Educational AI Assistant - Project Documentation

1. Introduction

• Project Title: Educational AI Assistant
• Team Members: S.Jerry Jinson
• Team Members:D.Kavibharathi
• Team Members:C.John Doniel
• Team Members:P.Antony Vikash

2. Project Overview

The purpose of the Educational AI Assistant is to provide learners with a platform that can explain complex topics in simple terms and generate interactive quizzes for practice. By leveraging AI and user inputs, the assistant supports personalized learning and improves concept retention. It acts as a real-time educational companion for both students and teachers.

• Features:

- Conversational Interface – Natural language interaction for ease of use.
- Concept Explanation – Converts difficult topics into simplified content.
- Quiz Generator – Dynamically generates quizzes based on input concepts.
- User-Friendly Interface – Developed using Gradio for clean accessibility.
- Extensible Modules – Can integrate with datasets and more AI tools in future.

3. Architecture

Frontend (Gradio): The frontend uses Gradio to provide a web-based interface where learners can input concepts, generate explanations, and take quizzes. Backend (FastAPI / Python): The backend manages AI integration, processes user queries, and handles API endpoints for quiz generation and explanations. LLM Integration: The assistant uses pre-trained LLMs (Large Language Models) to create accurate explanations and quizzes. Modular Design: The project is modular, making it scalable for future improvements such as tracking student progress and generating reports.

4. Setup Instructions

Prerequisites: - Python 3.9 or later - pip and virtual environment tools - Gradio installed Installation Process: - Clone the repository - Install dependencies from requirements.txt - Run the backend server (FastAPI optional) - Launch the Gradio interface to interact with the assistant

5. Folder Structure

app/ – Backend logic for FastAPI, AI integration, and APIs app/api/ – API routes for explanation and quiz generation ui/ – Gradio UI components (concept input, quiz tab, results display) assistant_core.py – Main logic for AI-based explanation quiz_generator.py – Handles quiz creation report_module.py – Generates PDF reports of student progress

6. Running the Application

1. Start the backend server (if using FastAPI APIs). 2. Launch the Gradio interface. 3. Enter a concept in the input field. 4. Click 'Explain' to generate an explanation. 5. Switch to 'Quiz Generator' tab to practice quizzes. 6. View explanations, quiz results, and download reports.

7. API Documentation

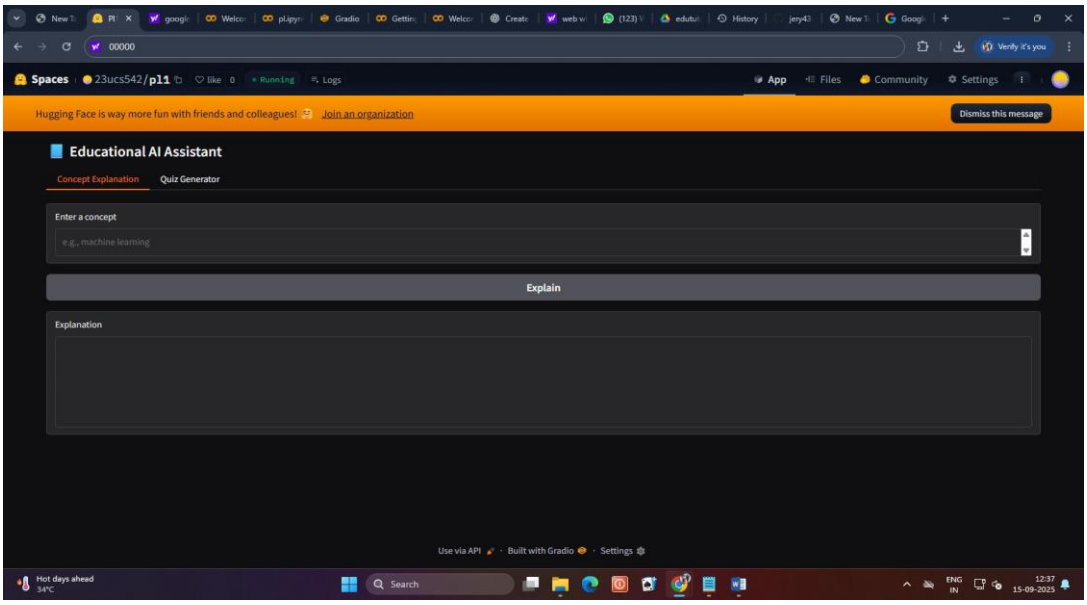
POST /explain – Takes a concept input and returns an AI-generated explanation POST /generate-quiz – Creates a quiz based on input topics GET /progress-report – Generates a progress report for learners

8. Authentication

For the demo version, the project runs in an open environment. Future secure deployments may include: - Token-based authentication (JWT or API keys) - Role-based access for admin, teacher, or student users - Session tracking for personalized learning

9. User Interface

The interface is designed to be simple and accessible for students. It includes: - Concept input field - 'Explain' button to generate explanations - 'Quiz Generator' tab for practice - Explanation and results section



https://jery43-35.github.io/pl1/

10. Testing

Testing was done through: - Unit Testing: Validating explanation and quiz generation modules. - Manual Testing: UI interaction and usability checks. - Edge Case Handling: Invalid inputs, empty queries.

11. Known Issues

- Limited domain knowledge (depends on model training). - No persistent user tracking in demo version. - Quiz variety limited to simple question formats.

12. Future Enhancements

- Support for multiple learning modes (audio, video explanations). - Integration with learning management systems (LMS). - Advanced quizzes (MCQs, coding tasks, interactive diagrams). - Student analytics and performance dashboards.