DevOps Viva QUE answers

Module 1: DevOps Introduction

1. What is DevOps?

Answer: DevOps is a set of practices that combines software development and IT operations. It aims to shorten the development cycle and provide continuous delivery with high software quality.

2. What are the key principles of DevOps?

Answer:

- Continuous Integration
- Continuous Delivery
- Infrastructure as Code
- Monitoring and Logging
- Collaboration

3. What are the benefits of implementing DevOps?

Answer:

- Faster development and deployment
- Improved collaboration
- Increased efficiency through automation
- Stable and secure environments

4. What is Git?

Answer: Git is a distributed version control system that helps track changes in source code during software development.

5. What is SDLC?

Answer: SDLC stands for Software Development Life Cycle. It's a process used to design, develop, and test software.

6. What is Agile programming?

Answer: Agile is a methodology that promotes continuous iteration of development and testing throughout the software development lifecycle.

◆ Module 2: Development with Git & GitHub

1. What is GitHub?

Answer: GitHub is a cloud-based platform that uses Git for version control and helps manage code repositories.

2. What is the use of `git init`?

Answer: It initializes a new Git repository in your project directory.

3. What is the use of `git clone`?

Answer: It copies an existing Git repository from a remote server (e.g., GitHub) to your local machine.

4. Explain `git add`, `commit`, `push`, and `pull`.

Answer:

- `git add`: Stages changes
- `git commit`: Saves changes locally
- `git push`: Uploads changes to remote
- `git pull`: Fetches and merges changes from remote

5. What is a fork in GitHub?

Answer: A fork is a copy of a repository. It allows you to freely experiment without affecting the original project.

6. What is a pull request?

Answer: A pull request lets you notify the project owner that you'd like to merge your changes into their repository.

◆ Module 3: GitLab Essentials

1. What is GitLab?

Answer: GitLab is a DevOps platform providing Git repository management, CI/CD, and security features.

2. How is GitLab different from GitHub?

Answer: GitLab provides built-in CI/CD features, while GitHub requires third-party tools like Jenkins for CI/CD.

3. What is CI/CD?

Answer: CI (Continuous Integration) involves automatically testing code, and CD (Continuous Deployment) automates the release of that code to production.

4. How do you interact with GitLab using Git commands?

Answer: The same Git commands (`clone`, `push`, `pull`, etc.) are used; you just provide the GitLab repo URL.

5. Mention one security concern while using GitLab.

Answer: Managing access control and securing CI pipelines to avoid unauthorized code execution.

◆ Module 4: Jenkins – Continuous Integration

1. What is Jenkins?

Answer: Jenkins is an open-source automation server that helps automate the building, testing, and deployment of software.

2. How do you install Jenkins?

Answer: Jenkins can be installed via `.war` file, Docker container, or native packages depending on the OS.

3. What is a Jenkins Pipeline?

Answer: A Jenkins Pipeline defines the steps to build, test, and deploy applications as code.

4. How do you integrate Jenkins with GitHub?

Answer: By installing GitHub plugin in Jenkins and adding the repo URL and credentials in Jenkins configuration.

5. What is a use case of Jenkins in large-scale environments?

Answer: Automating the build and test cycle for microservices architecture with multiple repositories.

◆ Module 5: Continuous Deployment – Docker

1. What is Docker?

Answer: Docker is a containerization platform that packages applications and dependencies together.

2. What is a Docker container?

Answer: A lightweight, standalone, executable package that includes everything needed to run an application.

3. What is the difference between a Docker image and container?

Answer:

- Image: Blueprint of the container
- Container: Running instance of an image

4. Mention Docker commands and their purpose:

- `docker build`: Create an image
- `docker run`: Run a container
- `docker images`: List images
- `docker ps`: List running containers

5. How do you secure Docker containers?

Answer:

- Use trusted base images
- Minimize image layers
- Avoid running as root
- Use Docker secrets

◆ Module 6: Configuration Management – Ansible

1. What is Configuration Management?

Answer: It is the process of maintaining systems, servers, and software in a desired, consistent state.

2. What is Ansible?

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

3. What is an Ansible Playbook?

Answer: A YAML file that defines a set of tasks to be executed on target machines.

4. What are Push and Pull models in Ansible?

Answer:

- **Push:** Ansible pushes configurations from a control node.
- **Pull:** Clients pull configurations (less common in Ansible).

5. What is the best practice for writing Ansible playbooks?

Answer:

- Use roles and variables
- Keep tasks modular
- Avoid hardcoding values
- Use version control

Module 1: DevOps Introduction & Basic Git Setup

Q1: How do you set up Git for the first time on your system?

A: git config --global user.name "Your Name"

git config --global user.email "you@example.com"

Q2: How do you initialize a Git repository?

A: git init

Q3: How can you check the current Git configuration?

A: git config –list

Module 2: Git & GitHub

Q1: How do you clone a remote GitHub repository?

A: git clone https://github.com/username/repo-name.git

Q2: How do you add and commit changes in Git?

A: git add .

git commit -m "Your message"

Q3: How do you push local changes to GitHub?

A: git push origin main

Q4: What is a pull request?

A: A pull request lets you tell others about changes you've pushed to a GitHub repository.

Q5: How do you fork a repository on GitHub?

A: Click the Fork button on the top-right of a repository on GitHub

Module 3: GitLab & CI/CD Basics

Q1: How do you create a new project in GitLab?

A: Log in -> Click on "New Project" -> Choose template or blank project -> Name it and create.

Q2: How do you use GitLab Web IDE?

A: Open your project -> Click on Web IDE -> Edit files -> Commit changes.

Q3: What file is used for defining CI/CD pipelines in GitLab?

A: .gitlab-ci.yml

Q4: Mention one advantage of GitLab over GitHub in CI/CD.

A: GitLab provides built-in CI/CD without requiring external tools

Module 4: Jenkins CI

Q1: How do you install Jenkins?

A: java -jar jenkins.war

Q2: How do you create a job in Jenkins?

A: Click New Item -> Name it -> Select Freestyle Project -> Configure Git repo, build steps.

Q3: How do you integrate Jenkins with GitHub?

A: Install GitHub plugin, use Webhooks, provide repo URL and credentials.

Q4: What is a Jenkins pipeline?

A: A scripted or declarative process in a Jenkinsfile to automate build, test, and deploy.

Module 5: Docker & Containerization

Q1: How do you check Docker installation?

A: docker --version

Q2: How do you create and run a Docker container?

A: docker run -it ubuntu

Q3: What command builds a Docker image?

A: docker build -t myapp .

Q4: How do you list running containers?

A: docker ps

Q5: How do you stop and remove a container?

A: docker stop <container_id>

docker rm <container_id>

Module 6: Ansible Configuration Management

Q1: What is an Ansible Playbook?

A: A YAML file containing automation tasks.

Q2: How do you run a playbook?

A: ansible-playbook playbook.yml

Q3: What is an inventory file in Ansible?

A: Contains list of target machines.

Q4: What language is used to write Ansible playbooks?

A: YAML (Yet Another Markup Language)

1. What is DevOps, and why is it important?

Answer:

DevOps is a **set of practices** that combines **software development (Dev)** and **IT operations (Ops)** to shorten the **software development lifecycle (SDLC)** while ensuring **high quality and reliability**.

2. How does DevOps differ from traditional IT operations?

Answer:

Aspect	Traditional IT Operations	DevOps
Development & Operations	Separate teams	Integrated teams
Deployment Frequency	Weeks/Months	Daily/Weekly
Automation	Limited	Extensive (CI/CD, IaC)
Collaboration	Siloed	Cross-functional
Feedback Loop	Slow	Fast (Continuous Monitoring)

3. What are the key principles of DevOps?

Answer:

1. **Collaboration** – Breaking silos between Dev & Ops
2. **Automation** – CI/CD, Infrastructure as Code (IaC)
3. **Continuous Integration & Continuous Deployment (CI/CD)**
4. **Monitoring & Logging** – Observability, real-time feedback
5. **Security (DevSecOps)** – Security integrated into SDLC

4. Explain the DevOps lifecycle.

Answer:

1. **Plan** – Jira, Trello
2. **Develop** – Git, GitHub
3. **Build** – Maven, Gradle
4. **Test** – Selenium, JUnit
5. **Release** – GitHub Actions, Jenkins
6. **Deploy** – Kubernetes, Docker
7. **Monitor** – Prometheus, Grafana

5. What are some common DevOps tools?

Answer:

- **CI/CD**: Jenkins, GitHub Actions, GitLab CI
- **Configuration Management**: Ansible, Puppet
- **Containerization**: Docker, Kubernetes
- **Monitoring & Logging**: Prometheus, Grafana, ELK Stack

6. What is CI/CD, and how does it work?

Answer:

CI/CD is a DevOps practice that automates code integration, testing, and deployment.

- **Continuous Integration (CI)** – Automates code merging & testing.
- **Continuous Deployment (CD)** – Automates production releases.

7. Explain the difference between Continuous Deployment and Continuous Delivery.

Answer:

Aspect	Continuous Delivery	Continuous Deployment
Automation	Deployments require manual approval	Fully automated deployments
Risk	Lower risk, manual control	Higher automation, requires testing reliability

8. What is version control, and why is Git used in DevOps?

Answer:

Version control tracks code changes, allowing collaboration. Git is widely used because of:

- **Branching & Merging** – Parallel development
- **Distributed Version Control** – No central dependency

9. What is Infrastructure as Code (IaC)?

Answer:

IaC automates infrastructure provisioning using code. Example: **Terraform, Ansible, CloudFormation**.

10. How does a DevOps engineer handle configuration management?

Answer:

Using **Ansible, Puppet, Chef**, engineers automate configuration setup, ensuring consistency.

11. What is a container, and how does Docker help DevOps?

Answer:

A container packages an app with dependencies, ensuring it runs identically anywhere. Docker simplifies container management.

1. What is DevOps?

DevOps is a software development approach that combines Development (Dev) and IT Operations (Ops) to automate and streamline the software development, testing, deployment, and maintenance process. It focuses on collaboration, automation, and continuous improvement, allowing businesses to deliver software faster, more efficiently, and with fewer errors. DevOps integrates Continuous Integration/Continuous Deployment (CI/CD), Infrastructure as Code (IaC), monitoring, and automation to ensure that software is built, tested, and released seamlessly.

2. What is a DevOps Engineer?

A **DevOps Engineer** is a professional who combines software development (Dev) and IT operations (Ops) skills to improve and streamline the process of developing, testing, and releasing software.

Their goal is to ensure that software is delivered quickly, efficiently, and reliably. They work to automate and integrate the processes between software development and IT teams, allowing for continuous delivery and continuous integration of software.

3. What is the use of SSH?

SSH(Secure Shell) is an access credential used in the SSH Protocol. In other words, it is a cryptographic network protocol that transfers encrypted data over the network. It allows you to connect to a server, or multiple servers, without having to remember or enter your password for each system that is to log in remotely from one system to another.

4. What is CI/CD?

CI And CD is the practice of automating the integration of code changes from multiple developers into a single codebase. It is a software development practice where the developers commit their work frequently to the central code repository (Github or Stash).

- **Continuous Integration**: With Continuous Integration, developers frequently commit to a shared common repository using a version control system such as Git. A continuous integration pipeline can automatically run builds, store the artifacts, run unit tests, and even conduct code reviews using tools like Sonar.
- **Continuous Delivery**: Continuous delivery helps developers test their code in a production-similar environment, hence preventing any last-moment or post-production surprises. These tests may include UI testing, load testing, integration testing, etc. It helps developers discover and resolve bugs preemptively.

What's the difference between DevOps & Agile?

Agile	DevOps
Agile is a method for creating software.	It is not related to software development. Instead, the software that is used by DevOps is pre-built, dependable, and simple to deploy.
An advancement and administration approach.	Typically a conclusion of administration related to designing.
The agile handle centers on consistent changes.	DevOps centers on steady testing and conveyance.
Agile relates generally to the way advancement is carried out, any division of the company can be spry on its hones. This may be accomplished through preparation.	DevOps centers more on program arrangement choosing the foremost dependable and most secure course.

8. What is the continuous testing process?

Continuous testing is a process of automated testing done on software continuously as soon as a piece of code is delivered by the developers. This testing is done at every stage starting from the initial stages of development until the deployment of software.

What do you mean by Configuration Management?

The process of controlling and documenting change for the development system is called Configuration Management. Configuration Management is part of the overall change management approach. It allows large teams to work together in s stable environment while still providing the flexibility required for creative work.

Explain the concept of branching in Git.

Branching means diverging from the mainline and continuing to work separately without messing with the mainline. Nearly every VCS has some form of branch support. In Git, a branch is simply a reference to the commit, where the following commits will be attached.

What is a GIT Repository?

Repositories in GIT contain a collection of files of various versions of a Project. These files are imported from the repository into the local server of the user for further updations and modifications in the content of the file. A VCS or the Version Control System is used to create these versions and store them in a specific place termed a repository.

16. What Is Jenkins?

Jenkins is a tool that is used for automation, and it is an open-source server that allows all the developers to build, test and deploy software. It works or runs on java as it is written in java. By using Jenkins we can make a continuous integration of projects(jobs) or end-to-endpoint automation.

22. What is Ansible?

Ansible is an open-source IT engine that automates application deployment, cloud provisioning, intra-service orchestration, and other IT tools. Ansible can be used to deploy the software on different servers at a time without human interaction. Ansible can also be used to configure the servers and create user accounts.

Ansible is an agent-less software which means there is no need to install the software in the nodes which means you need to do the SSH to connect the nodes to perform the required operations on the servers.

25. What is Git Bash?

Git Bash is a command-line interface (CLI) application for Windows that lets you communicate with Git, the version control system. Clone the repositories, commit changes, push and pull changes, and more are all possible using Git Bash. Git Bash can automate manual tasks with the scripts written by you. Git Bash helps you in a greater way to learn about Git and version control.

How to Make a CI-CD Pipeline in Jenkins?

DevOps professionals mostly work with pipelines because pipelines can automate processes like building, testing, and deploying the application. With the help of Continuous Integration / Continuous Deployment (CI/CD) Pipeline scripts we can automate the whole process which will increase productivity save lots of time for the organization and deliver quality applications to the end users.

39. Explain how you can set up a Jenkins job?

To set up a Jenkins job:

1. Open Jenkins and log in with your credentials.
2. Click "New Item" from the dashboard.
3. Enter a name for your job and select the job type (e.g., Freestyle project).
4. Click "OK" to create the job.
5. Configure your job by adding a description, source code management details (e.g., Git repository), and build triggers.
6. Add build steps, such as shell commands or invoking scripts.
7. Save the job and click "Build Now" to run it.

40. Explain the architecture of Docker.

Docker architecture consists of several key components:

1. **Docker Client**: Issues commands to the Docker daemon via a command-line interface (CLI).

2. **Docker Daemon (dockerd)**: Runs on the host machine, managing Docker objects like images, containers, networks, and volumes.
3. **Docker Images**: Read-only templates used to create Docker containers.
4. **Docker Containers**: Lightweight, portable, and executable instances created from Docker images.
5. **Docker Registry**: Stores and distributes Docker images; Docker Hub is a popular public registry.
6. **Docker Compose**: A tool for defining and running multi-container Docker applications using a YAML file.
7. **Docker Networking**: Allows containers to communicate with each other and with non-Docker environments.

41. What is the DevOps life cycle?

DevOps Lifecycle is the set of phases that includes DevOps for taking part in Development and Operation group duties for quicker software program delivery. DevOps follows positive techniques that consist of code, building, testing, releasing, deploying, operating, displaying, and planning. DevOps lifecycle follows a range of phases such as non-stop development, non-stop integration, non-stop testing, non-stop monitoring, and non-stop feedback. 7 Cs of DevOps are:

- Continuous Development
- Continuous Integration
- Continuous Testing
- Continuous Deployment/Continuous Delivery
- Continuous Monitoring
- Continuous Feedback
- Continuous Operations

42. What is the difference between Git Merge and Git Rebase?

Git Merge	Git Rebase
Git Merge merges two branches to create a “feature” branch.	Git Rebase rebases the feature branch to add the feature branch to the main branch.
Git Merge is comparatively easy.	Git Rebase is comparatively harder.
Git Merge safeguards history.	Git Rabse doesn’t safeguard history.
Git Merge is more suitable for projects with the less active main branch.	Git Rebase is suitable for projects with frequently active main branches.