Python for DevOps – Final Project Assignment

This session serves as a culmination of your learning in the Python for DevOps course.

You will select one project from the list below and develop a functional automation script or tool that demonstrates your understanding of Python's application in DevOps. Work individually or in pairs, and prepare to present your work at the end of the session.

Objective

Each project integrates multiple Python concepts and libraries such as subprocess, psutil, boto3, paramiko, logging, and GitPython. Your goal is to create a practical, working automation that performs a meaningful DevOps-related function.

Project 1: Remote Server Health Dashboard

Create a script that connects to multiple Linux servers (or localhost) using paramiko to gather remote system information. The script should:

1. Execute uptime, df -h, and free -m commands.
2. 2. Parse and save results into JSON or CSV format.
3. Log all successful and failed connections.
4. (Optional) Use pandas or matplotlib to visualize data trends across servers.

Project 2: Mini CI/CD Pipeline Simulator

Simulate a simplified CI/CD pipeline to demonstrate automation concepts. The pipeline should:

1. Pull the latest code from a Git repository using subprocess.
2. Run unit tests using unittest or another simple test file.
3. Package the code if tests pass using shutil or zipfile.
4. Deploy the package to a /deploy folder or AWS S3.
5. Log the result of each step with timestamps.
6. (Optional) Send a notification (email or webhook) after deployment.

Project 3: Scheduled Backup and Monitoring Agent

Implement a local monitoring and backup system. The script should:

1. Monitor a target directory for changes using watchdog.
2. Automatically back up changed files to AWS S3.
3. Maintain a local log of all uploads and events.
4. Send a notification if an upload fails.

Project 4: Automated System Audit and Backup Tool

Develop a daily automation script that collects and stores system health data. The script should:

1. Collect CPU, memory, and disk usage using psutil and shutil.
2. Save results with timestamps in a log file.
3. Compress and archive older logs.
4. Upload the archives to an AWS S3 bucket using boto3.
5. Keep only the last seven days of logs.
6. Log all operations using the logging module.
7. (Optional) Use subprocess to automate Git commits of log data.

Project Deliverables

Each team should submit and present the following:

1. A working Python script that performs the described automation.
2. A short 5-minute presentation explaining the purpose, design, and key features.
3. A brief written reflection describing what was learned and any challenges encountered.