Project Documentation

EduTutor AI: Personalized Learning Assistant

1. Introduction

Project Title: EduTutor AI: Personalized Learning Assistant

Team Member: HAMIRTHA H

Team Member: ABINAYA E

Team Member: LATHIKA M

Team Member: MERITTRA ROJH T

2. Project Overview

Purpose:

EduTutor AI is an AI-powered personalized education platform designed to make learning more engaging, adaptive, and effective. By leveraging IBM's Granite 3.2-2B instruct model, the assistant enables students to learn complex concepts in simple terms, generate quizzes, evaluate answers, and receive study plans based on weaknesses. Teachers benefit from performance analytics, class dashboards, and Google Classroom integration.

The goal is to create an intelligent assistant that supports students in self-paced learning and helps educators streamline class monitoring.

Features:

1. Conversational Interface – Natural language interaction for learning and Q&A.

2. Concept Explanation – Explains topics at different levels (basic, intermediate, advanced) with examples.

3. Quiz Generator – Creates quizzes of varying types and difficulty with answer keys.

4. Answer Evaluation – Provides instant scoring and feedback for submitted answers.

5. Diagnostic Test – Creates tests to identify student learning gaps.

6. Study Plan Generator – Builds personalized study plans based on weak areas.

7. Performance Analytics – Provides insights into learning progress and trends.

8. Google Classroom Sync – Imports courses and assignments seamlessly.

9. Educator Dashboard – Tracks class performance and quiz analytics.

10. Gradio UI – Interactive interface with multiple tabs for students and educators.

3. Architecture

Frontend (Gradio):

The UI is built using Gradio Blocks and Tabs with a clean theme. It includes dashboards, concept learning, quizzes, evaluations, analytics, diagnostic tests, Google Classroom sync, and educator view.

Backend (Transformers + Python APIs):

Python functions handle LLM queries, quiz generation, evaluation, and performance analysis. APIs can be extended for integration with external platforms.

LLM Integration (IBM Granite 3.2-2B):

Granite instruct model powers natural language understanding, concept explanation, and response generation.

ML Modules:

Quiz evaluation and performance analysis logic.

Analytics on student scores and trends.

Data Handling:

Courses and assignments (sample dataset).

Quiz results and performance records (simulated data).

4. Setup Instructions

Prerequisites:

Python 3.9 or later

pip (for installing dependencies)

GPU (optional, for faster model inference)

Hugging Face access to IBM Granite model

Installation Process:

1. Clone the repository.
2. Install dependencies: pip install -r requirements.txt.
3. Download and configure the Granite model.
4. Run the Gradio app:

python app.py

5. Access the UI in browser

5. Folder Structure

```
edututor_ai/
│── app.py              # Main Gradio interface
│── utils/
│    ├── concept_explainer.py
│    ├── quiz_generator.py
│    ├── answer_evaluator.py
│    ├── diagnostic_test.py
│    ├── study_plan.py
│    └── performance_analysis.py
│── data/
│    ├── sample_courses.json
│    ├── sample_assignments.json
│    └── sample_performance.json
│── requirements.txt
│── README.md
```

6. Running the Application

1. Start the Gradio app with:

```
python app.py
```

2. Navigate through the tabs:

Dashboard: Student profile and activity.

Learn: Concept explanations.

Quiz: Quiz generator.

Evaluate: Quiz answer evaluation.

Diagnostic: Gap-finding test generation.

Analytics: Student progress visualization.

Google Classroom: Sync and view assignments.

Educator View: Teacher dashboard.

7. API Documentation

(Currently integrated in Python functions, extendable to REST APIs with FastAPI)

POST /explain-concept – Returns explanation with examples.

POST /generate-quiz – Creates quiz with answers.

POST /evaluate-answers – Evaluates submitted answers.

POST /generate-study-plan – Returns personalized study plan.

POST /analyze-performance – Provides insights on quiz results.

8. Authentication

Current version runs in open mode for demo.

Future enhancements:

Role-based access (Student/Teacher).

Google OAuth for Classroom integration.

Secure API keys for external integrations.

9. User Interface

Navigation Tabs: Dashboard, Learn, Quiz, Evaluate, Diagnostic, Analytics, Classroom, Educator.

Student Features: Concept learning, quizzes, evaluations, study plan.

Teacher Features: Class analytics, student performance reports.

UI Theme: Purple-based clean layout for better accessibility.

10. Testing

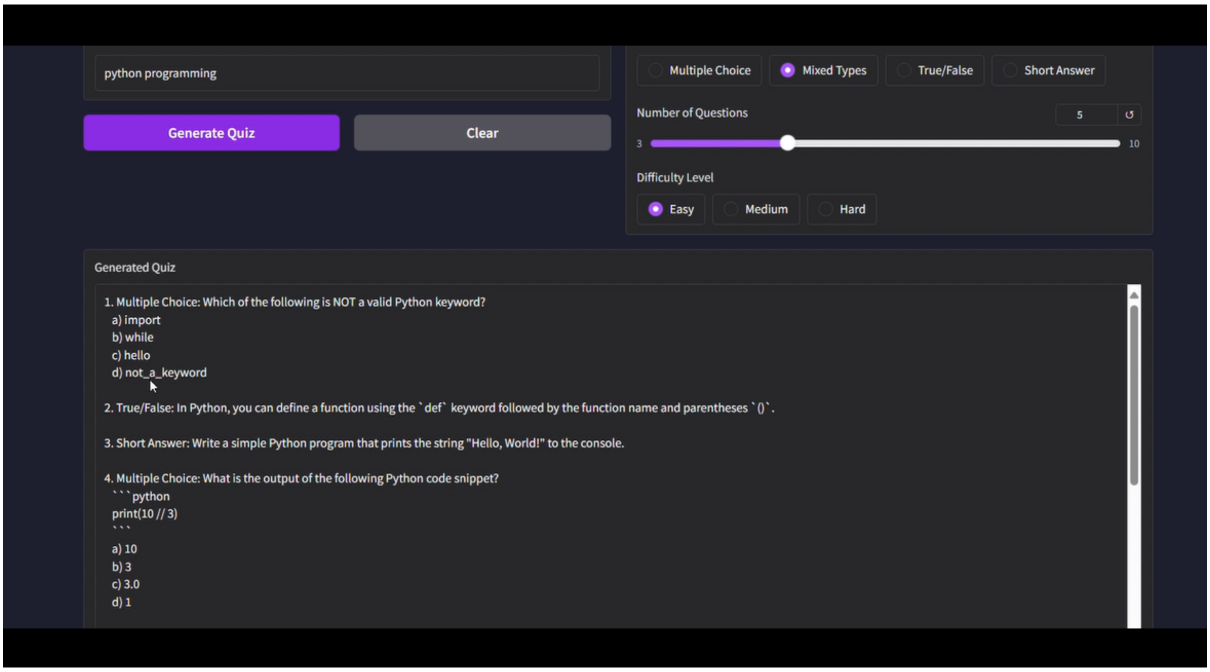
Unit Testing: Model outputs for explanation and quiz generation.

Functional Testing: Gradio buttons and tab workflows.

Manual Testing: Student inputs, assignment sync, report generation.

Edge Cases: Empty input, malformed data, long text inputs.

11. Screenshot



python programming

○ Multiple Choice ● Mixed Types ○ True/False ○ Short Answer

Number of Questions: 5 (3 – 10)

Difficulty Level: ● Easy ○ Medium ○ Hard

Generated Quiz

~~~
1. Multiple Choice: Which of the following is NOT a valid Python keyword?
  a) import
  b) while
  c) hello
  d) not_a_keyword

2. True/False: In Python, you can define a function using the `def` keyword followed by the function name and parentheses `()`.

3. Short Answer: Write a simple Python program that prints the string "Hello, World!" to the console.

4. Multiple Choice: What is the output of the following Python code snippet?
  ```python
  print(10 // 3)
  ```
  a) 10
  b) 3
  c) 3.0
  d) 1
~~~

12. Known Issues

Large quizzes may slow response time.

Google Classroom sync currently simulated (not real API).

Performance analytics uses sample data, not live user logs.

13. Future Enhancements

Real Google Classroom API integration.

Role-based user accounts with secure login.

Advanced adaptive learning (recommendations powered by ML).

Multi-language support for global usage.

Student progress history and report downloads.