**✅ Module 1: Introduction to DevOps**

**1. What is DevOps?**

DevOps is a set of practices that combines software development (Dev) and IT operations (Ops). It aims to shorten the development lifecycle and deliver high-quality software continuously.

**2. What are the key principles of DevOps?**

* Continuous Integration and Delivery (CI/CD)
* Automation of processes
* Collaboration and communication
* Monitoring and feedback
* Infrastructure as Code (IaC)

**3. How is DevOps different from traditional software development?**

DevOps promotes automation, quick delivery, and continuous feedback, whereas traditional methods follow a linear, siloed approach with longer delivery cycles.

**4. Benefits of DevOps:**

* Faster release cycles
* Improved collaboration
* Better product quality
* Reduced failure rates and faster recovery
* Continuous delivery and integration

**5. What is Agile, and how is it related to DevOps?**

Agile is a methodology focused on iterative development. DevOps complements Agile by automating and integrating the development and deployment processes.

**✅ Module 2: Git and GitHub**

**1. What is Git?**

Git is a distributed version control system that tracks changes in source code during software development.

**2. Difference between git clone and git fork:**

* git clone: Copies a repository to your local machine.
* fork: Creates a personal copy of someone else's repository on GitHub.

**3. Basic Git Commands:**

* git init: Initialize a new repository
* git add: Stage files for commit
* git commit: Save changes to local repo
* git push: Upload local commits to remote
* git pull: Fetch and merge from remote repo

**4. What is a pull request?**

A pull request (PR) is a GitHub feature to request merging changes from one branch/repo into another, often used in team collaborations.

**5. How to handle merge conflicts?**

Identify conflicting files, edit them to resolve conflicts, then use:

git add <file>

git commit

**✅ Module 3: GitLab Essentials**

**1. What is GitLab?**

GitLab is a DevOps platform that provides version control (Git), CI/CD, issue tracking, and project management in one interface.

**2. What is a CI/CD pipeline?**

A CI/CD pipeline automates the building, testing, and deployment of applications. CI stands for Continuous Integration, CD for Continuous Delivery or Deployment.

**3. How does GitLab support CI/CD?**

Through .gitlab-ci.yml configuration file, GitLab runs pipelines automatically on push or merge events.

**4. Difference between GitHub and GitLab:**

GitHub focuses more on collaboration, while GitLab provides an integrated CI/CD solution out-of-the-box.

**5. Security issues in GitLab:**

* Token management
* Access control
* Secure storage of secrets
* Proper pipeline permissions

**✅ Module 4: Jenkins for CI**

**1. What is Jenkins?**

Jenkins is an open-source automation server used to automate CI/CD pipelines.

**2. Steps to set up Jenkins:**

* Download and install Jenkins
* Start Jenkins service
* Access via browser (localhost:8080)
* Install recommended plugins
* Create and configure a job or pipeline

**3. How to integrate GitHub with Jenkins?**

* Install GitHub plugin in Jenkins
* Link your GitHub repo in the Jenkins job configuration
* Use a webhook in GitHub to trigger builds

**4. What is a Jenkins pipeline?**

A series of automated steps (build, test, deploy) written as code using a Jenkinsfile.

**5. Real-world use case of Jenkins:**

Automating builds and deployments of applications after every code change to maintain CI/CD.

**✅ Module 5: Docker for Containerization**

**1. What is containerization?**

It is a lightweight virtualization method where applications run in isolated environments called containers.

**2. What is Docker?**

Docker is a platform to develop, ship, and run applications inside containers.

**3. What are Docker images and containers?**

* Image: Read-only template with application code and dependencies.
* Container: Running instance of an image.

**4. Common Docker commands:**

docker build -t app .

docker run -d -p 80:80 app

docker ps

docker stop <container\_id>

**5. How to deploy an app using Docker?**

* Write a Dockerfile
* Build the image using docker build
* Run a container using docker run

**✅ Module 6: Ansible for Configuration Management**

**1. What is Ansible?**

Ansible is an open-source automation tool for configuration management, application deployment, and task automation.

**2. What is an Ansible Playbook?**

A YAML file defining tasks to be executed on hosts.

**3. Push vs Pull models:**

* Push: Master pushes configuration to nodes (used by Ansible)
* Pull: Nodes pull configuration from master

**4. Difference between ad-hoc commands and playbooks:**

* Ad-hoc: One-time tasks
* Playbook: Reusable and structured automation scripts

**5. What is YAML in Ansible?**

YAML is a human-readable data format used to write playbooks. It stands for “YAML Ain’t Markup Language”.