Evolusis – Backend Developer Assignment

Objective

Build a backend service that demonstrates your ability to integrate **LLM reasoning** (e.g., OpenAI GPT) with **external API tools** such as weather or Wikipedia lookups.
The goal is to simulate a simplified **AI agent** that can think, decide, and act — not just generate text.

Task Overview

Create a **FastAPI backend** that exposes a single endpoint:

Endpoint:
`POST /ask`

Request Example:

```
{
  "query": "What is the weather in Paris today?"
}
```

Response Example:

```
{
  "reasoning": "The user asked about weather, so I fetched data from
OpenWeather API and combined it with reasoning from GPT.",
  "answer": "It's 21°C and partly cloudy in Paris today."
}
```

What Your Agent Should Do

1. **Receive a user query** through the `/ask` endpoint.
2. **Decide intelligently** if it needs to:
    - Answer directly using an **LLM API** (e.g., OpenAI, Hugging Face, Anthropic, etc.), or
    - **Call an external API** (e.g., Weather, Wikipedia, or News) to get factual data.
3. **Combine** the external data (if fetched) with the LLM's reasoning to form a final, coherent answer.
4. Return both the reasoning and the final answer in a JSON response.

Technical Requirements

- Use **Python + FastAPI** for the backend.
- Use at least one **LLM API** (e.g., OpenAI GPT-4, GPT-3.5, or Hugging Face model).
- Use at least one **external API** (for example):
    - OpenWeatherMap API
    - Wikipedia API
    - News API
- Your agent should demonstrate **decision-making ability** — i.e., it knows when to use the LLM alone vs. when to make an API call.
- Code should be **clean, modular, and well-documented**.

Bonus (Optional Enhancements)

These are not mandatory but will earn bonus points:

- Implement a **short-term memory** to remember the last few user queries.
- Add **speech input/output** (using Whisper or ElevenLabs APIs).
- Use **LangChain** or similar frameworks for better tool orchestration.
- Include **error handling and logging** for agent decisions and API calls.

Submission Requirements

Please submit the following:

1. A **GitHub repository link** containing your code.
2. A short **README.md** explaining:
    - How your solution works.
    - Which APIs you used and why.
    - How to run and test the project.
3. A **2-3-minute Loom video** explaining your implementation and design approach.

Evaluation Criteria

Area	Weight	What We're Looking For
Technical Implementation	40%	Working FastAPI backend, proper API integration, logical flow
AI Reasoning Logic	25%	Intelligent decision-making between LLM and API
Code Quality	15%	Clean structure, comments, modularity
Creativity & Problem-Solving	10%	Innovative or elegant design choices
Documentation & Clarity	10%	Clear explanations in README

Example Queries to Test

- "What's the weather in London today?"
- "Who invented the telephone?"
- "Summarize the latest news about artificial intelligence."
- "What's the capital of Japan?"

Deliverables Deadline

Please submit your solution by Saturday 6 pm.