LangGraph Assignment: Create an Agent Using LLM and Custom Mathematical Functions

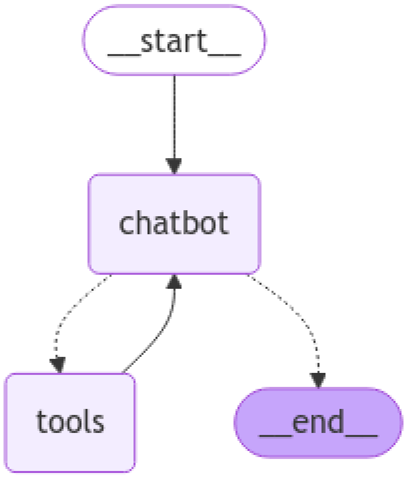
Problem Statement:

Create an agent using LangGraph that answers general questions using the **LLM,** and when asked to perform mathematical operations (addition, subtraction, multiplication, and division), it calls four predefined functions (`plus`, `divide`, `sub`, `mul`) for answering. The agent should handle both general and math-related queries seamlessly.

Requirements:

1. Use an LLM API (such as Groq API or Gemini API) or local LLM using Ollama to assist in reasoning. Make sure to set up an API key and install any required libraries like openai in Python.
2. You will create four custom functions for the following operations:
   - `plus(a, b)`: Add two numbers.
   - `subtract(a, b)`: Subtract two numbers.
   - `multiply(a, b)`: Multiply two numbers.
   - `divide(a, b)`: Divide two numbers, with error handling for division by zero.
3. When the agent receives a mathematical query (e.g., "What is 5 plus 3?" or "How much is 8 divided by 2?"), it should call the corresponding predefined function.
4. If the query is not mathematical, it should forward the message to the **LLM** for a general response.
5. The solution should handle the graph-building process, including tool integration and defining the appropriate graph flow using LangGraph.

Architecture:



Approach:

1. **Define Custom Functions**:
    - Implement `plus`, `subtract`, `multiply`, and `divide` as custom tools.
2. **Integrate LLM**:
    - Use LLM to answer general questions.
3. **Create LangGraph**:
    - Set up a state graph with two nodes: one for the chatbot (LLM) and one for the mathematical tools.
4. **Conditional Edges**:
    - Use conditional edges to call the mathematical tools when a math query is detected.
5. **Test Queries**:
    - Test the system with both general queries and math-related queries.

Expected Deliverables

1. **Code Submission**
    - A Python script or ipynb file implementing the **agent** using LLM and
    - **Report**
        1. A brief explanation (in markdowns) describing how you used the LLM and created the agent. Explain how the agent works
        2. Explain code and flow of your program.
