**DevOps & Git Interview Q&A**

### 1. What does git add do and why is it important before committing?

* git add moves changes from the working directory to the staging area (index).
* Importance:
  + Lets you decide which changes to include in a commit.
  + Prevents accidental commits of unwanted files.

### 2. How do you stage only specific changes in Git?

* Stage a specific file:
* git add file1.yaml
* Stage multiple specific files:
* git add file1.yaml file2.yaml
* Stage certain lines interactively:
* git add -p file.yaml

### 3. If you have to update 2 YAML files in a remote repo as a new joiner, how will you do it?

Steps:

1. Clone the repository:

* git clone <repo-url>  
  cd <repo-name>

1. Create a new branch:

* git checkout -b update-yaml-files

1. Edit the YAML files locally.
2. Stage only the updated files:

* git add file1.yaml file2.yaml

1. Commit with a meaningful message:

* git commit -m "Updated configuration in file1.yaml and file2.yaml"

1. Push your branch to remote:

* git push origin update-yaml-files

1. Raise a Pull Request (PR) for review.

### 4. What is Git stash & when do you use it?

* git stash temporarily saves uncommitted changes and cleans your working directory.
* Use cases:
  + Switch branches quickly without committing.
  + Pull latest changes without losing local progress.
* Commands:
* git stash # save changes  
  git stash list # view stashes  
  git stash apply # reapply last stash  
  git stash pop # reapply & remove from stash

### 5. Difference between git fetch & git pull

* git fetch: Downloads new commits from remote **without merging**.
* git pull: Fetches + merges changes into your local branch.
* Formula: git pull = git fetch + git merge

### 6. What is git cherry-pick & when would you use it?

* git cherry-pick <commit-hash> applies a specific commit from one branch to another.
* Use cases:
  + Apply a bug fix from a feature branch to main.
  + Copy urgent hotfixes to another branch.

Example:

git checkout main  
git cherry-pick a1b2c3d

### 7. Explain your experience in CI/CD tools and how you used it.

* Worked with **Jenkins, GitHub Actions, GitLab CI/CD**.
* Built pipelines for automated build, test, and deployment.
* Integrated **Docker, Kubernetes, SonarQube, and security scans** into pipelines.
* Automated deployment to AWS EKS using Helm charts.

### 8. How much experience you have in writing pipeline scripts?

* Around **X years** of experience writing pipeline scripts using **Groovy (Jenkinsfile), YAML (GitHub Actions/GitLab)**.
* Experience in implementing build, test, security scanning, and deployment stages.

### 9. Write a pipeline script implementing some tools in Groovy (Jenkinsfile)

pipeline {  
 agent any  
  
 stages {  
 stage('Checkout') {  
 steps {  
 git branch: 'main', url: 'https://github.com/example/repo.git'  
 }  
 }  
  
 stage('Build') {  
 steps {  
 sh 'mvn clean install'  
 }  
 }  
  
 stage('Docker Build & Push') {  
 steps {  
 sh 'docker build -t myapp:latest .'  
 sh 'docker tag myapp:latest myrepo/myapp:latest'  
 sh 'docker push myrepo/myapp:latest'  
 }  
 }  
  
 stage('Deploy to Kubernetes') {  
 steps {  
 sh 'kubectl apply -f k8s/deployment.yaml'  
 }  
 }  
 }  
}

### 10. Have you created pipeline script end-to-end? What kind of tools did you use?

* Yes, developed **end-to-end CI/CD pipelines**.
* Tools used:
  + **Jenkins/GitHub Actions** for orchestration.
  + **Maven/Gradle** for build.
  + **SonarQube** for code quality.
  + **Docker & Kubernetes** for containerization and deployment.
  + **Helm** for managing Kubernetes manifests.

### 11. How will you create GitHub Actions?

* Create a .github/workflows/ci.yml file in the repository.
* Example:

name: CI Pipeline  
on: [push, pull\_request]  
  
jobs:  
 build:  
 runs-on: ubuntu-latest  
 steps:  
 - name: Checkout code  
 uses: actions/checkout@v3  
  
 - name: Set up JDK  
 uses: actions/setup-java@v3  
 with:  
 java-version: '17'  
  
 - name: Build with Maven  
 run: mvn clean install

### 12. Did you give any ideas to your team or project in DevOps to improve deployments? If yes, what were they?

* Introduced **GitHub Actions** to replace manual Jenkins jobs for smaller projects.
* Suggested **blue-green deployment strategy** to reduce downtime.
* Implemented **Helm charts** for reusable Kubernetes manifests.
* Proposed **SonarQube & Snyk integration** for early security and quality checks.

### 13. How much will you rate yourself in Linux & Python?

* Linux: **8/10** (proficient with commands, scripting, troubleshooting).
* Python: **7/10** (automation scripts, API calls, CI/CD utilities).

### 14. How far are you good in using AI in DevOps? (GitHub Copilot)

* Used **GitHub Copilot** for generating CI/CD snippets, Helm templates, and Python automation scripts.
* Helps in faster prototyping and reducing repetitive coding.

### 15. How about Monitoring tools? What have you used in your organization?

* Worked with **Prometheus, Grafana, ELK Stack, New Relic**.
* Built custom Grafana dashboards for Kubernetes cluster metrics.
* Configured Prometheus alerts for pod resource usage.

### 16. How are alerts managed in Prometheus or Grafana?

* **Prometheus:** Uses Alertmanager to define rules (e.g., CPU > 80%). Alerts sent to Slack/Email.
* **Grafana:** Configured alert rules on dashboards. Alerts integrated with PagerDuty & MS Teams.