EduTutor AI: Personalized Learning with Generative AI and LMS Integration

# 1. Introduction

• Project Title: EduTutor AI – Personalized Learning with Generative AI

• Team Members:  
 - Challa Raghavendra – Frontend & Quiz Generator  
 - Ullikanti Khaleel Ahmad – Learning Modules & Library  
 - Sirasani Janardhan Reddy – AI Integration & Storage  
 - Gajavalli Manidhar – Educator Dashboard & Charts

# 2. Project Overview

• Purpose:  
 EduTutor AI offers personalized learning experiences using IBM’s Granite LLM. It helps students take adaptive quizzes, track progress, and allows educators to monitor performance.

• Features:  
 - AI-generated quizzes and learning modules  
 - Role-based user interface (student/educator)  
 - Dashboard analytics  
 - Quiz history and personalized feedback

# 3. Architecture

• Frontend: Built with Streamlit for real-time interaction and modular navigation.  
• Backend: Python logic integrates IBM Granite models and handles user roles.  
• Database: Local JSON-based file system for storing users and quiz results.

# 4. Setup Instructions

• Prerequisites:  
 - Python 3.10+  
 - Streamlit  
 - Huggingface Transformers  
 - IBM Granite API access

• Installation:  
 - Clone repo  
 - Run `pip install -r requirements.txt`  
 - Set API keys in `.env`  
 - Run `streamlit run app.py`

# 5. Folder Structure

• /Edututor\_ai\_project  
 - app.py (main app)  
 - core/quiz\_generator.py  
 - model\_setup.py  
 - users.json / quiz\_results.json  
 - .env

# 6. Running the Application

• Run the application using: `streamlit run app.py`

# 7. API Documentation

• Uses IBM Granite through `transformers` pipeline. Main interaction is via prompt generation for quizzes and modules.  
• Local file-based APIs for saving/loading results.

# 8. Authentication

• Role-based login (Student/Educator) using session state in Streamlit.  
• Stored credentials in users.json

# 9. User Interface

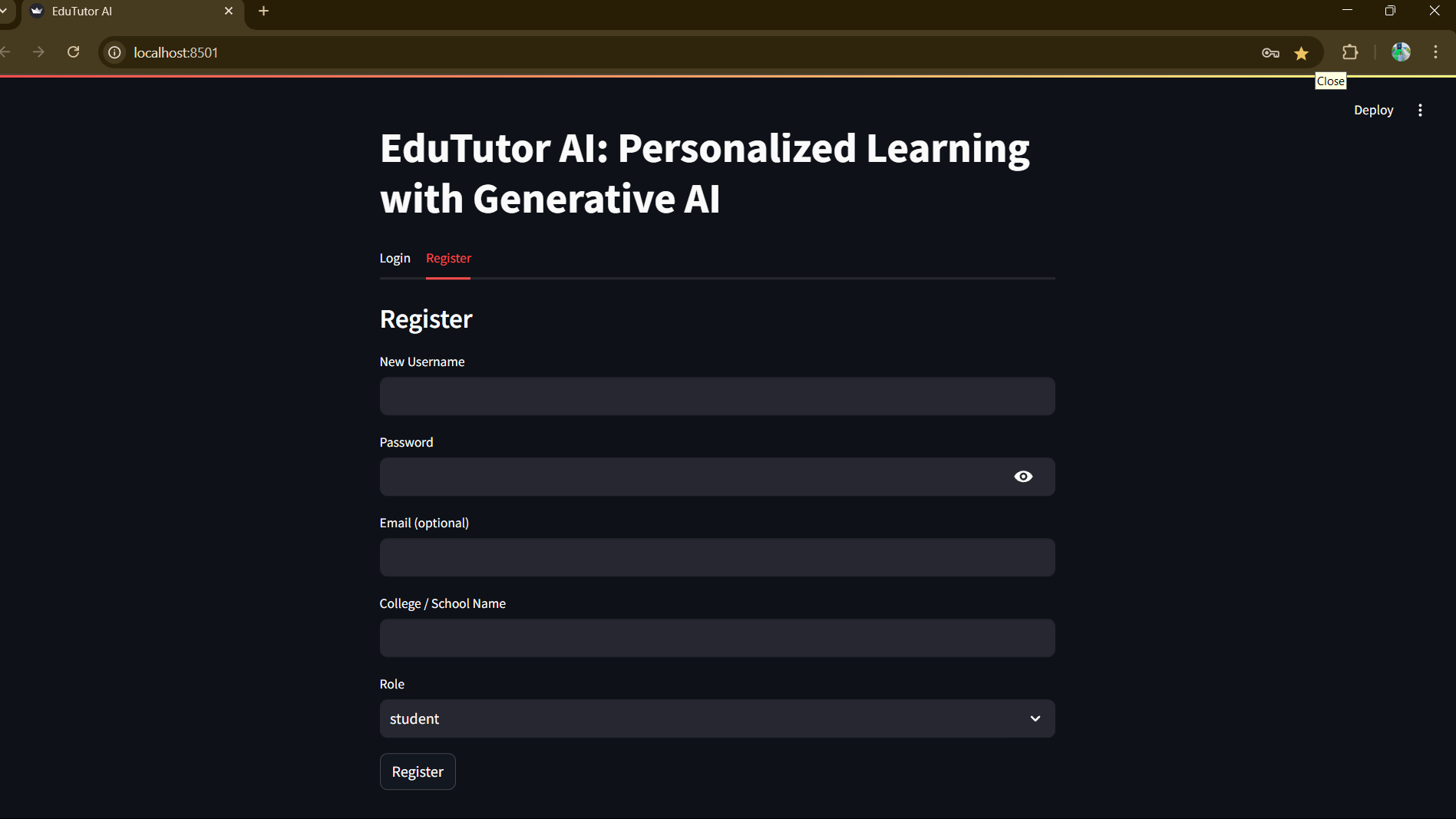
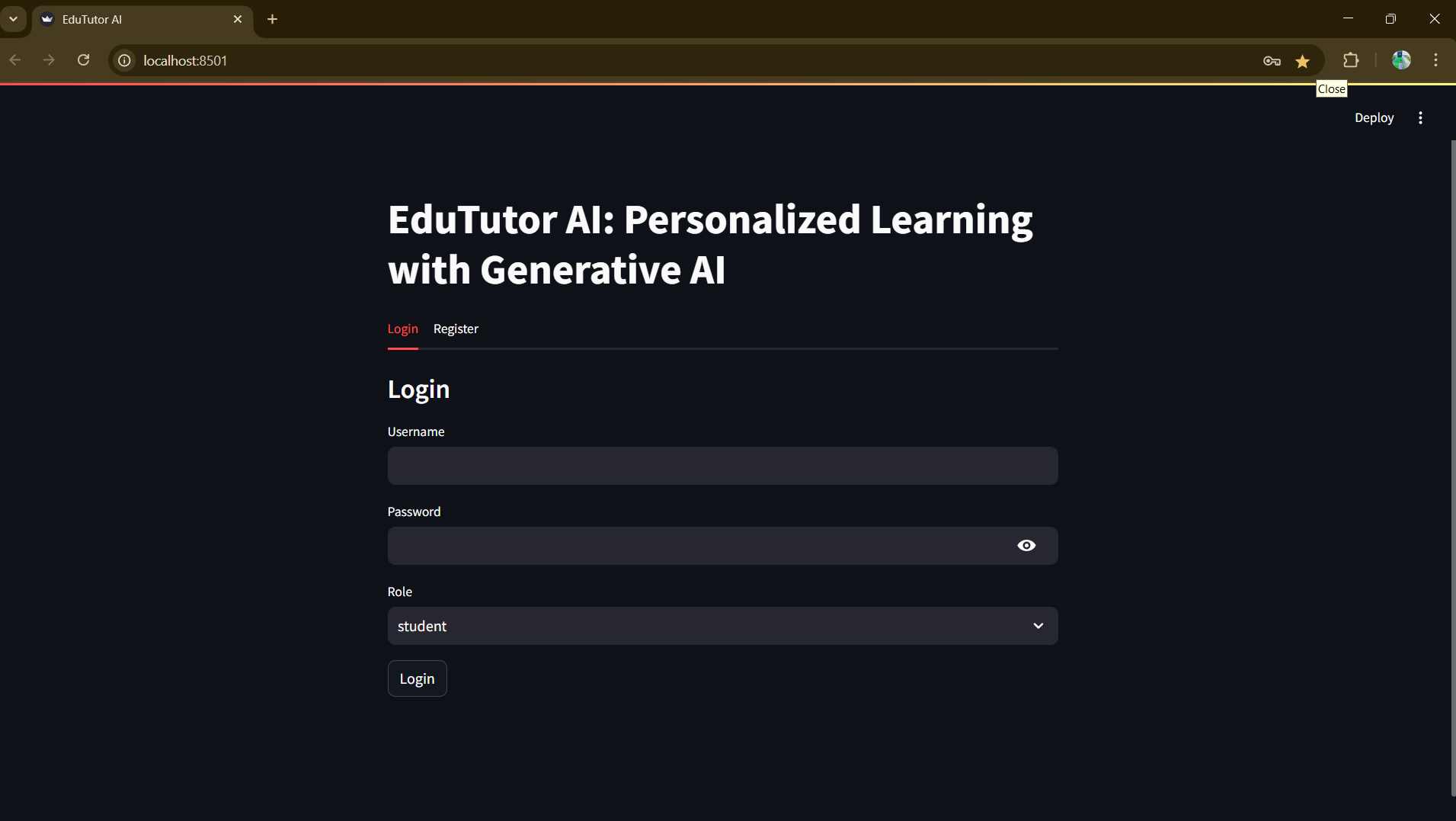
• Interactive tabs for login, quiz, modules, and dashboard.  
• Sidebar navigation for quick access.  
• Educator view shows analytics using Plotly.

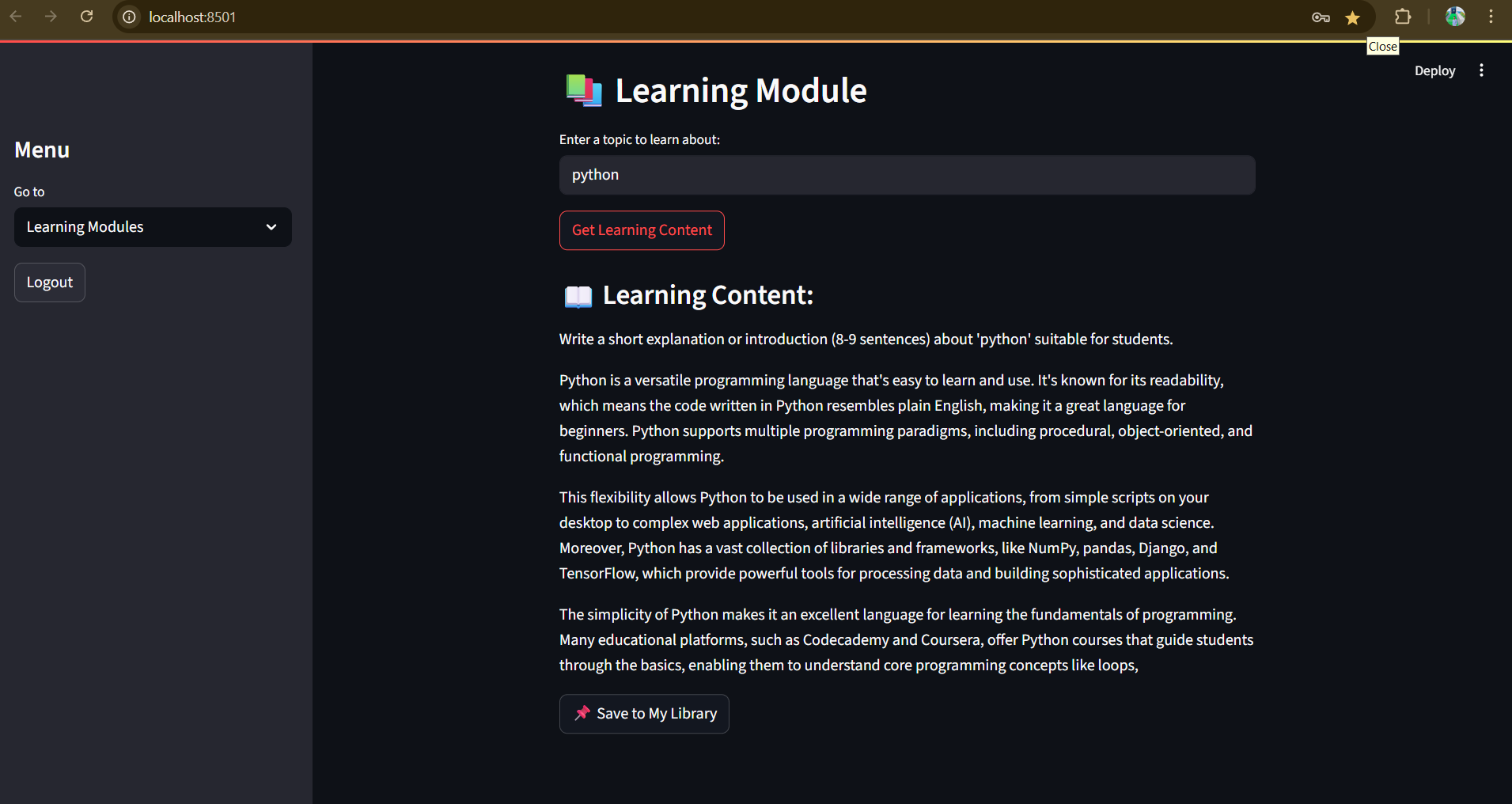
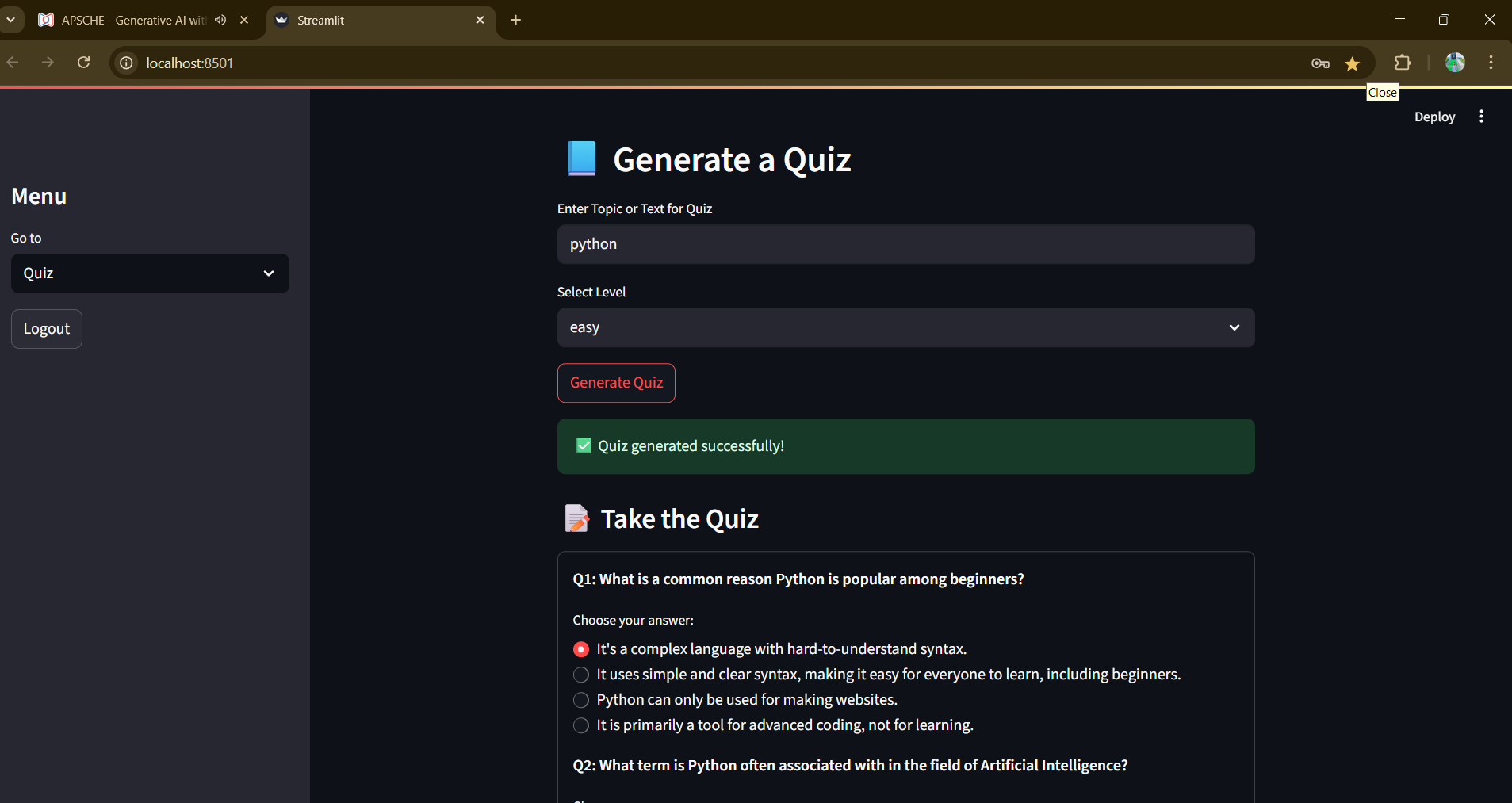
# 10. Testing

• Manual testing on quiz generation latency and role navigation.  
• Unit test functions used for model input/output validation.

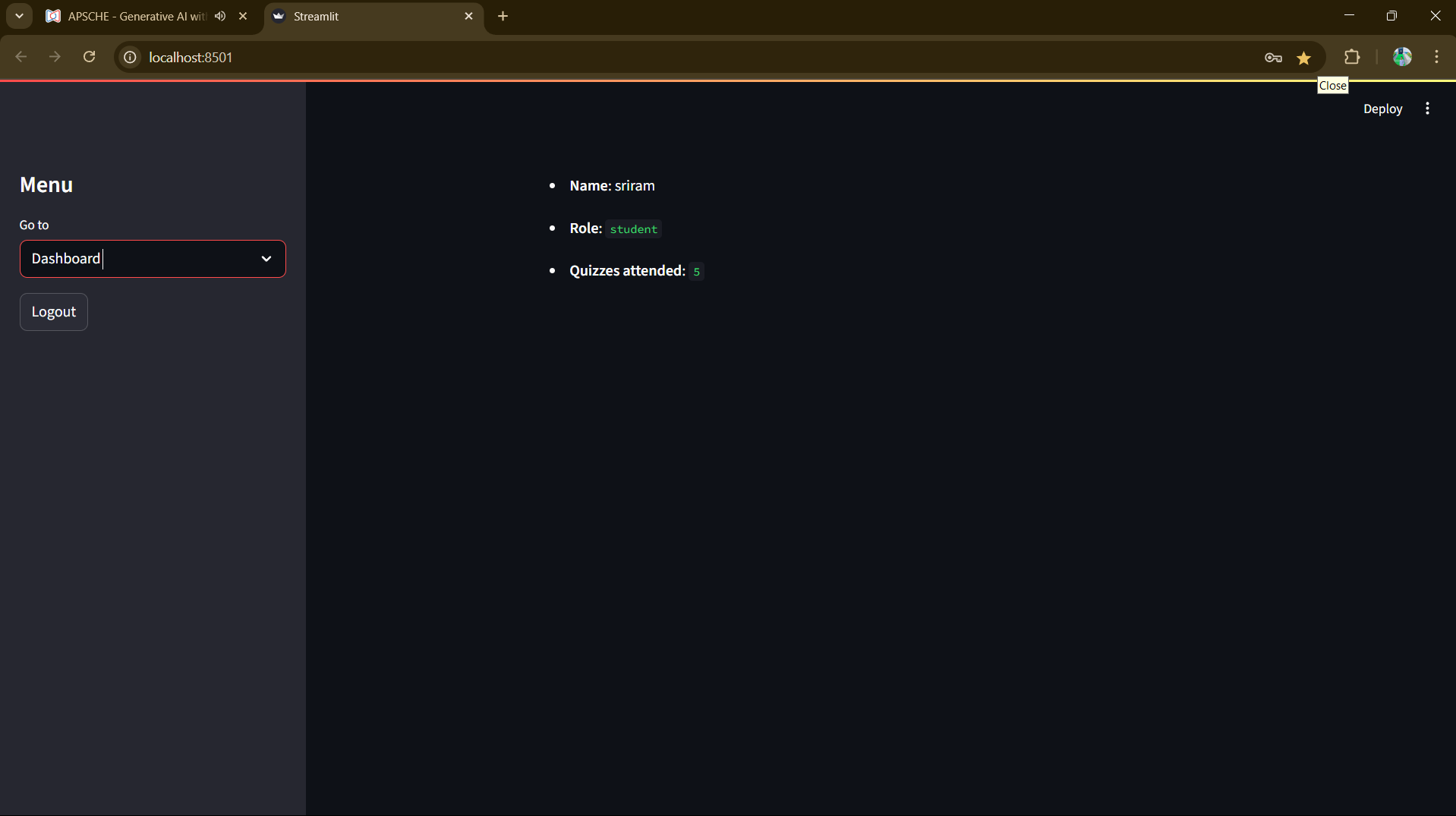
# 11. Screenshots or Demo

- Login/Register

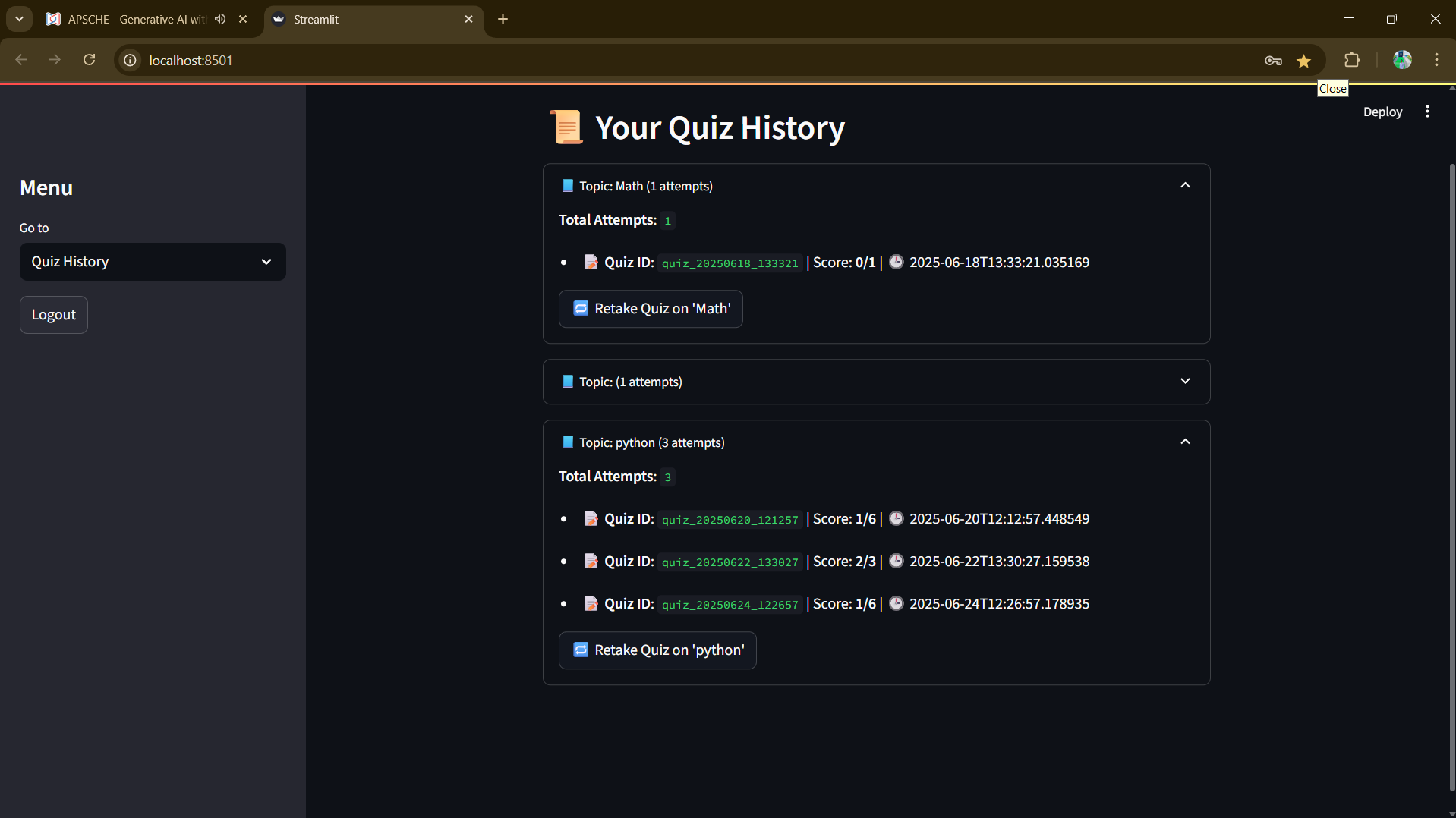
  
- Learning Module

  
- Quiz Generation

- Results Dashboard



-Quiz History



# 12. Known Issues

• Large model loading time  
• JSON storage not scalable for large teams  
• Occasional Streamlit rerender glitches

# 13. Future Enhancements

• Migrate to Firebase/PostgreSQL  
• Add voice-based interaction  
• Admin dashboard for class monitoring  
• Deploy on Hugging Face Spaces