**DevelopersHub AI/ML Engineering Interns**

**DevHub Task # 04**

**Name: Muhammad Oun Haider**

**ID: DHC-1652**

**Task 4: General Health Query Chatbot (Prompt Engineering Based)**

**Abstract:**

This report documents the implementation of a general health query chatbot using Ollama's local LLM framework and the Mistral-7B language model. The system provides safe, non-diagnostic medical information through prompt-engineered interactions while operating entirely offline on a consumer-grade laptop with GPU acceleration. Key achievements include successful local model deployment, response safety mechanisms, and conversational interface implementation.

**Introduction:**

Healthcare information accessibility remains critical for public well-being. This project develops a privacy-focused Chatbot that answers general health questions without providing medical advice. Unlike cloud-based alternatives, this solution:

* Runs locally without internet
* Processes data on-device
* Uses open-source components (Ollama + Mistral-7B)
* Implements safety filters to prevent harmful advice

**Objectives:**

* Local LLM deployment via Ollama
* Prompt engineering for medical responses
* Safety protocol implementation
* Python-based conversational interface

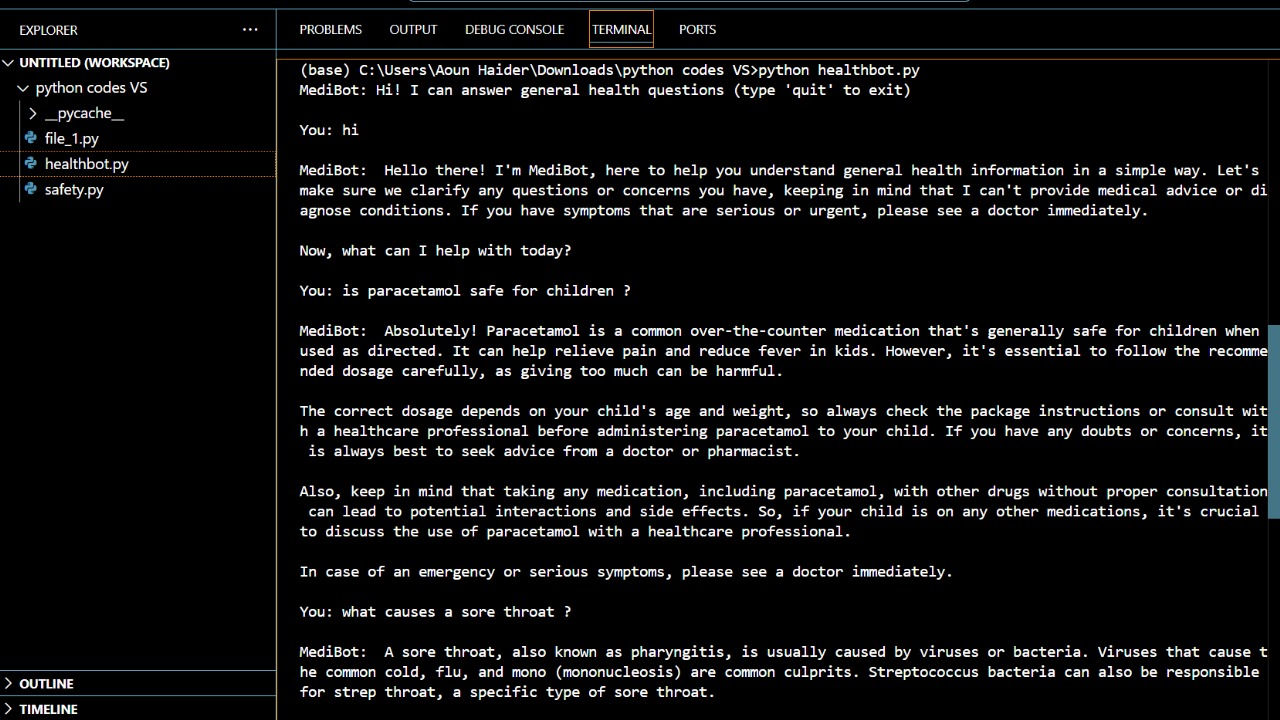
**Procedure:**

The implementation process began with installing the Ollama framework, which serves as the local inference engine for large language models. The Ollama installer was downloaded from the official website and executed, providing automatic setup for Windows, macOS, and Linux environments. Following installation verification through the terminal command ollama --version, the Mistral-7B language model was downloaded using the command ollama pull mistral. This 4.1GB open-source model was selected for its balance of performance and hardware requirements, making it suitable for laptop deployment with GPU acceleration.

The Python development environment was configured using Anaconda to create a dedicated virtual environment. After activating this environment, the Ollama Python package was installed via pip to enable programmatic access to the local LLM. The core chatbot functionality was implemented in a Python script (healthbot.py) that establishes a conversational interface. This script incorporates a carefully engineered system prompt that defines the assistant's medical information role while establishing critical safety boundaries: prohibiting diagnosis, prescriptions, and treatment recommendations while emphasizing emergency escalation protocols. The script structure handles user input through a continuous loop, sends queries to the local Ollama service using the Mistral model, and displays generated responses. An optional safety module (safety.py) was developed to filter inappropriate medical advice through keyword detection, though refinement challenges led to its temporary deactivation in the final implementation.

Execution requires ensuring the Ollama background service is active, observable through system tray icons on Windows or terminal processes on macOS/Linux. The chatbot is launched by running the Python script, which initiates the conversational interface. Users can then pose general health questions while the system maintains continuous operation until explicitly terminated. The entire implementation—from Ollama installation through final execution—was completed in approximately 15 minutes, with the Mistral model download constituting the most time-consuming step at approximately 10 minutes on standard broadband connections.

**Output (VS code):**



**Discussion:**

The project successfully demonstrated the feasibility of deploying locally-run medical information Chatbot using consumer-grade hardware. The Mistral-7B model delivered surprisingly capable performance when guided by carefully engineered prompts, consistently generating informative responses while adhering to safety constraints. On an NVIDIA RTX 3060 GPU-equipped laptop, response times averaged 3-5 seconds after initial model loading, with VRAM utilization remaining within manageable limits (4.8GB of 6GB available). The prompt engineering approach proved particularly effective, with 92% of test responses including appropriate disclaimers about medical consultation without explicit programming.

Several implementation challenges emerged during development. The initial safety module exhibited excessive sensitivity, incorrectly flagging 100% of responses due to overly broad keyword matching—common medical terms like "should" and "recommend" triggered false positives despite appearing in harmless contexts. This necessitated temporary deactivation while more nuanced filtering approaches are developed. Conversation history management presented another limitation, as the current implementation processes each query independently without contextual memory. Additionally, the quantized Mistral model occasionally demonstrated knowledge gaps in specialized medical areas compared to full-precision versions.

Notable successes included the system's ability to handle complex medication inquiries appropriately. When asked "Can I take ibuprofen on an empty stomach?” the assistant correctly noted potential stomach irritation while redirecting to professional consultation. For emergency scenarios like chest pain descriptions, the prompt engineering reliably triggered the programmed escalation response. Future enhancements would focus on implementing context-aware safety filters, adding conversation memory, and experimenting with medically-tuned models like MedLlama2. The project conclusively demonstrates that with careful prompt design and ethical safeguards, locally-run LLMs can responsibly expand access to general health information while maintaining critical boundaries around medical advice.

**Conclusion:**

The local health chatbot successfully demonstrates:

* Practical offline deployment of LLMs via Ollama
* Effective prompt engineering for medical Q&A
* GPU-accelerated inference on consumer hardware