**Project title**

**Project document**

**introduction:**

• project title: SUSTAINABLE SMART CITY ASSITANT USING IBM GRANITE LLM

• Team member:MOHAMMED ABUDHAHIR J

• Team member:SURYA PRASAD P

• Team member:JAMESH ANDREWSON J

• Team member:NITHESH H

**Project overview**

* **Purpose:**The main goal of Edu Tutor AI is to provide a personalized, adaptive, and interactive learning experience for students by leveraging AI. The system will analyze learners’ strengths, weaknesses, and learning styles to deliver customized lessons, practice problems, and feedback Traditional classroom teaching often follows a one-size-fits-all approach.Many learners struggle because teaching methods don’t match their pace or style.With AI, education can become adaptive, accessible, and engaging for every student.

**Features**

* **Personalized Learning Path**

AI generates customized study plans based on learners’ goals, interests, and performance.

**2. Adaptive Assessments**

Dynamic quizzes that adjust difficulty according to student performance.

**3. Real-Time Feedback**

Instant explanations and hints to support continuous improvement.

**4. Interactive Content**

Lessons delivered via text, visuals, and possibly voice or AR/VR.

**5. Progress Tracking & Analytics**

Dashboard for students (and teachers/parents) to track progress, strengths, and areas needing attention.

**6. Multilingual Support**

Content delivery in multiple languages for inclusivity.

**7. Gamification**

Badges, leaderboards, and rewards to keep learners motivated.

**Archiecture**

**AI/ML Models**: Natural Language Processing (for Q&A, tutoring), Recommendation Systems (for personalized study paths).

**Frontend:** Web (React/Angular) or Mobile App (Flutter/React Native).

**Backend:** Python (Flask/Django, FastAPI).

Database: PostgreSQL/MongoDB for student data & learning resources.

Integration: Speech recognition APIs, AR/VR support for immersive learning.

**Expected Outcomes**

Students receive custom-tailored learning experiences.

Improved engagement and academic performance.

Teachers/parents gain insights into student progress.

Scalable platform adaptable for schools, online courses, and self-learners.

**Potential Applications**

School and college learning support.

Online education platforms.

Corporate training and skill development.

Lifelong learning and self-paced educat

Here’s a clear setup guide for your Edu Tutor AI: Personalized Learning project:

**Setup Instructions :**

**1. Environment Setup**

Install Python 3.9+

Recommended IDE: VS Code / PyCharm / Jupyter Notebook

Install dependencies:pip install streamlit fastapi uvicorn openai scikit-learn pandas numpy

**2. Choose LLM Backend**

You can connect Edu Tutor AI to:

Open-source models (Hugging Face Transformers, LLaMA, Mistral)

Cloud-based APIs (OpenAI GPT-4/5, IBM Granite LLM, Cohere)

Example (OpenAI API setup):

pip install openai

Add API key in .env:

OPENAI\_API\_KEY=your\_api\_key\_here

**3. Data & Personalization**

Create a student profile database (CSV, JSON, or DB):

Name, age, grade/level

Learning style (visual, auditory, kinesthetic)

Strengths & weaknesses

Past performance (quiz/test scores)

Example (student\_profiles.json):

{

"id": 101,

"name": "Anita",

"grade": "10",

"learning\_style": "visual",

"strengths": ["Math"],

"weaknesses": ["History"]

}

**4. Core Modules**

Student Model → Tracks progress, adapts difficulty

Curriculum Module → Maps topics to student level

Recommendation Engine → Suggests personalized learning paths

Assessment Engine → Generates quizzes/tests dynamically

Feedback System → Provides instant feedback & motivation

**API Documentation**

**1.** Overview Edututor AI provides a personalized learning experience by leveraging AI-driven tutoring and adaptive feedback. The API allows developers to integrate these features into applications, websites, or platforms to enhance learning outcomes.

**Key Features:**

* Personalized tutoring sessions
* Adaptive question generation
* Progress tracking and analytics
* Multi-format content support (text, quizzes, images)
* Conversational AI interface for interactive learning

**2 Base URL**

https://api.edututor.ai/v1

**3. Authentication**

All API requests require an API key. Include the key in the request headers.

Header Example:Authorization: Bearer YOUR\_API\_KEY

Content-Type: application/json

**4. Endpoints**

Endpoint: /sessions

Method: POST

Description: Initializes a personalized learning session for a user.

Request Body:

{

"user\_id": "12345",

"subject": "Mathematics",

"level": "Grade 10"

}

**Response:**

{

"session\_id": "abcde12345",

"user\_id": "12345",

"subject": "Mathematics",

"level": "Grade 10",

"start\_time": "2025-09-17T16:00:00Z"

}

Generate a Question

Endpoint: /sessions/{session\_id}/questions

Method: POST

Description: Generates a personalized question for the user.

Request Body:

{

"topic": "Algebra",

"difficulty": "Medium"

}

**Response:**

{

"question\_id": "q123",

"question\_text": "Solve for x: 2x + 5 = 15",

"options": ["5", "10", "15", "20"]

}

5. Submit an Answer

Endpoint: /sessions/{session\_id}/answers

Method: POST

Description: Submits a user’s answer and receives feedback.

Request Body:

{

"question\_id": "q123",

"answer": "5"

}

Response:

{

"question\_id": "q123",

"correct": true,

"explanation": "2x + 5 = 15 → 2x = 10 → x = 5"

}

**6.Get User Progress**

Endpoint: /users/{user\_id}/progress

Method: GET

Description: Retrieves user’s learning progress, including completed questions, scores, and analytics.

Response:

{

"user\_id": "12345",

"completed\_questions": 20,

"correct\_answers": 18,

"accuracy": 90

}

**7. Error Handling**

API errors use standard HTTP status codes with a JSON message.

Example:

{

"error": "Invalid API key",

"status\_code": 401

}

Common Status Codes:

200 – Success

400 – Bad Request

401 – Unauthorized

404 – Not Found

500 – Internal Server Error

**User Interface:**

The Edututor AI user interface (UI) is designed to provide a simple, interactive, and personalized learning experience for students, teachers, and administrators. The UI supports both web and mobile platforms.

**1. Dashboard**

Purpose: Provides an overview of learning progress, active sessions, and quick access to features.

**Key Components:**

User Profile: Displays user information, level, and avatar.

**Progress Overview**: Visual charts showing completed lessons, quiz scores, and accuracy.

**Quick Actions:** Buttons for starting new sessions, resuming previous lessons, or generating questions.Notifications: Alerts for upcoming tasks, reminders, or feedback from AI.

**2. Tutoring Session Interface**

**Purpose**: Interactive area where students engage with personalized lessons.

**Features**:AI Chat Interface: Conversational AI guides the student, answers queries, and explains concepts.

**Question Panel**: Displays generated questions with multiple-choice or open-ended formats.

Answer Submission: Students submit answers and receive instant feedback with explanations.

Hints & Resources: Optional hints, reference materials, and example solutions.

Progress Tracking: Real-time updates on session progress and performance metrics.

**3. Content Library Interface**

**Purpose:** Provides access to curated learning materials.

**Features:**Subjects & Topics: Browse lessons by subject, topic, or difficulty level.

Search & Filter: Quickly find specific lessons or question sets.

Interactive Lessons: Multimedia content, including text, images, and videos.

Download/Bookmark: Option to save content for offline study or later reference.

**4. Analytics & Reporting Interface (For Teachers/Admins)**

**Purpose**: Track and analyze student performance.

**Key Components:**Student Progress: Individual or group-level performance metrics.

Question Performance: Statistics on difficult topics or commonly missed questions.

Reports: Generate detailed reports in PDF or CSV format.

Custom Recommendations: AI-driven suggestions for targeted interventions.

**Testing**

Testing is a critical phase to ensure the reliability, performance, and accuracy of Edututor AI before deployment. This includes functional testing, API testing, UI/UX testing, and performance evaluation

**1. Functional Testing**

**Purpose**: Verify that each feature works as expected.Test Cases Include:tarting and ending tutoring sessions.Generating personalized questions.Submitting answers and receiving feedback.Tracking user progress and analytics.

User authentication and session management.

**2. API Testing**

**Purpose**: Ensure all API endpoints function correctly and return expected results.

Endpoints to Test:

/sessions – Start sessions

/sessions/{session\_id}/questions – Generate questions

/sessions/{session\_id}/answers – Submit answers

/users/{user\_id}/progress – Retrieve progress

Validation: Status codes, response structure, error messages, and rate limiting.

**3. UI/UX Testing**

**Purpose**: Confirm that the user interface is intuitive, responsive, and accessible.

Test Scenarios:Navigation between dashboard, lessons, and quizzes.Responsiveness across devices (desktop, tablet, mobile).Accessibility features (screen readers, high contrast, keyboard navigation).Real-time updates of progress and feedback.

**4. Performance Testing**

**Purpose**: Ensure the system handles expected loads and remains responsive.Metrics to Measure:API response time under normal and peak load.Concurrent user sessions and system stabilityScalability of AI question generation and feedback processing.





