EduMentorAI

Appathon CSE-EJUST Challenge — Summer 2025

Zyad Tarik Awad Omar	120210275
Eyad Amgad Mostafa Mahmoud	120210276
Yousif Ibrahim Masoud	120210281
Hesham Ahmed Yousef Ebaid	120210064
Ahmed Mahmoud Abdelazim	120210274

Submitted to

Prof. Rami Zewail

1. Chosen Theme and Specific Problem Statement

Theme: Educational Technology (EdTech)
Problem Statement: EduMentorAI is an AI-powered educational platform designed to enhance university learning by integrating a Retrieval-Augmented Generation (RAG) chatbot with personalized study support features. The platform allows students to upload lecture materials—including notes, slides, and textbooks—which are processed to build a dynamic knowledge base. Leveraging the power of RAG, the chatbot provides accurate, context-aware answers to students' academic questions, mimicking a knowledgeable study companion.

Additionally, EduMentorAI automatically generates practice questions and quizzes tailored to the uploaded content, enabling students to reinforce understanding and prepare for exams more effectively. The platform not only promotes self-paced, active learning, but also bridges gaps in comprehension, especially in large or asynchronous university courses. EduMentorAI empowers students with smarter study tools, personalized feedback, and accessible academic support—anytime, anywhere.

2. Proposed Solution Overview

EduMentorAI will be developed as a cloud-hosted web platform built using Django. It integrates modern natural language processing techniques with a semantic search and retrieval mechanism to deliver a highly interactive learning environment.

Detailed Solution Components

1. **Lecture Material Upload:** Students can upload lecture notes, PDF slides, or textbook excerpts. Uploaded content will be processed using text extraction and embedding generation for semantic search.

2. **Knowledge Base Creation:** Extracted content is stored in a vector database (FAISS) with contextual metadata, allowing fast and relevant retrieval during queries.

3. **RAG Chatbot:** The chatbot combines semantic search results with a generative model (e.g., OpenAI GPT/OSS) to deliver precise, context-aware answers to student questions.

4. **Automated Quiz Generation:** Using NLP-based question generation models, the system creates quizzes—multiple choice, short answer, and true/false—directly from the uploaded content.

5. **Quiz Practice & Feedback:** Students can take the quizzes and receive instant scoring along with detailed explanations, reinforcing key concepts.

6. **Scalability and Accessibility:** Built with responsive web design and cloud hosting to support students on any device, from anywhere.

This approach ensures that learning is not just passive consumption but an active engagement process. Students receive instant, tailored support without waiting for instructor feedback, making the platform ideal for both in-person and online learning environments.

3. Key Features and User Flows

Key Features

- **Lecture Upload** – Add lecture materials in various formats.
- **Content Understanding** – AI parses and indexes material for fast retrieval.
- **Interactive Chatbot** – Context-aware Q&A for personalized assistance.
- **Quiz Generation** – Automatically generated quizzes based on uploaded content.
- **Quiz Practice** – Interactive quizzes with immediate feedback.
- **Slide Genrator** – Generate Slides using interactive chatbot.

User Flow Diagram

The diagram will depict a student's interaction with EduMentorAI, starting from login to quiz review. Steps include:

1. Login / Registration
2. Upload Lecture Materials
3. AI Processing & Knowledge Base Creation
4. Chatbot Q&A
5. Quiz Generation
6. Quiz Attempt & Feedback

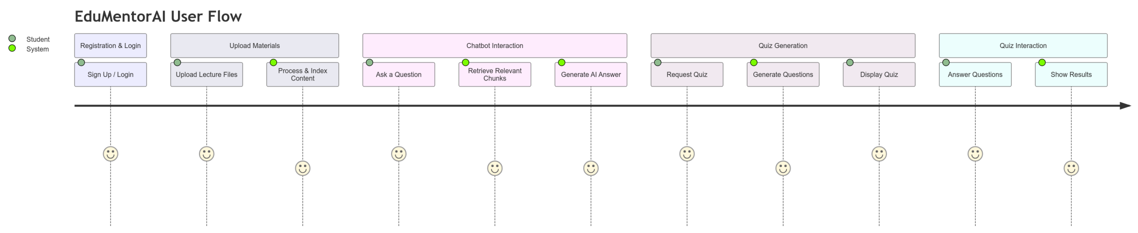

Figure 1: User flow diagram for EduMentorAI.

4. High-Level Architecture & Technology Stack

Technology Stack

- **Frontend:** Django Templates, HTML, CSS, JavaScript
- **Backend:** Django Framework
- **Database:** PostgreSQL for structured data storage
- **Vector Store:** FAISS for semantic search indexing
- **AI Models:** Hosted Model from Open Router

Architecture Diagram

The architecture diagram will show how:

- Users interact with the frontend.
- Django backend handles authentication and processing.
- Uploaded files are stored.
- Content embeddings are indexed in FAISS.
- The chatbot retrieves context from FAISS and queries Model.

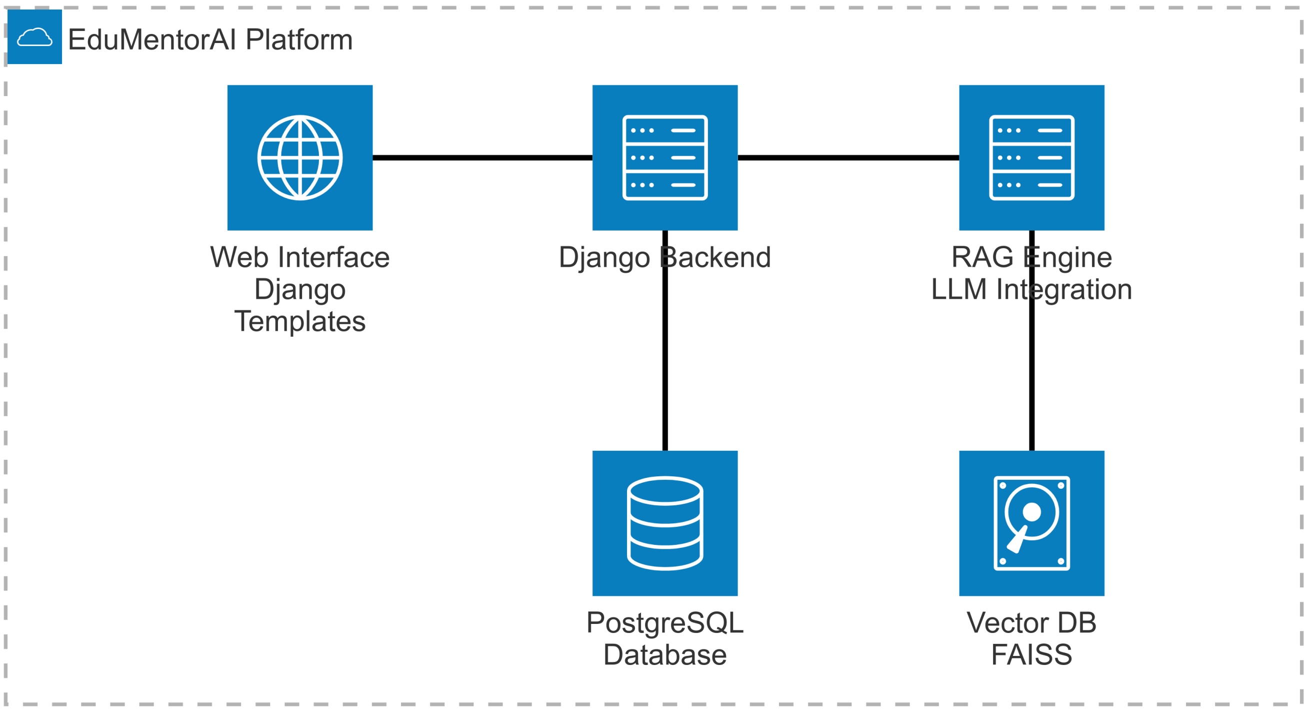

Figure 2: User flow diagram for EduMentorAI.

5. Projected Timeline and Milestones

- **Week 1:** Requirement gathering, architecture design.
- **Week 2:** Backend setup, database schema design.
- **Week 3:** File upload and preprocessing module.
- **Week 4:** RAG chatbot integration.
- **Week 5:** Quiz generation and feedback system.
- **Week 6:** Testing and bug fixes.
- **Week 7:** Deployment and final review.

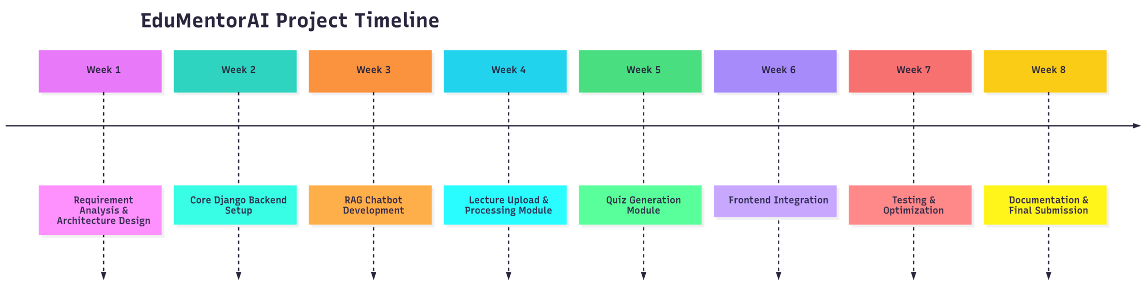
EduMentorAI Project Timeline
Week 1
Week 2
Week 3
Week 4
Week 5
Week 6
Week 7
Week 8
Requirement Analysis & Architecture Design
Core Django Backend Setup
RAG Chatbot Development
Lecture Upload & Processing Module
Quiz Generation Module
Frontend Integration
Testing & Optimization
Documentation & Final Submission

Figure 3: Project EduMentorAI.

6. Planned Use of Generative AI Tools in Each Development Phase

The development of **EduMentorAI** will strategically leverage Generative AI tools throughout the software lifecycle to accelerate ideation, improve development efficiency, ensure high-quality testing, and produce professional documentation. This integration will not only optimize team productivity but also serve as a real-world demonstration of the platform's own core philosophy—using AI as a collaborative partner in education and software engineering.

6.1. Phase 1: Ideation and Requirements Gathering

During the initial phase, Generative AI will be used to explore potential features, refine the problem statement, and produce initial user stories.

- **Prompt Example:** "Suggest 10 innovative features for an AI-powered educational platform that uses RAG and supports university students in self-study."
- AI-assisted brainstorming for quiz generation approaches, retrieval pipelines, and context-aware chat responses.
- Drafting competitive analysis summaries from research data.
- Producing initial wireframe suggestions based on textual descriptions of desired workflows.

6.2. Phase 2: System Design and Architecture

Generative AI will be utilized to produce architecture proposals, component diagrams, and high-level system specifications.

- **Prompt Example:** "Generate a high-level architecture for a Django-based web application with a RAG chatbot, vector database, and PostgreSQL backend."
- Generating UML sequence diagrams for the document upload and quiz-generation workflows.
- Providing alternative design patterns for integrating FAISS vector search with Django views.

6.3. Phase 3: Coding and Implementation

Generative AI tools (e.g., GitHub Copilot, ChatGPT) will help accelerate coding while maintaining code quality.

- **Prompt Example:** "Write a Django view function that handles PDF upload, extracts text, and indexes it into a FAISS vector store."
- Producing boilerplate Django templates for login, dashboard, and chatbot interfaces.
- Suggesting efficient query-chunking and embedding generation strategies.
- Refactoring code for optimization and adhering to PEP8 coding standards.

6.4. Phase 4: Testing and Quality Assurance

AI tools will generate and refine unit, integration, and end-to-end test cases.

- **Prompt Example:** "Generate Django unit tests for a function that retrieves top-3 relevant chunks from a FAISS vector store given a query."
- Automated generation of edge-case scenarios for quiz generation.
- AI-assisted performance testing scripts for evaluating RAG latency under different workloads.

6.5. Phase 5: Documentation and User Training Materials

Generative AI will be instrumental in creating both technical documentation and user guides.

- **Prompt Example:** "Generate a concise API reference for a Django REST endpoint that returns AI-generated quiz questions from stored lecture data."
- Generating README files, in-line docstrings, and developer onboarding guides.
- Producing clear end-user manuals with annotated screenshots.
- Translating technical documentation into simplified guides for non-technical stakeholders.

6.6. Phase 6: Continuous Improvement and Feedback Integration

Post-launch, AI tools will support bug triaging, feature suggestions, and analytics interpretation.

- **Prompt Example:** "Analyze chatbot usage logs and summarize the top 5 recurring topics where students request help."
- Summarizing user feedback for sprint planning.
- Generating proposals for incremental feature updates.

6.7. Expected Benefits of AI-Assisted Development

By incorporating Generative AI in all phases, **EduMentorAI** will:

- Reduce development time by automating repetitive coding and documentation tasks.
- Increase system design quality through AI-driven brainstorming and architecture validation.
- Improve testing coverage and bug detection through AI-generated test cases.
- Enhance project scalability with continuous AI-assisted iteration and refinement.

The integration of Generative AI is not merely a productivity booster—it aligns directly with the mission of **EduMentorAI** to empower learning through AI, demonstrating that the same principles can accelerate the creation of the platform itself.