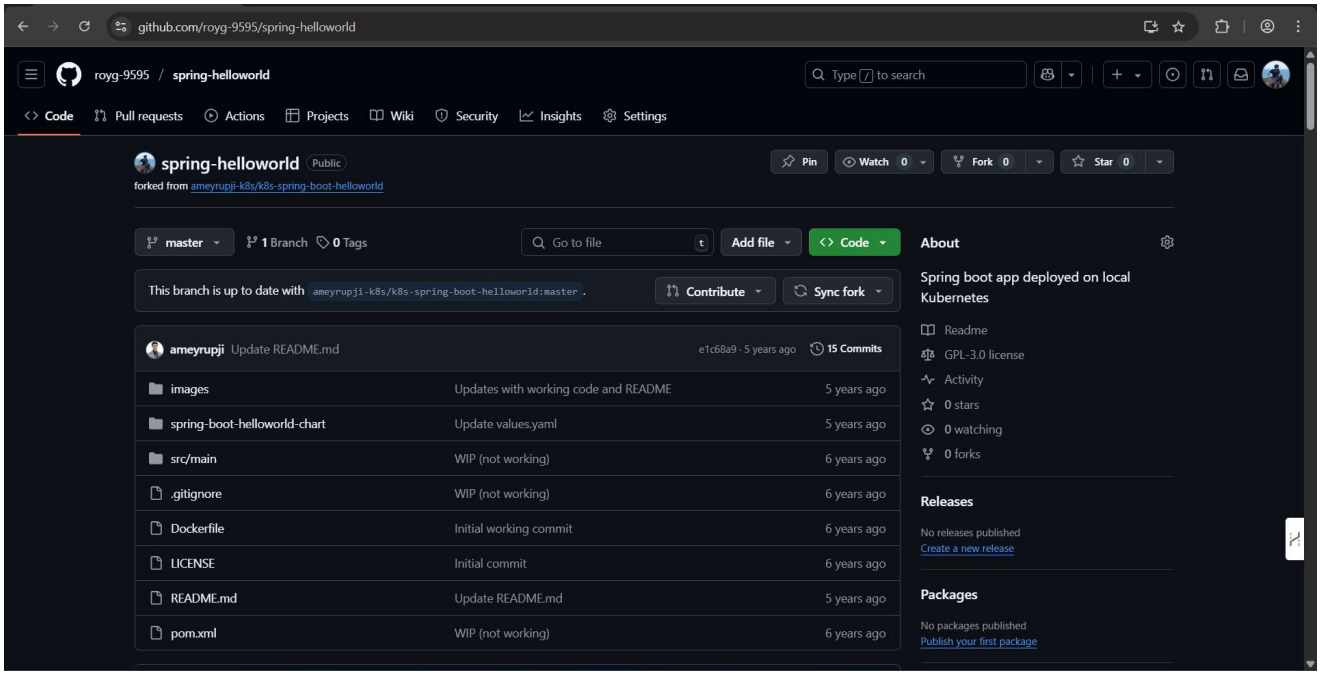
Jenkins-Docker-Kubernetes Assessment

Task: 1 Fork a Java Repository

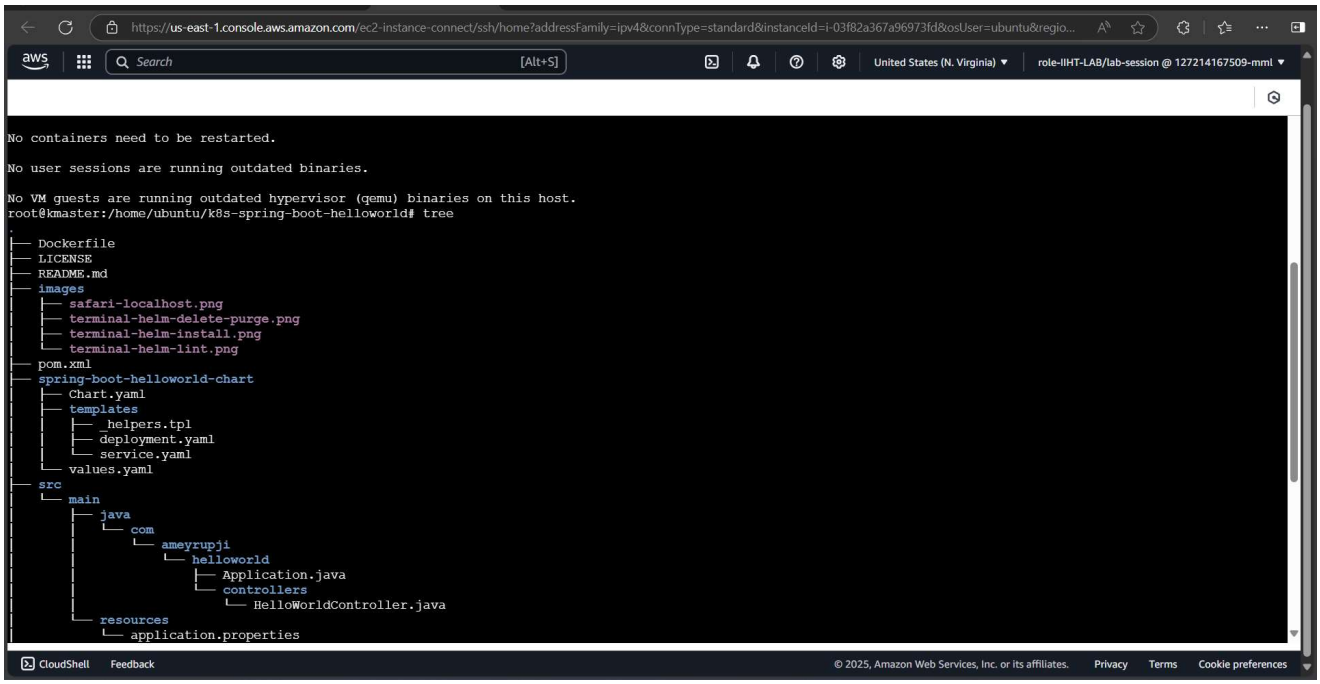
I have forked a GitHub repo from the source https://github.com/ameyrupji-k8s/k8s-spring-boot-helloworld to my GitHub account.

Following is my GitHub repo url:

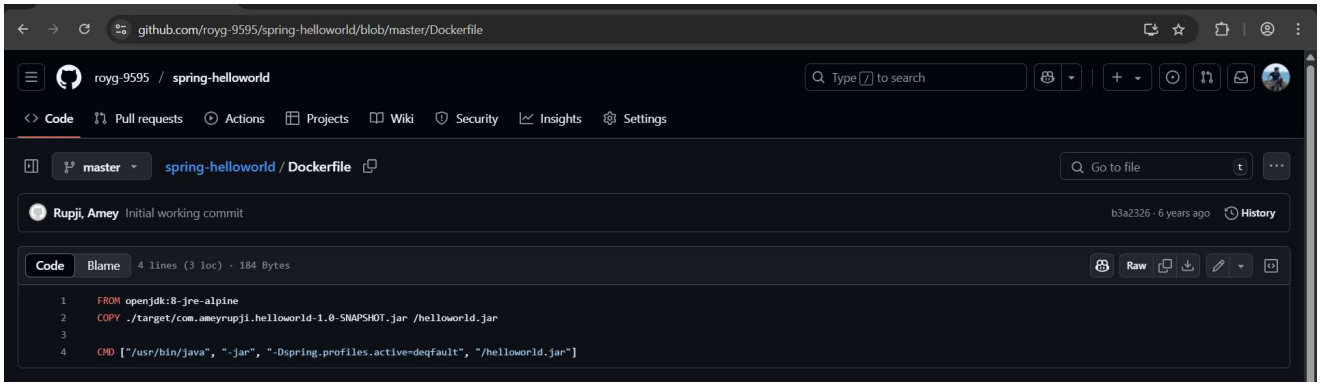
https://github.com/royg-9595/spring-helloworld



Cloned it into my EC2 machine



Task 2: Define a Dockerfile



Docker File:

FROM openjdk:8-jre-alpine

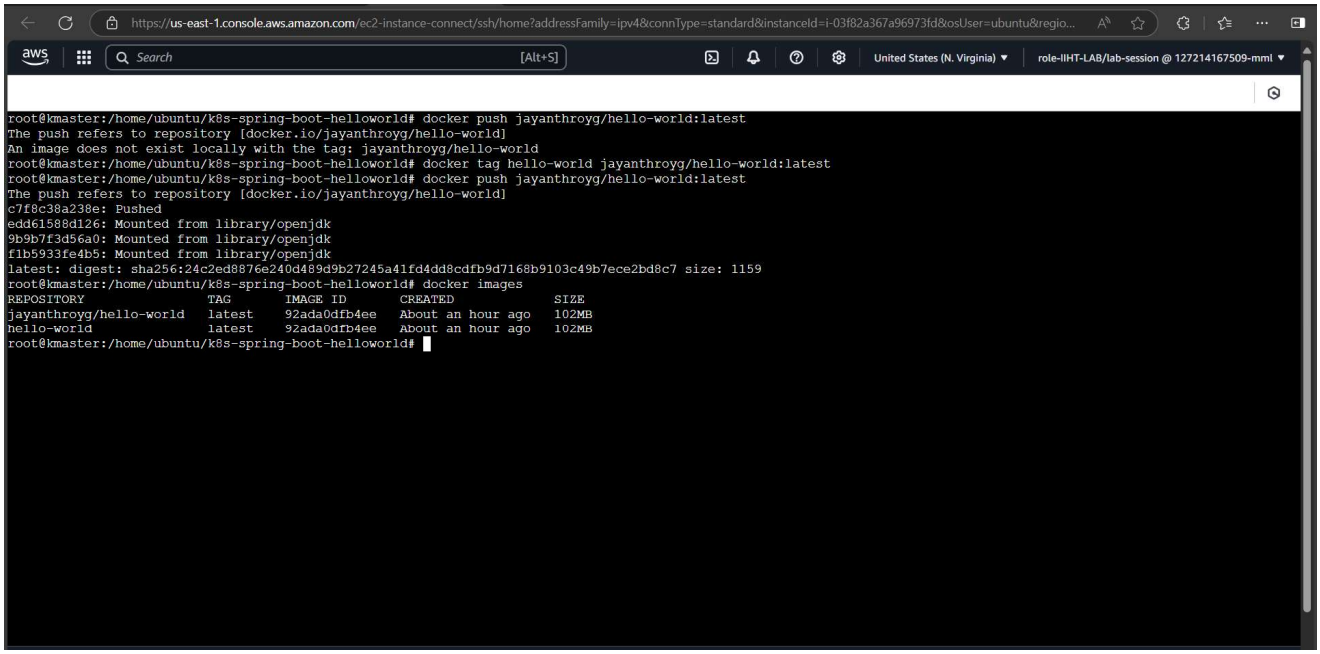
COPY ./target/com.ameyrupji.helloworld-1.0-SNAPSHOT.jar /helloworld.jar

CMD ["/usr/bin/java", "-jar", "-Dspring.profiles.active=deqfault", "/helloworld.jar"]

Task 3 : Build and Push Docker Image

Installed Docker in my EC2 instance

And now I have created the image using Dockerfile above



Pushed the docker image to my docker hub and following is the link

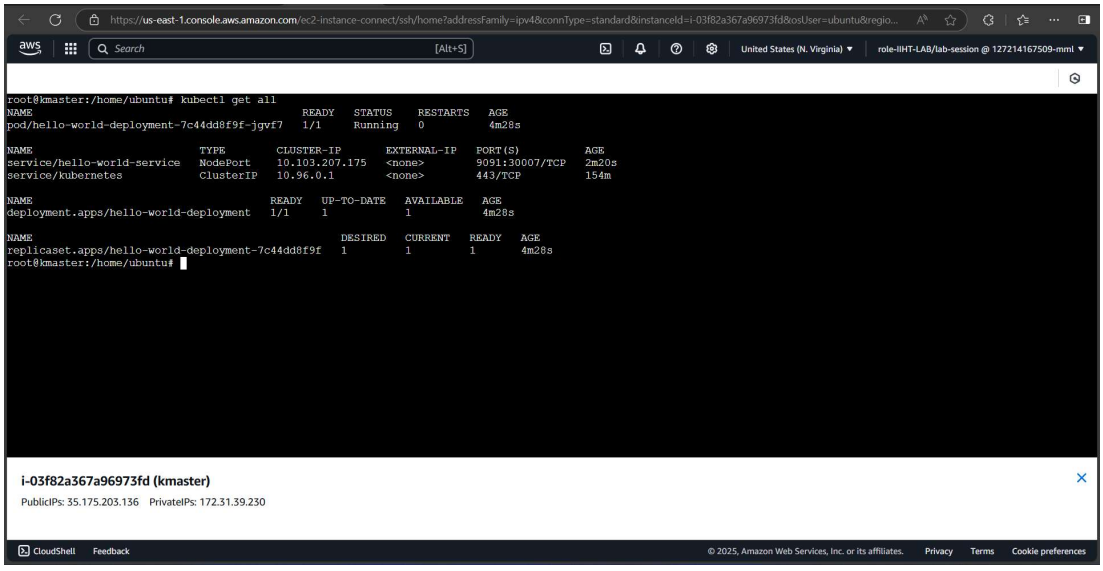
https://hub.docker.com/repository/docker/jayanthroyg/hello-world

Task 4: Create a Kubernetes Deployment

Deploy.yaml

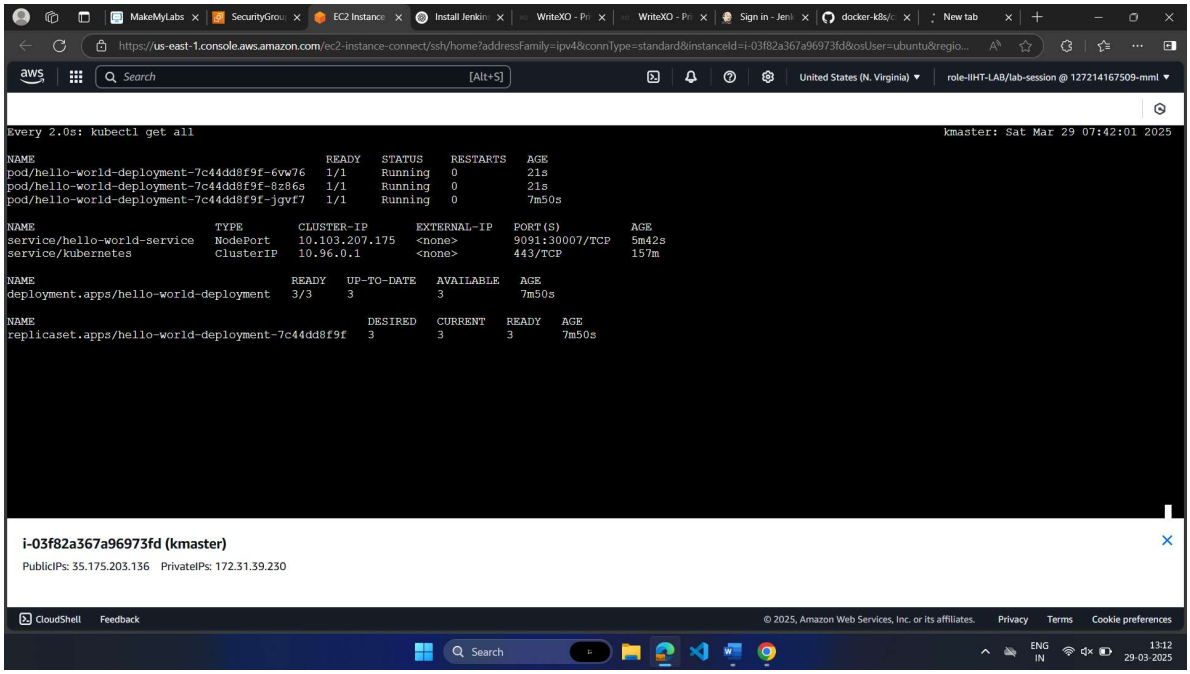
```
apiVersion: apps/v1
kind: Deployment
metadata:
 annotations:
  deployment.kubernetes.io/revision: "1"
 creationTimestamp: "2025-03-29T07:08:42Z"
 generation: 1
 labels:
  app: hello-world-deployment
 name: hello-world-deployment
 namespace: default
 resourceVersion: "10340"
 uid: 27b8513a-60a7-4c0f-8891-2d25b5305059
spec:
 progressDeadlineSeconds: 600
 replicas: 1
 revisionHistoryLimit: 10
 selector:
  matchLabels:
   app: hello-world-deployment
 strategy:
  rollingUpdate:
   maxSurge: 25%
   maxUnavailable: 25%
  type: RollingUpdate
 template:
  metadata:
   creationTimestamp: null
   labels:
    app: hello-world-deployment
  spec:
   containers:
   - image: jayanthroyg/hello-world
    imagePullPolicy: Always
    name: hello-world
    ports:
    - containerPort: 9091  # Update the containerPort to 9091
     protocol: TCP
    resources: {}
```

```
      terminationMessagePath: /dev/termination-log
      terminationMessagePolicy: File
    dnsPolicy: ClusterFirst
    restartPolicy: Always
    schedulerName: default-scheduler
    securityContext: {}
    terminationGracePeriodSeconds: 30
status:
 availableReplicas: 1
 conditions:
 - lastTransitionTime: "2025-03-29T07:08:47Z"
   lastUpdateTime: "2025-03-29T07:08:47Z"
   message: Deployment has minimum availability.
   reason: MinimumReplicasAvailable
   status: "True"
   type: Available
 - lastTransitionTime: "2025-03-29T07:08:42Z"
   lastUpdateTime: "2025-03-29T07:08:47Z"
   message: ReplicaSet "hello-world-deployment-5cd6fb998d" has successfully
progressed.
   reason: NewReplicaSetAvailable
   status: "True"
   type: Progressing
 observedGeneration: 1
 readyReplicas: 1
 replicas: 1
 updatedReplicas: 1
```

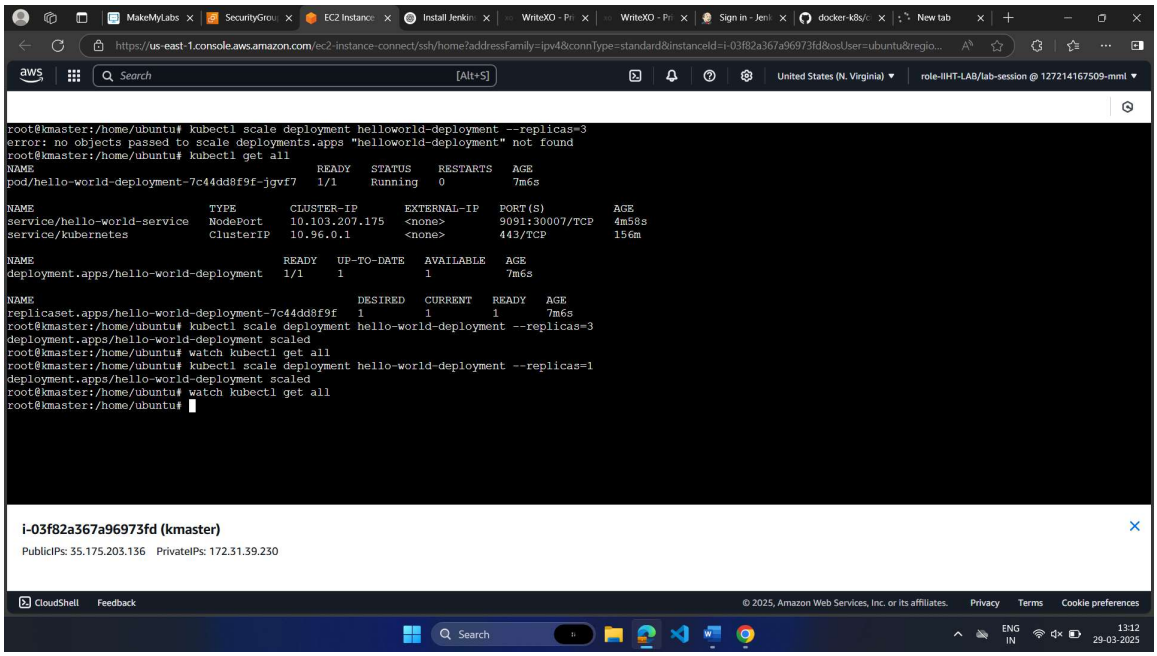


Scaled the application replicas to 3

Below is the screenshot



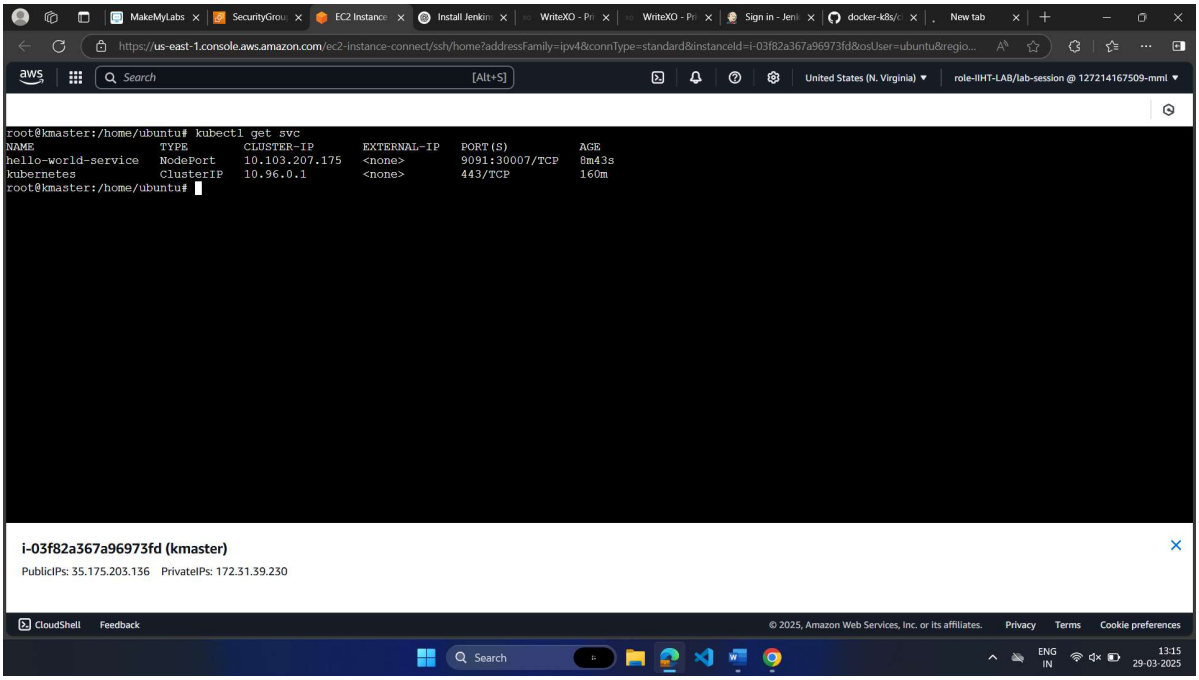
Scaled down to 1 replicas again, below is the screenshot



Task 6: Expose the Service Using NodePort

```
apiVersion: v1
kind: Service
metadata:
  name: hello-world-service
  namespace: default
spec:
  type: NodePort
  selector:
```

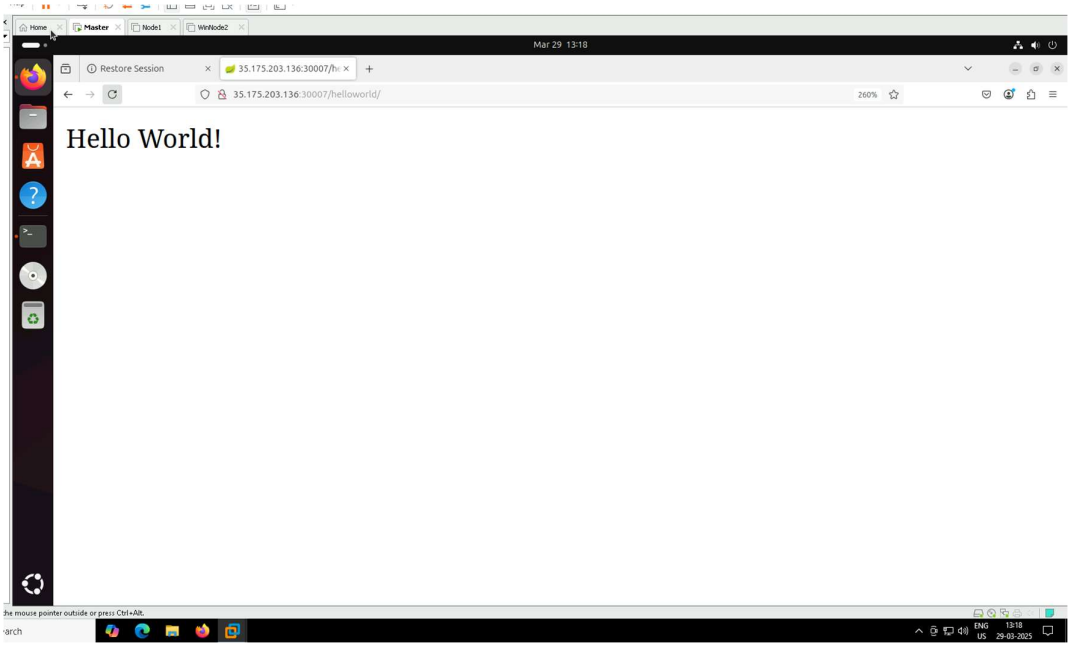
app: hello-world-deployment # This matches the label in your deployment YAML
ports:
- protocol: TCP
  port: 9091   # The internal port the service will listen on (match containerPort)
  targetPort: 9091 # The target port in the container
  nodePort: 30007   # The external port on which the service will be exposed



http://35.175.203.136:30007/helloworld/

The above is the application which returns Helloworld! output

Output



Task 7: Automate Deployment Using Jenkins