EduTutorAI: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 :R.KRISHNAMOORTHY

• Team member :K.GUGAN

• Teammember :A.KANNAN

• Teammember :B.KATHIRVEL

**2. Project Overview**

**Purpose**

EduTutor AI aims to enhance the quality of education by creating **AI-driven personalized learning paths**, adaptive quizzes, intelligent tutoring, and seamless LMS integration.It not only enhances student engagement and performance but also empowers teachers with intelligent tools and supports administrators with real-time analytics.

**Features**

* AI-powered personalized learning recommendations
* LMS integration (Google Classroom, Moodle, etc.)
* Automated quiz generation and grading
* Interactive chatbot assistant for student queries
* Real-time analytics dashboard for performance tracking
* Secure authentication and role-based user management

3. Architecture

**AI Engine**: IBM Granite model (granite-3.2-2b-instruct) from Hugging Face

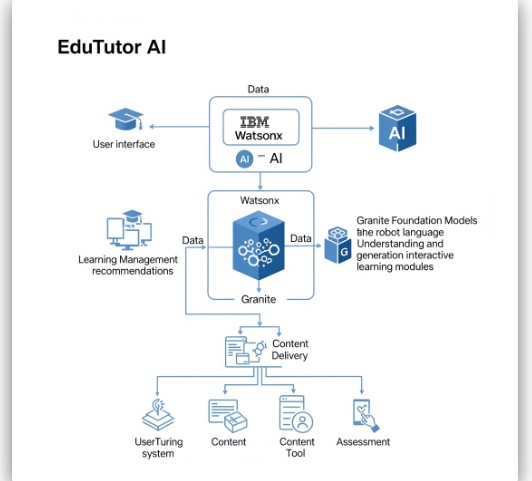
**Frontend**: Gradio (for Colab-based web interface) / React (for full UI)

**Backend**: Python APIs

**Database**: MongoDB/PostgreSQL (for storing users, progress, and results)

**Integration**: LMS APIs for syncing courses and assignments

**Deployment**: Google Colab with GPU (T4)



4. Setup Instructions

**Prerequisites**

* Python programming basics
* Hugging Face IBM Granite models
* Gradio framework knowledge
* Google Colab with GPU runtime
* GitHub account for version control

**Steps**

1. Clone repository:
2. git clone<https://github.com/krishnamoorthy6425/Edututor.git>
3. cd edututor-ai
4. Install dependencies:
5. pip install -r requirements.txt
6. npm install
7. Configure .env file with LMS API keys and Hugging Face access token.
8. Run migrations for database setup.
9. Start backend:

python app.py

1. Start frontend:

npm start

1. On Colab, install dependencies:

!pip install transformers torch gradio–q

1. Start backend and frontend servers.

5. Folder Structure

edututor-ai/

│── backend/

│ ├──api/ # REST API endpoints

│ ├── models/ # Database models, AI models

│ ├── services/ # Business logic

│── frontend/

│ ├── components/ # Reusable UI components

│ ├── pages/ # Student/Teacher/Admin pages

│── docs/ # Documentation

│── tests/ # Testing files

│── requirements.txt # Dependencies

│── README.md # Project description

6. Running the Application

 Open[Google Colab](https://colab.research.google.com/).

 Set runtime → **GPU (T4)**.

 Install dependencies:

!pip install transformers torch gradio -q

 Load IBM Granite model from Hugging Face.

 Run the Gradio app and get the **public shareable link**.<https://b2b3c4b17f34c0fcc1.gradio.live/>

7. API Documentation

Sample APIs:

* POST /api/login – Authenticate users
* GET /api/courses – Get available courses
* POST /api/generate-content – Generate personalized content
* POST /api/quiz – Generate AI-based quizzes
* GET /api/progress/:id – Fetch student progress report

8. Authentication

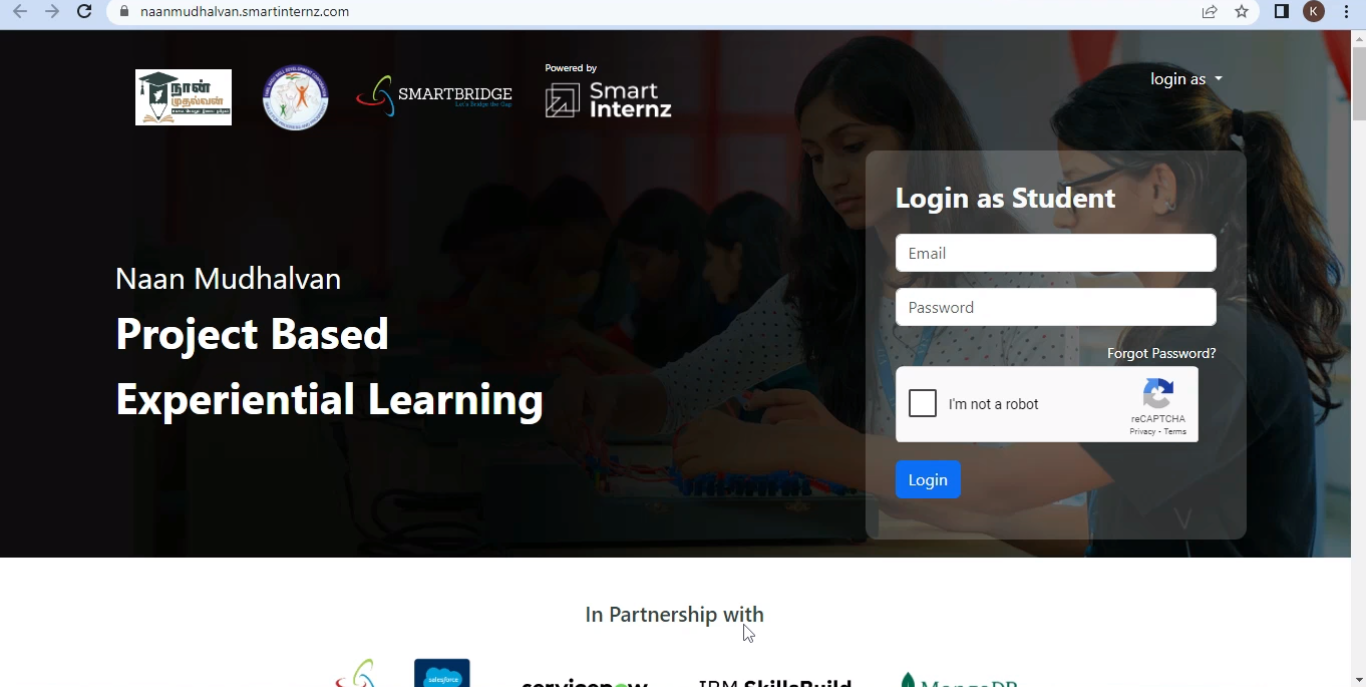
* **OAuth2**: For LMS integrations (Google Classroom/Moodle)
* **JWT Tokens**: For session-based authentication
* **Role-Based Access**:
  + **Student**: Access courses, quizzes, progress reports
  + **Teacher**: Create lessons, manage quizzes, grading
  + **Admin**: Manage users, monitor system

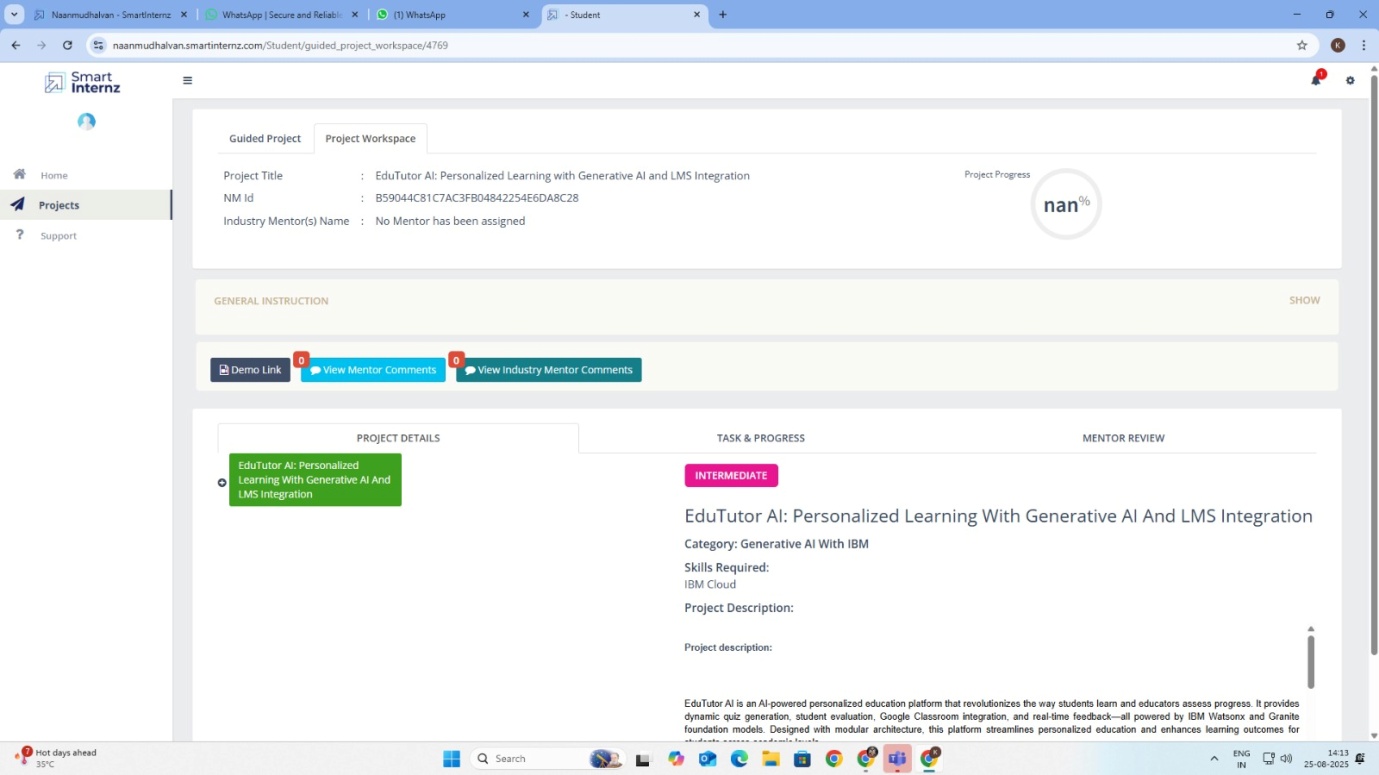
9.User Interface

**Student Dashboard**:  
Personalized study materials, quizzes, progress charts.

**Teacher Dashboard**:  
AI-generated quizzes, assignment tools, grading support.

**Admin Dashboard**:  
User management, system performance tracking.

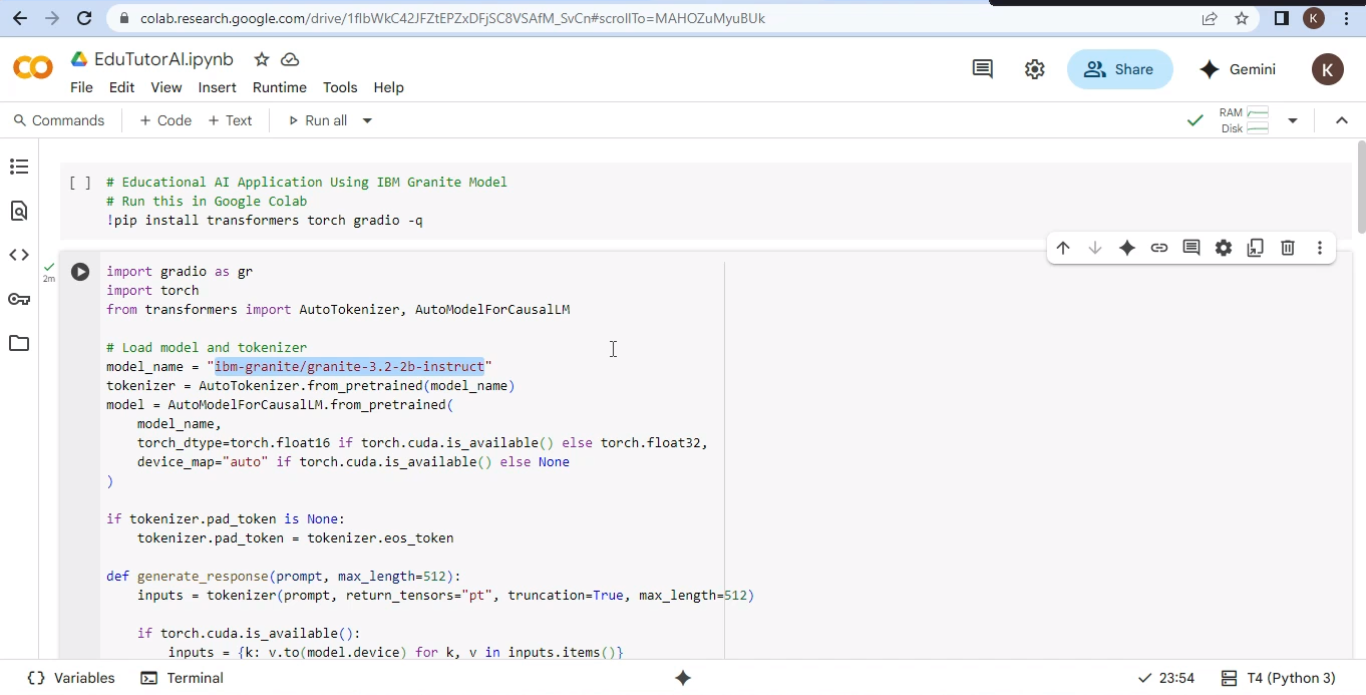


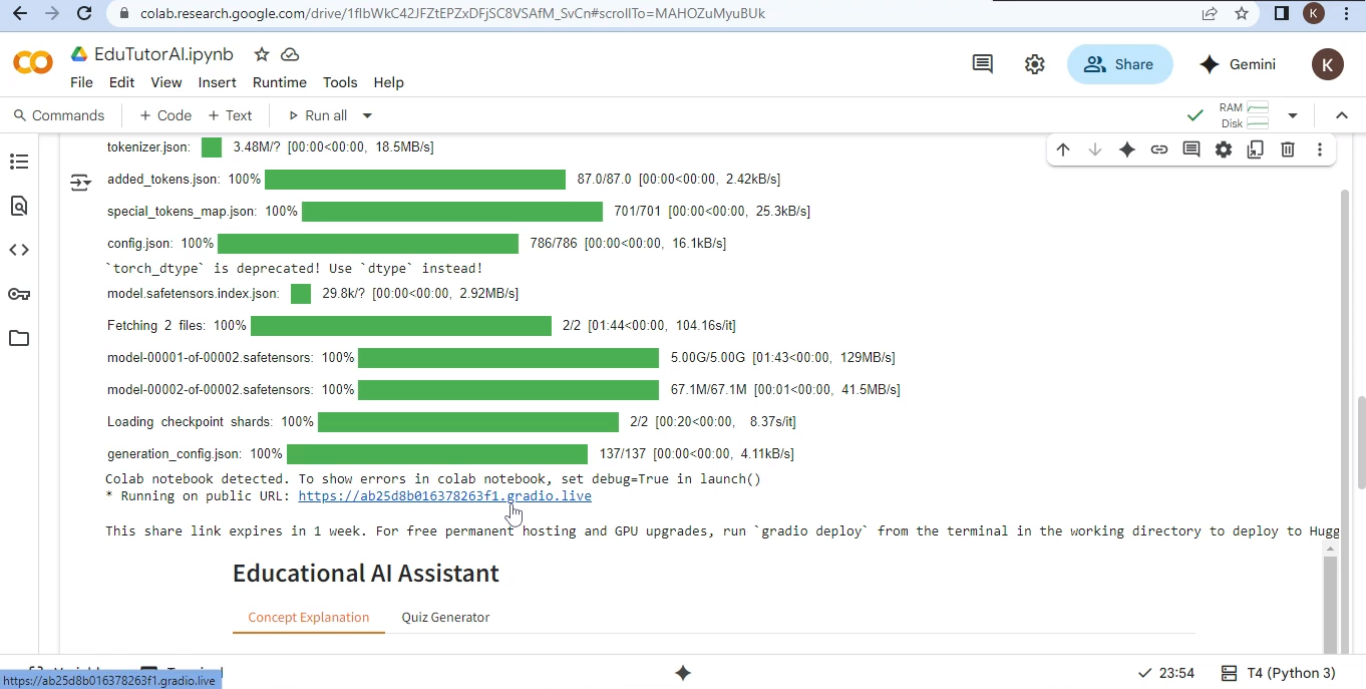


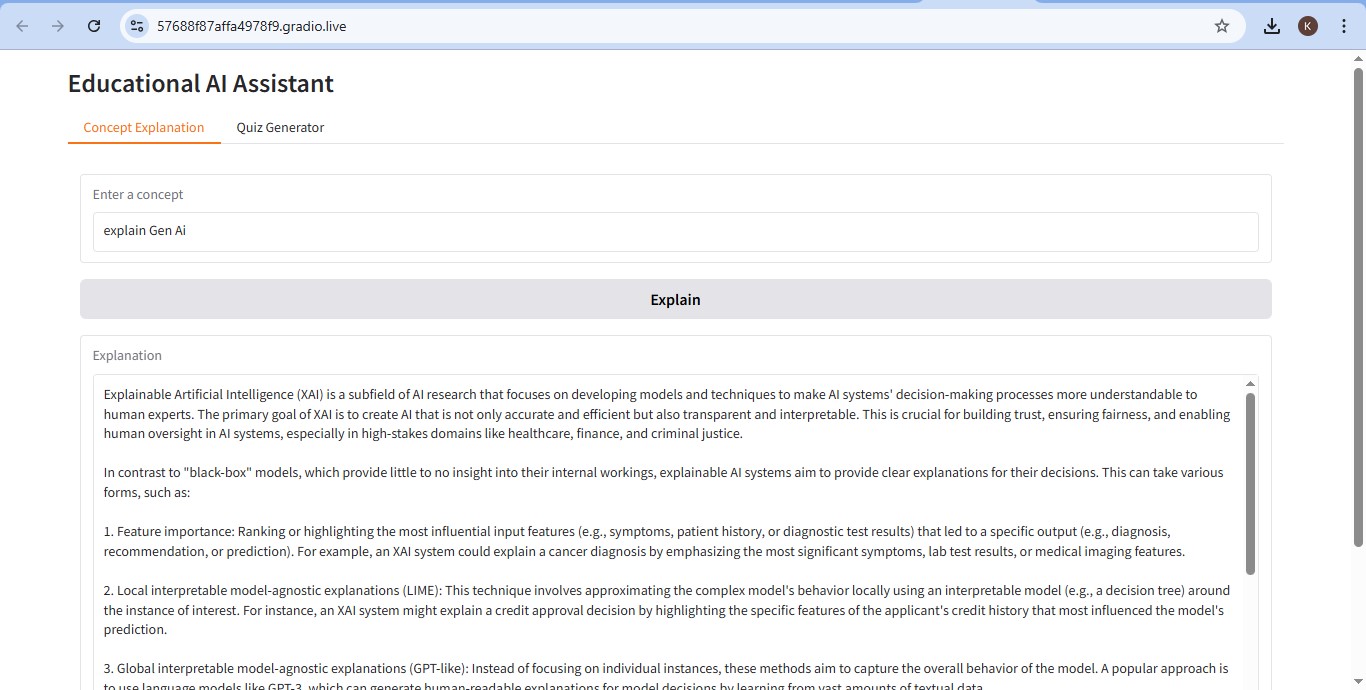
10.Testing

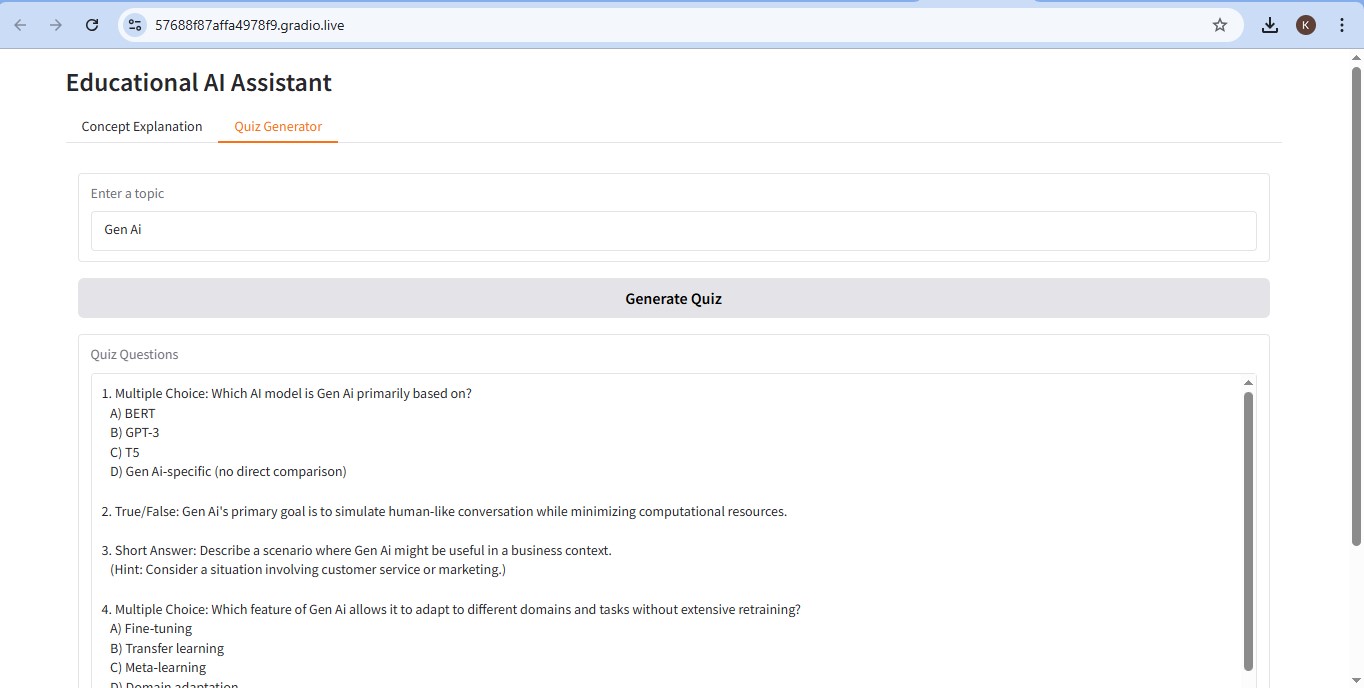
* **Unit Testing**: For API functions.
* **Integration Testing**: LMS connectivity.
* **UI Testing**: With React Testing Library / Gradio tests.
* **Load Testing**: Ensure Colab/Cloud can handle concurrent users.

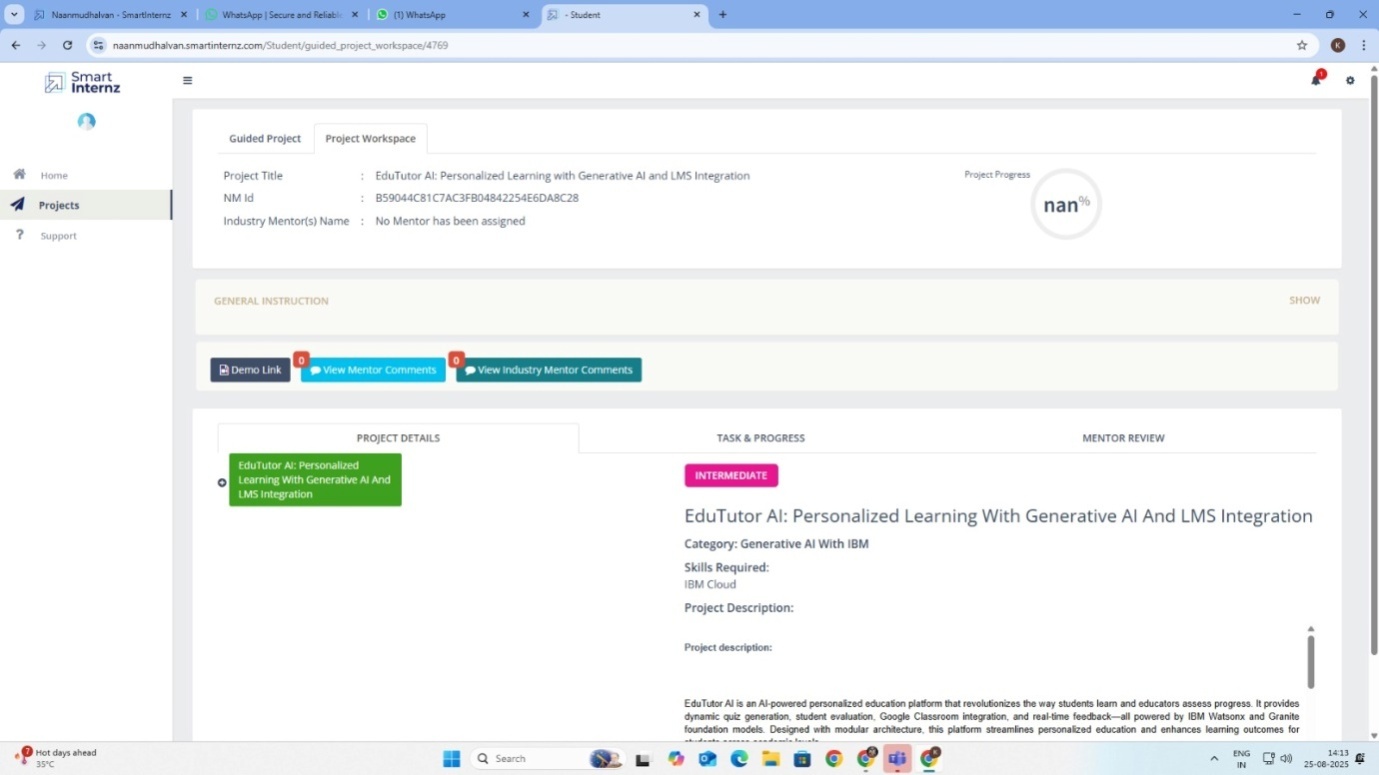
11.screen shots









12. Known Issues 

Not show for percentage issues in the project

13. Future enhancement

 Develop**mobile apps** (Android/iOS).

**Voice-based AI tutoring** for accessibility.

**Multi-language support** for regional learners.

 Integration with additional LMS platforms.

 Offline-first mode for rural learners.

14.conclusion

EduTutor AI: Personalized Learning with Generative AI and LMS Integration demonstrates how advanced AI technologies can transform education into a more **personalized, engaging, and efficient** experience. By leveraging **Generative AI models** and integrating them with **LMS platforms**, the project enables:

* **Students** to receive customized lessons, adaptive quizzes, and real-time guidance.
* **Teachers** to save time through automated content creation, grading support, and analytics-driven insights.
* **Administrators** to monitor learning outcomes and ensure better educational management.

Running seamlessly on **Google Colab with IBM Granite models**, the project also ensures accessibility with minimal setup. Though some limitations exist, EduTutor AI provides a strong foundation for future enhancements like **mobile apps, multilingual support, and voice-based tutoring**.

.