<u>AI-Powered Chat Interface for Querying PDF Documents Using Vector Similarity Search</u>

Objective: Develop a web-based chat interface that enables users to query content extracted from PDF documents. The system should use AI to generate contextual embeddings for text segments and employ vector similarity search to find the most relevant responses. The interface will be developed using Next.js 14.

> Note that many of the tools and libraries used in this task there are Python and TS libraries

- Always use TS.

Part 1: AI Model Development and Vector Similarity Search

- PDF Content Extraction:
    - Utilize a robust PDF parsing library to extract text from documents, ensuring clean and pre-processed output for accuracy in content retrieval.

- Embedding Generation:
    - Employ a tool like Langchain for prompt generation, creating contextual embeddings for text segments. Fine-tune this model on your PDF data to ensure high relevance and contextuality in embeddings.

- Vector Database Integration:
    - Leverage Pinecone (or a similar vector database) to store embeddings, implementing vector similarity search for efficient and relevant text segment retrieval in response to queries.

- API Development:
    - Design a RESTful API that interfaces with the vector database, fetching and serving text segments from the PDFs based on user queries with robust error handling.

Part 2: Chat Interface Development Using Next.js 13

- Chat UI Development:
    - Create a responsive chat interface using Next.js 13, enabling real-time user queries and displaying API responses in a conversational format.

- Integration with AI API:

    - Ensure seamless communication between the chat interface and the AI backend, displaying relevant PDF text segments as chat responses.

- User Feedback Integration:
    - Incorporate a mechanism for users to provide feedback on response relevancy and accuracy, which can be used for future improvements.

Part 3: Continuous Improvement and Learning

- Model Refinement:
    - Utilize user feedback and interaction data to refine the embedding model, adopting an iterative improvement approach for enhanced accuracy over time.

- Chat UI Enhancement:
    - Continuously explore Next.js 13 features to enhance the chat UI, focusing on user engagement, accessibility, and convenience features like voice input.

Deliverables:

- A Github repo
- A Loom.com video explaining what you built with Camera and Audio Voiceover
- Chat Web Application: A fully functional chat interface built with Next.js 13, including user documentation and challenges faced. - Video
- Project Report: Detailed documentation covering the project's methodology, technology stack, encountered challenges, and key learnings in the Github Repo Readme

Send an email to pradyu@professiorai.co and abhay@professorai.co with the deliverables above.