Northwestern Memorial Hospital – Healthcare Agent

1. Use of LangChain/LangGraph Framework

I built this hospital inquiry system using LangChain and LangGraph to create a network of specialized AI agents. Each department is represented by an agent that can answer domain-specific questions. I imported the necessary components:

```
from langchain_openai import ChatOpenAI
from langchain_core.prompts import ChatPromptTemplate
from langchain_core.messages import SystemMessage,
HumanMessage
from langgraph.graph import StateGraph, END
```

This creates a seamless coordination system where patient inquiries flow through an operator to the right department.

2. Graph Construction for Hospital Departments

Following the design diagram, I created a network of department agents:

- Operator Agent - Routes inquiries to appropriate departments
- ER Department Agent - Handles emergency services questions
- Primary Care Department Agent - Answers general healthcare inquiries
- Radiology Department Agent - Provides imaging services information
- Pediatrics Department Agent - Addresses children's health concerns
- Cardiology Department Agent - Answers heart-related questions
- Billing/Insurance Agent - Handles financial inquiries

I connected these into a coherent workflow:

```
workflow = StateGraph(AgentState)
workflow.add_node("operator", operator_agent)
# [Added all department nodes]
workflow.add_conditional_edges("operator", lambda state:
state["department"], {...})
```

3. Operator Agent Intent Classification

The operator agent analyzes incoming questions and routes them to the appropriate department using the exact prompt format specified:

```
prompt = ChatPromptTemplate.from_messages([
    SystemMessage(content="""You are an operator agent at
Northwestern Memorial Hospital.
    Your job is to analyze patient/visitor questions and
determine which department should handle them.
    Possible intents include: ER, primary_care, radiology,
pediatrics, cardiology, billing
    Return all relevant intents as a comma-separated list
without any duplication.
    Ensure that each intent is distinct and only included
if it is clearly relevant to the input."""),
    HumanMessage(content="Classify the user's intents based
on the following input: '{question}'")
])
```

4. Department-Specific Knowledge Base

I created a comprehensive knowledge base with 10 Q&A pairs for each department:

```
DEPARTMENT_KB = {
    "er": [
        {"question": "What are the current ER wait times?",
"answer": "Wait times vary..."},
        # 9 more Q&A pairs
    ],
    # All other departments with 10 Q&A pairs each
}
```

Each department agent first searches this knowledge base for relevant answers before generating responses.

5. Department Agent Response Generation

Department agents are specialized to handle questions in their domain:

```
def pediatrics_department_agent(state: AgentState) ->
AgentState:
    # Search knowledge base for matches
    # Generate appropriate response based on the question
    # Return detailed, accurate information
```

When a perfect match isn't found, the system uses relevant information to construct a helpful response.

6. End-to-End Processing Flow

The system ensures complete coordination from question input through routing to response:

1. Patient submits a question
2. Operator agent analyzes intent and routes accordingly
3. Department agent accesses specialized knowledge
4. Patient receives expert department-specific response

7. Test Case Demos

I ran the system with six test cases to verify correct routing and responses:

1. "How can I tell if my child has RSV?" → Pediatrics → Common symptoms explanation
2. "Can I visit my friend in the ER?" → ER → Visitor policy details
3. "How should I prepare for a CT scan?" → Radiology → Preparation instructions
4. "Has my doctor reviewed my test results?" → Primary Care → Results timeline
5. "Cardiologist appointments next week?" → Cardiology → Scheduling information
6. "Help understand my medical bill?" → Billing → Insurance explanation
7. Each test demonstrated successful intent classification, routing, and domain-appropriate responses from the specialized department agents.

GitHub repo: https://github.com/ryano0oceros/msds442-assign4-nwhospital