DevOps

Day 6

Date: 22.03.2025

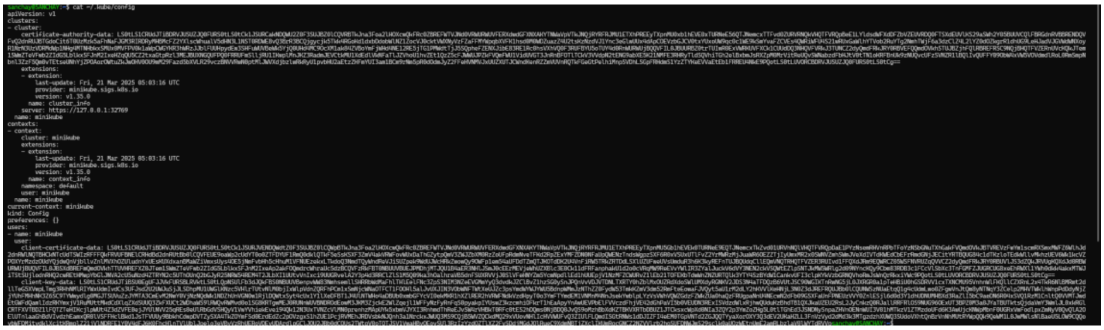
Topics Covered: Java Application Minikube deployment

Jenkins and Minikube Deployment

Deployment of the Docker Image with Kubernetes and Minikube in Jenkins

- Push the Docker Image to Docker Hub from Jenkins (Testing)
- Install Kubernetes Cloud credentials
- Create new credentials secret file from deployment.yml from GitHub
- Install Kubernetes and stages-view plugins
- Configure the script
- Deploy in Minikube

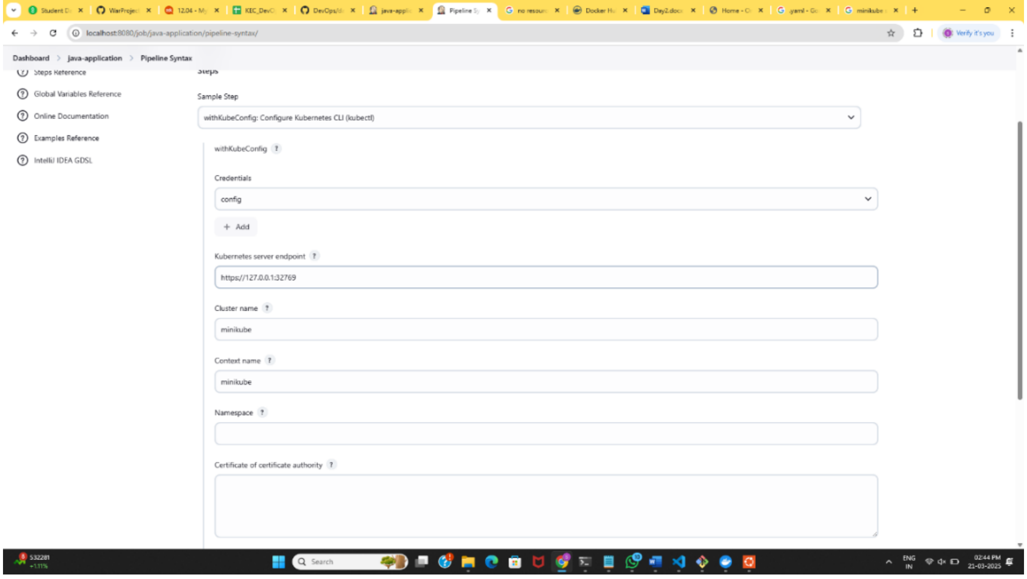
Configuring the Kubernetes config file by taking data from ca.crt



```
sanchay@SANCHAY:~$ cat ~/.kube/config
apiVersion: v1
clusters:
- cluster:
    certificate-authority-data: LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0t [illegible]
    extensions:
    - extension:
        last-update: Fri, 21 Mar 2025 05:03:16 UTC
        provider: minikube.sigs.k8s.io
        version: v1.35.0
      name: cluster_info
    server: https://127.0.0.1:32769
  name: minikube
contexts:
- context:
    cluster: minikube
    extensions:
    - extension:
        last-update: Fri, 21 Mar 2025 05:03:16 UTC
        provider: minikube.sigs.k8s.io
        version: v1.35.0
      name: context_info
    namespace: default
    user: minikube
  name: minikube
current-context: minikube
kind: Config
preferences: {}
users:
- name: minikube
  user:
    client-certificate-data: LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0t [illegible]
    client-key-data: LS0tLS1CRUdJTiBSU0EgUFJJVkFURSBLRVktLS0t [illegible]
```

Config data in config

Generating the pipline syntax for Kubernetes deployment



localhost:8080/job/java-application/pipeline-syntax/

Dashboard > java-application > Pipeline Syntax

Steps

Sample Step: withKubeConfig: Configure Kubernetes CLI (kubectl)

withKubeConfig

Credentials: config

Kubernetes server endpoint: https://127.0.0.1:32769

Cluster name: minikube

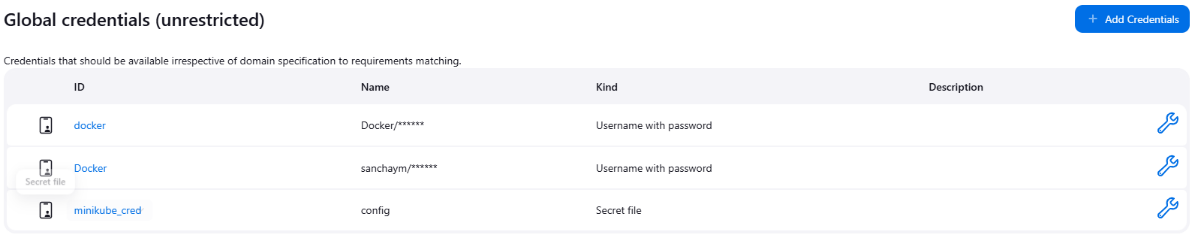
Context name: minikube

Namespace:

Certificate of certificate authority:

Pipeline syntax generation

Creating global credentials for deployment.yml



Global credentials (unrestricted)

Credentials that should be available irrespective of domain specification to requirements matching.

ID	Name	Kind	Description
docker	Docker/******	Username with password	
Docker	sanchaym/******	Username with password	
minikube_cred	config	Secret file	

Global minikube credentials

Deployment.yml

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-deploy
  labels:
    name: my-deploy
spec:
  replicas: 1
  selector:
    matchLabels:
      apptype: web-backend
  strategy:
    type: RollingUpdate
  template:
    metadata:
      labels:
        apptype: web-backend
    spec:
      containers:
      - name: my-web
        image: sanchaym/simplewebapp:latest
        ports:
        - containerPort: 9001
```

```
---
apiVersion: v1
kind: Service
metadata:
  name: my-service
  labels:
    app: my-service
spec:
  type: NodePort
  ports:
    - port: 9001
      targetPort: 8080
      nodePort: 30005
  selector:
    apptype: web-backend
```

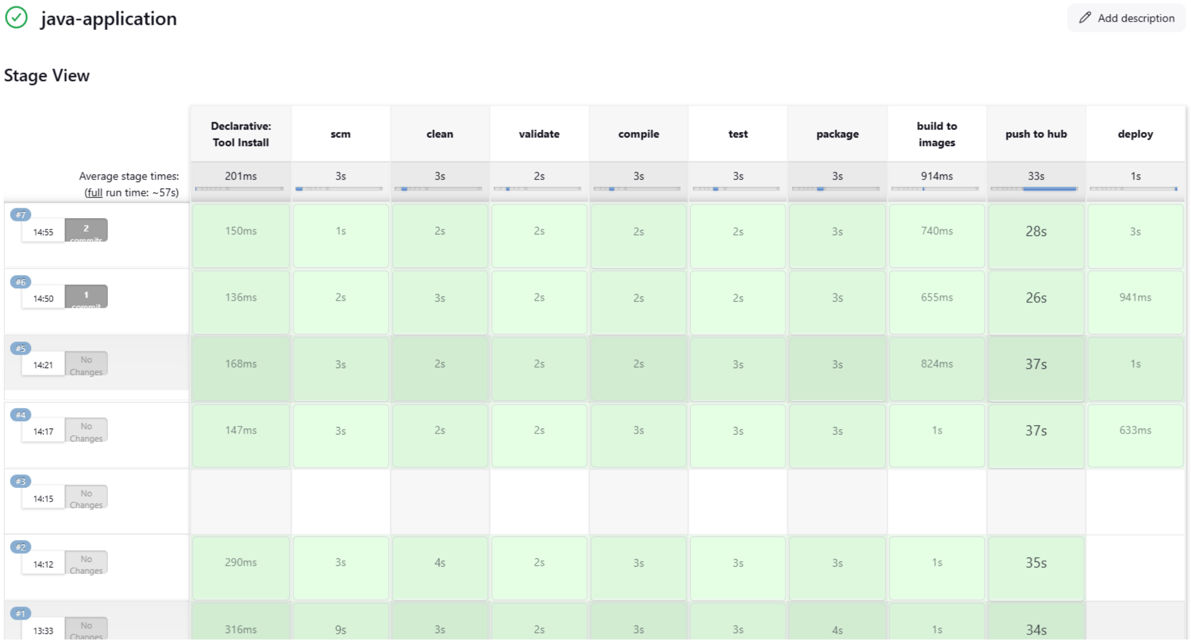
Script:

```
pipeline {
    agent any
tools {maven 'mvn'}
    stages {
        stage('scm') {
            steps {
        git 'https://github.com/Sanchay1054/WarProject.git'
            }
}
        stage('clean') {
            steps {
               sh "mvn clean"
           }
}
stage('validate') {
        steps {
          sh "mvn validate"
         }
}
stage('compile') {
        steps {
          sh "mvn compile"
}
}
stage('test') {
        steps {
          sh "mvn test"
}
}
stage('package') {
        steps {
          sh "mvn package"
}
}
stage('build to images') {
        steps {
          script{
            sh 'docker build -t sanchaym/simplewebapp .'
          }
   }
}
stage('push to hub') {
        steps {
          script{
           withDockerRegistry(credentialsId: 'Docker', url: 'https://index.docker.io/v1/') {
            sh 'docker push sanchaym/simplewebapp'
          }
        }
          }


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

Output:

Docker Image is deployed with minikube



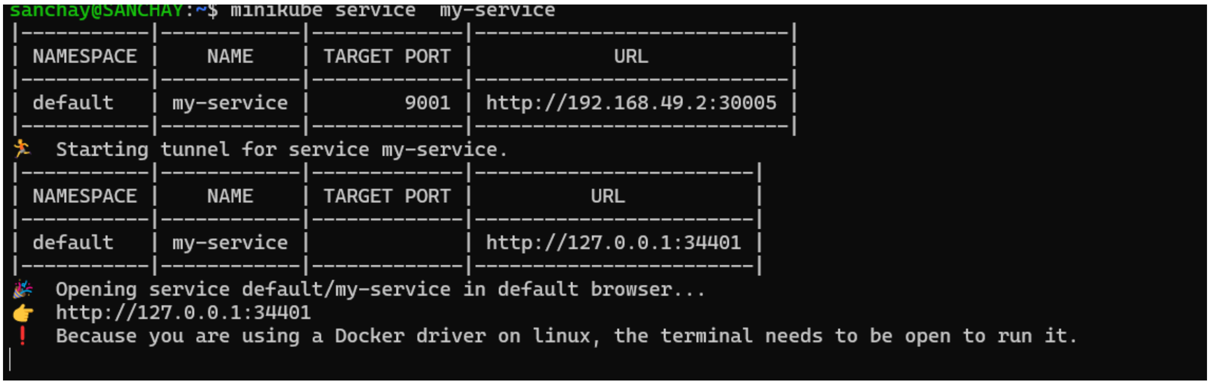
java-application

Stage View

Average stage times: (full run time: ~57s)	Declarative: Tool Install	scm	clean	validate	compile	test	package	build to images	push to hub	deploy
	201ms	3s	3s	2s	3s	3s	3s	914ms	33s	1s
#7 14:55 2 commits	150ms	1s	2s	2s	2s	2s	3s	740ms	28s	3s
#6 14:50 1 commit	136ms	2s	3s	2s	2s	2s	3s	655ms	26s	941ms
#5 14:21 No Changes	168ms	3s	2s	2s	2s	3s	3s	824ms	37s	1s
#4 14:17 No Changes	147ms	3s	2s	2s	3s	3s	3s	1s	37s	633ms
#3 14:15 No Changes										
#2 14:12 No Changes	290ms	3s	4s	2s	3s	3s	3s	1s	35s	
#1 13:33 No Changes	316ms	9s	3s	2s	3s	3s	4s	1s	34s	

Pipeline stages upto deploy

Getting the service my-service port number

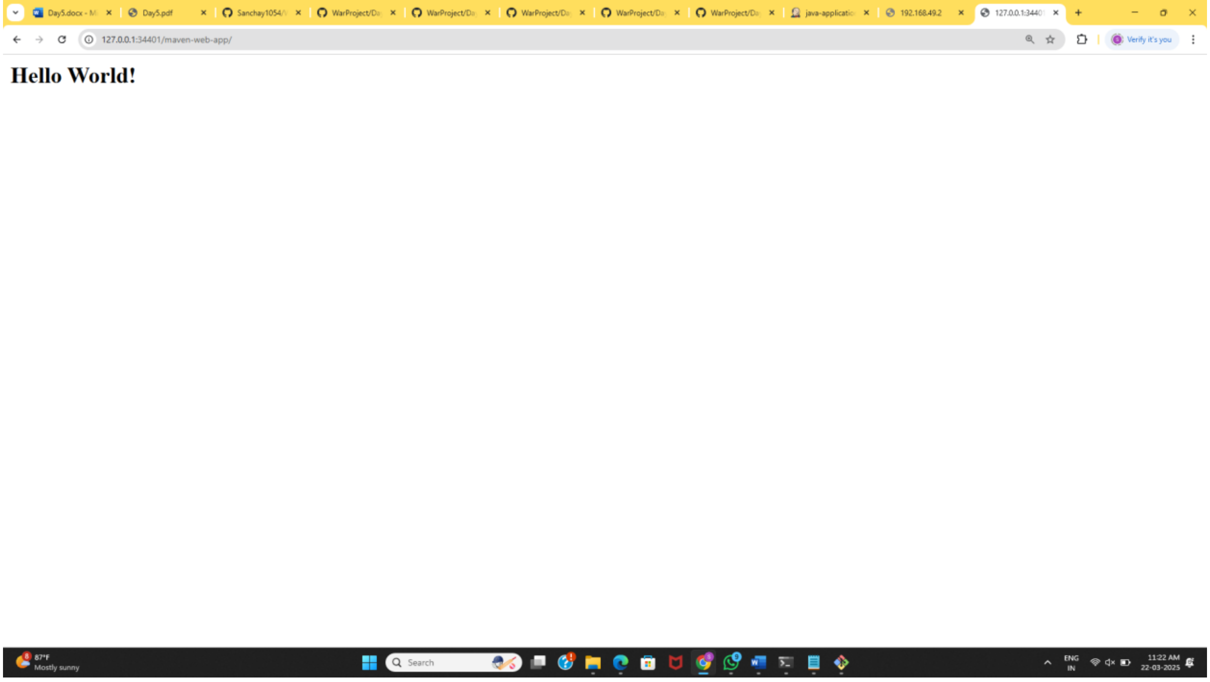


```
sanchay@SANCHAY:~$ minikube service  my-service
|-----------|------------|-------------|---------------------------|
| NAMESPACE |    NAME    | TARGET PORT |            URL            |
|-----------|------------|-------------|---------------------------|
| default   | my-service |        9001 | http://192.168.49.2:30005 |
|-----------|------------|-------------|---------------------------|
🏃  Starting tunnel for service my-service.
|-----------|------------|-------------|------------------------|
| NAMESPACE |    NAME    | TARGET PORT |          URL           |
|-----------|------------|-------------|------------------------|
| default   | my-service |             | http://127.0.0.1:34401 |
|-----------|------------|-------------|------------------------|
🎉  Opening service default/my-service in default browser...
👉  http://127.0.0.1:34401
❗  Because you are using a Docker driver on linux, the terminal needs to be open to run it.
```

service

output:

Deployment in minikube



127.0.0.1:34401/maven-web-app/

Hello World!

output