

DEVOPS

DAY – 6

FINAL PROJECT

Java Application Minikube Deployment:

1. sudo nano deployment

deployment.yml

```
{
  apiVersion: apps/v1
  kind: Deployment
  metadata:
    name: my-deploy
  labels:
    name: my-deploy
  spec:
    replicas: 4
    selector:
      matchLabels:
        apptype: web-backend
    strategy:
      type: RollingUpdate
  template:
    metadata:
      labels:
        apptype: web-backend
    spec:
```

containers:

- name: my-app

- image: bavyadharshini/simplewebapp:latest

ports:

- containerPort: 7076

}

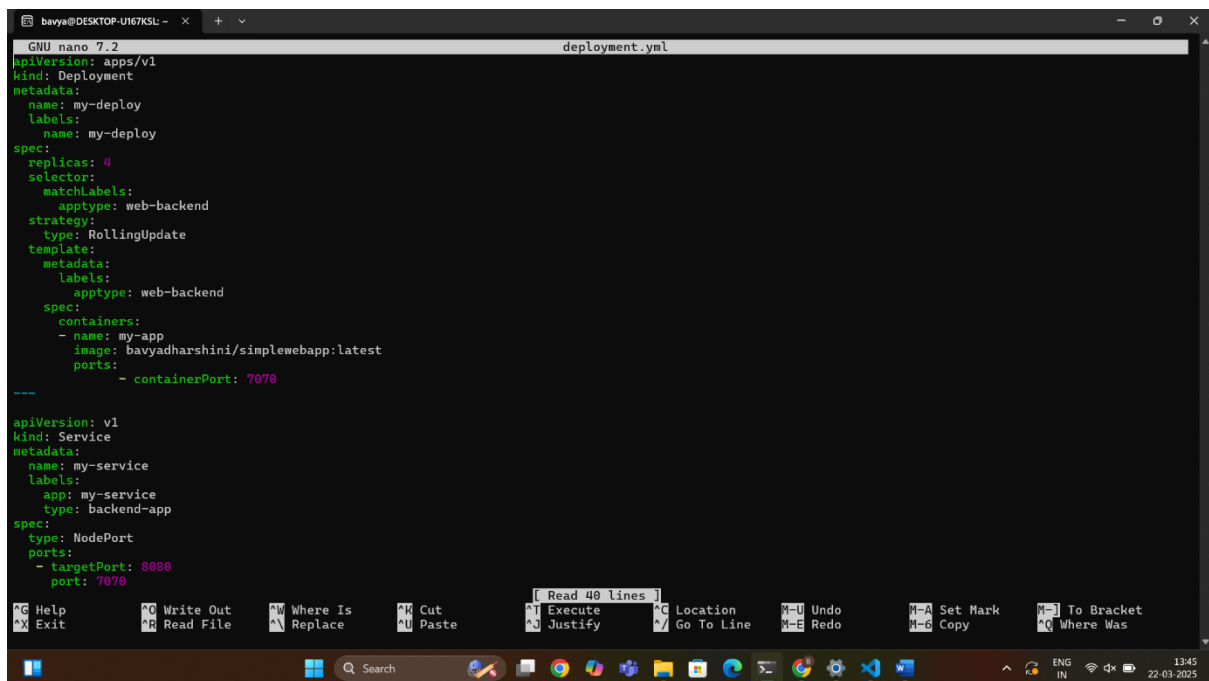
2. kubectl get pod

3. minikube service my-service

4. kubectl port-forward svc/my-service 9000:7070

URL: localhost:9000/maven-web-app/

Output:



The screenshot shows a terminal window with the nano 7.2 text editor open, editing a file named deployment.yaml. The file content is as follows:

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-deploy
  labels:
    name: my-deploy
spec:
  replicas: 4
  selector:
    matchLabels:
      apptype: web-backend
  strategy:
    type: RollingUpdate
  template:
    metadata:
      labels:
        apptype: web-backend
    spec:
      containers:
      - name: my-app
        image: bavyadharshini/simplewebapp:latest
        ports:
        - containerPort: 7070

---
apiVersion: v1
kind: Service
metadata:
  name: my-service
  labels:
    app: my-service
    type: backend-app
spec:
  type: NodePort
  ports:
  - targetPort: 8080
    port: 7070
```

The terminal window also shows a Windows taskbar at the bottom with various icons and the system clock indicating 13:45 on 22-03-2025.

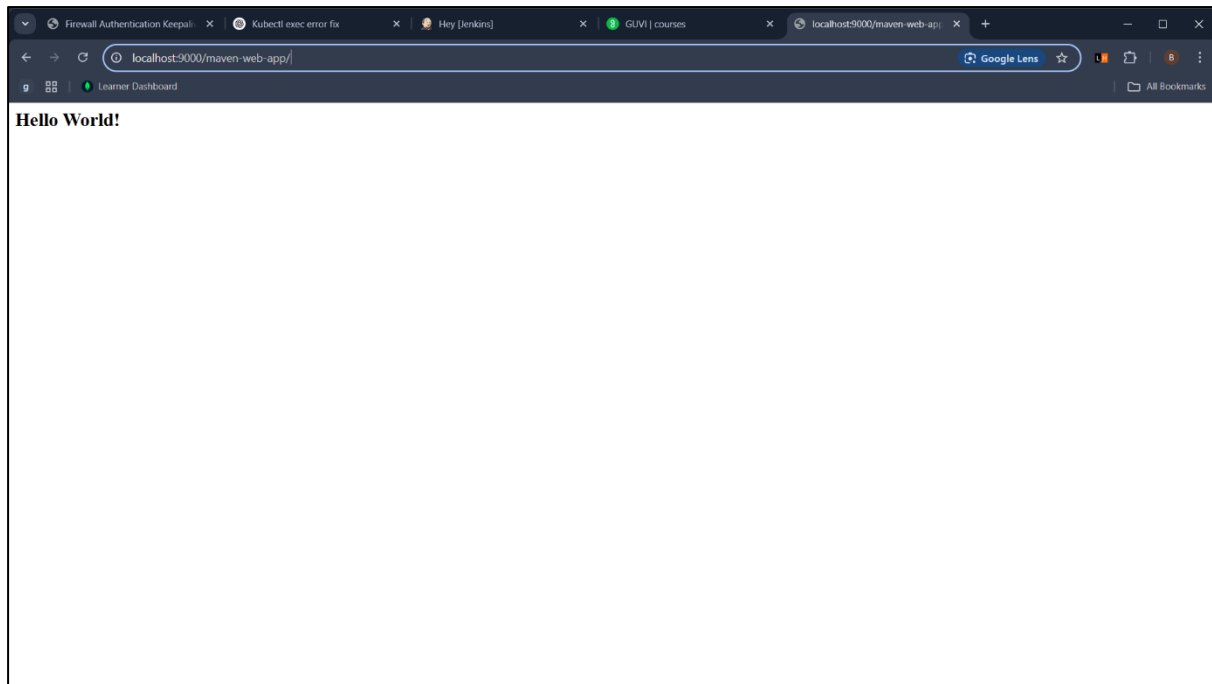
deployment.yaml

```
bavya@DESKTOP-U167KSL: ~  
Forwarding from [::1]:39997 -> 8080  
*Chavya@DESKTOP-U167KSL:~$ kubectl port-forward svc/my-service 9000:7070  
Forwarding from 127.0.0.1:9000 -> 8080  
Forwarding from [::1]:9000 -> 8080  
Handling connection for 9000  
Handling connection for 9000  
Handling connection for 9000  
Handling connection for 9000  
Handling connection for 9000  
Handling connection for 9000  
Handling connection for 9000  
*Chavya@DESKTOP-U167KSL:~$ kubectl get pod  
NAME          READY   STATUS    RESTARTS   AGE  
my-deploy-9ff857f79-djjdm 1/1     Running   0           14m  
my-deploy-9ff857f79-q8xgs 1/1     Running   0           14m  
my-deploy-9ff857f79-s454c 1/1     Running   0           14m  
my-deploy-9ff857f79-tm2hv 1/1     Running   0           14m  
bavya@DESKTOP-U167KSL:~$ kubectl exec -it my-deploy-9ff857f79-djjdm bin/bash  
error: exec [POD] [COMMAND] is not supported anymore. Use exec [POD] -- [COMMAND] instead  
See 'kubectl exec -h' for help and examples  
bavya@DESKTOP-U167KSL:~$ kubectl exec -it my-deploy-9ff857f79-djjdm -- bin/bash  
OCI runtime exec failed: exec failed: unable to start container process: exec: "bin/bash": stat bin/bash: no such file or directory: unknown  
command terminated with exit code 126  
bavya@DESKTOP-U167KSL:~$ kubectl exec -it my-deploy-9ff857f79-djjdm -- bin/sh  
OCI runtime exec failed: exec failed: unable to start container process: exec: "bin/sh": stat bin/sh: no such file or directory: unknown  
command terminated with exit code 126  
bavya@DESKTOP-U167KSL:~$ kubectl exec -it my-deploy-9ff857f79-djjdm -- bin/bash/  
OCI runtime exec failed: exec failed: unable to start container process: exec: "bin/bash/": stat bin/bash/: no such file or directory: unknown  
command terminated with exit code 126  
bavya@DESKTOP-U167KSL:~$ kubectl exec -it my-deploy-9ff857f79-djjdm -- /bin/bash  
root@my-deploy-9ff857f79-djjdm:/usr/local/tomcat# ls  
bin          conf          filtered-KEYS  LICENSE      native-jni-lib  README.md     RELEASE-NOTES  RUNNING.txt   upstream-KEYS  webapps.dist  
BUILDING.txt CONTRIBUTING.md lib           NOTICE  
root@my-deploy-9ff857f79-djjdm:/usr/local/tomcat# cd webapps  
root@my-deploy-9ff857f79-djjdm:/usr/local/tomcat/webapps# ls  
maven-web-app  maven-web-app.war  
root@my-deploy-9ff857f79-djjdm:/usr/local/tomcat/webapps# exit  
exit  
bavya@DESKTOP-U167KSL:~$ kubectl port-forward svc/my-service 9000:7070  
Forwarding from 127.0.0.1:9000 -> 8080  
Forwarding from [::1]:9000 -> 8080
```

Port Forwarding

```
bavya@DESKTOP-U167KSL: ~  
Forwarding from [::1]:42023 -> 8080  
*Chavya@DESKTOP-U167KSL:~$ sudo nano deployment  
[sudo] password for bavya:  
bavya@DESKTOP-U167KSL:~$ kubectl apply -f deployment.yml  
deployment.apps/my-deploy configured  
service/my-service unchanged  
bavya@DESKTOP-U167KSL:~$ kubectl get pod  
NAME          READY   STATUS    RESTARTS   AGE  
my-deploy-9ff857f79-djjdm 1/1     Running   0           7m25s  
my-deploy-9ff857f79-q8xgs 1/1     Running   0           7m25s  
my-deploy-9ff857f79-s454c 1/1     Running   0           7m25s  
my-deploy-9ff857f79-tm2hv 1/1     Running   0           7m25s  
bavya@DESKTOP-U167KSL:~$ minikube service my-service  
|-----|  
| NAMESPACE | NAME      | TARGET PORT | URL                               |  
|-----|  
| default   | my-service | 7070        | http://192.168.49.2:30001       |  
|-----|  
★ Starting tunnel for service my-service.  
docker@127.0.0.1's password: |-----|  
| NAMESPACE | NAME      | TARGET PORT | URL                               |  
|-----|  
| default   | my-service | 7070        | http://127.0.0.1:44821         |  
|-----|  
🔗 Opening service default/my-service in default browser...  
🔗 http://127.0.0.1:44821  
! Because you are using a Docker driver on linux, the terminal needs to be open to run it.  
💡 Stopping tunnel for service my-service.  
bavya@DESKTOP-U167KSL:~$ kubectl port-forward svc/my-service 0000:7070  
Forwarding from 127.0.0.1:39997 -> 8080  
Forwarding from [::1]:39997 -> 8080  
*Chavya@DESKTOP-U167KSL:~$ kubectl port-forward svc/my-service 9000:7070  
Forwarding from 127.0.0.1:9000 -> 8080  
Forwarding from [::1]:9000 -> 8080  
Handling connection for 9000  
Handling connection for 9000  
Handling connection for 9000  
Handling connection for 9000  
Handling connection for 9000  
Handling connection for 9000
```

Handling Output



Output

Commands:

Java Application Jenkins:

Visudo:

1. `sudo visudo`
2. Add jenkins ALL=(ALL) NOPASSWD: ALL

ssh installtion:

1. `sudo systemctl restart ssh.service`
2. `sudo systemctl restart sshd.service`
3. `sudo apt update`
4. `sudo apt install openssh-server`
5. `sudo systemctl restart ssh`
6. `sudo systemctl status ssh`
7. `ls /etc/systemd/system/sshd.service` or `ls /usr/lib/systemd/system/sshd.service`

8. sudo systemctl daemon-reload
9. sudo systemctl status ssh

Deployment:

1. cd ~/.kube
2. ls
3. cat config
4. sudo vi config
5. i
6. -data
7. cat url | base64 -w 0; echo
8. minikube start
9. kubectl get node

Pipeline Script:

```
pipeline {
    agent any

    stages {
        stage('SCM Checkout') {
            steps {
                git branch: 'main', url: 'https://github.com/Bavyadharshini-Rajaganapathy/simple-web-app.git'
            }
        }

        stage('Build') {
            steps {
                sh 'mvn clean'
                sh 'mvn install'
            }
        }
    }
}
```

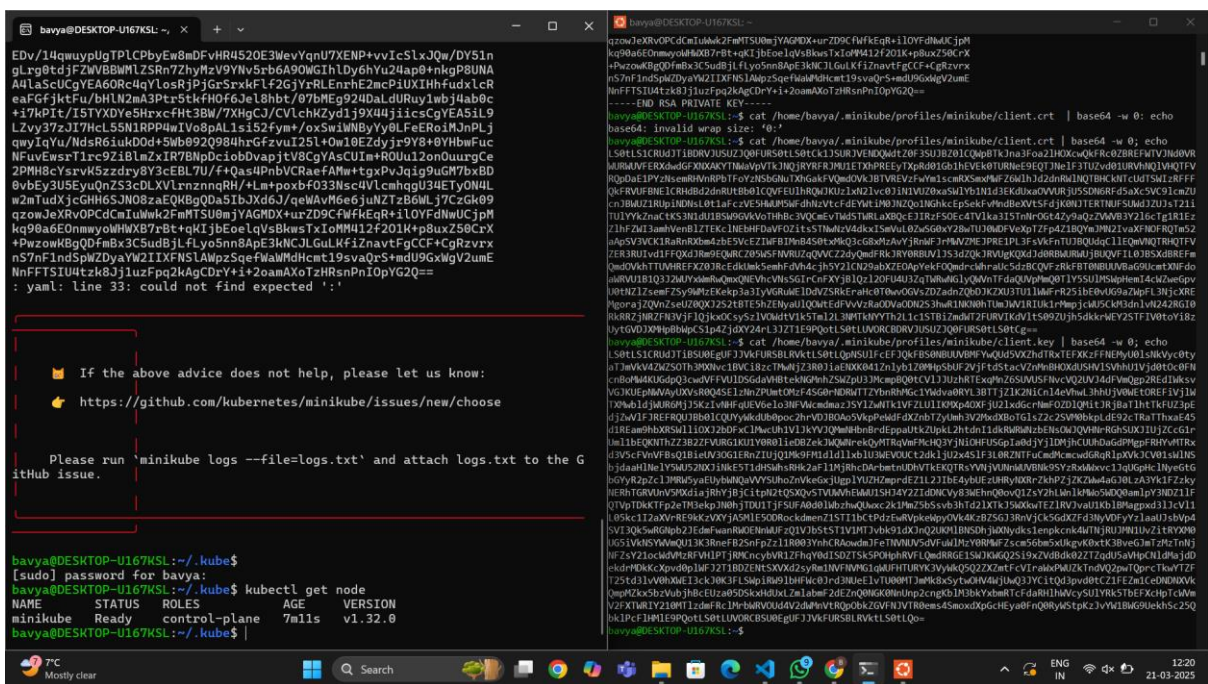
```

stage('Build Docker Image') {
    steps {
        script {
            sh 'docker build -t bavyadharshini/simplewebapp .'
        }
    }
}

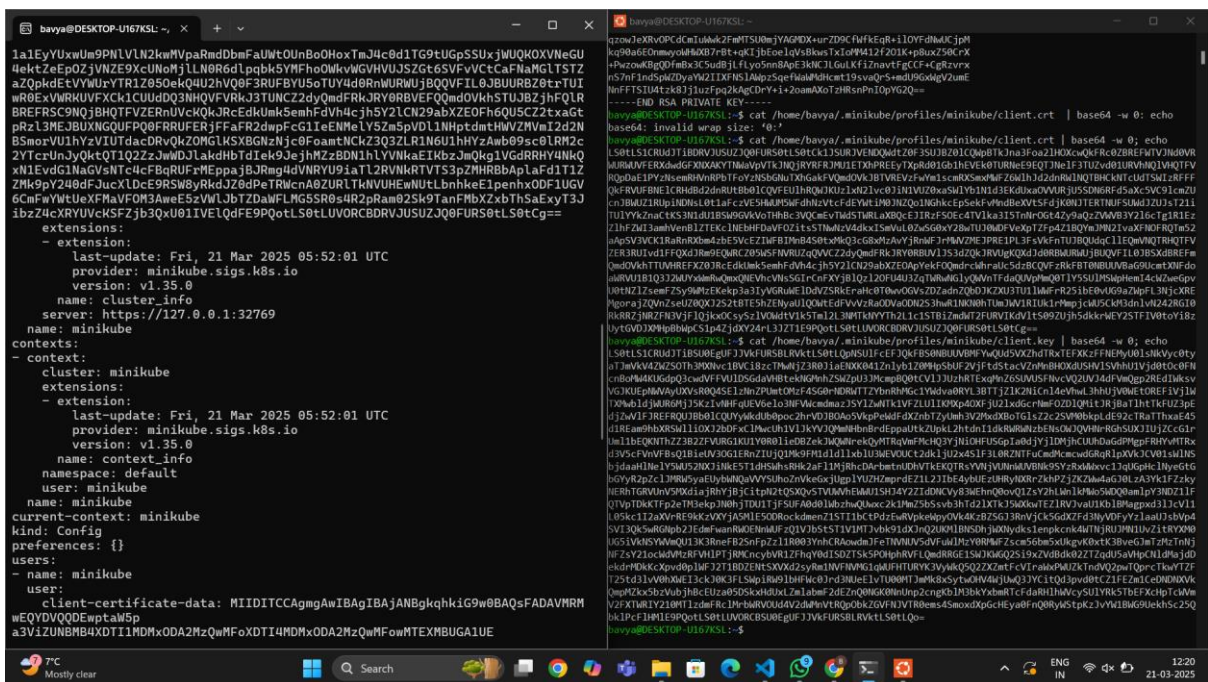
stage('Push to Docker Hub') {
    steps {
        script {
            withDockerRegistry(credentialsId: 'Docker_cred', url:
https://index.docker.io/v1/) {
                sh 'docker push bavyadharshini/simplewebapp'
            }
        }
    }
}

stage('test') {
    steps {
        withKubeConfig(caCertificate: "", clusterName: 'minikube',
contextName: 'minikube', credentialsId: 'minikube_cred', namespace: "",
restrictKubeConfigAccess: false, serverUrl: https://192.168.39.226:8443)
            sh 'kubectl apply -f deployment.yml --validate=false'
        }
    }
}

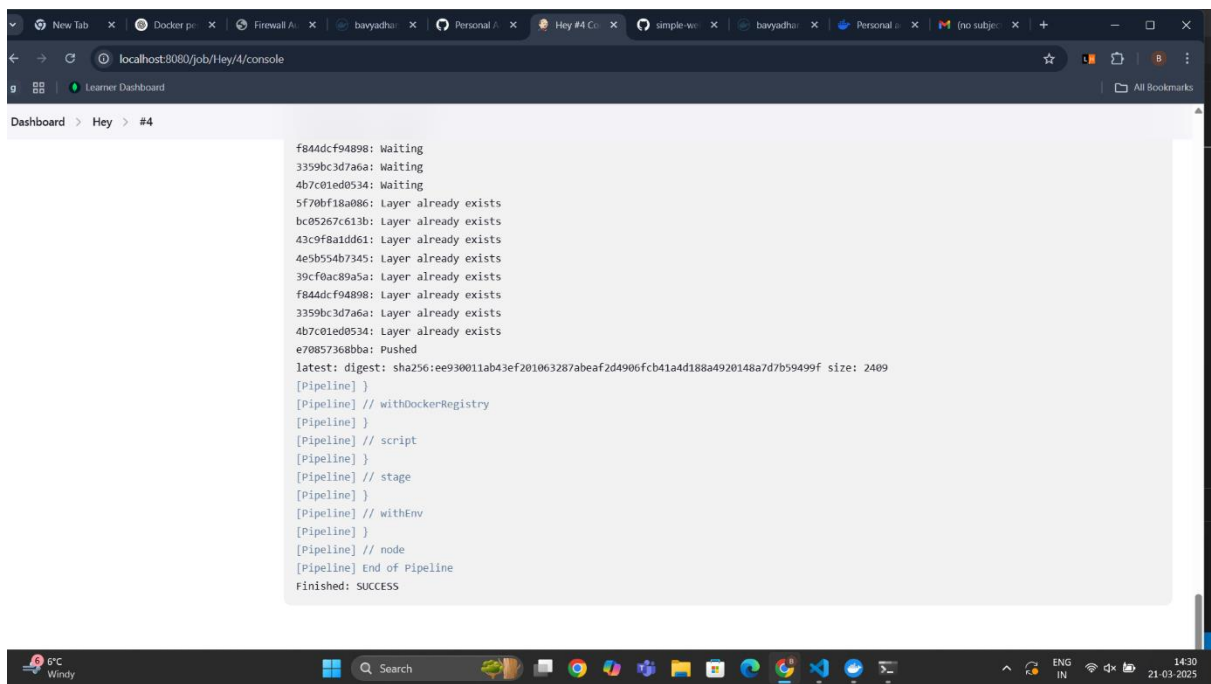
```



Deployment of Yml



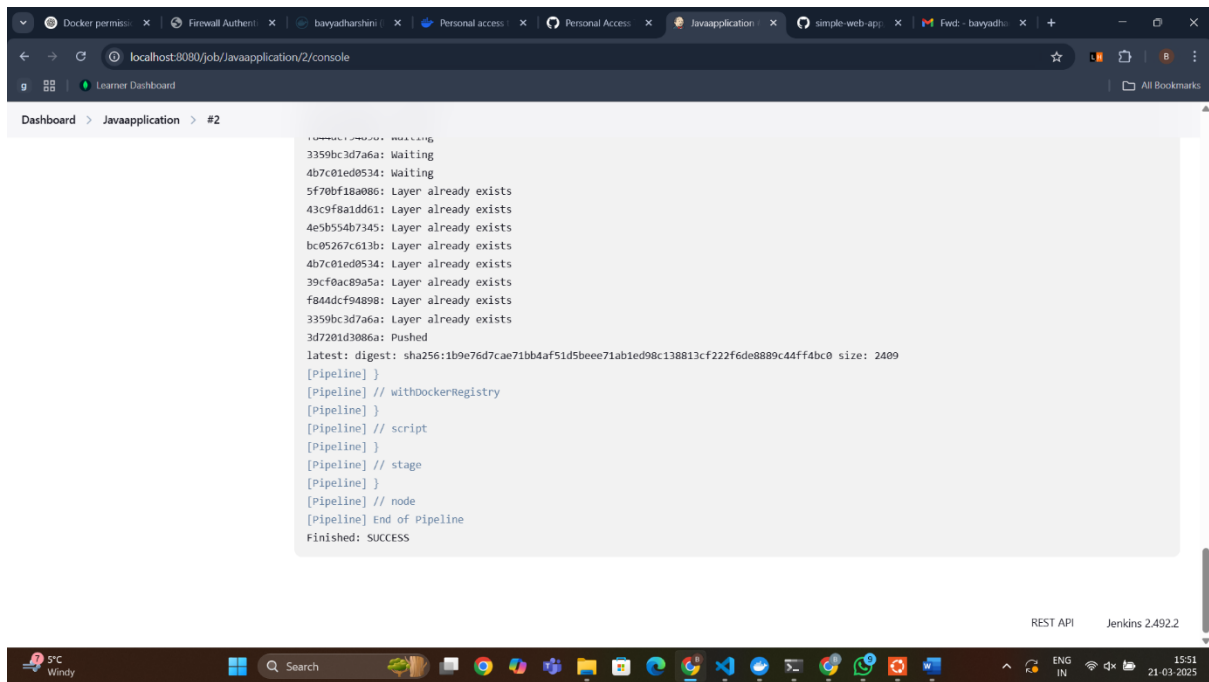
sudo vi config



JavaApplication Success Output

		SCM Checkout	Build	Build Docker Image	Push to Docker Hub
Average stage times:		2s	2s	389ms	11s
#6	15:48 No Changes				
#5	15:47 No Changes	752ms	3s	498ms	16s
#4	15:38 No Changes	784ms	3s	503ms	17s

Stage View of Output



Output