AI-Powered Chat Application – Submission for Software Engineer Intern Role

This project is a secure and scalable **AI-powered chat application** built with **FastAPI (backend)** and **React (frontend)**, designed to provide users with project-based conversational assistance while supporting authentication, file uploads, and seamless integration with the **OpenRouter API** for intelligent responses.

1. Prerequisites

This project is a **chat application** that leverages modern web technologies and AI integration. The core stack includes:

- Backend**:** FastAPI (Python) – for building a scalable and high-performance REST API.
- Frontend: React (JavaScript) – optional, if you are running the project with a UI.
- AI Integration: **OpenRouter API** with **Google Gemini 1.5 Pro** model – for intelligent and contextual AI-powered conversations.

2. Backend Setup

The backend is built using **FastAPI**. Follow these steps to set it up:

- **Clone the repository and navigate to the backend folder**:

  git clone <your-repo-link>

  cd backend
- **Create and activate a virtual environment (Optional but recommended)**:

  Mac/Linux

  python -m venv venv

  source venv/bin/activate

  Windows

  venv\Scripts\activate
- **Install the required dependencies**:

  pip install fastapi uvicorn python-dotenv requests python-jose[cryptography] PyPDF2

- **Create a** `.env` **file** in the backend folder and include the following:

  SECRET_KEY=supersecretkey

  OPENROUTER_API_KEY=your_openrouter_api_key_here

- **Start the backend server**:

  uvicorn main:app –reload

  The backend will now run at **http://127.0.0.1:8000.**

3. Frontend Setup

The frontend is built with **React**. To run it:

- **Navigate to the frontend folder**:

  cd frontend

- **Install required dependencies**:

  npm install

- **Start the development server**:

  npm run dev

  The frontend will now run at **http://localhost:5173**

4. Testing the API

You can test the backend using `curl` commands or tools like **Postman**.

- **Ping Route** (check if backend is active):

  curl http://127.0.0.1:8000/ping

- **Register User** (create a new user):

  curl -X POST http://127.0.0.1:8000/register \

  -H "Content-Type: application/json" \

  -d '{"email":"test@example.com","password":"1234"}'

- **Login** (get JWT token):

  curl -X POST http://127.0.0.1:8000/token \

  -H "Content-Type: application/x-www-form-urlencoded" \

  -d "username=test@example.com&password=1234"

- **Chat Example** (send a message to the AI):

  curl -X POST http://127.0.0.1:8000/chat \

  -H "Authorization: Bearer <your_jwt_token_here>" \

  -H "Content-Type: application/json" \

  -d '{"project_id": 1, "message": "Hello AI"}'

5. Notes

- Ensure your `.env` file is configured with the correct **API key** from **OpenRouter**.
- By default, AI responses are capped at **1000 tokens** to stay within free-tier credit limits.
- For production use, replace the current **in-memory storage** with a proper **database** (e.g., PostgreSQL, MySQL, MongoDB).
- Once configured, the system is fully set up and **ready for testing and usage**.