SELECCIÓN DE LENGUAJE Y PARADIGMA DE PROGRAMACION

Nuestro principal cometido es la robustez y flexibilidad del MVP, que sea fácil trabajar en él, sea seguro y no se complique solucionar problemas imprevistos. Hemos llegado a la conclusión de que Java es el lenguaje ideal, debido a grandes ventajas en desarrollo backend, gran compatibilidad y su paradigma de programación orientado a objetos.

Java es un lenguaje increíblemente efectivo para EduTech IA, ya que, dada la naturaleza de este proyecto, se espera que la plataforma funcione en multitud de dispositivos, y se pueda acceder a ella ya sea desde una app o la web, por ello las capacidades de portabilidad que proporciona Java son inigualables.

Además, Java ofrece un alto rendimiento gracias a su gran gestión de memoria, haciendo uso del Garbage Colector, esto significa una fuerte optimización del sistema que ayudarán con los grandes volúmenes de información que serán introducidos a la plataforma dada su cometido educativo.

Una de nuestras mayores preocupaciones era la seguridad, dado a que se manejan una gran cantidad de datos sensibles en plataformas educativas era vital tener un lenguaje seguro, y con Java gracias a la verificación de código bytecode y APIs de seguridad integrados, estos minimizan las vulnerabilidades creando así un entorno seguro en el que la información quede a buen recaudo.

Otro de los aspectos que más en cuenta hemos tenido era la facilidad de trabajar en el programa, y dado que Java es uno de los principales lenguajes de programación a nivel mundial podemos asegurarnos que nuevos integrantes a la larga en el equipo vayan a tener una curva de aprendizaje bastante calmada con el proyecto, además de que la popularidad de Java también nos aporta un gran acceso a numerosas librerías y frameworks como Spring Boot, Hibernate y Apache Kafka, que nos ayudaran a conseguir un backend eficiente y escalable.

Hemos decidido hacer uso de la programación orientada a objetos como paradigma ya que esto facilitará en gran medida la organización del código, además de la reutilización de gran parte de este gracias a las clases, también ayudando a la ampliación del código gracias a la encapsulación y la herencia y mejorando la adaptabilidad de este al hacer uso del polimorfismo y la abstracción, permitiendo generar una experiencia más personalizada para el usuario.