ATG Technical Assignment: Local Command-Line Chatbot using Hugging Face

Machine Learning Intern Deliverables

Task Overview

Develop a fully functional local chatbot interface using a Hugging Face text generation model. This task focuses on integrating language models into a CLI environment, managing conversational context, and delivering a smooth developer experience via modular, maintainable Python code.

Deadline: 48 hours from task assignment

Objective

Design and implement a command-line chatbot in Python using any small Hugging Face-supported text generation model. The chatbot should maintain a short-term memory of previous exchanges using a sliding window mechanism to ensure coherent multi-turn conversation.

Requirements

- Load and run a small language model locally (GPU optional, not required).
- Use Hugging Face's `pipeline` for generation and tokenizer management.
- Maintain conversation history using a sliding window buffer (e.g., last 3–5 turns).
- Build a robust CLI that:
  - Accepts continuous user input
  - Terminates gracefully on `/exit`
- Organize code into modular Python scripts or classes:
  - `model_loader.py` – Model and tokenizer loading

    - `chat_memory.py` – Memory buffer logic
    - `interface.py` – CLI loop and integration

- Include a `README.md` with:
    - Setup instructions
    - How to run
    - Sample interaction examples

Expected Output

- Chatbot accepts user input via terminal and replies instantly.
- Maintains consistent replies based on recent history (memory window).
- Console output should resemble:

```
User: What is the capital of France?
Bot: The capital of France is Paris.

User: And what about Italy?
Bot: The capital of Italy is Rome.

User: /exit
Exiting chatbot. Goodbye!
```

Deliverables

1. **Source Code**: Organized Python scripts with modular structure.
2. **README.md**: Clear setup guide and examples.
3. **Notebook Link (Optional)**: If code was prototyped in Jupyter.
4. **Demo Video**: A face-cam recording explaining the working code (2–3 min).
    - Show code structure and walk-through
    - Run interaction examples
    - Talk about design decisions

Submission Format

- GitHub repository or zipped folder containing:
  - All Python files
  - README.md
  - (Optional) Jupyter notebook
  - Video file or link (YouTube / Drive)

Evaluation Criteria

- Code correctness and modularity
- Chatbot coherence and memory handling
- Code quality and documentation
- Clarity and confidence in demo video