EduTutor AI: Personalized Learning with Generative AI and LMS Integration

Project Documentation

1. Introduction

• Project title : EduTutor AI: Personalized Learning with Generative AI and LMS

Integration • Team member : Keerthana. G.U

• Team member : Taruni.K

• Team member : Surekha.J

• Team member : Shree Nidhi.S

2. Project Overview

• Purpose:

EduTutor AI aims to revolutionize education by providing highly personalized learning experiences through integration of generative AI and Learning Management Systems (LMS). It adapts content dynamically based on each learner's style, progress, and preferences, fostering engagement and effective knowledge retention. The system supports educators by automating routine tasks, generating custom assessments, and offering actionable insights for student performance.

• Features:

• Generative AI Tutor: Provides tailored explanations, answers questions, and creates custom learning content dynamically.

• Quiz Generator: Auto-generates quizzes, assignments, and feedback for continuous evaluation.

• LMS Integration: Seamlessly connects with LMS platforms for synchronization of student data, progress tracking, and course materials.

• Personalized Learning Paths: Curates individualized learning journeys based on real-time analytics.

• Interactive Assessments: Generates quizzes and evaluations to track learning outcomes.

• Analytics Dashboard: Provides insights on student engagement, strengths, and areas needing improvement.

• User-friendly Interface: Gradio-powered UI for concept explanation and quiz generation.

3. Architecture

Frontend: • Built with Gradio (for prototype in Google Colab).

Backend: • Python with Hugging Face Transformers and PyTorch for AI model inference.

LLM Integration: • Hugging Face model: ibm-granite/granite-3.2-2b-instruct. Data Flow:

1 User enters a concept or topic.

2 Prompt is sent to AI model.

3 AI generates explanation or quiz.

4 Response is displayed in UI.

4. Code Implementation

• Dependencies:

```
!pip install transformers torch gradio --quiet
```

• Helper function generate_response() for inference.

• Concept Explanation: Explains topics with examples.

• Quiz Generator: Creates quizzes with answers.

• Gradio UI: Two tabs for Concept Explanation and Quiz Generator.

5. Setup Instructions

• Open in Google Colab or local environment.

• Install required dependencies.

• Run notebook cells to start the app.

• Gradio launches with two tabs: Concept Explanation & Quiz Generator.

6. API Documentation (Future LMS Integration)

- POST /ai-tutor/query - Generate explanations.
- POST /assessments/generate - Create quizzes.
- GET /lms/courses - Fetch LMS course data.
- GET /analytics/progress - Retrieve student progress insights.

7. Testing

- Unit tests for AI response generation.
- Manual testing through Gradio interface.
- Planned API testing with Postman/Swagger.

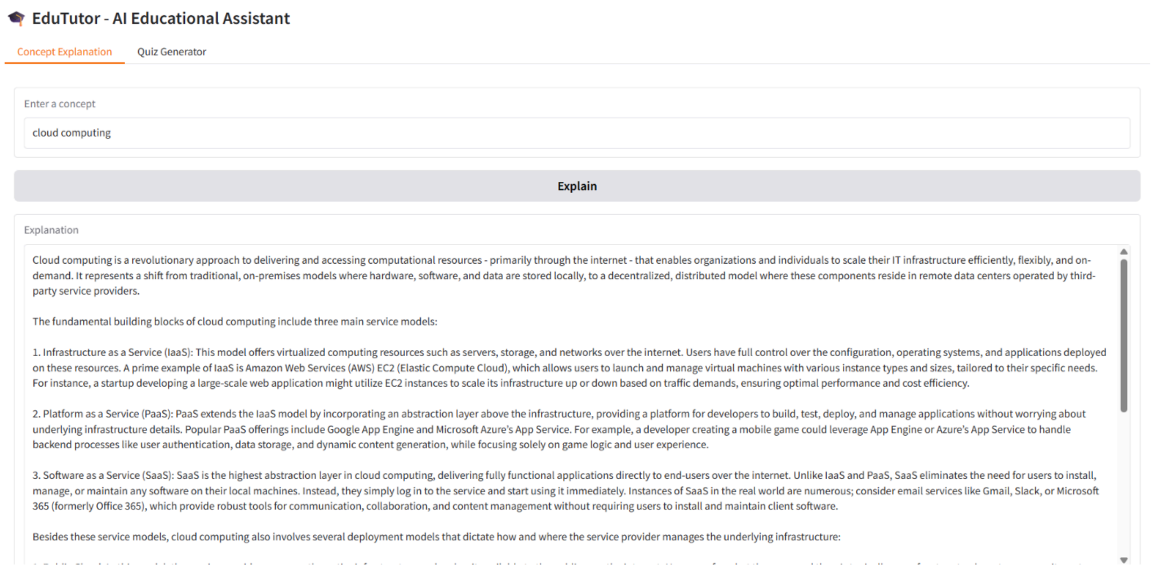
8. Known Issues and Future Enhancements

- Add LMS connectors (Moodle, Canvas, Blackboard).
- Enable voice and multimodal support.
- Enhance personalization using reinforcement learning.
- Expand to multi-language support.

9. User Interface

- Dashboard: Provides a simple and interactive landing page for students and teachers.
- Concept Explanation Tab: Textbox for entering concept → Button to generate explanation → Output panel with detailed explanation.
- Quiz Generator Tab: Textbox for entering topic → Button to generate quiz → Output panel with 5 varied questions and answers.
- Navigation: Implemented with Gradio Tabs for easy switching. Future: Side menu for more modules.
- Accessibility: Clean, minimal UI designed for both students and educators. Works in Colab with shareable links.

10. Screenshot



Enter a concept

cloud computing

Explanation

Cloud computing is a revolutionary approach to delivering and accessing computational resources - primarily through the internet - that enables organizations and individuals to scale their IT infrastructure efficiently, flexibly, and on-demand. It represents a shift from traditional, on-premises models where hardware, software, and data are stored locally, to a decentralized, distributed model where these components reside in remote data centers operated by third-party service providers.

The fundamental building blocks of cloud computing include three main service models:

1. Infrastructure as a Service (IaaS): This model offers virtualized computing resources such as servers, storage, and networks over the internet. Users have full control over the configuration, operating systems, and applications deployed on these resources. A prime example of IaaS is Amazon Web Services (AWS) EC2 (Elastic Compute Cloud), which allows users to launch and manage virtual machines with various instance types and sizes, tailored to their specific needs. For instance, a startup developing a large-scale web application might utilize EC2 instances to scale its infrastructure up or down based on traffic demands, ensuring optimal performance and cost efficiency.

2. Platform as a Service (PaaS): PaaS extends the IaaS model by incorporating an abstraction layer above the infrastructure, providing a platform for developers to build, test, deploy, and manage applications without worrying about underlying infrastructure details. Popular PaaS offerings include Google App Engine and Microsoft Azure's App Service. For example, a developer creating a mobile game could leverage App Engine or Azure's App Service to handle backend processes like user authentication, data storage, and dynamic content generation, while focusing solely on game logic and user experience.

3. Software as a Service (SaaS): SaaS is the highest abstraction layer in cloud computing, delivering fully functional applications directly to end-users over the internet. Unlike IaaS and PaaS, SaaS eliminates the need for users to install, manage, or maintain any software on their local machines. Instead, they simply log in to the service and start using it immediately. Instances of SaaS in the real world are numerous; consider email services like Gmail, Slack, or Microsoft 365 (formerly Office 365), which provide robust tools for communication, collaboration, and content management without requiring users to install and maintain client software.

Besides these service models, cloud computing also involves several deployment models that dictate how and where the service provider manages the underlying infrastructure: