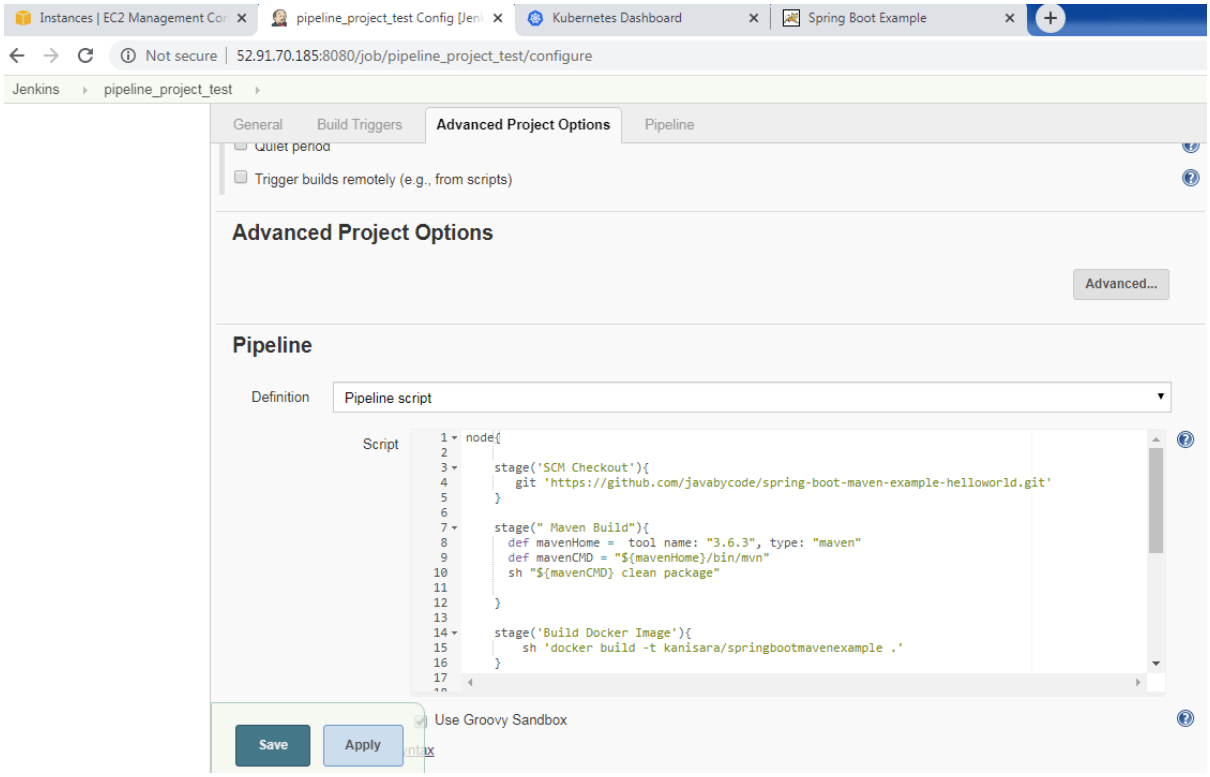
Step 1:

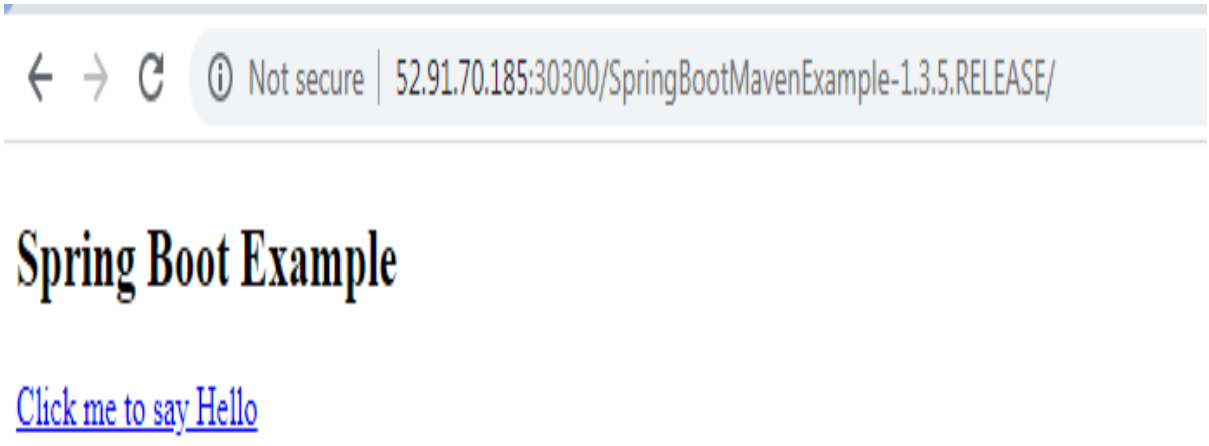
---------- Pre-request software Installation updates .

```
[ec2-user@ip-172-31-88-127 ~]$ minikube version
minikube version: v1.6.2
commit: 54f28ac5d3a815d1196cd5d57d707439ee4bb392
[ec2-user@ip-172-31-88-127 ~]$ kubectl get nodes
NAME       STATUS   ROLES    AGE   VERSION
minikube   Ready    master   13m   v1.17.0
[ec2-user@ip-172-31-88-127 ~]$ kubectl get pod --all-
namespaces
NAMESPACE     NAME                                   READY
STATUS    RESTARTS   AGE
kube-system   coredns-6955765f44-f28qg               1/1
Running   0          13m
kube-system   coredns-6955765f44-kd76n               1/1
Running   0          13m
kube-system   etcd-minikube                          1/1
Running   0          13m
kube-system   kube-addon-manager-minikube            1/1
Running   0          13m
kube-system   kube-apiserver-minikube                1/1
Running   0          13m
kube-system   kube-controller-manager-minikube       1/1
Running   0          13m
kube-system   kube-proxy-h56pz                       1/1
Running   0          13m
kube-system   kube-scheduler-minikube                1/1
Running   0          13m
kube-system   storage-provisioner                    1/1
Running   0          13m
[ec2-user@ip-172-31-88-127 ~]$ docker version
Client:
 Version:           18.09.9-ce
 API version:       1.39
 Go version:        go1.10.3
 Git commit:        039a7df
 Built:             Fri Nov  1 19:26:49 2019
 OS/Arch:           linux/amd64
 Experimental:      false

Server:
 Engine:
  Version:          18.09.9-ce
  API version:      1.39 (minimum version 1.12)
  Go version:       go1.10.3
  Git commit:       039a7df
  Built:            Fri Nov  1 19:28:24 2019
  OS/Arch:          linux/amd64
  Experimental:     false
[ec2-user@ip-172-31-88-127 ~]$ java -version
java version "1.8.0_131"
Java(TM) SE Runtime Environment (build 1.8.0_131-b11)
Java HotSpot(TM) 64-Bit Server VM (build 25.131-b11, mixed
mode)
[ec2-user@ip-172-31-88-127 ~]$ git version
```

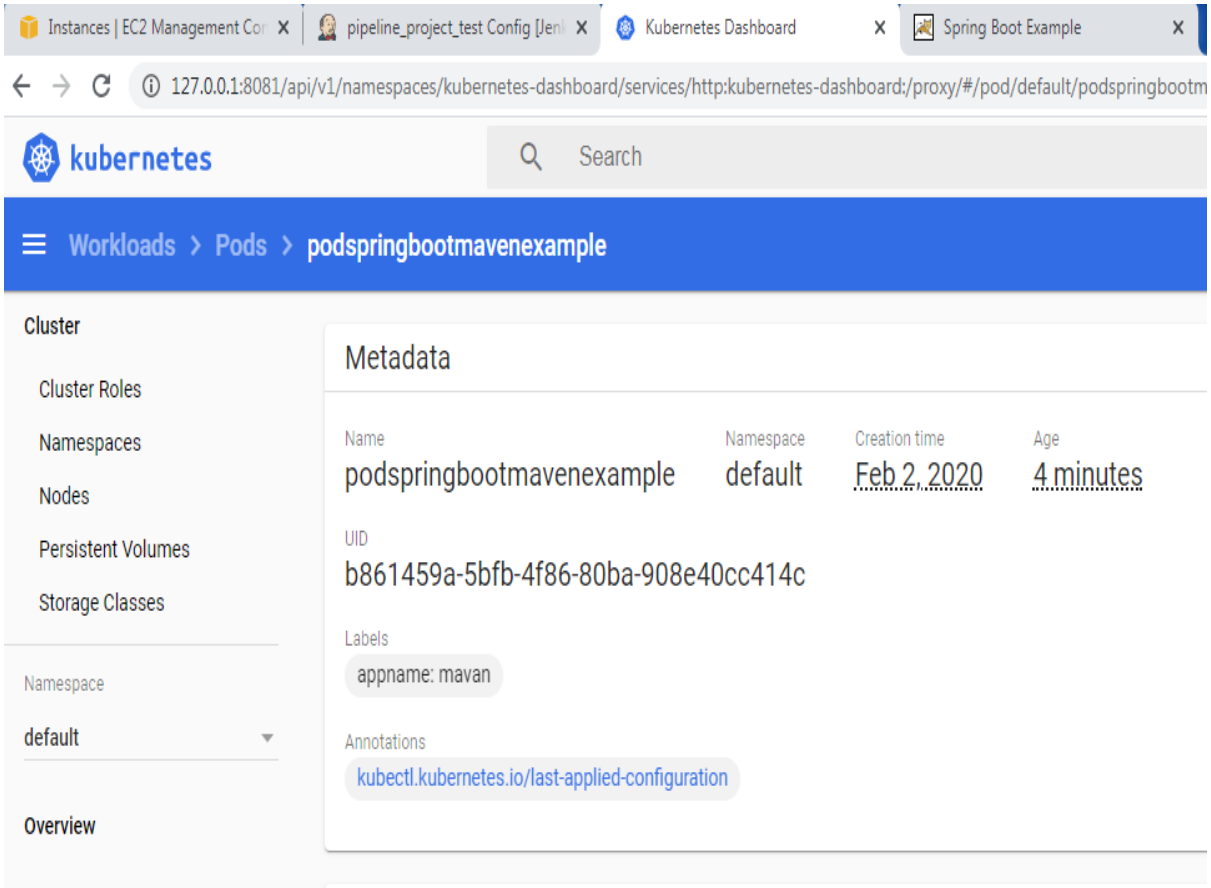
```
git version 2.23.1
[ec2-user@ip-172-31-88-127 ~]$ cd /var/lib/jenkins
[ec2-user@ip-172-31-88-127 jenkins]$ pwd
/var/lib/jenkins
[ec2-user@ip-172-31-88-127 jenkins]$

[ec2-user@ip-172-31-88-127 jenkins]$ sudo systemctl status
jenkins
● jenkins.service - LSB: Jenkins Automation Server
   Loaded: loaded (/etc/rc.d/init.d/jenkins; bad; vendor
preset: disabled)
   Active: active (running) since Mon 2020-02-03 07:29:56 UTC;
9s ago
     Docs: man:systemd-sysv-generator(8)
  Process: 31120 ExecStart=/etc/rc.d/init.d/jenkins start
(code=exited, status=0/SUCCESS)
    Tasks: 41
   Memory: 463.1M
   CGroup: /system.slice/jenkins.service
           └─31172 /etc/alternatives/java -
Dcom.sun.akuma.Daemon=daemonized -Djava.awt.headless=true -
DJENKINS_HOME=/var/lib/jenkin...

Feb 03 07:29:55 ip-172-31-88-127.ec2.internal systemd[1]:
Starting LSB: Jenkins Automation Server...
Feb 03 07:29:55 ip-172-31-88-127.ec2.internal runuser[31154]:
pam_unix(runuser:session): session opened for user jenkins by
(uid=0)
Feb 03 07:29:56 ip-172-31-88-127.ec2.internal runuser[31154]:
pam_unix(runuser:session): session closed for user jenkins
Feb 03 07:29:56 ip-172-31-88-127.ec2.internal jenkins[31120]:
Starting Jenkins [  OK  ]
Feb 03 07:29:56 ip-172-31-88-127.ec2.internal systemd[1]:
Started LSB: Jenkins Automation Serve

[ec2-user@ip-172-31-88-127 pipeline_project_test]$ sudo
service sonar status
SonarQube is running (32422).
```




Finally application was able access now. Please see the screenshot.




**Kubernetes console management**




**Jenkins Pipeline scripts**

```
node{
   stage('SCM Checkout'){
      git 'https://github.com/kanisara/spring-boot-maven-example-helloworld.git'
   }
   stage(" Maven Build"){
      def mavenHome =  tool name: "3.6.3", type: "maven"
      def mavenCMD = "${mavenHome}/bin/mvn"
      sh "${mavenCMD} clean package"

   }
   stage(" Quality control check"){
      def mavenHome =  tool name: "3.6.3", type: "maven"
      def mavenCMD = "${mavenHome}/bin/mvn"
      sh "${mavenCMD} sonar:sonar"

   }
stage(" deploy Artifact into remote repo"){
      def mavenHome =  tool name: "3.6.3", type: "maven"
      def mavenCMD = "${mavenHome}/bin/mvn"
```

```
    sh "${mavenCMD} deploy"
      }

   stage('Build Docker Image'){
      sh 'docker build -t kanisara/springbootmavenexample .'
   }

    stage('Push Docker Image'){
      withCredentials([string(credentialsId: 'Docker_Hub_Pwd', variable: 'Docker_Hub_Pwd')]) {
        sh "docker login -u kanisara -p ${Docker_Hub_Pwd}"
      }
      sh 'docker push kanisara/springbootmavenexample'
    }

    stage("Deploy To Kuberates Cluster"){
      sh 'kubectl apply -f springbootmavenexample.yml'
    }

}
```

********** Docker file *****

```
FROM tomcat:8.0.20-jre8

COPY target/SpringBootMavenExample-1.3.5.RELEASE.war
/usr/local/tomcat/webapps/SpringBootMavenExample-1.3.5.RELEASE.war
```

********** springbootmavenexample.yml

```
apiVersion: v1

kind: Pod

metadata:

  name: podspringbootmavenexample

  labels:

    appname: mavan

spec:

  containers:

    - name: springbootmavenexample-container
```

```
    image: kanisara/springbootmavenexample
    ports:
    - containerPort: 8080
...
---
apiVersion: v1
kind: Service
metadata:
 name: httpdnodeportservice
spec:
 selector:
   appname: mavan              #-- > Label of pod
 type: NodePort
 ports:
 - port: 80                    #--> Service Port
   targetPort: 8080            #--> Container Port
   nodePort: 30300
```