PROGRAMMING FOUNDATIONS :PYTHON

PALLAVI VAIDYA

COURSE CONTENTS

- Introduction to python and programming basics
- Data types, structures and operators
- Control structures and functions
- Introduction to APIs' Logging and testing
- OO programming and Packages
- Advanced API programming ,logging and testing

COURSE CONTENTS

- Databases and data handling in python
- Security Practices
- Cloud Security
- AWS Config
- Advanced Logging, Data Analysis, and Monitoring in Cloud Environments

EVALUATION PLAN

- Assignments (4):50%
- Quizzes(4): 20%
- Mid-Term:15%
- Final :15%
- In order to pass the course, you must achieve a 50% weighted average across the Term Tests and a minimum of 50% overall in the course.

LATE SUBMISSION

- A penalty of 10% of the value of the deliverable will be immediately deducted for late submissions.
- An additional 10% of the value of the deliverable will be deducted for each subsequent day (includes weekends and holidays) to a maximum of 3 days at which point the assignment or project will be assigned a mark of "zero".

INTRODUCTION TO PYTHON AND DEV-OPS

WHAT IS DEV-OPS?

- DevOps = Development + Operations
- A set of practices, tools, and cultural philosophies
- Brings together **software developers (Dev)** and **IT operations (Ops)**

GOALS OF DEV-OPS?

- Speed up software delivery 🚀
- Improve collaboration 🤝
- Automate processes ⚙️
- Ensure reliability & stability 🔒

SPEED UP SOFTWARE DELIVERY

- Automates the build, test, and deployment pipeline.
- Helps release new features and bug fixes to users faster.
- Reduces time-to-market.

IMPROVE COLLABORATION

- Breaks silos between development and operations teams.
- Encourages shared responsibility for code and infrastructure.
- Improves communication and teamwork.

AUTOMATE PROCESSES

- Replaces manual, error-prone tasks with automation.
- Covers testing, integration, deployments, monitoring, and scaling.
- Ensures consistency and reliability.

ENSURE RELIABILITY & STABILITY

- Continuous monitoring detects issues early.
- Rollback and recovery mechanisms minimize downtime.
- Ensures systems are secure, stable, and resilient.

KEY IDEAS OF DEV-OPS?

- **Continuous Integration (CI):** merge code changes frequently
- **Continuous Delivery/Deployment (CD):** automate release process
- **Automation:** testing, deployment, monitoring
- **Collaboration & Communication:** break silos
- **Monitoring & Feedback:** use data to improve

EXAMPLES

- **Without DevOps:**
- Developer → Ops → delays, miscommunication, “works on my machine”
- **With DevOps:**
- Automated testing & deployment pipeline
- Faster delivery, fewer errors, better teamwork

EXAMPLES

- Git
- Jenkins
- Docker
- Kubernetes
- Ansible
- Terraform
- AWS / Azure DevOps

PYTHON AND DEV-OPS

PYTHON

- High-level, interpreted programming language
- Emphasizes **readability and simplicity**
- Supports **multiple programming paradigms** (procedural, object-oriented, functional)

WHY PYTHON IN DEVOPS

- **Simple and readable syntax** → fast development
- **Cross-platform support** → works on Linux, Windows, macOS
- **Rich ecosystem of libraries** → automation, cloud, CI/CD
- **Strong community support** → plenty of DevOps tools and scripts

WHY PYTHON IN DEVOPS

- **Simple and readable syntax** → fast development
- **Cross-platform support** → works on Linux, Windows, macOS
- **Rich ecosystem of libraries** → automation, cloud, CI/CD
- **Strong community support** → plenty of DevOps tools and scripts

WHY PYTHON IN DEVOPS

- **Automation & Scripting:** Automate repetitive tasks (e.g., backups, deployments)
- **Configuration Management:** Tools like Ansible use Python modules
- **Monitoring & Logging:** Write scripts to monitor servers, analyze logs
- **CI/CD Pipelines:** Integrate with Jenkins, GitLab CI, or GitHub Actions
- **Cloud & Infrastructure:** Manage AWS, Azure, GCP using Python SDKs

PYTHON TOOLS IN DEVOPS

- **Ansible:** Automation and configuration management
- **Fabric / Invoke:** Remote command execution
- **Boto3:** AWS SDK for Python
- **SaltStack:** Server orchestration
- **Pytest / Robot Framework:** Automated testing

RUN PYTHON PROGRAM USING GITAND VS CODE

- **Windows**
- Download VS Code from the official site (search "VS Code download").
- Run the installer → keep defaults.
- (Optional) During setup, tick **"Add to PATH"** and **"Add 'Open with Code'"**.
- **macOS**
- Download the `.dmg`, drag **Visual Studio Code** to **Applications**.
- Open VS Code → **Cmd+Shift+P** → run: `Shell Command: Install 'code' command in PATH`.

RUN PYTHON PROGRAM USING GIT AND VS CODE

- Install Python
- Make sure Python is installed on your system.
- Check in terminal:
- python --version
- Install VS Code & Python Extension
- Open VS Code.
- Go to **Extensions (left sidebar → square icon)**.
- Search for **"Python"** by Microsoft and install it.

RUN PYTHON PROGRAM USING GIT AND VS CODE

- In VS Code, create a new file → save it as **`hello.py`**.
- Example code:
- print("Hello, VS Code!")
- **Select Python Interpreter**
- Press **Ctrl+Shift+P** (or Cmd+Shift+P on Mac).
- Search for **Python: Select Interpreter**.
- Pick the one that matches your Python installation or virtual environment.

RUN PYTHON PROGRAM USING GIT AND VS CODE

- Run with Play Button
- Open your **`hello.py`** file.
- You'll see a ▶ Run Python File button at the top right.
- Click it → your program runs in the terminal.

RUN PYTHON PROGRAM USING GIT AND VS CODE

- Clone the Repository
- Copy the repo link from GitHub (e.g., https://github.com/username/repo-name.git).
- In your terminal or VS Code:
- Move into the project folder:
- cd repo-name

RUN PYTHON PROGRAM