GENERATIVE AI WITH IBM CLOUD

Project Documentation

1. Introduction Project Title:

EduTutor AI – Personalized Learning with Generative AI and LMS Integration

Team Members:

- **Team Leader :** Dega Homesh
- **Team member :** Pagadala Bala Srinivas Kumar

---

2. Project Overview Purpose:

EduTutor AI is a lightweight educational tool designed to simplify learning for students and self-learners. It leverages Generative AI to enhance skills through interactive quizzes, helping users improve their knowledge and boost self-development.

Features:

- AI-based concept explanations
- Language learning in English.
- Topic-based quiz generation
- User login and registration
- Session tracking
- Quiz history

---

3. Architecture

Frontend (UI Layer):

- Built using **Gradio**, a Python-based UI framework
- Tabs and blocks are used to manage interface layout

- Users can log in ,and select a quiz topic and choose difficulty levels before atteming the quiz.

Backend (Application Logic):

- Entire backend logic is written in **Python**
- Integrates directly with **Hugging Face API** using the IBM Granite 3.3-2BInstruct model

Storage & Sessions:

- No database used currently
- Uses Python **dictionary** (user_sessions) to track session state and quiz attempts

---

4. Setup Instructions

Prerequisites:

- Python 3.10+
- VS code OR Jupyter Notebook
- Hugging Face account & API token **Installation Steps:**

1. Open the VS code or clone the repo
2. Install dependencies:
3. !pip install gradio transformers PyPDF2
4. Set your Hugging Face token in code:
5. from huggingface_hub import login
6. login('your-huggingface-api-token')
7. Run the notebook or Python file to start the app

---

5. Folder Structure

Since the project runs entirely in a .py file, there is no separate folder structure. However:

- main.py or notebook contains:

- All logic for concept

understanding, quiz generation,

and login

- Gradio UI layout
- Gradio UI layout

---

6. Running the Application

VS CODE:

- Open the notebook
- Run all cells
- A Gradio public link is generated (shared via share=True) **Local Machine:**
- Save code as edututor.py
- Run with:
- python edututor.py
- Open Gradio interface at http://localhost:7860

---

7. API Documentation

This project does not expose APIs directly (like REST), but functions are triggered internally via Gradio.

Function	Description
concept_understanding()	Returns AI explanation for a topic
language_learning()	Returns grammar and basics of selected language
generate_test_from_pdf()	Parses PDF and generates quiz
quiz_generator()	Generates topic-based multiple-choice questions
Function	**Description**
login_fn()	Handles login logic

run_all()	Main execution combining all features per user

---

8. Authentication

- Basic authentication using Python dictionary (users_db)
- Login tab asks for username/password
- No encryption used (basic educational demo)

---

9. User Interface

- Built using **Gradio Tabs and Blocks** □ Tabs include: Login, Register, Classroom
- Input components:
  - Textbox for concept/topic ○ Radio buttons for language ○ File upload for PDFs
- Output components:
  - Textbox displaying AI results ○ Auto-formatted quiz

---

10. Testing

- Manual functional testing of: ○ Concept explanation generation ○ PDF quiz output accuracy ○ Language grammar content ○ Login and registration validation
- Output was validated against known concepts and PDFs

---

11. Screenshots or Demo

Example Outputs:

- Concept explanation of "Generative AI"
- Grammar basics in Hindi
- Quiz from uploaded academic PDF
- Topic-based MCQs

📽️ **Demo Link**:
https://drive.google.com/file/d/1yM-XRP--orc7EiML8U9gTENLvBt5txV7/view?usp=drivesdk

---

12. Known Issues

- No persistent user session (resets on restart)
- Cannot evaluate user quiz answers yet
- Output from PDF depends on PDF formatting and clarity

---

13. Future Enhancements

- Add quiz scoring and feedback
- Integrate Firebase or MongoDB for storing sessions
- Add quiz result analytics
- LMS integration (like Moodle)
- Use Whisper or STT for voice input
