CICD

Yeah, coming to the CI CD pipeline, we are using a CI server as a Jenkins.Before setting up the CI CD pipeline what I'll be doing right, we have to configure a couple of servers for the CI CD process. So I used to configure the Jenkins server and sonar cube for the quality code analysis. And we are using a Nexus for storing all the artifacts. And also we have AWS ECR. So that is for to storing all that Docker images. So initially,I will be doing all the setup like creating the Jenkins server Nexus sonar and creating some repos in ECR, AWS ECR. And also, I'll be creating a Tomcat server for the deployment. If it is kind of if I'm doing the direct deployment on top top of Tomcat web server on the sense, we need a Tomcat as well as so if it is a containerized based technology deployment, I'll be going with a docker as well as. So that is one kind of thing. So initially, I will be configuring all the servers,then I will be doing the integration activity, like I'll be configuring a Jenkins with GitHub. So we can go with a couple of triggering mechanism. So the first thing is like we can go with a web book. Next one is Poll SCM. We can also configure a build trigger. So these are the trigger mechanism, if can able to configure with the Jenkins

with the git, then we can integrate a Jenkins with a Maven. So Maven is kind of a build tool where I'm getting the war files. So all the war files will be moved to the Jenkins a target directory. So that is one thing I can say. So once that WAR file is

built, before that, I need to integrate my Jenkins with a sonar cube . So for that integration, I need to attain one plugin that is called a sonar scanner, I have to install that sonar scanner plugin. Then I will be making the handshake between the Jenkins and Sonar. So in the sonar cube dashboard, I need to generate one token actually. So that token, I need to

configure in a Jenkins server for the authentication, then a Nexus for a Nexus integration, we have to use a Nexus uploader plugin. So that will be integrating with the Jenkins. So once this integration is done,

I will be writing the Jenkins file. So that Jenkins files, actually two types of Jenkins files are available. One is a declarative

file. Another one is a scripted pipeline. So we are using a scripted pipeline here. So completely, it was an a groovy script. So

I used to write 7  stages, actually. So the first Stage is like, so whenever you're starting with the scripted pipeline, the

groovy script, it starts with a node. So then the first stage is like source code checkout stage. So where I need to declare my GitHub credentials. So for authentication and all already, I configured my GitHub username and password in a Jenkins add

credentials area, so that it will making that authentication. So if it is a public repository, definitely we no need to do all those things. So in a project, we are keeping all those in a private repo. So I need to add my credentials in the Jenkins. So

the username of GitHub, I need to add the credentials in the Jenkins, that is under add credentials, once the checkout is

done. Next is a Maven. So Maven is completely building my WAR file. So where I'm declaring some couple of Maven goals, like a clean package, compile package, kind of things I need to declare. And the next stage is like a sonar cube stage. So

where I'm doing the quality code check for that one. So the some threshold limits we used to configure and also we need to attain some plugins for this one. So the plugins will be like a PMD, Feinberg static code analysis collector. So these are

the plugins I need to install into Jenkins for the sonar cube testing.And another stage is like, Nexus stage, so where I can store all my artifacts, so on one more stage we are having that is a Docker image stage where I'm building my Docker

images. So once the Docker image is built, we are storing all the images to the ECR registry, so we are using AWS ECR here.

So already I have integrated my AWS ECR with the Jenkins so that my Jenkins can able to find the exact ECR repo and it will

be pushing the images to the ECR. So this is the one stage, I'm adding on another stage is like, it's completely a Docker

container stage. So it will be launching the container. And one more stage I'm having like, it will be removing the existing

container. So I'll be adding that stage also. So this is like one of the continuous integration and continuous deployment or

delivery I can say. So these are the plugins and things I'm writing in my Jenkins file. So this is what how my ci CD pipeline works.

They are the ones who keep the lights on. I would put their responsibilities into following 6 buckets.

BUILD : Be it cloud based or on premise infrastructure, its the Devops Engineers who are responsible for building it. And they have to consider non functional requirements such as scalability, availability, security, disaster recovery when they do. In addition to building infrastructures, they are the ones who typically build VM and Container Images. Most of these tasks are automatable with tools such as terraform, packer etc.

INSTALL AND CONFIGURE : This involves installation and configurations of Operating Systems (which has now become easy with cloud and containers as base images are readily available), application stack including app servers, databases, middleware and anything in between. Devops engineers typically use declarative code to manage configurations with tools such as Ansible, Chef, Puppet.

CI/CD: Instant feedback is what continuous integration provides. Devops Engineers are the ones who are typically responsible for setting up and automating CI pipelines as well as defining and implementing Release/Deployment strategies. Be it container based or on cloud. Tools such as Jenkins, Spinnaker are useful to automate CI/CD pipelines.

ADMINISTER : Infrastructure is built, application stacks deployed, now what ? This is where the Devops Engineers spend most of their times. This includes application operations and debugging, a bit of database and storage administration, monitoring, optimising, security etc.

TROUBLESHOOT : This is a indispensable skill for a Devops Engineer and is useful for all tasks above. You are going to have issues, be it building infrastructures, configuration management, pipelines or just as part of running this infra. A Devops Engineer must have advanced level troubleshooting skills and must master the tools of the game.

AUTOMATE : Majority of the Devops Work can be automated. When not busy with issues or setting up things, a Devops Engineer typically spends time automating the recurring tasks.

If you ask any Devops Engineers what they do day in and day out basis, the answer would fall in one of the categories above