Lecture 20: Real-World Use Case – DevOps Agent

🎯 Learning Objectives

By the end of this lecture, you should be able to:

- Understand how LLM agents can support DevOps workflows.
- Build an agent that monitors, queries, and responds to system states.
- Integrate tools like shell commands, logging, and alerting systems.
- Deploy a task-specific automation agent for system health and maintenance.

🧩 Key Concepts

What Is a DevOps Agent?

An agent that:

- Observes system metrics or logs
- Executes diagnostic or repair commands
- Sends alerts or reports
- Operates on a schedule or in response to triggers

Typical DevOps Tasks

- Uptime and availability checks
- Log parsing and anomaly detection
- Disk/memory usage monitoring
- Automated remediation (e.g., restart service)

🛠️ Required Tools/Libraries

- Python
- psutil (for system info)
- subprocess (for shell execution)
- LangChain (optional for prompt/agent structure)
- cron or schedule (for time-based tasks)

  ```
  pip install psutil schedule langchain openai
  ```

🔬 Hands-on Exercise: Create a System Monitor Agent

Goal: Build an agent that checks CPU usage, restarts services if needed, and sends a summary.

Step 1: Define system info tools

```
import psutil

def get_cpu_load():
    return psutil.cpu_percent(interval=1)

def get_memory_usage():
    return psutil.virtual_memory().percent
```

Step 2: Define command execution tool

```
import subprocess

def restart_service(service_name):
    return subprocess.getoutput(f"sudo systemctl restart {service_name}")
```

Step 3: Implement decision logic with LLM

```
prompt = f"""
CPU Load: {get_cpu_load()}%
Memory Usage: {get_memory_usage()}%

If either is above 80%, recommend action.
"""

response = llm.generate(prompt)
print("Agent Decision:", response)
```

Step 4: Automate and schedule

```
import schedule
import time

def run_monitor():
    # Collect data, generate prompt, log result
    pass

schedule.every(5).minutes.do(run_monitor)

while True:
    schedule.run_pending()
    time.sleep(1)
```

Bonus:

- Add Slack or email notifications.
- Allow LLM to inspect logs and recommend configurations.
- Use a file-based memory to track past issues and resolutions.
