E-Exam Generator App (using RAG & agentic framework)

A. Project Overview

- An AI-powered app for generating multiple-choice questions (MCQs) and short questions by leveraging multiple Crew AI agents, powered by a PDF RAG Search Agent and user interface.
- It is a highly relevant and impactful project, given the growing demand for automated content creation in education, training, and assessment.

B. Rationale & Market Relevance

- Automation of content creation: Save significant time of educators, trainers, and content creators for crafting high-quality questions for assessments, quizzes, and practice tests.
- Modularity & Scalability: Each agent handles a specific task, making the system easy to maintain and extend. We can add more agents (e.g., for validation, difficulty level adjustment, etc.) as our app grows.
- User-Friendly: Users can upload PDFs and download generated content seamlessly.
- Cost Efficiency: Automating question generation reduces the need for hiring subject matter experts.
- **Existing Solutions**: -
  Tools like Quizlet, Kahoot, and Question mark offer question creation features, but few leverage AI for automated generation.
- **Differentiation: -**
  By using Crew AI agents, the app can offer advanced customization, validation, and scalability, setting it apart from competitors.

C. Methodology

1. **User Input:** The user uploads a pdf.

2. **PDF RAG Search Agent:** The PDF RAG search agent retrieves relevant sections from the PDF. The text is extracted, embedded, and stored in a vector database.

3. **Query Analysis Agent:** Determines the type of query (e.g., MCQ generation, short question).

4. **MCQ Generation Agent:** Generates multiple-choice questions based on the content.

5. **Short Question Generation Agent:** Generates short questions based on the content.

6. **Manager Agent:** Manages the flow of data between agents. Ensures the final output is presented to the user in a structured format.

7. **Download Agent:** The download agent formats the content and provides a downloadable file to the user.

Tools & Technologies: -

- **Crew AI**: For building and managing the multi-agent system.
- **Lang Chain or Llama Index**: For integrating RAG with PDF processing and retrieval.
- **Vector Database**: Pinecone, or other for storing and retrieving PDF embeddings.
- **Generative Model**: GPT-4 or similar for generating MCQs and short questions.
- **Backend Framework**: Fast API for handling file uploads and downloads.
- **Frontend**: Stream lit frontend framework for user interaction.

D. Workflow

1. **Set Up the PDF RAG Search Agent:**
   - Generate embeddings for the chunks and store them in a vector database.
   - Implement a retrieval mechanism to fetch relevant chunks based on user queries.
2. **Create the MCQ Generation Agent:**
   - Use an LLM (e.g., GPT-4) to generate MCQs from the retrieved content.
   - Example prompt: "Generate 5 MCQs based on the retrieved text".
3. **Develop the Short Question Agent:**
   - Use an LLM to generate short questions.
   - Example prompt: " Generate 5 short questions based on the retrieved text ".
4. **Integrate Crew AI:**
   - Define the roles and tasks of each agent.
5. **Download Output Agent:**
   - Format the generated content: Organize the MCQs and short questions into a user-friendly format (e.g., JSON, CSV, or PDF).
   - Provide download functionality: Allow users to download the generated content in their preferred format.
6. **Build the User Interface:**
   - Create a simple UI (e.g., web app) for users to upload pdf, input queries and receive outputs in the form of a pdf file to download.
