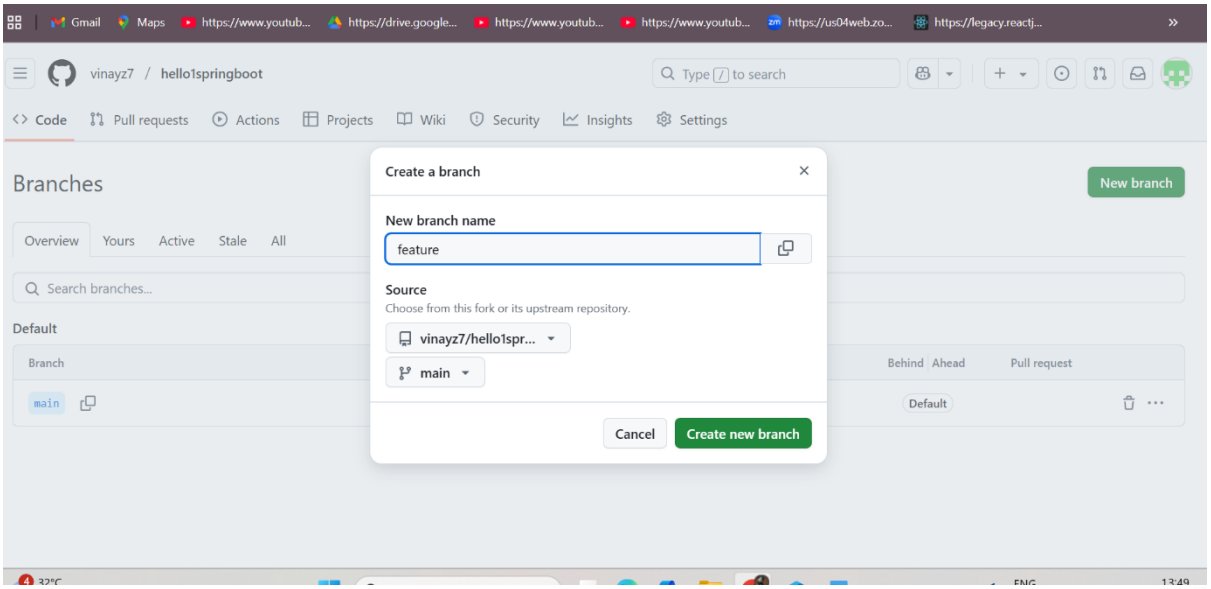
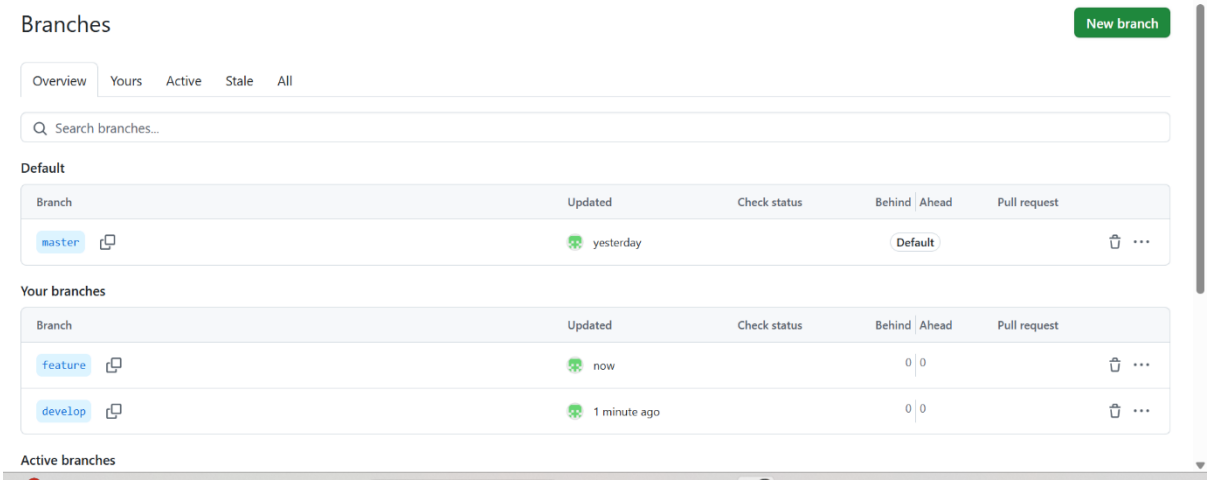
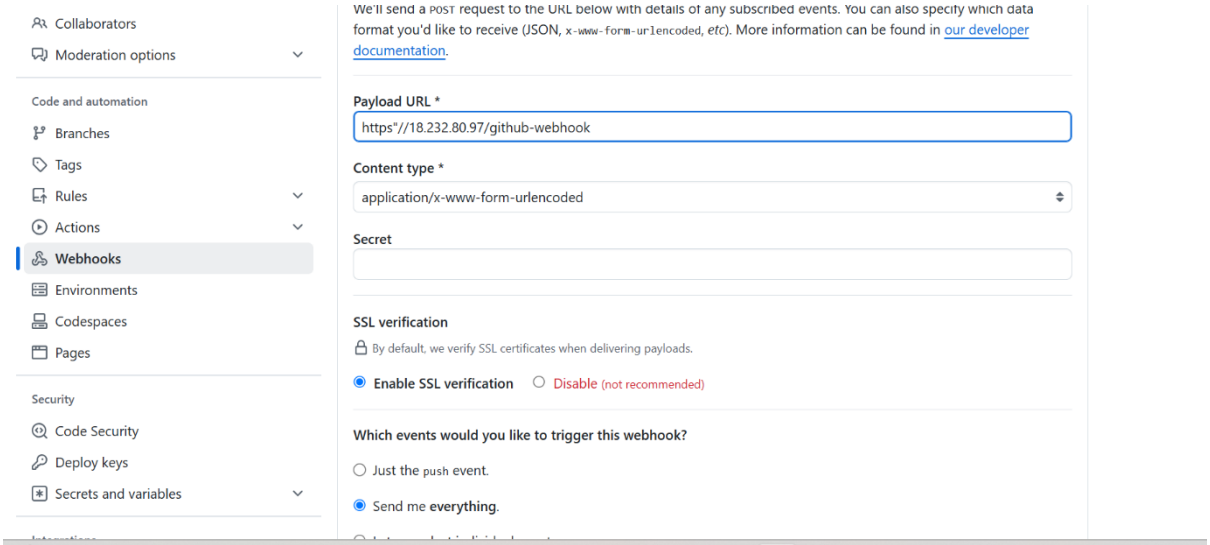
I. Branching Strategy and Execution Rules

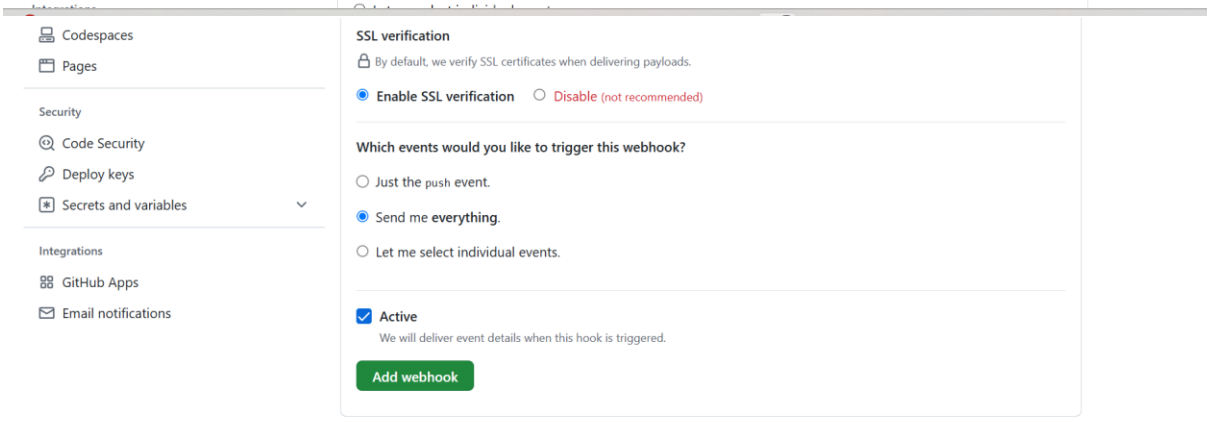


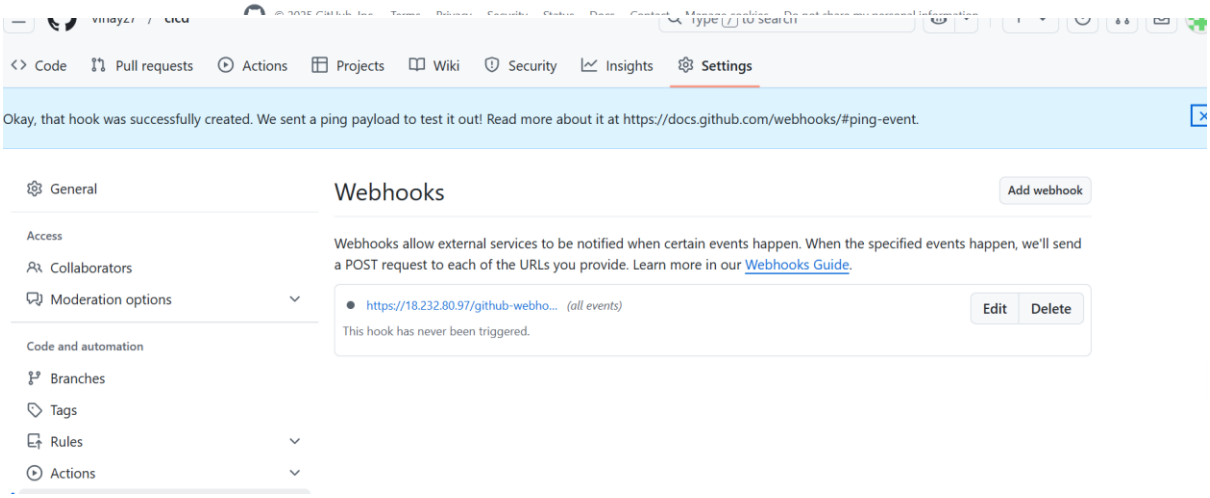


II. Webhook Integration

a. Configure a GitHub webhook that triggers Jenkins when:



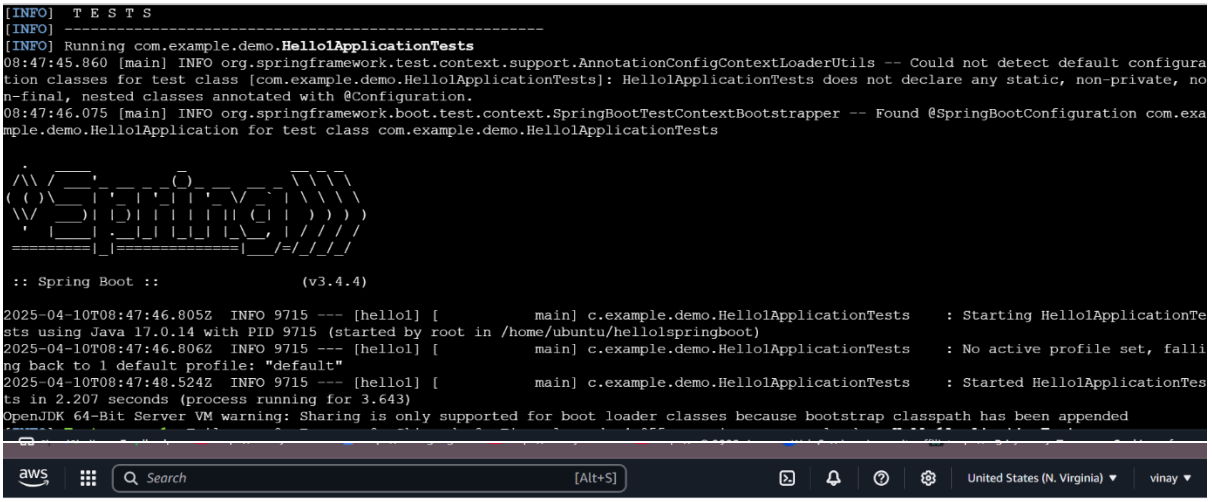


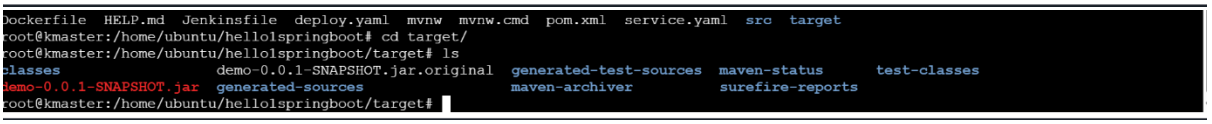


III. Building Java Code with Maven

a. Compile the application using Maven.

b. Generate .jar file as the build artifact.





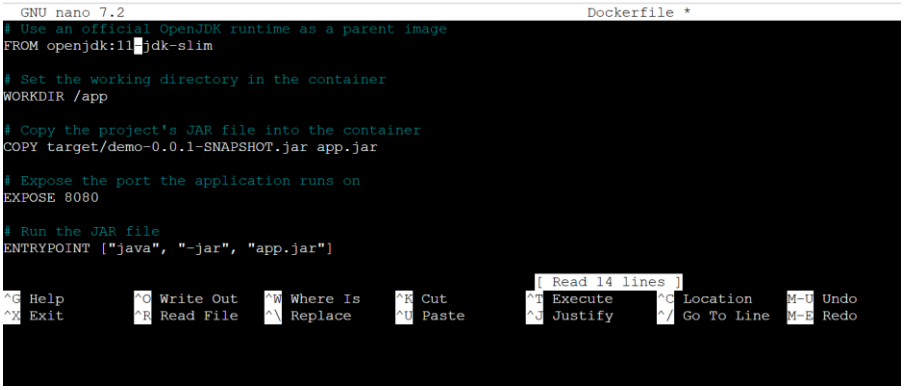
a. Write a Dockerfile with the following specifications:

i. Base Image: openjdk:11-jre-slim

ii. Copies the application JAR from the build output.

iii. Exposes port 8080 (the application port).

iv. Uses ENTRYPOINT [“java”, “-jar”, “/app.jar”].



Command : docker build -t javaimage .

```
 => => extracting sha256:69e15dccd787ba2cfe67f6abf5970ed88a5e019efbb499e499da3ab20b85fcc7                                  3.7
s
 => [internal] load build context                                                                                         0.4
s
 => => transferring context: 20.75MB                                                                                      0.3
s
 => [2/3] WORKDIR /app                                                                                                    2.5
s
 => [3/3] COPY target/demo-0.0.1-SNAPSHOT.jar app.jar                                                                     0.2
s
 => exporting to image                                                                                                    0.2
s
 => => exporting layers                                                                                                   0.2
s
 => => writing image sha256:f3fe57e9d45f817943f6d499b88faf7e55b49b044f0c1fb0792a8ad94aca234b                              0.0
s
 => => naming to docker.io/library/javaimage                                                                              0.0
s
root@kmaster:/home/ubuntu/hello1springboot# docker images
REPOSITORY                 TAG       IMAGE ID       CREATED          SIZE
javaimage                  latest    f3fe57e9d45f   13 seconds ago   444MB
<none>                     <none>    86f84ddb2bca   8 minutes ago    811MB
vinayz7/java-web-app-cicd  latest    a31fa9f1aeac   8 minutes ago    811MB
vinayz7/java-web-app-cicd  <none>    eadb2e34c745   14 minutes ago   811MB
helloworld                 latest    1f05d93baa49   23 hours ago     136MB
```

```
root@kmaster:/home/ubuntu/hello1springboot# docker run -d -it javaimage bash
97836936e3b62c7295e66e3cbe73705be6742834f807139e88d5f8ddc1ecc716
root@kmaster:/home/ubuntu/hello1springboot# docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
root@kmaster:/home/ubuntu/hello1springboot# docker ps -a
CONTAINER ID   IMAGE        COMMAND                  CREATED          STATUS                      PORTS     NAMES
97836936e3b6   javaimage    "java -jar app.jar b…"   8 seconds ago    Exited (1) 7 seconds ago              trusting_sutherland
e192a2ac049c   javaimage    "java -jar app.jar b…"   29 seconds ago   Exited (1) 28 seconds ago             jovial_snyder
eecda4aecf49   javaimage    "java -jar app.jar"      45 seconds ago   Exited (1) 44 seconds ago             condescending_chaplygin
dc9dfca710d3   helloworld   "python app.py"          23 hours ago     Exited (137) 20 hours ago             flask-container
root@kmaster:/home/ubuntu/hello1springboot#
```

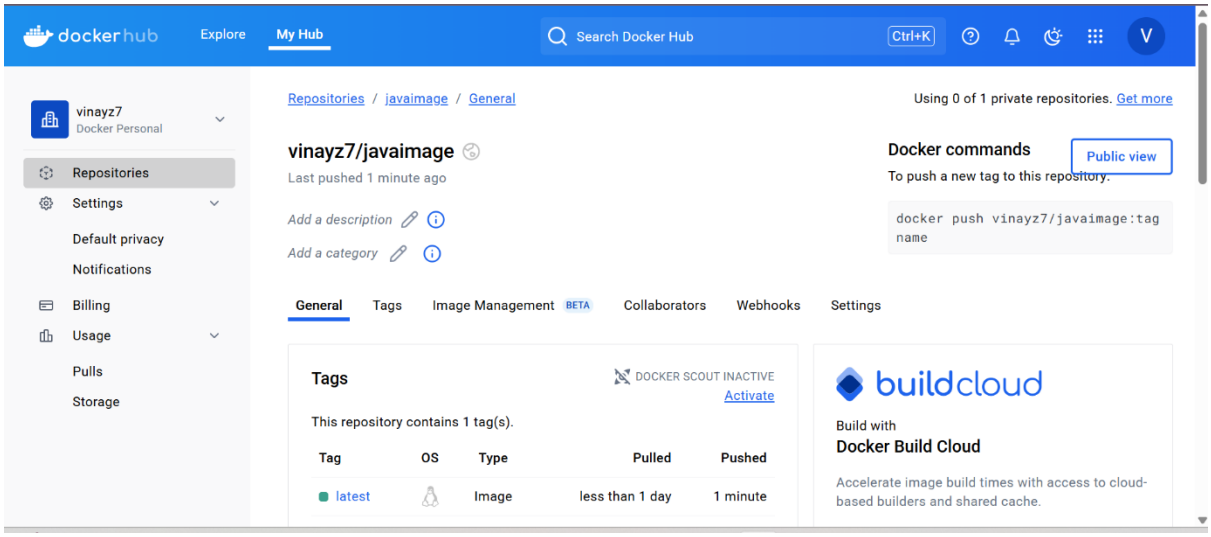
Docker push:

Dochub link: https://hub.docker.com/repository/docker/vinayz7/javaimage/general

```
root@kmaster:/home/ubuntu/hello1springboot# docker login
Authenticating with existing credentials... [Username: vinayz7]

i Info → To login with a different account, run 'docker logout' followed by 'docker login'


Login Succeeded
root@kmaster:/home/ubuntu/hello1springboot# docker tag javaimage:latest vinayz7/javaimage:latest
root@kmaster:/home/ubuntu/hello1springboot# docker push vinayz7/javaimage:latest
The push refers to repository [docker.io/vinayz7/javaimage]
ebc3b70d57e9: Pushed
2365b390f092: Pushed
eb6ee5b9581f: Mounted from library/openjdk
e3abdc2e9252: Mounted from library/openjdk
eafe6e032dbd: Mounted from library/openjdk
92a4e8a3140f: Mounted from library/openjdk
latest: digest: sha256:24f5528e9f085c9e06e168f8591d9c25bca43f030ebaa995cc7edc0a14f848a4 size: 1578
root@kmaster:/home/ubuntu/hello1springboot#
```



VI. Kubernetes Deployment

a. Write Kubernetes manifest files that include:

i. deployment.yaml (Defines the app deployment and replicas).

ii. service.yaml (Exposes the application internally).

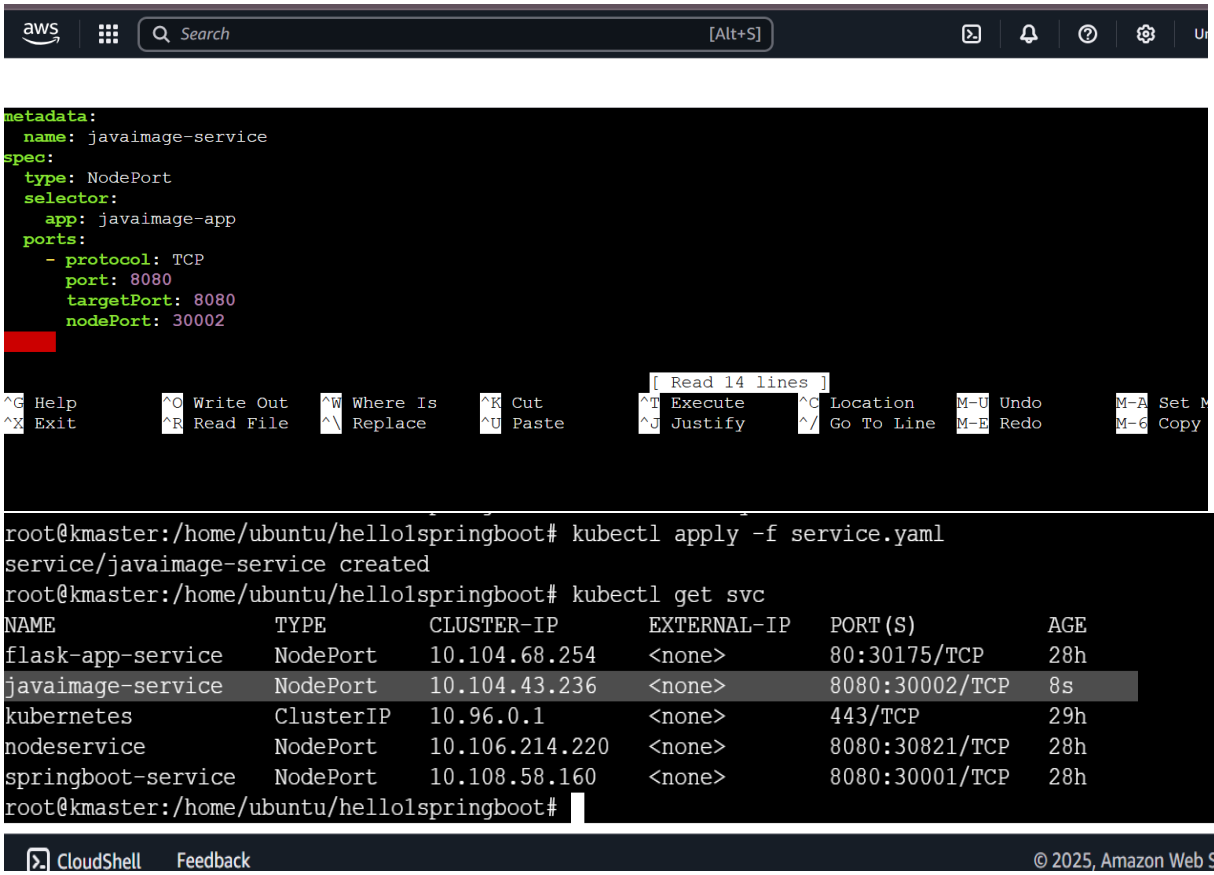
b. Add Kubernetes apply command to Jenkinsfile to enable deployment.

Kubectl apply -f deploy.yaml

```
root@kmaster:/home/ubuntu/hello1springboot# nano deploy.yaml
root@kmaster:/home/ubuntu/hello1springboot# kubectl apply -f deploy.yaml
deployment.apps/javaimage-app created
```

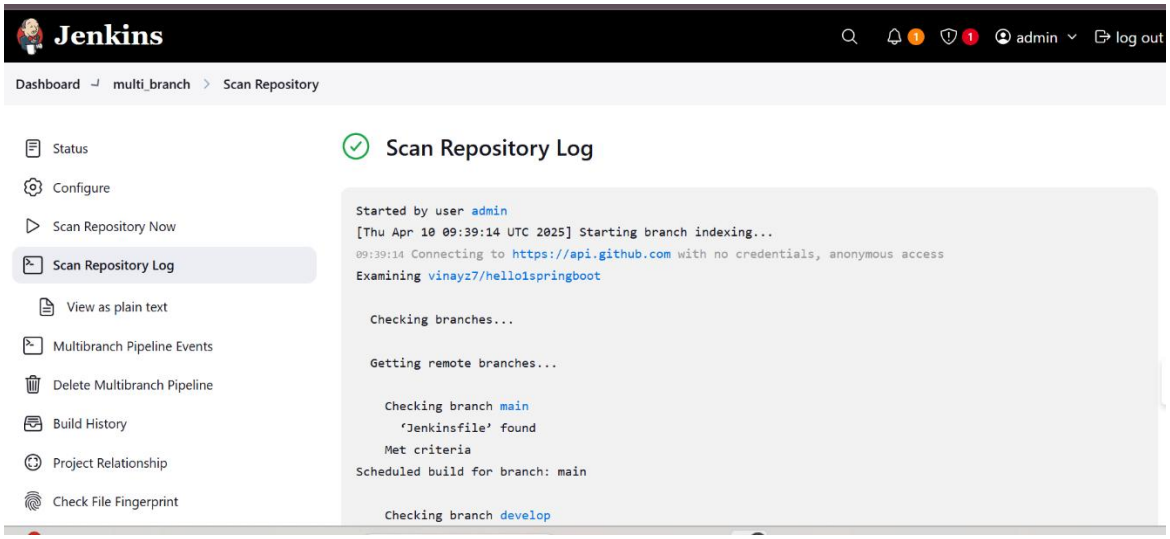
```
root@kmaster:/home/ubuntu/hello1springboot# kubectl get deploy
NAME                   READY   UP-TO-DATE   AVAILABLE   AGE
flask-app-deployment   2/2     2            2           28h
javaimage-app          1/1     1            1           53s
springboot-app         1/1     1            1           28h
tomcat-deploy          0/1     1            0           28h
root@kmaster:/home/ubuntu/hello1springboot#
```

service.yaml (Exposes the application internally).

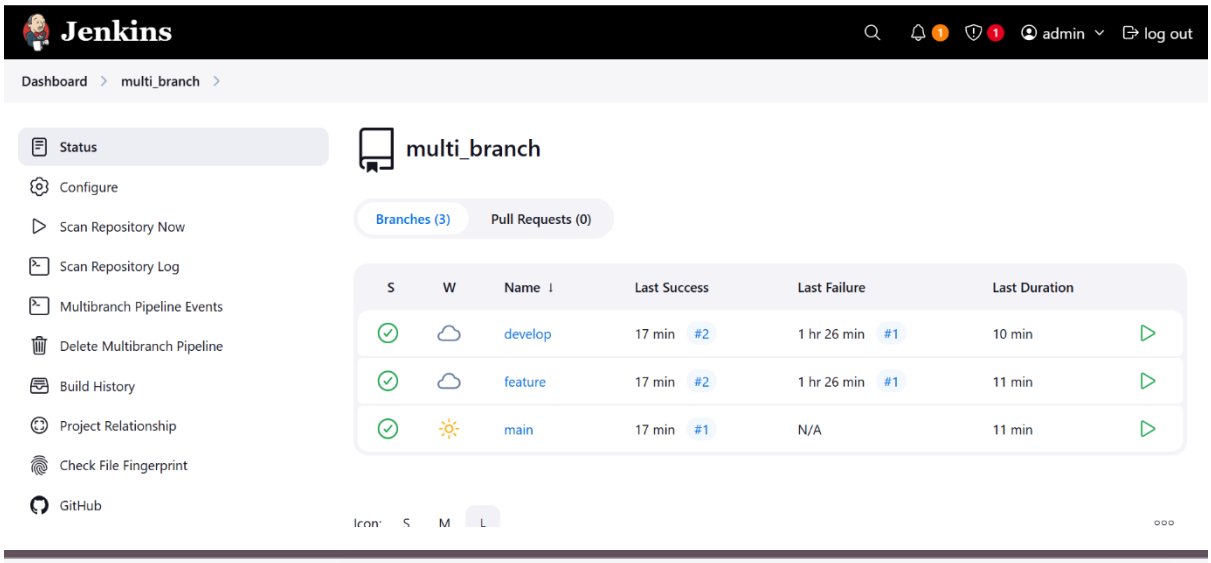


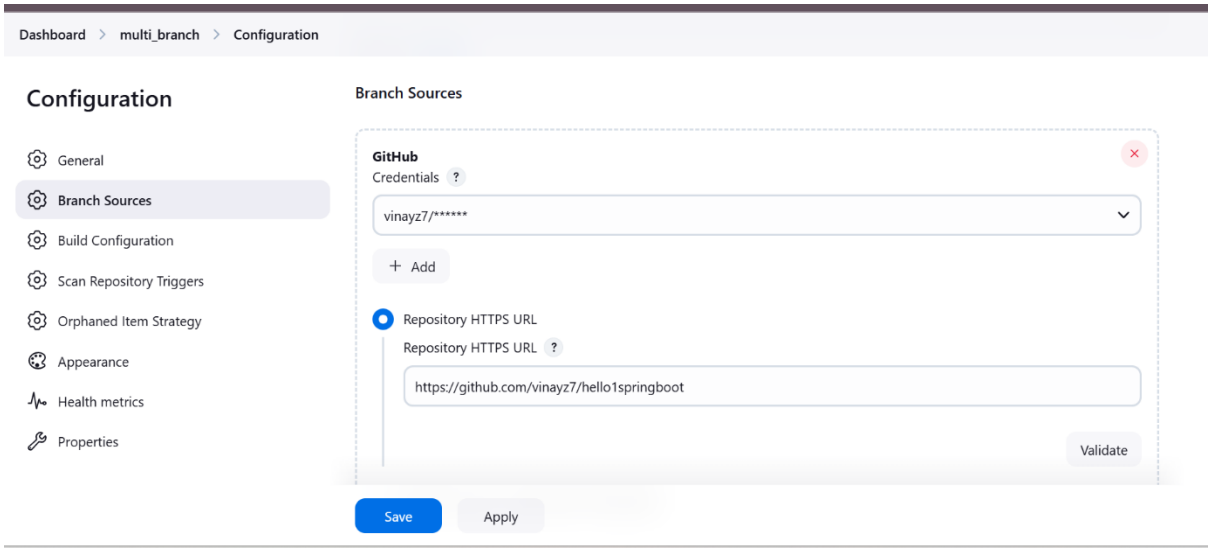
1. **Implement a CI/CD pipeline using Jenkins multi-branch pipelines , Docker for containerization, GitHub webhooks and Kubernetes for deployment.**

Github repo: https://github.com/vinayz7/hello1springboot



18.232.80.97/job/multi_branch/credentials





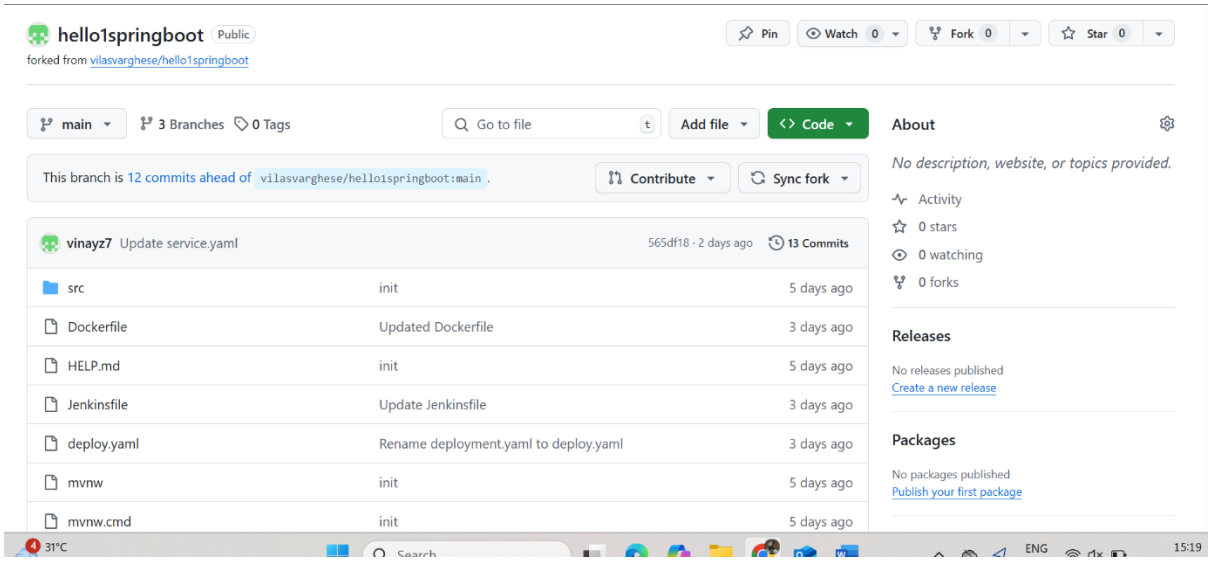
Jenkins pipe line script :

```
pipeline {
  agent any
  environment {
    DOCKER_IMAGE = 'vinayz7/javaimage:latest'
  }
  stages {
    stage('Clone Repository') {
      steps {
        git url: 'https://github.com/vinayz7/hello1springboot', branch: 'main'
      }
    }
   stage('build mvn for jar') {
      steps {
        sh 'mvn clean install'
      }
    }

  stage('Build Docker Image') {
      steps {
        sh 'docker build -t $DOCKER_IMAGE .'
      }
    }
    stage('Push Docker Image') {
      steps {
        withCredentials([usernamePassword(credentialsId: 'docker-hub-credentials',
usernameVariable: 'DOCKER_USER', passwordVariable: 'DOCKER_PASS')]) {

          sh """
            docker login -u $DOCKER_USER -p $DOCKER_PASS
          """
          sh 'docker push $DOCKER_IMAGE'
        }
```

```
        }
    }
    stage('Deploy to Kubernetes') {
        steps {
            sh 'kubectl apply -f deploy.yaml'
        }
    }
    stage('Expose Service') {
        steps {
            sh 'kubectl apply -f service.yaml'
        }
    }
  }
}
```



Webhook-after any changes commints in repo it will automatically trigger