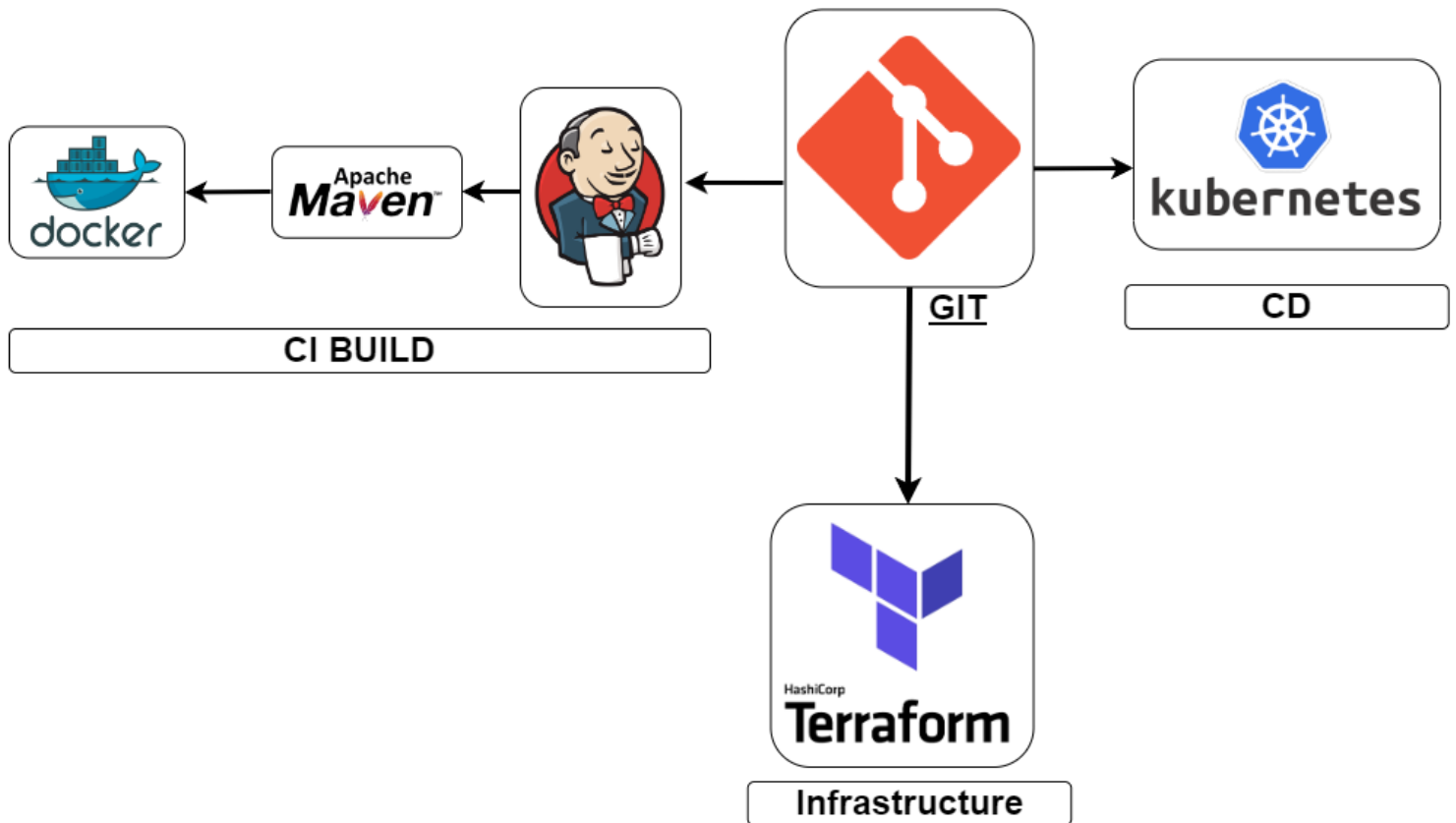


GitOps With Jenkins

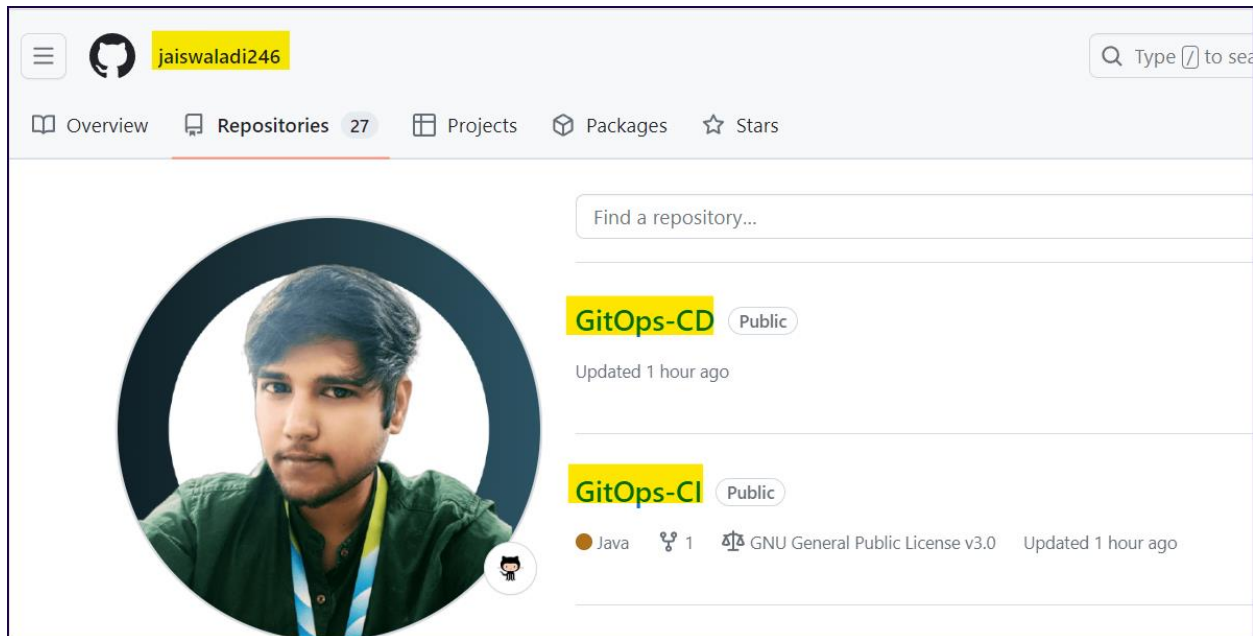
By-> Aditya Jaiswal From DevOps Shack



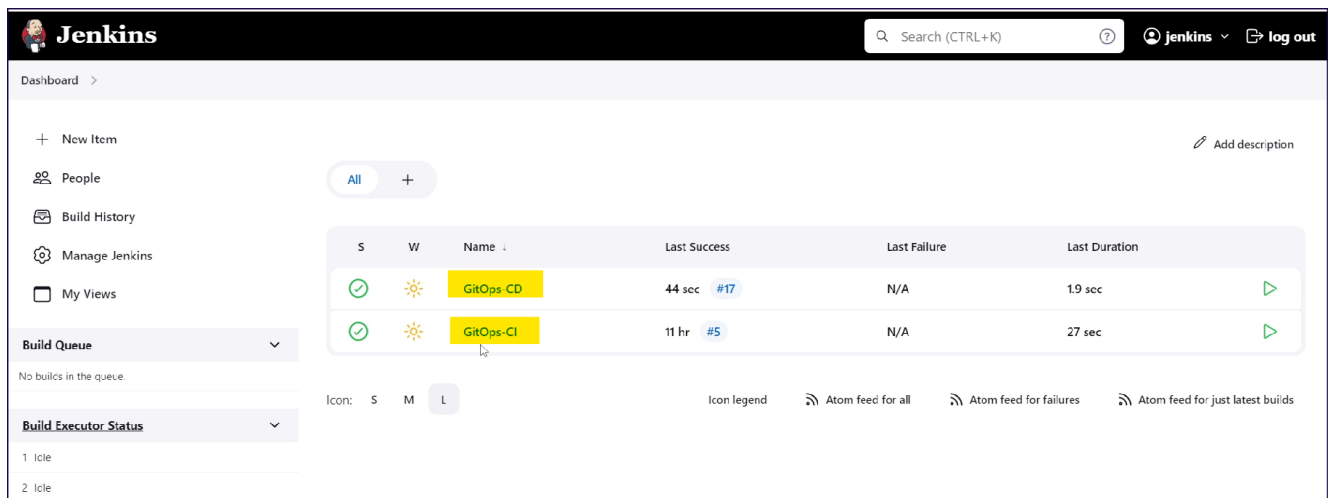
- GitOps is a software development and operational approach that streamlines the management and deployment of applications and infrastructure by using version control systems like **Git as the single source of truth** for defining and controlling the entire system
- GitOps can be used to build development pipelines, code applications, manage configurations, provision Kubernetes clusters, and deploy on Kubernetes or container registries.

Implementation

1. Created 2 repositories for CI & CD separately.



2. Created a instance of Jenkins



3. Configured the Jenkins Job to perform CI & CD separately.

```
1  pipeline {
2      agent any
3      tools{
4          jdk 'jdk11'
5          maven 'maven3'
6      }
7
8      stages {
9          stage('Git Checkout') {
10             steps {
11                 git branch: 'main', url: 'https://github.com/jaiswaladi246/GitOps-CI.git'
12             }
13         }
14         stage('Maven Build') {
15             steps {
16                 sh 'mvn clean package -DskipTests=true'
17             }
18         }
19         stage('Docker Build & Push') {
20             steps {
21                 script{
22                     withDockerRegistry(credentialsId: 'docker-cred', toolName: 'docker') {
23
24                         sh "docker build -t shopping-cart -f docker/Dockerfile ."
25                         sh "docker tag shopping-cart adijaiswal/shopping-cart:latest"
26                         sh "docker push adijaiswal/shopping-cart:latest"
27                     }
28                 }
29             }
30         }
31     }
32 }
33 }
```

4. Configured WEBHOOKS to Automate the CI & CD as below

Webhooks / Manage webhook

Settings

Recent Deliveries

We'll send a POST request to the URL below with details of any subscribed events. You can also specify which data format you'd like to receive (JSON, x-www-form-urlencoded, etc). More information can be found in [our developer documentation](#).

Payload URL *

http://3.111.171.246:8080/github-webhook/

Content type

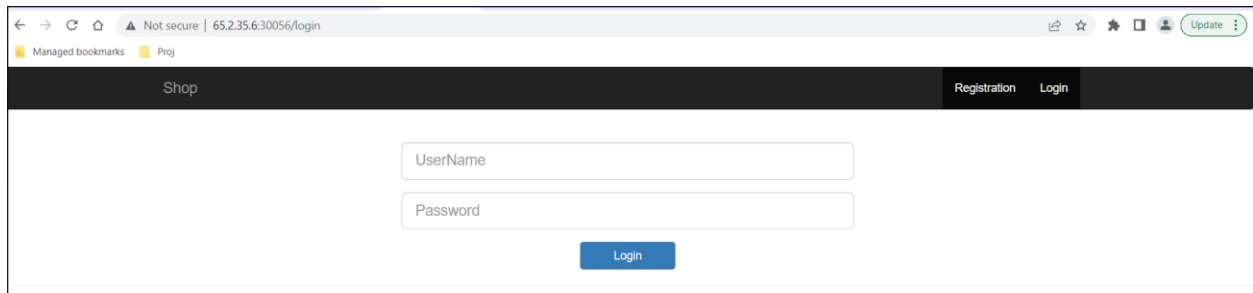
application/x-www-form-urlencoded

Secret

5. I set up my own Kubernetes Cluster

```
root@ip-172-31-44-191:/home/ubuntu# kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
debug                               1/1     Running   0           13h
spring-boot-k8s-deployment-784fc786b6-fm2r2  1/1     Running   0           12h
root@ip-172-31-44-191:/home/ubuntu#
```

6. Now We can manage complete CI & CD through scripts inside the git and use git as single source of truth.



The screenshot displays a web browser window with the address bar showing '65.2.35.6:30056/login'. The page features a dark navigation bar with the word 'Shop' on the left and 'Registration' and 'Login' links on the right. The main body of the page is white and contains a login form with two text input fields: 'UserName' and 'Password'. Below these fields is a blue button labeled 'Login'. The browser's status bar at the bottom indicates 'Not secure' and shows managed bookmarks for 'Proj'.

7. After every push or changes in the scripts the same changes will be applied to Deployed application.