# EDU TUTOR Project Documentation

## 1. Introduction

* **Project Title:** EDU TUTOR – Intelligent Learning & Tutoring Platform
* **Team Id:** NM2025TMID00808
* **Team Members:**
  + SWETHA.A
  + SANDHIYA.G
  + PRIYA.R

## 2. Project Overview

**Purpose**  
EduTutor uses natural language processing (NLP) and adaptive learning analytics to create an engaging study environment.  
Benefits include:

* Personalized learning for students
* Reduced workload for teachers
* Actionable insights for educators and institutions
* Secure and scalable protects student data and scales to support individual classrooms or entire institutions
* 24/7 learning support offers round-the-clock help so students can study and get answers anytime
* Adaptive feedback analyzes quizzes and chat interactions to recommend follow-up topics and resources
* Teacher and institution insights provide class-wide performance metrics and common learning gaps for targeted lesson planning

**Features**

• **Interactive AI Tutor**  
o Key Point – Real-time student interaction  
o Functionality – Provides instant, human-like answers to questions across subjects such as math, science, and languages

• **Student Performance Analysis**  
o Key Point – Personalized learning insights  
o Functionality – Evaluates quizzes and chat interactions to highlight strengths and areas that need improvement

• **Dynamic Teacher Dashboard**  
o Key Point – Actionable classroom analytics  
o Functionality – Visualizes student progress, quiz results, and common learning gaps for teachers and administrators

• **Assignment and Feedback Module**  
o Key Point – Continuous assessment  
o Functionality – Allows students to submit assignments or feedback and receive AI-generated grading suggestions or comments

• **Adaptive Learning Recommendations**  
o Key Point – Customized study plans  
o Functionality – Suggests topics, practice questions, and resources based on each learner’s pace and performance

**Use Case Scenarios**

• **Real-Time Tutoring** – Students ask questions and the AI explains concepts, provides step-by-step solutions, and offers examples.

• **Performance Analysis** – Quizzes and learning interactions are analyzed to identify strengths, weaknesses, and areas needing review.

• **Teacher Dashboard** – Educators monitor overall class performance, track individual student progress, and spot topics that require additional instruction.

## 3. Architecture

 **Frontend** with pages like *subjects* and *chat*

 **Backend** for quiz processing and analytics

 **LLM Integration** for AI tutoring

 **Data Handling** for student chats and quiz history

 **Visualization** for student progress and topic mastery

## 4. Setup Instructions

**Prerequisites**

Python 3.7+  
Flask  
PyTorch (with CUDA for GPU acceleration)  
Hugging Face libraries – transformers, accelerate, bitsandbytes  
Internet connection for the initial model download

**Installation Process**

1. Clone the repository and set up the structure

(app.py, templates/, static/).

1. Create & activate a virtual environment:

python -m venv env

source env/bin/activate # Linux/Mac

env\Scripts\activate # Windows

1. Install dependencies:

pip install -r requirements.txt

1. Configure IBM Granite model path:

ibm-granite/granite-3.3-8b-instruct

1. Run the Flask backend:

python app.py

## Folder Structure

**CCS**

app.py – Main Flask application

templates/ – HTML templates (index, about, subjects, chat, dashboard, login)

static/ – CSS, images, favicon

requirements.txt – Python dependencies

## 6. Running the Application

1. Start the Flask server: python app.py
2. Open a browser to http://localhost:5000
3. Navigate via the menu:
   * **Chat/Tutor:** Interact with the AI tutor
   * **Assignments:** Submit or review assignments
   * **Dashboard:** View student performance analytics
   * **Login:** Teacher or student authentication

## API Endpoints

| **Method** | **Endpoint** | **Description** |
| --- | --- | --- |

|  |  |  |
| --- | --- | --- |
| POST | /ask | Student asks a question for AI tutoring |

|  |  |  |
| --- | --- | --- |
| POST | /quiz | Submit quiz answers for grading and feedback |

|  |  |  |
| --- | --- | --- |
| POST | /feedback | Submit feedback or assignment comments |

|  |  |  |
| --- | --- | --- |
| GET | /dashboard | View aggregated performance metrics |

|  |  |  |
| --- | --- | --- |
| POST | /login | User authentication (student/teacher) |

|  |  |  |
| --- | --- | --- |
| POST | /logout | End session |

## 8. Authentication

* User login with session management
* **Planned:** Role-based access (student, teacher, admin)

## 9. User Interface

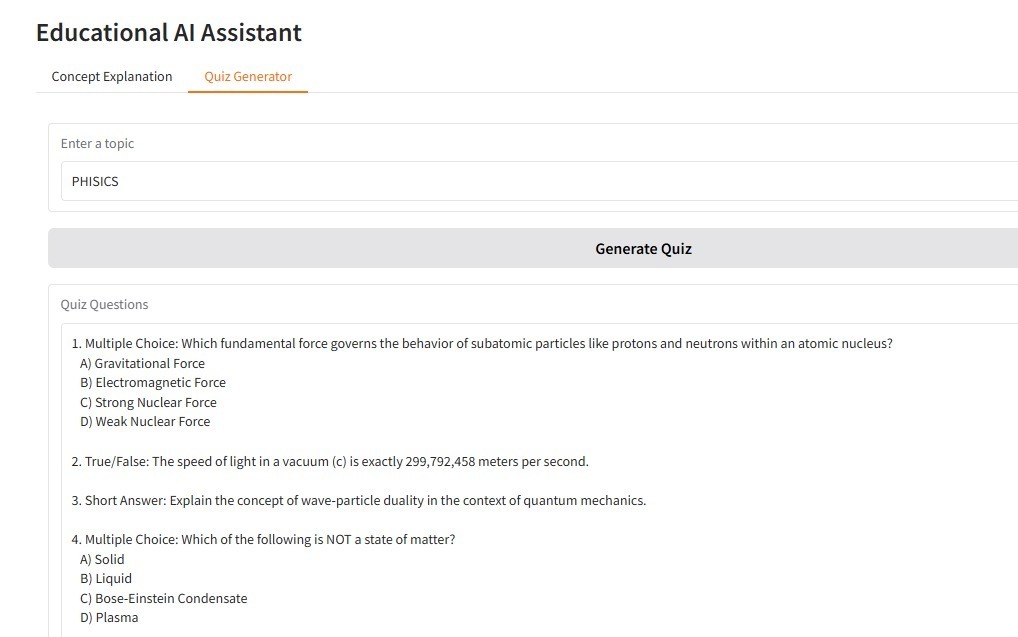
* **Index Page:** Welcome screen with “Start Learning” button
* **Login Page:** Secure student/teacher authentication
* **About Page:** Explains mission, features, and benefits
* **Tutor Page:** AI-powered interactive learning assistant
* **Dashboard:** Analytics of student performance and quiz results

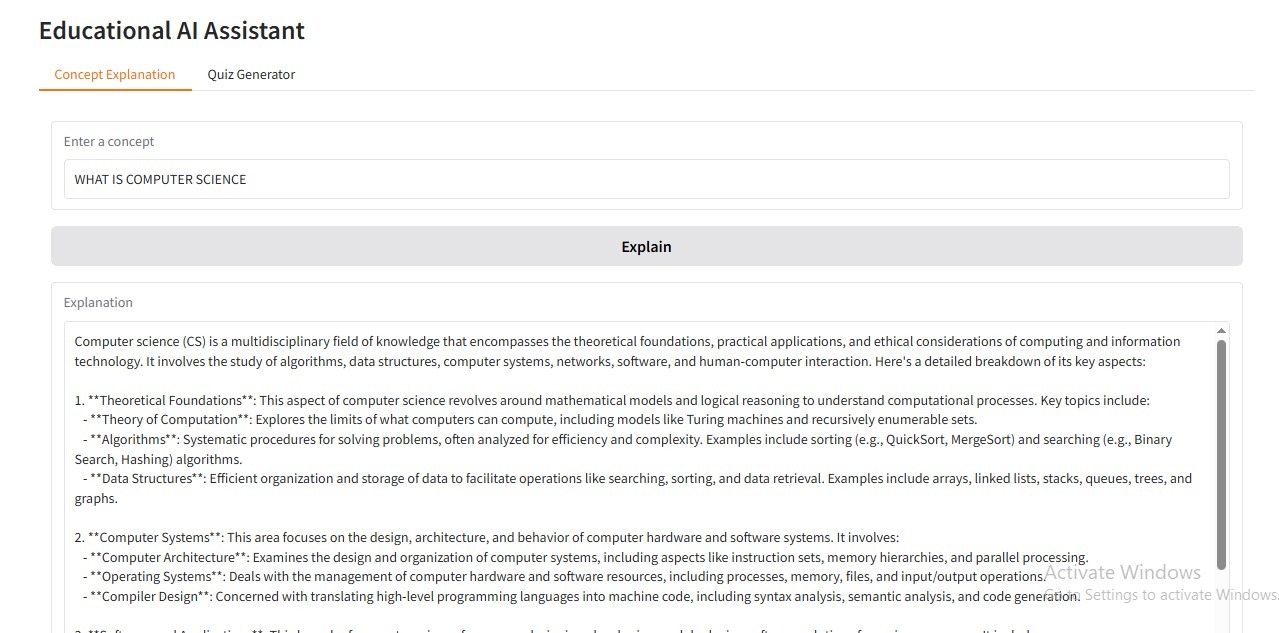
## 10. Testing

* **Unit Tests:** Flask routes, quiz scoring, AI tutor functions
* **Integration Tests:** End-to-end tutoring, quiz submission, and analytics flow
* **Manual Testing:** Question answering, assignment submission, login/logout
* **Edge Cases:** Empty input, invalid login, malformed quiz data

## 11. Screenshots

*(Insert UI mockups/screenshots: Index, Tutor Chat, Dashboard, Login, About)*





## 12. Known Issues

 In-memory data storage limits persistence

 CPU-only execution is slow

 Scalability limited until database integration

## 13. Future Enhancements

 Database integration for long-term progress tracking

 Multi-language tutoring support

 Mobile-friendly responsive design

 Adaptive curriculum recommendations based on learning style

 Gamified learning features (badges, levels)