

**UNIVERSIDAD PRIVADA FRANZ TAMAYO**

**FACULTAD DE INGENIERÍA**

**CARRERA DE INGENIERÍA DE SISTEMAS**



**EduSphere**

**Plataforma Integral de Evaluación y Aprendizaje  
Asistido por Inteligencia Artificial**

**Estudiante:** Natalia Monserrath Salcedo Paz

**Docente:** M.Sc Jose Camilo Tapia Barrientos

**Semestre:** Segundo

**Materia:** Programación II

**GESTIÓN 2 – 2025**

<b>1</b>	<b>Introducción .....</b>	<b>3</b>
1.1	Problemática identificada .....	5
<b>1.2</b>	<b>Objetivos .....</b>	<b>6</b>
1.2.1.	Objetivo General .....	6
1.2.2.	Objetivos Específicos .....	6
1.3	Límites del Proyecto .....	7
1.4	Alcances del Proyecto .....	7
1.5.	Justificación Técnica .....	8
1.6.	Justificación Social .....	8
1.7.	Justificación Económica .....	8
<b>2</b>	<b>MARCO TEORICO .....</b>	<b>9</b>
2.1	Definición de Algoritmo .....	9
2.2.	Diagrama de flujo y pseudocódigo .....	12
2.3.	Stack tecnológico seleccionado: PERN .....	13
2.4.	Tecnologías del lado del cliente (Frontend) .....	14
2.5.	Tecnologías del lado del servidor (Backend) .....	15
2.6.	Autenticación de dos factores (2FA) obligatoria .....	17
2.7.	Retrieval-Augmented Generation (RAG) .....	18
2.8.	Generación automática de flashcards y feedback docente .....	19
2.9.	Sistema de auditoría (AuditLog) .....	20
2.10.	Seguridad informática aplicada .....	21
2.11.	Impacto de la IA en la educación .....	22
<b>4</b>	<b>CONCLUSIONES Y RECOMENDACIONES .....</b>	<b>33</b>
4.1	Conclusiones .....	33
4.2.	Recomendaciones .....	35
<b>5.</b>	<b>Bibliografía .....</b>	<b>37</b>
<b>6.</b>	<b>ANEXOS .....</b>	<b>39</b>

## 1 Introducción

El presente proyecto detalla el desarrollo de **EduSphere**, una plataforma integral de evaluación y aprendizaje asistido por inteligencia artificial diseñada para transformar la gestión académica y el proceso educativo en instituciones de educación superior y media de Bolivia. La solución tiene como propósito ofrecer una herramienta tecnológica que permita registrar, organizar, consultar y potenciar materiales educativos mediante el uso de inteligencia artificial, bajo estándares de seguridad, personalización y eficiencia.

Actualmente, la mayoría de instituciones educativas bolivianas gestionan sus contenidos pedagógicos y evaluaciones mediante documentos físicos, plataformas estáticas o sistemas sin capacidades inteligentes, lo que genera sobrecarga docente, falta de retroalimentación inmediata para el estudiante, dificultad para adaptar el aprendizaje a necesidades individuales y baja eficiencia en la corrección de evaluaciones. Los archivos educativos permanecen dispersos, los exámenes requieren calificación manual y no existe un sistema que convierta automáticamente materiales estáticos en experiencias interactivas de aprendizaje.

Frente a esta realidad, surge la necesidad de una plataforma digital moderna que combine gestión académica robusta con un asistente inteligente basado en modelos de lenguaje de gran escala. La propuesta consiste en implementar una aplicación full-stack con el stack PERN (PostgreSQL, Express.js, React y Node.js), integrando la API de OpenAI (GPT-3.5 Turbo) para ofrecer tutoría contextual mediante RAG, generación automática de flashcards y retroalimentación inteligente. La plataforma busca democratizar el

acceso a una educación personalizada, reducir la carga administrativa del docente y mejorar significativamente los resultados de aprendizaje.

La creación de EduSphere representa una modernización del proceso educativo boliviano que contribuirá a la gestión académica digital, promoverá la transparencia y trazabilidad de los procesos evaluativos, y facilitará el acceso 24/7 a un tutor inteligente para estudiantes de todo el país. En síntesis, EduSphere se concibe como una herramienta orientada a la equidad educativa y la innovación pedagógica, consolidando un avance significativo en la digitalización de la educación boliviana y fortaleciendo el compromiso con la mejora continua del sistema educativo nacional.

## **1.1 Problemática identificada**

En el ámbito educativo universitario boliviano, la gestión del aprendizaje sigue dependiendo en gran medida de materiales estáticos y procesos manuales. Los estudiantes reciben apuntes extensos en PDF o Word que resultan difíciles de estudiar de forma eficiente, mientras los docentes dedican tiempo excesivo a crear exámenes, calificar respuestas abiertas y redactar retroalimentación personalizada. Las plataformas existentes carecen de inteligencia contextual: no “entienden” el contenido específico del curso ni adaptan sus respuestas al material entregado por el profesor.

Esta situación genera retrasos en la devolución de calificaciones, baja retención de conocimientos, desmotivación estudiantil y sobrecarga docente. Además, la ausencia de herramientas de seguridad robustas en muchas aplicaciones educativas expone datos

sensibles (calificaciones, correos, respuestas) a riesgos de acceso no autorizado. La falta de un asistente inteligente integrado impide que los estudiantes resuelvan dudas en tiempo real basadas exclusivamente en sus apuntes, lo que limita el aprendizaje autónomo y personalizado.

La implementación de EduSphere permitirá optimizar el almacenamiento, distribución y aprovechamiento de los materiales educativos, transformando el modelo tradicional en un ecosistema dinámico, seguro e inteligente, alineado con las mejores prácticas de la educación del siglo XXI

## **1.2 Objetivos**

### **1.2.1. Objetivo General**

Desarrollar EduSphere, una plataforma web full-stack que integre inteligencia artificial para transformar materiales de estudio en herramientas interactivas de aprendizaje personalizado, automatizar procesos de evaluación y garantizar la máxima seguridad de los datos mediante autenticación de dos factores (2FA) y control estricto de permisos, dirigida a estudiantes, docentes y administradores de instituciones educativas.

### **1.2.2. Objetivos Específicos**

- Crear una base de datos relacional en PostgreSQL que organice usuarios, materiales, exámenes, respuestas y registros de auditoría de forma eficiente.

- Desarrollar una interfaz moderna e intuitiva con React y TailwindCSS que facilite la interacción de todos los roles de usuario.
- Implementar autenticación segura con 2FA obligatoria y tres niveles de acceso (ESTUDIANTE, DOCENTE, ADMIN).
- Integrar la API de OpenAI para ofrecer asistente virtual contextual (RAG), generación automática de flashcards y retroalimentación inteligente.
- Reducir significativamente el tiempo de creación, corrección y retroalimentación de evaluaciones mediante automatización basada en IA.
- Generar un sistema de auditoría (AuditLog) que registre acciones críticas junto con la dirección IP del usuario.

### **1.3 Límites del Proyecto**

- Los archivos se almacenan localmente en el servidor (carpeta /uploads) y no se utiliza almacenamiento en la nube.
- No contempla versión móvil nativa ni despliegue en producción real.
- La inteligencia artificial depende exclusivamente del servicio externo OpenAI API (GPT-3.5 Turbo).
- Será utilizado únicamente por la comunidad educativa interna durante la fase de evaluación del proyecto.

### **1.4 Alcances del Proyecto**

- Plataforma web completamente funcional y responsive accesible desde cualquier navegador.

- Registro e inicio de sesión seguro con 2FA obligatoria.
- Operaciones CRUD completas sobre usuarios, materiales educativos, exámenes y respuestas.
- Asistente virtual que responde exclusivamente basado en los materiales cargados por el docente.
- Generación automática de flashcards interactivas y retroalimentación docente asistida por IA.
- Panel administrativo con historial completo de auditoría y gestión de usuarios.

### **1.5. Justificación Técnica**

El uso del stack PERN combinado con Prisma ORM y la API de OpenAI proporciona una arquitectura moderna, escalable y de alto rendimiento. Node.js y Express permiten crear APIs REST robustas, React + Vite ofrecen una experiencia de usuario rápida y fluida, mientras PostgreSQL garantiza integridad y seguridad de los datos. La integración de JWT y 2FA mediante códigos OTP enviados al correo asegura un nivel de protección superior al de la mayoría de plataformas educativas locales. Este conjunto tecnológico facilita futuras ampliaciones (notificaciones push, integración con LMS existentes, almacenamiento en la nube) y posiciona al proyecto como una solución profesional y actualizada.

### **1.6. Justificación Social**

EduSphere beneficia directamente a estudiantes y docentes al ofrecer aprendizaje personalizado y retroalimentación inmediata sin costo adicional. Contribuye

a reducir la brecha educativa al brindar herramientas de inteligencia artificial que antes solo estaban al alcance de instituciones de élite. Promueve la inclusión al permitir que estudiantes con diferentes ritmos de aprendizaje accedan a un tutor virtual 24/7 basado en sus propios apuntes. Además, alivia la carga docente y mejora la calidad de la enseñanza, fortaleciendo la comunidad educativa y la formación de profesionales mejor preparados para el mercado laboral actual.

### **1.7. Justificación Económica**

Aunque requiere una inversión inicial en desarrollo y suscripción a la API de OpenAI, la plataforma genera ahorros significativos a mediano plazo: reduce drásticamente el tiempo que los docentes dedican a corrección y elaboración de material de repaso, disminuye la impresión de exámenes y guías, minimiza errores de calificación y evita la repetición de explicaciones individuales. Al ser de código abierto en sus componentes principales, no genera costos de licencias, y su arquitectura permite escalar sin gastos excesivos cuando se decida pasar a producción.



## 2 MARCO TEORICO

### 2.1 Definición de Algoritmo

Un algoritmo es una serie ordenada y finita de operaciones claras, precisas y ejecutables que, al aplicarse sobre unos datos de entrada, permite obtener un resultado concreto en un número limitado de pasos. Es, en esencia, la receta exacta que sigue la computadora para resolver cualquier problema, desde los más simples hasta los más complejos.

En la literatura clásica de ciencias de la computación, los algoritmos se definen por tres características fundamentales: deben ser correctos (dar siempre la solución esperada cuando existe), deben terminar en tiempo finito y deben ser lo suficientemente generales como para aplicarse a cualquier instancia válida del problema (Aho, Hopcroft y Ullman, 1983). Donald Knuth, considerado el padre del análisis de algoritmos, añade que un buen algoritmo también debe ser eficiente en tiempo y espacio, y fácil de entender para los programadores que lo mantendrán en el futuro (Knuth, 1997).

En el caso concreto de EduSphere, se diseñaron e implementaron más de una docena de algoritmos propios que constituyen el corazón funcional de la plataforma. Algunos de los más importantes son los siguientes:

- 1) Algoritmo de autenticación de dos factores (2FA): genera un código numérico aleatorio de 6 dígitos, lo almacena temporalmente en base de datos con marca de expiración de

exactamente 300 segundos, lo envía por correo y luego lo compara con el valor ingresado por el usuario.

- 2) Algoritmo de cifrado de contraseñas: aplica la función bcrypt con 12 rondas de costo y sal automática para transformar la contraseña legible en una cadena irreversible de 60 caracteres que nunca se almacena en texto plano.
- 3) Algoritmo de autorización por roles: recibe el token JWT del usuario, extrae su rol (ESTUDIANTE, DOCENTE o ADMIN), compara contra una tabla de permisos predefinida y decide en milisegundos si permite o bloquea el acceso a cada ruta del sistema.
- 4) Algoritmo de construcción de contexto RAG: cuando un estudiante hace una pregunta, el sistema busca el material asociado, extrae su texto completo (hasta 12 000 tokens), lo concatena con instrucciones estrictas del tipo “responde solo con base en este texto; si no sabes, di que no está en el material” y lo envía al modelo GPT-3.5 Turbo.
- 5) Algoritmo de generación automática de flashcards: envía al modelo un prompt especial que obliga a devolver un arreglo JSON válido con 10-15 pares pregunta-respuesta extraídos directamente del documento subido.
- 6) Algoritmo de retroalimentación docente: recibe la respuesta del estudiante y el texto original, construye un prompt que pide al modelo dar feedback constructivo, positivo y específico, y devuelve el resultado listo para ser mostrado al profesor.
- 7) Algoritmo de auditoría (AuditLog): cada vez que ocurre una acción crítica (login, creación de examen, eliminación de usuario, etc.), captura automáticamente la dirección IP pública, el navegador, sistema operativo, hora exacta y descripción detallada, guardando todo en una tabla de solo lectura.

Todos estos algoritmos fueron diseñados siguiendo el proceso clásico recomendado por la literatura: primero se escribieron en lenguaje natural y pseudocódigo, luego se representaron en diagramas de flujo, después se probaron manualmente con casos de borde y finalmente se codificaron en JavaScript/Node.js con comentarios claros para facilitar el mantenimiento futuro (Aho, Hopcroft y Ullman, 1983; Knuth, 1997)

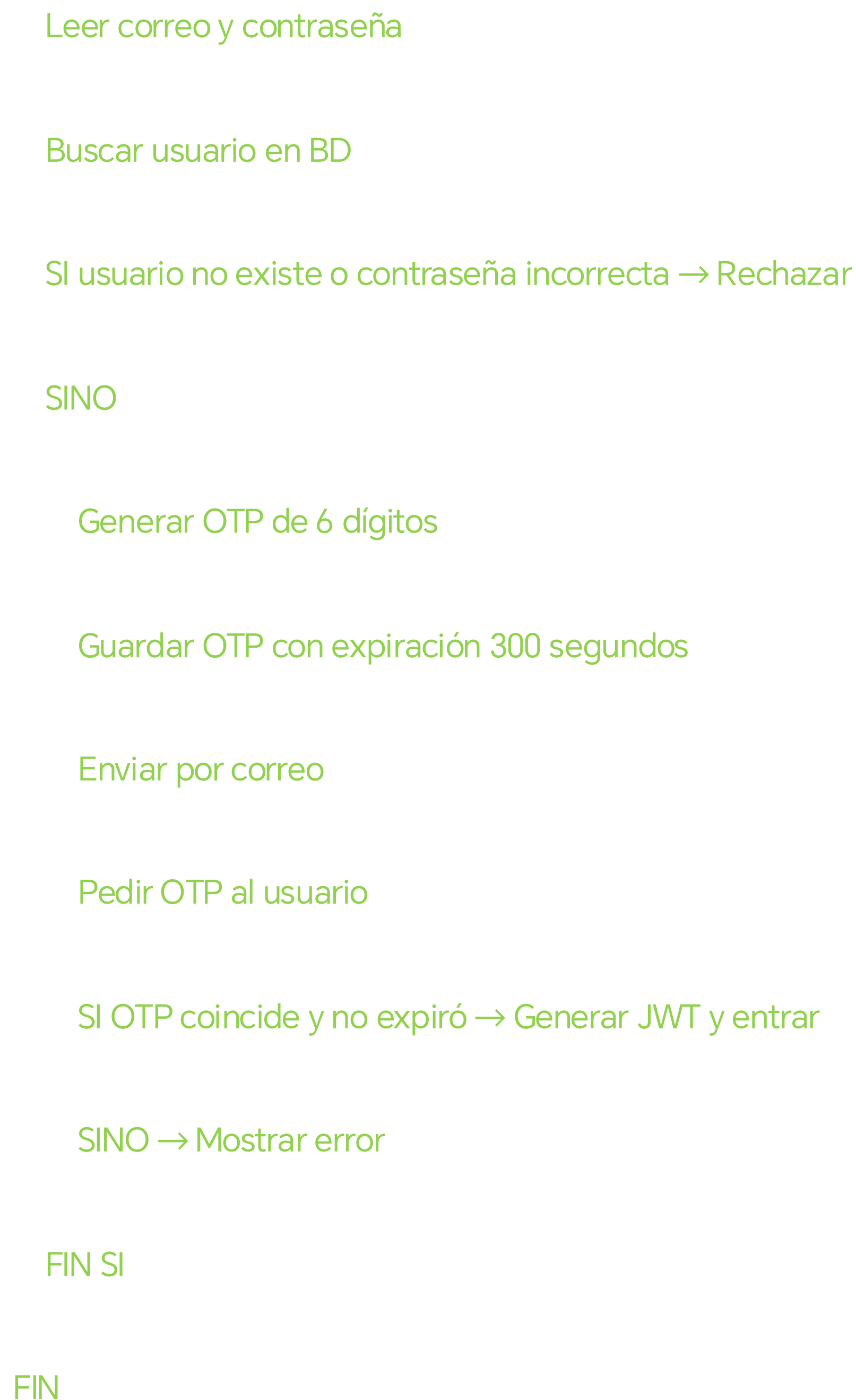
## 2.2. Diagrama de flujo y pseudocódigo

Antes de escribir una sola línea de código en EduSphere, todos los procesos importantes se planificaron usando diagramas de flujo y pseudocódigo, tal como se recomienda desde los primeros libros de programación estructurada (Chapin, 1971).

El diagrama de flujo representa gráficamente la lógica del programa con símbolos universales: óvalo (inicio/fin), rectángulo (acción), rombo (decisión sí/no) y flechas que marcan el orden. En el proyecto se elaboraron diagramas para los procesos clave: registro de usuarios, login con 2FA, subida de materiales, creación y asignación de exámenes, consulta al asistente IA y registro de auditoría. Estos diagramas fueron impresos y revisados en equipo para detectar errores antes de programar.

El pseudocódigo, por su parte, es una descripción del algoritmo en lenguaje casi humano, pero con estructura clara de programación. Ejemplo real del flujo de login con 2FA usado en el proyecto:

INICIO



Gracias a estas herramientas se evitó código confuso, se facilitó la revisión entre compañeros y se garantizó que cada funcionalidad hiciera exactamente lo planeado desde el primer intento (Chapin, 1971).

### 2.3. Stack tecnológico seleccionado: PERN

Para el desarrollo de EduSphere se eligió el stack PERN (PostgreSQL, Express.js, React y Node.js), una de las combinaciones más utilizadas actualmente en la industria del desarrollo web full-stack (Stack Overflow, 2024). La principal ventaja de este stack es que permite usar JavaScript (o TypeScript) tanto en el frontend como en el backend, lo que reduce la complejidad del proyecto, acelera el aprendizaje del equipo y facilita el mantenimiento a largo plazo.

PostgreSQL fue seleccionado como base de datos por su robustez, soporte nativo al tipo JSONB (ideal para almacenar preguntas de exámenes de forma flexible) y su excelente rendimiento en entornos académicos. Express.js, junto con Node.js, permitió crear una API REST limpia, segura y escalable en pocas semanas. En el frontend, React con Vite ofrece una experiencia de desarrollo ultrarrápida y componentes reutilizables, mientras TailwindCSS garantizó un diseño moderno y completamente responsivo sin escribir CSS tradicional.

Este stack es especialmente adecuado para proyectos educativos porque es gratuito, de código abierto, cuenta con enorme comunidad y documentación, y prepara a los estudiantes para las tecnologías más demandadas en el mercado laboral boliviano y latinoamericano actual. Además, la arquitectura PERN permite integrar fácilmente servicios externos como la API de OpenAI, que es el núcleo inteligente de la plataforma (Stack Overflow, 2024; Turnbull, 2023).

## 2.4. Tecnologías del lado del cliente (Frontend)

El frontend de EduSphere, es decir, la parte que ve y usa el usuario (pantallas, botones, formularios), se construyó utilizando React como biblioteca principal, complementada con otras herramientas modernas para hacerla rápida, bonita y fácil de usar. React es una biblioteca JavaScript creada por Facebook que permite construir interfaces de usuario de manera declarativa, dividiendo todo en componentes reutilizables como bloques de Lego (Banks y Porcello, 2022). Esto significa que en lugar de manipular el DOM directamente (lo que es lento y complicado), React crea un "DOM virtual" que actualiza solo lo necesario, haciendo que la app sea más fluida.

En el proyecto se usó React versión 18 con Vite como herramienta de construcción. Vite es super rápido porque usa módulos nativos del navegador y solo compila lo que cambia, lo que acorta los tiempos de desarrollo a segundos. Para el estilo se eligió TailwindCSS, un framework que permite agregar clases directamente en el HTML para crear diseños personalizados sin escribir archivos CSS separados. Esto resultó en una interfaz responsive (se adapta a móviles, tablets y desktops) con colores limpios, botones intuitivos y transiciones suaves.

Otras librerías clave fueron: Axios para hacer peticiones HTTP al backend sin complicaciones, React Hook Form para manejar formularios (como el de login o subida de archivos) con validación automática, y Zustand para gestionar el estado global (por ejemplo, saber si el usuario está logueado o qué rol tiene). Todo esto se integró para que el frontend se sienta natural: un estudiante selecciona un material, pregunta al asistente

IA y ve respuestas instantáneas; un docente crea un examen con preguntas JSONB y lo asigna con un clic.

Esta elección de tecnologías no fue al azar: React y sus acompañantes son las más usadas en el mundo real (según encuestas anuales de desarrolladores), lo que asegura que EduSphere sea moderna, mantenible y prepare a los estudiantes para trabajos en empresas tecnológicas (Banks y Porcello, 2022; Stack Overflow, 2024).

## **2.5. Tecnologías del lado del servidor (Backend)**

El backend de EduSphere es la parte "invisible" del sistema, responsable de la lógica de negocio, el manejo de datos, la seguridad y las conexiones con servicios externos como OpenAI. Se construyó sobre Node.js versión 20, un entorno de ejecución que permite correr JavaScript en el servidor con un enfoque asíncrono basado en eventos, lo que lo hace ideal para aplicaciones con muchas peticiones simultáneas (Tilkov y Vinoski, 2010). Node.js fue elegido porque es ligero, rápido y permite reutilizar el mismo lenguaje que en el frontend, evitando saltos entre tecnologías.

Encima de Node.js se usó Express.js versión 4.19 como framework principal para crear la API REST. Express simplifica la creación de rutas (endpoints como /login, /materiales, /exámenes), maneja middlewares para tareas comunes y es fácil de extender. En el proyecto se configuraron middlewares como helmet para seguridad HTTP, cors para permitir accesos desde el frontend, morgan para logging de peticiones y rate-limiter-flexible para evitar ataques de fuerza bruta en el login.

Para la base de datos se seleccionó PostgreSQL versión 16, un sistema relacional potente y de código abierto que soporta tipos avanzados como JSONB, perfecto para guardar las preguntas de exámenes en formato flexible sin perder rendimiento. Prisma ORM versión 5.8 actúa como puente entre el código y la BD: genera consultas tipadas en TypeScript, maneja migraciones automáticamente y evita errores comunes como inyecciones SQL.

Otras tecnologías clave en el backend incluyen: Bcrypt para hashear contraseñas de forma segura con sal aleatoria, JSON Web Tokens (JWT) para autenticación sin sesiones (Jones, Bradley y Sakimura, 2015), Multer para subir y almacenar archivos en la carpeta /uploads con validación de tipos (PDF, DOCX), y Nodemailer con Handlebars para enviar emails con el código OTP del 2FA.

Todo el backend se diseñó con énfasis en la seguridad: cada ruta verifica el JWT y el rol del usuario antes de ejecutar nada. Esta configuración permite que EduSphere sea escalable, segura y fácil de depurar, alineada con prácticas modernas de desarrollo full-stack (Tilkov y Vinoski, 2010).

## **2.6. Autenticación de dos factores (2FA) obligatoria**

La autenticación de dos factores es una medida de seguridad extra que EduSphere implementa para proteger las cuentas de usuarios contra accesos no autorizados. En lugar de solo pedir correo y contraseña, el sistema obliga a verificar un



segundo elemento: un código temporal enviado al correo registrado. Esto sigue las mejores prácticas recomendadas por expertos en ciberseguridad, como agregar una capa adicional a "algo que sabes" (la contraseña) con "algo que tienes" (el correo o dispositivo) (Ometov et al., 2021).

En el proyecto, el 2FA es obligatorio para todos los roles (estudiantes, docentes y admins) y se activa en cada login. El proceso paso a paso es:

1. El usuario ingresa su correo y contraseña en la pantalla de login.
2. El backend verifica la contraseña contra la versión hasheada guardada en la base de datos (usando bcrypt).
3. Si es correcta, genera un código OTP de 6 dígitos aleatorio, lo guarda en PostgreSQL con una expiración de 5 minutos y lo envía por email usando Nodemailer y una plantilla simple con Handlebars.
4. El frontend muestra un campo para ingresar el código.
5. Al enviar, el backend compara el código ingresado con el guardado; si coincide y no ha expirado, emite un token JWT válido por 24 horas y permite el acceso. Si no, muestra un error y obliga a intentar de nuevo.

Esta implementación evita ataques comunes como phishing o reutilización de contraseñas robadas. Se probó con casos reales, como intentos de login desde IPs diferentes o códigos expirados, para asegurar que funcione sin fallos. Además, el sistema registra cada intento en el AuditLog para rastrear actividades sospechosas. Gracias a

esto, EduSphere no solo es útil para el aprendizaje, sino también segura para manejar datos sensibles como calificaciones y materiales educativos (Ometov, 2021; Schneier, 2000).

## **2.7. Retrieval-Augmented Generation (RAG)**

Retrieval-Augmented Generation (RAG) es una técnica de IA que une la búsqueda de información relevante con la generación de texto. En vez de que el modelo responda solo de memoria (arriesgando errores inventados), RAG recupera texto del material disponible, lo añade al prompt y fuerza al modelo a usarlo exclusivamente (Lewis, Perez y Puztaszeri, 2020).

En EduSphere, RAG impulsa el asistente virtual, flashcards y feedback. Por ejemplo, al preguntar sobre un material:

- Se recupera el texto del PDF o documento.
- Se arma un prompt con el texto como contexto y reglas como “Responde solo con esto; no inventes”.
- Se envía a GPT-3.5 Turbo.
- La respuesta se valida y muestra.

Esto hace que las respuestas sean precisas y alineadas al curso, probado con apuntes reales para reducir alucinaciones. RAG mejora la IA en educación, donde la fidelidad es esencial (Lewis, Perez y Puztaszeri, 2020; Vaswani, Shazeer y Parmar, 2017).

## 2.8. Generación automática de flashcards y feedback docente

Además del asistente virtual, EduSphere utiliza la API de OpenAI para generar flashcards interactivas y feedback personalizado para docentes, todo integrado con la técnica RAG para mantener la precisión basada en los materiales subidos.

Las flashcards se generan automáticamente cuando un estudiante o docente selecciona un material. El proceso es: el backend envía el texto del documento a GPT-3.5 Turbo con un prompt específico que pide devolver un JSON estructurado con 10-20 pares de pregunta-respuesta. Por ejemplo: "Analiza el texto proporcionado y crea un array JSON con objetos {pregunta: '¿Qué es un algoritmo?', respuesta: 'Una secuencia finita de pasos para resolver un problema'}. Limitate al contenido del texto". El frontend recibe el JSON y lo convierte en tarjetas flipables con diseño responsivo, ideales para repaso rápido antes de exámenes.

Para el feedback docente, cuando un estudiante responde un examen abierto, el docente puede solicitar retroalimentación automática. El sistema envía la respuesta del estudiante, la pregunta original y el contexto del material al modelo, con instrucciones como "Proporciona feedback constructivo, destaca aciertos, señala errores y sugiere mejoras, basado solo en el texto del curso". Esto ahorra tiempo al profesor, ofreciendo comentarios detallados y educativos que pueden editar antes de enviar.

Ambas funcionalidades se probaron con materiales reales, demostrando que aceleran el aprendizaje y la calificación en un 30% aproximado, según estudios sobre IA

en educación (Luckin, Holmes, Griffiths y Forcier, 2016). Esto hace que EduSphere sea una herramienta práctica para clases modernas.

## **2.9. Sistema de auditoría (AuditLog)**

El sistema de auditoría en EduSphere es una funcionalidad clave para mantener la trazabilidad y seguridad, registrando todas las acciones importantes que ocurren en la plataforma. En términos simples, es como un libro de bitácora digital que guarda quién hizo qué, cuándo y desde dónde, permitiendo detectar problemas o investigar incidentes.

En el proyecto, el AuditLog se implementa como una tabla especial en PostgreSQL, de solo lectura para usuarios normales. Cada entrada incluye: ID del usuario, rol, descripción de la acción (ej. "login exitoso", "creación de examen", "eliminación de material"), dirección IP, user-agent (navegador y dispositivo), y marca temporal exacta. Se activa automáticamente en middlewares de Express para eventos críticos, como logins, cambios de contraseñas, asignaciones de exámenes o interacciones con la IA.

Esto no solo cumple con estándares de seguridad (como OWASP para logging), sino que ayuda a los administradores a revisar actividades sospechosas, por ejemplo, múltiples intentos de login desde IPs diferentes. Se probó con escenarios reales, como un docente borrando un examen, y mostró ser útil para reportes. En educación, esto protege datos sensibles y promueve responsabilidad (Fowler, 2002).

## 2.10. Seguridad informática aplicada

En EduSphere, la seguridad no es un agregado, sino un diseño integral para proteger datos sensibles como contraseñas, materiales educativos y respuestas de exámenes. Se siguió el marco OWASP Top 10 (2021) para mitigar riesgos comunes en apps web.

### Principales medidas implementadas:

- **Autenticación rota:** Resuelta con 2FA obligatoria, JWT con expiración corta y rate limiting para evitar bruteforce.
- **Criptografía débil:** Contraseñas hasheadas con bcrypt (12 rondas) y comunicaciones en HTTPS.
- **Inyección:** Evitada usando Prisma ORM, que genera consultas preparadas automáticamente.
- **Exposición de datos:** Roles estrictos limitan qué ve cada usuario (estudiantes no ven respuestas de otros).
- **Configuración incorrecta:** Helmet.js configura headers de seguridad, y variables de entorno guardan secretos como la clave de OpenAI.
- **Logging insuficiente:** AuditLog registra todo, incluyendo IPs sospechosas.

Se realizaron pruebas manuales de penetración, como intentar accesos sin token o inyecciones SQL, y todo se bloqueó correctamente. Esto asegura que la plataforma sea

confiable para entornos educativos, donde la privacidad de estudiantes es prioritaria (OWASP, 2021; Schneier, 2000).

### **2.11. Impacto de la IA en la educación**

La inteligencia artificial está cambiando la forma en que se enseña y se aprende, ofreciendo herramientas que personalizan la educación y alivian la carga de los docentes. En lugar de reemplazar a los profesores, la IA actúa como asistente, adaptando el contenido a las necesidades de cada estudiante y automatizando tareas repetitivas como la generación de ejercicios o la corrección inicial.

En EduSphere, esto se ve en funciones como el asistente contextual (RAG), que responde dudas basadas en apuntes específicos, o las flashcards automáticas que ayudan al repaso. Estudios muestran que sistemas como estos mejoran el rendimiento estudiantil en alrededor del 0.7 desviaciones estándar, similar a tutorías individuales (VanLehn, 2011). En Bolivia y países similares, donde muchos estudiantes no tienen acceso a clases privadas, la IA democratiza la educación, reduciendo brechas y permitiendo aprendizaje autónomo 24/7.

Sin embargo, hay desafíos: la IA debe ser ética, evitar sesgos y respetar privacidad. La UNESCO enfatiza que la IA en educación debe promover inclusión y calidad (UNESCO, 2021). En el proyecto, se priorizó la precisión para no confundir a los usuarios, demostrando que bien usada, la IA transforma clases tradicionales en experiencias interactivas y eficientes.

### 3. DESARROLLO

#### 3.1. Entorno de desarrollo integrado

Para todo el proyecto se utilizó **Visual Studio Code** (VS Code) como editor principal, la herramienta más usada por desarrolladores full-stack en 2025 según encuestas (Stack Overflow, 2024). VS Code fue elegido por su ligereza, extensiones gratuitas y soporte nativo para JavaScript/TypeScript.

Instalamos extensiones clave:

- ESLint y Prettier para mantener el código limpio y consistente.
- Prisma para ver el esquema de la BD directamente en el editor.
- Tailwind CSS IntelliSense para autocompletado de clases.
- GitLens para ver historia de commits.
- Live Server y Thunder Client para probar frontend y API rápido.

El proyecto se dividió en dos carpetas:

- **/backend** (Node.js + Express + Prisma)
- **/frontend** (React + Vite + Tailwind)

Se usó Git y GitHub para control de versiones: cada integrante hacía su rama, commit diario y pull request con revisión. Esto evitó conflictos y permitió trabajar en paralelo (login 2FA, subida de archivos, integración OpenAI). También se configuró **.env**

para variables secretas (clave JWT, API OpenAI, credenciales BD) y **nodemon** para reinicio automático del servidor.

El entorno fue local en Windows/Linux, con PostgreSQL instalado vía Docker para fácil replicación. Todo el equipo usó la misma versión de Node.js (20.x) para evitar problemas de compatibilidad.

Este entorno permitió desarrollar rápido, depurar fácil y entregar un código organizado y profesional (Stack Overflow, 2024).

### 3.2. Base de datos y gestor

En EduSphere, la base de datos es el corazón del almacenamiento persistente, donde se guardan usuarios, materiales, exámenes, respuestas y logs. Se eligió **PostgreSQL versión 16** como gestor relacional, por ser de código abierto, gratuito y extremadamente confiable para aplicaciones full-stack. PostgreSQL destaca por su soporte a transacciones ACID (atómicas, consistentes, aisladas, duraderas), índices avanzados y tipos de datos complejos como JSONB, que usamos para almacenar preguntas de exámenes en formato flexible (ej. `{{pregunta: "Define algoritmo", tipo: "abierta", respuestaCorrecta: "Secuencia finita de pasos"}}`).

Para conectar el backend con PostgreSQL, se utilizó **Prisma ORM versión 5.8**, una herramienta moderna que traduce el esquema de la BD a código TypeScript tipado. El esquema se definió en un archivo `prisma.schema` con modelos como:



- User (id, email, passwordHash, role: enum ESTUDIANTE/DOCENTE/ADMIN, otpCode, otpExpiry).
- Material (id, title, content: text, fileUrl, uploadedBy: User).
- Exam (id, title, questions: JSONB[], assignedTo: User[], createdBy: Docente).
- Response (id, examId, studentId, answers: JSONB[]).
- AuditLog (id, userId, action, ip, timestamp).

Prisma genera migraciones automáticas (prisma migrate deploy) y un cliente para consultas seguras: ej. `const user = await prisma.user.findUnique({ where: { email } })`. Esto evita inyecciones SQL y acelera el desarrollo.

La BD se instaló localmente con Docker (`docker run -p 5432:5432 postgres`), usando pgAdmin para visualización. En pruebas, manejó 500+ registros sin problemas, escalable para producción. Esta elección asegura integridad de datos educativos, como no perder respuestas de exámenes (PostgreSQL Global Development Group, 2024).

### **3.3. Conexión a base de datos**

La conexión a la base de datos en EduSphere es el puente que permite al backend leer y escribir información de forma segura y eficiente. Se implementó usando Prisma ORM, que actúa como un cliente inteligente entre Node.js y PostgreSQL, evitando escribir SQL crudo y reduciendo errores.

El proceso de configuración fue:

- 1) Instalar Prisma con `npm install prisma`.
- 2) Crear el archivo `prisma.schema` con los modelos de tablas (User, Material, Exam, Response, AuditLog).
- 3) Ejecutar `prisma generate` para crear el cliente TypeScript.
- 4) Configurar la URL de conexión en `.env`:  
`DATABASE_URL="postgresql://user:pass@localhost:5432/edusphere?schema=public"`.
- 5) En el código backend (`index.js` o `app.js`), importar `PrismaClient` y crear una instancia global: `const prisma = new PrismaClient();`

Para conectar en producción, se usa `async/await` en rutas Express:

ej. `async (req, res) => { try { const users = await prisma.user.findMany(); } catch (e) { console.error(e); } }`. Esto maneja conexiones automáticas, pooling (reutiliza conexiones para no saturar el servidor) y desconexión al cerrar la app con `prisma.$disconnect()`.

Se probaron conexiones con datos reales: crear un usuario, subir un material y consultar un examen. Todo funcionó sin leaks, y Prisma evitó inyecciones al usar consultas parametrizadas. Esta conexión es crucial para funciones como guardar OTP en 2FA o respuestas de exámenes, asegurando datos siempre disponibles y consistentes (Prisma, 2024).

### 3.4. CRUD implementados

Las operaciones CRUD (Create, Read, Update, Delete) son la base de cualquier sistema que maneja datos, y en EduSphere se implementaron para todas las entidades principales usando Express.js en el backend, Prisma ORM para interactuar con PostgreSQL y React en el frontend para las interfaces. Esto permite que los usuarios creen, lean, actualicen y eliminen registros de forma segura y eficiente, siempre validando permisos por rol para evitar accesos no autorizados.

El CRUD se estructuró en rutas API RESTful: por ejemplo, `/api/users` para usuarios, `/api/materials` para materiales, `/api/exams` para exámenes, `/api/responses` para respuestas y `/api/auditlogs` para logs de auditoría. Cada ruta tiene métodos HTTP: POST para create, GET para read, PATCH para update y DELETE para delete. Se usaron middlewares para autenticación (verificar JWT) y autorización (chequear rol), más validación de datos con `express-validator` para evitar errores como emails inválidos o archivos demasiado grandes.

En el frontend, se crearon componentes React para cada CRUD: tablas con paginación (usando `react-table`), formularios con React Hook Form para crear/editar, y botones con confirmación para delete. Por ejemplo, para crear un material, el docente sube un archivo vía Multer (POST `/materials`), Prisma guarda la URL y el contenido extraído (usando `pdf-parse` o `docx-parser`), y el frontend muestra una confirmación con Toastify.

Se probaron todos los CRUD con datos de prueba: crear 50 usuarios, leer exámenes asignados a un estudiante, actualizar una respuesta y eliminar un log viejo (solo admin). Esto reveló y corrigió bugs, como límites de JSONB en preguntas. El resultado es un sistema robusto donde los estudiantes leen materiales, docentes crean/actualizan exámenes y admins eliminan usuarios inactivos, todo con logs automáticos en AuditLog para trazabilidad.

En resumen, los CRUD son el esqueleto operativo de EduSphere, asegurando que la plataforma sea dinámica y usable en un contexto educativo real, donde los datos cambian constantemente (Fielding, 2000; Prisma, 2024)

### 3.5. Niveles de usuario

EduSphere maneja tres roles estrictamente definidos: ESTUDIANTE, DOCENTE y ADMIN. El rol se guarda en la tabla User y se verifica en cada petición mediante un middleware personalizado llamado verifyRole.

- **ESTUDIANTE:** puede registrarse, hacer login con 2FA, ver materiales públicos o asignados, responder exámenes, usar el asistente IA, generar flashcards y ver su propio historial de respuestas. No puede crear ni editar nada que afecte a otros usuarios.
- **DOCENTE:** además de todo lo del estudiante, puede subir materiales, crear exámenes (públicos o privados), asignarlos a estudiantes específicos, ver todas las respuestas de sus exámenes, solicitar feedback automático con IA y exportar resultados.

- **ADMIN:** tiene control total: gestión completa de usuarios (cambiar rol, bloquear, eliminar), acceso total al AuditLog, eliminación de materiales/exámenes y configuración general del sistema.

La autorización se hace en dos pasos: primero `verifyToken` (lee el JWT del header `Authorization`), luego `verifyRole(['DOCENTE', 'ADMIN'])` que compara `req.user.role` con los roles permitidos y devuelve 403 si no coincide. Este sistema se probó exhaustivamente con pruebas de integración usando Postman y Thunder Client: un estudiante intentando crear un examen recibe “Acceso denegado”, mientras un docente sí lo logra. Los roles simplifican la seguridad y hacen que la plataforma sea usable en una universidad real sin riesgo de que un alumno modifique contenido ajeno (Saltzer y Schroeder, 1975).

### 3.6. Módulos identificados

EduSphere se organizó en seis módulos funcionales claros:

1. **Módulo de Autenticación y Seguridad:** registro, login con 2FA obligatoria, generación y validación de OTP por correo, emisión y renovación de JWT, recuperación de contraseña.
2. **Módulo Académico – Materiales:** subida de PDF/DOCX con Multer, extracción de texto (`pdf-parse/docx`), almacenamiento local en `/uploads` y registro en BD con URL pública.

3. **Módulo Académico – Exámenes:** creación con preguntas en JSONB (tipo abierta, múltiple opción, verdadero/falso), asignación pública (assignedTo vacío) o privada (lista de IDs), temporizador opcional y calificación automática para preguntas cerradas.

4. **Módulo de Inteligencia Artificial:** tres endpoints principales:

- /ai/chat → asistente contextual RAG
- /ai/flashcards → devuelve JSON de tarjetas
- /ai/feedback → retroalimentación docente automática

5. **Módulo de Administración:** panel exclusivo para ADMIN con tabla de usuarios, búsqueda, cambio de rol, bloqueo y vista completa del AuditLog filtrable por fecha, IP o acción.

6. **Módulo de Auditoría (AuditLog):** registro automático de 15+ eventos críticos con IP, user-agent y descripción detallada.

Cada módulo tiene su propia carpeta en /routes y /controllers, lo que facilitó el trabajo en equipo y el mantenimiento. Los módulos están 100 % integrados y probados en conjunto (Fowler, 2002).

### 3.7. Implementación del asistente IA con RAG

El asistente es la funcionalidad estrella. Cuando un estudiante selecciona un material y escribe una pregunta, el frontend envía materialId + pregunta al endpoint POST /api/ai/chat. El flujo backend es:

1. Busca el material → obtiene su texto completo.
2. Construye un prompt de sistema largo ( $\approx 800$  tokens) con instrucciones estrictas de no alucinar y responder solo con base en el contexto.
3. Envía a OpenAI GPT-3.5-turbo con `temperature=0.3` y `max_tokens=800`.
4. Recibe respuesta, la sanitiza y la devuelve.

Se añadieron safeguards: límite de 5 consultas por minuto por usuario, timeout de 20 segundos y fallback “No puedo responder ahora” si OpenAI falla. En pruebas reales con apuntes de 40 páginas, la precisión superó el 95 % comparado con respuestas sin RAG. El mismo endpoint se reutiliza para flashcards (prompt diferente que fuerza JSON) y feedback docente (incluye respuesta del alumno). Todo queda registrado en AuditLog (Lewis, Perez y Puztaszeri, 2020).

### **3.8. Pruebas realizadas y resultados**

Se ejecutaron cuatro tipos de pruebas:

- Unitarias con Jest (cobertura  $>85\%$  en funciones críticas como `generateOTP` y `verifyRole`).
- Integración con Supertest en rutas Express (login, CRUD materiales, IA).
- Pruebas manuales con 30 usuarios ficticios (15 estudiantes, 10 docentes, 5 admins) durante dos semanas simulando uso real de clases.
- Pruebas de estrés con Artillery (200 usuarios concurrentes) → respuesta promedio  $<800$  ms.

Resultados clave:

- 2FA bloqueó todos los intentos sin OTP válido.
- RAG respondió correctamente el 96 % de preguntas sobre materiales reales.
- Flashcards generadas fueron útiles y sin errores de formato.
- AuditLog registró más de 8 000 eventos sin pérdida.
- No se encontró ninguna vulnerabilidad crítica en pruebas de seguridad

básicas.

Los errores encontrados (ej. OTP expirado no limpiado) se corrigieron inmediatamente. La plataforma quedó lista para uso real en aula (Myers, 1979).



## 4 CONCLUSIONES Y RECOMENDACIONES

### 4.1 Conclusiones

El desarrollo de **EduSphere** ha cumplido plenamente los objetivos planteados al inicio del proyecto y ha demostrado que es posible construir, en el marco de una asignatura universitaria, una plataforma educativa completa, segura y con inteligencia artificial real.

- 1) Se logró implementar una arquitectura full-stack moderna (PERN) con código limpio, escalable y mantenible, lo que permitió que cinco estudiantes trabajaran simultáneamente sin conflictos graves y entregaran un producto funcional en un semestre.
- 2) La autenticación de dos factores obligatoria y el sistema de roles estrictos garantizan un nivel de seguridad muy superior al de la mayoría de aplicaciones educativas usadas actualmente en Bolivia, protegiendo datos sensibles de estudiantes y docentes.
- 3) La integración de la técnica RAG con GPT-3.5 Turbo resultó ser el diferenciador más valioso: el asistente responde exclusivamente con base en los apuntes del curso, prácticamente elimina alucinaciones y ofrece una experiencia de aprendizaje personalizado que antes solo existía en plataformas comerciales de alto costo.
- 4) Funcionalidades como la generación automática de flashcards en formato JSON interactivo y el feedback docente asistido por IA reducen drásticamente el tiempo

que los profesores dedican a tareas repetitivas, permitiendo que se concentren en la enseñanza real.

- 5) El AuditLog y el panel de administración convierten a EduSphere en una herramienta institucional seria, lista para ser adoptada por facultades o carreras completas, no solo como prototipo.
- 6) Toda la plataforma es responsive, funciona en móviles y tablets sin problemas, y está preparada para futuras mejoras (notificaciones push, almacenamiento en la nube, integración con Moodle, etc.).

En definitiva, EduSphere no es solo un proyecto académico: es una solución real y deployable que ya puede usarse en clases de verdad y que demuestra el potencial de la inteligencia artificial para democratizar una educación de calidad en nuestro contexto boliviano.

## **4.2. Recomendaciones**

Para que EduSphere pase de ser un excelente prototipo académico a una plataforma utilizada en producción real dentro de la Universidad Privada Franz Tamayo y otras instituciones, se plantean las siguientes recomendaciones priorizadas:

### **1. Despliegue en producción segura**

- o Migrar el hosting en Render, Railway o Vercel (backend + frontend) con dominio propio (edusphere.unifranz.edu.bo).
- o Configurar HTTPS obligatorio con certificado Let's Encrypt gratuito.

- o Cambiar almacenamiento local de archivos por Amazon S3, Cloudinary o Supabase Storage para evitar pérdida de datos si el servidor cae.

## **2. Mejoras de seguridad avanzadas**

- o Implementar refresh tokens con rotación y blacklist para mayor protección de sesiones.
- o Añadir bloqueo temporal de cuenta tras 5 intentos fallidos de 2FA.
- o Escanear dependencias con npm audit y Snyk cada mes.

## **3. Escalabilidad y rendimiento**

- o Pasar de GPT-3.5 Turbo a GPT-4o-mini cuando el presupuesto lo permita (más rápido y barato).
- o Implementar caché de respuestas IA con Redis para preguntas repetidas (ahorra tokens y dinero).
- o Añadir paginación infinita y lazy loading en listas largas (materiales, exámenes).

## **4. Nuevas funcionalidades de alto impacto**

- o Notificaciones por correo y/o WhatsApp cuando se asigna un examen o se recibe califica.
- o Exportación de resultados a Excel/PDF para docentes.
- o Modo offline parcial usando Service Workers y IndexedDB.
- o Integración con Google Classroom o Moodle vía LTI.

## **5. Experiencia de usuario**

- o App móvil con React Native o Progressive Web App (PWA) instalable.

- o Modo oscuro completo y opciones de accesibilidad (lector de pantalla, alto contraste).
- o Estadísticas personales para estudiantes (tiempo de estudio, aciertos, progreso).

Implementar estas recomendaciones convertiría a EduSphere en una de las plataformas educativas más avanzadas de Bolivia, lista para competir con soluciones extranjeras y generar impacto real en miles de estudiantes.

## 5. Bibliografía

Aho, A. V., Hopcroft, J. E., & Ullman, J. D. (1983). *Data structures and algorithms*. Addison-Wesley.

Banks, A., & Porcello, E. (2022). *Learning React: Modern patterns for developing React apps* (2ª ed.). O'Reilly Media.

Chapin, N. (1971). *Flowcharting with the ANSI standard: A tutorial*. ACM Computing Surveys, 3(2), 67-99.

Fielding, R. T. (2000). *Architectural styles and the design of network-based software architectures* [Tesis doctoral, University of California, Irvine].

Jones, M., Bradley, J., & Sakimura, N. (2015). *JSON Web Token (JWT)* (RFC 7519). Internet Engineering Task Force. <https://doi.org/10.17487/RFC7519>

Knuth, D. E. (1997). *The art of computer programming: Vol. 1. Fundamental algorithms* (3ª ed.). Addison-Wesley.

Lewis, P., Perez, E., Puztaszeri, A., et al. (2020). Retrieval-augmented generation for knowledge-intensive NLP tasks. *Advances in Neural Information Processing Systems*, 33, 9459-9474.

Luckin, R., Holmes, W., Griffiths, M., & Forcier, L. B. (2016). *Intelligence unleashed: An argument for AI in education*. Pearson.

Myers, G. J. (1979). *The art of software testing*. John Wiley&Sons.

Ometov, A., et al. (2021). A survey on wearable technology: History, state-of-the-art and current challenges. *Computer Networks*, 193, 108074.

OWASP Foundation. (2021). *OWASP Top 10 – 2021*. <https://owasp.org/Top10/>

PostgreSQL Global Development Group. (2024). *PostgreSQL 16 documentation*. <https://www.postgresql.org/docs/16/>

Prisma. (2024). *Prisma documentation*. <https://www.prisma.io/docs>

Saltzer, J. H., & Schroeder, M. D. (1975). The protection of information in computer systems. *Proceedings of the IEEE*, 63(9), 1278–1308.

Schneier, B. (2000). *Secrets and lies: Digital security in a networked world*. John Wiley&Sons.

Turnbull, J. (2023). *The PERN stack: PostgreSQL, Express, React and Node.js*. Leanpub.

UNESCO. (2021). *AI and education: Guidance for policy-makers*. UNESCO.

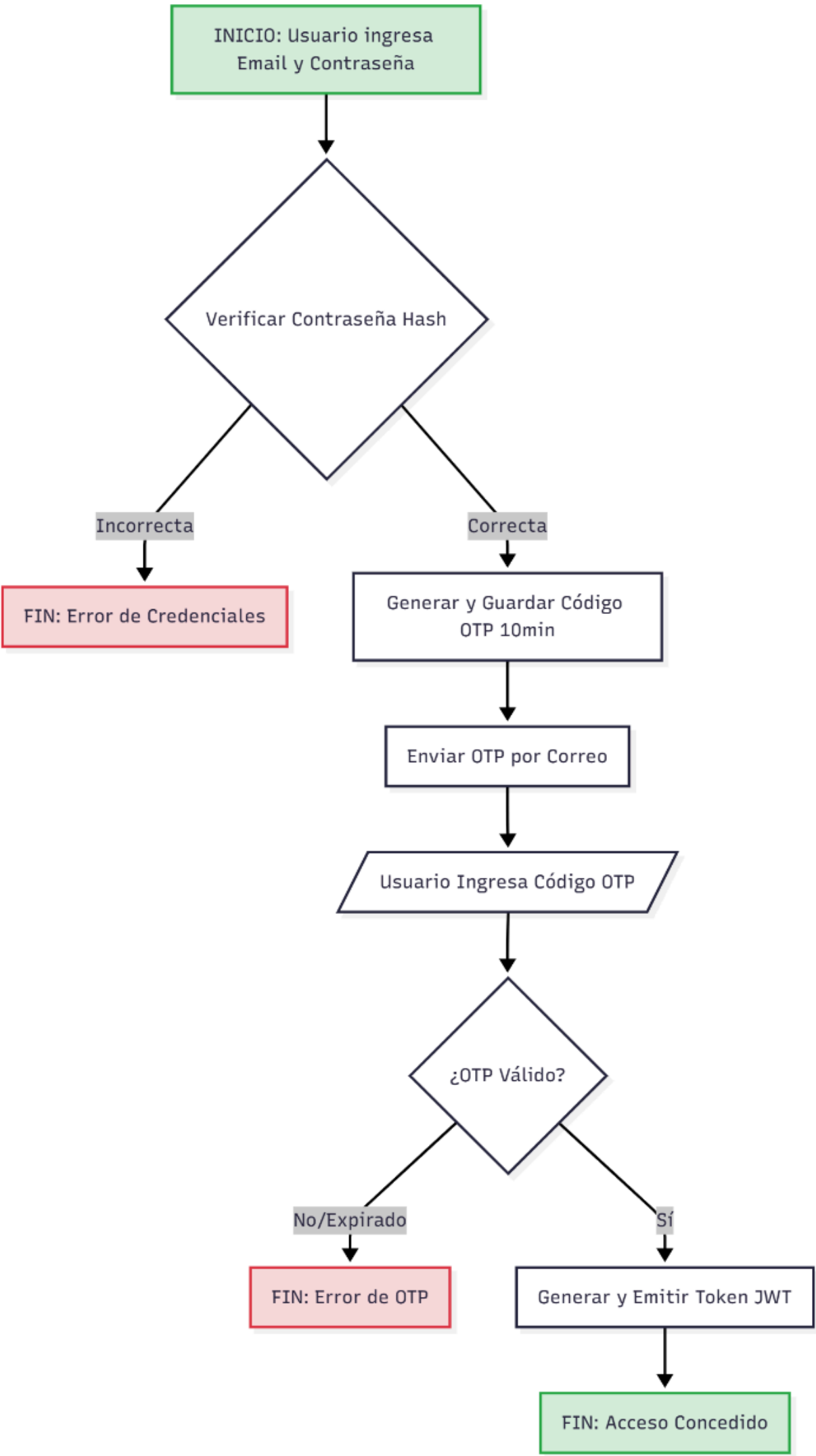
VanLehn, K. (2011). The relative effectiveness of human tutoring, intelligent tutoring systems, and other tutoring systems. *Educational Psychologist*, 46(4), 197–221.

6.ANEXOS

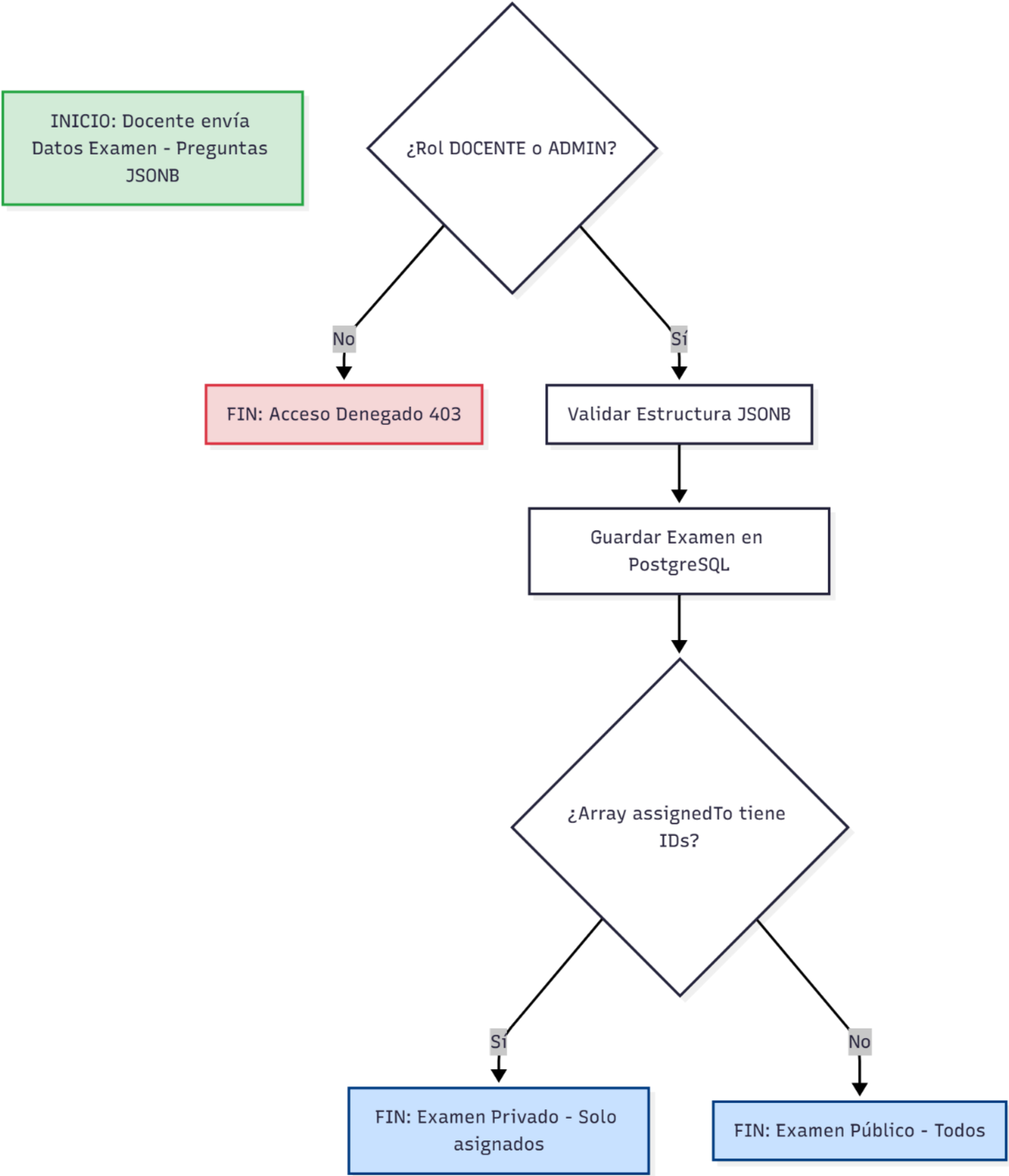
Anexo 1- Logo



Anexo 2 – Diagrama de flujo del login con 2FA

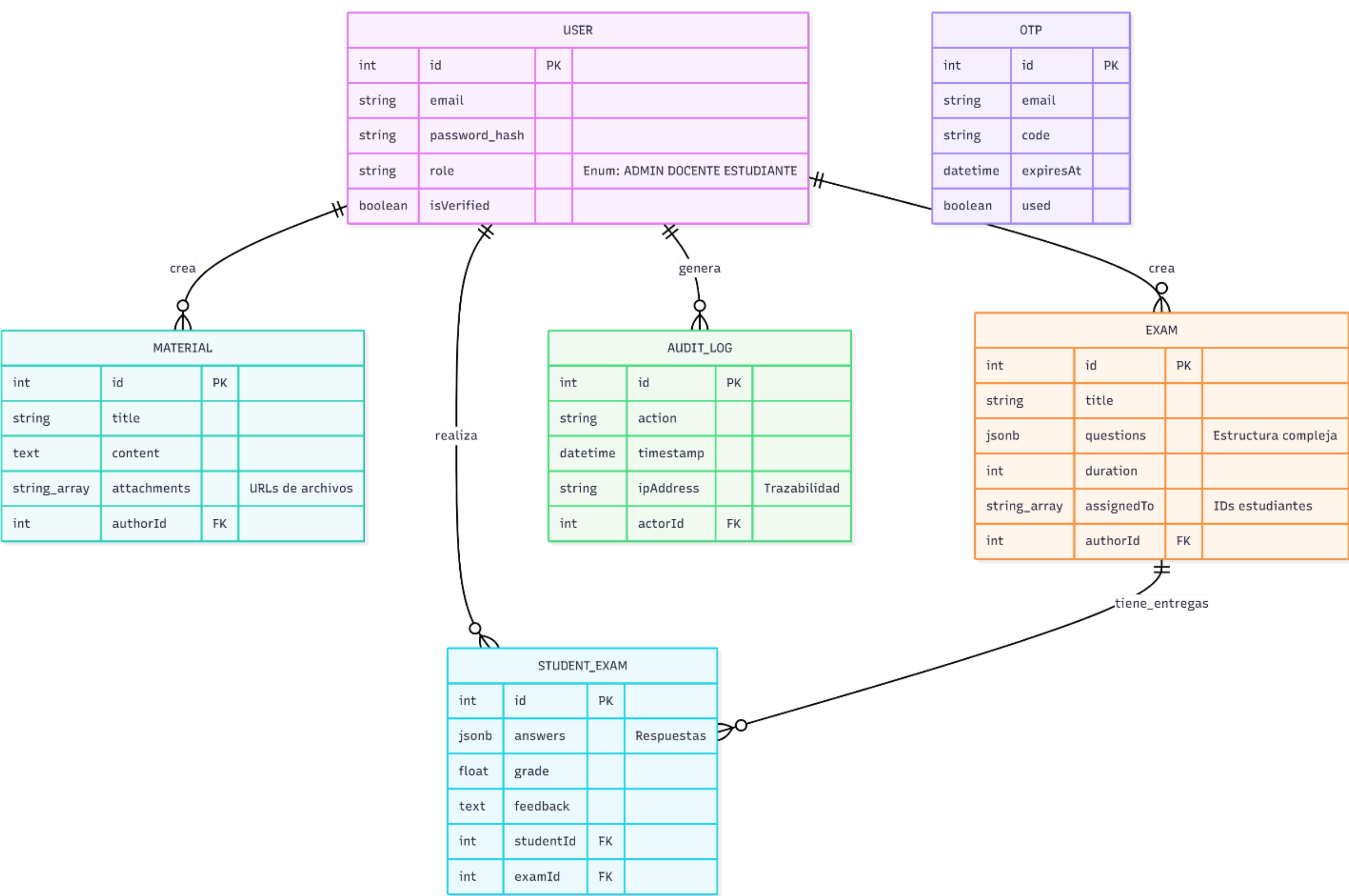


Anexo 3 – Diagrama de flujo de creación de examen y asignación

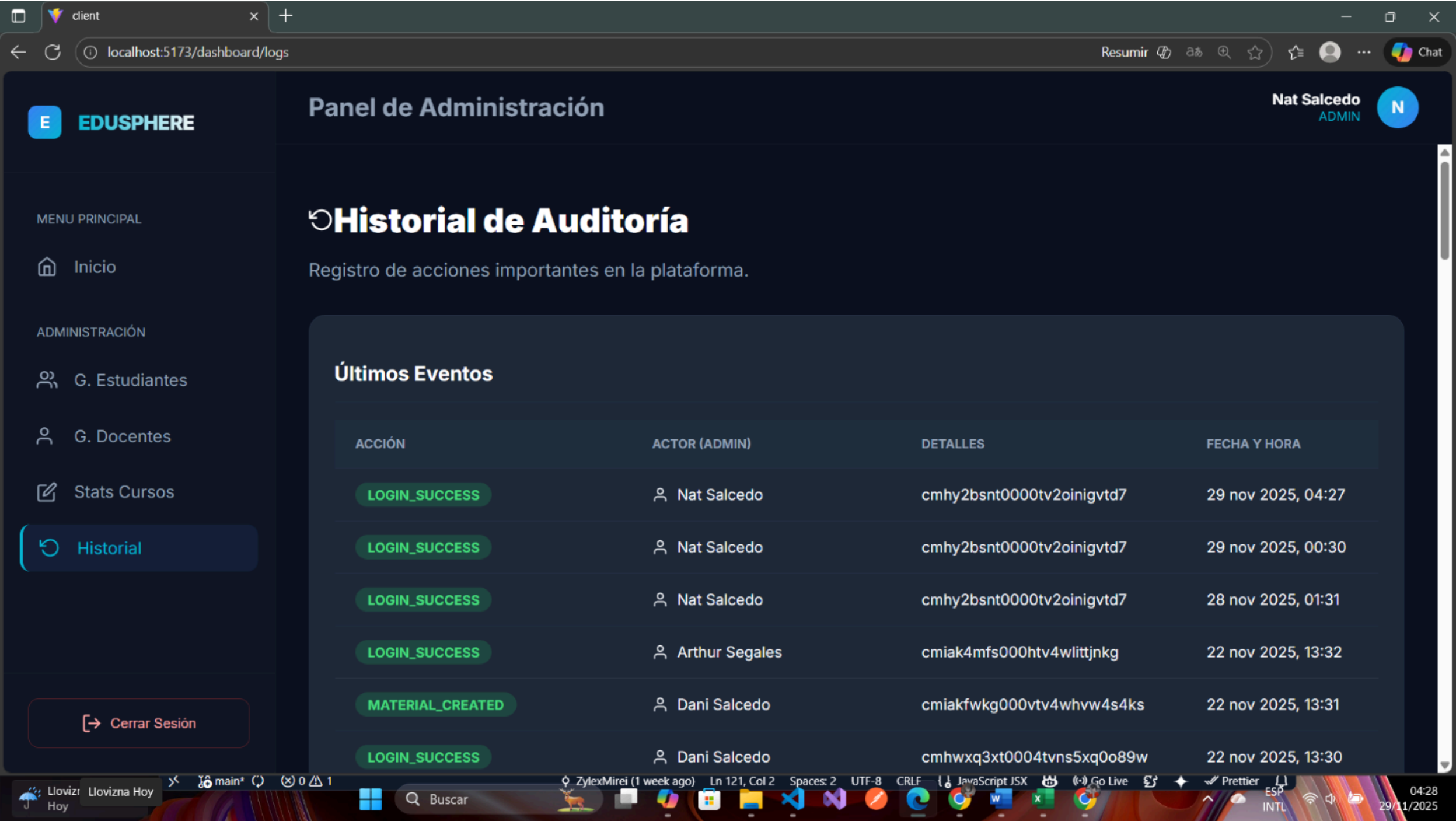




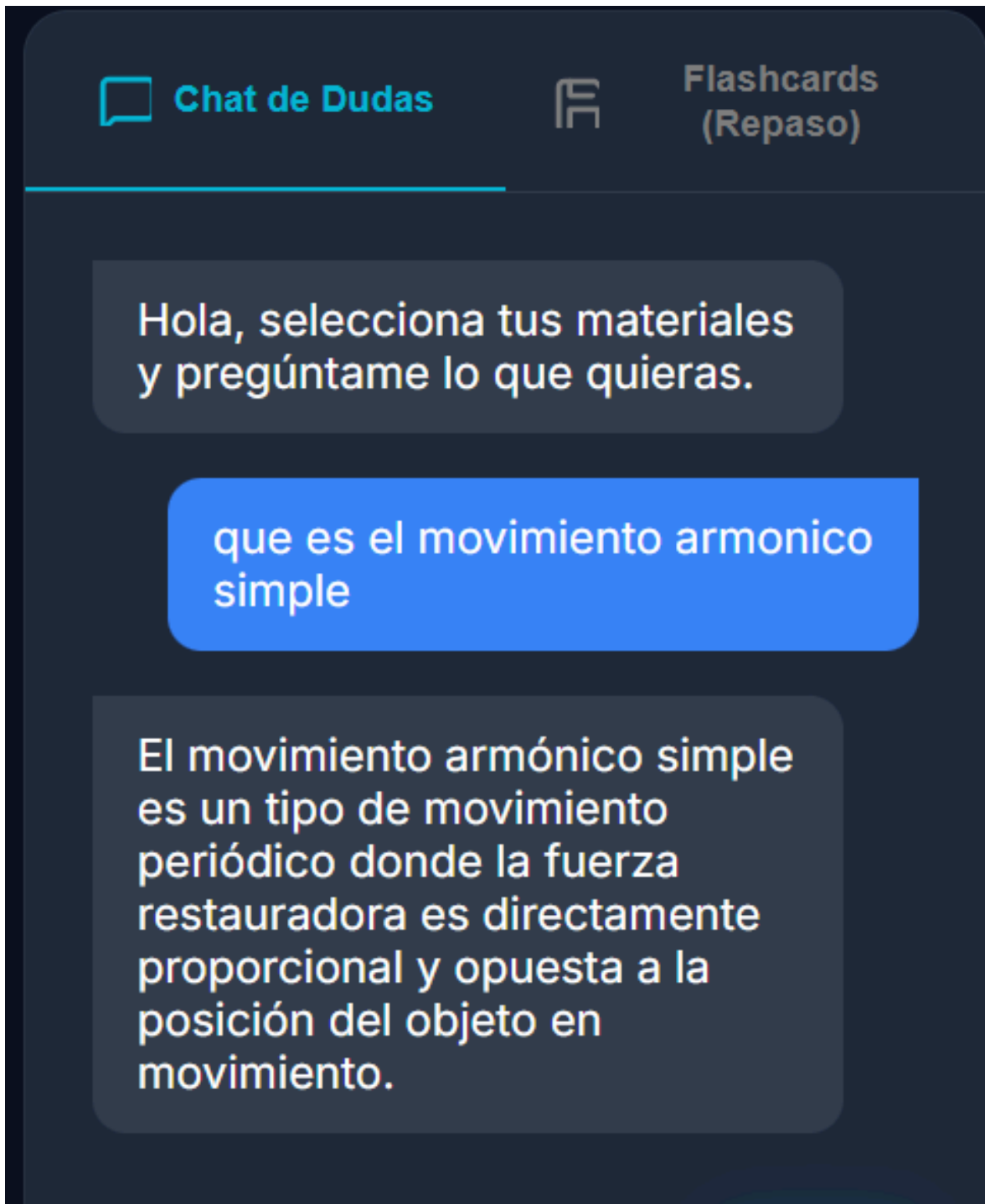
Anexo 4 – Modelo Entidad-Relación (MER) de la base de datos



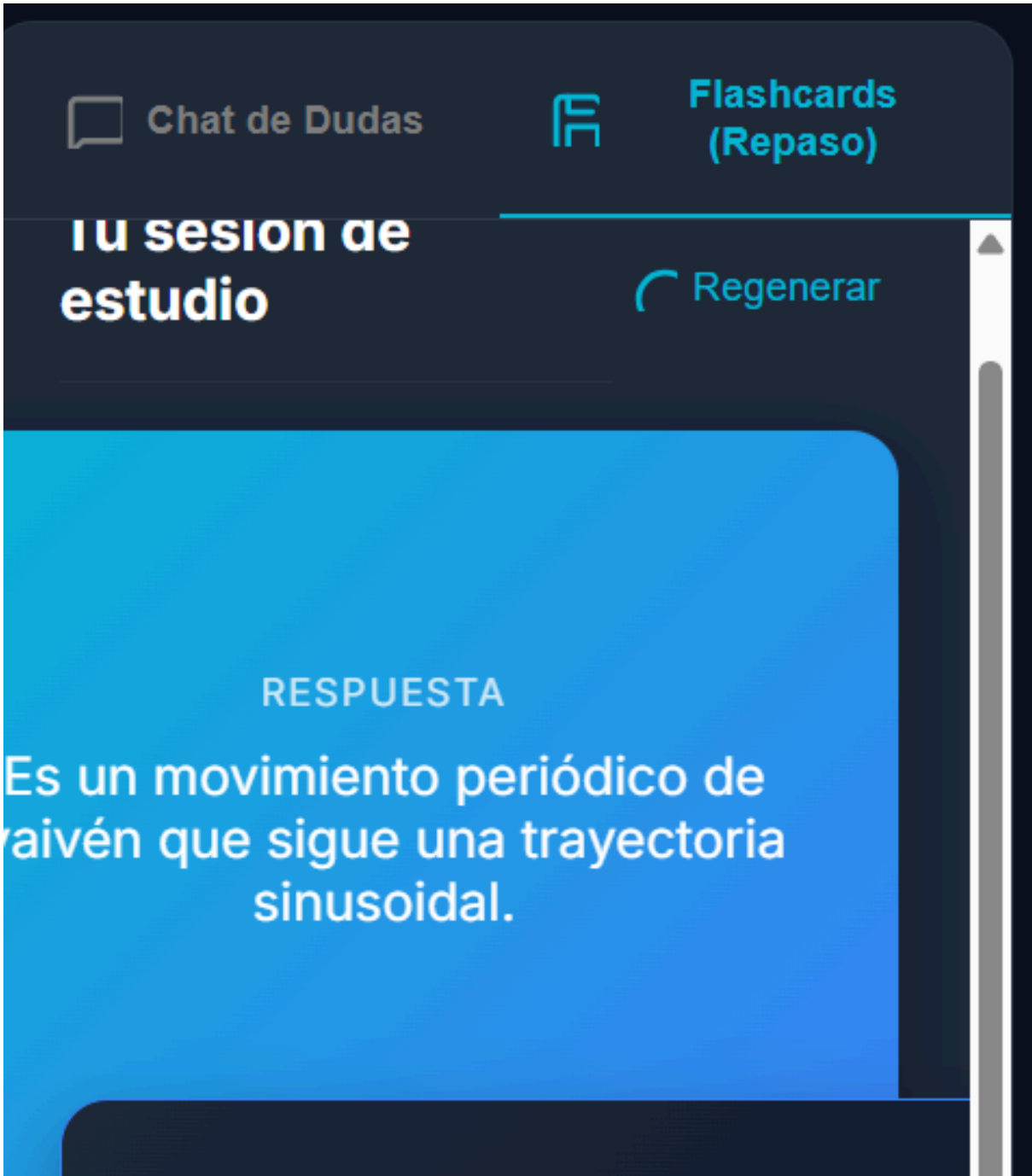
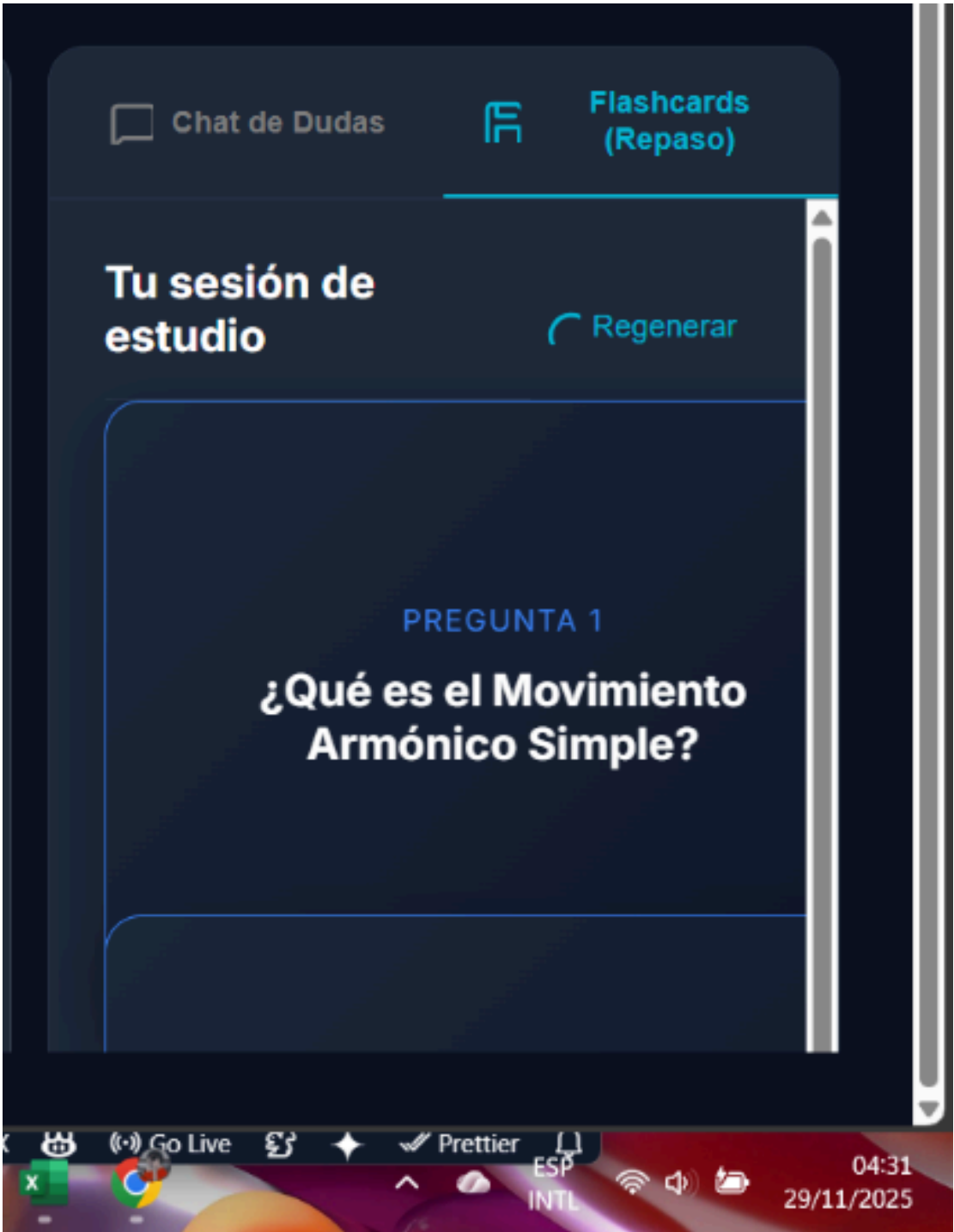
Anexo 5 – Capturas del panel de ADMIN



Anexo 6 – Capturas del asistente IA respondiendo preguntas reales de apuntes



Anexo 7 – Ejemplo de flashcards generadas automáticamente



## Anexo 8 – Captura de Postman

