PROJECT DOCUMENTATION:

Edu Tutor AI: Personalized Learning with Generative AI and LMS Integration

1.Introduction:

EduTutor AI is a virtual learning assistant that uses Generative AI to provide personalized study plans, quizzes, and progress tracking. It helps students learn effectively and supports teachers with automated reports and analytics.

Title: Edu Tutor AI:Personalized Learning with Generative AI and LMS Integration

Team ID: NM2025TMID11336

Team leader: HEMALATHA K

Team members: SALINI S & RATHIKA K & VIJAYADHARASHINI S & MAHESWARI T

2.Project Overview:

- **Goal**: Make learning interactive, personalized, and accessible.
- **Users**: Students and Teachers.
- **Built With**: Streamlit (frontend), FastAPI (backend), Generative AI (LLM), Pinecone (vector search).

Features

- Personalized study path recommendations.
- Automated quiz and assignment generation.
- AI chatbot for instant doubt solving.
- Performance tracking and analytics.
- Teacher support with student summaries.

3.Architecture:

- **Frontend**: Streamlit dashboard.
- **Backend**: FastAPI handles requests.
- **LLM**: Generative AI for tutoring.
- **Pinecone**: Vector search for study resources.
- **ML Modules**: Forecast student progress, detect performance drops.

4.Setup Instructions

Prerequisites

- Python 3.9+
- Pip or conda
- API key for LLM (e.g., OpenAI, IBM Granite, etc.)
- Pinecone API key
- Git

Installation Process

1. Clone the repository
2. Install dependencies from `requirements.txt`
3. Create a `.env` file and configure credentials
4. Run the backend server using FastAPI
5. Launch the frontend via Streamlit

5.Folder Structure:

```
EduTutor-AI/
├── app/           # Backend logic (APIs, models, services)
├── ui/            # Frontend (pages, components, chatbot)
├── modules/       # AI/ML modules (quiz, reports, recommender)
├── data/          # Sample datasets
├── requirements.txt
├── .env           # Config keys
├── main.py        # FastAPI entry point
└── dashboard.py   # Streamlit entry point
```

6.Running the Application:

Backend (FastAPI)

1. Open terminal and navigate to project folder.
2. Run: `uvicorn main:app --reload`
3. FastAPI backend will start at: **http://127.0.0.1:8000**

Frontend (Streamlit)

1. Open another terminal.
2. Run: `streamlit run dashboard.py`
3. Streamlit frontend will open in browser: **http://localhost:8501**

7.API Documentation:

- **POST /chat/ask** → Send a question, get AI-generated answer.
- **POST /quiz/generate** → Generate quiz for selected topic.
- **POST /upload/assignment** → Upload assignment for analysis.
- **GET /report/student/{id}** → Get performance report of a student.
- **POST /feedback/submit** → Submit student/teacher feedback.

8.Authentication:

- JWT-based authentication system.
- User registration and login.
- Role-based access (student/teacher).

9.User Interface:

- Streamlit-based dashboard.
- Student chat interface for Q&A.
- Teacher dashboard for reports.
- Progress visualization charts.

10.Testing:

- **Unit Testing**: Core functionalities.
- **Integration Testing**: API + UI.
- **Model Validation**: Accuracy of AI recommendations.
- **Tools**: Pytest, Postman.

11.Screenshot:

12.Known Issues:

- LLM sometimes gives generic answers.
- Internet connection required for AI and vector search.
- Free API limits (LLM & Pinecone).
- Performance varies with large datasets.

13.Future Enhancements:

- LMS integration (Google Classroom, Moodle).
- Mobile app version.
- Multilingual tutoring.
- AR/VR immersive learning.
- Career guidance recommendations.

14.Objectives & Scope:

Objectives:

- Provide AI-driven learning support.

- Reduce teacher workload with automation.
- Improve student performance with personalization.

Scope:

- Covers students and teachers.
- Supports both personalized and group learning.

15.Problem Statement:

Traditional education faces:

- Overloaded teachers, limited personal guidance.
- Lack of personalized support for students.
- Difficulty in tracking progress.

EduTutor AI solves this with AI-driven tutoring and analytics.

16.Use Case Scenarios:

- **Student**: Gets instant answers + personalized quizzes.
- **Teacher**: Uploads results → AI generates reports.
- **Exam Prep**: AI generates practice test based on weak topics.

17.Technology Stack:

- **Frontend**: Streamlit
- **Backend**: FastAPI
- **Database**: MongoDB / PostgreSQL
- **AI/LLM**: Generative AI (OpenAI, IBM Granite, etc.)
- **Vector Search**: Pinecone
- **ML Models**: Scikit-learn, TensorFlow, PyTorch
- **Auth**: JWT, OAuth2

18.Performance Metrics:

- API response time < 1 sec.
- Accuracy of quiz generation and recommendations.
- Forecasting error (RMSE/MAE).
- Scalability: Number of concurrent users supported.

19.Limitations:

- Not a replacement for teachers.
- Accuracy depends on quality of training data.
- Internet connection required.
- Free-tier API restrictions.

20.References:

- OpenAI / IBM Granite documentation.
- Pinecone DB documentation.
- Streamlit & FastAPI official docs.
- Research papers on AI in education.

Conclusion

EduTutor AI demonstrates how AI can transform education with personalized learning, automated assessments, and teacher support. It enhances accessibility, reduces workload, and improves engagement. With future upgrades like LMS integration, multilingual features, and AR/VR modules, it can become a complete digital learning companion.