MVP De Startup-EduTech-Ia Informe Final

En este informe detallo el desarrollo del MVP de la plataforma startup EduTech IA. El objetivo de este MVP es crear una plataforma que personalice los cursos y contenidos a la necesidad de cada usuario.

Resumen Ejecutivo

A lo largo del proyecto se han tomado decisiones clave en cuanto a tecnologías, metodologías de desarrollo y estrategias de integración continua para garantizar la estabilidad y seguridad del software.

En primer lugar, se investigó sobre la selección del leguaje y paradigma de programación, para determinar que lo más beneficioso es utilizar Java con programación orientado a objetos como base del proyecto.

(Explicación del Diagrama)

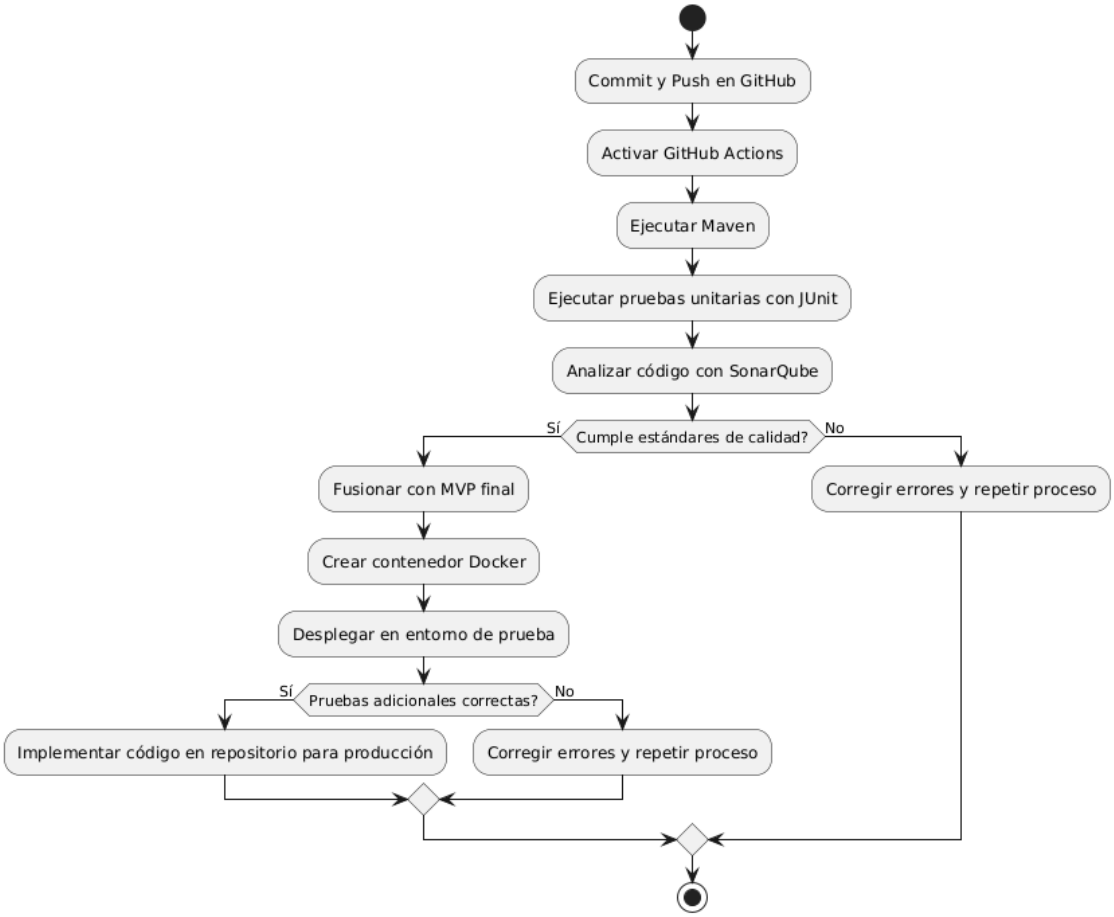
Después se elaboró un diagrama de clases para representar la estructura base que tendrá el MVP en el que decidí crear 7 clases. Habrá una clase “Usuario” que será abstracta que formara las bases para la clase “Alumno” y “Profesor”, el funciona miento del programa tienen la siguiente estructura, hay una clase “Gestor de cursos” en la que se almacenaran todos los cursos, esta gestionara la creación modificación y eliminación de estos, los profesores pueden acceder a esta para gestionarlos. La clase “Alumno” puede acceder a la clase “Curso” para inscribirse en cualquier curso, pero también puede utilizar la clase “Filtro” para ver los cursos recomendados para él. La clase “Filtro” utiliza un criterio para seleccionar cursos específicos de la lista de todos los cursos que se encuentran en el gestor de cursos. Por último “Curso” puede acceder a “Contenido” para agregarlo o eliminarlo de su lista.

<Picha Aquí para ver diagrama>

A continuación, se seleccionaron las herramientas y configuración del entorno de desarrollo. El IDE sobre el que se trabajara será el Intelli J por su soporte avanzado a Java y integración con Git que se usara para un buen control de versiones. Los archivos se subirán a un repositorio en la nube utilizando GitHub que nos permite un buen control y almacenamiento sobre el proyecto.

Siguiendo con la planificación lo siguiente fue desarrollar el pipeline de integración y entrega continua. En este apartado la misión era seleccionar las herramientas necesarias para garantizar la calidad y desarrollo del software, integrando unas acciones que automatizen las pruebas y comprobaciones del código a la hora de subirse. Para esto se seleccionaron las GitHub actions, que permiten automatizar el flujo de trabajo dentro de los repositorios de GitHub.

La idea es que a la hora de realizar un commit and push en el repositorio local, se activen las acciones automáticamente. El flujo es el siguiente:



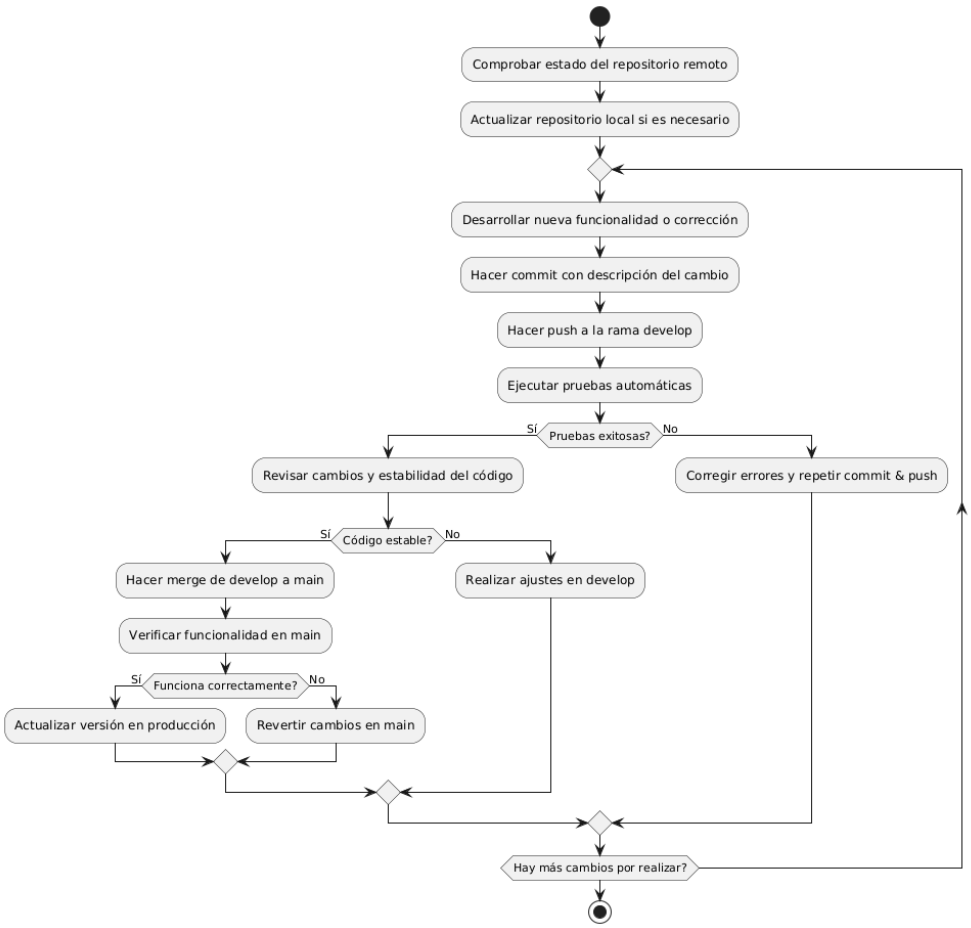
Las herramientas utilizadas aquí son:

- Maven, para la compilación del código por sus buenos resultados
- JUnit, para verificación del código y comprobación de que este correcto.
- SonarQube, para asegurar que el código cumple con los estándares de calidad definidos.
- Docker, para una ejecución en un entorno controlado y seguro.

También se ha elaborado un protocolo a seguir de las practicas y medidas a seguir para garantizar la seguridad del proyecto durante su desarrollo. Este se centra en el seguimiento de varios pasos a la hora de realizar cualquier cambio en el proyecto. Principalmente nos centraremos en la integración con GitHub que es donde se encuentra el proyecto completo. El repositorio lo tenemos dividido en dos ramas principales, main y develop.

El funcionamiento es el siguiente, cualquier cambio se trabaja en el repositorio local desde el cual se le hará un commit (que comente el cambio realizado) y un push a la rama develop (o sus respectivas subramas), una vez llegue a develop aquí se ejecutaran las acciones unitarias etc. Si todo esta correcto y ya se ha comprobado que no hay ningún error es cuando ya se podrá hacer un pull reques para pasar los cambios a la rama main que es la rama de producción donde se encuentra la última versión del proyecto funcional.

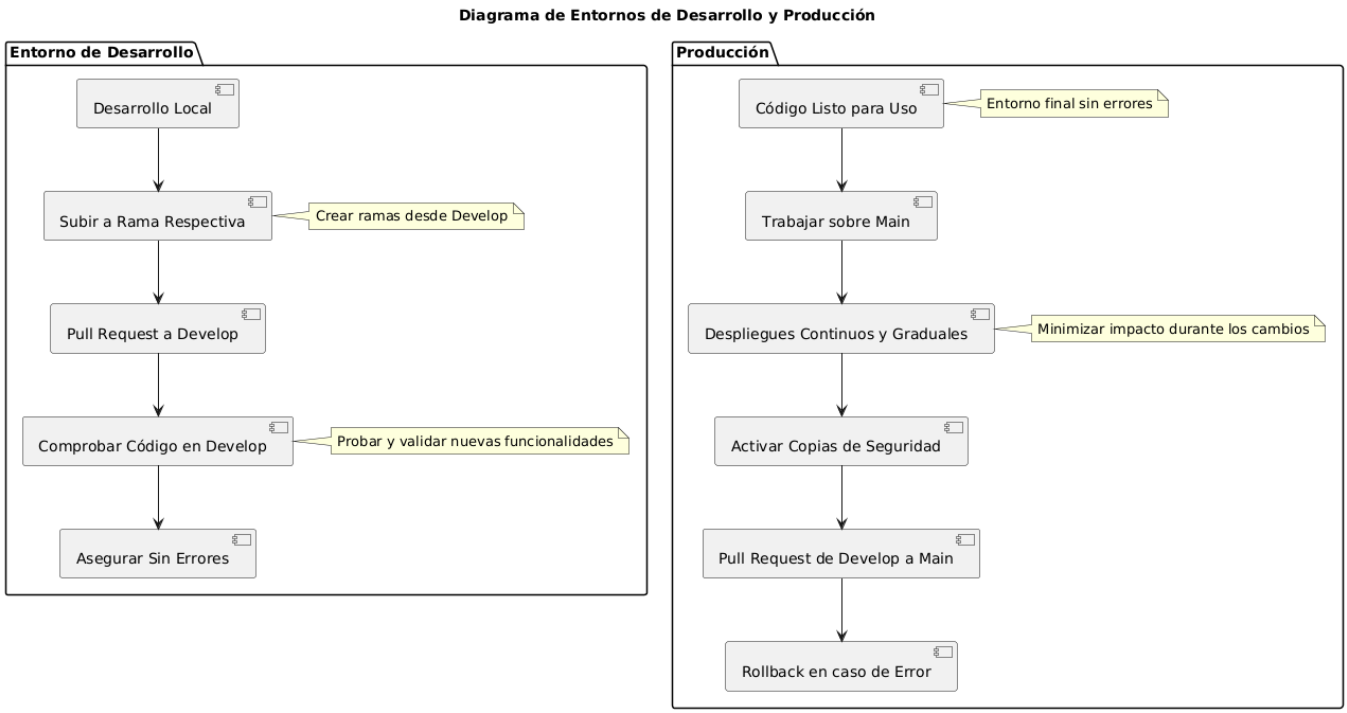
Además de el anterior flujo es importante seguir un desarrollo seguro y de calidad en cuanto al código siguiendo los estándares y normas de seguridad.



Por último, en la planificación de este proyecto se definió una estrategia de gestión y optimización del entorno. Dos entornos principales, desarrollo y producción.

- Entorno de Desarrollo:
    - Es en el que se suben los cambios.
    - Se utiliza para probar y validar nuevas funcionalidades.
    - Se crean ramas desde develop según sea necesario.
    - El código se desarrolla localmente y se sube s su respectiva rama.
    - Se hace “pull request” a develop si es necesario.
    - Comprobar el código en develop para asegurar que no haya errores antes de pasarlo a producción.

- Producción:
    - Entorno final donde debe haber código listo para uso sin errores.
    - Se trabaja sobre el main.
    - Se realizan despliegues continuos y graduales para minimizar impacto.
    - Se debe activar copias de seguridad antes de cualquier cambio.
    - Se hace un “pull request” para fusionar los cambios de develop a main, en caso de detectar algún error se realiza rollback para solucionar desarrollo



Justificación de Elecciones

-Sobre lenguaje y paradigma de programación

Como ya se a mencionado se a optado por la utilización de Java con programación orientada a objetos, debido a sus múltiples ventajas en términos de rendimiento, seguridad y compatibilidad. Entre las razones que justifican esta elección se encuentran las siguientes:

JAVA

- Independencia de plataforma, ya que java utiliza la Java Virtual Machine, que permite ejecutar el mismo código en diferentes sistemas operativos sin la necesidad de realizar modificaciones, asegurando así la portabilidad del software.
- Seguridad integrada, gracias a su tipado estático y sus mecanismos de control de acceso (private, protected, public), Java previene errores en tipo de ejecución y protege los datos sensibles. Además, su gestor de memoria automático ayuda a evitar problemas de fugas de memoria.
- Alto rendimiento, con la ayuda del compilador Just-In-Time, Java optimiza el código en tiempo de ejecución, mejorando así la eficiencia del del software y permite manejar múltiples usuarios simultáneamente.
- Alto soporte, Java es un estándar mundial con una amplia comunidad, esto facilita el acceso a documentación, soporte y herramientas avanzadas para optimizar el desarrollo.

POO

- Modularidad y reutilización, este permite dividir el software en clases y objetos reutilizables, facilitando la organización y mantenimiento del código.
- Encapsulamiento y seguridad, protege los datos al restringir el acceso directo a los atributos de las clases mediante modificadores de acceso.
- Mantenibilidad y escalabilidad, el uso de herencia y polimorfismo facilita la extensión del código sin modificar funcionalidades existentes.

Gracias a estos principios, el desarrollo del MVP se realizará de manera estructurada y eficiente asegurando que el sistema sea flexible y fácil de mantener.

-Sobre las herramientas y configuración del entorno de desarrollo

Para garantizar un correcto flujo de trabajo y eficiente gestión del código, se seleccionaron las siguientes herramientas por su potencia y eficiencia:

- Intelli J, como entorno de desarrollo por su compatibilidad con Java, su integración con herramientas de CI/CD y su facilidad para la depuración del código.
- Git y GitHub como sistema de control de versiones, ya que permiten un desarrollo colaborativo, control de cambios y automatización de tareas mediante GitHub Actinos.
- Maven para la automatización del proceso de compilación, facilitando la estructura del proyecto y la ejecución de pruebas.
- JUnit para la ejecución de pruebas unitarias, asegurando el correcto funcionamiento del código antes de su integración.
- SonarQube elegido por su calidad a la hora de analizar la calidad del código, identificar vulnerabilidades, errores y problemas de mantenimiento.

- Docker, por su eficiencia y seguridad a la hora de crear contenedores que permiten ejecutar el software en un entorno controlado, asegurando compatibilidad y portabilidad.

-Sobre la metodología a seguir

La metodología utilizada durante el desarrollo esta pasada para evitar la mayor cantidad de errores que se pueda y además tener un registro completo de la evolución del código gracias a las funcionalidades de Git y GitHub. Como ya se explicó brevemente el proceso es el siguiente, en el repositorio local una vez hecho cambios necesarios se realizará un commit que tiene que estar bien comentado para así poder ubicar fácilmente el cambio en la línea de tiempo, después se hace el push para que se subo a GitHub y donde se ejecutaran los test.

Luego se ha seleccionado la metodología de desarrollar todo en ramas diferentes a la main. Para pasarlas a develop en la que se realiza una comprobación final para asegurar que todo funciona correctamente antes de llevar los cambios al main.

Seleccione esta metodología porque de esta forma se evitan grandes problemas en el entorno de producción al resolverlos y probarlos antes de juntarlos con el producto que esta cara al público.

-Sobre los protocolos de seguridad seguidos

Además del control del entorno de desarrollo y producción con las diferentes ramas ya mencionadas, he seleccionado también protocolos de integridad y confidencialidad del proyecto. Las medidas adoptadas son las siguientes:

He establecido un control de acceso, asignando permisos específicos para evitar que usuarios no autorizados puedan realizar cambios en el código o acceder a información sensible.

Se realizarán auditorias de seguridad periódicas con el objetivo de evaluar la vulnerabilidad del código y configuración del entorno.

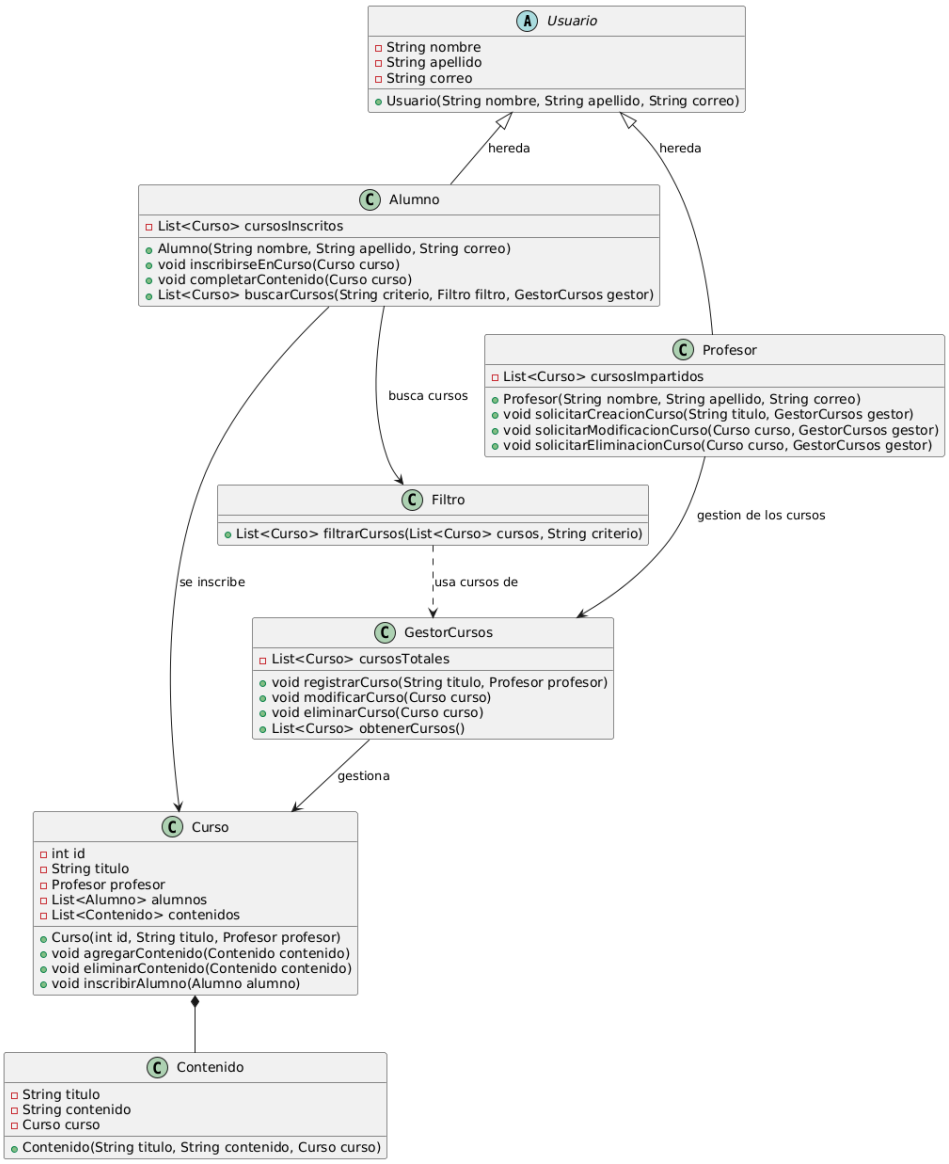
También se implementarán copias de seguridad regulares, esto para evitar la perdida de datos en caso de un fallo grabe.

Por último, se seguirá una política de actualización continua tanto de las herramientas como de las librerías utilizadas en el proyecto, esto con el fin de mantener a la ultima el software protegido de vulnerabilidades y errores ya conocidos.

Enlaces y Recursos de otras Tareas

- Repositorio de GitHub: https://github.com/MarcosAlonso05/Startup-EduTech-IA

- Diagrama de clases:



- Video: En el GitHub

Conclusión

El desarrollo del MVP de la plataforma EduTech IA se ha llevado a cabo con un enfoque riguroso en la planificación, implementación de tecnologías adecuadas y protocolos de seguridad. La elección de Java como lenguaje de programación y la adopción de metodologías ágiles, junto con un sólido sistema de gestión de versiones y una integración continua, aseguran que el producto final sea de alta calidad y cumpla con los estándares de seguridad y eficiencia.

El próximo paso en el proyecto es realizar pruebas de usuario con el MVP, para recopilar feedback que permita mejorar la experiencia y funcionalidad de la plataforma. Con este enfoque centrado en el usuario, se espera que EduTech IA logre personalizar y optimizar el aprendizaje para cada individuo, cumpliendo así su objetivo fundamental de proporcionar una educación accesible y adaptada a las necesidades de todos. A medida que avanzamos en el desarrollo, se evaluarán nuevas funcionalidades y características que se integrarán en futuras versiones de la plataforma, manteniendo siempre el compromiso con la calidad, la seguridad y la satisfacción del usuario.

(Nota Como Estudiante y mi Progreso)

En este proyecto he podido investigar bastante sobre muchas cosas que no conocía, sobre todo he ampliado mi habilidad con GitHub puesto que hasta ahora no lo había utilizado mucho y no sabia muchas de sus funciones básicas. Además, he entrado al mundo de las GitHub acctions que me era totalmente desconocido antes de iniciar el proyecto y sobre el que tengo que investigar mucho más, lo mismo me ha pasado con Docker.