

EduTutor AI

Final Project Report

1. INTRODUCTION

1.1 Project Overview

EduTutor AI is an AI-powered personalized education platform designed to revolutionize how students learn and how educators assess academic progress. By integrating AI models (IBM Watsonx, Granite LLM) with Google Classroom and Pinecone, it dynamically generates quizzes, tracks performance analytics, provides adaptive testing, and ensures data-driven learning insights.

1.2 Purpose

The purpose of this project is to deliver an AI-based personalized education assistant that:

- Generates tailored quizzes based on Google Classroom topics.
- Tracks student performance with real-time analytics.
- Adapts quiz difficulty using diagnostic results.
- Integrates seamlessly with existing learning management systems.

2. IDEATION PHASE

2.1 Problem Statement

Students often struggle to find personalized academic support aligned with their curriculum. Educators lack real-time analytics to track individual student progress efficiently. EduTutor AI bridges this gap by combining AI-driven quiz generation, adaptive learning, and performance insights.

2.2 Empathy Map Canvas

Who are we empathizing with?

Students and educators managing online classrooms.

What do they need to do?

Students: Improve learning with tailored quizzes and feedback.

Educators: Track performance and adjust teaching accordingly.

What do they need to do?

Students: Improve learning with tailored quizzes and feedback.

Educators: Track performance and adjust teaching accordingly.

What do they see?

Generic, non-personalized assessments.

What do they hear?

Limited access to tools offering adaptive learning insights.

What do they say and do?

Students: Seek better, personalized quizzes.

Educators: Request analytics for effective teaching.

What do they think and feel?

Students: Frustration over static assessments.

Educators: Need actionable, real-time data.

2.3 Brainstorming

- AI-powered quiz generation.
- Google Classroom integration.
- Diagnostic test for new users.
- Adaptive difficulty quizzes.
- Educator analytics dashboard.
- Real-time feedback with topic and difficulty mapping.

3. REQUIREMENT ANALYSIS

3.1 Customer Journey Map

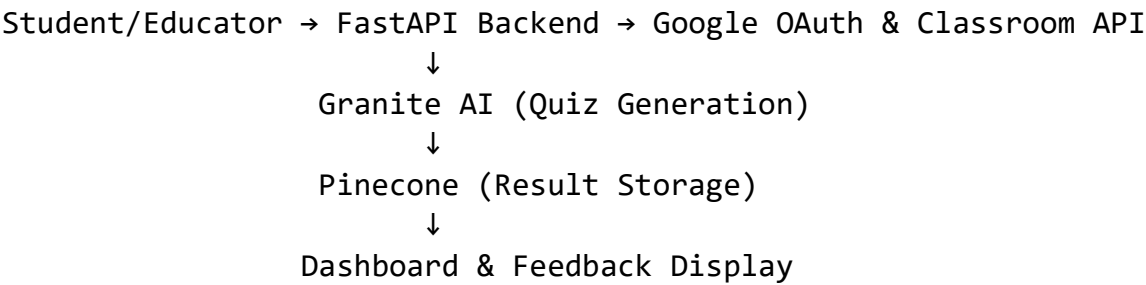
CUSTOMER JOURNEY MAP

Stage	Student Actions	Educator Actions	System Reactions
Login	Google OAuth	Google OAuth	Authenticate users
Course Sync	Connect to Google Classroom	View student list	Fetch course data
Diagnostic Test	Attempt quiz	–	Store and analyze results
Quiz Attempt	Attempt quizzes	View quiz insights	Adaptive quiz generation
Results	View feedback	Analyze dashboard	Store results and update Pinecone

3.2 Solution Requirement

- Google OAuth authentication
- Google Classroom API integration
- IBM Granite model API for quiz generation
- Adaptive difficulty logic via diagnostic score tracking
- Educator dashboard with Pinecone-based insights
- Real-time feedback

3.3 Data Flow Diagram



3.4 Technology Stack

- **Backend:** FastAPI (Python)
- **AI Model:** IBM Granite via Hugging Face
- **Database:** Pinecone Vector Database
- **Authentication:** Google OAuth 2.0
- **Frontend:** Jinja2 Templates, HTML/CSS
- **Cloud APIs:** Google Classroom API

4. PROJECT DESIGN

4.1 Problem Solution Fit

The platform ensures personalized assessments based on students’ classroom topics and skill level, offering educators real-time insights, thus addressing the problem of static assessments.

4.2 Proposed Solution

EduTutor AI dynamically generates diagnostic and adaptive quizzes using AI models, integrates with Google Classroom to sync student data, and provides real-time performance analytics to educators via Pinecone vector search.

4.3 Solution Architecture

Refer to the earlier detailed Solution Architecture section. Key components:

- FastAPI modular backend
- Granite AI API client for quiz generation
- Pinecone vector DB for adaptive insights
- Google Classroom API integration
- Student and Educator dashboards

5. PROJECT PLANNING & SCHEDULING

5.1 Project Planning

Phase	Timeline (Dates)	Deliverables
Requirement Analysis	18 June 2025	Finalized Requirement Document
API Integrations	18 – 19 June 2025	Google OAuth Integration, Google Classroom API Configuration
Diagnostic Quiz Module	20 – 21 June 2025	Diagnostic Quiz Generation Logic

Adaptive Quiz Logic	21 – 22 June 2025	Adaptive Quiz Difficulty Adjustment Logic
Educator Dashboard	23 – 24 June 2025	Real-time Analytics Dashboard Module
Testing & Deployment	25 – 27 June 2025	Performance and Functional Testing Reports, Final Deployment

6. FUNCTIONAL AND PERFORMANCE TESTING

6.1 Performance Testing

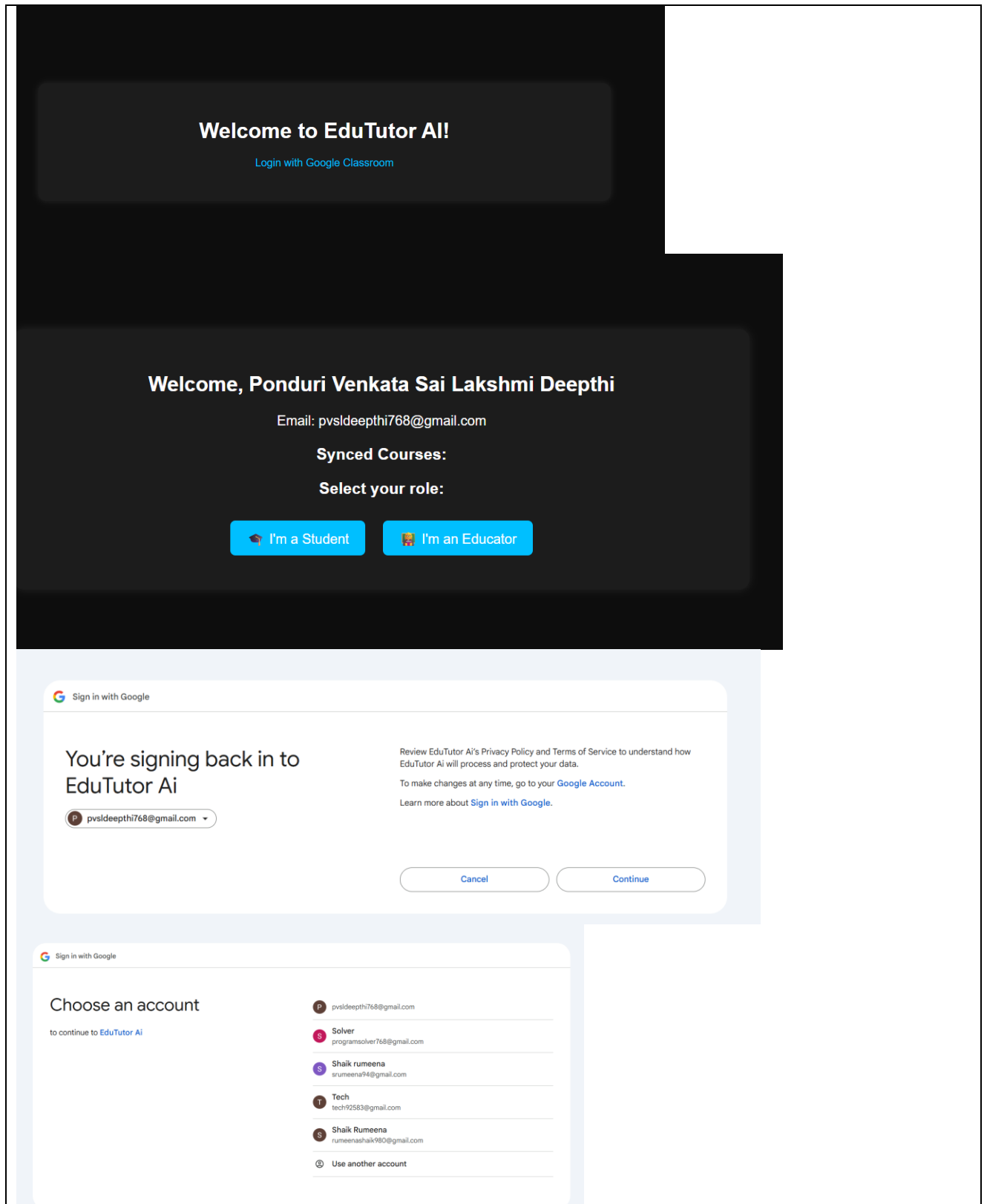
- FastAPI response time: < 300ms for API calls
- Pinecone search latency: < 150ms
- AI quiz generation average: < 2s per quiz

Functional Testing Results:

- Google OAuth — Passed
- Google Classroom Sync — Passed
- Diagnostic Quiz Generation — Passed
- Adaptive Quiz Adjustment — Passed
- Educator Dashboard Analytics — Passed

7. RESULTS

7.1 Output Screenshots



Welcome, Ponduri Venkata Sai Lakshmi Deepthi

Your Courses:

mathematics 2nd btech

Select subject for Diagnostic Test:

mathematics 2nd btech

Start Diagnostic Test

Enter topic for Quiz:

probability

Difficulty:

Medium

Generate Quiz

[Logout](#)

Welcome, Ponduri Venkata Sai Lakshmi Deepthi

Your Courses:

mathematics 2nd btech

Select subject for Diagnostic Test:

mathematics 2nd btech

Start Diagnostic Test

Enter topic for Quiz:

Enter topic

Difficulty:

Medium

Generate Quiz

[Logout](#)

Welcome, Ponduri Venkata Sai Lakshmi Deepthi

Your Courses:

mathematics 2nd btech

Select subject for Diagnostic Test:

mathematics 2nd btech

Start Diagnostic Test

Enter topic for Quiz:

Enter topic

Difficulty:

Medium

Generate Quiz

[Logout](#)

Quiz Results - probability


Score: 0/6

[← Back to Dashboard](#)

Diagnostic Test

Submit Diagnostic Test

You have been successfully logged out.

 [Go back to login again.](#)

8. ADVANTAGES & DISADVANTAGES

Advantages

- **AI-Powered Personalized Education**
 - EduTutor AI dynamically generates quizzes tailored to individual students' courses and learning levels using IBM Granite LLM, enhancing engagement and learning outcomes.
- **Real-Time Insights for Educators**
 - Educators can access live dashboards powered by Pinecone, offering instant visibility into student performance, attempted topics, and progress trends — enabling data-driven instruction.
- **Seamless Integration with Google Classroom**
 - Direct synchronization of student course data from Google Classroom ensures alignment with institutional syllabi and allows automated topic-based quiz generation.
- **Adaptive Quiz Difficulty Based on Diagnostic Performance**
 - The system conducts an initial diagnostic test to gauge student ability and intelligently adjusts quiz difficulty in subsequent attempts, offering an optimized learning experience.
- **Cloud-Based, Scalable Architecture**
 - Built on FastAPI and Pinecone with external AI and OAuth services, the platform is scalable and can support a growing number of users and real-time data without compromising performance.

Disadvantages

- **Requires Internet Connectivity**
 - As EduTutor AI depends on cloud APIs (Google OAuth, Google Classroom, Granite AI, Pinecone), uninterrupted internet access is essential for full functionality.
- **Dependency on External APIs**
 - The system's key services rely on third-party APIs (Granite LLM via Hugging Face, Google Classroom, Google OAuth), making it dependent on their availability, quota limits, and service stability.
- **Limited Offline Access**
 - Since all quiz generation, result processing, and data fetching occur via online services, the application offers no offline functionality.

- **Potential API Costs**
 - Usage of IBM Granite LLM, Pinecone, and Google Classroom API could incur costs as data and usage volume scales, especially for institutional or enterprise-grade deployments.

9. CONCLUSION

EduTutor AI successfully delivers a scalable, AI-powered personalized learning solution that integrates with existing LMS platforms like Google Classroom, offering real-time, adaptive learning journeys for students and insightful analytics for educators.

10. FUTURE SCOPE

- Integration with additional LMS systems like Moodle, Canvas
- AI-powered video lectures and content recommendations
- Mobile app version for students and educators
- Expand adaptive difficulty algorithms using student learning history

11. APPENDIX

Source Code:

▼ EDU AI

▼ app

> __pycache__

▼ api

> __pycache__

▼ routes

> __pycache__

🔌 auth.py

🔌 educator.py

🔌 quiz.py

🔌 student.py

🔌 __init__.py

▼ core

> __pycache__

🔌 __init__.py

🔌 granite_client.py

🔌 oauth.py

🔌 pinecone_client.py

▼ services

> __pycache__

🔌 diagnostic_engine...

🔌 quiz_generator.py

▼ static

★ favicon.ico

style.css

▼ templates

<> base.html

<> diagnostic_test.htm

<> educator_dashboa..

<> educator_results.ht

```
> api > routes > 🔌 student.py > 🔌 submit_quiz
from fastapi import APIRouter, Request, Form
from fastapi.responses import HTMLResponse, RedirectResponse
from fastapi.templating import Jinja2Templates
from app.services.diagnostic_engine import generate_diagnostic_test
from app.services.quiz_generator import generate_quiz
from app.core.pinecone_client import save_quiz_result
import json
router = APIRouter()
templates = Jinja2Templates(directory="app/templates")

@router.get("/dashboard")
async def student_dashboard(request: Request):
    user = request.session.get("user")
    courses = request.session.get("courses", [])
    if not user:
        return RedirectResponse(url="/auth/login")
    return templates.TemplateResponse("student_dashboard.html", {"request": request, "user": user, "courses": courses})

@router.post("/diagnostic_test")
async def start_diagnostic_test(request: Request, subject: str = Form(...)):
    user = request.session.get("user")
    if not user:
        return RedirectResponse(url="/auth/login")
    diagnostic_test = generate_diagnostic_test(subject)
    return templates.TemplateResponse("diagnostic_test.html", {"request": request, "test": diagnostic_test, "subject": subject})

@router.post("/generate_quiz")
async def generate_quiz_route(request: Request, topic: str = Form(...), difficulty: str = Form("medium")):
    user = request.session.get("user")
    if not user:
        return RedirectResponse(url="/auth/login")

    quiz = generate_quiz(topic, difficulty)
    request.session["last_quiz"] = quiz
    request.session["last_quiz_topic"] = topic
    request.session["last_quiz_difficulty"] = difficulty
    return templates.TemplateResponse("quiz_questions.html", {
```

Dataset Link

- Google Classroom API — synced data on demand

GitHub & Project Demo Link

- GitHub: https://github.com/pvsldeephthi-768/EduTutor_AI.git