Analista Programador

Diseño y desarrollo de aplicaciones

Ingeniería de Software

“Retro Trivia”

Leonardo Suárez
Francisco Machado
Ángel Machado

2024

Índice

1. Introducción

El presente proyecto consiste en el desarrollo de un juego educativo de preguntas y respuestas, con la particularidad de que utilizará inteligencia artificial (IA) para generar preguntas dinámicas, enriqueciendo la experiencia de usuario. Este juego estará disponible tanto en modo individual como multijugador, lo que permitirá escalarlo para diferentes tipos de usuarios.

Para lograr este objetivo, se hará uso de APIs externas, como ChatGPT, para generar preguntas adaptadas a las preferencias de los jugadores, y Google Cloud Text-to-Speech, que permitirá una accesibilidad más inclusiva al transformar texto en voz. El uso de estas tecnologías busca mejorar la interacción del jugador, brindando una experiencia personalizada y fluida.

Este proyecto forma parte de una colaboración interdisciplinaria entre las materias de Desarrollo de Aplicaciones Distribuidas (DDA) e Ingeniería de Software, aplicando conceptos de arquitectura de software, metodologías ágiles, e integración de sistemas distribuidos para crear una solución completa y escalable.

1.1 Presentación del problema

Hoy en día, existen pocos juegos educativos que aprovechen las capacidades de la inteligencia artificial para generar contenido dinámico. Además, la mayoría de los juegos de preguntas están diseñados para un solo jugador o no permiten la escalabilidad para multijugador. Este proyecto busca resolver esta carencia creando un juego que se pueda adaptar a diferentes niveles de conocimiento y que ofrezca modos de juego tanto individual como multijugador.

1.2 Trazabilidad entre necesidades y objetivos

La trazabilidad asegura que todas las necesidades identificadas tengan un objetivo relacionado y que no se deje de lado ningún aspecto crítico del proyecto. A continuación, la tabla con la relación entre necesidades y objetivos:

Necesidad	**Objetivo relacionado**
Interfaz intuitiva	Crear una interfaz gráfica amigable e intuitiva
Generación dinámica de preguntas	Integrar la API de ChatGPT para la generación de preguntas dinámicas
Sistema de autenticación	Implementar sistemas de autenticación y seguridad
Sistema de temporización	Añadir un sistema de cronómetro para la limitación del tiempo de respuesta
Persistencia de datos	Desarrollar un sistema de persistencia que almacene el progreso y resultados de los jugadores
Funcionalidad multijugador en tiempo real	Añadir funcionalidad de juego en tiempo real para varios jugadores simultáneamente
Escalabilidad del sistema	Garantizar que el sistema pueda soportar múltiples jugadores al mismo tiempo, tanto en modo individual como multijugador

1.3 Casos de Uso

1.3.1. Caso de Uso 1: Registro y Autenticación del Jugador

Elemento	**Descripción**
CU Nº	1
RF Nº	1
Autores	Jugador
Versión	1.0
Descripción	El jugador debe registrarse con un correo electrónico y contraseña únicos para poder acceder al juego. Luego podrá autenticarse usando las mismas credenciales.
Precondición	El jugador no debe estar registrado previamente en el sistema.
Poscondición	El jugador queda registrado en el sistema y puede autenticarse para jugar.
Curso Normal	1. El jugador accede a la pantalla de registro. 2. El jugador completa los campos de correo y contraseña. 3. El sistema verifica los datos y confirma el registro.
Curso Alternativo	3.A Si el correo ya está registrado, el sistema informa del error y solicita un correo diferente. 3.B Si la contraseña no cumple con los requisitos, se muestra un mensaje de error.
Rendimiento	Medio

Frecuencia esperada	Alta
Importancia	Alta
Urgencia	Alta

1.3.2. Caso de Uso 2: Selección de Categoría de Preguntas

Elemento	Descripción
CU Nº	2
RF Nº	2
Autores	Jugador
Versión	1.0
Descripción	El jugador debe seleccionar una categoría de preguntas antes de iniciar el juego. Una ruleta de categorías se mostrará para que el jugador elija.
Precondición	El jugador está autenticado en el sistema.
Poscondición	El sistema selecciona la categoría elegida y carga preguntas relacionadas con dicha categoría.
Curso Normal	1. El jugador accede a la pantalla de selección de categoría. 2. La ruleta muestra varias opciones de categorías. 3. El jugador elige una categoría y comienza el juego.
Curso Alternativo	3.A Si no se selecciona una categoría en un tiempo determinado, el sistema selecciona una por defecto.

Rendimiento	Medio
Frecuencia esperada	Alta
Importancia	Alta
Urgencia	Media

1.3.3. Generación de Preguntas Dinámicas

Elemento	**Descripción**
CU Nº	3
RF Nº	3
Autores	Sistema
Versión	1.0
Descripción	El sistema debe generar preguntas dinámicas basadas en la categoría seleccionada por el jugador utilizando la API de ChatGPT.
Precondición	El jugador ha seleccionado una categoría de preguntas.
Poscondición	El sistema genera una nueva pregunta dinámica relacionada con la categoría elegida y muestra las opciones de respuesta.
Curso Normal	1. El sistema envía una solicitud a la API de ChatGPT para obtener una pregunta de la categoría seleccionada. 2. La API responde con una pregunta dinámica y opciones de

	respuesta. 3. El sistema muestra la pregunta al jugador.
Curso Alternativo	2.A Si la API no responde, el sistema selecciona una pregunta predefinida para evitar demoras en el juego.
Rendimiento	Medio
Frecuencia esperada	Alta
Importancia	Alta
Urgencia	Alta

1.4 Especificación de Requerimientos

1.4.1. Requerimientos funcionales

- RF-01: El sistema debe permitir a los usuarios registrarse y autenticarse utilizando un correo electrónico y contraseña únicos.
- RF-02: El sistema debe mostrar una ruleta con categorías de preguntas para que los jugadores puedan seleccionar la categoría de su preferencia.
- RF-03: El sistema debe integrarse con la API de ChatGPT para generar preguntas dinámicas basadas en las categorías seleccionadas por los jugadores.
- RF-04: El sistema debe permitir tanto juegos individuales como multijugador.

1.4.2. Requerimientos funcionales

- RNF-01: El sistema debe ser escalable para soportar hasta 1000 jugadores simultáneamente.
- RNF-02: El sistema debe tener una latencia mínima, garantizando que las preguntas y respuestas se muestren en menos de 2 segundos.
- RNF-03: La interfaz gráfica debe ser intuitiva y seguir principios de usabilidad, accesible para jugadores de todas las edades.
- RNF-04: La plataforma debe cumplir con estándares de seguridad y protección de datos, en conformidad con regulaciones como la GDPR

1.5 Alcance y Limitaciones

Alcance:

- Implementación de un juego educativo de preguntas y respuestas con modalidades de juego individual y multijugador en tiempo real.
- Integración con la API de ChatGPT para generar preguntas dinámicas.
- Sistema de autenticación y seguridad para jugadores, almacenando los datos de progreso y resultados.
- Se incluirá una ruleta de categorías para la selección de preguntas.
- Persistencia de los datos de los jugadores, incluyendo resultados de las partidas.

Limitaciones:

- No se incluirá personalización avanzada de avatares o perfiles de jugadores.
- El modo multijugador se limitará a un máximo de 2 jugadores por partida en la primera iteración.
- El sistema no incluirá análisis avanzados de las respuestas de los jugadores en la primera versión.

1.6 Estudio de Alternativas (y selección)

Para el desarrollo del juego, se consideraron varias tecnologías y APIs:

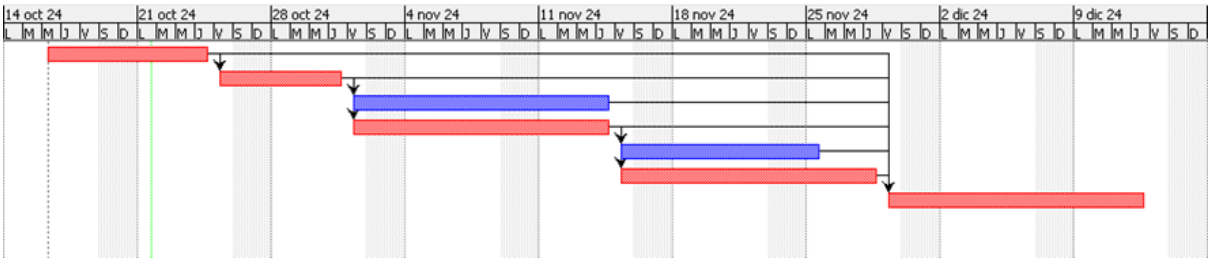
- APIs para generación de preguntas: Se evaluaron ChatGPT y GPT-4 frente a otros generadores de contenido como Open Trivia API. Selección: ChatGPT fue seleccionado debido a su capacidad de generar contenido más avanzado y dinámico.
- Backend para la gestión del juego: Se compararon Sockets vs REST API para manejar el modo multijugador en tiempo real. Selección: Se optará por Sockets, ya que permite la comunicación bidireccional en tiempo real, lo cual es clave para el juego multijugador.
- Almacenamiento en la nube: Se compararon Google Cloud y Amazon Web Services (AWS) para la persistencia de datos. Selección: Google Cloud fue seleccionada por su integración más directa con Google Text-to-Speech y otros servicios que planeamos utilizar.

1.7 Plan de Proyecto

El desarrollo del proyecto se dividirá en iteraciones de dos semanas, siguiendo una metodología ágil. Habrá 4 iteraciones en total, cada una con una duración de 2 semanas. Al finalizar cada sprint, se entregará una nueva funcionalidad o mejora, permitiendo recibir retroalimentación constante.

A continuación, el **cronograma de tareas** (en formato Gantt):

	Nombre	Duracion	Inicio	Terminado	Predecesores
1	Análisis de requerimientos	7 days	16/10/24 08:00 AM	24/10/24 05:00 PM	
2	Diseño de la Arquitectura	5 days	25/10/24 08:00 AM	31/10/24 05:00 PM	1
3	Desarrollo del Frontend	10 days	01/11/24 08:00 AM	14/11/24 05:00 PM	2
4	Desarrollo del Backend	10 days	01/11/24 08:00 AM	14/11/24 05:00 PM	2
5	Integración con la API ChatG	7 days	15/11/24 08:00 AM	25/11/24 05:00 PM	4
6	Implementación del Multijuga	10 days	15/11/24 08:00 AM	28/11/24 05:00 PM	4
7	Pruebas y Ajustes	10 days	29/11/24 08:00 AM	12/12/24 05:00 PM	1;2;3;4;5;6



Este cronograma es una aproximación inicial y puede ajustarse durante el desarrollo, dependiendo de los tiempos de cada iteración.

1.8 Ciclo de Vida y Metodología Seleccionada

La metodología seleccionada para el desarrollo del proyecto será Scrum iterativo, un enfoque ágil que nos permitirá adaptarnos rápidamente a los cambios y mejorar la entrega de valor continuo. Cada sprint tendrá una duración de 2 semanas, con una revisión y retrospectiva al final de cada iteración.

Roles:

- **Product Owner**: Definirá las prioridades y los objetivos clave del juego.
- **Scrum Master**: Asegurará que el equipo siga los principios ágiles y elimine impedimentos.
- **Equipo de desarrollo**: Encargado del desarrollo del frontend, backend, y la integración de APIs.

Sprints:

- **Sprint 1**: Definir el alcance del sistema y realizar el diseño de la arquitectura.
- **Sprint 2**: Desarrollar la funcionalidad básica de la interfaz y la autenticación.
- **Sprint 3**: Implementar la generación dinámica de preguntas.

- **Sprint 4**: Implementar el modo multijugador y realizar pruebas finales.

Ciclo de Vida del Proyecto - Enfoque Iterativo

Dado que en este proyecto implementamos **sprints** y seguiremos una metodología **ágil** con entrega incremental, el ciclo de vida sigue un enfoque iterativo donde desarrollamos funcionalidades en fases sucesivas. Este es un ciclo de vida **iterativo incremental**, donde se realiza un ciclo completo de trabajo en cada sprint.

Iniciación del Proyecto

Objetivos:

- Definir los objetivos generales del proyecto.
- Identificar los stakeholders (partes interesadas), es decir, todos aquellos que tengan algún interés o influencia en el proyecto.
- Definir el alcance inicial del proyecto, así como las funcionalidades que se desarrollarán.

Actividades:

- Definición de requerimientos iniciales.
- Establecimiento del equipo y recursos.
- Creación de un cronograma inicial (como el Gantt que mencionamos).
- Documentación del alcance del proyecto (qué se hará y qué no).

Análisis y Requerimientos (Sprint 1)

Objetivos:

- Analizar las necesidades del usuario y las funcionalidades del sistema.
- Definir los requerimientos funcionales y no funcionales del sistema.
- Priorizar las funcionalidades que serán implementadas.

Actividades:

- Reunir los requerimientos de todas las partes interesadas.
- Crear historias de usuario para cada funcionalidad clave.
- Definir las especificaciones del sistema: qué hace y cómo debería comportarse.
- Crear los criterios de aceptación de cada funcionalidad.

Entregables:

- Documento de especificación de requerimientos.
- Lista de historias de usuario.
- Diagrama de casos de uso.

Diseño de la Arquitectura del Sistema (Sprint 2)

Objetivos:

- Diseñar la arquitectura del sistema y la estructura general del código.
- Asegurarse de que el sistema siga principios de diseño como SOLID y GRASP.

Actividades:

- Crear el diagrama de clases del sistema.
- Diseñar la estructura del frontend (HTML, CSS, JS) y el backend (Spring Boot, WebSocket).
- Identificar las interfaces y servicios que permitirán el desacoplamiento y extensibilidad del código.
- Establecer la configuración inicial del proyecto: repositorios, estructura de archivos, servicios, controladores, etc.

Entregables:

- Diagrama de arquitectura del sistema.
- Estructura inicial del proyecto: organización de carpetas y módulos de frontend y backend.
- Diagrama de base de datos (si es necesario).

Desarrollo del Sistema (Sprints 3 a 6)

Objetivos:

- Implementar las funcionalidades clave del sistema, de forma incremental a lo largo de varios sprints.
- Integrar el frontend y el backend mediante APIs REST y WebSocket.

Actividades:

- Sprint 3: Desarrollo del frontend con la interfaz inicial y lógica de usuario.
- Sprint 4: Desarrollo del backend con los servicios y controladores.
- Sprint 5: Implementación de la ruleta de categorías y la lógica de preguntas.
- Sprint 6: Implementación del sistema multijugador y comunicación en tiempo real.

Entregables:

- Código funcional del frontend y backend.
- Integración del sistema con la API de ChatGPT.
- Módulo multijugador con WebSocket.
- Primeras versiones funcionales del sistema entregadas y probadas en cada sprint.

Integración y Pruebas (Sprint 7)

Objetivos:

- Probar la funcionalidad del sistema en su conjunto.
- Asegurarse de que las funcionalidades individuales trabajen correctamente en conjunto.
- Verificar el rendimiento y seguridad del sistema.

Actividades:

- Realizar pruebas unitarias de cada componente del sistema.
- Realizar pruebas de integración para asegurar que el frontend y backend interactúan correctamente.
- Realizar pruebas de aceptación para validar que el sistema cumpla con los requerimientos de las partes interesadas.
- Probar el sistema multijugador y la sincronización entre jugadores.
- Corrección de errores y ajustes según los resultados de las pruebas.

Entregables:

- Informe de pruebas con los resultados.
- Sistema ajustado y corregido según las pruebas.
- Entrega del sistema completamente funcional.

1.9 Estimación de Tiempos y Recursos

Estimación con Planning Poker

Se dividen las historias de usuario según las funcionalidades principales del sistema.

Registro de usuario:

Descripción: Como usuario, quiero registrarme en el sistema para poder acceder al juego.

Estimación inicial: 3 puntos (baja complejidad).

Autenticación de usuario:

Descripción: Como usuario registrado, quiero poder iniciar sesión para acceder a mis partidas.

Estimación inicial: 3 puntos.

Mostrar preguntas dinámicas:

Descripción: Como jugador, quiero que se me presenten preguntas dinámicas para responder durante la partida.

Estimación inicial: 5 puntos (complejidad media).

Implementar el cronómetro:

Descripción: Como jugador, quiero que el sistema controle el tiempo de respuesta a las preguntas.

Estimación inicial: 8 puntos (complejidad alta debido a la integración del tiempo real).

Respuesta a preguntas:

Descripción: Como jugador, quiero que el sistema valide mis respuestas y me muestre el resultado.

Estimación inicial: 5 puntos (complejidad media).

Consultas de historial y ranking:

Descripción: Como jugador, quiero poder consultar mi historial de partidas y mi posición en el ranking.

Estimación inicial: 8 puntos (requiere acceso a la base de datos).

Estimación total en puntos de historia:

Si sumamos los puntos de historia iniciales para las historias de usuario:

3 + 3 + 5 + 8 + 5 + 8 = 32 puntos de historia

1.10 Análisis - Diagrama de Clases

Cardinalidad

Jugador - Sistema de Autenticación:

1 a 1: Cada Jugador se autentica mediante el Sistema de Autenticación.

Jugador - Partida:

1 a N: Un Jugador puede participar en varias Partidas, y una Partida puede tener varios Jugadores.

Partida - Pregunta:

1 a N: Una Partida contiene muchas Preguntas, pero cada Pregunta pertenece a una única Partida.

Partida - Respuesta:

1 a N: Una Partida genera muchas Respuestas por parte de los jugadores, y cada Respuesta pertenece a una única Partida.

Pregunta - Respuesta:

1 a N: Cada Pregunta puede tener múltiples Respuestas de diferentes jugadores, pero cada Respuesta pertenece a una única Pregunta.

Partida - Multijugador:

1 a 1: Cada Partida multijugador está gestionada por un único sistema de Multijugador.

Partida - Cronómetro:

1 a 1: Cada Partida tiene un único Cronómetro que gestiona el tiempo de respuesta.

Jugador - Ranking:

1 a N: Un Ranking puede tener muchos Jugadores, y cada Jugador puede aparecer en varios rankings.

Partida - Configuración del Juego:

1 a 1: Cada Partida tiene una única Configuración del Juego.

API Manager - Pregunta:

1 a N: El API Manager genera muchas Preguntas a partir de la API, pero cada Pregunta es generada por una única instancia del API Manager.

Diagrama de clases