CANDIDATE **ASSESMENT**

Jetlink.

2025

TASK

Version 1.3

PREPARED BY

Jetlink Team

TASK

Implementing a Conversational AI Chatbot with Long Term Memory

Objective:
The goal of this case study is for the candidate to implement a replica of ChatGPT’s memory tool.

The candidate will create a conversational AI chatbot capable of:

- Maintaining short-term memory (session-based context).
- Utilizing long-term memory (persistent user details across multiple sessions).
- Allowing the user to interact through a simple and intuitive UI.

Detailed Explanation of ChatGPT's Memory Tool

ChatGPT’s memory tool functions as follows:

Short-Term Memory (STM):

- Maintains context within a single conversation or session.
- Stores recent interactions, questions, and responses temporarily.
- Gets reset once the conversation ends or a new session starts.
-

Long-Term Memory (LTM):

- Stores user-specific details, preferences, and contextual information across multiple sessions.
- Enables personalized conversations by recalling stored user data.
- Allows for updating or deleting stored information as requested by the user.

Data Handling:

- Memory storage relies on embeddings (vector representations of information) to efficiently retrieve relevant past context.
- Embedding-based retrieval enhances the chatbot's ability to recall and utilize relevant historical information effectively.

Task

Your task is to develop a conversational AI chatbot that replicates the functionality of ChatGPT's memory tool as described above.

Technical Requirements

- **Programming Language:** Python (mandatory)
- **LLM Backend:** Use Google's Gemini-2.0-Flash model for response generation (provides up to 1M free tokens per day).
- **Embeddings:** Use Google's Text Embedding API (free tier available).
- **Frontend & Database:** Choose your preferred tech stack to build a user-friendly UI and persistent data storage solution.
- **Important:** DO NOT use API-based vector databases. Store embeddings locally (e.g., local file storage, SQLite, Excel, or in-memory).

Memory Management Details

You must clearly differentiate between and implement both short-term and long-term memory:

1. Short-Term Memory (STM)

- What it is: Contextual memory limited to an active session.
- Usage: Keeps track of the conversation flow within a session to generate contextually relevant responses.
- Implementation Expectation: Typically handled via session cookies, tokens, or server-side session management. Automatically cleared upon session termination or restart.

2. Long-Term Memory (LTM)

- What it is: Persistent, cross-session storage of key user-specific information.
- Usage: Personalized interactions, recalling user preferences, previously shared details, or contextual data.
- Implementation Expectation: Use embeddings to efficiently store and retrieve relevant user data. Integrate a simple, local persistent storage system. Provide functionality to update, retrieve, or delete stored data at user's request.

Deliverables

- A fully functional chatbot system replicating ChatGPT’s memory feature.
- Clearly structured and documented Python code.
- Simple, user-friendly UI for demonstration purposes.
- Instructions to set up and run the project locally

Evaluation Criteria

Your implementation will be evaluated on the following points:

1. Accuracy of Memory Implementation:
   - Clear distinction and correct functioning of both STM and LTM.
   - Efficient embedding-based memory retrieval system.

2. Technical Depth & Optimization:
   - Effective use of Google's Text Embedding API for memory retrieval.
   - Proper integration of Gemini-2.0-Flash LLM for natural and coherent chatbot responses.

3. Software Engineering Best Practices:
   - Clean, modular, maintainable codebase.
   - Appropriate documentation and readability.
   - Robust handling of edge cases and errors.

4. Intelligent Long-Term Storage:
   - Intelligent management of long-term data storage.
   - Avoidance of unnecessary personal data storage.
   - Effective data retrieval and privacy considerations.

5. UI Functionality:
   - Intuitive UI demonstrating STM and LTM clearly.
   - Easy-to-use interface to test memory features.

Bonus Points

Deployment using Docker.

Final Notes

This case study is designed to rigorously assess your capability in both AI Engineering (memory management, embeddings, LLM integration) and general software development skills.

Focus on demonstrating clarity of thought, technical expertise, and effective memory replication consistent with ChatGPT's original functionality.

Good luck!