GENERATIVEAIWITHIBMCLOUD

**ProjectDocumentation**

# Introduction Project Title:

**EduTutor AI – Personalized Learning with Generative AI and LMS Integration Team Members:**

# TeamLeader : Sd Shalu

**Teammember:** Purilla Mahidhar

**Teammember:** Polireddi Karthik

**Teammember:** Ravulapalli Komali

# Project Overview Purpose:

EduTutor AI is a light weight educational tool that’s implifies learning for students and self-learners. It uses GenerativeAI to provide concept explanations, language grammar tutorials, and auto-generated quizzes from uploaded PDFs.

# Features:

* + AI-based concept explanations
  + Language learning in English
  + Quiz generator
  + Topic-based quiz generation
  + Topic Summarizer

# Architecture

**Frontend(UILayer):**

* + Built using **Gradio**,a Python-based UIframework
  + Tabs and blocks are used to manage interface layout
  + Users can login,enter input,select language,and upload PDFs through a clean UI

# Backend(ApplicationLogic):

* + Entire backend logic is writtenin **Python**
  + Functions handle input,parse PDFs using PyPDF2,and format prompts
  + Integrates directly with **Hugging FaceAPI** using the IBM Granite3.3-2B- Instruct model

# Storage&Sessions:

* + No database used currently
  + Uses Python **dictionary**(user\_sessions)to track session state and quiz attempts

# Setup Instructions Prerequisites:

* + Python3.10+
  + VScode
  + Hugging Face account & API token

# Installation Steps:

1. Create virtual environments

cd fastapi\_app

python -m venv venv

venv\Scripts\activate

source venv/bin/activate

cd ../streamlit\_app

python -m venv venv

venv\Scripts\activate

source venv/bin/activate

1. Install dependencies

pip install -r requirements.txt

pip install -r requirements.txt

1. Run the FastAPI server

uvicorn main:app –reload

1. Run the Streamlit app

streamlit run app.py

# 5.Folder Structure

project\_root/

├── credentials/

│ └── watsonx.json

│

├── fastapi\_app/

│ ├── \_\_pycache\_\_/

│ ├── venv/

│ ├── main.py

│ ├── utils.py

│ └── requirements.txt # Dependencies for FastAPI

│

├── streamlit\_app/

│ ├── components/ # Custom Streamlit components

│ │ ├── \_\_pycache\_\_/

│ │ ├── Chatbot.py

│ │ ├── CodeHelper.py

│ │ ├── QuizGenerator.py

│ │ └── TopicSummarizer.py

│ │

│ ├── venv/ # Virtual environment for Streamlit

│ ├── app.py # Streamlit main app entry point

│ ├── requirements.txt # Dependencies for Streamlit

│ └── styles.css # Custom styles for Streamlit

# 6.Running the Application

cd fastapi\_app

python -m venv venv

venv\Scripts\activate

source venv/bin/activate

cd ../streamlit\_app

python -m venv venv

venv\Scripts\activate

source venv/bin/activate

pip install -r requirements.txt

pip install -r requirements.txt

uvicorn main:app –reload

streamlit run app.py

# 7. API Documentation:

This project does not expose APIs directly(likeREST),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 |
|  |  |
| quiz\_generator() | Generates topic-based multiple-choice  questions |

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |

# Authentication

* + Basic authentication using Python dictionary(users\_db)
  + Login tabasks for username/password
  + Noencryption used(basiceducationaldemo)

# UserInterface

Builtusing **GradioTabsand Blocks**

Tabsinclude:Login,Register,Classroom

Input components:

* + 1. Textbox for concept/topic
    2. Radio buttons for language
    3. File upload for PDFs

Output components:

* + 1. Textbox displaying AI results
    2. Auto-formatted quiz

# Testing

Manual functional testing of:

* + 1. Concept explanation generation
    2. PDFquizoutput accuracy
    3. Language grammar content
    4. Login and registration validation

Output was validated against known concepts and PDFs

# Screenshots or Demo

Example Outputs:

Concept explanation of"GenerativeAI"

Grammar basics in English

Quiz from uploaded academic PDF

Topic-based MCQs

**DemoLink**:<https://drive.google.com/file/d/1x_MSNYBD9BwSRuH3z8BazQMSpmiTQmzC/view?usp=sharing>

# Known Issues

No persistent user session(resetsonrestart)

Can not evaluate user quiz answers yet

Output from PDF depends on PDF formatting and clarity

# Future Enhancements

Add quizs coring and feedback

Integrate Fire base or MongoDB for storings essions

Add quiz result analytics

LMS integration(like Moodle)

Use Whisper or STT for voice input