PROJECT NAME: CODE GENE AI

TITLE : DEEPSEEK UI

SUBMITTED BY: J.MANISHA

MILESTONE 2

1. Project Overview

The aim of this project is to develop an intelligent chatbot using Streamlit and the DeepSeek API. This chatbot provides users with an interactive platform to ask questions and receive instant, context-aware responses powered by a cloud-based AI language model. The project demonstrates how advanced natural language processing (NLP) can be integrated into a simple and intuitive web interface, ensuring seamless conversational interactions. Additionally, the system preserves message history to maintain the continuity of dialogue, enhancing the overall user experience.

2.Project Objectives

- To develop an interactive chatbot with a clean and user-friendly interface.
- To integrate DeepSeek API for generating real-time, intelligent conversational responses.
- To store and display chat history, ensuring smooth and continuous dialogue.
- To provide users with a responsive and seamless conversational experience through Streamlit.
- To handle errors and exceptions effectively, ensuring system reliability and stability.

3.Introduction:

Groq is a technology company that provides powerful tools and APIs to run AI models efficiently in the cloud. Instead of requiring a high-end computer or GPU, it allows developers to access advanced AI models remotely. In my code, Groq is used to power the DeepSeek chatbot, handling all the complex computations so the chatbot can respond quickly and accurately. This makes my project lightweight, fast, and scalable, letting the focus stay on building a great user experience rather than worrying about heavy hardware.

4. Approach

- I studied about deepseek and streamlit and its work process.I also came to know its applications and types of modesl are present.

To build this project, I followed a step-by-step learning and development process:

1. **Understanding the Requirements**

   I. I Installed a Python of version 3.12
   II. I Installed VS code to run my codes
   III. I Created a Virtual Enivornmnet
   IV. To create Virtual Environment ,I Followed these steps

   - Created a new folder for my project
   - To Create virtual environment
   - **python -m venv venv** the command used
   - It is Activated.
   - To activate On Windows:
   - **venv\Scripts\activate** this command is used.

V. The Command **pip install streamlit ollama** is used to download the required dependices.

VI. I Installed Ollama from the google

- The Command **ollama pull deepseek-r1:1.5b,ollama pull deepseek-coder:1.3b** is used to download the model.
- The Command **ollama run deepseek-r1:1.5b,ollama run deepseek-coder:1.3b,** is used to run the model in the Command Prompt.

VII. The Python code is saved in created file .

VIII. Run this Command in the Terminal **streamlit run (filename.py)** to create a Chatbot.This will open your chatbot in the browser at http://localhost:8501.

2.Creating UI using API Key

Steps to Get Your Groq API Key for DeepSeek

1. **Sign Up / Log In**
   - Go to https://console.groq.com
   - Create an account or login if you already have one.
2. **Go to Documentation / Quickstart**
   - On the console, navigate to **Overview → Quickstart** or **API Keys** section.
3. **Create a New API Key**
   - Click **"Create API Key"**.
   - Give it a **name** (e.g., DeepSeekKey or any name you like).
   - Click **Create**.
4. **Copy Your API Key**
   - A string will be generated (usually long, alphanumeric).
   - Copy it and **save it securely** (like in Notepad or a password manager).
   - Write the Python code and save it.
5. The Command **pip install streamlit groq** is used to download the groq in the terminal in vs code.
6. Run this Command in the terminal **streamlit run(filname.py)** to create a chatbot.

4. Methodology

4.1 Tools & Technologies

- **Python 3.12** – Programming language for developing the chatbot.
- **Libraries:**
  - **Streamlit** – For building the web-based chat interface.
  - **Groq** – For accessing the DeepSeek language model via API.
- **Virtual Environment** – For managing project dependencies in isolation.
- **IDE:** VS Code – For code development and debugging.

4.2 System Design

- **Frontend (UI):** Streamlit interface with chat input, message display, and sidebar showing last messages.
- **Backend (Processing):** Groq API handles AI response generation using the DeepSeek model.

- **Data Handling:** st.session_state is used to store and maintain conversation history across multiple interactions.

4.3 Workflow

1. User enters a message in the chat input.
2. The query is sent to the **DeepSeek model** via Groq API.
3. The model generates a response.
4. Both the user message and AI response are saved in session state.
5. Messages are displayed in a chat-like format in the main panel, with the last exchange shown in the sidebar.

4.4 Testing & Debugging

During development, the chatbot was tested multiple times to ensure smooth functionality. Key issues identified and resolved:

- **API Key Errors** → The chatbot could not generate responses if the API key was missing. This was fixed by implementing a sidebar input and also supporting Streamlit secrets.
- **Missing Dependencies** → The app failed to run when required libraries (like streamlit or groq) were not installed. Resolved by installing the missing packages using pip install streamlit groq.
- **Chat History Display** → Initially, previous messages were not displayed correctly. Updated session state handling to show the last user and bot messages properly in the sidebar.
- **Hardware Limitation** → My laptop does not have a large GPU, which makes running heavy AI models locally impractical. Using Groq's cloud API allowed the chatbot to generate responses efficiently without relying on local GPU resources.

5. Outcomes:

- Successfully developed a real-time chatbot using DeepSeek and Groq API.
- Built a user-friendly Streamlit interface with chat input, message display, and last message tracking in the sidebar.
- Implemented manual API key entry in the sidebar for authentication with Groq.
- Enabled conversation history management using st.session_state.
- The chatbot generates fast and accurate AI responses without requiring a high-end GPU locally.
- Learned to handle common issues like missing dependencies, chat history display, and API-related errors (e.g., 402 Payment Required).

6. Conclusion:

The DeepSeek chatbot project demonstrates the integration of a cloud-based AI model with a user-friendly Streamlit interface. By leveraging the Groq API, the chatbot is able to generate fast and accurate responses without requiring a high-end local GPU. The application successfully maintains conversation history, handles user input effectively, and provides a seamless chatting experience. Overall, this project

highlights the practical use of cloud AI services to build responsive and scalable chatbot applications while addressing common challenges like API authentication, session management, and error handling.