*Edututor AI: Personalized Learning*

**Project Documentation**

**1. Introduction**

* **Project Title:** *Edututor AI: Personalized Learning*
* **Team Leader : Siddhik Raj M**
* **Team Member : Nithish Raj S**
* **Team Member : Abishek Y**
* **Team Member : Harish M**

Education today requires smarter, more adaptive tools to help students learn at their own pace. *Edututor AI* was developed as a personalized learning assistant that explains academic concepts in detail and generates practice quizzes. Using artificial intelligence and a simple user interface, the system brings an interactive and student-friendly learning experience.

**2. Project Overview**

* **Purpose:**  
  The purpose of *Edututor AI* is to act as a **personal tutor** that can explain topics in plain language and provide practice questions for self-assessment. It helps students improve understanding, retain knowledge, and build confidence through interactive learning.
* **Features:**

**Concept Explanation**

* + *Key Point:* AI-powered explanations
  + *Functionality:* Provides structured, detailed explanations with examples for any topic.

**Quiz Generator**

* + *Key Point:* Active learning
  + *Functionality:* Generates 5 quiz questions (MCQ, true/false, short answer) and an answer key.

**Conversational Interface**

* + *Key Point:* Student-friendly interaction
  + *Functionality:* Students can type queries in natural language and get clear, direct responses.

**Gradio Interface**

* + *Key Point:* Simple and interactive
  + *Functionality:* Offers a tab-based web UI with sections for explanations and quizzes.

**3. Architecture**

* **Frontend (Gradio):**  
  A tab-based user interface where students can choose between "Concept Explanation" and "Quiz Generator."
* **Backend (Transformers + PyTorch):**  
  The Hugging Face transformers library and PyTorch are used to run the AI model efficiently.
* **LLM Integration (IBM Granite):**  
  The project uses the ibm-granite/granite-3.2-2b-instruct model, designed for detailed instruction and knowledge generation.
* **Platform (Google Colab):**  
  The entire project runs in Google Colab, making it accessible without local setup and capable of using free GPU acceleration.

**4. Setup Instructions**

**Prerequisites:**

* Google Account
* Internet connection
* Google Colab environment

**Steps in Google Colab:**

1. Open the project notebook.
2. Install dependencies:
3. !pip install gradio transformers torch
4. Run the code cells to initialize the model and interface.
5. A **shareable Gradio link** will appear—open it in your browser to use the app.

**5. File Structure**

Since the project runs on Colab, all components are inside a single notebook:

Edututor\_AI\_Colab.ipynb

│── Model loading and tokenizer setup

│── generate\_response function

│── concept\_explanation function

│── quiz\_generator function

│── Gradio interface (two tabs)

**6. Running the Application**

* Open the notebook in Colab.
* Execute the cells step by step.
* A Gradio app link will appear.
* Access the interface in your browser:
  + **Concept Explanation Tab:** Enter a topic → Get explanation.
  + **Quiz Generator Tab:** Enter a topic → Get quiz with answers.

**7. API Documentation**

* **concept\_explanation(concept)**
  + Input: Topic (string)
  + Output: Detailed explanation with examples
* **quiz\_generator(concept)**
  + Input: Topic (string)
  + Output: 5 quiz questions + answer section

These functions can later be exposed as APIs for integration into other platforms.

**8. Authentication**

Currently, *Edututor AI* runs as an **open-access demo** in Colab, where anyone with the link can use the app. This works for prototypes but is not secure for large-scale use.

Future versions can integrate:

* **API Key Access:** Only users with valid keys can access the tutor.
* **Google/OAuth Login:** Secure login through Google accounts.
* **Role-Based Access:** Different permissions for Students, Teachers, and Admins.
* **Session Tracking:** Save progress so students can revisit past quizzes and explanations.

This ensures the system is both **secure** and **personalized**.

**9. User Interface**

The interface is built with Gradio for **simplicity and usability**:

* Two main tabs: Concept Explanation & Quiz Generator.
* Textboxes for entering topics.
* Buttons to trigger responses.
* Output areas to display explanations or quizzes.

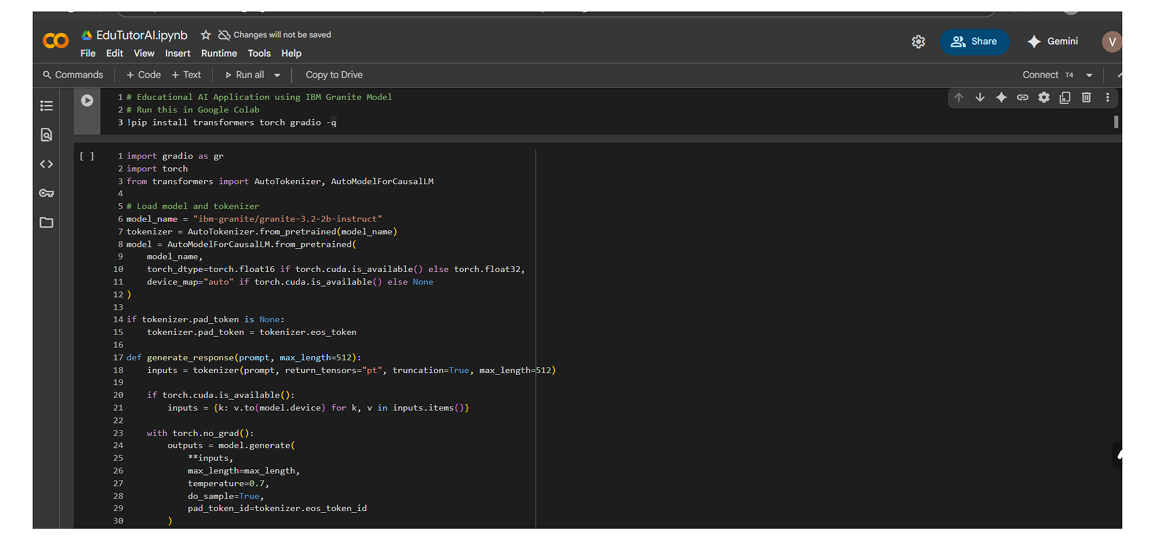
This design is beginner-friendly and requires no technical knowledge.

**10. Testing**

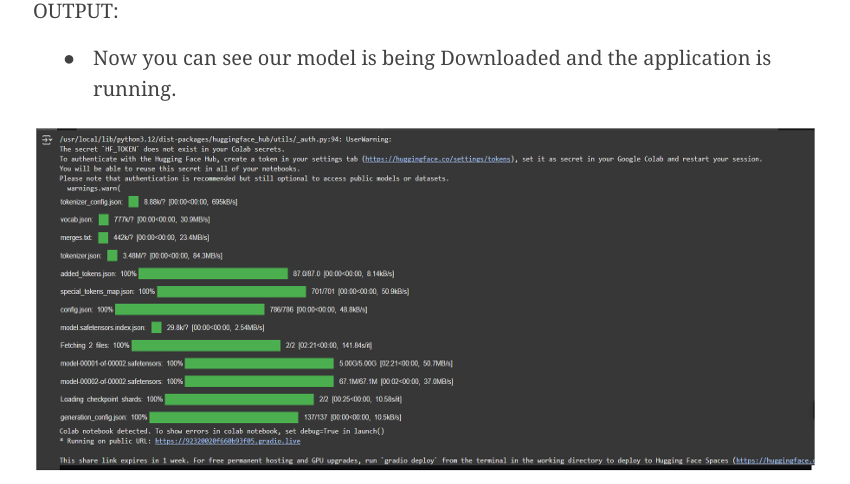
Testing was done in different stages:

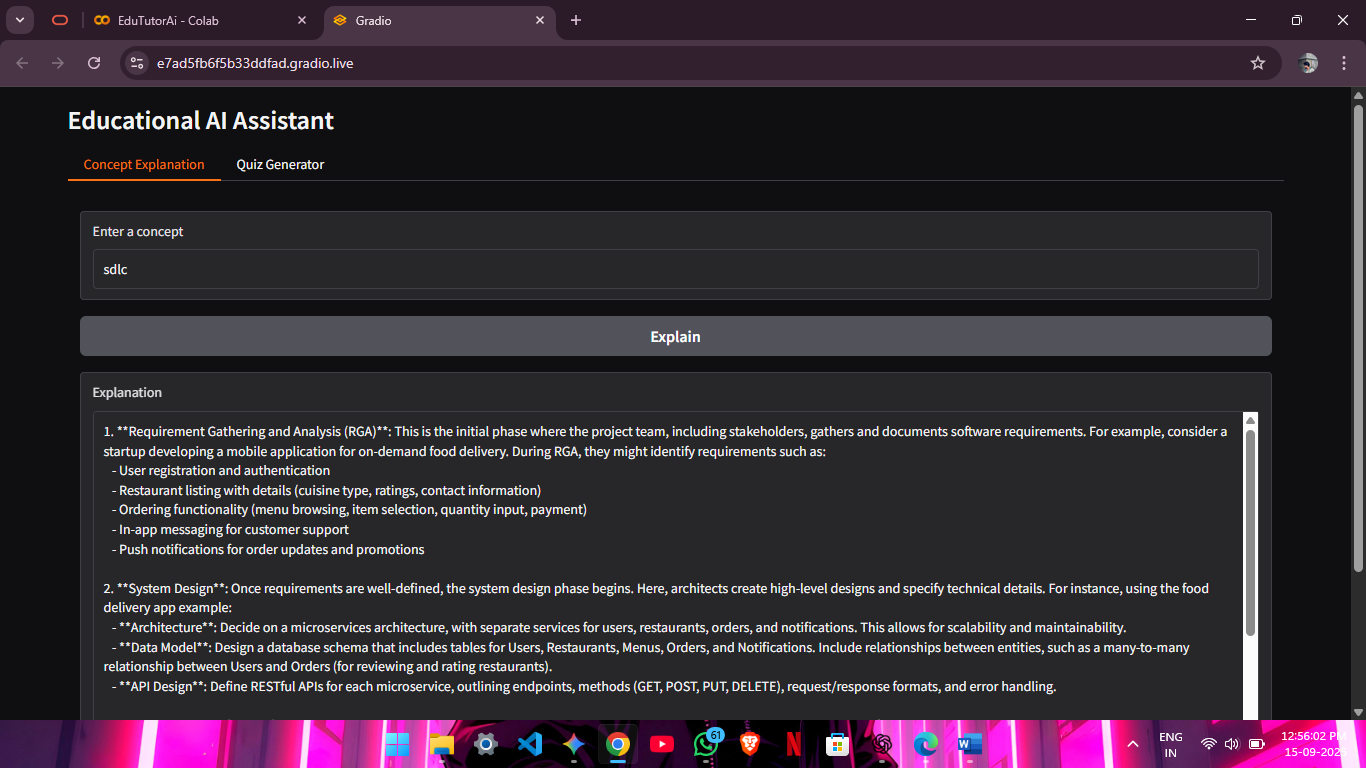
* **Unit Testing:** Verified that functions return correct outputs without errors.
* **Functional Testing:** Checked both tabs to ensure they generate correct responses.
* **Manual Testing:** Tried multiple subjects like Math, Science, and Technology.
* **Edge Cases:** Tested empty inputs, very long prompts, and unusual topics.

**11. Screenshots**

****

****

****

****

**12. Known Issues**

**Although *Edututor AI* works as intended for demonstrations, there are some limitations and challenges that need to be addressed for a production-ready system:**

* **1. Dependency on Internet and Colab Runtime**
  + **Since the project runs in Google Colab, it requires a stable internet connection.**
  + **Colab sessions also disconnect after a period of inactivity, which can interrupt usage.**
* **2. Limited GPU Resources**
  + **In the free version of Colab, GPU availability is restricted.**
  + **If many users try to access the system or if large models are used, performance may slow down.**
* **3. Response Formatting**
  + **While explanations are generally clear, quiz outputs sometimes lack consistent formatting.**
  + **For example, answers may not always align with the intended question type.**
* **4. Single-User Access**
  + **The current demo supports only one user session at a time.**
  + **A multi-user setup would require backend deployment on a dedicated server.**
* **5. No Persistent Storage**
  + **User history (past explanations and quizzes) is not saved.**
  + **Once the session ends, all generated content is lost.**
* **6. Limited Input Types**
  + **Currently, the system only accepts text-based inputs.**
  + **It does not yet support file uploads (PDFs, Docs) or voice input.**
* **7. Model Limitations**
  + **The AI responses depend heavily on the Granite model’s training data.**
  + **In some cases, answers may be too generic or slightly inaccurate, especially for very niche topics.**

**13. Future Enhancements**

**While *Edututor AI* is currently a functional prototype, there are several opportunities to expand its capabilities and make it more impactful in real-world learning environments.**

* **1. Document-Based Learning**
  + **Enable users to upload study materials like PDFs, Word files, or lecture notes.**
  + **The AI can extract key points, explain concepts from the uploaded material, and generate quizzes tailored to that content.**
  + **This feature would help students learn directly from their own resources.**
* **2. Voice Interaction (Speech-to-Text and Text-to-Speech)**
  + **Add support for voice input and output, allowing students to ask questions verbally and receive spoken answers.**
  + **This would make the platform more accessible for younger students, visually impaired learners, and those who prefer audio learning.**
* **3. Adaptive Learning Paths**
  + **Track user performance in quizzes and explanations.**
  + **Based on progress, the system can recommend easier or more advanced topics, ensuring a personalized learning journey.**
  + **For example, if a student struggles with algebra, the system could suggest simpler practice before moving to harder problems.**
* **4. Gamification of Learning**
  + **Introduce features like scores, badges, streaks, and leaderboards to make learning fun and engaging.**
  + **Gamification can boost motivation and encourage regular practice.**
* **5. Teacher Dashboard & Analytics**
  + **Develop a dedicated teacher interface where educators can monitor student progress.**
  + **Teachers can view quiz results, identify weak areas, and assign topics for AI-generated practice.**
* **6. Multi-Language Support**
  + **Extend support for multiple languages so students worldwide can benefit.**
  + **For example, explanations and quizzes could be generated in English, Hindi, Tamil, or other regional languages.**