EduTutorAI –

Personalized Learning with Generative AI and LMS Integration

Project Documentation

1. Introduction

Title – EduTutorAI – Personalized Learning with Generative AI and LMS Integration.

Team Member – Pavithra P

Team Member – Preethi R

Team Member – Vyshalini N

Team Member – Yuvashree D

2. Project Overview

Purpose :

EduTutorAI is an AI-powered personalized education platform that revolutionizes the way students learn and educators assess progress. It provides dynamic quiz generation, student evaluation, Google Classroom integration, and real-time feedback—all powered by IBM Watsonx and Granite foundation models. Designed with modular architecture, this platform streamlines personalized education and enhances learning outcomes across academic levels.

Features :

Dynamic Quiz Generation

- **Key Point:** AI-powered assessments
- **Functionality:** Uses IBM Watsonx Granite LLM to generate multiple-choice questions and quizzes dynamically for any academic topic.

Personalized Feedback

- **Key Point:** Instant evaluation
- **Functionality:** Student responses are evaluated automatically, and adaptive feedback is provided based on performance level.

Educator Dashboard

- **Key Point:** Real-time performance insights
- **Functionality:** Displays student quiz history, scores, recent topics, and analytics fetched from Pinecone for monitoring progress.

Google Classroom Integration

- **Key Point:** Seamless academic sync
- **Functionality:** Syncs courses, class names, and subjects from Google Classroom to auto-generate relevant quizzes aligned with curriculum.

Diagnostic Testing & Adaptive Quizzing

- **Key Point:** Tailored learning journey
- **Functionality:** Conducts an initial diagnostic test; adjusts quiz difficulty and topic relevance dynamically to match student ability.

Pinecone Vector Database

- **Key Point:** Smart learning memory
- **Functionality:** Stores student embeddings, quiz history, and metadata to enable adaptive learning and similarity-based topic recommendations.

Streamlit User Interface

- **Key Point:** Intuitive dashboards
- **Functionality:** Provides a user-friendly frontend for both students and educators, with separate panels for quizzes, analytics, and history.

FastAPI Backend

- **Key Point:** Modular API architecture
- **Functionality:** Handles login, quiz generation, answer evaluation, and data sync with Pinecone and Google Classroom through secure endpoints.

3. Architecture :

- **Frontend (Stream lit):**

The frontend is built with Stream lit, offering an interactive web UI with multiple pages including dashboards, file uploads, chat interface, feedback forms, and report viewers. Navigation is handled through a sidebar using the stream lit-option-menu library. Each page is modularized for scalability.

- **Backend (Fast API):**

Fast API serves as the backend REST framework that powers API endpoints for document processing, chat interactions, eco tip generation, report creation, and vector embedding. It is optimized for asynchronous performance and easy Swagger integration.

- **LLM Integration (IBM Watsonx Granite):**

Granite LLM models from IBM Watsonx are used for natural language understanding and generation. Prompts are carefully designed to generate summaries, sustainability tips, and reports.

- **Vector Search (Pinecone):**

Uploaded policy documents are embedded using Sentence Transformers and stored in Pinecone. Semantic search is implemented using cosine similarity to allow users to search documents using natural language queries.

4. Setup Instructions:

Prerequisites:

- Python 3.9+
- API Keys for IBM Watsonx + Pinecone + Google Classroom
- Internet access

Installation Process :

- Clone the repository
- Install dependencies from requirements.txt
- Create a .env file and configure credentials
- Run the backend server using Fast API
- Launch the frontend via Stream lit
- Upload data and interact with the modules

5. Folder Structure

- app/ – Contains all Fast API backend logic including routers, models, and integration modules.
- app/api/ – Subdirectory for modular API routes like chat, feedback, report, and document vectorization.
- ui/ – Contains frontend components for Stream lit pages, card layouts, and form UIs.
- smart_dashboard.py – Entry script for launching the main Stream lit dashboard.
- granite_llm.py – Handles all communication with IBM Watsonx Granite model including summarization and chat.
- document_embedder.py – Converts documents to embeddings and stores in Pinecone.
- edututorai.py – Explain topics and generate quiz.
- anomaly_file_checker.py – Flags unusual values in uploaded KPI data.
- report_generator.py – Constructs AI-generated sustainability reports.

6. Running the Application

To start the project:

- Launch the FastAPI server to expose backend endpoints.
- Run the Streamlit dashboard to access the web interface.
- Navigate through pages via the sidebar.
- Upload documents or CSVs, interact with the chat assistant, and view notes of the topic and generate quiz. All interactions are real-time and use backend APIs to dynamically update the frontend.

Frontend (Stream lit):

The frontend is built with Stream lit, offering an interactive web UI with multiple pages including dashboards, file uploads, chat interface, feedback forms, and report viewers. Navigation is handled through a sidebar using the stream lit-option-menu library. Each page is modularized for scalability.

Backend (Fast API):

Fast API serves as the backend REST framework that powers API endpoints for document processing, chat interactions, eco tip generation, report creation, and vector embedding. It is optimized for asynchronous performance and easy Swagger integration.

7. API Documentation

Backend APIs available include:

- POST /chat/ask – Accepts a user query and responds with an AI-generated message
- POST /upload-doc – Uploads and embeds documents in Pinecone
- GET /search-docs – Returns semantically similar policies to the input query
- GET /get-topic – Provides explanation topic and generate quiz
- POST /submit-feedback – Stores citizen feedback for later review or analytics

Each endpoint is tested and documented in Swagger UI for quick inspection and trial during development.

8. Authentication

Each endpoint is tested and documented in Swagger UI for quick inspection and trial during development. This version of the project runs in an open environment for demonstration. However, secure deployments can integrate:

- Token-based authentication (JWT or API keys)
- OAuth2 with IBM Cloud credentials
- Role-based access (admin, citizen, researcher)
- Planned enhancements include user sessions and history tracking.8. Authentication

9. User Interface

- Student Panel: Login, dashboard, take quiz, view history.
- Educator Panel: Analytics dashboard showing student progress.
- Streamlit sidebar navigation for modular UI.

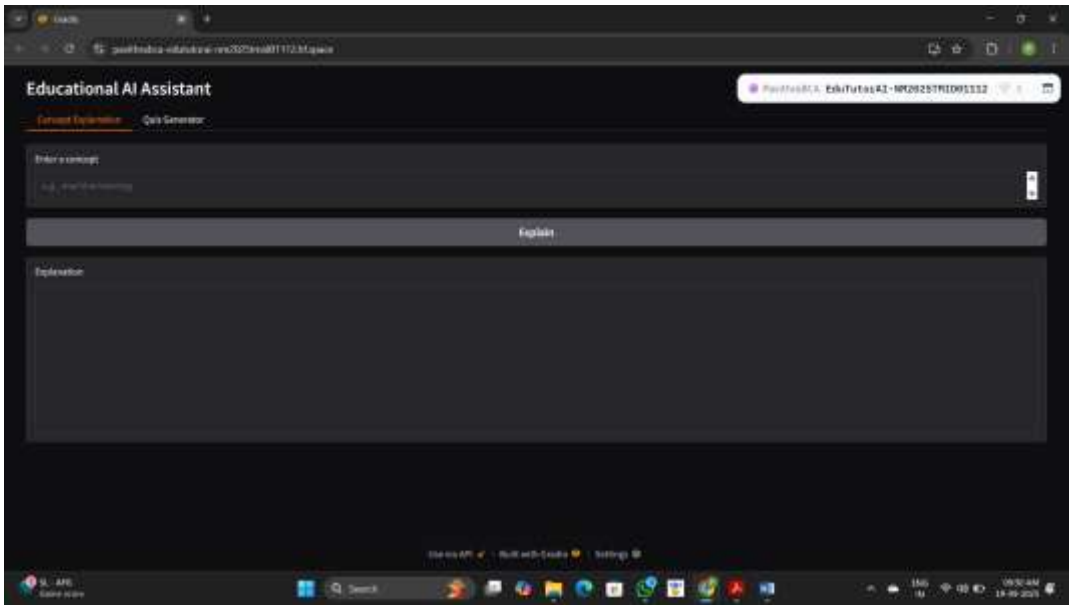
10. Testing

Testing was done in multiple phases:

- Unit Testing: For prompt engineering functions and utility scripts
- API Testing: Via Swagger UI, Postman, and test scripts
- Manual Testing: For file uploads, chat responses, and output consistency
- Edge Case Handling: Malformed inputs, large files, invalid API keys

Each function was validated to ensure reliability in both offline and API-connected modes.

11.Screenshots



12. Known Issues

- Long model loading time on Hugging Face.
- Free tier Pinecone limitations.
- Google OAuth setup complexity.

13. Future Enhancements

- Multi-language quiz generation.
- Voice-based quiz interaction.
- Extended LMS support beyond Google Classroom.
- Mobile app integration.