**EduTutor AI with IBM – Project Documentation**

**Project Title:** EduTutor AI – Personalized Learning with IBM Granite.

**Team Members:** 1. J. Abi Jenifer

2. G. Angel Mary

3. A. Ammena Fathima

4. A. Afrin Fathima

.**Institution:** Shri Krishnaswamy College for Women.

**Course:** BCA.

**Date:** 19th September 2025.

**Table of Contents**

| **S. No.** | **Section Title** | **Page No.** |
| --- | --- | --- |
| 1 | Abstract | 3 |
| 2 | Project Overview | 4 |
| 3 | Features | 5 |
| 4 | System Architecture | 7 |
| 5 | Technology Stack | 9 |
| 6 | Setup Instructions | 11 |
| 7 | Folder Structure | 12 |
| 8 | Workflow Explanation | 13 |
| 9 | API Documentation | 15 |
| 10 | User Interface (UI/UX) | 16 |
| 11 | Testing | 17 |
| 12 | Screenshots Section | 19 |
| 13 | Known Issues & Limitations | 21 |
| 14 | Future Enhancements | 22 |
| 15 | Conclusion | 24 |
| 16 | References | 25 |

# 

# ****1.Abstract****

Artificial Intelligence (AI) is transforming education by enabling personalized, adaptive, and interactive learning experiences. Traditional teaching approaches often struggle to meet the diverse needs of students, as they follow a one-size-fits-all method. With the advent of generative AI and advanced language models, it is now possible to create systems that provide tailored content, explain concepts at varying levels, and adapt to the learner’s pace. This project, **EduTutor AI**, utilizes IBM’s Granite model, available through Hugging Face, to develop a personalized learning assistant. The system generates **concept explainers, quizzes, and adaptive study material** using generative AI techniques. It is deployed in **Google Colab** with a **Gradio-based interface**, ensuring accessibility and ease of use. The workflow involves exploring the Naan Mudhalvan Smart Interz Portal, selecting an IBM Granite model, implementing the system in Colab, and finally publishing the code on GitHub for collaboration and version control. The project demonstrates how generative AI can enhance learning outcomes by simplifying complex concepts and engaging students through interactive assessments. While challenges such as runtime limitations and GPU constraints in Colab were faced, the results highlight the potential of EduTutor AI as a foundation for future **scalable, multilingual, and AI-driven educational tools**.

## 2.Project Overview

EduTutor AI is an intelligent, AI-powered learning assistant designed to provide personalized support to students and educators. The project leverages **IBM Granite generative models** to deliver tailored explanations, generate practice quizzes, and summarize study content in simple, learner-friendly language. Unlike traditional educational systems that offer the same content to every learner, EduTutor AI adapts to individual needs, enabling students to learn at their own pace while assisting teachers in reducing repetitive tasks.

### *Purpose*

The primary goal of EduTutor AI is to bridge the gap between traditional one-size-fits-all learning and the diverse needs of individual learners. By using generative AI, the system creates interactive and adaptive learning experiences that make education more engaging, accessible, and efficient.

### *Key Features*

* **Concept Explainer** – Simplifies complex topics into clear, easy-to-understand explanations.
* **Quiz Generator** – Creates practice questions and assessments dynamically based on selected topics.
* **Adaptive Learning** – Adjusts to the student’s progress and performance for a customized experience.
* **Lightweight Deployment** – Runs on Google Colab using Gradio, ensuring minimal setup and accessibility without requiring advanced infrastructure.

### *Benefits*

* **For Students**: Offers personalized guidance, interactive revision tools, and instant feedback.
* **For Teachers**: Reduces workload by automating quizzes, notes, and routine explanations.
* **For Self-Learners**: Provides a flexible, self-paced learning companion that supports independent study.

### *Relevance*

EduTutor AI is more than just a prototype; it is a scalable framework that can evolve with additional features like voice-based tutoring, multilingual support, and integration with existing learning platforms. By combining modern AI with practical deployment tools, EduTutor AI sets the foundation for the future of personalized, technology-driven education.

## 3.Features

EduTutor AI is designed with a set of robust features that aim to simplify learning and make it more interactive, engaging, and adaptive. Each feature has been carefully designed to align with the needs of students and educators while leveraging the power of IBM Granite models for generative AI.

### *Concept Explainer*

This feature enables learners to ask questions on any topic and receive simple, clear, and step-by-step explanations. The AI adapts its explanation style based on the learner’s level, whether beginner, intermediate, or advanced. By breaking down complex ideas into manageable concepts, students gain deeper understanding without relying solely on textbooks.

### *Quiz Generator*

EduTutor AI can generate quizzes automatically from chosen topics or study materials. Quizzes can include multiple-choice questions, short answers, or fill-in-the-blanks. This feature allows learners to practice actively and test their knowledge immediately after studying. Teachers can also use it to quickly create assessment materials.

### *Adaptive Learning*

The system tracks a learner’s performance and progress across sessions. Based on quiz results, previous questions, and interaction history, EduTutor AI recommends customized practice questions, explanations, and study paths. This ensures that each learner receives a unique, tailored experience that evolves over time.

### *Deployment on Google Colab*

EduTutor AI is designed to run seamlessly on Google Colab with minimal setup. Learners and developers do not need powerful local hardware; instead, they can leverage Colab’s T4 GPU to run AI models efficiently. This ensures accessibility to anyone with an internet connection.

### *IBM Granite Model Integration*

At the core of EduTutor AI lies IBM Granite models hosted on Hugging Face. These models bring natural language understanding and generation capabilities to the platform, making features like explanations, quiz generation, and feedback highly effective. Granite’s efficiency ensures fast responses, while its adaptability provides relevant, context-aware content.

## 

## 4.System Architecture

The architecture of EduTutor AI is organized into multiple layers, each responsible for a specific function, working together to provide a seamless personalized learning experience. The design is lightweight, modular, and cloud-friendly, making it suitable for academic as well as real-world usage.

### *User Interaction Layer*

The **frontend interface** is developed using **Gradio**, offering a clean and interactive environment for learners. Through this interface, students can ask questions, receive explanations, attempt quizzes, and view personalized suggestions. The focus is on simplicity, ensuring that even first-time users can navigate easily without technical knowledge.

### *Application Logic Layer*

This layer acts as the “brain” of the system. It processes requests coming from the interface, communicates with the AI models, and prepares structured responses for learners. It also manages quiz evaluations, tracks user progress, and coordinates the overall workflow of the application.

### *AI Intelligence Layer*

At the core of the system lie **IBM Granite models** integrated via Hugging Face. These models are responsible for tasks such as:

* Breaking down complex topics into simple explanations
* Generating customized quizzes on selected subjects
* Summarizing educational resources for quick understanding  
  The modular nature of this layer ensures that updated or additional AI models can be integrated without disrupting other components.

### *Data Handling Layer*

EduTutor AI supports storage and retrieval of learning materials, generated quizzes, and student progress. Currently, lightweight storage solutions such as Google Drive or CSV files are used, but the system can be easily extended to cloud databases like Firebase or MongoDB for large-scale adoption.

### *Deployment Layer*

The project runs on **Google Colab**, utilizing the **T4 GPU** for high-performance AI processing. This eliminates the need for powerful local hardware and ensures that the system is accessible from anywhere with internet connectivity. For collaboration and version control, the project is also hosted on GitHub.

### Workflow Overview

1. The learner interacts with the **Gradio interface**.
2. The **Application Logic Layer** processes the request.
3. The **AI Intelligence Layer** (IBM Granite) generates responses.
4. If required, the **Data Handling Layer** stores or retrieves information.
5. The final output is delivered back to the learner instantly.

## 5. Technology Stack

EduTutor AI has been built using a carefully selected set of tools and technologies that ensure scalability, efficiency, and ease of use. The stack combines modern AI frameworks, cloud-based deployment environments, and user-friendly development tools to provide an optimal learning experience.

### *Programming Language*

* **Python 3**: Python is the backbone of EduTutor AI. Its extensive library support, simple syntax, and strong AI/ML ecosystem make it ideal for implementing machine learning models, data processing, and application logic.

### *AI/ML Frameworks*

* **IBM Granite Models (Hugging Face Integration)**: Granite models provide state-of-the-art natural language understanding and generation. They are used for explaining concepts, generating quizzes, and summarizing learning content.
* **Transformers & Torch**: Hugging Face’s Transformers library and PyTorch framework are utilized for model loading, fine-tuning, and inference.

### *Frontend / User Interface*

* **Gradio**: A lightweight, web-based UI framework that allows learners to interact with EduTutor AI. It provides simple input fields, buttons, and response displays, making the application accessible to non-technical users.

### *Backend / Application Logic*

* **Google Colab Environment**: The system runs in Google Colab, offering access to free GPUs (T4) for efficient AI processing. It eliminates the need for powerful local machines and ensures easy setup for students and developers.

### *Data Handling*

* **CSV/Google Drive Integration**: For lightweight storage, EduTutor AI uses CSV files and Google Drive integration. These handle quiz data, student progress records, and project files. The design is modular and can be extended to cloud databases like Firebase or MongoDB.

### *Version Control*

* **Git & GitHub**: Version control is managed through Git, with GitHub serving as the central repository. This enables team collaboration, project backup, and easy deployment tracking.

### *Deployment & Accessibility*

* **Google Colab Notebooks**: Acts as the main development and deployment platform. Users can run the project by simply opening the notebook, installing dependencies, and executing the code.
* **Gradio Share Links**: Once deployed, the system generates a public Gradio link, allowing learners to access the application directly through their browser.

## 6.Setup Instructions

Setting up EduTutor AI is simple and requires minimal technical knowledge. The project has been designed to run primarily on **Google Colab**, eliminating the need for powerful local hardware. By following the steps below, users can quickly deploy and interact with the system.

### *Prerequisites*

* **Python 3.8 or above** installed (if running locally).
* Basic knowledge of **Google Colab** environment.
* Access to **Hugging Face** account for IBM Granite models.
* A **GitHub account** for version control and project uploads.

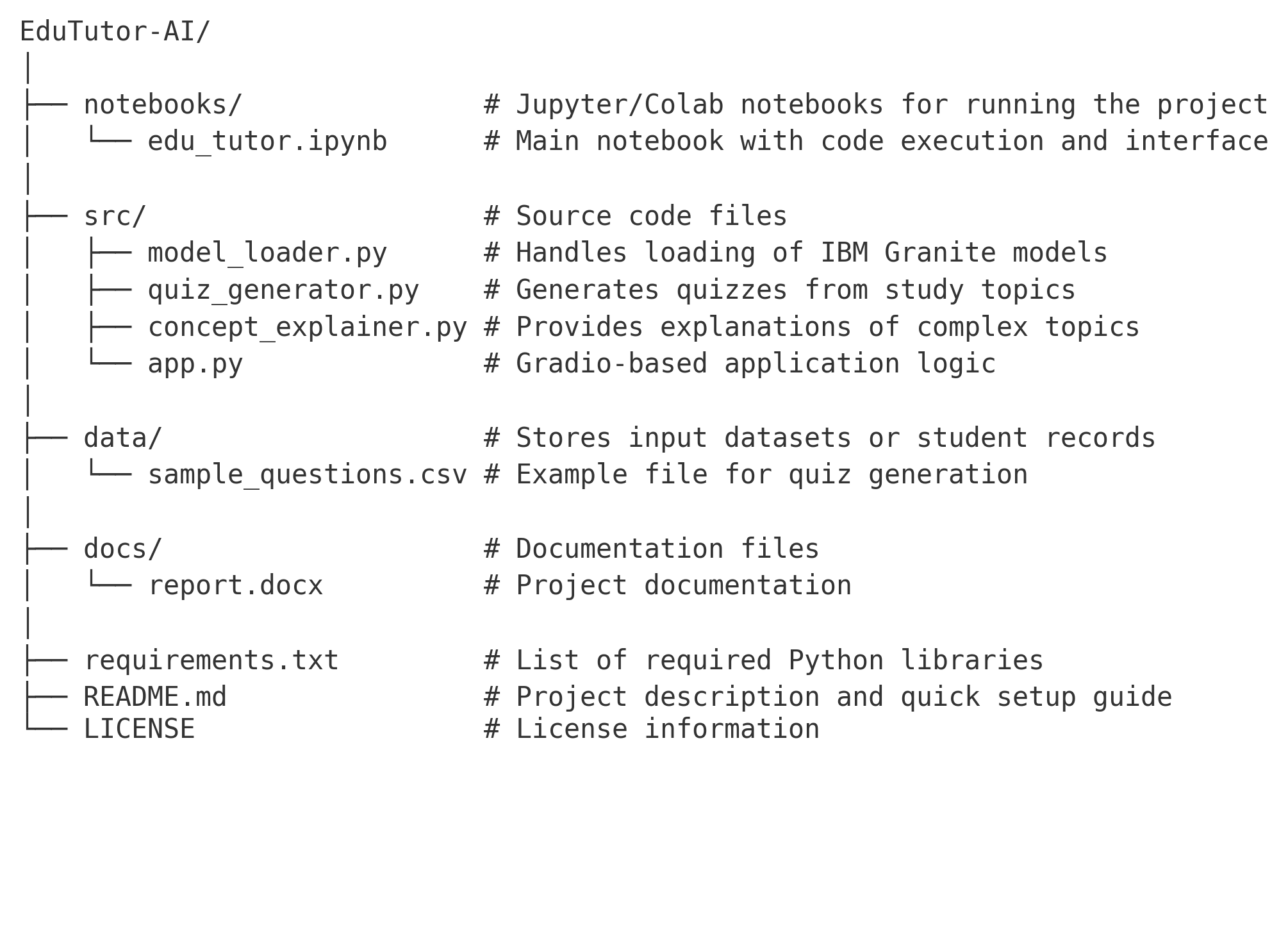
### *Installation Steps*

1. **Open Google Colab**: Go to [Google Colab](https://colab.research.google.com/) and create a new notebook.
2. **Set Runtime**: Navigate to Runtime > Change Runtime Type and select **T4 GPU** for faster model execution.
3. **Install Dependencies**: Run the following command in the first cell:
4. !pip install transformers torch gradio -q
5. **Integrate IBM Granite Model**: Use Hugging Face to access and load the required Granite model (e.g., granite-3.2-2b-instruct).
6. **Access Interface**: Once the setup is complete, click on the generated **Gradio share link** to interact with EduTutor AI.

## 

## 7.Folder Structure

The EduTutor AI project follows a simple and modular folder structure to ensure clarity, easy navigation, and scalability. Each folder and file has a specific purpose, making it easier for developers, students, and educators to understand and contribute.



This structure ensures separation of concerns:

* **notebooks/** for experimentation,
* **src/** for core application logic,
* **data/** for datasets, and
* **docs/** for project documentation.

Such an arrangement allows the project to remain organized, collaborative, and easy to maintain as it grows.

## 8.Workflow Explanation

The workflow of EduTutor AI outlines the step-by-step process through which the system delivers personalized learning experiences. Each stage ensures smooth interaction between the user, the application, and the underlying AI models.

### *Step 1: User Interaction*

Learners begin by accessing the **Gradio interface** hosted on Google Colab. They can either enter a question, request a concept explanation, or choose to generate quizzes.

### *Step 2: Request Processing*

The input is received by the **application layer**, which validates the request and passes it to the AI model. At this stage, the system also checks if the input relates to concept explanation, quiz generation, or summarization.

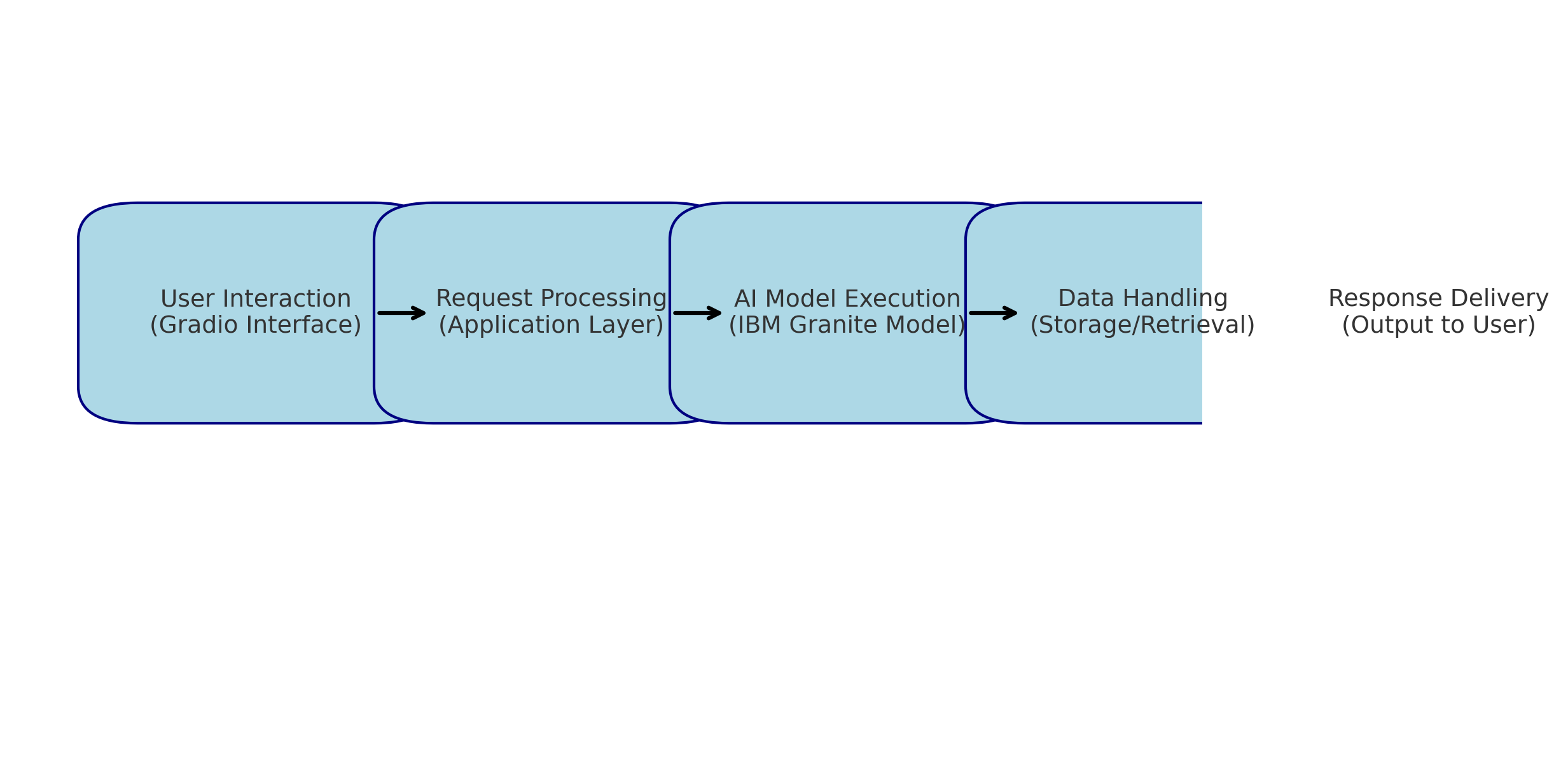
### *Step 3: AI Model Execution*

The request is processed by the **IBM Granite model** integrated through Hugging Face. Depending on the task, the model either explains the concept, creates quizzes, or generates summaries.

### *Step 4: Data Handling*

If required, the system retrieves relevant datasets (such as stored quiz questions) or saves newly generated content. This ensures that user progress and generated resources can be reused later.

Finally, the processed output is displayed to the learner through the Gradio interface. The user receives immediate feedback in the form of explanations, practice questions, or summarized notes.



This streamlined workflow ensures that EduTutor AI functions as a responsive, adaptive, and learner-friendly platform, bridging the gap between traditional study methods and AI-powered personalized learning

## 9. API Documentation

EduTutor AI provides a set of simple yet powerful APIs that enable interaction between the user interface, backend logic, and AI models. These APIs ensure that requests are processed efficiently and outputs are generated in real time. The endpoints are lightweight, making them suitable for academic use as well as future scalability.

### *Available Endpoints*

1. **POST /ask-question**
   * **Description**: Accepts a user query (concept explanation or doubt) and returns a response generated by the IBM Granite model.
   * **Input**: JSON containing {"question": "Explain Newton’s First Law"}
   * **Output**: AI-generated explanation in plain text.
2. ***POST /generate-quiz***
   * **Description**: Creates quizzes dynamically from a given topic or uploaded dataset.
   * **Input**: JSON containing {"topic": "Photosynthesis", "num\_questions": 5}
   * **Output**: List of multiple-choice or short-answer questions.
3. ***POST /summarize-content***
   * **Description**: Summarizes long study material into concise notes.
   * **Input**: JSON containing raw text or document link.
   * **Output**: Key bullet points or short summary.
4. ***POST /upload-data***
   * **Description**: Allows uploading of CSV files or documents to generate custom quizzes and explanations.
   * **Input**: File upload (CSV, TXT, or PDF).
   * **Output**: Confirmation of upload and availability for quiz generation.
5. ***GET /progress-report***
   * **Description**: Returns a basic progress report for a learner (if tracking is enabled).
   * **Output**: JSON summary with attempted quizzes, performance, and recommendations.

### *Key Features of APIs*

* Built with **Python (FastAPI/Flask)** for lightweight REST functionality.
* Easy integration with **Gradio frontend** for seamless interaction.
* Modular structure to allow future addition of endpoints (e.g., voice queries, multilingual support).

## 10. User Interface (UI/UX)

The user interface of EduTutor AI is designed to be simple, interactive, and accessible to learners of all backgrounds. Since the system is deployed using **Gradio on Google Colab**, users can access it directly through a browser without installing any additional software. The design emphasizes clarity and ease of use, ensuring that students and teachers can focus on learning rather than navigating complex menus.

### *Key Elements of the Interface*

1. **Input Box** – Allows users to type their queries, such as asking for explanations or requesting quizzes.
2. **Action Buttons** – Options to choose specific tasks like Explain Concept, Generate Quiz, or Summarize Content.
3. **Output Panel** – Displays the AI-generated responses, including detailed explanations, quiz questions, or short summaries.
4. **File Upload Option** – Enables users to upload documents (PDF/CSV/TXT) for custom content generation.
5. **Shareable Link** – A unique Gradio share link is generated, making the interface accessible to others without requiring installation.

### *Design Principles*

* **Simplicity**: Minimalistic design with only essential components.
* **Accessibility**: Runs on any device with an internet connection.
* **Responsiveness**: Real-time feedback ensures smooth interaction.
* **Scalability**: The layout can be extended with additional features such as voice-based input or multilingual support.

## 11. Testing

Testing was an essential part of ensuring that EduTutor AI functions reliably across different scenarios. Since the system is designed to handle user queries, generate quizzes, and summarize content, a combination of **unit testing, functional testing, and user-level testing** was carried out.

### *Types of Testing Performed*

1. **Unit Testing**
   * Verified individual modules such as quiz generator and concept explainer.
   * Ensured that each function produced the expected output for different inputs.
2. **Functional Testing**
   * Tested the overall application flow in Google Colab.
   * Checked whether input queries were correctly passed to the IBM Granite model and whether appropriate responses were generated.
3. **Interface Testing**
   * Verified that the **Gradio interface** accepted inputs, displayed outputs, and handled file uploads without errors.
   * Ensured that the layout was user-friendly and responsive.
4. **Performance Testing**
   * Evaluated response times when generating explanations or quizzes.
   * Confirmed that the system ran efficiently with Google Colab’s T4 GPU.
5. **User Acceptance Testing (UAT)**
   * A small group of students and educators interacted with the system.
   * Feedback was collected on usability, clarity of responses, and overall effectiveness.

### *Results*

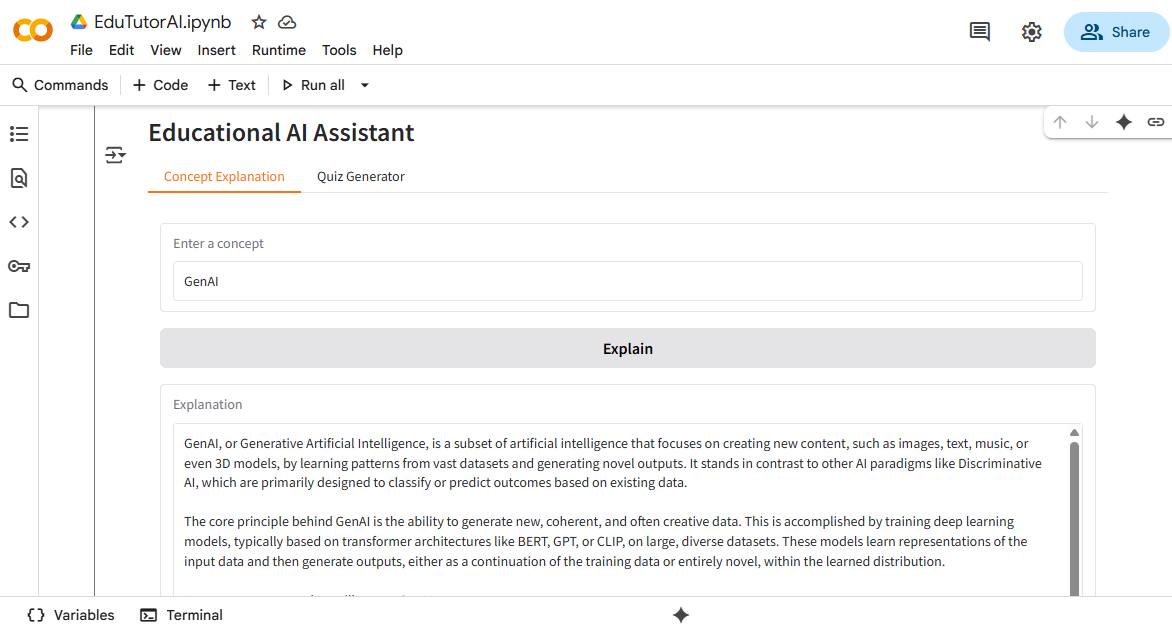
The testing phase confirmed that EduTutor AI consistently produced accurate explanations, generated quizzes as expected, and summarized study materials effectively. Minor improvements, such as refining quiz formats and adding error handling for invalid inputs, were identified as areas for enhancement.

## 12. Screenshots

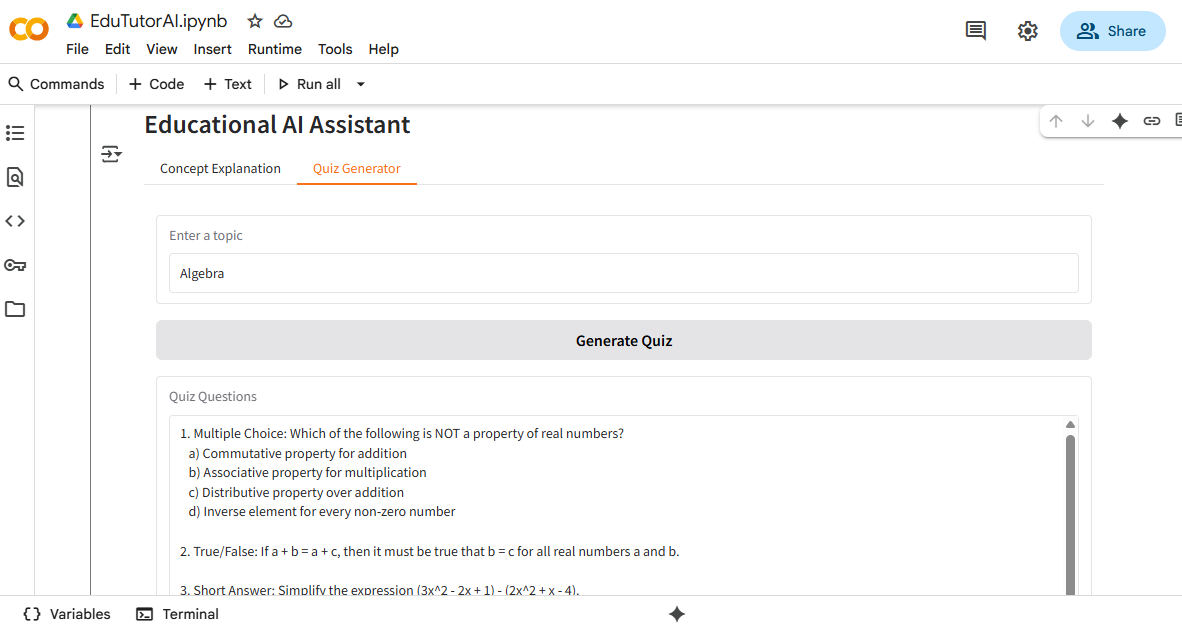
The following screenshots demonstrate the functionality of **EduTutor AI** and highlight its interactive features.

### *Concept Explanation*

The screenshot below shows the **Concept Explanation** tab in action. A learner has entered the topic “GenAI” into the input box, and the system generates a clear, simplified explanation using the IBM Granite model. This feature allows students to quickly understand complex ideas in a concise and learner-friendly manner.



### 🔹 Quiz Generator

The second screenshot (Fig. 2) displays the **Quiz Generator** tab. In this example, the topic “Algebra” was entered, and the system automatically generated a variety of questions, including multiple-choice, true/false, and short-answer types. This feature supports active learning by allowing students to test their knowledge immediately after studying a topic.

These visuals confirm that EduTutor AI successfully provides both concept-level learning support and practice-based assessments, fulfilling its goal of creating a personalized and interactive learning experience.

**13. Known Issues and Limitations**

While EduTutor AI demonstrates strong potential as a personalized learning assistant, certain limitations and challenges were identified during development and testing. These issues highlight areas where future improvements are needed:

### *Known Issues*

1. **Response Variability** – Since the system relies on generative AI models, explanations or quizzes may vary slightly each time, which can sometimes lead to inconsistent phrasing.
2. **Limited Error Handling** – The application may fail or return incomplete responses if a user provides malformed inputs or unsupported file formats.
3. **Dependency on Internet Connectivity** – The system is designed to run on Google Colab and Hugging Face APIs, which require a stable internet connection. Any disruption may cause the interface to stop functioning.

### *Limitations*

1. **Restricted Deployment** – Currently, the project runs on Google Colab, which is not suitable for large-scale or production-level usage.
2. **Hardware Constraints** – Although Colab offers GPU access, performance may slow down during peak usage or with larger datasets.
3. **Single-Language Support** – At present, the system primarily supports English. Learners requiring multi-language explanations may not be fully supported.
4. **Basic User Tracking** – The system does not yet include detailed analytics or long-term student progress tracking, limiting its ability to provide deeper insights.

These issues do not undermine the project’s core functionality but serve as indicators for future enhancement and optimization.

## 14. Future Enhancements

EduTutor AI has been designed as a foundation for personalized and adaptive learning. While the current version offers core features such as concept explanation, quiz generation, and summarization, there is significant potential to expand its capabilities in future iterations. Some planned improvements include:

### *Multilingual Support*

Enable learners to interact with the system in multiple languages, making it more inclusive for non-English speakers and international users.

### *Voice-Based Interaction*

Integrate speech recognition and text-to-speech modules so that students can ask questions verbally and receive spoken explanations, improving accessibility.

### *Mobile Application*

Develop a dedicated mobile app to provide on-the-go access, ensuring students can use EduTutor AI conveniently outside of traditional classroom or desktop environments.

### *Advanced Analytics*

Incorporate dashboards that track student performance, highlight areas of improvement, and provide progress reports for teachers and learners.

### *Integration with Learning Management Systems (LMS)*

Connect EduTutor AI with popular LMS platforms (such as Moodle or Google Classroom) to deliver seamless support within existing educational ecosystems.

### *Expanded Content Types*

Extend functionality to support diagrams, interactive simulations, and subject-specific practice modules for STEM and non-STEM disciplines.

By implementing these enhancements, EduTutor AI can evolve into a comprehensive and scalable solution that not only supports individual learners but also integrates seamlessly into classrooms and institutional frameworks.

## 15. Conclusion

EduTutor AI represents a step forward in making education more **personalized, accessible, and adaptive** through the integration of generative AI. By leveraging **IBM Granite models**, the system provides students with simplified explanations, generates quizzes dynamically, and summarizes study materials in an interactive manner. The lightweight deployment on **Google Colab with Gradio** ensures that learners and educators can use the platform without significant technical or infrastructure requirements. Although the current version has certain limitations—such as language support and dependency on internet access—the project demonstrates strong potential for growth. With planned enhancements such as multilingual features, voice-based tutoring, and integration with Learning Management Systems (LMS), EduTutor AI can evolve into a comprehensive educational companion. Ultimately, this project not only supports students in self-paced learning but also empowers teachers by automating repetitive academic tasks, thus contributing to a more efficient and learner-centric education system.

## 16. References

1. IBM Granite Models – Hugging Face. Available at: <https://huggingface.co/ibm-granite>
2. Gradio Documentation – Building Machine Learning Interfaces. Available at: <https://www.gradio.app/guides/>
3. Google Colab – Using GPUs in Colab. Available at: <https://colab.research.google.com/>
4. Python Official Documentation. Available at: <https://docs.python.org/3/>
5. Git Documentation – Version Control System. Available at: <https://git-scm.com/docs/git>
6. Naan Mudhalvan Smart Internz Portal. Available at: <https://naanmudhalvan.smartinternz.com/>