Chat Assistant with Session Memory

Goal

Build a working demo of a chat assistant backend (using a terminal, Streamlit, or Gradio) that supports:

1. **Session (short-term) memory** is implemented via automatic summarization when the conversation context exceeds a configurable limit.
2. **Query understanding and refinement,** including rewriting ambiguous user queries, augmenting queries with relevant session memory, and generating clarifying questions when the user's intent remains unclear.

What we evaluate

- Clear pipeline design (chat input → session memory → query understanding → final prompt construction)
- Stable, structured outputs (schema-first approach)
- Clear code organization and documentation, with included test data

Duration

7 days

Allowed tools

- Python implementation required
- Any LLM / API (OpenAI, Claude, Gemini, open-source models, etc.)
- Any framework (LangChain, LangGraph, OpenAI Agents SDK, Google ADK), or no framework at all

Deliverables (Must Have)

1. Runnable Project

- A runnable demo of the chat assistant backend
- CLI / terminal demo is sufficient; Streamlit or Gradio is optional

2. Documentation

- `README.md` including:
  - Setup instructions

    - How to run the demo
    - High-level design explanation
    - Any assumptions or limitations

3. Structured Outputs

- Session summarization output must follow a clearly defined schema
- Query understanding output must follow a clearly defined schema

4. Test Data

- Provide at least **three conversation logs** (JSONL or text) that demonstrate:
    - Session memory being triggered
    - Ambiguous user queries

Functional Requirements

A. Session Memory via Summarization (Core)

Objective: Automatically trigger session summarization when the conversation context exceeds a configurable threshold (e.g., ~10k tokens).

Inputs: Conversation messages (role, content; timestamp optional)

Trigger: When context size exceeds the configured threshold

- **Core**: heuristic counting (characters or words) is acceptable
- **Plus:** tokenizer-based token counting

Output: A structured session summary (schema defined by the candidate)

Example (illustrative only):

```
{
  "session_summary": {
    "user_profile": {"prefs": [], "constraints": []},
    "key_facts": [],
    "decisions": [],
    "open_questions": [],
    "todos": []
  },
  "message_range_summarized": {"from": 0, "to": 42}
}
```

Memory Behavior

- Store the generated summary as **short-term session memory**
- Memory may be stored using the file system or a database

B. Query Understanding Pipeline (Core)

When receiving a new user query, the system should perform the following steps:

Step 1 — Rewrite / Paraphrase (if ambiguous)

- Detect whether the query is ambiguous
- If ambiguous, generate a rewritten or clarified version
- Output must include `is_ambiguous: true | false`

Step 2 — Context Augmentation

- Build an augmented context by combining:
  - The most recent N conversation messages
  - Relevant fields from short-term session memory

Step 3 — Clarifying Questions (if still unclear)

- If the query remains unclear after rewriting, generate **1–3 clarifying questions** for the user

Example (illustrative only):

```
{
  "original_query": "...",
  "is_ambiguous": true,
  "rewritten_query": "...",
  "needed_context_from_memory": ["user_profile.prefs",
"open_questions"],
  "clarifying_questions": ["...", "..."],
  "final_augmented_context": "..."
}
```

Demo Requirements (Core)

Your demo must clearly demonstrate **both** of the following flows:

Flow 1 — Session Memory Trigger

- Load a long conversation log
- Show context size increasing
- Demonstrate summarization being triggered
- Print or log the generated summary

Flow 2 — Ambiguous query handling

- Run at least one ambiguous query from the test data
- Show: Query rewriting → Context augmentation → Clarifying questions (if applicable)

Scoring Rubric (10 points total)

Category	Points
Core features work end-to-end	0–6
Structured outputs & basic validation	0–1
Code structure & readability	0–2
Documentation & included test data	0–1