Assessment Task - 1

Build a Personal Productivity Assistant with LangChain

Course: *Functions, Tools, and Agents with LangChain*

Objective

Build a conversational agent that helps users **manage tasks and reminders** using LangChain's **LCEL**, **OpenAI function calling**, and **tool routing**.

Task Requirements

1. Structured Output Generation with Function Calling

Define **three schemas** using OpenAI function calling:

- **add_task(title: str, deadline: str)**
  - Adds a new task with a deadline.
- **set_reminder(task_title: str, reminder_time: str, priority: str)**
  - Sets a reminder for a specific task at a given time with priority (high, medium, low).
- **get_query(type: str)**
  - Retrieves tasks or reminders based on the user's query (e.g., "What are my tasks for today?").

Ensure the **LLM correctly maps natural language inputs** to structured outputs.

Examples:

✅*User Input:* "Add a task to submit the project report by Monday."
✅*Structured Output:*

```
□{
  "title": "Submit the project report",
  "deadline": "(Monday's date based on execution context)"
}
```

□✅*User Input:* "Remind me to call John at 3 PM with high priority."
✅*Structured Output:*

```
□{
  "task_title": "Call John",
  "reminder_time": "15:00",
  "priority": "high"
}
```

□✅*User Input:* "What tasks do I have today?"
✅*Structured Output:*

```
□{
 "type": "tasks"
}
```

□Store the Tasks / Reminders using arrays, no need of implementing DB

---

2. Implement LCEL Chains

Use **LangChain Expression Language (LCEL)** to:

- **Chain** the LLM with custom functions.
- **Handle ambiguous queries** by prompting the user for more details (e.g., *"I need help organizing my day"* → "I can help you fetch tasks / reminders or create task / reminders").

---

3. Tagging and Entity Extraction

Build a **chain to extract**:

- **Deadlines** (dates/times).
- **Priority Levels** (e.g., "high", "low").

This extraction should **auto-populate function arguments** for add_task and set_reminder.

---

4. Tool Selection and Routing

Create three tools:

1. **add_task** – Adds a task based on extracted details.
2. **set_reminder** – Creates reminders with priority and time.
3. **get_query** – Routes queries like "What's my schedule today?" to retrieve relevant information.

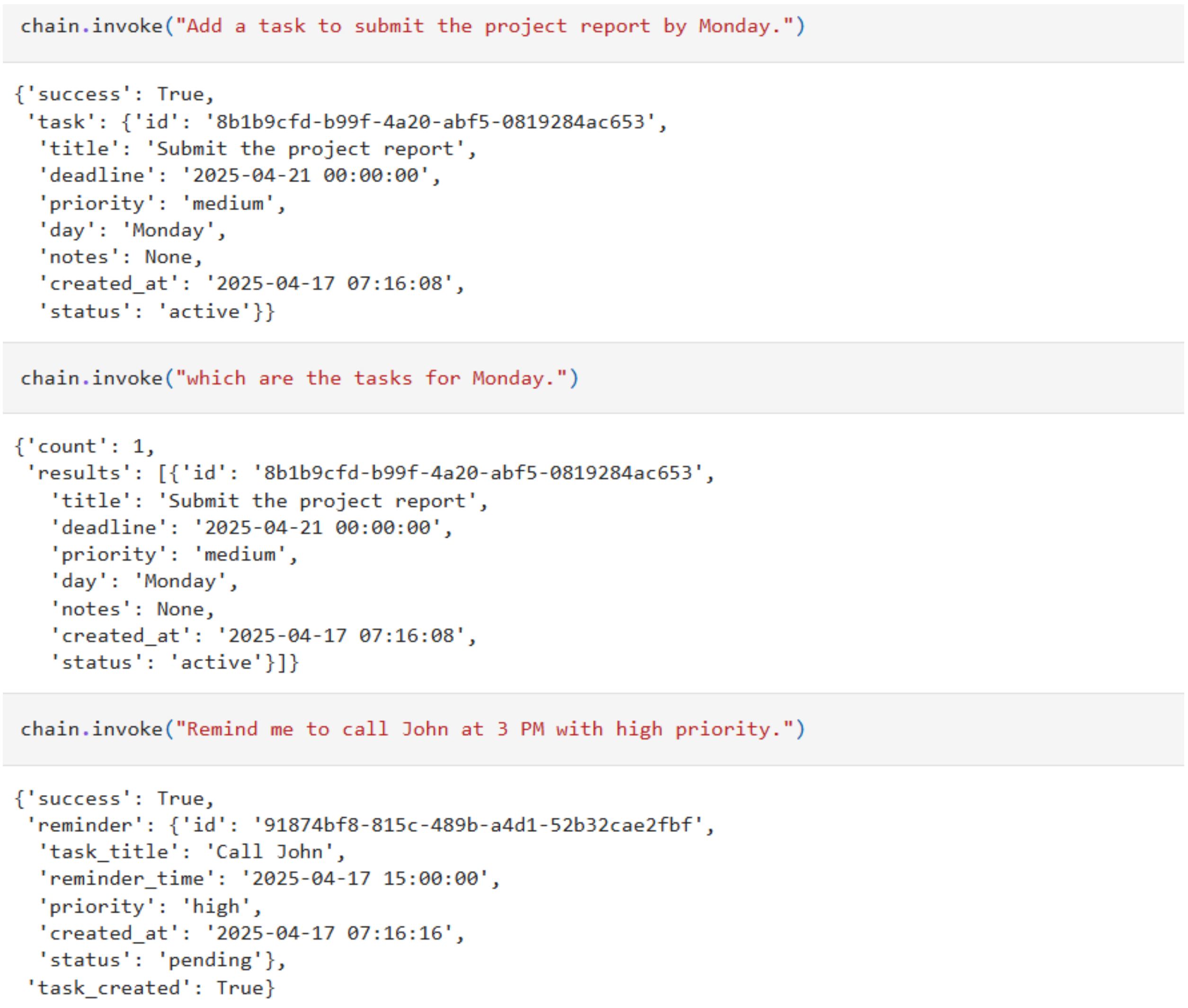
Implement **routing logic** to select the correct tool based on user intent.

Deliverables

1. Code Submission

Provide a **Python script or Jupyter Notebook** that includes:
✅**LCEL chains** for extraction, routing, and tool execution.
✅**Example dialogues** demonstrating the assistant's functionality.

Notebook :
https://github.com/shivamgiri007/ShivamOnboarding/blob/main/Assignmetns/Assesment1.ipynb



```
chain.invoke("Add a task to submit the project report by Monday.")
```

```
{'success': True,
 'task': {'id': '8b1b9cfd-b99f-4a20-abf5-0819284ac653',
  'title': 'Submit the project report',
  'deadline': '2025-04-21 00:00:00',
  'priority': 'medium',
  'day': 'Monday',
  'notes': None,
  'created_at': '2025-04-17 07:16:08',
  'status': 'active'}}
```

```
chain.invoke("which are the tasks for Monday.")
```

```
{'count': 1,
 'results': [{'id': '8b1b9cfd-b99f-4a20-abf5-0819284ac653',
   'title': 'Submit the project report',
   'deadline': '2025-04-21 00:00:00',
   'priority': 'medium',
   'day': 'Monday',
   'notes': None,
   'created_at': '2025-04-17 07:16:08',
   'status': 'active'}]}
```

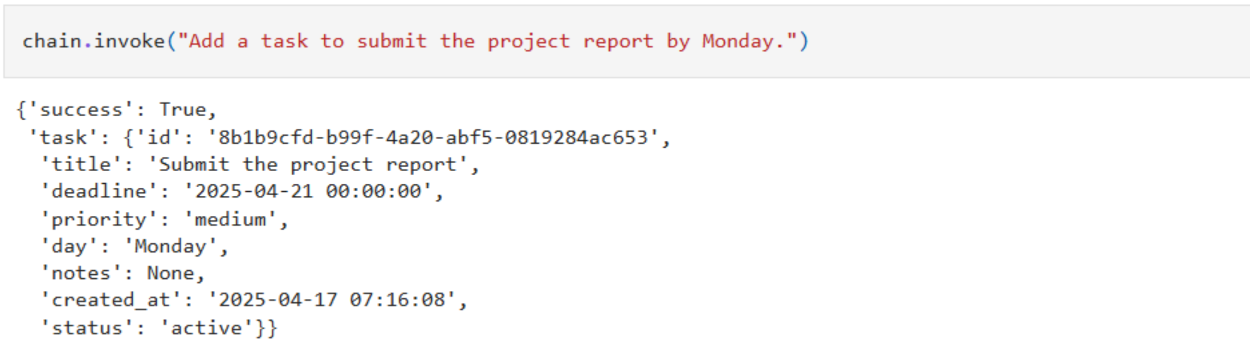
```
chain.invoke("Remind me to call John at 3 PM with high priority.")
```

```
{'success': True,
 'reminder': {'id': '91874bf8-815c-489b-a4d1-52b32cae2fbf',
  'task_title': 'Call John',
  'reminder_time': '2025-04-17 15:00:00',
  'priority': 'high',
  'created_at': '2025-04-17 07:16:16',
  'status': 'pending'},
 'task_created': True}
```

2. Example Interactions

Submit **three sample conversations**, including:

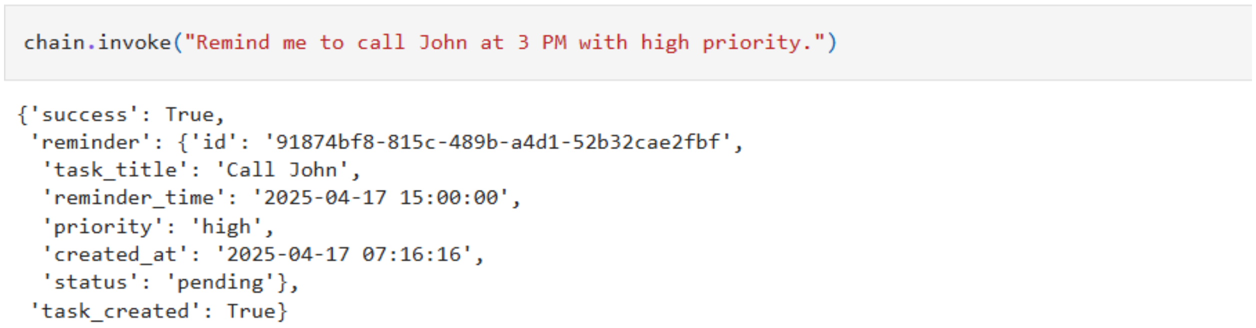
1. **Task Addition** (showing deadline extraction).



```
chain.invoke("Add a task to submit the project report by Monday.")
```

```
{'success': True,
 'task': {'id': '8b1b9cfd-b99f-4a20-abf5-0819284ac653',
  'title': 'Submit the project report',
  'deadline': '2025-04-21 00:00:00',
  'priority': 'medium',
  'day': 'Monday',
  'notes': None,
  'created_at': '2025-04-17 07:16:08',
  'status': 'active'}}
```

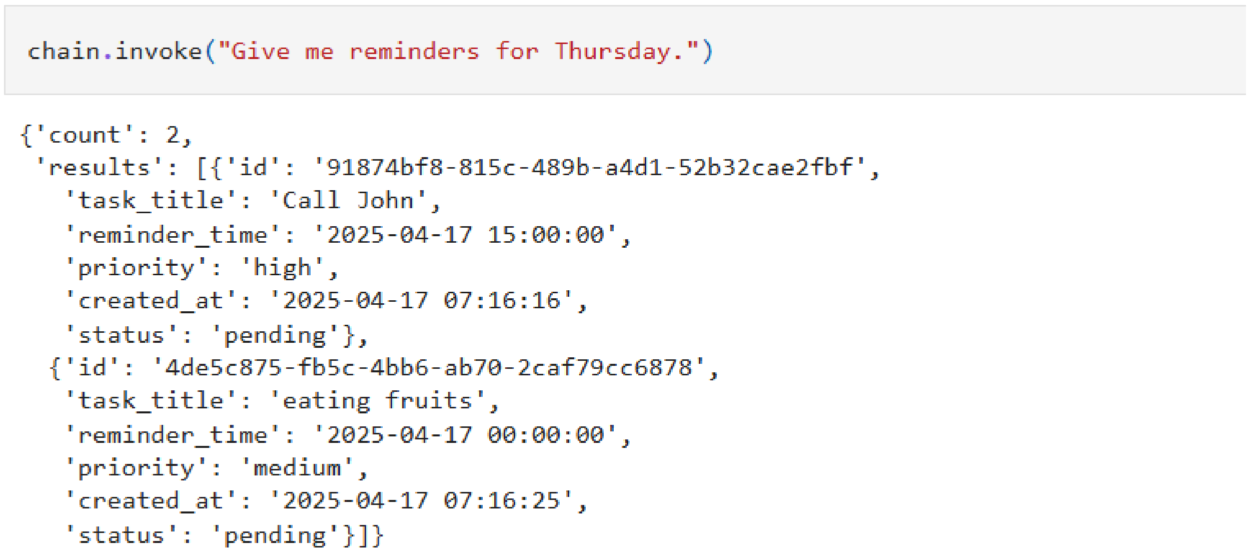
2. **Reminder Addition** (showing priority extraction and reminder time).



```
chain.invoke("Remind me to call John at 3 PM with high priority.")
```

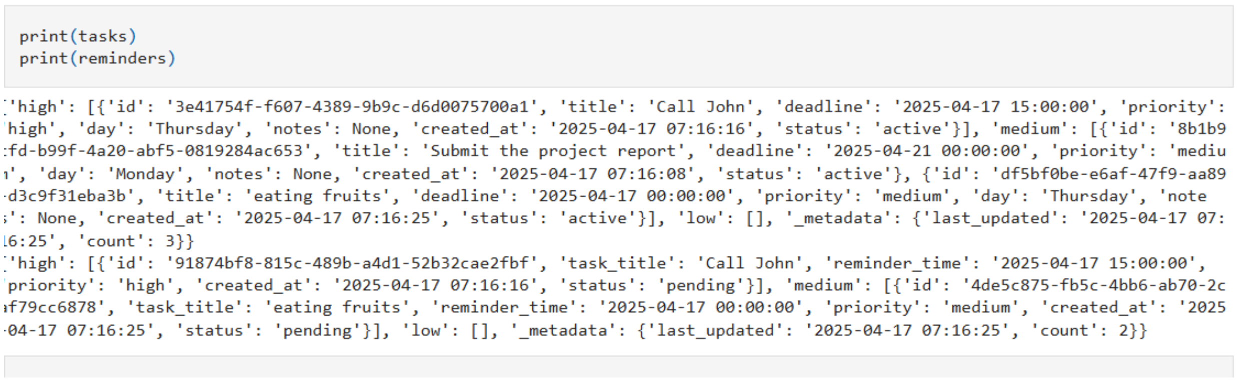
```
{'success': True,
 'reminder': {'id': '91874bf8-815c-489b-a4d1-52b32cae2fbf',
  'task_title': 'Call John',
  'reminder_time': '2025-04-17 15:00:00',
  'priority': 'high',
  'created_at': '2025-04-17 07:16:16',
  'status': 'pending'},
 'task_created': True}
```

3. **Routed Query Handling** (e.g., retrieving tasks or reminders).



```
chain.invoke("Give me reminders for Thursday.")
```

```
{'count': 2,
 'results': [{'id': '91874bf8-815c-489b-a4d1-52b32cae2fbf',
   'task_title': 'Call John',
   'reminder_time': '2025-04-17 15:00:00',
   'priority': 'high',
   'created_at': '2025-04-17 07:16:16',
   'status': 'pending'},
  {'id': '4de5c875-fb5c-4bb6-ab70-2caf79cc6878',
   'task_title': 'eating fruits',
   'reminder_time': '2025-04-17 00:00:00',
   'priority': 'medium',
   'created_at': '2025-04-17 07:16:25',
   'status': 'pending'}]}
```



```
print(tasks)
print(reminders)
```

```
'high': [{'id': '3e41754f-f607-4389-9b9c-d6d0075700a1', 'title': 'Call John', 'deadline': '2025-04-17 15:00:00', 'priority':
high', 'day': 'Thursday', 'notes': None, 'created_at': '2025-04-17 07:16:16', 'status': 'active'}], 'medium': [{'id': '8b1b9
fd-b99f-4a20-abf5-0819284ac653', 'title': 'Submit the project report', 'deadline': '2025-04-21 00:00:00', 'priority': 'mediu
', 'day': 'Monday', 'notes': None, 'created_at': '2025-04-17 07:16:08', 'status': 'active'}, {'id': 'df5bf0be-e6af-47f9-aa89
d3c9f31eba3b', 'title': 'eating fruits', 'deadline': '2025-04-17 00:00:00', 'priority': 'medium', 'day': 'Thursday', 'note
': None, 'created_at': '2025-04-17 07:16:25', 'status': 'active'}], 'low': [], '_metadata': {'last_updated': '2025-04-17 07:
6:25', 'count': 3}}
'high': [{'id': '91874bf8-815c-489b-a4d1-52b32cae2fbf', 'task_title': 'Call John', 'reminder_time': '2025-04-17 15:00:00',
priority': 'high', 'created_at': '2025-04-17 07:16:16', 'status': 'pending'}], 'medium': [{'id': '4de5c875-fb5c-4bb6-ab70-2c
f79cc6878', 'task_title': 'eating fruits', 'reminder_time': '2025-04-17 00:00:00', 'priority': 'medium', 'created_at': '2025
04-17 07:16:25', 'status': 'pending'}], 'low': [], '_metadata': {'last_updated': '2025-04-17 07:16:25', 'count': 2}}
```

3. Short Reflection (200–300 words)

- Explain your **approach to routing** and handling **ambiguous inputs**.

  Ans : Created Functions with by converting tools to openai functions, binded this tools to the llm , created a custom routing function to pass the attributes of function_call to that particular tool , binded this tools in router , and created llm chain using langchain expression language to let llm decide which functionality llm should have to do.

- Describe **one challenge** faced while using **LCEL** and how you resolved it.

  Ans : Faced Issue in realising the implementation of LCEL , Faced Issue in passing values using correct parsers as per task requirement .