Data Extraction and Action Agent for Event Management

Objective: Build an action-based LangChain agent that can extract structured information from natural language event descriptions and then use that information to "add" an event to a simulated calendar system.

Scenario: An event organizer wants an intelligent assistant that can process user requests to schedule events. The assistant should identify key event details (date, time, title, location, attendees) and then confirm the event addition.

Requirements:

1. **Define Event Management Tool (Simulated):**
    - **Add Event Tool:** Create a Python function add_event_to_calendar that simulates adding an event. It should accept parameters: title (str), date (str, e.g., "YYYY-MM-DD"), time (str, e.g., "HH:MM"), location (str, optional), and attendees (list of str, optional). The function should return a confirmation message including the details it received (e.g., "Event 'Meeting' on 2025-07-15 at 10:00 AM successfully added.").
2. **Agent Development with Structured Output Parsing:**
    - Use LangChain's initialize_agent with an appropriate agent type.
    - Crucially, the agent needs to perform *information extraction* before calling the tool. Consider using:
        - **PydanticOutputParser:** Define a Pydantic model for EventDetails (title, date, time, location, attendees) to guide the LLM in extracting structured data from the user's natural language input.
        - **Tool Description:** The add_event_to_calendar tool's description should clearly specify the expected input parameters and their types.
    - The agent should be able to:
        - Parse natural language input (e.g., "Schedule a team meeting for tomorrow at 3 PM in Conference Room A with Alice and Bob about Q3 planning.").
        - Extract the title, date, time, location, and attendees into the structured format.
        - Call the add_event_to_calendar tool with the extracted parameters.
        - Handle missing information gracefully by either asking clarifying questions or using default values (e.g., current date if "tomorrow" is given).
3. **Agent Interaction:**
    - Test the agent with various event scheduling requests:
        - "Book a dental appointment for me on July 20, 2025, at 10:00 AM."
        - "Organize a project kickoff meeting next Monday at 9 AM." (Requires inferring the date)

    - "I need to add a reminder to buy groceries this evening." (Less structured, see how it handles)
    - "Create an event: Presentation on new features next Friday at 2 PM, in the main hall, with the sales team."

Deliverables:

- A Python script (event_agent.py) containing the agent, tool, and Pydantic model definitions.
- A requirements.txt file.
- Instructions on running the agent and examples of interactions with different event descriptions.

Evaluation Criteria:

- Accuracy of information extraction from natural language into structured data using Pydantic.
- Agent's ability to correctly map extracted data to the add_event_to_calendar tool's parameters.
- Effective use of agent capabilities for multi-step reasoning (extract then act).
- Robustness in handling variations in user input and missing information.
- Clarity and helpfulness of the agent's responses.