TRIAL AI PROJECT

José Juan Arellano Juárez

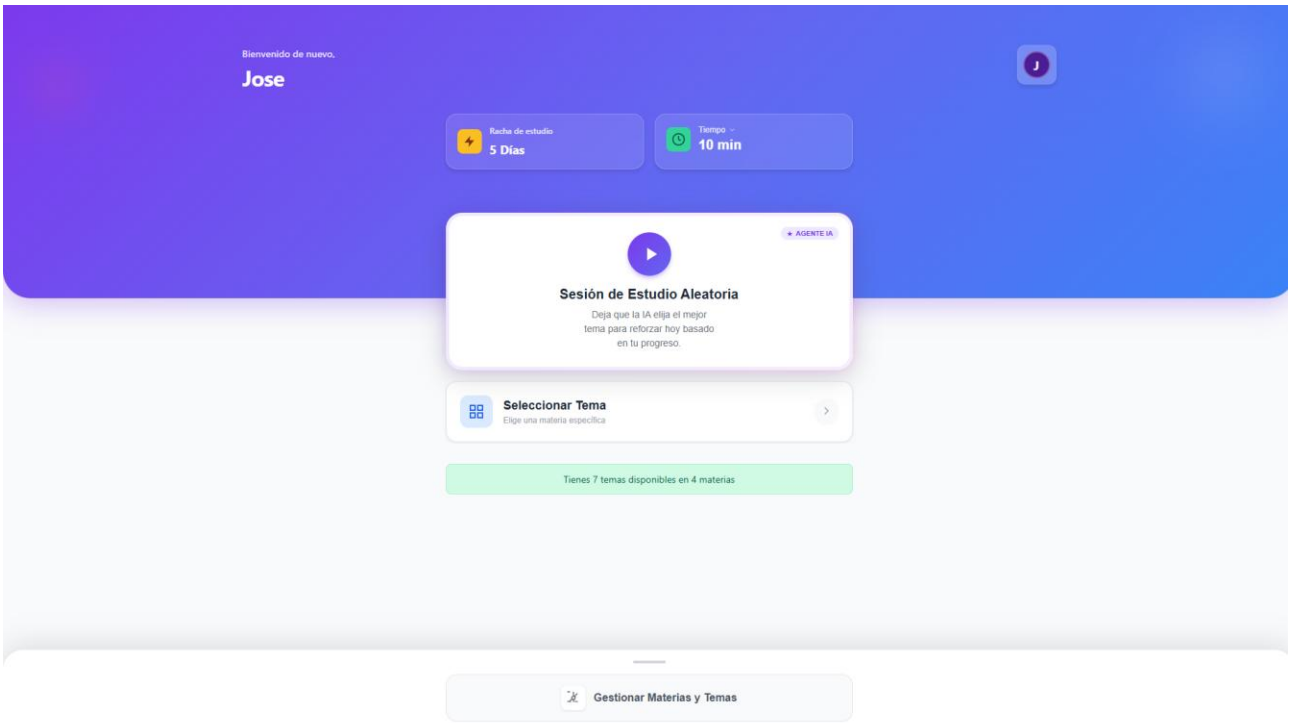
StudySprint
Tu asistente de estudio

PROBLEM STATEMENT

Students managing multiple subjects must manually process large volumes of academic material such as papers, textbooks, and PDF notes, often without enough time to organize them effectively. This repetitive process of identifying key concepts and structuring study materials leads to inefficient study sessions, low information retention, and poor use of limited time. The problem is intensified by fragmented study windows, such as commuting or short breaks, where students struggle to decide what to study next. Reducing this manual planning and cognitive overhead is essential to help students study more efficiently and consistently.

PROTOTYPE

Small app that demonstrates the AI agent



WRITE-UP

Problem Approach

Initial Analysis

The problem was approached by identifying a core inefficiency in students' workflows. To address this, the solution focused on three main aspects: automating content extraction, adapting generated content to short study sessions (5–30 minutes), and maintaining engagement while verifying comprehension.

Architecture Design

A full-stack architecture with clear separation of concerns was designed:

- **Backend (Python / FastAPI)** for AI orchestration, PDF processing, and data persistence.
- **Frontend (React)** providing a simple, mobile-first interface with a real-time session timer.
- **AI Service Layer** integrating OpenAI GPT-4o-mini for adaptive content generation.
- **Database (SQLite)** storing subjects, topics, documents, and study sessions.

Key Design Decisions

Study content is generated dynamically based on session duration using a reading-speed heuristic to control length. A lightweight visual countdown timer supports focus without being intrusive. All AI-generated output follows a fixed structure (learning objective, content, key concepts) to improve readability and enable consistent presentation.

AI Tools in Development

A. GitHub Copilot (Throughout Development)

GitHub Copilot was used throughout development to accelerate implementation and improve code quality. It assisted with generating boilerplate code for FastAPI endpoints, React components, and database interactions. Copilot was also used during refactoring to simplify code structure and maintain readability.

B. OpenAI GPT-4o-mini (Core Feature)

The OpenAI GPT-4o-mini model was used as the core AI component for dynamic educational content generation. Prompt engineering was applied to adapt content length and depth to different session durations, ensuring that generated material fit realistic study time constraints. Prompts enforced a structured output format with clearly defined sections to maintain consistency and readability across sessions.

C. Claude / ChatGPT (Development Assistance)

Claude and ChatGPT were used as development support tools to debug complex frontend and backend issues, validate architectural decisions, and design supporting mechanisms

such as retry logic and a lightweight post-processing pipeline for cleaning and structuring AI-generated outputs.

Challenges faced and solutions

Challenge 1: Adapting Content Length to Time and Cost Constraints

A key challenge was controlling the amount of content generated by the language model according to session duration, since excessive token usage increases cost and harms usability in short sessions.
Solution: A duration-based generation strategy was implemented, estimating content length using an average reading speed and dynamically adjusting token limits. Prompts were conditioned on session duration to balance content relevance, time constraints, and API cost.

Challenge 2: Preserving Readability During Post-Processing

Early text-cleaning steps unintentionally removed paragraph structure, resulting in large, unstructured text blocks that reduced readability.
Solution: The post-processing pipeline was adjusted to preserve line breaks and paragraph spacing while still removing unwanted formatting artifacts.

Challenge 3: Maintaining User Attention and Motivation

Text-only study sessions risked becoming monotonous, particularly during short or distracted study windows.
Solution: A lightweight visual countdown timer with color transitions was added to reinforce focus, along with immediate quiz performance feedback to support motivation and learning awareness.

DEMO VIDEO

Link: **José Arellano - Trial AI Project - Study Sprint**