DevMentor

El desafío consiste en crear **"DevMentor"** una plataforma inteligente e interactiva para la enseñanza de programación que incorpore interacción con modelos IA de lenguaje multimodal, capaz de ofrecer un entorno enriquecedor y dinámico para aprender a programar de manera efectiva y atractiva. Este tipo de sistema no solo proporcionará instrucción y recursos educativos, sino que también facilitará la interacción directa con un asistente de inteligencia artificial para resolver dudas, obtener ejemplos de código y practicar la escritura de programas y probarlos en tiempo real. A continuación, se indican orientaciones generales de cómo podría estructurarse este proyecto:

OBJETIVOS DEL PROYECTO

1. **Enseñanza Personalizada y Dinámica de Programación**: Adaptar el contenido y los desafíos de programación a las preferencias y al nivel de habilidad de cada estudiante.
2. **Interacción y Soporte en Tiempo Real**: Utilizar un modelo IA de lenguaje avanzado (multimodal) para ofrecer asistencia instantánea y feedback sobre los ejercicios de programación.
3. **Fomentar el Pensamiento Crítico y la Resolución de Problemas**: Desarrollar habilidades esenciales en los estudiantes mediante la resolución de problemas de programación complejos y proyectos prácticos.
4. **Creación de una Comunidad de Aprendizaje**: Establecer un entorno donde los estudiantes puedan interactuar entre sí, compartir conocimientos y colaborar en proyectos.

FUNCIONALIDADES PRINCIPALES

1. Módulo de Aprendizaje Adaptativo:

- **Contenidos Dinámicos**: Cursos que abarcan desde fundamentos de programación hasta temas avanzados como desarrollo web, programación de aplicaciones móviles y ciencia de datos.
- **Personalización del Aprendizaje**: Adaptación de los contenidos y los retos según el progreso y las preferencias del estudiante, utilizando algoritmos de inteligencia artificial.

2. Interfaz de Codificación Interactiva:

- **Entorno de Desarrollo Integrado (IDE) Online**: Un espacio donde los estudiantes puedan escribir, probar y depurar su código directamente en la plataforma.

- **Ejercicios y Proyectos Prácticos**: Tareas que permitan a los estudiantes aplicar lo aprendido en proyectos reales y ejercicios de codificación.

3. Asistente Inteligente de Programación (AIP):

- **Soporte Instantáneo**: Uso de modelos de lenguaje multimodales avanzados para responder preguntas, explicar conceptos y proporcionar ejemplos de código en tiempo real.
- **Feedback Automatizado**: Evaluación inmediata de los ejercicios de codificación con sugerencias de mejora y correcciones.

4. Análisis de Desempeño y Retroalimentación:

- **Seguimiento del Progreso**: Monitoreo del avance de los estudiantes mediante dashboards que muestran estadísticas clave sobre su aprendizaje.
- **Evaluaciones y Certificaciones**: Pruebas regulares para evaluar el conocimiento adquirido y otorgar certificados de logro.

5. Comunidad y Colaboración:

- **Foros y Chats Grupales**: Espacios para que los estudiantes discutan temas de programación, resuelvan dudas y compartan recursos.
- **Proyectos Colaborativos**: Oportunidades para trabajar en grupo en proyectos de código abierto o competiciones de programación.

6. Interfaz de configuración pedagógica:

- Gestión de contenidos (multimediales) por el docente/instructor.
- Configuración de modelos de IA de lenguaje avanzado (multimodal).

Tecnología y Seguridad

- **Front-end**: HTML5, CSS3 y JavaScript con frameworks como React para crear una interfaz de usuario rica y responsiva.
- **Back-end**: Node.js o Python para manejar la lógica del servidor, integración con bases de datos y procesamiento de la interacción con el modelo de lenguaje.
- **Integración de IA**: Incorporación de modelos IA de lenguaje multimodales Open Source para facilitar la interacción inteligente.
- **Base de Datos**: Utilizar PostgreSQL o MongoDB para el almacenamiento de datos de usuario, progreso del curso y contribuciones al foro.
- **Seguridad**: Implementación de medidas de seguridad robustas para proteger la información personal y el trabajo de los estudiantes.

PUBLICO OBJETIVO

- Instituciones educativas públicas y privadas
- Docentes formales y no formales
- Estudiantes del sistema educativo formal
- Estudiantes del sistema educativo no formal

USUARIOS POTENCIALES

- Administradores de la plataforma
- Autoridades institucionales
- Docentes en general
- Tutores/colaboradores
- Estudiantes

EPICS DE REQUERIMIENTOS NO FUNCIONALES

EPIC 1: SEGURIDAD DEL SISTEMA

Descripción: Proteger el sistema y los datos de los usuarios contra accesos no autorizados, pérdidas de información y otros riesgos de seguridad.

Historias de Usuario:

1. **Como administrador, quiero implementar autenticación de dos factores para asegurar el acceso seguro a la plataforma.**

   Criterios de Aceptación:

   La autenticación de dos factores debe ser requerida para todos los usuarios al iniciar sesión en la plataforma.

   La implementación debe soportar al menos dos métodos de autenticación secundaria, como SMS y aplicaciones de autenticador.

   La funcionalidad de 2FA debe ser probada para garantizar que no se pueda acceder a ninguna cuenta sin la verificación correcta de ambos factores.

   Debe existir una opción para recuperar el acceso a la cuenta en caso de que el usuario pierda su segundo factor de autenticación, sin comprometer la seguridad.

   La implementación de 2FA debe ser documentada claramente para usuarios y administradores sobre cómo activarla y utilizarla.

2. **Como desarrollador, necesito cifrar los datos sensibles almacenados en la base de datos para proteger la información personal de los usuarios.**

   Criterios de Aceptación:

   Todos los datos personales sensibles (como números de identificación, información financiera, datos de contacto, etc.) almacenados en la base de datos deben estar cifrados tanto en reposo como en tránsito.

   El cifrado debe utilizar algoritmos estándar de la industria reconocidos por su robustez, como AES-256.

   Los desarrolladores deben asegurar que el cifrado y descifrado de datos no afecte significativamente el rendimiento de la aplicación.

   Debe haber políticas y procedimientos en lugar para la gestión segura de las claves de cifrado, incluyendo su rotación periódica y almacenamiento seguro.

   Se deben realizar pruebas para verificar que los datos cifrados no puedan ser leídos o alterados sin acceso a las claves de cifrado adecuadas.

3. **Como administrador, quiero realizar auditorías de seguridad periódicas para identificar y mitigar vulnerabilidades potenciales.**

   Criterios de Aceptación:

   Las auditorías de seguridad deben ser realizadas al menos semestralmente por un equipo interno o una tercera parte independiente y cualificada.

   Las auditorías deben incluir pruebas de penetración, revisión de código y evaluación de la infraestructura de TI para identificar vulnerabilidades.

   Se debe crear un informe detallado de cada auditoría que incluya los hallazgos, las recomendaciones y un plan de acción para abordar cualquier problema identificado.

   Los resultados de la auditoría y las acciones correctivas tomadas deben ser documentados y revisados con los stakeholders para garantizar la transparencia y la mejora continua de la seguridad.

   Se debe establecer un protocolo para la implementación rápida de medidas correctivas en respuesta a las vulnerabilidades identificadas en las auditorías.

EPIC 2: ESCALABILIDAD DEL SISTEMA

Descripción: Permitir que el sistema pueda expandirse en capacidad de forma eficiente, sin degradar el rendimiento o la calidad del servicio.

Historias de Usuario:

1. **Como diseñador de sistema, quiero diseñar una arquitectura que permita el escalado horizontal para manejar aumentos en la demanda de usuarios sin degradar el rendimiento.**

   Criterios de Aceptación:

   La arquitectura diseñada debe permitir añadir más servidores (o instancias) al sistema sin interrupción del servicio existente.

   Debe demostrarse mediante pruebas de carga que al duplicar la cantidad de servidores, el sistema puede manejar al menos el doble de la carga de trabajo sin degradación del rendimiento.

   El diseño debe incluir componentes de software que sean stateless (sin estado), permitiendo que cualquier solicitud de usuario pueda ser atendida por cualquier instancia del servidor.

   Se debe proporcionar documentación técnica detallada que explique cómo escalar el sistema, incluyendo guías para configurar y desplegar nuevas instancias.

Las pruebas deben confirmar que no hay puntos únicos de fallo en la arquitectura que puedan causar una interrupción del servicio durante el escalado.

2. **Como administrador de sistemas, necesito implementar balanceo de carga entre servidores para distribuir equitativamente las solicitudes de usuario.**

   Criterios de Aceptación:

   El balanceador de carga debe ser capaz de distribuir las solicitudes de usuarios de manera equitativa entre todos los servidores disponibles.

   Se deben realizar pruebas para verificar que el balanceador de carga redistribuye automáticamente las solicitudes a otros servidores en caso de fallo de uno de ellos.

   La implementación debe soportar diferentes algoritmos de balanceo de carga (como round-robin, menor conexión, IP hash) y permitir su configuración según las necesidades del sistema.

   Se debe documentar el proceso de configuración del balanceador de carga, incluyendo cómo añadir o remover servidores del pool de balanceo.

   Las pruebas de estrés deben demostrar que el tiempo de respuesta del sistema no se ve afectado negativamente por el balanceo de carga bajo diferentes condiciones de tráfico.

EPIC 3: ALTA DISPONIBILIDAD

Descripción: Asegurar que el sistema esté disponible para los usuarios según los niveles de servicio acordados.

Historias de Usuario:

1. **Como DevOps, quiero implementar un diseño de sistema con redundancia en los componentes críticos para asegurar una alta disponibilidad.**

   Criterios de Aceptación:

   Todos los componentes críticos del sistema (como bases de datos, servidores de aplicaciones y sistemas de manejo de estado) deben tener al menos un duplicado en funcionamiento.

   Las pruebas de fallo deben mostrar que en caso de caída de un componente crítico, su duplicado toma el control sin interrupción del servicio y sin pérdida de datos.

   La transición de control a los componentes redundantes debe completarse en un tiempo no superior a los criterios predefinidos aceptables para la operación del negocio (por ejemplo, menos de 1 minuto).

Debe existir una automatización para la detección de fallos y la conmutación a los componentes duplicados.

Se debe proporcionar documentación sobre la configuración de la redundancia, incluyendo guías para la operación y mantenimiento de los componentes duplicados.

2. **Como administrador de sistemas, necesito configurar y mantener servidores de respaldo activos para garantizar el servicio continuo durante fallos del servidor principal.**

   Criterios de Aceptación:

   Debe haber al menos un servidor de respaldo configurado y listo para tomar el control en caso de fallo del servidor principal.

   Las pruebas deben confirmar que el servidor de respaldo puede activarse automáticamente y asumir todas las funciones críticas del servidor principal sin intervención manual y en un tiempo máximo predefinido.

   El servidor de respaldo debe estar siempre sincronizado con el servidor principal, con una diferencia máxima de datos aceptable definida por las necesidades del negocio (por ejemplo, sincronización cada 5 minutos).

   Se debe realizar una auditoría mensual para verificar que los servidores de respaldo están completamente operativos y actualizados en términos de software y datos.

   La documentación debe incluir procedimientos detallados para la activación del servidor de respaldo, así como para la resolución de problemas y la restauración del servicio tras un fallo.

EPIC 4: RENDIMIENTO Y TIEMPO DE RESPUESTA

Descripción: Optimizar la velocidad de respuesta y la eficiencia del procesamiento del software para asegurar que cumpla con las expectativas y necesidades del usuario.

Historias de Usuario:

1. **Como desarrollador, quiero optimizar las consultas de base de datos para reducir los tiempos de carga de las páginas web.**

   Criterios de Aceptación:

   Las consultas optimizadas deben demostrar una reducción del tiempo de ejecución en un porcentaje significativo comparado con las versiones anteriores, por ejemplo, una mejora del 50% en los tiempos de carga.

   La optimización debe ser realizada sin afectar la integridad de los datos; todas las consultas optimizadas deben retornar los mismos resultados que antes de su optimización.

Se deben realizar pruebas de rendimiento en un entorno de prueba que simule el tráfico de usuario real para validar la mejora en los tiempos de carga.

Las optimizaciones aplicadas deben estar documentadas claramente, incluyendo detalles sobre los cambios realizados y las razones detrás de estos cambios.

Los cambios en la base de datos deben ser revisados y aprobados por al menos un experto en bases de datos del equipo para asegurar que no se introduzcan nuevos problemas de rendimiento o bugs.

2. **Como desarrollador de software, necesito implementar técnicas de caché para mejorar el tiempo de respuesta del sistema.**

   Criterios de Aceptación:

   La implementación de caché debe reducir el tiempo de respuesta del sistema para las operaciones que involucran datos frecuentemente accesados, con una mejora objetivo, por ejemplo, reducción de tiempos de respuesta en un 40%.

   Debe haber mecanismos para invalidar la caché de manera adecuada cuando los datos subyacentes cambian, para asegurar que la información proporcionada a los usuarios es actual y precisa.

   La solución de caché debe ser escalable y fácilmente configurable para ajustarse a diferentes volúmenes de datos y patrones de acceso.

   Se debe documentar la configuración de la caché, incluyendo parámetros como tiempo de vida del dato en caché, estrategias de evicción y configuraciones de precarga.

   Las pruebas de integración deben verificar que la caché funciona como se espera tanto en condiciones normales de operación como bajo carga alta, sin causar degradación del servicio o errores.

EPIC 5: MODELADO DE LA BASE DE DATOS (RELACIONAL)

Diseño y Optimización de la Base de Datos.

Historia 1: Diseño de un esquema de base de datos normalizado

Como administrador BD, quiero diseñar un esquema de base de datos normalizado para minimizar la redundancia y maximizar la integridad de los datos.

Criterios de Aceptación:

1. El esquema debe estar normalizado al menos hasta la Tercera Forma Normal (3NF) para minimizar la redundancia.

2. Debe existir documentación que describa claramente todas las entidades, atributos y relaciones entre las tablas.
3. Las relaciones entre tablas deben asegurar la integridad referencial.
4. Se debe realizar una revisión por pares del diseño con al menos un arquitecto de datos y un desarrollador backend.

Historia 2: Implementación de índices en columnas frecuentemente consultadas

Como desarrollador, necesito implementar índices en las columnas más consultadas para mejorar el rendimiento de las búsquedas.

Criterios de Aceptación:

1. Los índices deben ser creados en las columnas identificadas como las más utilizadas en las consultas, basadas en un análisis previo de uso.
2. Se debe demostrar, mediante pruebas de rendimiento, una mejora en el tiempo de respuesta de las consultas más frecuentes.
3. La implementación de índices no debe degradar significativamente el rendimiento de inserción en la base de datos.

Historia 3: Implementación de procedimientos almacenados

Como administrador de base de datos, quiero implementar procedimientos almacenados para manejar operaciones comunes y mejorar la seguridad del acceso a los datos.

Criterios de Aceptación:

1. Los procedimientos almacenados deben implementarse para todas las operaciones CRUD (Crear, Leer, Actualizar, Borrar) comunes.
2. Cada procedimiento almacenado debe tener manejo de errores para tratar los fallos de ejecución de manera adecuada.
3. Debe haber documentación técnica para cada procedimiento, incluyendo parámetros de entrada y formato de los resultados.

Historia 4: Encriptación de columnas de datos sensibles

Como encargado de seguridad, necesito encriptar las columnas que almacenan datos sensibles para proteger la información de los usuarios contra accesos no autorizados.

Criterios de Aceptación:

1. Las columnas que contienen datos personales sensibles (por ejemplo, identificaciones personales, información financiera) deben estar cifradas en reposo.

2. El método de cifrado utilizado debe cumplir con las normativas de seguridad de datos aplicables (como GDPR, HIPAA, etc.).
3. Se debe verificar la encriptación mediante pruebas que demuestren que los datos no pueden ser leídos sin la clave de descifrado adecuada.

Historia 5: Establecimiento de restricciones de integridad y relaciones entre tablas

Como desarrollador, quiero establecer restricciones de integridad y relaciones entre tablas para asegurar la consistencia de los datos a través de la aplicación.

Criterios de Aceptación:

1. Todas las restricciones de integridad (por ejemplo, claves primarias, claves foráneas, restricciones de unicidad) deben ser configuradas correctamente en el esquema de la base de datos.
2. Las relaciones entre tablas deben permitir la cascada adecuada de actualizaciones y eliminaciones para mantener la consistencia de los datos.
3. Se debe realizar una validación para asegurar que no existan registros huérfanos ni violaciones de integridad en las transacciones de prueba.

EPIC 6: MODELADO DE LA BASE DE DATOS (NO SQL – MONGO DB)

Historia 1: Diseño de un esquema de base de datos flexible

Criterios de Aceptación:

1. El diseño del esquema debe soportar la flexibilidad del esquema de documentos de MongoDB, permitiendo variaciones en la estructura de datos entre documentos.
2. Debe existir documentación que describa claramente las colecciones, documentos y campos utilizados, incluyendo los tipos de datos y los índices aplicados.
3. Se debe demostrar la capacidad del esquema para evolucionar sin necesidad de migraciones costosas de datos.

Historia 2: Implementación de índices para optimizar consultas

Criterios de Aceptación:

1. Se deben crear índices en los campos que son frecuentemente utilizados en las consultas para mejorar el rendimiento de lectura.
2. Las pruebas de rendimiento deben mostrar una mejora significativa en los tiempos de respuesta de las consultas después de la indexación.

3. Debe evaluarse el impacto de la indexación en el rendimiento de escritura y en el uso de espacio de almacenamiento.

Historia 3: Uso de agregaciones para manejar consultas complejas

Criterios de Aceptación:

1. Se deben implementar pipelines de agregación para procesar datos y realizar cálculos complejos dentro de la base de datos.
2. Cada pipeline de agregación debe estar bien documentado con ejemplos de entrada y salida.
3. Las agregaciones deben ser optimizadas para minimizar el tiempo de procesamiento y el uso de recursos.

Historia 4: Encriptación de datos sensibles

Criterios de Aceptación:

1. Los datos sensibles almacenados en documentos deben estar cifrados en reposo utilizando la encriptación ofrecida por MongoDB o soluciones de terceros.
2. El cifrado debe cumplir con los estándares de seguridad aplicables y ser verificado por pruebas de seguridad.
3. Se debe proporcionar un método seguro para la gestión de las claves de cifrado.

Historia 5: Validación de datos al nivel de la base de datos

Criterios de Aceptación:

1. Deben establecerse reglas de validación en las colecciones para asegurar que los datos ingresados cumplan con los formatos y restricciones requeridos.
2. Se debe probar la validación mediante inserciones de prueba para confirmar que los datos inválidos son rechazados.
3. Las reglas de validación deben permitir futuras modificaciones sin interrupciones significativas en el servicio.

EPICS FUNCIONALES

EPIC 1: MÓDULO DE APRENDIZAJE ADAPTATIVO

Descripción: crear un entorno de aprendizaje que se adapte automáticamente a las necesidades individuales de los estudiantes, utilizando algoritmos avanzados de inteligencia artificial para personalizar el contenido y los desafíos de programación. La meta es mejorar la experiencia de aprendizaje al ajustar la dificultad y los temas según el progreso y las preferencias de cada usuario, facilitando un aprendizaje más eficaz y personalizado.

Historias de Usuario:

1. **Como estudiante, quiero que el sistema adapte los contenidos de aprendizaje basados en mi progreso y retroalimentación para mejorar mi experiencia de aprendizaje.**

   Criterios de Aceptación:

   El sistema debe rastrear automáticamente el progreso y la interacción del estudiante con el material de aprendizaje y utilizar estos datos para ajustar el contenido y la dificultad de futuras lecciones y ejercicios.

   Se deben realizar pruebas para verificar que los ajustes en el contenido y la dificultad se basan en métricas objetivas de desempeño del estudiante, como las calificaciones en pruebas y la frecuencia de consulta de temas específicos.

   El estudiante debe recibir recomendaciones personalizadas sobre qué temas estudiar a continuación, basadas en su rendimiento anterior y en las áreas que requieren mayor refuerzo.

   Los estudiantes deben poder visualizar un dashboard que refleje su progreso y cómo el sistema está adaptando el contenido a sus necesidades.

   Se debe obtener una valoración de satisfacción del estudiante sobre la personalización del contenido, con un índice de satisfacción objetivo de al menos el 80%.

2. **Como docente, necesito tener la capacidad de personalizar los retos y ejercicios según las necesidades de mis estudiantes.**

   Criterios de Aceptación:

   Los docentes deben tener la capacidad de modificar el contenido, la dificultad y los tipos de ejercicios proporcionados a los estudiantes a través de una interfaz sencilla e intuitiva.

   Debe existir la funcionalidad para que los docentes puedan asignar tareas específicas a estudiantes individuales o grupos basándose en su rendimiento académico y necesidades educativas.

Las personalizaciones hechas por el docente deben ser aplicadas inmediatamente en el entorno de aprendizaje del estudiante y reflejadas la próxima vez que el estudiante acceda al sistema.

Se deben realizar pruebas para asegurar que las modificaciones hechas por el docente no interfieran con los algoritmos de personalización automática del sistema, excepto cuando sea explícitamente deseado por el docente.

Se debe recopilar feedback de los docentes sobre la facilidad de uso y efectividad de las herramientas de personalización, con el objetivo de alcanzar una calificación de satisfacción del 90%.

EPIC 2: INTERFAZ DE CODIFICACIÓN INTERACTIVA

Descripción: desarrollar un entorno de desarrollo integrado (IDE) online accesible directamente en la plataforma. Este IDE permitirá a los estudiantes escribir, ejecutar, probar y depurar su código en un ambiente controlado y soportado por herramientas educativas. Además, incluirá ejercicios prácticos y proyectos que motivarán a los estudiantes a aplicar lo aprendido y a desarrollar habilidades de programación prácticas y relevantes.

Historias de Usuario:

1. **Como estudiante, quiero utilizar un IDE online integrado para escribir, probar y depurar mi código directamente en la plataforma.**

   Criterios de Aceptación:

   El IDE debe permitir a los estudiantes escribir, ejecutar y depurar código en una variedad de lenguajes de programación, incluidos al menos Java, Python y JavaScript.

   Debe soportar la sintaxis de resaltado, autocompletado y detección de errores en tiempo real para mejorar la experiencia de codificación.

   El tiempo de respuesta desde que se ejecuta el código hasta que se muestra el resultado no debe exceder los 2 segundos en condiciones normales de operación.

   El IDE debe permitir a los estudiantes guardar y recuperar sus proyectos de código en cualquier momento.

   Debe haber documentación disponible dentro de la plataforma que explique cómo utilizar todas las características del IDE.

2. **Como estudiante, necesito recibir ejercicios prácticos que me permitan aplicar lo aprendido en proyectos reales.**

   Criterios de Aceptación:

   Los ejercicios prácticos deben cubrir un rango de dificultad desde principiante hasta avanzado, permitiendo a los estudiantes progresar a su propio ritmo.

Cada ejercicio debe incluir una descripción clara de los objetivos de aprendizaje, instrucciones detalladas, y criterios de evaluación específicos.

Debe existir un mecanismo de envío y retroalimentación automática que proporcione a los estudiantes comentarios instantáneos sobre su código, incluyendo errores y sugerencias de mejora.

Los estudiantes deben tener la opción de ver ejemplos de soluciones correctas después de completar los ejercicios, para comparar con sus propios intentos.

Se debe recolectar feedback de los estudiantes sobre la utilidad de los ejercicios en su proceso de aprendizaje, con el objetivo de mantener una calificación de satisfacción del usuario de al menos el 85%.

EPIC 3: ASISTENTE INTELIGENTE DE PROGRAMACIÓN (AIP)

Descripción: implementar de un asistente inteligente de programación alimentado por modelos avanzados de IA de lenguaje multimodal. El asistente ofrecerá soporte en tiempo real, retroalimentación instantánea, ejemplos de código y explicaciones detalladas, ayudando a los estudiantes a resolver dudas de forma eficiente y a mejorar su comprensión de los conceptos de programación.

Historias de Usuario:

1. **Como estudiante, quiero recibir asistencia instantánea y feedback sobre mi código de un modelo de IA de lenguaje para mejorar mi aprendizaje.**

   Criterios de Aceptación:

   El asistente debe ser capaz de analizar y proporcionar feedback sobre el código escrito por el estudiante en un tiempo máximo de 5 segundos después de la solicitud.

   El feedback debe incluir tanto errores sintácticos como sugerencias de mejora en cuanto a las prácticas de codificación, con explicaciones claras y concisas de los errores.

   El asistente debe ser capaz de interactuar en al menos los lenguajes de programación más comunes utilizados en la plataforma, como Java, Python, y JavaScript.

   Debe haber una función para que el estudiante pueda solicitar múltiples rondas de feedback para el mismo fragmento de código, permitiendo un diálogo iterativo con el asistente.

   El asistente debe mostrar una alta precisión en las correcciones y sugerencias proporcionadas, con una tasa de exactitud mínima del 90% en las pruebas validadas internamente.

2. **Como estudiante, necesito que el sistema me proporcione ejemplos de código y explicaciones en tiempo real cuando tengo dudas.**

Criterios de Aceptación:

El asistente debe ofrecer ejemplos de código relevantes a las preguntas o problemas planteados por los estudiantes dentro de los 3 segundos siguientes a la solicitud.

Las explicaciones proporcionadas deben ser claras y ajustarse al nivel de comprensión y experiencia del estudiante, basándose en su historial de interacción con la plataforma.

El asistente debe ser capaz de generar ejemplos y explicaciones en una variedad de contextos de programación, asegurando la relevancia y la aplicabilidad a las tareas del estudiante.

Los ejemplos de código y las explicaciones deben ser verificables, es decir, el código proporcionado debe ser ejecutable y producir los resultados esperados cuando se prueba en el IDE de la plataforma.

Se debe recoger feedback continuo de los estudiantes sobre la utilidad de los ejemplos y las explicaciones para ajustar y mejorar la capacidad de respuesta y precisión del asistente.

EPIC 4: ANÁLISIS DE DESEMPEÑO Y RETROALIMENTACIÓN

Descripción: proporcionar herramientas analíticas que permitan tanto a estudiantes como a docentes monitorear y evaluar el progreso en el aprendizaje. Incluirá la implementación de dashboards y reportes que muestren estadísticas detalladas sobre el rendimiento de los estudiantes, ayudando a identificar áreas de fortaleza y de mejora, y facilitando la certificación de habilidades adquiridas.

Historias de Usuario:

1. **Como estudiante, quiero ver estadísticas detalladas sobre mi progreso en el aprendizaje para evaluar mis fortalezas y áreas de mejora.**

   Criterios de Aceptación:

   El dashboard del estudiante debe mostrar estadísticas que incluyan al menos el porcentaje de cursos completados, puntuaciones en pruebas, tiempo promedio de estudio por sesión y progreso en habilidades específicas.

   Las estadísticas deben actualizarse en tiempo real o con una latencia no mayor a 24 horas después de completar una actividad de aprendizaje.

   Los estudiantes deben poder acceder a visualizaciones gráficas (por ejemplo, gráficos de barras, líneas o círculos) que faciliten la interpretación rápida de su progreso y rendimiento.

Debe ser posible para los estudiantes filtrar y desglosar sus datos por curso, tema o período de tiempo para un análisis más detallado.

Se debe realizar pruebas de usabilidad para asegurar que los estudiantes encuentren el dashboard intuitivo y útil, con una tasa de satisfacción del usuario de al menos el 85%.

2. **Como docente, necesito poder acceder a informes de desempeño de mis estudiantes para adecuar mi enseñanza.**

   Criterios de Aceptación:

   Los docentes deben tener acceso a informes que proporcionen un análisis detallado del rendimiento de cada estudiante, así como agregados por clase o grupo de estudio.

   Los informes deben incluir métricas clave como la asistencia, las calificaciones promedio, la participación en foros y actividades, y el progreso en competencias específicas.

   Debe existir la funcionalidad para que los docentes generen informes personalizados basados en diferentes parámetros, como períodos de tiempo, tipos de actividad o categorías de habilidades.

   Los informes deben ser exportables en formatos comunes como PDF o Excel para facilitar su análisis y compartirlos con otros educadores o administradores.

   Se debe garantizar la protección de la privacidad y los datos de los estudiantes, cumpliendo con las regulaciones de protección de datos pertinentes como GDPR o FERPA en su manejo y visualización.

EPIC 5: COMUNIDAD Y COLABORACIÓN

Descripción**:** fomentar una comunidad de aprendizaje activa y colaborativa. Se creará un espacio para que los estudiantes interactúen mediante foros y chats grupales, y participen en proyectos colaborativos. Estos espacios permitirán el intercambio de conocimientos, la resolución colaborativa de problemas y la participación en competiciones de programación, enriqueciendo la experiencia educativa y fomentando una red de apoyo entre pares.

Historias de Usuario:

1. **Como estudiante, quiero participar en foros y chats grupales para discutir temas de programación con mis compañeros.**

   Criterios de Aceptación:

   La plataforma debe proporcionar foros y salas de chat grupales donde los estudiantes puedan iniciar discusiones, responder a preguntas y compartir recursos.

Los foros y chats deben soportar funciones básicas como publicar mensajes, responder a otros usuarios, editar sus propios mensajes y buscar hilos de discusión.

Debe existir un sistema de moderación que permita a los administradores y a ciertos usuarios designados gestionar el contenido de los foros para asegurar un ambiente respetuoso y constructivo.

Los foros y chats deben ser accesibles en cualquier dispositivo con acceso a internet y deben cargar y funcionar correctamente en menos de 3 segundos bajo una conexión estándar.

Se debe realizar una encuesta de satisfacción entre los estudiantes para asegurar que los foros y chats son útiles y mejoran su experiencia de aprendizaje, apuntando a una tasa de satisfacción de al menos el 80%.

2. **Como estudiante, necesito colaborar en proyectos de código abierto para ganar experiencia práctica en un entorno de equipo.**

   Criterios de Aceptación:

   La plataforma debe permitir a los estudiantes formar grupos y trabajar en proyectos de código abierto, con la capacidad de compartir y editar código en tiempo real.

   Debe haber herramientas integradas para la gestión de proyectos, como tableros de tareas, sistemas de seguimiento de errores y versiones, y un repositorio de código.

   Los proyectos deben permitir la integración con herramientas externas de control de versiones, como Git, para facilitar la colaboración y el seguimiento del progreso.

   Debe proporcionarse documentación y tutoriales sobre cómo los estudiantes pueden iniciar, gestionar y colaborar en proyectos, asegurando que los usuarios comprendan cómo hacer uso efectivo de estas herramientas.

   Se deben realizar pruebas para asegurar que los estudiantes puedan interactuar sin problemas técnicos y que los datos del proyecto estén seguros y respaldados regularmente.

EPIC 6: INTERFAZ DE CONFIGURACIÓN PEDAGÓGICA

Descripción: facilitar a los docentes y administradores la gestión y actualización de contenidos educativos, así como la configuración de los modelos de IA utilizados en la plataforma. Esto incluirá herramientas para ajustar la dificultad y los tipos de contenido ofrecidos, permitiendo una personalización detallada y eficaz que se adapte a las necesidades cambiantes de los estudiantes y del entorno educativo.

Historias de Usuario:

1. **Como docente, quiero gestionar y actualizar los contenidos de cursos para mantenerlos relevantes y actualizados.**

   Criterios de Aceptación:

   Los docentes deben poder añadir, editar y eliminar contenido de los cursos a través de una interfaz intuitiva y segura en la plataforma.

   La plataforma debe soportar diferentes tipos de contenido, incluyendo texto, video, y recursos interactivos, y permitir a los docentes incorporar estos formatos fácilmente en los cursos.

   Cada cambio en el contenido debe registrarse en un historial de versiones accesible para los docentes, permitiendo revertir a versiones anteriores si es necesario.

   Los cambios en el contenido deben reflejarse en tiempo real para los estudiantes una vez publicados por el docente.

   Se debe realizar pruebas para asegurar que la plataforma soporta la carga de usuarios simultáneos gestionando contenidos sin degradar el rendimiento ni la funcionalidad.

2. **Como administrador, necesito configurar y ajustar modelos de IA para optimizar la asistencia proporcionada a los usuarios.**

   Criterios de Aceptación:

   Los administradores deben tener la capacidad de configurar y ajustar los parámetros de los modelos de IA utilizados para la asistencia y personalización del aprendizaje a través de una interfaz de usuario dedicada.

   La plataforma debe permitir la implementación de actualizaciones y mejoras en los modelos de IA sin interrumpir el servicio activo ni comprometer los datos existentes.

   Debe existir un sistema de registro que detalle todos los cambios realizados en la configuración de los modelos de IA, incluyendo quién hizo el cambio y cuándo.

   Los ajustes en los modelos de IA deben ser probados para verificar su efectividad en mejorar las recomendaciones de aprendizaje y la precisión de las respuestas del asistente de IA.

   La plataforma debe incluir medidas de seguridad robustas para proteger la configuración de los modelos de IA y los datos utilizados para su entrenamiento y operación.