1. What is Git and why is it used?

Git is a distributed version control system (DVCS) used to track changes in source code during software development. It allows multiple developers to collaborate, manage code versions, and maintain a history of changes. Git was created by Linus Torvalds in 2005

* Tracking changes in source code.
* Collaborating with teams.
* Managing different versions of software.
* Branching and merging code.

1. Explain the difference between Git and other version control systems.

* Git is a distributed version control system (DVCS)
* Centralized version control system (CVCS) like SVN. It is the oldest version of GIT.
* It consists of only the current version of data but it does not consist any previous version data.

1. How do you initialize a Git repository?

* By using git init command to initialize a git repo

1. What is the purpose of the .gitignore file?

* The .gitignore **file** is used to **exclude specific files and directories** from being tracked by Git. This helps keep the repository clean by ignoring unnecessary, sensitive, or system-generated files.

1. How do you stage changes in Git?

* By using git add <file name> or git add .

1. What is the difference between git commit and git commit -m?

* git commit opens an editor to write a commit message.
* git commit -m "message" allows adding a commit message directly from the command line.

1. How do you create a new branch in Git?

* By using git cheackout -b < branch name>

1. What is the difference between git merge and git rebase?

* Both git merge and git rebase are used to combine changes from one branch into another, but they work differently.

git merge: Combines changes from one branch into another by creating a **new merge commit**. This preserves the commit history.

--> git merge feature-branch

git rebase: Moves or replays commits from one branch onto another, **rewriting history** for a cleaner linear commit structure.

--> git rebase main.

1. How do you resolve merge conflicts in Git?

* Identify conflicts using git status, manually edit conflicting files, then use git add <file> followed by git commit.

1. What is the purpose of git stash?

* git stash temporarily saves uncommitted changes, allowing you to switch branches without losing work.

1. Explain the use of git pull and git fetch.

* git fetch downloads changes from a remote repository but does not merge them into your current branch.
* git pull downloads changes and automatically merges them into your current branch. It is essentially git fetch followed by git merge.

1. How do you revert a commit in Git?

* Use git revert <commit\_hash> to create a new commit that undoes changes from a specific commit.

1. What is the difference between git clone and git fork?

* git clone is a copies a repository, while git fork creates a new repository under a different owner, allowing independent development.

1. Explain the concept of remote repositories in Git.

* Remote repositories are versions of your project hosted on a server (e.g., GitHub, GitLab, Bitbucket). They allow collaboration and backups.

1. What are Git tags and how do you use them?

* Tags are used to mark specific points in history, often for releases. Use git tag -a v1.0 -m "Version 1.0" to create an annotated tag.

1. How do you view the commit history in Git?

* By using git log to see commit history.

1. What is the purpose of git diff?

* It shows changes between commits, branches, or working directories.

1. How do you delete a branch in Git?

* Use git branch -d <branch\_name> to delete a local branch and git push origin --delete <branch\_name> to delete a remote branch.

1. What are Git hooks and how are they used?

* Git hooks are scripts that execute before or after Git events (e.g., pre-commit, post-merge) to automate tasks.

1. Explain the concept of a pull request in Git.

* A pull request (PR) is a request to merge code changes into a remote repository, typically used in collaborative development.

1. What is DevOps and why is it important?

* Devops is culture which varies from organization to organization (or) it is the process of delivery by ensuring by the automation(CI/CD) in place, ensuring the quality by continuous monitoring & testing.

1. To deliver the s/w or project or product on time.

2. To deliver the product with high quality(accurately).

3. To deliver by automation -CI/CD tools.

4. To deliver by continuous monitoring & testing.

5. In order to upgrade its version.

6. Enhances communication between development and operations teams.

1. Explain the key principles of DevOps.

Collaboration and communication: Breaking down silos between development and operations.

1. **Automation:** Automating repetitive tasks.

2. **Continuous integration and continuous delivery (CI/CD):** Automating the build, test, and deployment processes.

3. Infrastructure as Code (IaC): Managing infrastructure through code.

4. **Monitoring and logging:** Tracking system performance and identifying issues.

1. What are the benefits of continuous integration (CI)?

* Early detection of bugs.
* Faster feedback.
* Reduced integration problems.
* Improved code quality.

1. What is continuous delivery (CD) and how does it differ from continuous deployment?

* **Continuous delivery:** Automates the release process, ensuring that code is always ready to be deployed to production.
* Continuous deployment: Automatically deploys every change that passes the automated tests to production.
* The key difference is that continuous delivery requires manual approval for production deployments, while continuous deployment is fully automated.

1. Explain the concept of Infrastructure as Code (IaC).

* IaC allows managing infrastructure using code, ensuring consistency and automation (e.g., Terraform, Ansible).

1. What tools are commonly used in a DevOps pipeline?

* **Version control:** Git
* CI/CD: Jenkins, GitLab CI, CircleCI, GitHub Actions.
* Configuration management: Ansible, Chef, Puppet.
* **Containerization:** Docker, Kubernetes.
* **Infrastructure as Code:** Terraform, CloudFormation.
* **Monitoring:** Prometheus, Grafana

1. What is the role of configuration management in DevOps?

* Configuration management ensures that systems are consistently configured and maintained. It automates the process of setting up and managing servers and applications.

1. How does containerization help in a DevOps environment?

* Containers (like Docker) provide consistent environments across different stages of the development lifecycle.
* They make applications portable and scalable.
* They simplify deployment and management.

1. What is the purpose of monitoring in DevOps?

* Monitoring ensures system health, detects issues early, and helps optimize performance.

1. What are microservices and how do they relate to DevOps?

* Microservices are small, independent services that improve scalability and maintainability, making DevOps practices more effective.

1. Explain in detail about devops tools?

**Version Control**: Git

**Build Automation**: Maven, Gradle

**CI/CD**: Jenkins, GitLab CI/CD

**Containerization**: Docker

**Orchestration**: Kubernetes

**Monitoring**: Prometheus, Grafana

**Infrastructure as Code**: Terraform, Ansible