Disha-Mitra: AI-Powered Study Assistant

1. Project

Github link: https://github.com/Rishirk2107/AI-Powered-Study-Asst

Overview Name:

Disha-Mitra
Purpose: An AI-powered MERN web app helping students study effectively via personalized tools based on their own study materials.

Key Features:
- JWT-based Login & Signup
- Flashcard generation from uploaded PDFs
- QA Chatbot for context-aware questioning
- AI-powered quiz generator from study material
- Smart study schedule planner with calendar UI
- Light/dark mode & profile management

2. Tech Stack

- Frontend: ReactJS, React Router, Tailwind CSS, Context API
- Backend: Node.js + Express (MVC)
- Database: MongoDB with Mongoose
- AI Services: FastAPI (Python), LangChain, Groq or similar
- Authentication: JWT
- File Upload: Multer
- Deployment: AWS S3 (frontend), EC2 (backend + AI), Nginx, Certbot/ACM (optional SSL)

3. Architectue

Flowchart:
1. User uploads PDF via frontend
2. React routes request to protected dashboard
3. File sent to backend via Multer
4. PDF processed -> sent to FastAPI AI server
5. Response stored in MongoDB
6. Displayed via React UI

Component Diagram:

Frontend: Auth Pages Dashboard Flashcards, Chatbot, Quiz, Scheduler
Backend (Express): Routes: /auth, /materials, /chat, /quiz, /schedule; Controllers & Middleware
AI Server (FastAPI): /upload, /ask, /generate-quiz, /generate-schedule
Database (MongoDB): Users, Materials, Flashcards, Messages, Quizzes, Schedules

4. Folder Structure

frontend/

Disha-Mitra: AI-Powered Study Assistant

```
src/
 components/
 pages/
 context/
 routes/
 App.js
```

backend/
backend/
controllers/
models/
routes/
middleware/
server.js

ai-server/
ai-server/
main.py
endpoints/
services/

5. Setup Instructions

git clone https://github.com/user/disha-mitra
cd disha-mitra

Frontend:
cd frontend
npm install
npm start

Backend:
cd backend
npm install
touch .env # Add DB_URI, JWT_SECRET
npm run dev

AI Server:
cd ai-server
pip install -r requirements.txt
uvicorn main:app --reload

MongoDB: Use Atlas or Local setup

Deployment:
Upload React build to S3
Deploy backend and AI to EC2
Use Nginx for reverse proxy + SSL with Certbot

6. Authentication Flow

- User signs up/logs in JWT generated
- Token stored in localStorage
- Token sent with protected API calls
- Express middleware verifies JWT
- React Router guards private routes

7. Flashcard & Material Upload Flow

Disha-Mitra: AI-Powered Study Assistant

1. PDF uploaded via dashboard
2. Renamed using UUID
3. Sent to backend stored in /uploads
4. Backend sends file to AI server
5. AI returns flashcards stored in MongoDB
6. Frontend fetches and displays

8. QA Chatbot Flow

1. User uploads file (context)
2. Asks questions in chatbox
3. React sends question to backend AI server
4. AI responds with streamed reply
5. Messages rendered in chat bubbles

9. AI Quiz Flow

1. User uploads study material
2. API processes and returns MCQs:

```
{
  "question": "What is AI?",
  "options": ["Art Intelligence", "Artificial Intelligence"],
  "answer": "Artificial Intelligence"
}
```

3. React renders quiz UI

10. Study Scheduler Flow

1. User inputs topics & deadline
2. AI suggests study plan (topics + dates)
3. Saved in MongoDB per user
4. Displayed via calendar component

11. API Reference

Auth
POST /api/auth/signup
POST /api/auth/login

Materials
GET /api/materials
POST /api/materials/upload

Flashcards
GET /api/flashcards?materialId=xyz

Chat
POST /api/chat/upload

POST /api/chat/ask

Quiz
POST /api/quiz

Schedule
POST /api/schedule
GET /api/schedule/user/:userId