Edu Tutor AI

Project Documentation

1. Introduction

• Project Title: Edu Tutor AI

• Team Leader : SIVAPRIYA S

Team Member 1: KAILASH S

Team Member 2: DHARUNESH KUMAR R

Team Member 3: AKKIM BASHA S

Team Member 4: SIRANJEEVI M

2. Project Overview

Purpose: Edu Tutor AI is an intelligent virtual learning assistant designed to enhance education by providing personalized tutoring, adaptive learning paths, and real-time student support.

- Conversational Interface – Natural interaction with students.
- Automated Grading & Feedback – Provides instant results and feedback.
- Personalized Learning Paths – Adaptive lessons based on progress.
- Content Summarization & Generation – Summarizes textbooks and generates quizzes.
- Performance Analytics – Tracks and visualizes student progress.
- Multi-Modal Input Support – Handles text, PDFs, and images.
- Exam Preparation Mode – Generates mock tests and flashcards.
- Teacher Collaboration Tools – Lesson prep and resource sharing.

3. Architecture

Frontend (Streamlit/Gradio): Interactive dashboard for students and teachers. Backend (FastAPI): API framework for tutoring, grading, and analytics.

LLM Integration (OpenAI/IBM Watsonx Granite): AI models for tutoring and summarization. Vector Search (Pinecone/FAISS): Stores and searches educational material. ML Modules: Recommendation and analytics for learning progress.

4. Setup Instructions

1 Python 3.9+ and pip installed.
2 Clone repository and install dependencies.
3 Configure environment variables in .env file.
4 Run FastAPI backend server.
5 Launch Streamlit/Gradio frontend.
6 Upload material or start tutoring session.

5. Folder Structure

app/ – Backend logic app/api/ – API routes ui/ – Frontend dashboards and quizzes tutor_ai.py – Core AI tutoring logic grading_engine.py – Automated grading analytics_engine.py – Performance tracking lesson_generator.py – Lesson and quiz generation

6. Running the Application

• Start FastAPI backend • Launch Streamlit UI • Navigate via sidebar • Students/teachers interact with assistant • Real-time responses and analytics displayed

7. API Documentation

POST /chat/ask – Student queries answered POST /upload-doc – Upload textbooks or notes GET /search-docs – Semantic search of material POST /grade-quiz – Automated grading GET /progress-report – Student progress analytics

8. Authentication

Supports JWT, API keys, role-based access (Student, Teacher, Parent, Admin), and OAuth2 integration.

9. User Interface

Sidebar navigation, student dashboards, teacher dashboards, quizzes, progress graphs, and resource sharing tools.

10. Testing

Unit testing, API testing with Swagger/Postman, manual testing of tutoring, edge case handling for large files.

11. Screenshots

(To be added after UI implementation)

12. Known Issues

Limited support for handwritten notes, requires internet access, curriculum-specific fine-tuning needed.

13. Future Enhancements

Voice-based tutoring, AR/VR immersive learning, offline lightweight AI, gamification for motivation.