DuocUC® | ESCUELA DE INFORMÁTICA Y TELECOMUNICACIONES

Brain Boost
Unlock Your Mind's Power

Brain Boost
Capstone

Integrantes: **Macarena Bertero**
Ignacio León
Claudio Valencia

Índice

Contexto

El presente documento describe la solución técnica y las especificaciones de un proyecto de desarrollo diseñado para mejorar el proceso de estudio de los alumnos del Duoc UC. Actualmente, el aprendizaje de contenidos teóricos en la educación superior a menudo se ve limitado por métodos de estudio pasivos y poco atractivos, como la simple lectura de textos o la memorización. Esta realidad genera una baja motivación en los estudiantes y dificulta la retención efectiva del conocimiento.

La necesidad que se identificó es transformar este proceso de aprendizaje en una experiencia interactiva y lúdica que no solo capte la atención del estudiante, sino que también adapte el contenido a sus necesidades individuales, fomentando un estudio más eficiente y dirigido. La solución propuesta busca resolver este problema aplicando tecnologías de vanguardia para convertir un proceso tradicional en una experiencia de aprendizaje gamificada y personalizada.

Solución Propuesta y/o Factor Diferenciador

La solución propuesta se diferencia de las herramientas de estudio tradicionales al integrar un sistema de gamificación adaptativa impulsado por inteligencia artificial. A diferencia de las plataformas de cuestionarios genéricas, este proyecto utiliza la API de Google Gemini para analizar el rendimiento del usuario y generar preguntas y desafíos dinámicos que se enfocan en las áreas de debilidad del estudiante.

El factor diferenciador clave de nuestra aplicación es la personalización inteligente del aprendizaje. El sistema registra cada error y acierto del estudiante, creando un perfil de sus fortalezas y debilidades. Con base en este análisis, la inteligencia artificial no solo genera contenido nuevo, sino que también prioriza el refuerzo de los temas donde el usuario comete más errores, convirtiendo las deficiencias en oportunidades de mejora. Esto asegura que el tiempo de estudio sea eficiente y dirigido, transformando la experiencia de aprendizaje en un proceso continuo de adaptación y crecimiento, lo cual cumple con los criterios solicitados y ofrece una solución innovadora y efectiva a la problemática planteada

Objetivos del Proyecto

1. Objetivo General

Desarrollar una aplicación de aprendizaje adaptativo, basada en gamificación e inteligencia artificial, para los estudiantes del Duoc UC, con el fin de mejorar su retención de conocimiento en materias teóricas.

2. Objetivos Específicos

- Gestión de Usuarios: Implementar un sistema robusto de gestión de usuarios con tres roles diferenciados (Estudiante, Profesor, Administrador), asegurando su autenticación segura y la gestión de permisos.
- Integración de Contenido Inteligente: Diseñar y construir un microservicio (IA Service) que se integre con la API de Google Gemini para generar preguntas dinámicas y personalizadas, analizando el rendimiento del estudiante para identificar áreas de debilidad.
- Implementación de Gamificación: Desarrollar un módulo de juego que incorpore mecánicas como puntajes, rankings y desafíos, incentivando la participación activa del estudiante y proporcionando un entorno de aprendizaje lúdico y motivador.
- Construcción de una Arquitectura Escalable: Implementar una arquitectura de microservicios sobre Google Cloud Platform (GCP) con una base de datos relacional (PostgreSQL) y una no relacional, garantizando la escalabilidad y el mantenimiento a largo plazo de la solución.

Alcances del Proyecto

1. Funcionalidades Incluidas

- Gestión de Perfiles por Rol: La aplicación incluirá tres tipos de roles de usuario: Estudiante, Profesor, y Administrador. Cada rol tendrá un conjunto de permisos y funcionalidades específicas.
- Módulo de Estudiante: Este perfil tendrá acceso exclusivo al módulo de juego. Los estudiantes podrán seleccionar materias precargadas según su carrera, jugar, y

visualizar su progreso. Podrán editar aspectos básicos de su perfil, como su foto o ícono.

- Módulo de Profesor: Los profesores tendrán acceso a un módulo de visualización en el cual podrán ver los rankings y el progreso de los estudiantes de sus cursos. Además, tendrán la capacidad de cargar contenido teórico (documentos, extractos, etc.) por materia. Este contenido es la fuente que posteriormente la Inteligencia Artificial (IA) analizará para generar preguntas dinámicas. Adicionalmente, el profesor podrá crear, editar y eliminar preguntas de forma manual en el banco de contenido teórico de las materias que tenga asignadas. Finalmente, tendrán la capacidad de cargar masivamente alumnos a las asignaturas que tengan a su cargo, facilitando la inscripción y la administración de grandes grupos de alumnos.
- Módulo de Administrador: Este rol tendrá los privilegios más altos, con acceso a todas las funcionalidades del sistema. Podrá gestionar usuarios, contenido, y configuraciones generales de la aplicación.
- Módulo de Gamificación: Implementación de mecánicas de juego como niveles, desafíos, puntajes, y un sistema de logros para mantener a los estudiantes motivados con el aprendizaje. El sistema registrará los errores de los usuarios, lo que permitirá generar cuestionarios de refuerzo adaptativos, enfocados en las áreas de debilidad detectadas.
- Banco de Contenido Teórico: La aplicación contendrá un banco de preguntas y respuestas relacionadas con las materias teóricas de la carrera, organizadas de manera jerárquica: una carrera tiene muchas asignaturas, y cada asignatura contiene muchas preguntas. Esto permite un manejo y una categorización del contenido clara.
- Integración Inteligente con Google Gemini: Se desarrollará un microservicio dedicado a la comunicación con la API de Gemini. Esta integración no solo generará preguntas dinámicas y personalizadas, sino que también analizará el historial de respuestas del usuario para identificar áreas de debilidad. La inteligencia artificial fomentará el contenido en el que el usuario comete más errores, reforzando así el aprendizaje de manera dirigida y eficiente. El sistema cuenta con un módulo de inteligencia que, al recibir la solicitud de un nuevo cuestionario, procesa el historial de rendimiento del usuario. Este análisis permite identificar los temas o conceptos en los que el estudiante ha tenido más dificultades. Con esta información, el sistema genera preguntas de refuerzo, asegurando que el contenido se adapte de forma inteligente y dirigida a las necesidades de aprendizaje del usuario.Este enfoque asegura un proceso de aprendizaje adaptativo, donde la inteligencia artificial utiliza el rendimiento del estudiante para dirigir y reforzar el conocimiento de manera personalizada.

- Sistemas de Estadísticas y Analíticas: Se registrarán las interacciones y el progreso del usuario en una base de datos no relacional para futuras analíticas, permitiendo comprender mejor los patrones de aprendizaje.
- Versión web: Se incluirá el desarrollo de una aplicación web cuyo alcance estará estrictamente limitado a las funcionalidades de gestión y administración de contenido, siendo la plataforma móvil la única destinada a la experiencia de juego. El administrador utilizará la versión web para gestionar usuarios, roles y configuraciones generales del sistema. Mientras que el perfil de profesor podrá cargar contenido teórico masivo, gestionar el banco de preguntas y cargar alumnos de manera masiva a las asignaturas asignadas.

- Interfaz de Usuario (UI) y Experiencia de Usuario (UX) intuitiva: El diseño se centrará en una interfaz limpia y una navegación sencilla, asegurando una experiencia de juego fluida y accesible.

2. Funcionalidades Excluidas

- Desarrollo para iOS: El proyecto se enfocará exclusivamente en la plataforma Android utilizando lenguajes nativos. No se contempla una versión nativa para el sistema operativo iOS en esta etapa.
- Versión de Escritorio: No se desarrollarán versiones de la aplicación para programas de escritorio.
- Integración con el Sistema Académico del Duoc UC: El juego funcionará de manera independiente del sistema de registro y notas oficial de la institución.
- Modo Multijugador en Tiempo Real: Las funcionalidades multijugador se limitarán a tablas de clasificación (rankings) y desafíos asincrónicos, para mantener el foco en el aprendizaje individual.

Relación con el Perfil de Egreso y los Intereses Profesionales

Este proyecto integra y aplica directamente las competencias del perfil de egreso de la carrera, demostrando la capacidad del equipo para enfrentar un desafío tecnológico real. La

solución aborda de forma integral la creación de una propuesta informática, la cual se desarrolla utilizando la metodología ágil Scrum para sistematizar el proceso. Esto se evidencia en la construcción de un modelo arquitectónico de microservicios y un modelo de datos escalable, que en conjunto permiten implementar una solución integral que automatiza y optimiza los procesos de aprendizaje.

El proyecto también demuestra la capacidad de programar consultas y rutinas para la manipulación de información, construir programas con buenas prácticas de codificación y realizar pruebas de calidad en cada iteración. Finalmente, se abordan las vulnerabilidades sistémicas en la sección de seguridad, demostrando un enfoque profesional y completo. La participación en este proyecto se alinea directamente con nuestros intereses de forjar una carrera en el desarrollo de software, la gestión de la nube y la inteligencia artificial, aplicando la capacidad de generar ideas innovadoras y resolver problemas de forma proactiva.

1.Intereses Profesionales

La participación en este proyecto permite al equipo adquirir experiencia práctica en áreas de alta demanda en el mercado laboral, como el desarrollo de software y arquitecturas de microservicios, el despliegue en la nube con GCP, la integración de inteligencia artificial y el desarrollo de bases de datos escalables. Esto se alinea directamente con los intereses de forjar una carrera como desarrollador de software con especialización en tecnologías de backend e IA.

Requisitos del Proyecto

1. Requisitos Funcionales

El sistema debe permitir que los usuarios se autentiquen de manera segura, asignando roles con permisos específicos:

- **Requisitos del Estudiante:**
  - ➢ El estudiante debe poder acceder al juego y seleccionar su carrera y materia.

- El sistema debe analizar el historial de respuestas del estudiante para identificar los temas o conceptos en los que comete más errores.
- El juego debe ajustar dinámicamente el contenido, priorizando la presentación de preguntas y desafíos sobre los temas que el estudiante necesita reforzar. Esto asegura un aprendizaje personalizado y enfocado en sus debilidades.
- El sistema debe mostrar al estudiante su puntaje, progreso individual y las áreas específicas de mejora.
- El estudiante debe poder editar su foto o ícono de perfil.

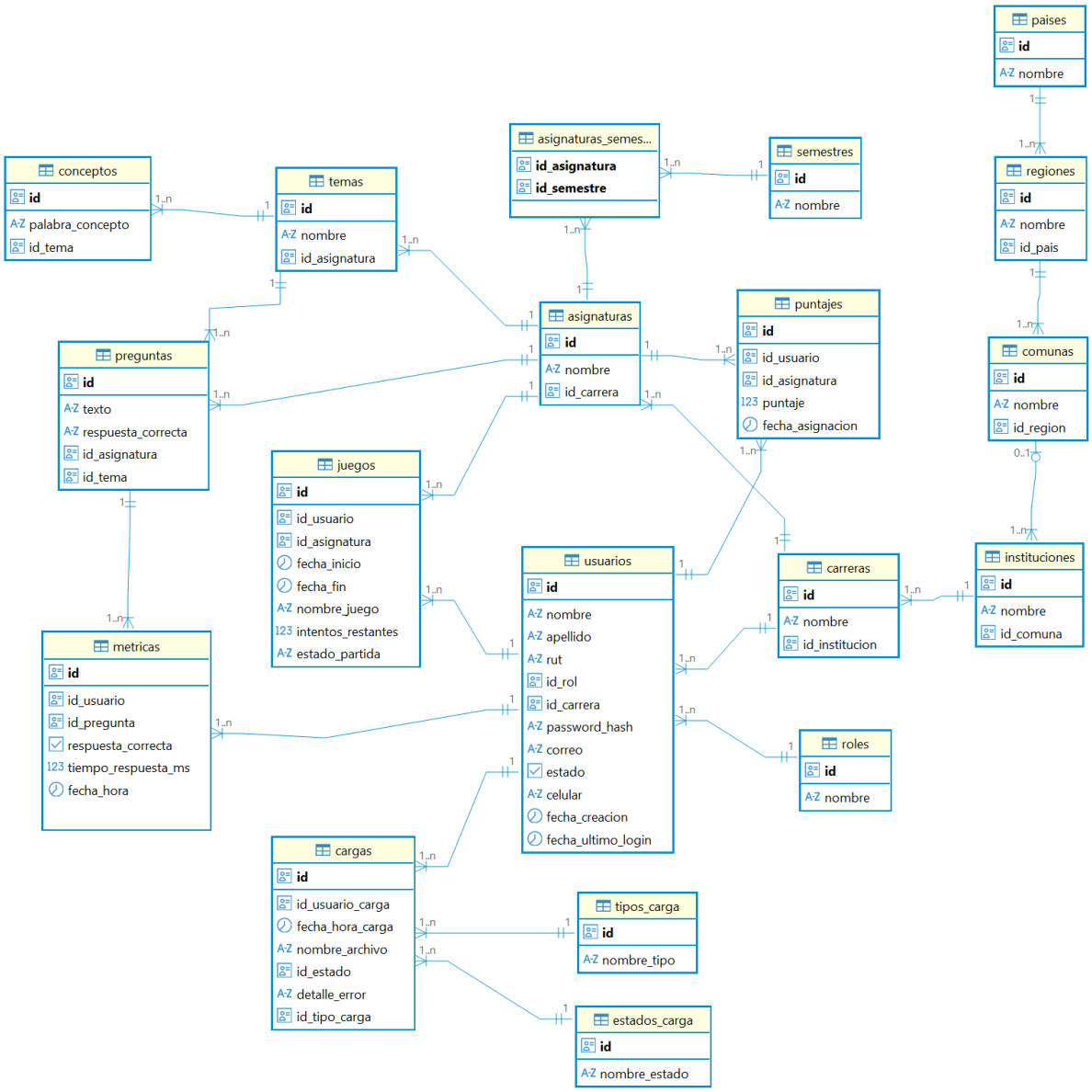
- **Requisitos del Profesor:**
  - El profesor debe tener acceso únicamente a los datos de los estudiantes y el contenido de las asignaturas que le han sido asignadas por el Administrador.
  - El sistema debe permitir al profesor ver los rankings y acceder a estadísticas de rendimiento detalladas del grupo y de los estudiantes individuales en sus cursos asignados.
  - El profesor debe tener la funcionalidad para cargar contenido teórico (ej. archivos PDF o texto) por materia. Este contenido debe ser almacenado para que el Módulo de IA lo analice posteriormente.
  - El profesor debe tener acceso a un módulo de administración para crear, editar o eliminar preguntas de forma manual en el banco de contenido teórico de las materias bajo su cargo.
  - El profesor debe tener la capacidad de cargar masivamente alumnos a las asignaturas que tenga asignadas, utilizando un archivo, lo que facilitará el proceso de inscripción
- **Requisitos del Administrador:**
  - El administrador debe tener privilegios para gestionar todos los aspectos de la aplicación, incluyendo usuarios, roles, materias y el contenido general.
- **Requisitos Generales:**
  - El sistema debe permitir al usuario iniciar una sesión de juego o desafío en cualquier momento.
  - El sistema debe presentar preguntas teóricas de forma aleatoria o temática.
  - El sistema debe validar la respuesta del usuario y proporcionar retroalimentación inmediata, utilizando la API de Google Gemini para generar respuestas dinámicas y personalizadas.
  - El sistema debe registrar y actualizar el puntaje y el progreso del usuario después de cada sesión de juego.

- ➢ El sistema debe mostrar un ranking general y por curso de los puntajes más altos entre los usuarios.

2. Requisitos No Funcionales

- Rendimiento: El tiempo de respuesta de la aplicación para las interacciones del usuario no debe exceder los 2 segundos, y el tiempo de carga de las preguntas de Gemini no debe superar los 5 segundos.
- Seguridad: Las contraseñas de los usuarios deben ser encriptadas. La comunicación con los microservicios debe realizarse a través de canales seguros (HTTPS).
- Usabilidad: La interfaz de usuario debe ser intuitiva y fácil de usar para un público estudiantil.
- Escalabilidad: La arquitectura de microservicios debe permitir un crecimiento futuro en el número de usuarios y la adición de nuevos módulos sin comprometer el rendimiento general.%
- Disponibilidad: El sistema debe tener una disponibilidad del 99% para garantizar que los usuarios puedan estudiar en cualquier momento.

Diagrama de Base de Datos



[capstonebbdd-v3.png]

1.Diccionario de Datos

1.1. Tablas de Ubicación e Institución

Estas tablas definen las entidades geográficas y organizacionales necesarias para el contexto educativo.

- **paises**: Almacena los países. Su objetivo es proporcionar una referencia de ubicación global para las instituciones.
    - ➢ id: Identificador único del país.
    - ➢ nombre: Nombre del país.

- **regiones**: Contiene las regiones de cada país. Ayuda a organizar la ubicación de manera jerárquica.
    - ➢ id: Identificador único de la región.
    - ➢ nombre: Nombre de la región.
    - ➢ id_pais: Clave foránea que la relaciona con la tabla paises.
- **comunas**: Almacena las comunas de cada región. Proporciona el nivel más específico de ubicación.
    - ➢ **i**d: Identificador único de la comuna.
    - ➢ nombre: Nombre de la comuna.
    - ➢ id_region: Clave foránea que la relaciona con la tabla regiones.
- **instituciones**: Almacena las instituciones educativas como Duoc UC.
    - ➢ id: Identificador único de la institución.
    - ➢ nombre: Nombre de la institución.
    - ➢ id_comuna: Clave foránea que la relaciona con la tabla comunas.

1.2. Tablas de Contenido Académico

Estas tablas definen la estructura curricular y los roles de los usuarios dentro del sistema.

- **roles**: Contiene los roles de usuario (Estudiante, Profesor, Administrador). Es fundamental para la gestión de permisos en la aplicación.
    - ➢ id: Identificador único del rol.
    - ➢ nombre_rol: Nombre del rol.
- **carreras**: Almacena las carreras que ofrece la institución.
    - ➢ id: Identificador único de la carrera.
    - ➢ nombre: Nombre de la carrera.
    - ➢ id_institucion: Clave foránea que la relaciona con la tabla instituciones.
- **semestres**: Almacena los semestres académicos.
    - ➢ id: Identificador único del semestre.
    - ➢ nombre: Nombre del semestre (ej. "Semestre 1", "Semestre de Verano").
- **asignaturas**: Contiene las materias académicas. Cada asignatura pertenece a una carrera.
    - ➢ id: Identificador único de la asignatura.
    - ➢ nombre: Nombre de la asignatura.
    - ➢ id_carrera: Clave foránea que la relaciona con la tabla carreras.

- **asignaturas_semestre**: Esta es una tabla intermedia que resuelve la relación de muchos a muchos (N:M) entre asignaturas y semestres, permitiendo que una asignatura se imparta en varios semestres y viceversa.
    - ➢ id_asignatura: Clave foránea que la relaciona con la tabla asignaturas.
    - ➢ id_semestre: Clave foránea que la relaciona con la tabla semestres.
- **temas:** Esta tabla centraliza la categorización temática del contenido para los juegos.
    - ➢ id: Identificador único del tema.
    - ➢ nombre: Nombre del tema
    - ➢ id_asignatura: Clave foranea que la relaciona con la tabla asignaturas.

1.3. Tablas de Usuarios y Gamificación

Esta es la sección maneja la interacción del usuario y el registro del juego.

- **usuarios**: Almacena la información de los usuarios. Se conecta directamente a carreras para indicar a qué carrera pertenece cada usuario.
    - ➢ id: Identificador único del usuario.
    - ➢ nombre: Nombre del usuario.
    - ➢ apellido: Apellido del usuario.
    - ➢ rut: Rut del usuario.
    - ➢ id_rol: Clave foránea que la relaciona con la tabla roles.
    - ➢ id_carrera: Clave foránea que la relaciona con la tabla carreras.
    - ➢ password_hash: Contraseña encriptada para la autenticación.
    - ➢ correo: Correo electrónico, usado como identificador único.
    - ➢ estado: Indica si la cuenta está activa o no.
    - ➢ celular: Número de celular del usuario.
    - ➢ fecha_creacion: Fecha en que se creó la cuenta.
    - ➢ fecha_ultimo_login: Fecha del último acceso del usuario.
- **preguntas**: El banco de contenido teórico.
    - ➢ id: Identificador único de la pregunta.
    - ➢ texto: El enunciado de la pregunta.
    - ➢ respuesta_correcta: La respuesta correcta de la pregunta.
    - ➢ tema: Tema de la pregunta (ej. "JOINs" en SQL).
    - ➢ id_asignatura: Clave foránea que la relaciona con la tabla asignaturas.
- **conceptos:** Almacena las palabras o frases a adivinar para los juegos.
    - ➢ id: Identificador unico del concepto.
    - ➢ palabra_concepto: Palabra que contiene un concepto.

- ➢ id_tema: Clave foranea que lo relaciona con la tabla tema.

- **juegos**: Registra cada sesión de juego. Es el historial de partidas del usuario.
  - ➢ id: Identificador único del juego/sesión.
  - ➢ id_usuario: Clave foránea que la relaciona con la tabla usuarios.
  - ➢ id_asignatura: Clave foránea que la relaciona con la tabla asignaturas.
  - ➢ nombre_juego: Nombre del juego
  - ➢ fecha_inicio: Momento en que inició el juego.
  - ➢ fecha_fin: Momento en que terminó el juego.
  - ➢ intentos_restantes: Número de intentos.
  - ➢ estado_partida: Estado de la partida (En curso, Ganado, Perdido)
- **metricas**: Registra cada respuesta individual para analizar el rendimiento.
  - ➢ id: Identificador único de la métrica.
  - ➢ id_usuario: Clave foránea que la relaciona con la tabla usuarios.
  - ➢ id_pregunta: Clave foránea que la relaciona con la tabla preguntas.
  - ➢ respuesta_correcta: Valor booleano (TRUE/FALSE) que indica si la respuesta fue correcta.
  - ➢ tiempo_respuesta_ms: Tiempo que el usuario tardó en responder.
  - ➢ fecha_hora: Momento en que se registró la respuesta.
- **puntajes**: Almacena el puntaje acumulado de un usuario en una asignatura específica.
  - ➢ id: Identificador único del puntaje.
  - ➢ id_usuario: Clave foránea que la relaciona con la tabla usuarios.
  - ➢ id_asignatura: Clave foránea que la relaciona con la tabla asignaturas.
  - ➢ puntaje: Puntaje total acumulado.
  - ➢ fecha_asignacion: Fecha en que se registró o actualizó el puntaje.

1.4. Tablas de Carga y Auditoría

- **estados_carga**: Contiene los estados de una carga masiva (ej. "Completado", "Pendiente", "Fallido").
  - ➢ id: Identificador único del estado.
  - ➢ nombre_estado: Nombre del estado de la carga.
- **tipos_carga**: Almacena los tipos de cargas (ej. "Carga de Usuarios", "Carga de Contenido").
  - ➢ id: Identificador único del tipo.

- nombre_tipo: Nombre del tipo de carga.

- **cargas**: Registra el historial de todas las cargas masivas.
  - id: Identificador único de la carga.
  - id_usuario_carga: Clave foránea que la relaciona con la tabla usuarios.
  - fecha_hora_carga: Fecha y hora de la carga.
  - nombre_archivo: Nombre del archivo original.
  - id_estado: Clave foránea que la relaciona con la tabla estados_carga.
  - detalle_error: Detalles del error en caso de que la carga falle.
  - id_tipo_carga: Clave foránea que la relaciona con la tabla tipos_carga.

Decisiones Clave y Ajustes al Diseño

A lo largo de la fase de planificación y desarrollo inicial, el equipo realizó ajustes críticos al diseño original para optimizar la arquitectura, mejorar el rendimiento y garantizar la factibilidad del proyecto

2.1. Evolución de la Arquitectura de Microservicios

Para asegurar una gestión clara de responsabilidades y preparar el terreno para la lógica compleja, se segmentaron los roles del backend de la siguiente manera:

Ajuste Realizado	Impacto en la Arquitectura	Justificación
Inclusión del Game Engine	Se añadió un servicio CORE dedicado.	**Facilitador (Modularidad):** Centraliza la gestión del ciclo de vida de la partida y la lógica específica de los tipos de juego (Quizz/Ahorcado), liberando

		al BFF de la orquestación transaccional.
Inclusión del Scoring Service	Se añadió un servicio CORE dedicado.	**Dificultad (Rendimiento)**: Aísla el cálculo del puntaje, el progreso y el ranking. Esto permite optimizar el rendimiento de la lógica numérica y facilita la implementación de pruebas unitarias detalladas.
Modelo Único de IA (Gemini 2.5 Flash)	Eliminación de la estrategia de doble modelo (Flash y Pro).	**Dificultad (Riesgo de Cuota):** Decisión basada en la mejor relación cuota/rendimiento y la compatibilidad técnica (facilitador) con el SDK actual, asegurando la continuidad del servicio en el Nivel Gratuito.

2.2. Ajustes al Modelo de Datos

El modelo de datos evolucionó para soportar una lógica de contenido más flexible y escalable, eliminando redundancias.

Ajuste Realizado	Tablas Involucradas	Justificación
Creación de Tablas de Contenido	+ temas y + conceptos	**Facilitador (Gestión de Contenido):** La tabla temas centraliza la categorización, permitiendo que preguntas y conceptos utilicen la misma estructura temática, lo cual simplifica la búsqueda y la gestión.
Eliminación de Redundancia de Ranking	- ranking	**Dificultad (Consistencia):** La tabla ranking fue eliminada. El ranking es ahora un cálculo dinámico del Scoring Service (facilitador), eliminando el riesgo de inconsistencia entre la tabla de puntajes y

		una tabla materializada de ranking.

Metodología

Para la gestión y desarrollo de este proyecto, se ha seleccionado la metodología ágil Scrum. Esta elección se fundamenta en su capacidad para gestionar proyectos complejos de manera eficiente y flexible, lo que es ideal para un proyecto de desarrollo de software.

Las principales razones para la elección de Scrum son:

- Adaptabilidad: Permite reaccionar de manera ágil a los cambios en los requisitos del proyecto o a los hallazgos inesperados durante el desarrollo, asegurando que el producto final siempre cumpla con las necesidades del cliente.
- Entrega de Valor Temprana y Continua: El trabajo se divide en "sprints" cortos, lo que permite entregar funcionalidades operativas de forma incremental. Esto nos permitirá mostrar progreso de manera constante y obtener retroalimentación temprana del profesor.
- Gestión del Riesgo: Al abordar el desarrollo en iteraciones cortas, los riesgos se identifican y gestionan de forma proactiva, evitando que se conviertan en problemas mayores.
- Transparencia: Con la planificación y revisión de cada sprint, todos los involucrados en el proyecto tienen una visibilidad completa del estado del proyecto y de los avances logrados.

Plan de Trabajo

El cronograma de trabajo para las siguientes etapas se detalla a continuación.

Cada sprint tendrá una duración de dos semanas, lo que nos permitirá un desarrollo iterativo e incremental. La siguiente planificación está alineada con la Carta Gantt del proyecto.

Gestión y Priorización de Tareas

Para la gestión de las tareas y el seguimiento del progreso, el equipo utiliza Jira.

- Historias de Usuario y Priorización: Todas las funcionalidades se han descompuesto en Historias de Usuario, las cuales son registradas en Jira. El equipo se reúne semanalmente para revisar el Backlog y priorizar las historias.
- Asignación al Sprint: Las tareas y las historias de usuario se asignan al Sprint según su prioridad técnica y su impacto en el valor del producto, garantizando que el trabajo entregado sea siempre el más relevante.
- Reuniones Diarias (Daily Scrum): El equipo realiza reuniones cortas y frecuentes para sincronizar el avance, identificar impedimentos y ajustar el foco del trabajo del día.

Sprint 1: Pre-desarrollo (Del 16 al 30 de agosto)

- Objetivo: Establecer las bases teóricas y de diseño del proyecto para asegurar un inicio de desarrollo sólido y bien planificado.
- Actividades Clave:
    - ➢ Definición de los alcances del proyecto.
    - ➢ Identificación de los requisitos funcionales y no funcionales.
    - ➢ Identificación de la problemática y justificación de la relevancia del proyecto.
    - ➢ Elección y justificación de la metodología de trabajo (Scrum).
    - ➢ Análisis de Modelo de IA a utilizar.
    - ➢ Preparación de los ambientes de desarrollo y de la infraestructura en la nube.
    - ➢ Diseño del modelo de la arquitectura.
    - ➢ Elaboración del diagrama de base de datos.
    - ➢ Selección y validación de las tecnologías a utilizar.

- Facilitador: El *expertise* del equipo en Java y Spring Boot nos permite diseñar una arquitectura de microservicios con convenciones estandarizadas desde el inicio.
- Obstaculizador: Ambigüedad o alcance no definido de la funcionalidad de la IA, lo cual podría forzar rediseños en la arquitectura

- Mitigación: Priorizar la Prueba de Concepto del IA Service en el diseño. Se definirá un contrato *mock* de API entre el IA Service y el Content Service para que el desarrollo del *backend* pueda avanzar.

Sprint 2: Base Tecnológica (Del 31 de agosto al 14 de septiembre)

- Objetivo: Configurar el entorno de desarrollo e implementar los servicios de autenticación y gestión de usuarios.
- Backlog del Sprint:
    - ➢ Configuración de la VPC y PostgreSQL en GCP.
    - ➢ Implementación del Esquema de Persistencia (Tablas Usuarios/Roles).
    - ➢ Desarrollo y Documentación de Contratos API (Auth Service).
    - ➢ Implementación de la lógica de seguridad (Hashing y Módulo JWT).
    - ➢ Desarrollo de la Vista de Login y Registro (Frontend).
    - ➢ Integración y Pruebas E2E del flujo de Login/JWT.

- Facilitador: Dominio de la Nube, experiencia previa del equipo en la configuración de la infraestructura en GCP y la gestión de la base de datos PostgreSQL.
- Obstaculizador: Problemas de conectividad segura entre los microservicios (a través del BFF) y la instancia de PostgreSQL, exponiendo datos críticos.
- Mitigación: Configuración estricta de Firewall Rules de GCP. Se asegurará que la conexión a PostgreSQL sea solo interna, deshabilitando cualquier acceso público directo.

Sprint 3: Integración con IA (Del 15 al 29 de septiembre)

- Objetivo: Desarrollar el servicio clave para la integración con la IA y la generación de contenido dinámico.
- Backlog del Sprint:
    - ➢ Implementación de la integración funcional con la API de Google Gemini (IA Service).

- ➢ Desarrollo del endpoint de carga de documentos (PDF/Texto) para el Content Service.
- ➢ Implementación de la lógica de generación de preguntas a partir de documentos subidos.
- ➢ Implementación de la funcionalidad de carga masiva de usuarios por archivo (Excel).
- ➢ Desarrollo y testing del mecanismo de fallback (cuota agotada) y caché

- Facilitador: El IA Service es un microservicio independiente, permitiendo que el desarrollo del Auth y Content Service avance en paralelo sin dependencias críticas.
- Obstaculizador: Riesgo de Interrupción por Cuotas: Exceder los límites de la versión gratuita de la API de Google Gemini, lo que resultaría en una interrupción total del servicio de preguntas dinámicas y afectaría la experiencia de juego. Además, la latencia de la versión gratuita podría ser inconsistente.
- Mitigación:

  1. Implementar un mecanismo de *fallback* en el IA Service que, al fallar la conexión con Gemini (código de error 429 - Cuota Excedida), entregue una pregunta del banco de contenido estático.

  2. Utilizar mecanismos de caché para las preguntas generadas por la IA, reduciendo la necesidad de llamar a la API en partidas consecutivas.

Sprint 4: Lógica de Gamificación (Del 30 de septiembre al 13 de octubre)

- Objetivo**:** Implementar el motor de gamificación y la lógica de las partidas.
- Backlog del Sprint:
  - ➢ Elaboración del Documento de Diseño de Lógica de Juego y Gamificación
  - ➢ Refactorización del IA Service.
  - ➢ Refactorización del Content Service (Flujo de Carga Masiva).
  - ➢ Contenerización (Dockerización) de Servicios Core.
  - ➢ Despliegue de Staging en GCP.
  - ➢ Testing de servicios.

- ➢ Revisión de Código Cruzada

- Facilitador: La lógica se desarrolla en Spring Boot, que ofrece frameworks robustos para la implementación de lógica de negocio (puntajes, logros y rankings).
- Obstaculizador: Errores en las fórmulas de cálculo del progreso adaptativo o en la jerarquía del *ranking*, afectando la experiencia de juego y la credibilidad del sistema.
- Mitigación: Pruebas Unitarias exhaustivas y detalladas en la capa de servicio para validar todas las reglas de negocio. Se realizarán revisiones de código cruzadas.

Sprint 5: Conexión Frontend y Estructura del Game Engine(Del 14 al 27 de octubre)

- Objetivo: Priorizar la entrega del flujo crítico de Autenticación y Carga de Contenido, y dejando el Game Engine estructurado para el desarrollo de la lógica compleja.
- Backlog del Sprint:
  - ➢ Desarrollo de la interfaz de usuario de la pantalla de inicio (rol Profesor) y la vista para subir archivos de contenido en la aplicación móvil.
  - ➢ Conexión del *frontend* móvil para ejecutar el flujo completo: Login → Obtener JWT → Carga de Contenido.
  - ➢ Desarrollo de los *endpoints* iniciales del Game Engine y la estructura de código (controller, service, repository) lista para el desarrollo.
  - ➢ Adición de nuevas entidades y actualización del esquema de persistencia para soportar la lógica de juego.

- Facilitador: La refactorización y el diseño de la lógica de juego del Sprint 4 proporcionan un diseño sólido. La prioridad en el *frontend* del flujo crítico asegura un entregable funcional y visible.
- Obstaculizador: Problemas de rendimiento en el orquestador debido a la agregación de múltiples llamadas de microservicios, afectando el tiempo de respuesta final al frontend.
- Mitigación:Implementación de Pruebas de Carga para identificar cuellos de botella. Uso de patrones de Programación Reactiva (si el *framework* lo permite) para manejar las llamadas concurrentes de forma eficiente.

Sprint 6: Lógica de Juego, Scoring y Vistas de Partida (Del 28 de octubre al 10 de noviembre)

- Objetivo: Implementar la lógica completa de juego, separando las responsabilidades de flujo (*Game Engine*) y cálculo (*Scoring*), y completar la vista de la experiencia de juego en el *frontend*.
- Backlog del Sprint:
    - ➢ Implementación de la lógica del juego (flujo), y codificación de los módulos específicos para las estructuras de los juegos.
    - ➢ Desarrollo de las funciones que calculan el puntaje bruto de una pregunta y la lógica de seguimiento de progreso/ranking.
    - ➢ Desarrollo de los *endpoints* para iniciar una partida y procesar una respuesta (conexión del Game Engine con Content).
    - ➢ Desarrollo de los *endpoints* del Game Engine para mostrar el ranking global y puntaje personal.
    - ➢ Implementación de la vista de preguntas, la lógica de respuesta y la visualización de *feedback* post-respuesta en el *frontend* móvil.
    - ➢ Pruebas funcionales del flujo de juego completo: Móvil → BFF → Game Engine → IA Service.

- Facilitador: La lógica compleja del *Game Engine* tiene su propio servicio dedicado, mejorando la modularidad. La estructura del Sprint 5 asegura que la conexión ya esté estable.
- Obstaculizador: Fallos críticos en las fórmulas de cálculo del Game Engine o en la persistencia del ranking, afectando la experiencia de juego central.
- Mitigación: Implementación de pruebas unitarias exhaustivas y detalladas en la capa de servicio de Game Engine/Scoring para validar todas las reglas de negocio de puntaje antes de la integración al *frontend*.

Sprint 7: Perfiles, Calidad, Cierre y Entrega Final (Del 11 al 22 de noviembre)

- Objetivo: Completar las vistas finales de usuario, ejecutar pruebas de calidad (UAT), corregir *bugs* y formalizar la documentación y la presentación final del proyecto.
- Backlog del Sprint:

- ➢ Desarrollo de la interfaz de usuario para perfil de usuario (edición de icono y contraseña).
- ➢ Pruebas de Calidad y Usabilidad (UAT).
- ➢ Pruebas de Estrés y Seguridad
- ➢ Corrección de Errores y Bugs
- ➢ Redacción del Informe Final, Preparación de la Presentación Final y Demo.

.

- Facilitador: Los Sprints anteriores incluyeron la creación de Pruebas Unitarias, simplificando la etapa de UAT.
- Obstaculizador: Descubrimiento de vulnerabilidades de seguridad o fallos en el flujo de conexión de Game Engine que requieran un *hotfix* de última hora.
- Mitigación: Realización de varios simulacros de la presentación para asegurar la fluidez y estabilidad del sistema en el entorno de GCP antes de la entrega final.

Gantt - Capstone

Factibilidad del Proyecto

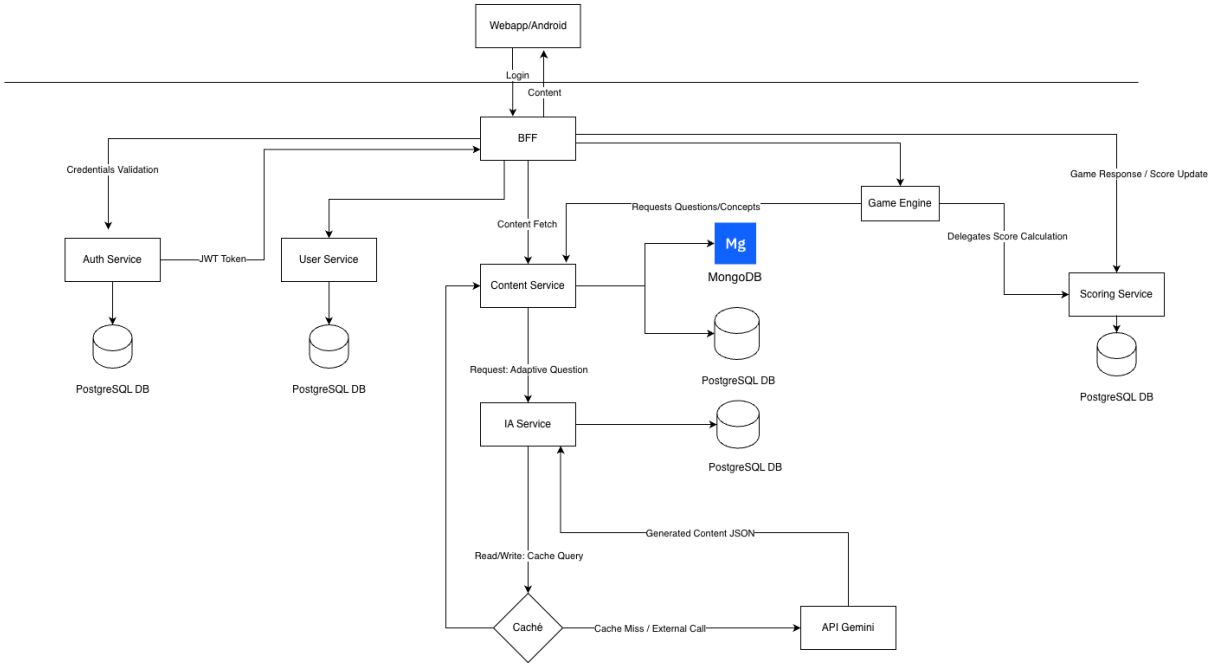
Este proyecto es factible de ser realizado, ya que se han tomado en cuenta los siguientes factores para garantizar su correcta ejecución.

- Alcance y Tiempo: El alcance del proyecto ha sido definido de manera precisa para ser completado en el plazo establecido. El plan de trabajo con la metodología Scrum, con sprints de dos semanas, nos permite monitorear el avance de manera constante y ajustar el plan si fuera necesario. Hemos superado exitosamente la fase de Planificación y Diseño (Sprint 1) y hemos completado la Base Tecnológica y la Integración de Servicios (Sprints 2, 3 y 4). Este avance concreto válida la aplicabilidad de nuestro plan de trabajo y reduce significativamente el riesgo de incumplimiento de plazos.
- Recursos y Herramientas: La factibilidad del proyecto se sustenta en la capacidad técnica especializada del equipo. Poseemos conocimiento y experiencia previa en Java y Spring Boot. Esto garantiza que el *backend* será desarrollado con eficiencia, escalabilidad y siguiendo las mejores prácticas. Esta habilidad, junto con la elección de tecnologías *open source* (Kotlin y PostgreSQL) y la utilización de la plataforma de

nube Google Cloud Platform (GCP), asegura el acceso a todas las herramientas necesarias para el desarrollo y despliegue del proyecto sin incurrir en barreras de costos significativas.

- Viabilidad de Mercado: La propuesta de valor de este proyecto es altamente atractiva y factible, ya que no se han encontrado en el mercado local soluciones similares que integren de forma efectiva la gamificación con la inteligencia artificial para personalizar el aprendizaje. Esto representa una oportunidad única para desarrollar un producto innovador que satisface una necesidad real en el ámbito educativo.
- Manejo de Riesgos: La metodología Scrum nos permite identificar y mitigar riesgos de manera temprana. Por ejemplo, la integración con la API de Gemini, uno de los puntos más complejos, se ha planificado para el Sprint 3, lo que nos da suficiente tiempo para abordar cualquier dificultad técnica. Además, la modularidad de los microservicios asegura que un problema en un componente no detenga el desarrollo completo del sistema.

Arquitectura



arquitectura-v1.3.5.png

El proyecto se desarrollará bajo una arquitectura de Microservicios, lo cual garantiza la escalabilidad horizontal y la modularidad de cada componente. El sistema se desplegará en

la nube de Google Cloud Platform (GCP), utilizando el patrón Backend for Frontend (BFF) para optimizar el rendimiento de la aplicación móvil y la interfaz web.

El BFF actúa como el único punto de entrada para todos los clientes (aplicación móvil y versión web). Su función principal no es solo enrutar peticiones, sino también orquestar y consolidar la información proveniente de múltiples microservicios en una única respuesta eficiente. Este componente es esencial para la seguridad, ya que actúa como un perímetro que protege las claves privadas del sistema de ser expuestas al cliente final.

1. Persistencia y Almacenamiento de Datos

El sistema utiliza dos tipos de almacenamiento, cada uno adaptado a un propósito específico para optimizar la carga de trabajo y el rendimiento:

- Base de Datos Transaccional (PostgreSQL): Esta es la base de datos principal y central del sistema. Se utiliza para almacenar datos críticos que requieren alta integridad referencial, incluyendo:
  - ➢ Información de usuarios y la asignación de roles.
  - ➢ Estructura definitiva del banco de contenido académico (carreras, asignaturas, temas y las preguntas que el profesor gestiona mediante el CRUD).
  - ➢ Registros de la lógica de gamificación (juegos, métricas y puntajes).

- Base de Datos No Relacional (MongoDB): Este almacenamiento se destina exclusivamente a la ingesta y el *staging* de datos voluminosos. Se utiliza solamente para recibir y almacenar el contenido teórico (archivos o texto) que el profesor carga de manera masiva. Una vez que este contenido es procesado por el IA Service y transformado en preguntas estructuradas, el contenido original puede ser gestionado, liberando esta base de datos de tareas transaccionales.

2. Microservicios Principales

La lógica de negocio se divide en los siguientes microservicios independientes:

2.1 Auth Service

Es la autoridad de identidad del sistema. Se encarga de procesar el registro y el inicio de sesión, validando credenciales y generando el JSON Web Token (JWT), que se usa para la autorización del usuario.

Endpoint	Método HTTP	Propósito
/api/auth/login	POST	Autentica al usuario mediante credenciales. Si el acceso es exitoso, retorna un JWT que debe ser usado en todas las peticiones posteriores.

2.2 User Service

El User Service gestiona el ciclo de vida de las cuentas de usuario, incluyendo la creación de nuevos perfiles, la asignación de roles y la gestión de la carga masiva.

Endpoint	Método HTTP	Propósito
/api/auth/register	POST	Crea una nueva cuenta de usuario en el sistema. Genera un *hash* seguro para la contraseña antes de persistir los datos en PostgreSQL.
/api/users/upload	POST	Permite al Profesor (rol validado) cargar un archivo (ej., CSV/Excel) con los datos de múltiples alumnos. El servicio procesa el archivo, aplica *hashing* a las contraseñas temporales y crea las cuentas en masa.

2.3 Content Service

Maneja la fuente del contenido académico. Es el único servicio con acceso a ambas bases de datos:

- Se conecta a MongoDB para leer el contenido en bruto que ha subido el profesor.
- Se conecta a PostgreSQL para la gestión CRUD del banco de preguntas estructuradas (preguntas y conceptos).
- Se comunica con el IA Service para solicitar contenido adaptativo o la transformación de datos brutos.

2.4 Game Engine

Es el orquestador de la experiencia de juego y el responsable de la lógica transaccional de la partida. Recibe las peticiones de inicio de juego del BFF, interactúa con el Content Service para obtener el pool de preguntas/conceptos necesarios (según el tipo de juego: Quizz o Ahorcado) y gestiona el estado en la tabla juegos (Ej: intentos_restantes, estado_partida). Registra cada respuesta o acción del usuario en la tabla metricas y al finalizar la partida, envía el resultado detallado al Scoring Service para el cálculo del puntaje.

2.5 IA Service (Servicio de Inteligencia Artificial)

Actúa como un *proxy* seguro y es el motor de la personalización:

- Recibe la solicitud Request: Adaptive Question desde el Content Service.
- Gestiona el recurso mediante Read/Write: Cache Query, consultando primero el Caché para mitigar la latencia y los costos de la API externa.
- Analiza el historial de errores del usuario para solicitar preguntas dinámicas a la API de Google Gemini y reforzar el aprendizaje.

Endpoint	Método HTTP	Propósito
/api/ai/pdf/resumen	POST	Recibe un archivo adjunto (PDF/Texto) y utiliza el modelo de IA para procesar

		el contenido y generar un título y un resumen conciso.
/api/ai/pdf/quiz?numQuestions=5	POST	Recibe un archivo adjunto y genera un número específico de preguntas (numQuestions) junto con sus opciones de respuesta, transformando el texto en un cuestionario estructurado.

2.6 Scoring Service:

Es el motor de cálculo de la lógica de gamificación. Recibe las interacciones de los jugadores desde el Game Service, calcula los puntajes (puntajes) y el progreso, y es el responsable de calcular el ranking en tiempo real (no persiste una tabla ranking física)

Avance del Desarrollo e Implementación

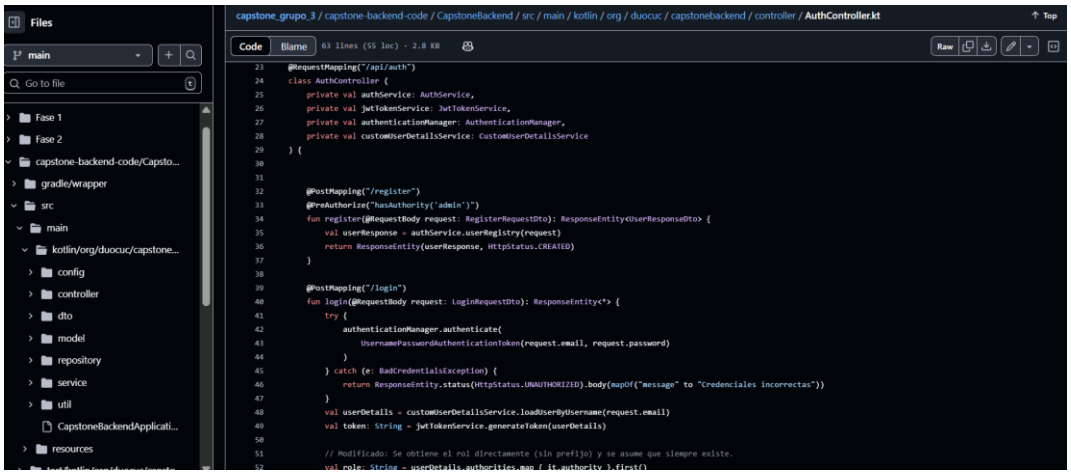
Esta sección documenta el estado actual del desarrollo del proyecto, demostrando que la arquitectura de microservicios ha sido implementada en código Kotlin con Spring Boot, y que los módulos críticos de seguridad, contenido e inteligencia artificial están operativos.

4.1 Estructura de Microservicios y Módulos de Persistencia

La arquitectura diseñada fue implementada con una segmentación clara de la lógica de negocio, asegurando la escalabilidad y la modularidad del sistema.

Estructura de la Base de Código

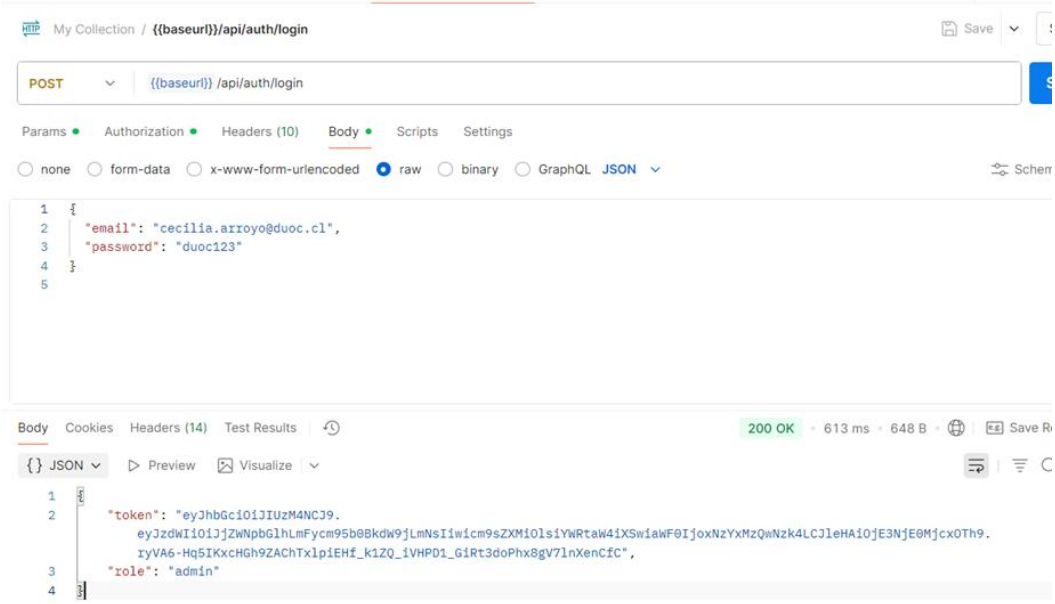
La base de código valida la segmentación de la lógica de negocio en microservicios independientes. El desarrollo actual incluye los cinco servicios CORE (auth-service, content-service, ia-service, game-engine, y scoring-service).

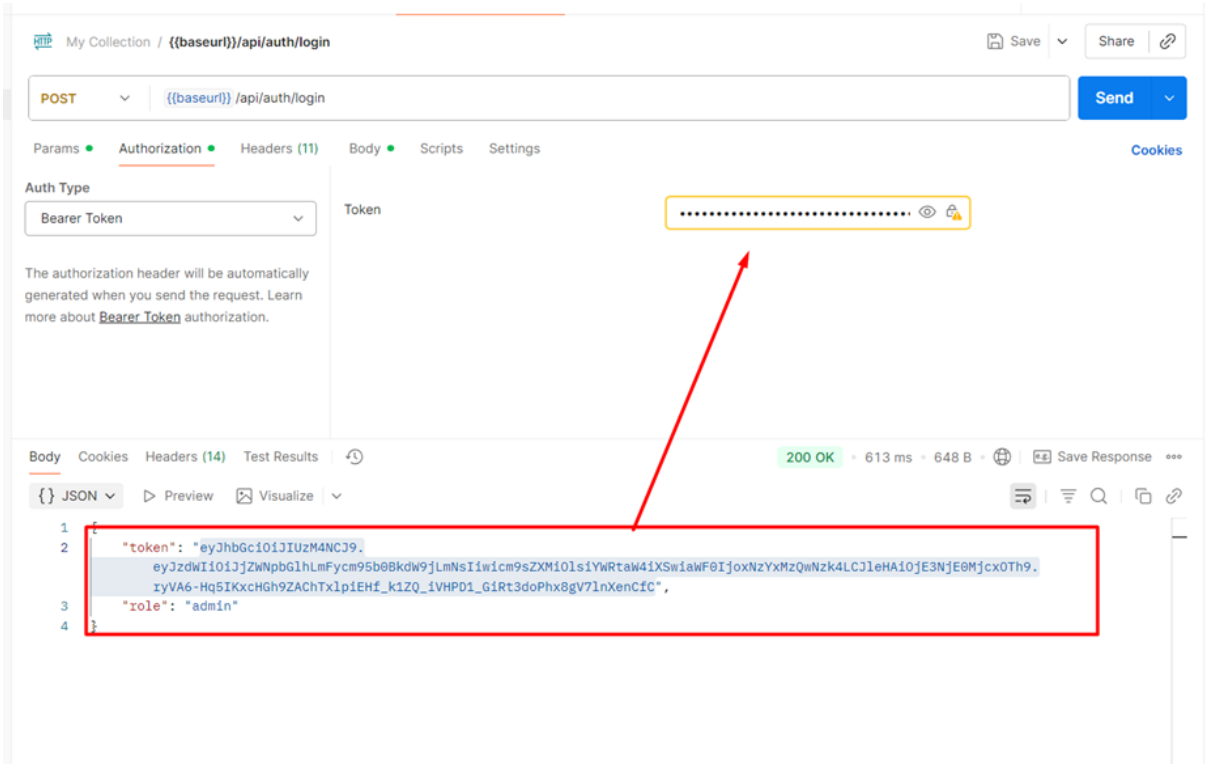


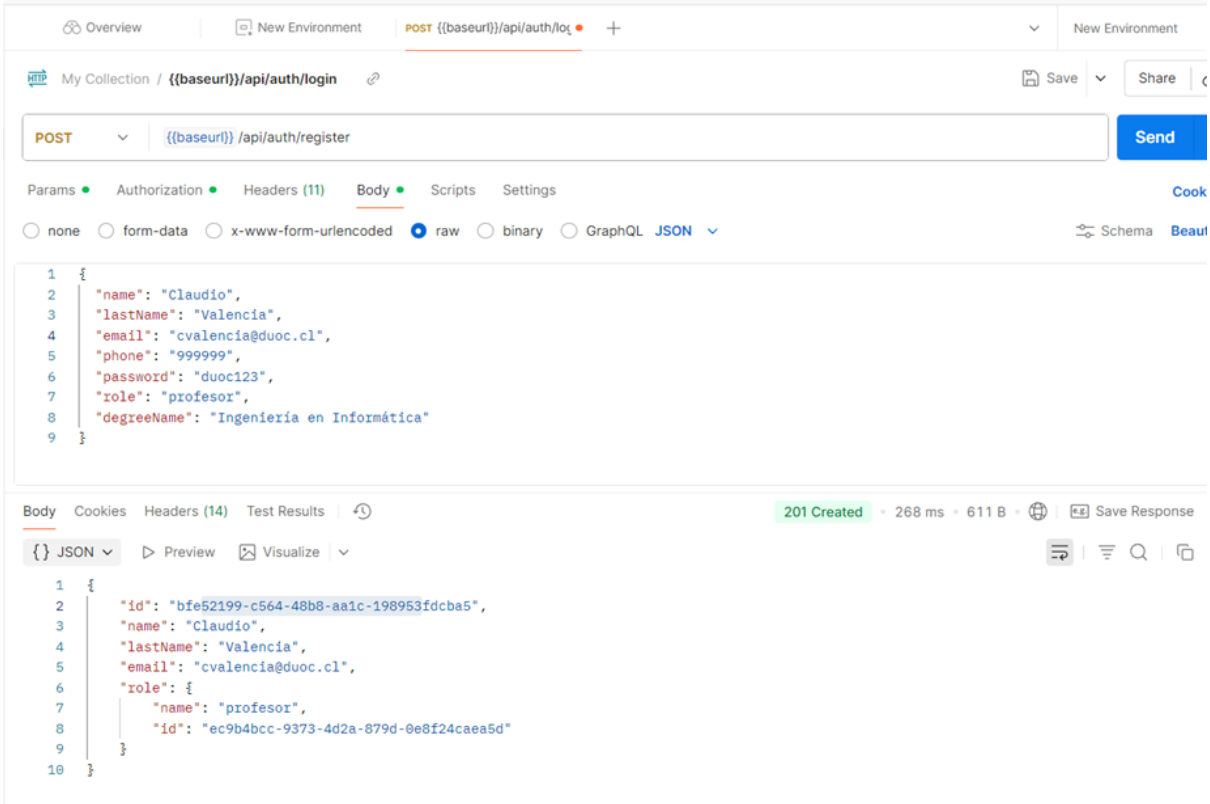
Evidencia de Seguridad (Auth Service)

El Auth Service se encuentra operativo, cumpliendo con los estándares de seguridad de la industria y la implementación de la lógica de sesión del usuario.

- Hashing y JWT: Se implementó el Hashing de contraseñas y la generación/validación del JSON Web Token, lo cual es la base de la seguridad del sistema.




My Collection / {{baseurl}}/api/auth/login
Save
Share
POST
{{baseurl}} /api/auth/login
Send
Params
Authorization
Headers (11)
Body
Scripts
Settings
Cookies
Auth Type
Bearer Token
Token
The authorization header will be automatically generated when you send the request. Learn more about Bearer Token authorization.
Body
Cookies
Headers (14)
Test Results
200 OK
613 ms
648 B
Save Response
JSON
Preview
Visualize
{
"token": "eyJhbGciOiJIUzM4NCJ9.
eyJzdWIiOiJjZWNpbGlhLmFycm95b0BkdW9jLmNsIiwicm9sZXMiOlsiYWRtaW4iXSwiaWF0IjoxNzYxMzQwNzk4LCJleHAiOjE3NjE0MjcxOTh9.
ryVA6-Hq5IKxcHGh9ZAChTxlpiEHf_k1ZQ_iVHPD1_GiRt3doPhx8gV7lnXenCfC",
"role": "admin"
}


Overview
New Environment
POST {{baseurl}}/api/auth/lo
New Environment
My Collection / {{baseurl}}/api/auth/login
Save
Share
POST
{{baseurl}} /api/auth/register
Send
Params
Authorization
Headers (11)
Body
Scripts
Settings
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
JSON
Schema
{
"name": "Claudio",
"lastName": "Valencia",
"email": "cvalencia@duoc.cl",
"phone": "999999",
"password": "duoc123",
"role": "profesor",
"degreeName": "Ingeniería en Informática"
}
Body
Cookies
Headers (14)
Test Results
201 Created
268 ms
611 B
Save Response
JSON
Preview
Visualize
{
"id": "bfe52199-c564-48b8-aa1c-198953fdcba5",
"name": "Claudio",
"lastName": "Valencia",
"email": "cvalencia@duoc.cl",
"role": {
"name": "profesor",
"id": "ec9b4bcc-9373-4d2a-879d-0e8f24caea5d"
}
}

Query | Query History | Scratch Pad

```
1
2  SELECT * FROM usuarios;
```

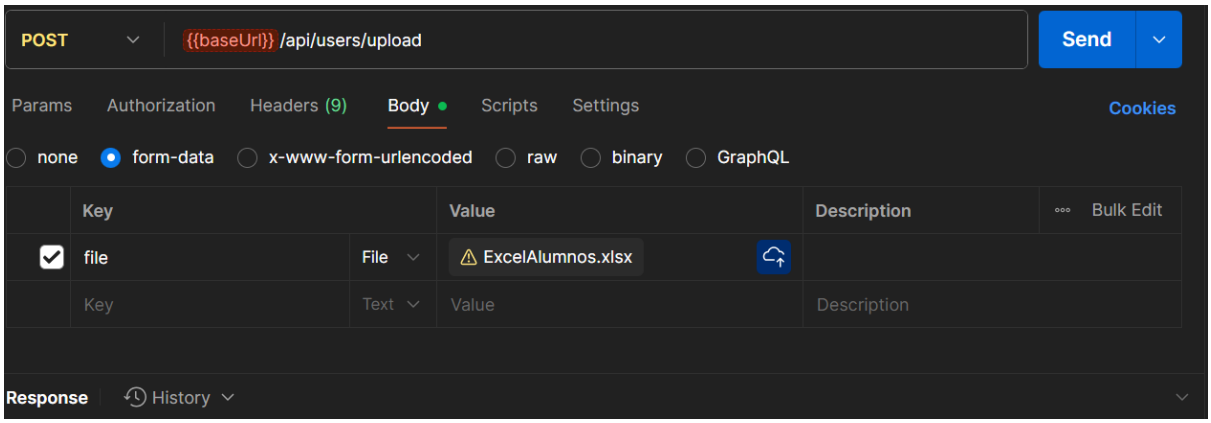
Data Output | Messages | Notifications

Showing rows: 1 to 4 | Page No: 1 of 1

	id [PK] uuid	nombre character varying (100)	apellido character varying (100)	id_rol uuid	id_carrera uuid
1	63a890fe-58ec-44a5-99ba-800af6d27c36	Cecilia	Arroyo	5084555d-2083-42e5-b974-20c3af65c2...	c9bf0ed1
2	b12b868c-fac5-49b3-bacb-54c98b3faf86	Profe	Demo	ec9b4bcc-9373-4d2a-879d-0e8f24caea5d	c9bf0ed1
3	f7cb2690-e48a-4f5f-8b36-07338ff9c697	Cristian	Espinoza	ec9b4bcc-9373-4d2a-879d-0e8f24caea5d	c9bf0ed1
4	bfe52199-c564-48b8-aa1c-198953fdcba5	Claudio	Valencia	ec9b4bcc-9373-4d2a-879d-0e8f24caea5d	c9bf0ed1

- **Carga Masiva de Usuarios:** El Auth Service también implementa la funcionalidad para procesar archivos de carga masiva de alumnos por parte del profesor. Este *endpoint* se encarga de validar los datos, aplicar el Hashing a las contraseñas iniciales y registrar las nuevas cuentas de usuario en PostgreSQL.

```kotlin
@Service  2 Usages  ignacio-leon-m
open class FileUploadService(
    private val authService: AuthService
) {
    open fun processExcelFile(file: MultipartFile): List<RegisterRequestDto> {
        val students = mutableListOf<RegisterRequestDto>()
        file.inputStream.use { input ->
            val workbook = XSSFWorkbook( stream = input)
            val sheet = workbook.getSheetAt( index = 0) // first sheet
            for (row in sheet.drop( n = 1)) { // drop(1) to skip header row
                val name = row.getCell( p0 = 0)?.stringCellValue ?: ""
                val lastName = row.getCell( p0 = 1)?.stringCellValue ?: ""
                val email = row.getCell( p0 = 2)?.stringCellValue ?: ""
                val phone = row.getCell( p0 = 3)?.stringCellValue ?: ""
                val password = name.lowercase() + "1234" // Default password
                students.add(RegisterRequestDto(name, lastName, email, phone, password))
            }
        }
        return students
    }
```



Evidencia de Integración (IA Service)

El IA Service se encuentra operativo, asegurando que los servicios sean funcionales y seguros.

4.2. Motores de Gamificación e Inteligencia Artificial

Se completó el desarrollo de la estructura de los motores de juego y se aseguró la integración del *proxy* de Inteligencia Artificial.

Estructura del Game Engine

Se definieron y codificaron los *endpoints* iniciales del Game Service y el Scoring Service. La estructura de código base (controladores, servicios y repositorios) está lista para recibir la lógica compleja de las partidas y el cálculo de puntajes.

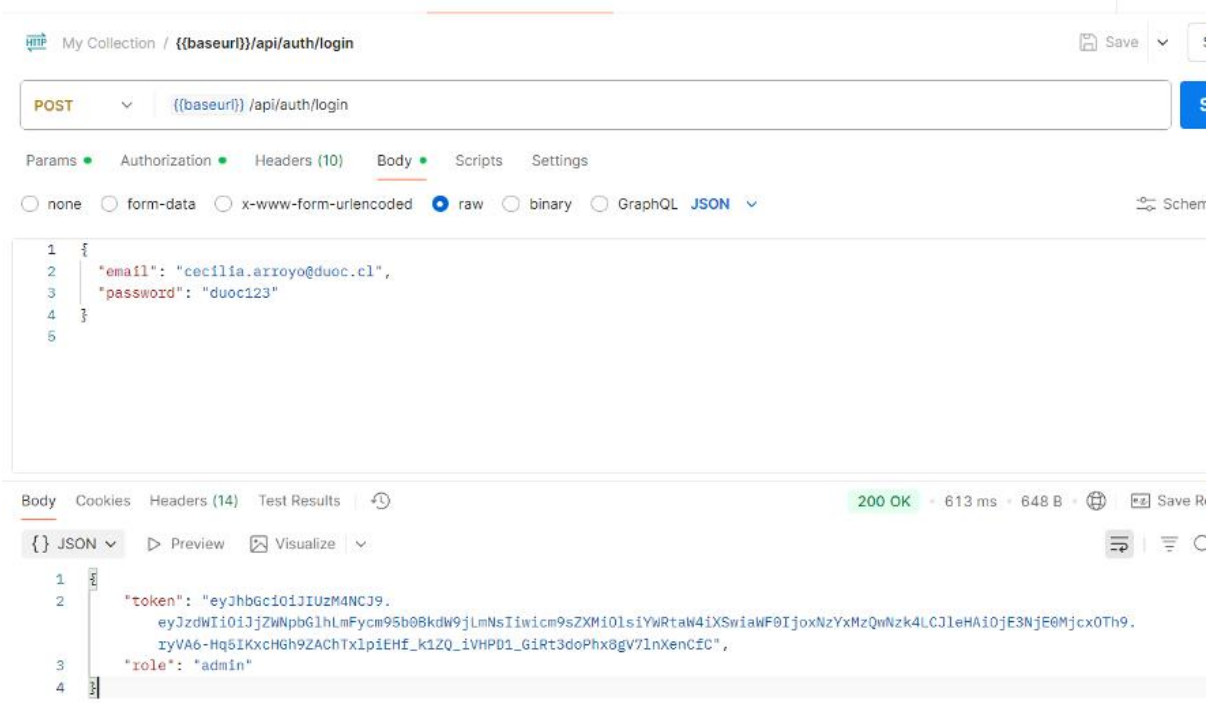
Las entidades de persistencia en PostgreSQL han sido creadas para soportar el ciclo de vida y el registro detallado de la jugabilidad.

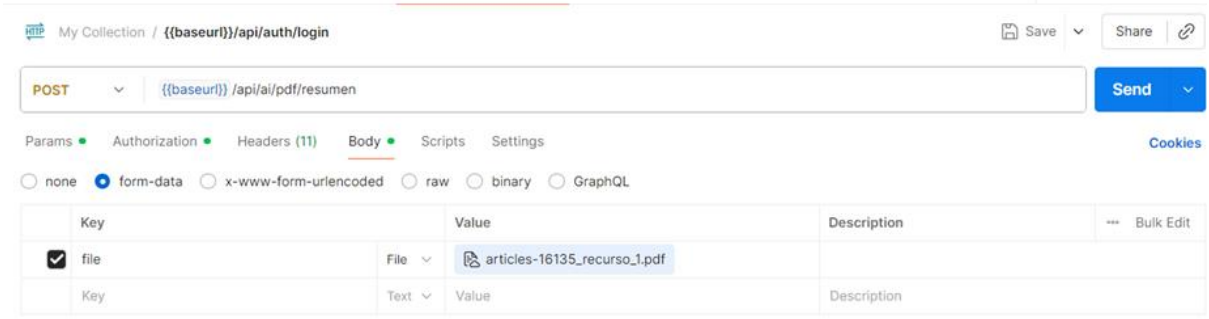
IA Service y Content Service

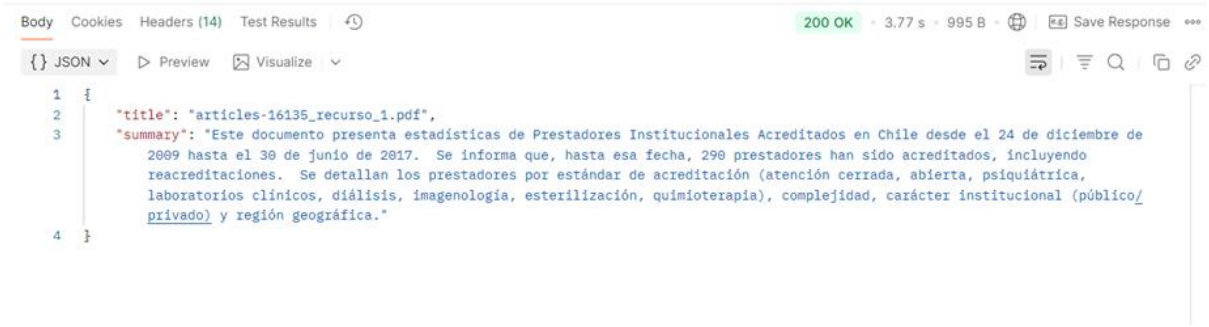
El Content Service implementa la lógica para el *endpoint* de subida de archivos de contenido teórico. Cuando el profesor carga documentos, estos son registrados y almacenados en MongoDB para su procesamiento posterior.

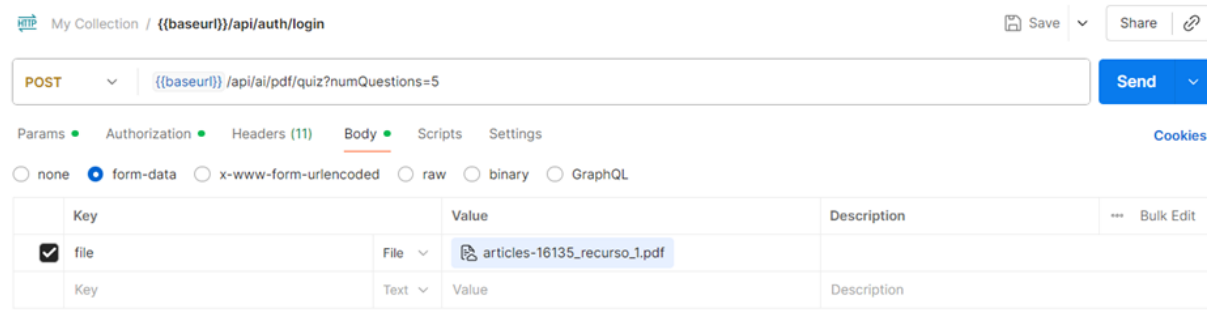
El IA Service está integrado con la API de Google Gemini 2.5 Flash, sirviendo como *proxy* para la generación de contenido.

Se implementó y probó el mecanismo de *fallback*, garantizando la continuidad del servicio al redirigir automáticamente la solicitud al banco de contenido estático (PostgreSQL) cuando se detecta el error de cuota (HTTP 429).

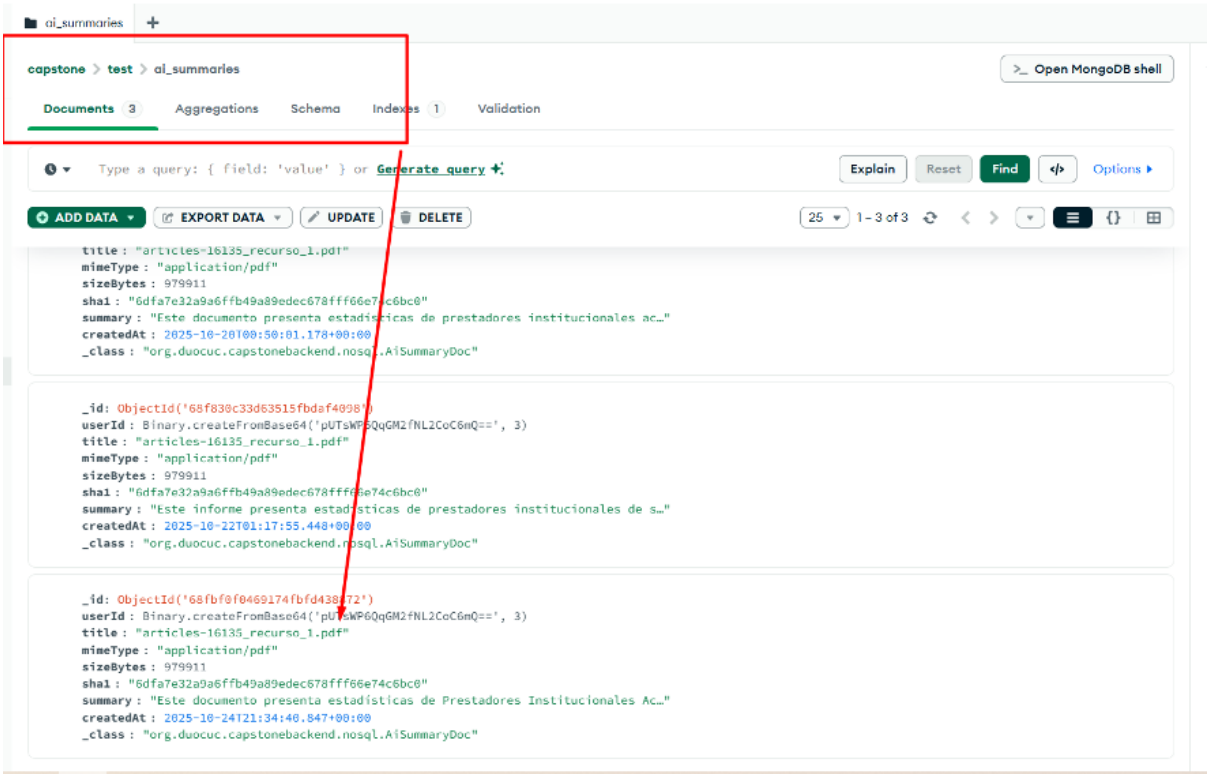



My Collection / {{baseurl}}/api/auth/login
Save
Share
POST
{{baseurl}} /api/ai/pdf/resumen
Send
Params
Authorization
Headers (11)
Body
Scripts
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
Key
Value
Description
Bulk Edit
file
File
articles-16135_recurso_1.pdf
Key
Text
Value
Description


Body
Cookies
Headers (14)
Test Results
200 OK
3.77 s
995 B
Save Response
JSON
Preview
Visualize
{
"title": "articles-16135_recurso_1.pdf",
"summary": "Este documento presenta estadísticas de Prestadores Institucionales Acreditados en Chile desde el 24 de diciembre de 2009 hasta el 30 de junio de 2017. Se informa que, hasta esa fecha, 290 prestadores han sido acreditados, incluyendo reacreditaciones. Se detallan los prestadores por estándar de acreditación (atención cerrada, abierta, psiquiátrica, laboratorios clínicos, diálisis, imagenología, esterilización, quimioterapia), complejidad, carácter institucional (público/privado) y región geográfica."
}


My Collection / {{baseurl}}/api/auth/login
Save
Share
POST
{{baseurl}} /api/ai/pdf/quiz?numQuestions=5
Send
Params
Authorization
Headers (11)
Body
Scripts
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
Key
Value
Description
Bulk Edit
file
File
articles-16135_recurso_1.pdf
Key
Text
Value
Description


Body
Cookies
Headers (14)
Test Results
200 OK
4.99 s
1.55 KB
Save Response
JSON
Preview
Visualize
{
"questions": [
{
"question": "¿Cuál es el período que abarca el documento de estadísticas de prestadores institucionales acreditados?",
"options": [
"Del 1 de enero de 2010 al 31 de diciembre de 2016",
"Del 24 de diciembre de 2009 al 30 de junio de 2017",
"Del 1 de julio de 2010 al 30 de junio de 2017",
"Del 24 de diciembre de 2008 al 30 de junio de 2016"
],
"answerIndex": 1
},
{



Tecnologías

1. Stack Tecnológico Principal

- Lenguajes de Programación: Kotlin (con Java) para el desarrollo de la aplicación móvil nativa en Android.
- Base de Datos:
  - ➢ Relacional (Postgres): Para datos estructurados como perfiles de usuario y contenido de preguntas.
  - ➢ No Relacional (MongoDB): Para datos flexibles y dinámicos como estadísticas de juego y logs.
- Servicios de IA: Google Gemini API para la generación de contenido dinámico y retroalimentación inteligente.
- Servicios en la Nube y Despliegue: Se utilizarán plataformas de nube como Google Cloud Platform (GCP). Se empleará Docker para el empaquetado de cada microservicio.

2. Elección del Modelo de Inteligencia Artificial

La integración de la IA se gestionará exclusivamente por el IA Service de nuestra arquitectura. La elección del modelo de Google Gemini se centra en balancear la capacidad de razonamiento con la gestión estricta de la cuota de uso y la compatibilidad técnica del proyecto.

2.1. Modelo Único Seleccionado: Gemini 2.5 Flash

Nuestra estrategia principal es la unificación total del motor de IA bajo el modelo Gemini 2.5 Flash. Esta decisión estratégica se basa en una combinación de requerimientos técnicos, financieros y de rendimiento:

Criterio	Justificación de la Elección del Modelo
Compatibilidad con el SDK	El código se ha desarrollado sobre el SDK de Java google-genai (versión 2.5+), garantizando la compatibilidad total y eliminando los errores de referencias obsoletas (InputContent/blobs) que afectaban a versiones anteriores.
Estabilidad y Features	Flash ofrece mejor seguimiento de instrucciones y generación de JSON válido, crucial para la estructuración de datos de Quizz y Conceptos.
Rendimiento y Latencia	Las variantes Flash están optimizadas para la velocidad. Su baja latencia es indispensable para ofrecer una experiencia de juego fluida, la generación rápida de preguntas bajo demanda, y un procesamiento eficiente en la carga de documentos.
Gestión de Cuota (Free-Tier)	Como proyecto académico, operamos en el Nivel Gratuito. Gemini 2.5 Flash ofrece la mejor relación rendimiento/costo y posee la mayor cuota de llamadas gratuitas (mejor RPM). Este modelo es el más adecuado para sostener el 100% de las operaciones.

2.2. Rol Estratégico Unificado en el Sistema

Componente	Rol Estratégico de Gemini 2.5 Flash
Motor de Juego (Alta Frecuencia)	Generación instantánea de preguntas de Quizz o conceptos de Ahorcado a partir de *prompts* y *temas*. Su baja latencia asegura una jugabilidad en tiempo real.
Motor de Análisis y Carga (Baja Frecuencia)	Análisis y estructuración de contenido masivo (archivos PDF/DOCX) cargado por el profesor. Se aprovecha su capacidad de razonamiento para extraer y formatear el conocimiento, incluso sin la potencia máxima de la versión Pro.

2.3. Mecanismo de Control de Cuotas y Continuidad

Dado que el Nivel Gratuito de la API opera bajo una Cuota Dinámica Compartida, el riesgo de interrupción del servicio (Error HTTP 429 - Cuota Excedida) persiste. Para mitigar este riesgo y garantizar la continuidad del servicio, el diseño del IA Service incluye los siguientes protocolos:

1. Optimización del Consumo: Se utilizará Gemini 2.5 Flash para el 100% de las operaciones, consumiendo la menor cantidad de *tokens* posible por llamada y maximizando la eficiencia de la cuota.
2. Mecanismo de Fallback (Contenido Estático): Se implementará una lógica estricta en el IA Service para que, al detectar un error de cuota agotada (HTTP 429), el sistema active automáticamente un fallback. Este mecanismo recurrirá al banco de contenido estático almacenado en PostgreSQL.

Esta doble estrategia permite aprovechar el poder de la IA en su versión gratuita sin comprometer la estabilidad del sistema, transformando el obstáculo de la cuota en un requerimiento no funcional que hemos abordado directamente con diseño de software.

3. Entorno Cloud y Seguridad

El proyecto se sustenta en una arquitectura de microservicios desplegada en Google Cloud Platform (GCP). El backend principal se aloja en una instancia de Google Compute Engine (GCE), server-capstone-g3, que está configurada con 2 vCPUs y 1 GB de RAM, siendo adecuada para entornos de desarrollo y pruebas. Se ha configurado un servidor web Nginx para actuar como un proxy inverso para la aplicación de backend y servir contenido estático. La base de datos, PostgreSQL, se utiliza como la base de datos principal del sistema.

3.1.Configuración de Seguridad

La seguridad del sistema ha sido una prioridad, especialmente después de una reconfiguración completa para abordar vulnerabilidades iniciales. El equipo ha implementado medidas para dejar todo el sistema seguro en su configuración actual, gestionando la seguridad a través de las reglas de Firewall de GCP y una configuración estricta de los servicios.

- Acceso a la Base de Datos Restringido: La configuración inicial, que permitía conexiones remotas desde cualquier dirección IP, ha sido eliminada. Ahora, el acceso al puerto 5432 de PostgreSQL está restringido a un rango de IP específico para evitar conexiones no autorizadas.
- Autenticación Reforzada: El acceso SSH por contraseña ha sido deshabilitado, y ahora el acceso al servidor se realiza exclusivamente a través de claves SSH para garantizar una conexión segura. Para el acceso a la base de datos, se ha creado un nuevo usuario seguro (cvalencia) con una contraseña fuerte y aleatoria. La autenticación exige el método robusto scram-sha-256 para todas las conexiones.
- Recomendaciones para Futuro: Se ha definido un plan para el entorno de producción que garantiza una arquitectura aún más segura. El procesamiento de documentos con la API de Gemini se realizará en un servidor de backend que se comunicará con la base de datos de forma local, evitando exponer las claves de la API. Se recomienda el despliegue de esta aplicación en Google Cloud Run para una mayor seguridad, escalabilidad y eficiencia de costos en un entorno de producción.

3.2. Diseño e Implementación del Módulo de Autenticación (Auth Service)

El Auth Service fue el primer microservicio desarrollado (Sprint 2), sirviendo como la Capa de Seguridad a nivel de Aplicación y el punto de entrada para la autenticación, utilizando Java/Kotlin y el *framework* Spring Boot.

- Desarrollo del Contrato API: Se documentaron formalmente los DTOs y los *endpoints* (/auth/login, /auth/register) utilizando OpenAPI/Swagger. Este documento se estableció como el Contrato API oficial que el *frontend* de la aplicación móvil debe consumir, garantizando una integración fluida y un desarrollo paralelo.
- Implementación de la Lógica de Seguridad:
  - ➢ Hashing de Contraseñas: Para asegurar las credenciales, se implementó el algoritmo de *hashing* dentro del código del Auth Service.
  - ➢ Módulo JWT (JSON Web Token): Se desarrolló el módulo de JWT en Kotlin/Spring Boot. Este módulo es responsable de generar y firmar un *token* cifrado al iniciar sesión, que incluye los *claims* esenciales del usuario.

- Flujo de Integración *Frontend*: El desarrollo del *frontend* en el Sprint 2 se centró en consumir el *endpoint* /auth/login, recibir el JWT y almacenarlo de forma segura para que pueda ser enviado en la cabecera de autenticación (Authorization: Bearer <token>) en todas las peticiones subsiguientes. Esto valida la identidad del usuario en cada microservicio de la arquitectura.

Conclusión

Esta entrega marca la culminación exitosa de la fase de diseño, planificación detallada y la implementación de los microservicios CORE del proyecto, validando la solidez de nuestra Arquitectura de Microservicios que garantiza la modularidad y escalabilidad.

Hemos establecido una clara separación de responsabilidades, con el Auth Service gestionando la seguridad mediante JWT y Hashing, y el User Service a cargo de la creación de perfiles y la carga masiva, todo respaldado por una estrategia de persistencia que utiliza PostgreSQL para la integridad de datos y MongoDB para la ingesta de contenido masivo.

El equipo ha avanzado rigurosamente bajo la metodología Scrum, logrando la integración funcional con la API de Google Gemini y la implementación crucial del mecanismo de *fallback* al banco estático, mitigando eficazmente el riesgo de interrupción del servicio. Con la estructura del Game Service y el Scoring Service definida y lista para la lógica compleja, y con una colaboración equitativa demostrada en el código, el proyecto se encuentra en un punto óptimo para enfocarse plenamente en el desarrollo final de la lógica de gamificación en los próximos Sprints, proyectando una materialización exitosa y profesional de esta solución educativa.