1. Branching Strategies in Git – Feature Branches, Release branches.  
Feature branching – A feature branch model keeps all of the changes for a particular feature inside of a branch. When the feature is fully tested and validated by automated tests, the branch is then merged into master.  
Release branching – Once the develop branch has acquired enough features for a release, you can clone that branch to form a Release branch. Creating this branch starts the next release cycle, so no new features can be added after this point, only bug fixes, documentation generation, and other release-oriented tasks should go in this branch. Once it is ready to ship, the release gets merged into master and tagged with a version number. In addition, it should be merged back into the develop branch, which may have progressed since the release was initiated.

2. Basic git commands used regularly – syncing Repos, Fetch, Pull, Push, Clone, Merging, Rebasing, Stashing  
syncing Repos:  
Git pull: fetch and merge any commits from the tracking remote branch  
command pulls new changes or commits from a particular branch from your central repository and updates your target branch in your local repository.  
Git fetch: fetch down all the branches from that Git remote  
is also used for the same purpose but it works in a slightly different way. When you perform a git fetch, it pulls all new commits from the desired branch and stores it in a new branch in your local repository. If you want to reflect these changes in your target branch, git fetch must be followed with a git merge. Your target branch will only be updated after merging the target branch and fetched branch. Just to make it easy for you, remember the equation below:  
Git pull = git fetch + git merge  
GIT STASH : when you’ve been working on part of your project, things are in a messy state and you want to switch branches for some time to work on something else. The problem is, you don’t want to do a commit of half-done work just so you can get back to this point later. The answer to this issue is Git stash.  
Stashing takes your working directory that is, your modified tracked files and staged changes and saves it on a stack of unfinished changes that you can reapply at any time.  
‘git clone’ retrieve an entire repository from a hosted location via URL  
git push: Transmit local branch commits to the remote repository branch  
git Merge:merge a remote branch into your current branch to bring it up to date  
git rebase:apply any commits of current branch ahead of specified one  
rebase command is used to integrate changes from one branch into another. It is an alternative to the “merge” command

Git rebase :

Git merge :

3. When to do Git Rebase?  
I want the point at which I branched to move to a new starting point  
if you started doing some development and then another developer made an unrelated change. You probably want to pull and then rebase to base your changes from the current version from the repository.

4. When do you use Git stashing?  
If you want to continue working where you had left your work then ‘git stash apply‘ command is used to bring back the saved changes onto your current working directory.  
5. Jenkins – What is pipeline as a code?

A pipeline as code file specifies the stages, jobs, and actions for a pipeline to perform  
6. Jenkins- How do you run multiple tasks in parallel in a Jenkins file

**Pipelining** is an implementation technique where multiple instructions are overlapped in execution.

7. Write a sample Jenkins file with maven Build(stage1) , Sonar analysis and deployment ( stages can run in parallel)

pipeline {  
agent any  
stages{  
stage('SCM Checkout') {  
steps {  
git '[https://github.com/Gkasiraju/mvn\_sonar.git'](https://github.com/Gkasiraju/mvn_sonar.git%27)  
}  
}  
stage ('Build') {  
steps {  
sh '/opt/maven/bin/mvn clean sonar:sonar -Dmaven.test.skip=true'  
}  
}  
stage('run-parallel-branches') {  
  
parallel{  
stage ('SonarQube analysis') {  
steps {  
withSonarQubeEnv('sonar'){  
sh '/opt/maven/bin/mvn clean package sonar:sonar -Dsonar.password=admin123 -Dsonar.login=admin -Dmaven.test.skip=true'  
  
}  
}  
}  
stage ('Artifact-Deployment') {  
steps {  
sh '/opt/maven/bin/mvn clean deploy -Dmaven.test.skip=true'  
}  
}  
}  
}  
stage ('Release') {  
steps {  
sh 'export JENKINS\_NODE\_COOKIE=dontkillme ;nohup java -jar $WORKSPACE/target/\*.jar &'  
}  
}  
}  
}

8. What is a declarative pipeline?

**Declarative pipeline** is a relatively new feature that supports the **pipeline** as code concept. It makes the **pipeline** code easier to read and write. This code is written in a Jenkinsfile which can be checked into a source control management system such as Git

9. What plugins have you worked on so far?

Sonarqube scanner, sonarquality gates

10. How to automatically migrate/upgrade Jenkins ( How plugins can be exported to new instance)

**plug-in** is a software component that adds a specific feature to an existing computer program

11. Sonarqube Analysis, Quality gate settings

12. How Jenkins identifies new commits in Git and triggers pipeline?

By using webhooka and poolscm

13. Webhooks and Poll SCM

Webhook----it means whenever developer commits the code then webhook is trigger to Jenkins.

Poll SCM—it means doring configuration of our job jenkins builds the periodically ..

14. Ansible – configuration Management – What is an Ansible Role and Ansible Module? Differences

In Ansible, the role is the primary mechanism for breaking a playbook into multiple files. This simplifies writing **complex playbooks**, and it makes them easier to reuse. The breaking of playbook allows you to logically break the playbook into reusable components.

15. What all modules have you got hands-on while writing playbooks

Service, ping command

ansible webservers -m service -a "name=httpd state=started"

ansible webservers -m ping

ansible webservers -m command -a "/sbin/reboot -t now"

16. How Ansible handles replay, when certain modules , tasks are not executed?

By default Ansible stops executing tasks on a host when a task fails on that host. You can use ignore\_errors to continue on in spite of the failure:

 [Ignoring failed commands](https://docs.ansible.com/ansible/latest/user_guide/playbooks_error_handling.html#ignoring-failed-commands)

 **-** name**:** Do not count this as a failure

 ansible.builtin.command**:** /bin/false

 ignore\_errors**:** yes

  [Ignoring unreachable host errors](https://docs.ansible.com/ansible/latest/user_guide/playbooks_error_handling.html#ignoring-unreachable-host-errors)

 **-** name**:** This executes, fails, and the failure is ignored

 ansible.builtin.command**:** /bin/true

 ignore\_unreachable**:** yes

  **-** name**:** This executes, fails, and ends the play for this host

 ansible.builtin.command**:** /bin/true

  [Resetting unreachable hosts](https://docs.ansible.com/ansible/latest/user_guide/playbooks_error_handling.html#resetting-unreachable-hosts)

 [Handlers and failure](https://docs.ansible.com/ansible/latest/user_guide/playbooks_error_handling.html#handlers-and-failure)

 [Defining failure](https://docs.ansible.com/ansible/latest/user_guide/playbooks_error_handling.html#defining-failure)

 **-** name**:** Fail task when the command error output prints FAILED

 ansible.builtin.command**:** /usr/bin/example-command -x -y -z

 register**:** command\_result

 failed\_when**:** "'FAILED'incommand\_result.stderr"

17. Docker – How do you Build Images

18. Kubernetes – How Kubernetes orchestrates a Docker container?

19. What are the seven layers of the OSI model?

A) physical layer

data link layer

network layer

transport

session layer

presentations layer

application

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Employee Code** | **Employee Name** | **Base Location** | **Current Location** | **Mode 1 Trained on?** | **Mode 2 Trained on?** | **Other skills** | **Certification ?** | **HCL DOJ** |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Employee Code** | **Employee Name** | **Base Location** | **Current Location** | **Mode 1 Trained on?** | **Mode 2 Trained on?** | **Other skills** | **Certification ?** | **HCL DOJ** |
| 51894749 | Madem Shyam Sai Satish | Bangalore | Visakhapatnam | Core Java, HTML ,CSS, JS,SPRING | DevOps, Tools - Jenkins, Git, Maven, Sonarqube, Nexus, Slack, ansible basics | C, Python(Basics) | Udemy, Percipo certificates | 8th Oct-2020 |
| 51894687 | Gudla kasiraju | Bangalore | Srikakulam AP | Core Java, HTML ,CSS, JS,SPRING | DevOps, Tools - Jenkins, Git, Maven, Sonarqube, Nexus, Slack, ansible basics | C, Python, Right now I am going with chef and puppet, Docker, courses given by TEX ACADEMY | Udemy, Percipo certificates | 8th Oct-2020 |
| 51894835 | Sasapu Teja | Bangalore | Srikakulam | Core Java, HTML ,CSS, JS,SPRING | DevOps, Tools - Jenkins, Git, Maven, Sonarqube, Nexus, Slack, ansible basics | C and python  basics | Udemy, Percipo certificates | 8th Oct-2020 |
|  |  | Bangalore |  | Core Java, HTML ,CSS, JS,SPRING | DevOps, Tools - Jenkins, Git, Maven, Sonarqube, Nexus, Slack, ansible basics |  | Udemy, Percipo certificates | 8th Oct-2020 |
|  |  | Bangalore |  | Core Java, HTML ,CSS, JS,SPRING | DevOps, Tools - Jenkins, Git, Maven, Sonarqube, Nexus, Slack, ansible basics |  | Udemy, Percipo certificates | 8th Oct-2020 |
|  |  | Bangalore |  | Core Java, HTML ,CSS, JS,SPRING | DevOps, Tools - Jenkins, Git, Maven, Sonarqube, Nexus, Slack, ansible basics |  | Udemy, Percipo certificates | 8th Oct-2020 |
|  |  | Bangalore |  | Core Java, HTML ,CSS, JS,SPRING | DevOps, Tools - Jenkins, Git, Maven, Sonarqube, Nexus, Slack, ansible basics |  | Udemy, Percipo certificates | 8th Oct-2020 |
|  |  | Bangalore |  | Core Java, HTML ,CSS, JS,SPRING | DevOps, Tools - Jenkins, Git, Maven, Sonarqube, Nexus, Slack, ansible basics |  | Udemy, Percipo certificates | 8th Oct-2020 |
|  |  | Bangalore |  | Core Java, HTML ,CSS, JS,SPRING | DevOps, Tools - Jenkins, Git, Maven, Sonarqube, Nexus, Slack, ansible basics |  | Udemy, Percipo certificates | 8th Oct-2020 |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |