

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

---

## **Table of Contents**

1. Introduction .....	2
2. Project Overview.....	2
Scenarios .....	2
Project Flow .....	3
3. Architecture .....	3
4. Setup Instructions .....	3
5. Milestones .....	4
6. Screenshots & Output.....	4
7. Known Issues.....	7
8. Future Enhancements .....	7
9. Testing .....	5

## 1.Introduction

- Project title : EduTutor AI: Personalized Learning with Generative AI and LMS Integration
- Team leader : Maddasani Nithya Sree
- Team member : Madhuri.K
- Team member : Monika.K
- Team member : Monisha.K

## 2. Project Overview:

EduTutor AI is an AI-powered personalized education platform that revolutionizes the way students learn and educators assess progress. It provides dynamic quiz generation, student evaluation, Google Classroom integration, and real-time feedback—all powered by IBM Watsonx and Granite foundation models. Designed with modular architecture, this platform streamlines personalized education and enhances learning outcomes for students across academic levels.

### Scenarios:

#### Scenario 1: Personalized Learning Experience

A student logs into EduTutor AI and synchronizes their courses using their Google Classroom credentials. The platform analyzes course data, generates quizzes on key topics using the Granite LLM, and assesses responses for instant feedback.

#### Scenario 2: Educator Dashboard & Performance Insights

Educators can log in to view real-time quiz performance of all students.

The dashboard highlights quiz history, scores, last topics attempted, and insights fetched from the Pinecone vector database.

#### Scenario 3: Diagnostic Testing and Adaptive Quizzing

Students undergo a diagnostic test generated by IBM Watsonx models. Based on the results, the platform adapts quiz difficulty and topic relevance.

#### Scenario 4: Google Classroom Integration

EduTutor AI syncs courses directly from Google Classroom, enabling automatic quiz topic generation and curriculum alignment.

#### Project Flow:

- User Input: Students log in using credentials or Google Classroom and request a quiz by selecting topic and difficulty.
- AI Quiz Generation: Watsonx+Granite models generate MCQs and store answers securely.
- User Quiz Submission: Students submit answers, backend evaluates and stores results in Pinecone DB.
- Feedback Loop: Educators access real-time analytics via dashboard.

### 3. Architecture

- FastAPI Backend – Manages login, quiz generation, evaluation, classroom sync, and database operations.
- Watsonx + Granite Models – AI model for question generation and adaptive learning.
- Pinecone Vector DB – Stores embeddings and quiz history.
- Streamlit Frontend – Role-based UI for students and educators.

## 4. Setup Instructions

### **Prerequisites:**

- fastapi
- uvicorn
- langchain\_ibm
- pinecone
- streamlit
- google-auth-oauthlib
- google-api-python-client
- python-dotenv

**Installation:** pip install -r requirements.txt

Environment Variables (.env):

WATSONX\_MODEL\_ID=granite-13b-instruct-v2

WATSONX\_API\_KEY=your\_ibm\_watsonx\_api\_key

WATSONX\_ENDPOINT=https://us-south.ml.cloud.ibm.com

WATSONX\_PROJECT\_ID=your\_project\_id

PINECONE\_API\_KEY=your\_pinecone\_api\_key

PINECONE\_INDEX\_NAME=edututor

## 5. Milestones

### **Milestone 3:** AI Integration with IBM Watsonx

- Model Setup: Granite model loaded via langchain\_ibm.WatsonxLLM.
- Prompt Template: Quiz questions generated dynamically.
- Quiz Parsing: Structured JSON output for display and evaluation.

## Milestone 5: Pinecone Vector DB Integration

- Stores user profiles with embeddings.
- Updates quiz metadata after submission.
- Enables performance analysis for educators.

## Milestone 6: Streamlit Frontend UI

- Student panel with login, quiz dashboard, and history.
- Educator panel with class-level analytics.

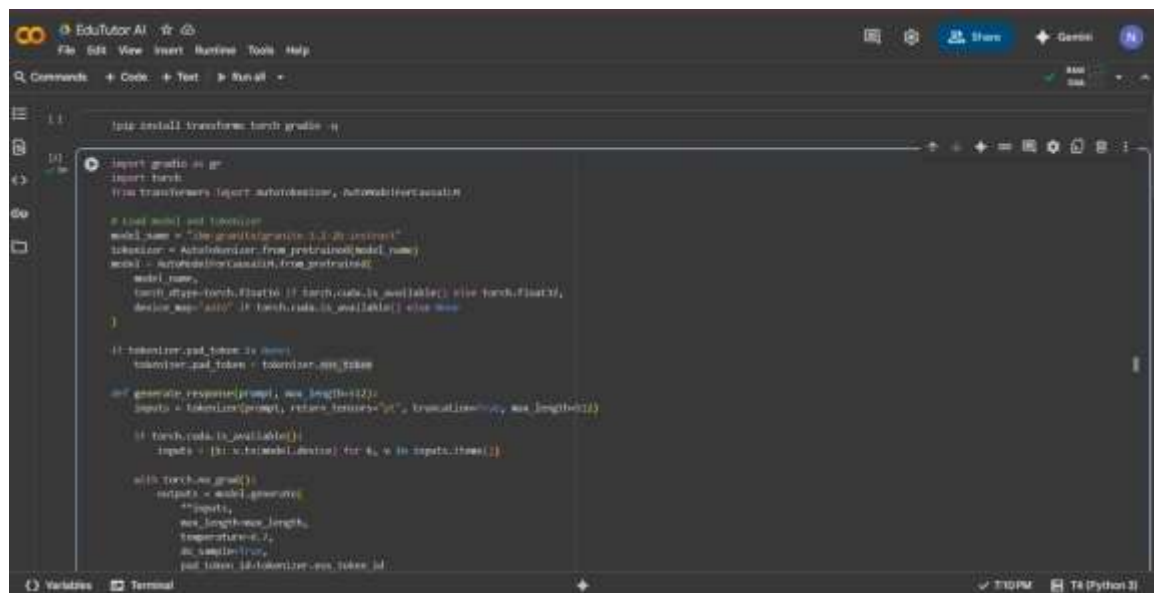
## Milestone 7: Functional Verification

- Verified backend, AI, and database integration using sample test data.

## 6. Project Screenshots:

This is the output of the Edututor AI

Step 1: Code of the Edututor AI

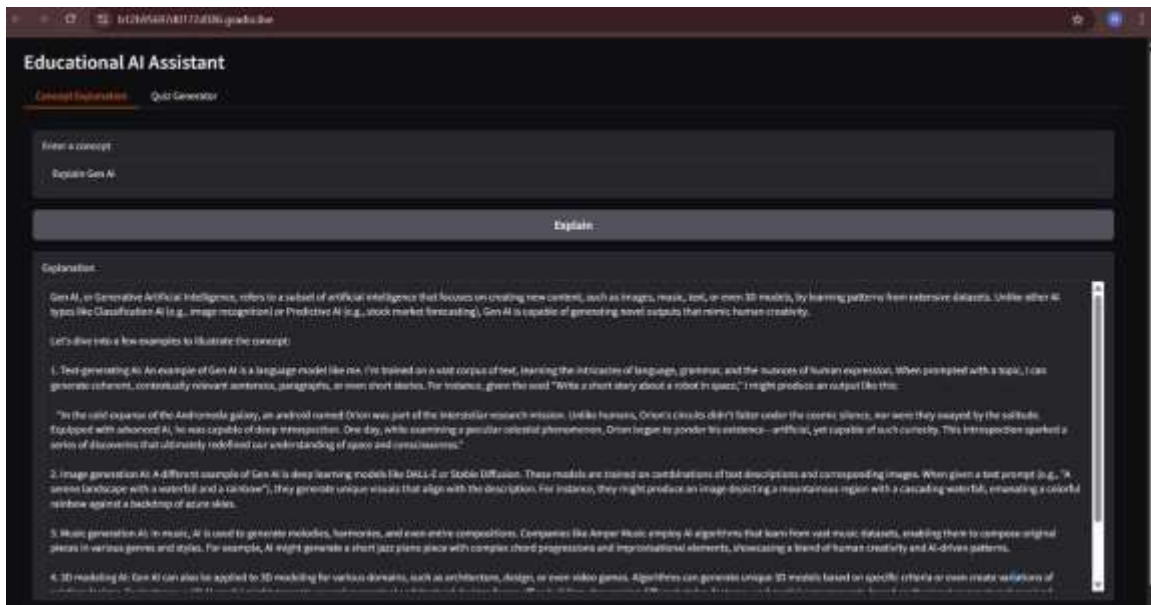


```
1: pip install transformers torch torchvision\n\n2: import gradio as gr\nimport torch\nfrom transformers import AutoTokenizer, AutoModelForCausalLM\n\n3: # Load model and tokenizer\nmodel_name = \"the_paul_github/1.1.0-1000000\"\ntokenizer = AutoTokenizer.from_pretrained(model_name)\nmodel = AutoModelForCausalLM.from_pretrained(\n    model_name,\n    torch_dtype=torch.float16 if torch.cuda.is_available() else torch.float32,\n    device_map='auto' if torch.cuda.is_available() else None\n)\n\n4: if tokenizer.pad_token is None:\n    tokenizer.pad_token = tokenizer.eos_token\n\n5: def generate_response(prompt, max_length=128):\n    inputs = tokenizer(prompt, return_tensors='pt', truncation=True, max_length=128)\n\n    if torch.cuda.is_available():\n        inputs = inputs.to(model.device) for k, v in inputs.items()\n\n    with torch.no_grad():\n        outputs = model.generate(\n            **inputs,\n            max_length=max_length,\n            temperature=0.7,\n            do_sample=True,\n            pad_token_id=tokenizer.eos_token_id\n        )
```

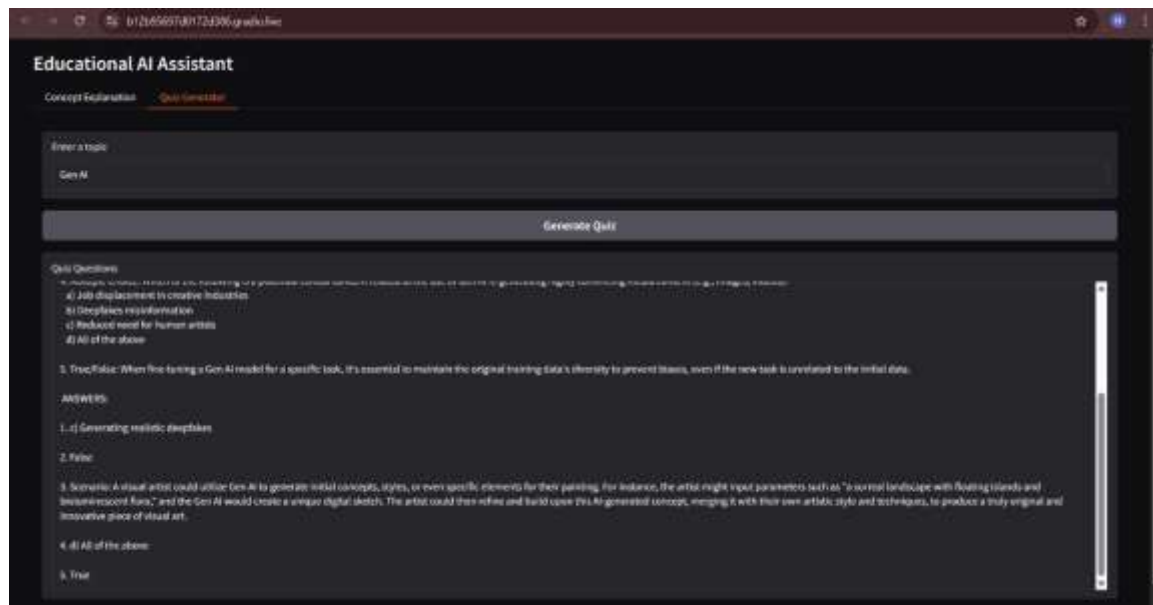
## Step 2: Public url of edututor AI



### Step 3: AI-Powered Concept Explanation Screen Output



### Step 4: AI-Powered Quiz Generator Screen Output



## 7. Known Issues

- Streamlit frontend not fully implemented.
- Some errors occur during quiz result display.
- UI responsiveness needs improvement.
- Google Classroom sync occasionally times out.

## 8. Future Enhancements

- Add full-fledged mobile app support for Android/iOS.
- Improve adaptive learning using reinforcement learning techniques.
- Integrate push notifications for quiz reminders.
- Enhance UI/UX with better visualizations.
- Add multi-language support for regional students.
- Include gamification features like leaderboards and badges.

## 9. Testing

- Unit testing for API routes.
- Manual testing for login, quiz generation, and submission.
- Edge case testing for invalid input, large quiz sets, and network issues.