This is Gautami , I've 8+ years of experience into Devops and Cloud Engineering. Worked on AWS cloud. I was working with cognizant. Coming to my roles and responsibilities , I've been a part of Devops and Infra team , we take care of build and releases, Deployments and related issues, Tools Administration, Managing docker base images, Promoting and Deploying applications in staging to prod. Our applications are deployed on AWS EC2 and EKS. So we support AWS Infra, we use terrform to spin up the infrastructure. we have promotheus, Grafana and ELK stack to monitor Infra and applications. Coming to my techstack , I've worked on Git, bitbucket, Gitlab, Maven,Sonarqube, Jenkins, Nexus, Docker , Kubernetes , AWS , Terraform and ansible. Day to day activities.

Normally Developers raise promotion tickets in jira, once it is raised we deploy application on staging , post validation we deploy on prod env.if there is any build issues , we get support tickets which we work on. On tools side we work on upgrading the tools recently we have upgraded nexus 2 to nexus 3. I work on docker base images where i used to write dockerfile , build the dockerfile and publish the image to AWS ECR. We have calls with developers on Tuesday and Thursday where they come up with Infra requirments and we work on provisioning infra on aws using terrafrom , we use ansible galaxy roles for configuration management as well. On daily basis we have devops scrum calls within our team at 12 PM , we discuss progress on tickets assigned , if there is any blockers or roadmap for next tasks. we work on 2 weeks sprint model. We get on-call rotational shifts for monitoring the infra and application for 1 week in a month . we get alert notifications from pagerduty. CICD pipeline stages

---------------------

we are using multibranch pipelines, we have shared libraries to support our pipelines , ours is declarative scripting. we have

stage(checkout)- where we oull code from scm (git clone)

stage(build) - In this stage we build the code using maven ( mvn clean package)

stage(sonar) - we do static code analysis and unit test coverage in this stage (mvn sonar:sonar ) (cobertura or jacoco for unit test coverage)

stage(publish) - once quality gates are passed , we publish the artifact to Nexus repo. if branch is dev, we pupblish snapshot artficact , if branch is release we publish release artifact. (mvn deploy)

stage(docker) - In this stage , we create a application image and publish this image to AWS ECR. Git branching strategies

------------------------

Our entire build pipeline is work based on branching strategies .

feature branch - So developers start building their features in this branch .

Dev branch - Developer raise PR to merge the feature branch changes to dev branch . Dev Lead or manager has access to review/Approve PR and merge to Dev branch, based on feature branch build , they take the decision.

- (Bugfix Brnach) Once the code is merged to dev branch , we build application and deploy the snapshot artifact to staging env. if QA team finds any bug they raise bugfix ticket and dev team creates bugfix branch , they fix the bug and raise PR to dev branch. Once bugfix is merged to dev, we do release cuts.

Release branch - During Release cuts , we create Release branch from Dev branch and buil Release artifact and deploy it on Pre-prod env. If there is any issue on validation, Validation team creates hotfix ticket .

- (Hotfix branch) Developers create hotfix branch from the ticket and fix the issue and raised PR to Dev branch and new Release branch from dev branch is created and build the release artifact . Once everything goes fine , Release Artifact is deployed On production.

Master Branch - Once production is stable , we merge the Release changes to Master branch to make sure that Master branch code and Production is same. Deployment -On-EC2 :

==============================

We are following blue-green deployment strategy to deploy applications on EC2. We have 3 jobs to do it in sequence.

--> First job is packer job which we create or update base AMI whenever there is upgrade or change in requirement.

-->Second job is to create application AMI image by using base AMI image. Basically this will pull the base AMI from Amazon AMI(Amazon Machine Image), create an EC2 instance out of it and script pulls Ansible galaxy roles , install dependencies and deploys the service. Once the service is up , it checks health and once health check is green and successful, we create application AMI and publish it to Amazon AMIs with tag of application name, version and timestamp.

-->Third job has few deploy.yaml files which will have the environment details like DEV,STG and PROD. Based on the environment parameter that we pass while triggering, job will pull the application image from Amazon AMIs and create an instance. This instance is attached to target group, Autoscaling Group and security groups.

Once the new instance gets scaled , old instances gets terminated.