

# **EduTutor AI: Personalized Learning with Generative AI and LMS Integration**

The structure and style will mirror the sample doc you provided (with sections like Introduction, Project Overview, Architecture, Setup Instructions, Folder Structure, Running the Application, API Documentation, User Interface, Testing, Known Issues, Future Enhancements).

- EduTutor AI: Personalized Learning with Generative AI and LMS Integration

## **1. Introduction:**

- Project Title: EduTutor AI: Personalized Learning with Generative AI and LMS Integration
- Team Member:A.Kishothpana
- Team Member:T.Nivedha
- Team Member:K.Lavanya

## **2. Project Overview:**

- Purpose:

EduTutor AI is an AI-powered educational assistant designed to personalize learning for students and educators. It integrates Generative AI capabilities with Learning Management Systems (LMS) to enhance the teaching and learning experience. The system supports concept explanations and quiz generation, enabling learners to grasp difficult topics more easily and test their understanding in real time. By leveraging IBM Granite's advanced LLM, the application provides interactive, adaptive, and student-centered learning experiences that can be accessed through a simple web interface.

- Features:

## **1. Concept Explanation**

- Key Point: Provides detailed explanations of academic concepts.
- Functionality: Accepts a user-defined concept (e.g., "machine learning") and generates clear explanations with examples.

## **2. Quiz Generator**

- Key Point: Generates practice quizzes for better retention.
- Functionality: Creates 5 quiz questions on a given topic (multiple choice, true/false, short answer), with an answer key provided at the end.

## **3. AI-Powered Insights**

- Key Point: Uses IBM Granite LLM for adaptive learning.

Functionality: Generates context-aware explanations and questions tailored to user inputs.

## **4. User-Friendly Interface**

- Key Point: Provides a simple, tab-based interactive UI.
- Functionality: Built with Gradio, featuring tabs for explanations and quizzes, with instant output.

## **5. Accessible Deployment**

- Key Point: Runs locally or can be shared publicly.

- **Functionality:** Deployed using Gradio with `share=True`, enabling easy demonstrations and student access.

### **3. Architecture:**

The architecture of EduTutor AI is designed to provide a seamless, AI-driven learning experience that integrates with educational workflows. It consists of four major layers: Frontend (Gradio), Backend (Python), LLM Integration (IBM Granite LLM), and Deployment Layer.

#### **1. Frontend (Gradio)**

Technology Used: Gradio (Blocks & Tabs).

Purpose: Provides an intuitive, web-based interface for students and teachers.

Design:

Concept Explanation Tab → Accepts a topic and displays an AI-generated explanation.

Quiz Generator Tab → Accepts a topic and returns 5 AI-generated quiz questions with answers.

Benefits:

Easy-to-use, interactive design.

No technical skills required from the user.

Tabbed navigation allows switching between features without confusion.

#### **2. Backend (Python Application Logic)**

Technology Used: Python 3.9+, Torch, Transformers.

Purpose: Manages the processing logic and orchestrates communication between user input, the AI model, and the interface.

Core Functions:

1. `generate_response(prompt, max_length)` → Handles prompt creation, tokenization, model inference, and decoding of responses.

2. `concept_explanation(concept)` → Builds a structured explanation prompt, ensuring clarity and detail.

3. `quiz_generator(concept)` → Creates quizzes with multiple question types and returns them with an answer key.

Error Handling: Manages edge cases such as empty inputs, excessively long queries, or unavailable GPU.

### **3. LLM Integration (IBM Granite LLM)**

Model Used: `ibm-granite/granite-3.2-2b-instruct`.

Integration Method: Hugging Face Transformers library.

Key Steps:

Tokenization: User input is converted into tokens.

Model Inference: Tokens are processed by the Granite LLM to generate contextual responses.

Decoding: Tokens are decoded back into human-readable text.

Optimizations:

Runs on GPU (CUDA) if available for faster response.

Falls back to CPU for wider compatibility.

Uses `temperature=0.7` and `do_sample=True` for controlled creativity.

Advantages:

Provides natural, conversational explanations.

Generates varied quiz questions for better assessment.

### **4. Deployment Layer**

Powered By: Gradio's hosting mechanism.

Modes of Deployment:

Local Deployment: Runs on `http://127.0.0.1:7860` when executed locally.

Public Access: With `share=True`, generates a temporary public link for student/teacher collaboration.

Scalability:

Can be hosted on cloud platforms for integration into real-world LMS systems.

Supports classrooms, online tutoring, and remote education scenarios.

#### **4. Setup Instructions:**

Prerequisites

Python 3.9 or later

pip & virtual environment tools

Hugging Face transformers, torch, gradio libraries installed

#### **Installation**

1. Clone the repository

```
git clone <your_repo_link>
```

```
cd edututor-ai
```

2. Install dependencies

```
pip install -r requirements.txt
```

3. Run the application

```
python edututorai.py
```

#### **5. Folder Structure:**

EduTutorAI/

edututorai.py - # Main application script

requirements.txt - # Dependencies

README.md - # Project documentation

data/ - # (Optional) Sample input/output

outputs/ - # (Optional) Generated quizzes/explanations

#### **6. Running the Application:**

1. Start the app:

```
python edututorai.py
```

2. Gradio launches a local server → <http://127.0.0.1:7860>

3. If share=True, a public link is generated.

4. Navigate between tabs:

Concept Explanation Tab → Enter a concept → Get detailed explanation.

Quiz Generator Tab → Enter a topic → Receive 5 quiz questions + answers.

## **7. API Documentation (Future Expansion):**

Planned API endpoints for LMS integration:

POST /explain-concept → Input: concept, Output: detailed explanation.

POST /generate-quiz → Input: topic, Output: quiz + answers.

GET /lms-integration → Retrieve personalized content recommendations from LMS.

## **8. User Interface:**

**Tab Layout:**

- Concept Explanation → Text input + button → Explanation output.
- Quiz Generator → Text input + button → Quiz output.
- Design: Minimal, student-friendly, accessible on browser.
- Interaction: Real-time AI responses.

## **9. Testing:**

- Unit Testing: For functions `generate_response()`, `concept_explanation()`, `quiz_generator()`.
- Manual Testing: UI tested by entering various concepts/topics.

**Edge Cases:**

- Long concepts truncated properly.
- Invalid/empty inputs handled gracefully.

## **10. Screenshots:**

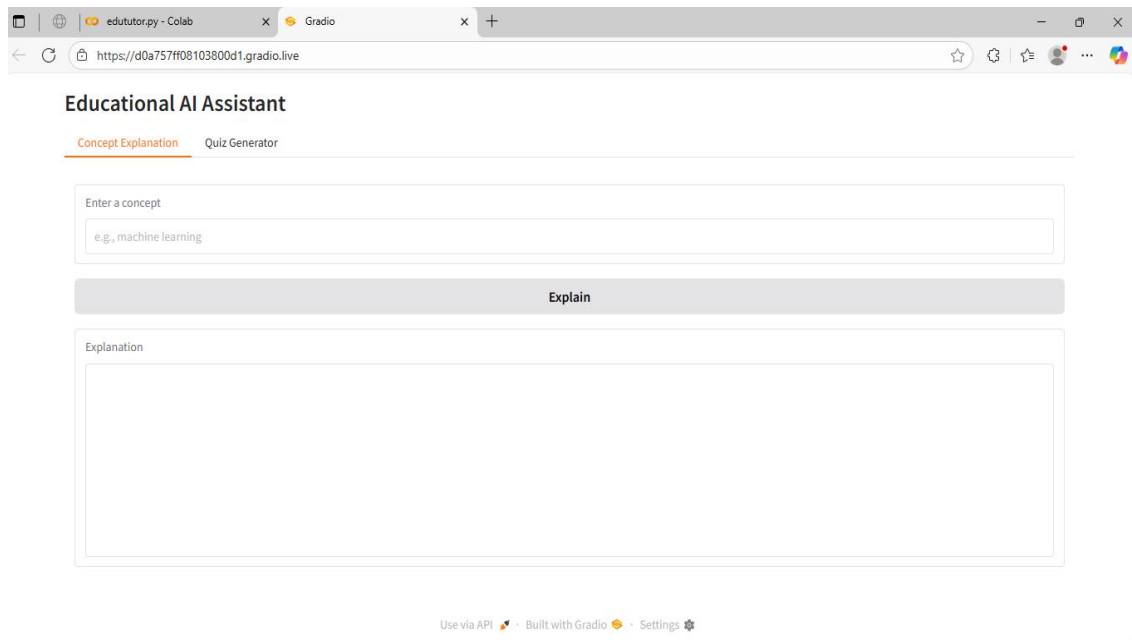


Fig 1.educational ai assistant

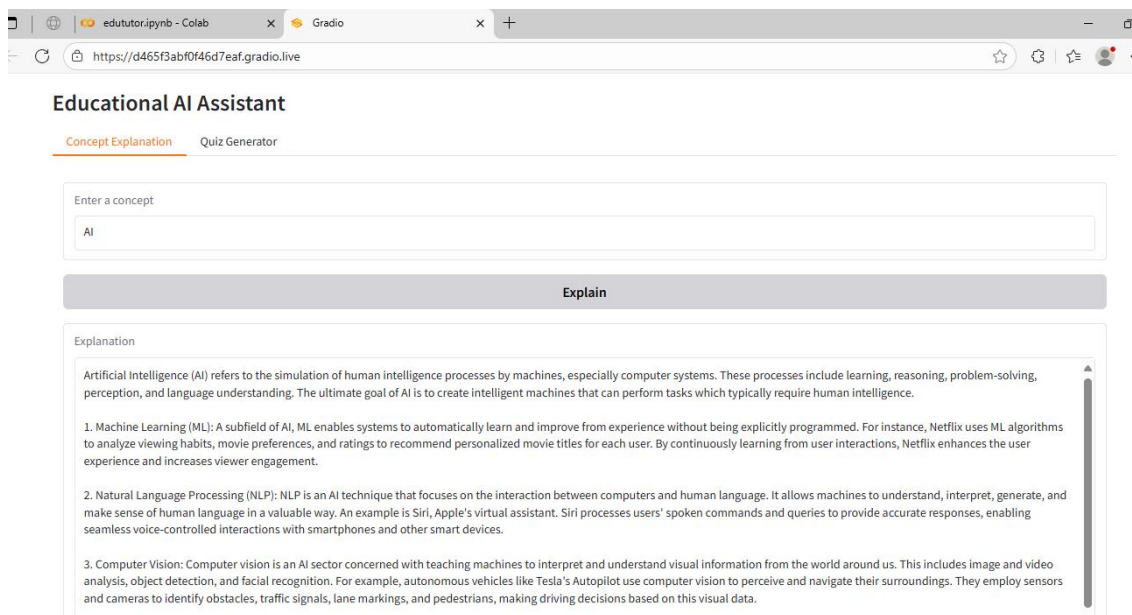


Fig 2.concept explanation

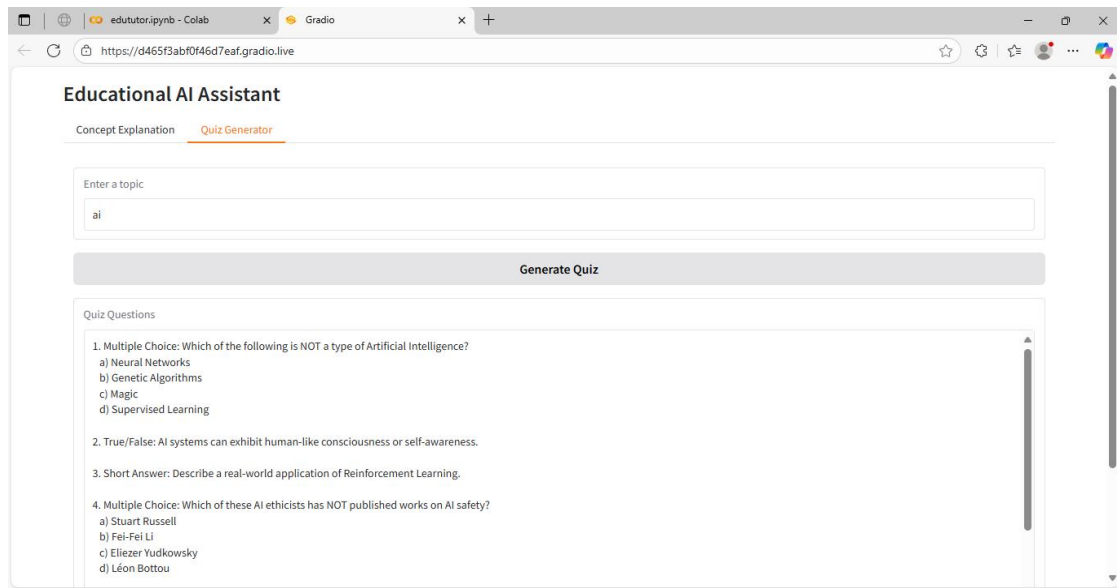


Fig 3.ai-quiz generator

## 11. Known Issues:

- Long responses may get truncated due to max token limits.
- First-time model load may cause latency.
- No authentication in current version.
- Limited to text input (no multimedia integration yet).

## 12. Future Enhancements:

LMS Integration: Seamless syncing with Moodle, Canvas, Google Classroom.

Multimedia Content: Support for diagrams, video explanations.

Adaptive Learning: Personalized pathways based on student progress.

Authentication & Roles: Teacher/student logins with history tracking.

Cloud Deployment: Host on scalable cloud platforms for real-world classrooms.