**EduTutor AI - Python-Based AI Learning Platform Documentation**

**1. Introduction**

**Project Title:** EduTutor AI: Python-Powered Learning with Generative AI  
**Team Members:**

Team ID : LTVIP2025TMID32123

Team Size : 4

Team Leader : Sai Krishna Gajula

Team member : Rama Charan Challa

Team member : Thota Bhanu Venkata Chandrakala

Team member : Ravi Naresh Darapureddy

**2. Project Overview**

**Purpose:**  
EduTutor AI revolutionizes digital education by providing an AI-powered learning assistant built entirely with Python. The platform uses cutting-edge natural language processing to deliver personalized quizzes, explanations, and academic support through an intuitive Streamlit web interface.

**Features:**

* AI-generated quizzes with instant feedback
* Natural language Q&A with subject matter experts
* Student performance dashboards
* Cloud-based operation with local fallback options
* Mobile-responsive web interface

**3. Architecture**

**Core Architecture:**

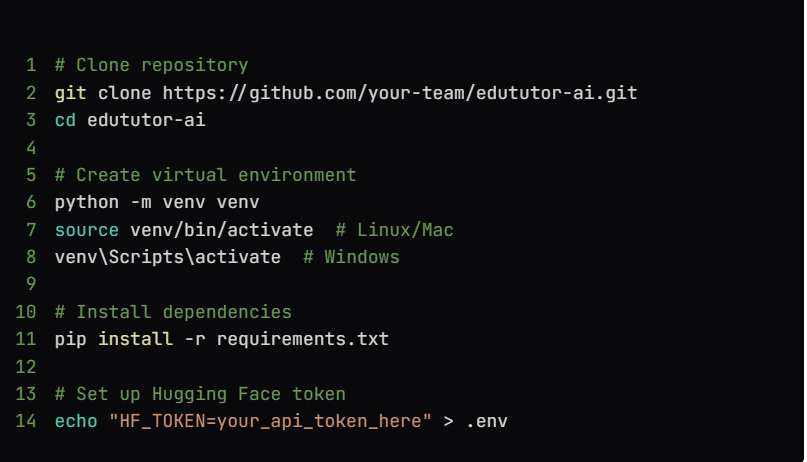
1. **Natural Language Processing:**  
   Uses Hugging Face's **google/flan-t5-xl** as primary model with **ibm/granite-3b-instruct** as fallback
2. **Interface Layer:**  
   Streamlit-powered web application with:
   * Student quiz dashboard
   * Educator analytics view
   * Question history tracking
3. **Knowledge Base:**  
   Curated question banks with:
   * AI-generated question variants
   * Solution explanations
   * Difficulty level tagging

**4. Setup Instructions**

**Requirements:**

* Python 3.8+
* Hugging Face account (free API token)
* 8GB+ RAM recommended

**Installation:**

**5. Folder Structure**

RunCopy code

EduTutor-AI/

├── app.py # Main Streamlit application

├── questions.py # Manual question sets for quiz (AI, ML, etc.)

├── load\_flan\_model.py # Optional script for loading models locally

├── requirements.txt # All dependencies

└── README.md # You’re reading it!

**6. System Operation**

**Normal Operation:**

1. Users access via web browser at **localhost:8501**
2. Application loads with minimal dependencies
3. AI models are loaded on-demand from Hugging Face

**Offline Mode:**

* Can operate with locally cached models
* Limited functionality without internet

**7. Key Interfaces**

**Student View:**

* Subject selection panel
* Interactive quiz interface
* AI chat assistant
* Performance history

**Educator View:**

* Class performance metrics
* Quiz customization
* Answer pattern analysis

**8. Known Issues**

* Delay when loading large AI models
* Occasional inaccuracies in free-form answers
* Mobile rendering quirks on some devices

**9. Future Enhancements**

* Optimize model loading with quantization
* Add voice interaction
* PDF textbook integration
* Collaborative whiteboard feature
* Automatic lesson plan generation

**Acknowledgments**

* Hugging Face for open models
* Streamlit for rapid UI development
* IBM for Granite models
* Open education resource contributors

This Python-native implementation provides equivalent functionality to the original stack while simplifying deployment and reducing dependencies - ideal for educational settings where JavaScript frameworks may present unnecessary complexity.