis, lógica y eficiencia del código, y procesamiento de lenguaje natural para generar retroalimentación comprensible. Se realizaron pruebas con docentes y estudiantes para garantizar la precisión y utilidad de las retroalimentaciones.

Los resultados demuestran una reducción significativa en el tiempo de evaluación, con un 87% de usuarios satisfechos con la claridad de las retroalimentaciones y una aceptación del 100% en términos de aprendizaje de nuevos conceptos. El sistema mejora la experiencia educativa al proporcionar retroalimentación detallada, personalizada y en tiempo real.

Se concluye que la solución no solo optimiza la carga docente, sino que también promueve un aprendizaje profundo y eficaz, posicionando a ESPOL como pionera en la integración de IA en la educación.

Palabras Clave: Aprendizaje Automático, Sistemas Educativos, Evaluación Automatizada

Abstract

The project develops an automatic feedback module using artificial intelligence for ESPOL's Aula Virtual educational platform. Its primary goal is to enhance the evaluation of Python programming assignments in the Data Science and Artificial Intelligence program. The initiative addresses the increasing need for personalized and timely feedback, optimizing teaching resources through AI integration.

The development utilized LTI standards for Canvas integration, machine learning techniques to analyze code syntax, logic, and efficiency, and natural language processing to generate understandable feedback. Tests with faculty and students ensured the precision and effectiveness of the feedback.

Results showed a significant reduction in evaluation time, with 87% of users satisfied with feedback clarity and 100% of students reporting meaningful learning improvements. The system enhances the educational experience by providing detailed, personalized, and real-time feedback.

It is concluded that the solution not only reduces teaching workloads but also fosters deep and effective learning, positioning ESPOL as a leader in AI integration in education.

Keywords: Machine Learning, Educational Systems, Automated Assessment.

Índice general

Abreviaturas

ESPOL Escuela Superior Politécnica del Litoral

LTI Interoperabilidad de Herramientas de Aprendizaje

NLP Procesamiento de Lenguaje Natural

ITS Sistemas de Tutorización Inteligente

Índice de figuras

Índice de tablas

Capítulo 1

1.1 Introducción

En la era digital actual, la educación superior se encuentra en un punto de inflexión, enfrentando desafíos sin precedentes en la forma de impartir conocimientos y evaluar el aprendizaje de los estudiantes. La creciente demanda de educación en línea, ha puesto de manifiesto la necesidad de herramientas innovadoras que puedan mejorar la calidad de la enseñanza y el aprendizaje en entornos virtuales. Como señalan Garrison y Kanuka (2004), "el potencial transformador del aprendizaje mixto (combinación de aprendizaje presencial y en línea) en la educación superior es vasto" [1].

En este contexto, la ESPOL de Ecuador, ha identificado una oportunidad crucial para mejorar la experiencia educativa de sus estudiantes, particularmente en áreas técnicas como programación, ciencias de datos y aprendizaje automático. La institución utiliza una plataforma de gestión del aprendizaje (LMS) basada en Canvas, conocida como Aula Virtual, que sirve como el principal medio de interacción entre estudiantes y profesores en el entorno digital.

En la carrera de Ciencias de Datos e Inteligencia Artificial de ESPOL, las materias relacionadas con la programación, como "Machine Learning", "Fundamentos de Programación" o "Ciencias de Datos", requieren que los estudiantes desarrollen tareas y proyectos que involucran la implementación de soluciones algorítmicas mediante código, particularmente en Python. Proporcionar retroalimentación detallada, precisa y oportuna sobre estas implementaciones es esencial para el proceso de aprendizaje, ya que permite a los estudiantes identificar sus errores, mejorar la lógica de sus algoritmos, y adoptar mejores prácticas de programación.

Sin embargo, a medida que el número de estudiantes en estos cursos técnicos aumenta, los profesores y ayudantes se enfrentan a un desafío significativo: proporcionar retroalimentación individual a cada estudiante sobre sus trabajos y códigos de programación. Este proceso,

fundamental para el aprendizaje efectivo, se ha vuelto cada vez más demandante en términos de tiempo y recursos.

La importancia de la retroalimentación en el proceso de aprendizaje no puede subestimarse. Hattie y Timperley (2007) argumentan que "la retroalimentación es una de las influencias más poderosas en el aprendizaje y el rendimiento" [3]. En el contexto específico de la enseñanza de la programación, la retroalimentación juega un papel aún más crucial. Según un estudio realizado por Keuning et al. (2018), "la retroalimentación automatizada en la enseñanza de la programación puede mejorar significativamente el rendimiento de los estudiantes y reducir la carga de trabajo de los instructores" [4].

El proceso de proporcionar retroalimentación individual a cada estudiante en estas materias puede ser extremadamente demandante para los profesores y ayudantes. Esto se debe a la necesidad de evaluar no solo la corrección sintáctica del código, sino también su eficiencia, claridad y la calidad de la lógica utilizada. A medida que el número de estudiantes matriculados en estas materias aumenta, los docentes enfrentan mayores cargas de trabajo, lo que puede llevar a varios problemas como demoras en la retroalimentación, una retroalimentación muy superficial o la falta de entendimiento de la retroalimentación en los estudiantes.

Es en este escenario donde surge la necesidad de desarrollar un módulo de retroalimentación automática basado en inteligencia artificial (IA) que pueda integrarse perfectamente con el Aula Virtual de ESPOL.

1.2 Descripción del Problema

En la carrera de Ciencias de Datos e Inteligencia Artificial, las materias relacionadas con la programación, como "Machine Learning", "Fundamentos de Programación" y "Ciencias de Datos", requieren que los estudiantes desarrollen tareas y proyectos que involucran la implementación de soluciones algorítmicas mediante código, particularmente en Python.

Proporcionar retroalimentación detallada, precisa y oportuna sobre estas implementaciones es esencial para el proceso de aprendizaje, ya que permite a los estudiantes identificar sus errores, mejorar la lógica de sus algoritmos, y adoptar mejores prácticas de programación.

Sin embargo, el proceso de proporcionar retroalimentación individual a cada estudiante en estas materias puede ser extremadamente demandante para los profesores y ayudantes. Esto se debe a la necesidad de evaluar no solo la corrección sintáctica del código, sino también su eficiencia, claridad y la calidad de la lógica utilizada. A medida que el número de estudiantes matriculados en estas materias aumenta, los docentes enfrentan mayores cargas de trabajo, lo que puede llevar a varios problemas como demoras en la retroalimentación, una retroalimentación muy superficial o la falta de entendimiento de la retroalimentación en los estudiantes.

La incorporación de un sistema de retroalimentación automatizada basado en inteligencia artificial puede mitigar estos problemas.

1.3 Justificación del Problema

La calidad de la retroalimentación en el proceso educativo es un factor determinante en el aprendizaje de los estudiantes, especialmente en áreas técnicas como la programación, donde la precisión y la claridad son fundamentales. En la carrera de Ciencias de Datos e Inteligencia Artificial de ESPOL, materias como "Fundamentos de Programación" y "Ciencias de Datos" requieren que los estudiantes implementen soluciones algorítmicas a problemas complejos mediante código, particularmente en Python. Sin embargo, la tarea de revisar manualmente cada una de estas entregas presenta importantes desafíos.

El aumento del número de estudiantes y la creciente complejidad de las tareas plantean una carga significativa sobre los profesores y ayudantes, quienes deben proporcionar retroalimentación detallada, tanto a nivel de sintaxis como de lógica y eficiencia del código. Este proceso, al realizarse de forma manual, es altamente demandante en tiempo y recursos humanos, lo que con frecuencia resulta en demoras en las evaluaciones y retroalimentación que, en muchos casos, puede ser superficial o no suficientemente personalizada.

Una retroalimentación tardía o poco precisa tiene consecuencias directas sobre el aprendizaje de los estudiantes, quienes dependen de estos comentarios para corregir errores, mejorar la calidad

de su código y adquirir mejores prácticas de programación. Además, la falta de retroalimentación detallada limita su capacidad para identificar debilidades en la comprensión de los conceptos fundamentales que deben dominar para avanzar en su formación académica y profesional.

Dado este escenario, es necesario buscar soluciones que optimicen el proceso de evaluación y retroalimentación, sin comprometer la calidad ni la personalización de los comentarios proporcionados a los estudiantes. La inteligencia artificial ofrece una oportunidad única para automatizar la primera fase de este proceso. Un módulo de retroalimentación automática, basado en IA, podría analizar las entregas de código de forma eficiente y precisa, proporcionando comentarios inmediatos sobre la sintaxis, la lógica y la eficiencia del código. De este modo, los profesores y ayudantes podrían centrarse en revisar casos más complejos y en personalizar la retroalimentación donde sea necesario.

1.4 Objetivos

1.4.1 Objetivo general

Desarrollar un módulo de retroalimentación automática basado en inteligencia artificial para la plataforma de Aula Virtual basada en Canvas, que evalúe código proporcionando retroalimentación y que permita ser revisada y personalizada por profesores y ayudantes de las materias en la carrera de Ciencias de Datos e Inteligencia Artificial.

1.4.2 Objetivos específicos

1. Diseñar un motor de inteligencia artificial para el análisis de códigos de programación en Python, considerando aspectos sintácticos, la lógica y la eficiencia de las soluciones propuestas por los estudiantes.
2. Desarrollar una herramienta LTI integrada con la plataforma Aula Virtual de ESPOL, que permita la retroalimentación automática de las tareas de programación de los estudiantes.

3. Crear una interfaz de usuario intuitiva para que los profesores y ayudantes revisen, editen y personalicen la retroalimentación generada por la IA antes de enviarla a los estudiantes.
4. Evaluar el módulo de retroalimentación automática para asegurar precisión, eficiencia y correcta integración con la plataforma Canvas.

1.5 Marco teórico

El presente proyecto se enmarca en la intersección de varios campos de estudio y tecnologías emergentes que están transformando la educación superior, particularmente en el ámbito de la enseñanza de la programación. En este contexto, es fundamental comprender los conceptos clave y las investigaciones previas que sustentan el desarrollo de un sistema de retroalimentación automática basado en inteligencia artificial (IA).

El aprendizaje automático y el procesamiento de lenguaje natural son dos áreas de la IA que han demostrado un gran potencial en el campo educativo. Zawacki-Richter et al. (2019) señalan que estas tecnologías están revolucionando la forma en que se imparte y evalúa la educación, especialmente en entornos en línea [4]. En el contexto específico de la evaluación de código, el aprendizaje automático ha permitido desarrollar sistemas capaces de identificar patrones de código correcto e incorrecto, yendo más allá de la simple comprobación sintáctica. Por ejemplo, Wang et al. (2018) desarrollaron un sistema basado en aprendizaje profundo capaz de detectar errores semánticos en código Python [5].

Por otro lado, el NLP juega un papel crucial en la generación de retroalimentación comprensible para los estudiantes. Keuning et al. (2019) demostraron cómo las técnicas de NLP pueden utilizarse para generar explicaciones detalladas de errores de programación, mejorando significativamente la comprensión de los estudiantes [3]. Esta capacidad de proporcionar retroalimentación clara y detallada es fundamental para el éxito de cualquier sistema de evaluación automática.

Los Sistemas de Tutorización Inteligente representan una aplicación avanzada de la IA en educación. Estos sistemas utilizan técnicas de IA para proporcionar instrucción y retroalimentación personalizada. VanLehn (2011) ha señalado que, en ciertas tareas educativas, los ITS pueden ser tan efectivos como los tutores humanos [6].

La retroalimentación generada por IA en el contexto de la programación ha ganado especial relevancia con el surgimiento de modelos de lenguaje más avanzados. Zhang et al.

(2024) realizaron un estudio significativo sobre las percepciones y preferencias de los estudiantes respecto a la retroalimentación generada por IA en tareas de programación [7]. Sus hallazgos subrayan la importancia de varios aspectos fundamentales en el diseño de sistemas de retroalimentación automatizada:

- La necesidad de explicaciones claras y detalladas que ayuden a los estudiantes a comprender no solo dónde están los errores, sino también por qué son errores.
- La importancia de proporcionar ejemplos concretos y contextualizados que ilustren tanto los problemas identificados como las soluciones propuestas.
- La preferencia por retroalimentación constructiva que vaya más allá de la identificación de errores, ofreciendo sugerencias específicas para mejorar el código.
- La valoración de un enfoque holístico que considere tanto aspectos técnicos como pedagógicos en la retroalimentación proporcionada.
-

La base teórica de este proyecto también se apoya en importantes teorías del aprendizaje. La teoría del andamiaje, propuesta por Wood, Bruner y Ross (1976), sugiere que los estudiantes aprenden mejor cuando reciben apoyo gradual que se va retirando a medida que ganan competencia [8]. Esta teoría es particularmente relevante en el diseño de sistemas de retroalimentación automática, ya que implica proporcionar explicaciones detalladas inicialmente, reduciendo gradualmente el nivel de detalle a medida que el estudiante progresa.

Asimismo, el concepto de retroalimentación formativa, destacado por Black y Wiliam (1998), es central en este proyecto [9]. En el contexto de la programación, la retroalimentación formativa puede incluir sugerencias para mejorar el código, explicaciones de conceptos mal entendidos, y orientación sobre mejores prácticas. Esta retroalimentación no solo ayuda a los

estudiantes a corregir errores, sino que también fomenta un aprendizaje más profundo y una mejor comprensión de los conceptos de programación.

La evaluación automatizada de programas es un campo de investigación activo en la educación en ciencias de la computación. Ihantola et al. (2010) proporcionaron una revisión exhaustiva de los sistemas de evaluación automática de programas, destacando sus beneficios y desafíos [10]. Estos sistemas suelen emplear una combinación de análisis estático (examinando el código sin ejecutarlo) y análisis dinámico (ejecutando el código con casos de prueba). Cada enfoque tiene sus ventajas: el análisis estático puede detectar problemas de estilo y potenciales errores, mientras que el análisis dinámico puede verificar la corrección funcional.

Capítulo 2

2.1 Metodología.

El presente proyecto abarca el diseño, desarrollo e implementación de un módulo de retroalimentación automática basado en inteligencia artificial, integrado con el Aula Virtual de ESPOL. El sistema se enfocará específicamente en la evaluación de código Python, siendo esta la principal herramienta de programación utilizada en las materias de la carrera de Ciencias de Datos e Inteligencia Artificial.

2.2 Análisis

2.2.1 Requerimientos Funcionales

El sistema debe:

- Analizar código Python en términos de sintaxis, eficiencia y lógica.
- Generar comentarios detallados por cada error detectado y recomendaciones generales para mejorar.
- Permitir la personalización de la retroalimentación por los profesores.
- Integrarse con Canvas usando el estándar LTI.
- Proporcionar acceso al historial de retroalimentación de cada estudiante.

2.2.2 Requerimientos No-Funcionales

El módulo debe ser:

- Escalable: Manejar múltiples cursos y grandes volúmenes de datos.
- Eficiente: Generar retroalimentación en tiempo real o con mínima latencia.
- Seguro: Proteger la información de los estudiantes y docentes según las normativas vigentes.
- Fácil de Usar: Contar con interfaces intuitivas y compatibles con diferentes dispositivos.

2.3 Alcance y limitaciones de la solución

2.3.1 Alcance:

- Foco en materias de programación dentro de la carrera de Ciencias de Datos e Inteligencia Artificial (Fundamentos de Programación – Ciencias de Datos).
- Análisis de código en Python únicamente.

2.3.2 Limitaciones:

- La precisión del análisis dependerá de la calidad de los prompts.
- La retroalimentación automatizada podría no abarcar casos extremadamente complejos, como el análisis de imágenes.

2.4 Riesgos y Beneficios

2.4.1 Riesgos:

- Dependencia tecnológica: Problemas de integración o fallos en el sistema podrían afectar su uso.
- Fallo de la Api de Lamas, por versionamiento o soporte.

2.4.2 Beneficios:

- La precisión del análisis dependerá de la calidad de los prompts.
- La retroalimentación automatizada podría no abarcar casos extremadamente complejos, como el análisis de imágenes.

2.5 Usuarios de la Solución

Profesor: Encargado de revisar y supervisar la calificación proporcionada automáticamente por la IA.

Acciones:

- Analizar la retroalimentación
- Modificar la retroalimentación y calificación en caso de ser necesario
- Enviar la retroalimentación a cada uno de los estudiantes

Estudiante: Receptor de la retroalimentación proporcionada por la IA y supervisada por el profesor

Acciones:

- Recibir retroalimentaciones de los trabajos
- Consultar retroalimentaciones pasadas

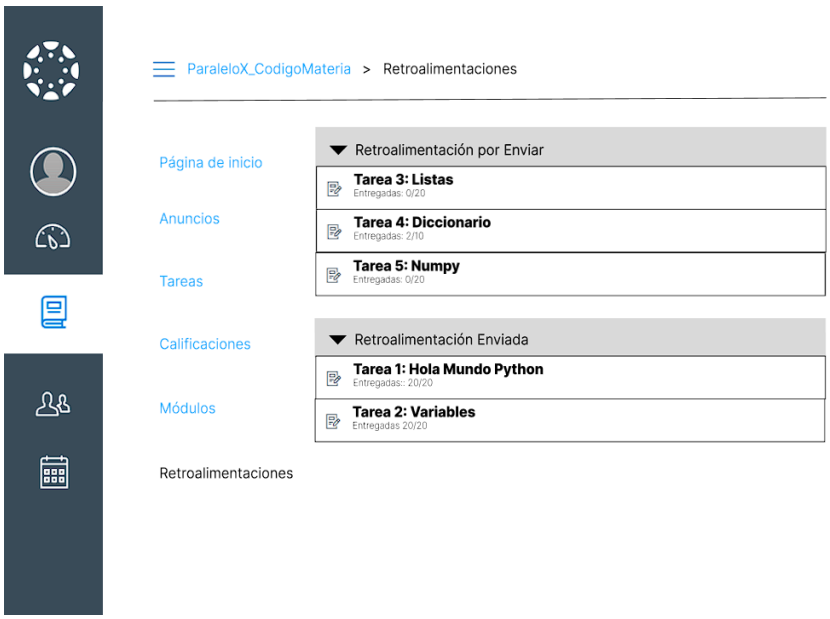
2.6 Prototipado

Se realizó el prototipado del módulo a agregar en la plataforma Figma, donde se diseñaron las siguientes vistas.

En la siguiente figura se presenta el apartado “Cursos” de la plataforma Aula Virtual con el nuevo módulo llamado “Retroalimentaciones” donde se enlistan las retroalimentaciones por revisar y enviar y por separado las retroalimentaciones ya enviadas y calificadas.

Figura 1

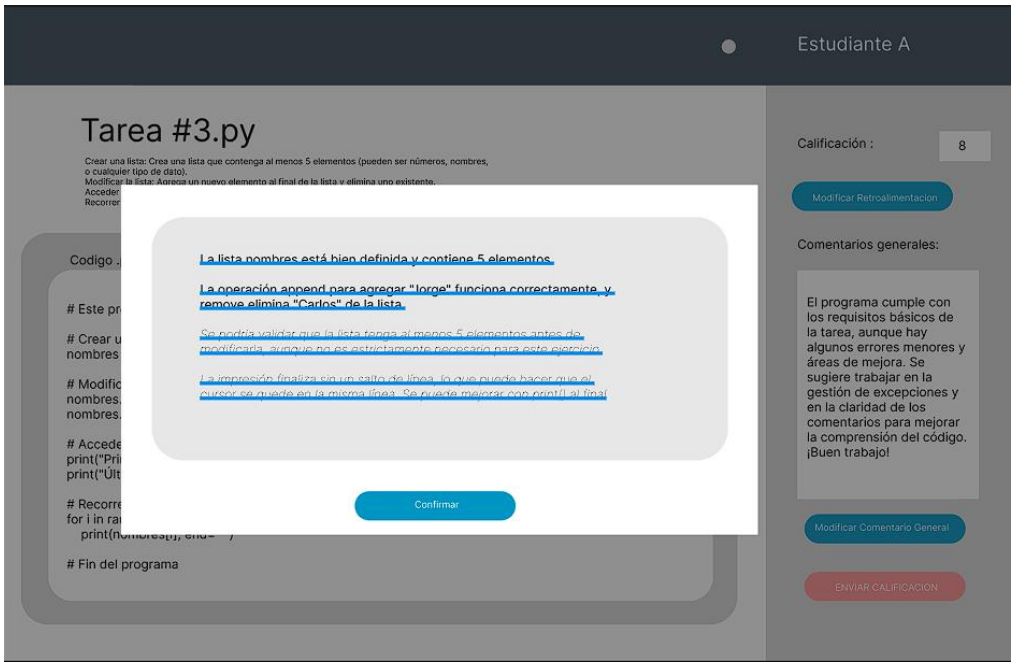
Pantalla de Vista Principal [Autoría propia]



En la siguiente figura se encuentra en el apartado de revisión de la retroalimentación donde los comentarios pueden ser modificados y enviados al estudiante.

Figura 2

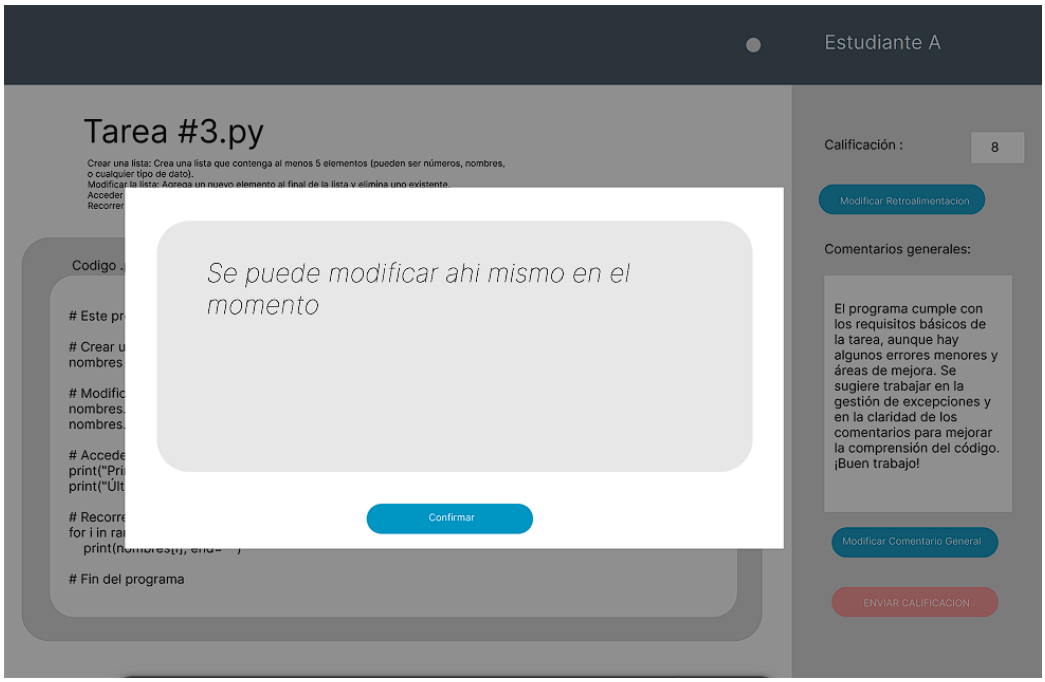
Pantalla de Revisión de Retroalimentación [Autoría propia]



En la siguiente figura se puede apreciar cómo será posible la edición de la retroalimentación del código, en el mismo momento.

Figura 3

Pantalla de Modificación de Retroalimentación [Autoría propia]



Vista de retroalimentación enviada al estudiante.

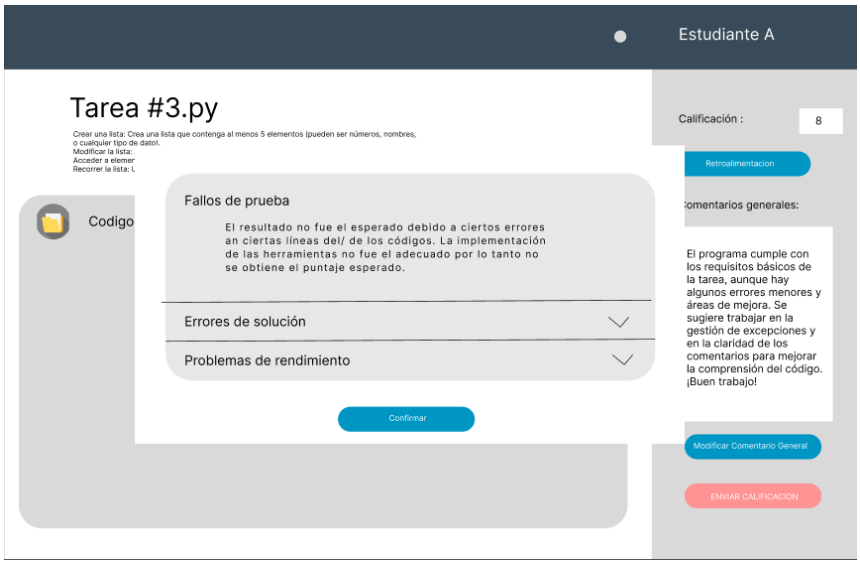
Figura 4

Pantalla de Estudiante [Autoría propia]

Vista de modificación de retroalimentación clasificando los comentarios, mostrando una sección.

Figura 5

Pantalla de Vista de Profesor [Autoría propia]

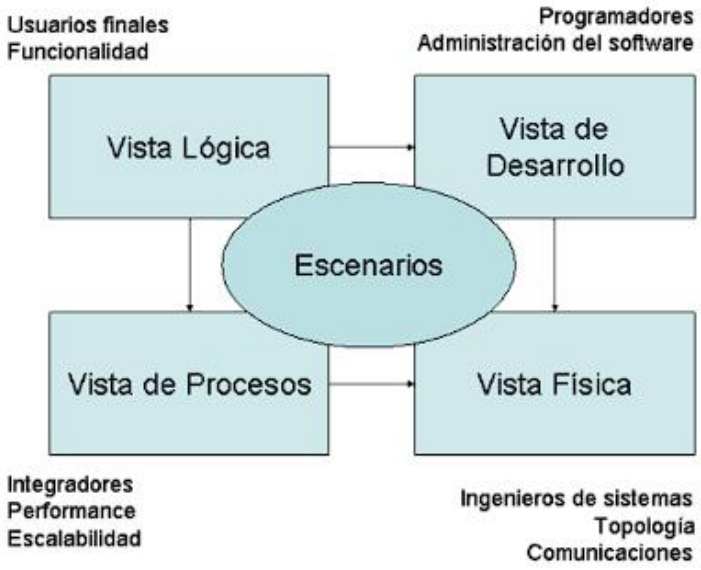


2.7 Diseño de Solución

Para la propuesta de solución se decidió basarse en el modelo de vistas “4+1” planteado por Philippe Kruchten.

Figura 6

Modelo de "4+1"vistas [11]

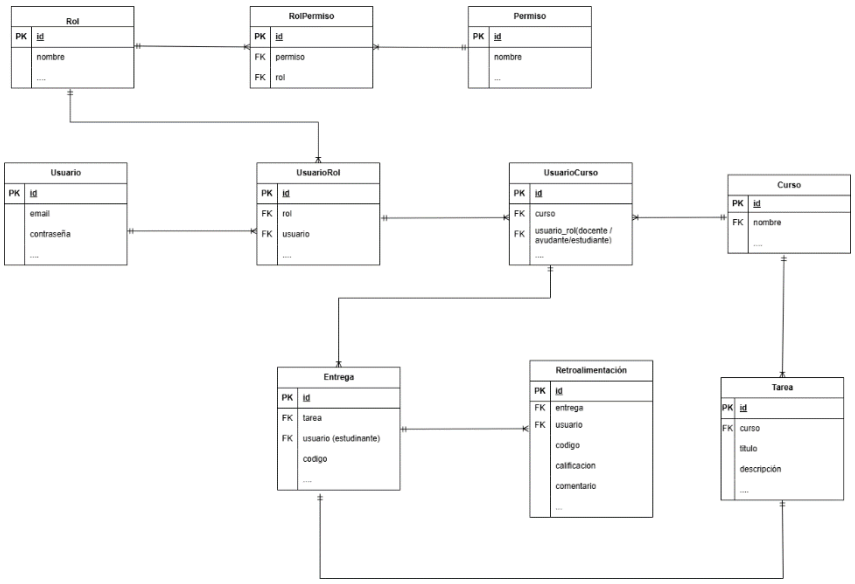


2.7.1 Vista Lógica

En la **Figura 7**, se presenta la vista lógica del sistema de retroalimentación automática, destacando la relación entre usuarios, roles, cursos, tareas y retroalimentaciones. Cada usuario está vinculado a uno o más roles mediante la tabla UsuarioRol, mientras que la asignación a cursos específicos se realiza a través de UsuarioCurso. Las tareas (Tarea) están asociadas a los cursos, y las entregas (Entrega) representan los códigos enviados por los estudiantes. Estas entregas son evaluadas, generando una Retroalimentación que incluye comentarios y calificaciones. Además, la relación entre Rol y Permiso asegura una gestión clara de los accesos y funciones en el sistema, proporcionando una estructura eficiente y escalable.

Figura 7

Diagrama del Modelo Lógico [Autoría propia]

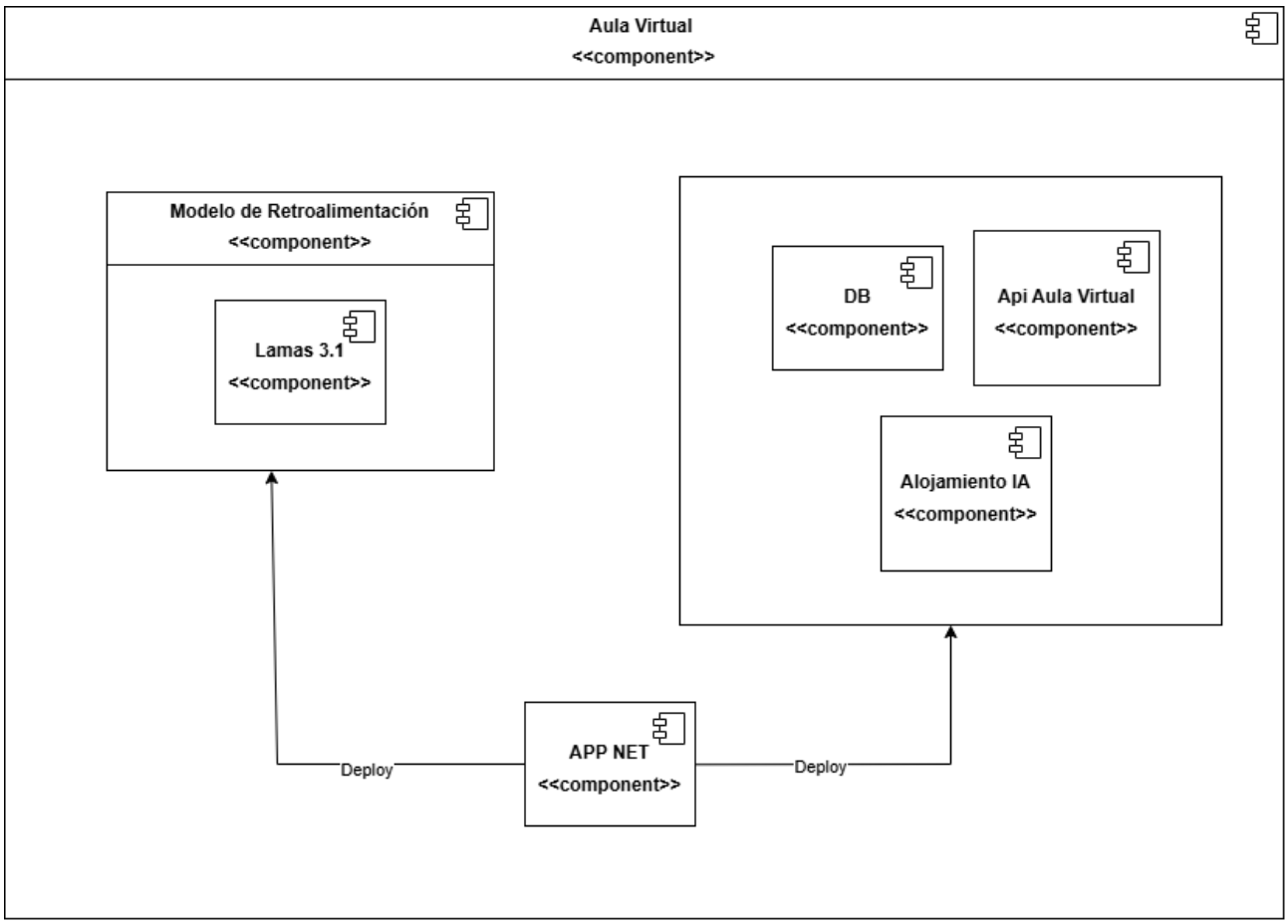


2.7.2 Vista de Implementación

En la **Figura 8**, se presenta la vista de implementación del sistema de retroalimentación automática. El modelo de retroalimentación, alojado en la capa central del sistema, interactúa con los componentes de la base de datos (DB) y la API del Aula Virtual, los cuales gestionan los datos y las integraciones con la plataforma educativa. El alojamiento de la inteligencia artificial (IA) maneja los procesos de análisis y generación de retroalimentaciones. Por último, el componente APP NET facilita el despliegue y acceso desde diferentes dispositivos, asegurando la interacción eficiente entre los estudiantes y el sistema a través de la arquitectura definida.

Figura 8

Diagrama del Modelo de Implementación [Autoría propia]



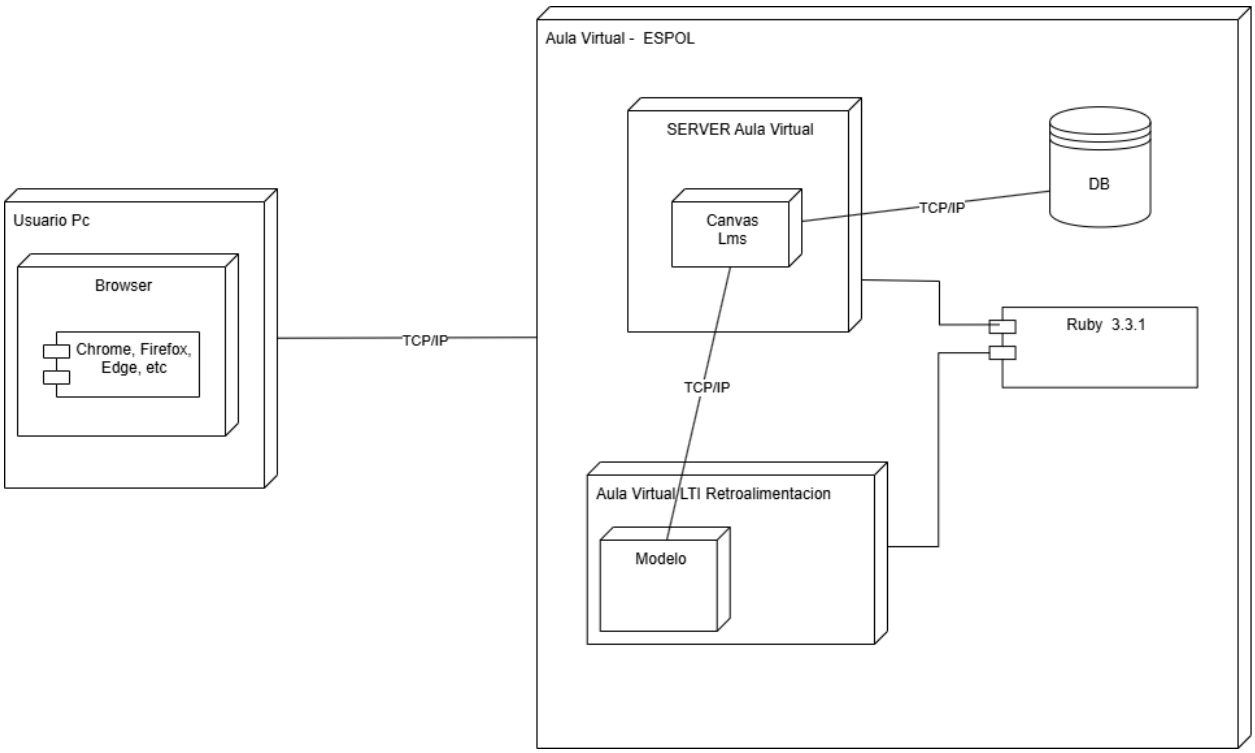
2.7.3 Vista de Despliegue

En la **Figura 9**, se detalla la vista de despliegue del sistema de retroalimentación automática. Los usuarios acceden al sistema a través de navegadores web como Chrome, Firefox o Edge, utilizando el protocolo TCP/IP para comunicarse con el servidor del Aula Virtual alojado en ESPOL. Este servidor integra el sistema Canvas LMS, que interactúa directamente con la base de datos (DB) y el componente de retroalimentación LTI, donde se ejecuta el modelo de IA. Además, el componente Ruby 3.3.1 se emplea para procesos de soporte y ejecución del modelo, asegurando la generación de retroalimentaciones precisas.

Este despliegue asegura la conectividad y escalabilidad, garantizando una experiencia fluida para estudiantes y docentes.

Figura 9

Diagrama del Modelo de Despliegue [Autoría propia]



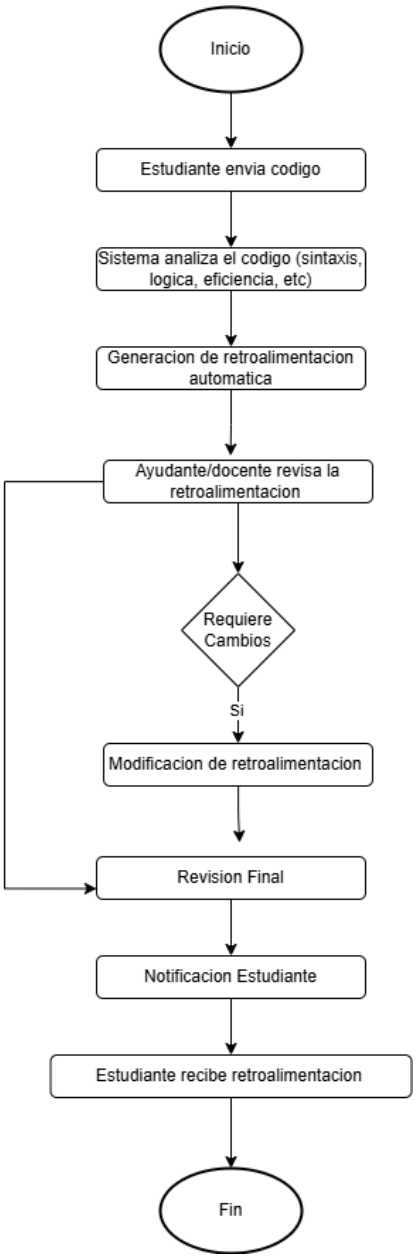
2.7.4 Vista de Proceso

En la **Figura 10**, se presenta la vista de proceso del sistema, detallando el flujo de trabajo desde la entrega de código por parte del estudiante hasta la recepción de la retroalimentación final. El proceso inicia con el estudiante enviando su código al sistema, que lo analiza automáticamente en términos de sintaxis, lógica y eficiencia. Posteriormente, se genera una retroalimentación inicial mediante IA, la cual es revisada por un ayudante o docente. Si se detecta la necesidad de ajustes, estos son realizados en la etapa de modificación, tras lo cual se lleva a cabo una revisión final. Finalmente, el estudiante es notificado y recibe la

retroalimentación ajustada, cerrando así el ciclo. Este flujo asegura precisión, personalización y rapidez en la entrega de comentarios.

Figura 10

Diagrama del Modelo de Sistema [Autoría propia]



La base de datos del sistema de retroalimentación automática se basa en el diagrama lógico presentado anteriormente (**Figura 7**), organizando las entidades principales y sus relaciones para garantizar la integridad y eficiencia del sistema. Incluye tablas como

Usuario, que gestiona la información de estudiantes, docentes y ayudantes, Curso, que almacena los detalles de las materias, y Tarea, que define las actividades asignadas en cada curso. Las entregas de código se registran en la tabla Entrega, vinculada tanto a los estudiantes como a las tareas, mientras que la retroalimentación generada por el sistema y revisada por los docentes se almacena en Retroalimentación, asociada directamente a las entregas. Además, se incluyen tablas para la gestión de roles y permisos, permitiendo un control granular de accesos y funciones dentro del sistema. Este diseño sigue un modelo relacional, asegurando la trazabilidad entre cursos, tareas, entregas y retroalimentaciones, y facilitando la escalabilidad para futuros requerimientos.

Capítulo 3

3.1 Resultados y análisis

3.1.1 Descripción

El proyecto de retroalimentación automática fue diseñado para asistir a los docentes en la evaluación de tareas de programación en materias de la carrera de Ciencias de Datos e Inteligencia Artificial de ESPOL. Este módulo, integrado con el Aula Virtual mediante LTI, utiliza inteligencia artificial para analizar códigos en Python, generando retroalimentación personalizada sobre aspectos de sintaxis, lógica y eficiencia. Los docentes podrán revisar y modificar los comentarios antes de enviarlos a los estudiantes.

Este plan detalla las acciones necesarias para evaluar la efectividad del módulo antes de su implementación oficial.

3.1.2 Tabla de Costos

La tabla 3-1 presenta un desglose detallado de las actividades necesarias para el desarrollo del proyecto, incluyendo la estimación de horas requeridas, el costo por hora y el costo total de cada actividad.

3.1.2.1 Descripción de las Actividades:

Levantamiento de Requerimientos (Canvas) (Frontend):

Esta actividad implica recopilar y documentar las necesidades funcionales del módulo en el entorno frontend.

Horas estimadas: 20 horas.

Costo total: $500.

Levantamiento de Requerimientos para el Backend (Ollama):

Consiste en identificar y definir los aspectos técnicos y funcionales necesarios para desarrollar la API y el backend del sistema.

Horas estimadas: 16 horas.

Costo total: $320

Desarrollo del Frontend:

Implementación de la interfaz de usuario, asegurando que sea funcional y amigable para los usuarios finales.

Horas estimadas: 18 horas.

Costo total: $360.

Desarrollo de la API con Llama:

Creación de la lógica del backend mediante la integración con la API de Llama, que será responsable de procesar y gestionar las solicitudes del sistema.

Horas estimadas: 24 horas.

Costo total: $360.

Pruebas con Ejemplos de Código:

Verificación del correcto funcionamiento del módulo mediante casos prácticos y ejemplos de código, evaluando su rendimiento y precisión.

Horas estimadas: 30 horas.

Costo total: $300.

Pruebas con Validaciones de Profesores:

Esta actividad involucra la revisión del sistema por parte de profesores, quienes evaluarán la efectividad y claridad de las retroalimentaciones generadas.

Horas estimadas: 20 horas.

Costo total: $200.

Corrección de Código:

Ajustes finales y solución de errores detectados durante las pruebas, asegurando que el sistema cumpla con los requisitos establecidos.

Horas estimadas: 15 horas.

Costo total: $225.

Tabla 1

Tabla de Costos

Actividades	Horas	Costo por Hora	Costo Total
Levantamiento de requerimientos (Canvas) (Frontend)	20	$25	$500
Levantamiento de requerimientos para el backend (Ollama)	16	$20	$320
Desarrollo del frontend	18	$20	$360
Desarrollo de la API con Llama	24	$15	$360
Pruebas con ejemplos de código	30	$10	$300
Pruebas con validaciones de profesores	20	$10	$200
Corrección de código	15	$15	$225

El total estimado asegura la cobertura de las actividades críticas del proyecto, manteniendo el enfoque en la calidad y usabilidad del sistema. Este plan de costos refleja un enfoque práctico y realista para el desarrollo del módulo.

3.1.3 Implementación

El desarrollo del módulo permite ingresar a tareas donde se analiza el desenvolvimiento de los estudiantes o grupos de estudiantes en cada una de ellas. Donde se aprecia la generación de retroalimentación de manera automática y donde el profesor/ayudantes es capaz de revisar esta retroalimentación y poder editar la misma para finalmente enviar cada una de las tareas con su respectiva revisión y calificación.

En la figura 11 se puede apreciar el historial de retroalimentaciones por revisar de parte de los profesores, a los cuales puede acceder para proceder con la revisión de la retroalimentación de cada estudiante como se aprecia en la figura 12. En la figura 14 se puede apreciar la forma en que se muestra la retroalimentación detallada de cada tarea.

Figura 11

Vista de Historial de Retroalimentaciones [Autoría propia]

Figura 12

Vista de tareas por estudiante [Autoría propia]

Figura 13

Vista de retroalimentación para profesores [Autoría propia]

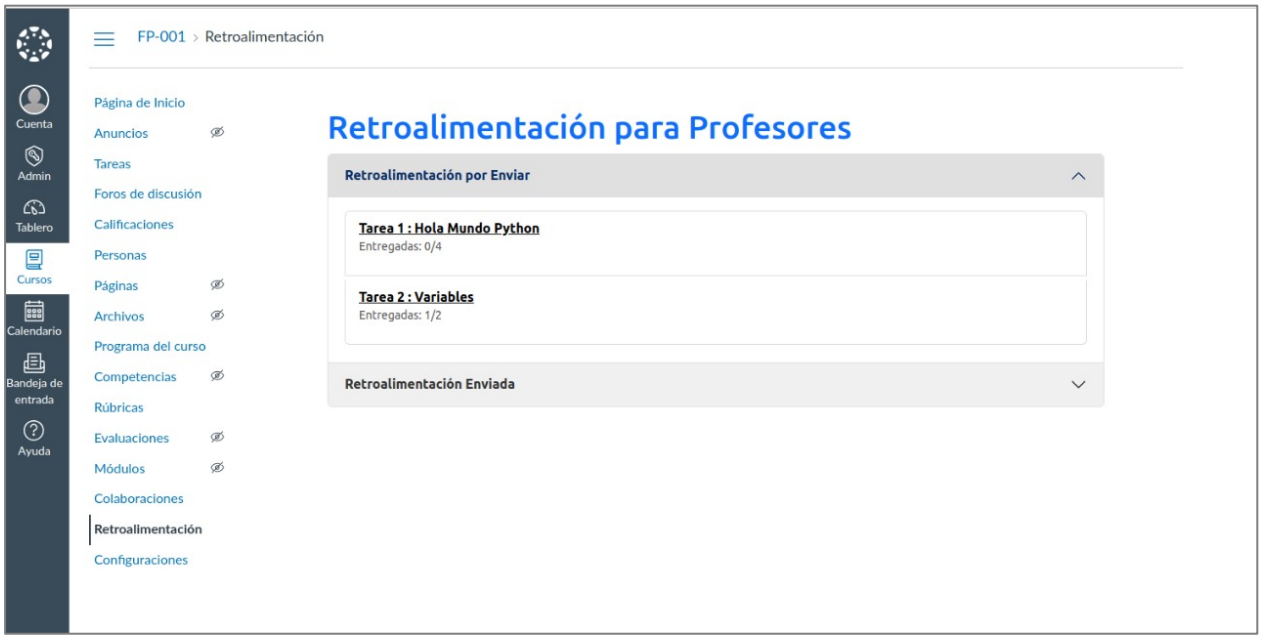


Figura 14

Vista de Pop-up de la retroalimentación detallada [Autoría propia]

3.2 Participantes

El estudio será llevado a cabo en el campus Gustavo Galindo de ESPOL. Se espera la participación de 20 personas:

Estudiantes: 10 estudiantes de las carreras de Ciencias de Datos e Ingeniería en Computación.

Profesores: 10 docentes que imparten o han impartido la materia de Fundamentos de Programación.

3.2.1 Perfil de los Participantes

Estudiantes: Edad: 20-24 años.

Nivel: Materias intermedias y avanzadas de Ing. Computación.

Profesores

Especialización: Experiencia directa en la enseñanza de Fundamentos de Programación.

3.3 Aspecto Evaluado #1: Nivel de Satisfacción

3.3.1 Metodología

Se aplicará una encuesta de satisfacción tras el uso del módulo, con las siguientes preguntas en escala de opción múltiple:

1. ¿Qué tan intuitivo es el módulo de retroalimentación automática?
2. ¿La retroalimentación proporcionada por el sistema fue clara y comprensible?
3. ¿La retroalimentación fue demasiado técnica o se ajustó a tu nivel de conocimiento?
4. ¿El tiempo de respuesta del sistema al evaluar tu código fue aceptable?
5. ¿Crees que las sugerencias te ayudaron a aprender algo nuevo sobre Python?

3.3.2 Meta

- Al menos el 80% de los participantes deben calificar la experiencia como satisfactoria (4 o 5 en las preguntas de satisfacción general e intuitividad).

3.4 Aspecto Evaluado #2: Precisión de la Retroalimentación

3.4.1 Metodología

Se evaluará la precisión de los comentarios generados por el módulo comparándolos con la retroalimentación proporcionada manualmente por los docentes. Los comentarios serán clasificados como "útiles" o "no útiles" por los docentes.

3.4.2 Meta

- Al menos el 90% de los comentarios generados deben ser considerados útiles.

3.5 Protocolo de Evaluación

3.5.1 Uso de Modulo

- Actividad: Revisar y modificar retroalimentaciones generadas por el módulo.
- Duración: 30 minutos.

3.5.2 Encuesta de Satisfacción

- Objetivo: Obtener retroalimentación sobre la experiencia del usuario.
- Duración: 10 minutos.

3.6 Conclusión Esperada

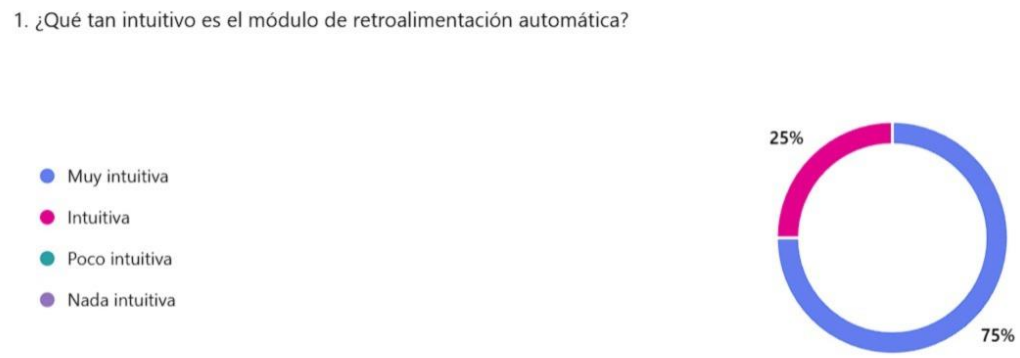
El plan de evaluación busca garantizar que el módulo cumple con los objetivos propuestos en términos de usabilidad, precisión y mejora del aprendizaje. Tras la recolección de datos, se analizarán los resultados y se realizarán ajustes en función de la retroalimentación obtenida.

3.7 Resultados

Luego de realizar las respectivas pruebas con los usuarios y que estos nos brinden retroalimentación a partir de la encuesta planteada en la metodología de evaluación, se obtuvieron los siguientes resultados:

Figura 15

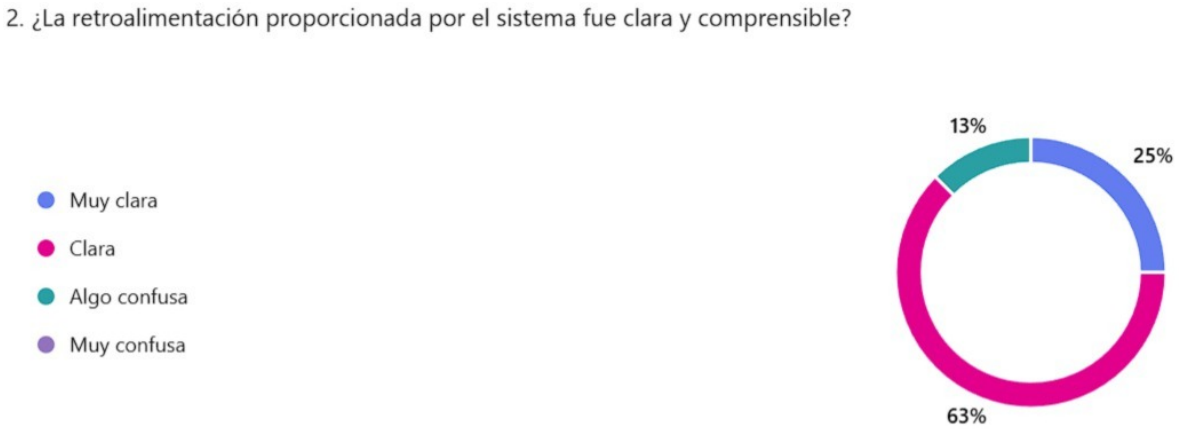
Resultados pregunta #1



Para el 75% de los usuarios encuestados, el módulo de retroalimentación fue muy intuitivo. Para el resto, fue simplemente intuitivo. Este resultado nos brinda una clara aceptación de la intuitividad que tiene el módulo.

Figura 16

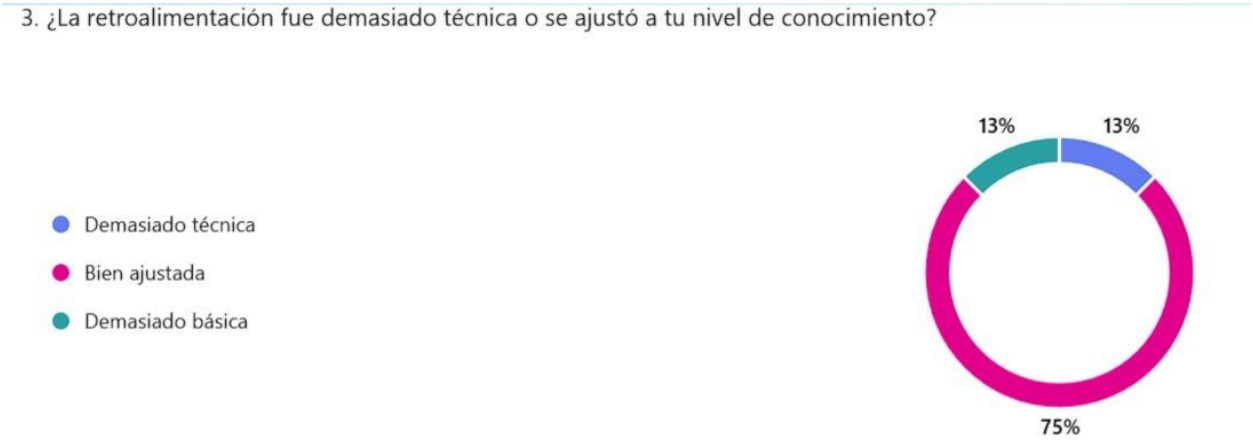
Resultados pregunta #2



Para el 87% de los usuarios encuestados, la retroalimentación proporcionada fue al menos clara. Sin embargo, existe un 13% que no la percibió del todo clara. Este resultado nos indica que la retroalimentación podría ser mejorada, no obstante, tiene buena aceptación entre los usuarios.

Figura 17

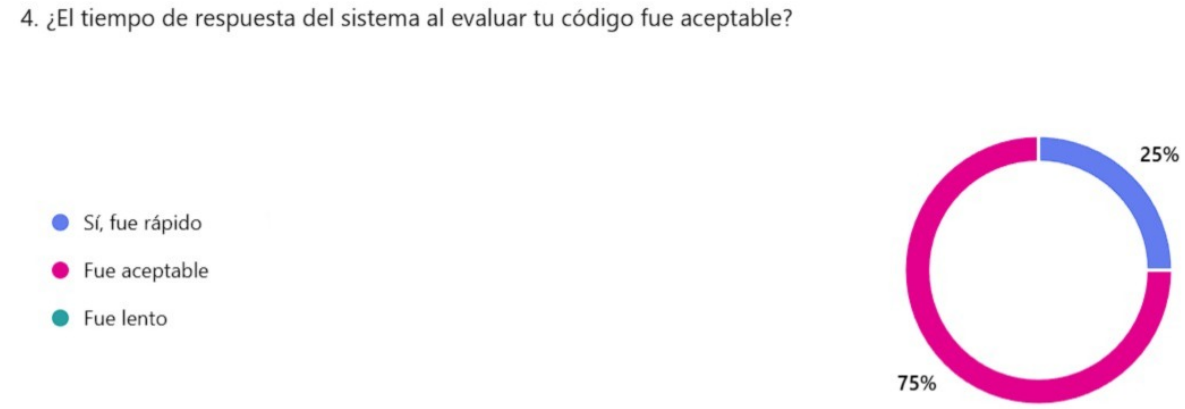
Resultados pregunta #3



Para alrededor de un cuarto de los encuestados, la retroalimentación no se ajustó a sus preferencias de aprendizaje. Este resultado indica que el modelo utilizado para la generación de la retroalimentación podría ajustarse más en base al nivel de aprendizaje de los usuarios. Sin embargo, la mayoría lo percibe como bien ajustado.

Figura 18

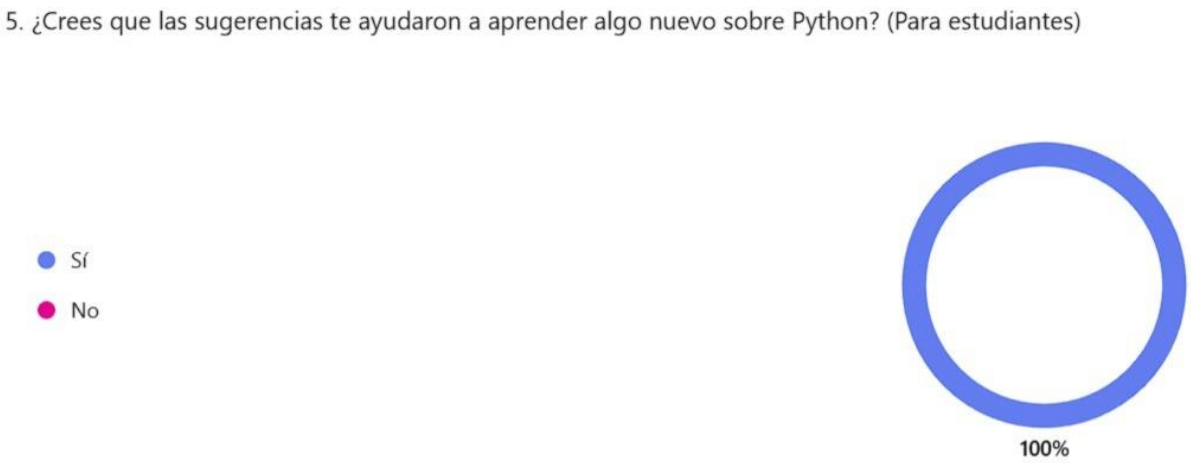
Resultados pregunta #4



A la mayoría de los usuarios encuestados les pareció un tiempo de respuesta aceptable al generarse la retroalimentación. Ya que se contó con recursos limitados para la generación de respuestas con el modelo de inteligencia artificial, es un resultado razonable. De todas maneras, la optimización del tiempo de respuesta siempre va a ser una prioridad.

Figura 19

Resultados pregunta #5



A todos los usuarios encuestados que contaron como perfil “estudiante”, aprendieron algo nuevo gracias a la retroalimentación. Esto demuestra la efectividad de la evaluación de las tareas ingresadas por estudiantes por parte del modelo de inteligencia artificial.

Resultados esperados

- Reducción de carga docente: Ya que la retroalimentación de tareas se produce automáticamente, los profesores pueden dedicar más tiempo a tareas complejas y personalizadas, delegando las revisiones rutinarias al módulo automatizado.
- Incremento en la satisfacción estudiantil: Retroalimentación más rápida y detallada mejora el aprendizaje.
- Eficiencia: Reducción significativa del tiempo requerido para evaluar trabajos.

Capítulo 4

4.1 Conclusiones y recomendaciones

4.1.1 Conclusiones

- Este sistema no solo mejora la calidad de la retroalimentación, sino que también fomenta un aprendizaje más profundo y significativo, permitiendo a los estudiantes abordar problemas complejos con mayor confianza y precisión.
- La integración de inteligencia artificial en procesos educativos rutinarios marca un avance significativo hacia la automatización y personalización del aprendizaje en plataformas virtuales, posicionando a la ESPOL como pionera en soluciones tecnológicas educativas.
- El módulo de retroalimentación automática ha demostrado ser una solución eficiente para superar los desafíos asociados a la evaluación manual de tareas de programación, optimizando los recursos docentes y mejorando la experiencia educativa.

4.1.2 Recomendaciones

- Soporte para otros lenguajes de programación: Se sugiere expandir el módulo para que sea compatible con más lenguajes de programación, lo que aumentará su aplicabilidad en diferentes materias.
- Análisis de datos del aprendizaje: Se propone integrar herramientas de análisis que permitan a los docentes identificar patrones en el desempeño de los estudiantes, como áreas de dificultad en común o tendencias de mejora.
- Integración institucional: Se propone que el módulo sea integrado oficialmente en las políticas académicas de ESPOL, posicionándolo como un recurso diario para la evaluación de tareas.
- Estudios longitudinales: Se propone que se realicen estudios a largo plazo para medir el impacto del módulo en el aprendizaje de los estudiantes, evaluando aspectos como la capacidad de retener conocimiento y la mejora en habilidades prácticas.
- Optimización del modelo de IA: Se propone actualizar periódicamente el modelo Llama 3.2 para incorporar avances en inteligencia artificial y garantizar que el módulo se mantenga en el avance tecnológico.

Referencias

[1] Garrison, D. R., & Kanuka, H. (2004). Blended learning: Uncovering its transformative potential in higher education. The internet and higher education, 7(2), 95-105.

[2] Hattie, J., & Timperley, H. (2007). The power of feedback. Review of educational research, 77(1), 81-112.

[3] Keuning, H., Jeuring, J., & Heeren, B. (2018). A systematic literature review of automated feedback generation for programming exercises. ACM Transactions on Computing Education (TOCE), 19(1), 1-43.

[4] Zawacki-Richter, O., Marín, V. I., Bond, M., & Gouverneur, F. (2019). Systematic review of research on artificial intelligence applications in higher education–where are the educators?. International Journal of Educational Technology in Higher Education, 16(1), 39.

[5] Wang, K., Singh, R., & Su, Z. (2018). Dynamic neural program embeddings for program repair. In International Conference on Learning Representations.(TOCE), 19(1), 1-43.

[6] VanLehn, K. (2011). The relative effectiveness of human tutoring, intelligent tutoring systems, and other tutoring systems. Educational Psychologist, 46(4), 197-221.

[7] Zhang, Z., Dong, Z., Shi, Y., Price, T., Matsuda, N., & Xu, D. (2024). Students' Perceptions and Preferences of Generative Artificial Intelligence Feedback for Programming. Proceedings of the AAAI Conference on Artificial Intelligence, 38(21), 23250-23258.

[8] Wood, D., Bruner, J. S., & Ross, G. (1976). The role of tutoring in problem solving. Journal of child psychology and psychiatry, 17(2), 89-100.

[9] Black, P., & Wiliam, D. (1998). Assessment and classroom learning. Assessment in Education: principles, policy & practice, 5(1), 7-74.

[10] Ihantola, P., Ahoniemi, T., Karavirta, V., & Seppälä, O. (2010). Review of recent systems for automatic assessment of programming assignments. In Proceedings of the 10th Koli calling international conference on computing education research (pp. 86-93).

[10] Zhang, Z., Dong, Z., Shi, Y., Price, T., Matsuda, N., & Xu, D. (2024). Students' Perceptions and Preferences of Generative Artificial Intelligence Feedback for Programming. Proceedings of the AAAI Conference on Artificial Intelligence, 38(21), 23250-23258.

[11] Anónimo, «MODELADO Y GESTIÓN DE LA INFORMACION,» Blogger, 28 de agosto del 2015.

Apéndice A

Preguntas de Formulario de Retroalimentación

1. ¿Qué tan intuitivo es el módulo de retroalimentación automática?
   Opciones de respuesta:
   - Muy intuitiva
   - Intuitiva
   - Poco intuitiva

2. ¿La retroalimentación proporcionada por el sistema fue clara y comprensible?
   Opciones de respuesta:
   - Muy clara
   - Clara
   - Algo confusa
   - Muy confusa

3. ¿La retroalimentación fue demasiado técnica o se ajustó a tu nivel de conocimiento?
   Opciones de respuesta:
   - Demasiado técnica
   - Bien ajustada
   - Demasiado básica

4. ¿El tiempo de respuesta del sistema al evaluar tu código fue aceptable?
   Opciones de respuesta:
   - Sí, fue rápido
   - Fue aceptable
   - Fue lento

5. ¿Crees que las sugerencias te ayudaron a aprender algo nuevo sobre Python? (Para estudiantes)
   Opciones de respuesta:
   - Sí
   - No

Apéndice B

Guía de Uso

Revisar Retroalimentaciones que hacen falta por revisar dentro del modulo llamado “Retroalimentación” al lado izquierdo del curso.

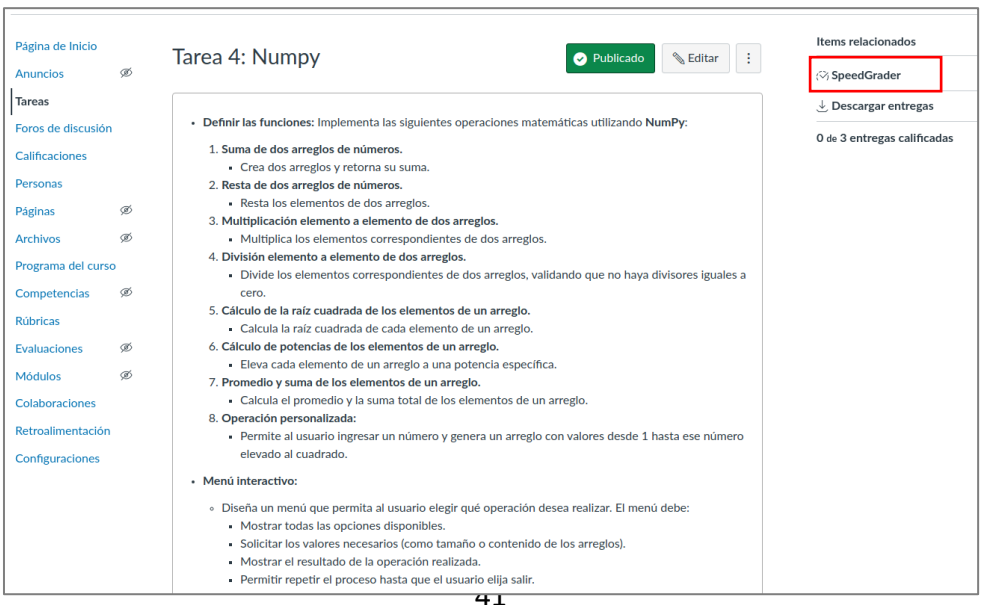
Figura B.1

Vista del módulo de retroalimentación

Seleccionar la Tarea que se desea revisar y se mostrará la vista de la descripción de la tarea, luego en el lado derecho superior seleccionar el “SpeedGrader”.

Figura B.2

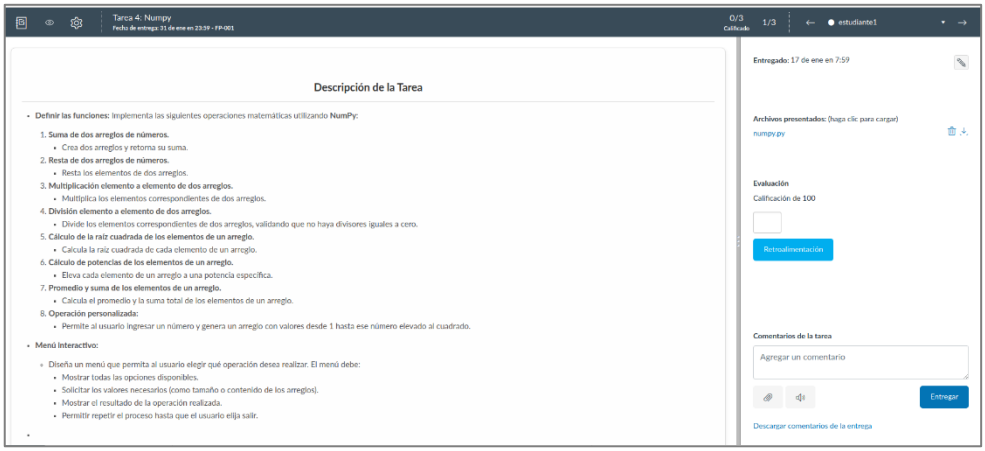
Vista de la descripción de la tarea



Ya en SpeedGrader, se muestra en la parte inferior la descripción de la tarea y el lado derecho el botón de retroalimentación.

Figura B.3

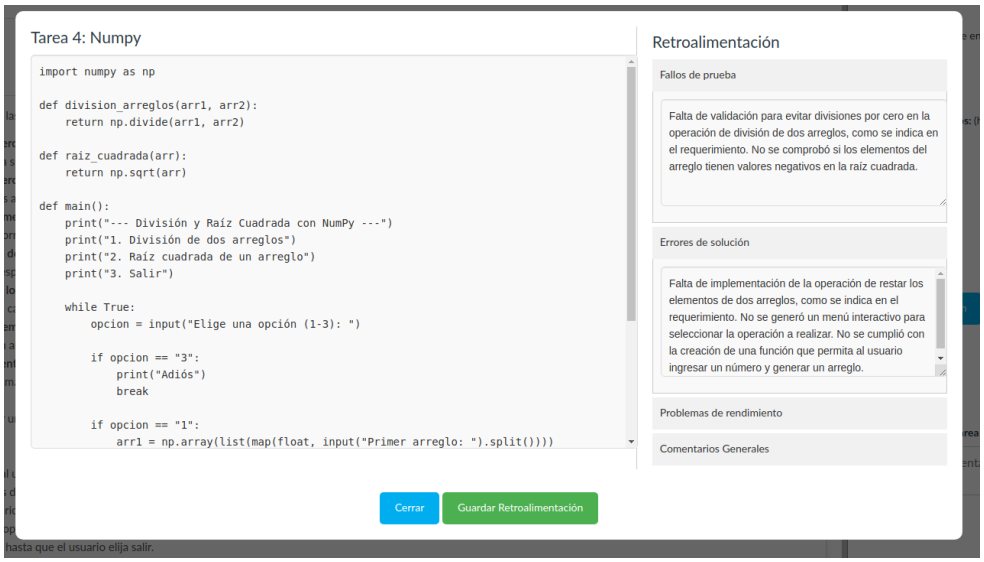
Vista del SpeedGrader con botón de retroalimentación y descripción de la tarea



Al presionar el botón de retroalimentación, se mostrará un pop-up, en el lado izquierdo se muestra el código que se está analizando en base a la orden de la tarea, y en el lado derecho se muestra la retroalimentación generada por la IA dentro de cuadro de texto que se pueden modificar para su posterior envío al estudiante.

Figura B.4

Vista del Pop-up del código analizado y la retroalimentación generada



Al presionar el botón de “Guardar Retroalimentación” la retroalimentación se guardará en el cuadro de texto de los comentarios del SpeedGrader, para luego poder enviarlo o modificarlo si se desea.

Figura B.5

Vista del comentario ya guardado

Apéndice C

Historias de Usuario

Tabla C.1

Historia de Usuario HU-001-2024

Identificador (ID) de la Historia	Rol	Característica / Funcionalidad	Razón / Resultado	Número (#) de Escenario	Criterio de Aceptación (Título)	Contexto	Evento	Resultado / Comportamiento esperado
HU-001-2024	Como Estudiante	Recibir retroalimentación detallada sobre mi código de programación	Con la finalidad de poder identificar y corregir errores, mejorar la lógica y adoptar mejores prácticas de programación.	1	Análisis de errores sintácticos, lógicos y de eficiencia	Cuando el estudiante entrega su código de programación	Cuando el sistema analiza el código	El sistema debe generar comentarios sobre errores sintácticos, lógicos y de eficiencia en el código del estudiante.
				2	Explicaciones claras y sugerencias de mejora	Cuando el sistema identifica problemas en el código	Después del análisis del código	comentarios generados deben incluir explicaciones claras sobre los problemas identificados y sugerencias específicas para mejorar el código.
				3	Entrega oportuna de la retroalimentación	Después del análisis del código	Cuando se despliegue el listado de categorías a seleccionar.	La retroalimentación debe ser entregada al estudiante de manera oportuna, sin retrasos significativos.

Tabla C.2

Historia de Usuario HU-002-2024

	Enunciado de la Historia				Criterios de Aceptación			
Identificador (ID) de la Historia	Rol	Característica / Funcionalidad	Razón / Resultado	Número (#) de Escenario	Criterio de Aceptación (Título)	Contexto	Evento	Resultado / Comportamiento esperado
HU-002-2024	Como Profesor	Revisar y personalizar la retroalimentación automática	Con la finalidad de poder editar, agregar o modificar los comentarios generados por el sistema antes de enviarlos a los estudiantes.	1	Acceso a la retroalimentación generada automáticamente	Cuando el profesor quiere revisar los comentarios generados por el sistema	Después del análisis automático del código de los estudiantes	sistema debe permitir a los profesores acceder fácilmente a la retroalimentación generada automáticamente para cada estudiante.
				2	Edición, adición y modificación de comentarios	Cuando el profesor quiere personalizar la retroalimentación	Después de revisar los comentarios generados automáticamente	Los profesores deben poder editar, agregar o modificar los comentarios generados por el sistema antes de enviarlos a los estudiantes.
				3	Proceso de revisión y personalización intuitivo y eficiente	Cuando el profesor revisa y personaliza la retroalimentación	Durante el proceso de revisión y personalización	El proceso de revisión y personalización de la retroalimentación debe ser intuitivo y eficiente para los profesores.

Tabla C.3

Historia de Usuario HU-003-2024

	Enunciado de la Historia				Criterios de Aceptación			
Identificador (ID) de la Historia	Rol	Característica / Funcionalidad	Razón / Resultado	Número (#) de Escenario	Criterio de Aceptación (Título)	Contexto	Evento	Resultado / Comportamiento esperado
HU-003-2024	Como Profesor	Agilizar el proceso de revisión de códigos de programación	Con la finalidad de poder dedicar más tiempo a brindar apoyo y retroalimentación personalizada a los estudiantes.	1	Análisis y generación de comentarios eficiente	Cuando el profesor necesita revisar los códigos de programación de los estudiantes	Después del análisis automático del código	El sistema de retroalimentación automática debe analizar y generar comentarios sobre los códigos de programación de los estudiantes de manera eficiente.
				2	Acceso fácil a la retroalimentación generada	Cuando el profesor quiere revisar los comentarios generados por el sistema	Después del análisis automático del código	Los ayudantes profesor poder acceder fácilmente a la retroalimentación generada por el sistema para cada estudiante.
				3	Más tiempo para apoyo y retroalimentación personalizada	Cuando el profesor ha revisado los códigos de los estudiantes	Después de la revisión automatizada	El tiempo ahorrado en la revisión automática debe permitir a los profesores dedicar más esfuerzo a brindar apoyo y retroalimentación personalizada a los estudiantes.

Tabla C.4

Historia de Usuario HU-004-2024

	Enunciado de la Historia				Criterios de Aceptación			
Identificador (ID) de la Historia	Rol	Característica / Funcionalidad	Razón / Resultado	Número (#) de Escenario	Criterio de Aceptación (Título)	Contexto	Evento	Resultado / Comportamiento esperado
HU-004-2024	Como Estudiante	Acceder a un historial de retroalimentación de mis códigos anteriores	Para revisar los errores y mejoras pasadas y así aprender de ellos y evitar cometer los mismos errores en el futuro.	1	Consulta de retroalimentación pasada	Cuando el estudiante desea revisar sus entregas previas	Al acceder al historial de retroalimentación	El sistema debe permitir al estudiante acceder a la retroalimentación de sus códigos anteriores, mostrando los comentarios y sugerencias recibidas en cada entrega.
				2	Identificación de patrones de error	Cuando el estudiante revisa el historial de retroalimentación	Al analizar los comentarios de entregas anteriores	El sistema debe permitir al estudiante identificar patrones en sus errores frecuentes y sugerencias de mejora, facilitando el aprendizaje continuo

Tabla C.5

Historia de Usuario HU-005-2024

	Enunciado de la Historia				Criterios de Aceptación			
Identificador (ID) de la Historia	Rol	Característica / Funcionalidad	Razón / Resultado	Número (#) de Escenario	Criterio de Aceptación (Título)	Contexto	Evento	Resultado / Comportamiento esperado
HU-005-2024	Como Profesor	Acceder a un tablero de control con el estado de revisión de las entregas de los estudiantes	Para gestionar las tareas pendientes de revisión y asegurar que se cumplan los tiempos de entrega de retroalimentación	1	Visualización del estado de cada entrega	Cuando el profesor revisa el estado de las entregas pendientes	Al acceder al tablero de control del sistema	El sistema debe mostrar el estado de cada entrega, indicando si ha sido revisada, en proceso o pendiente, y la fecha límite para cada retroalimentación.
				2	Organización de entregas pendientes	Cuando el profesor necesita priorizar las entregas que requieren atención inmediata	Al ordenar las entregas según su fecha de vencimiento	El sistema debe permitir al profesor ordenar las entregas por prioridad o fecha de entrega, facilitando una gestión eficiente del tiempo