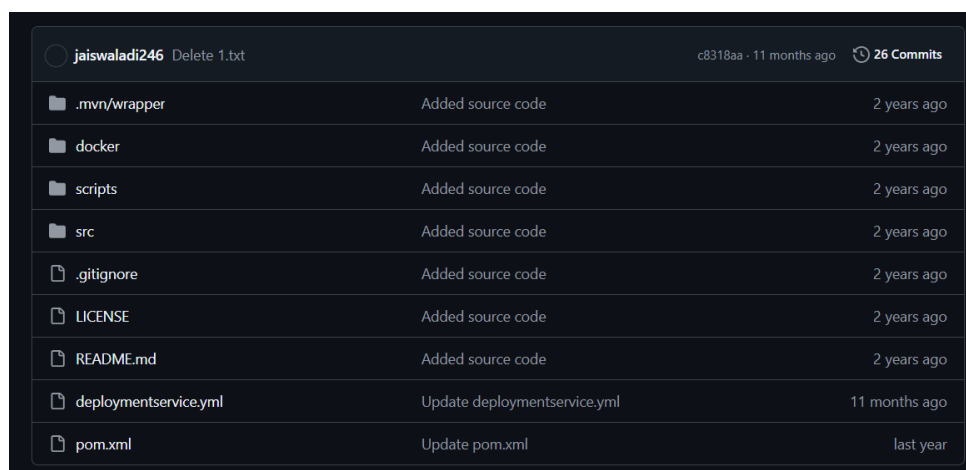


Task 5

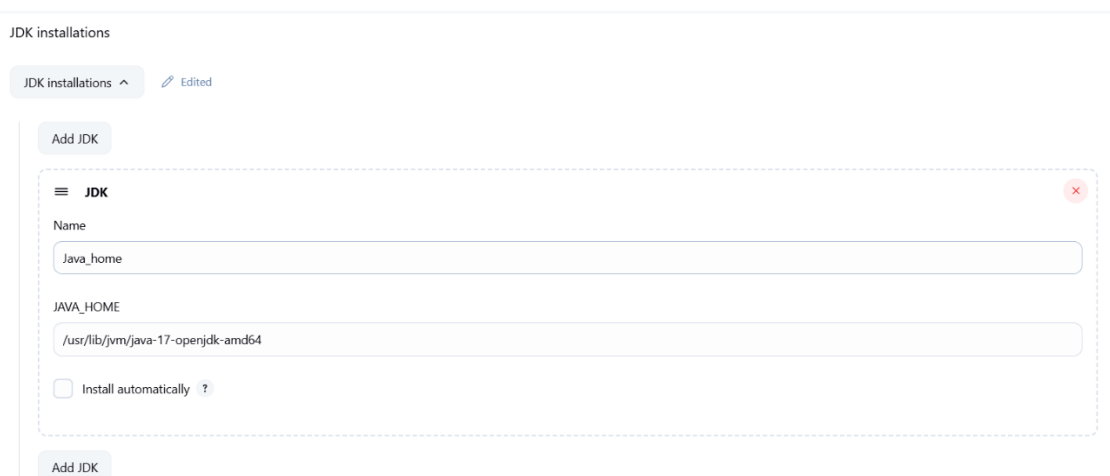
Create a Jenkins freestyle job to pull an HTML website from GitHub, build and deploy it on Tomcat using Maven, and manage the deployment with Kubernetes.

1. Preparing the Repository.



jaiswaladi246	Delete 1.txt	c8318aa · 11 months ago	🕒 26 Commits
📁 .mvn/wrapper	Added source code	2 years ago	
📁 docker	Added source code	2 years ago	
📁 scripts	Added source code	2 years ago	
📁 src	Added source code	2 years ago	
📄 .gitignore	Added source code	2 years ago	
📄 LICENSE	Added source code	2 years ago	
📄 README.md	Added source code	2 years ago	
📄 deployment-service.yml	Update deployment-service.yml	11 months ago	
📄 pom.xml	Update pom.xml	last year	

2. Adding JDK and Maven Path to Jenkins.



JDK installations

JDK installations ^ Edited

Add JDK

Name	JAVA_HOME	Install automatically ?
Java_home	/usr/lib/jvm/java-17-openjdk-amd64	<input checked="" type="checkbox"/>

Add JDK

Maven installations

Maven installations ^ Edited

Add Maven

Maven

Name

mvn

MAVEN_HOME

/mnt/c/Program Files/apache-maven-3.9.9-bin/apache-maven-3.9.9

☐ Install automatically ?

Add Maven

3. Adding GitHub Repository Link.

Source Code Management

Connect and manage your code repository to automatically pull the latest code for your builds.

☐ None

☒ Git ?

Repositories ?

Repository URL ?

https://github.com/balachandrankk/Task5_ekart.git

Credentials ?

- none -

+ Add

Advanced ▾

4. Choosing **Invoke top-level Maven targets** in build steps and Selecting the Maven and adding command to quickly build an artifact without executing tests.

Build Steps

Automate your build process with ordered tasks like code compilation, testing, and deployment.

Invoke top-level Maven targets ?

Maven Version

mvn

Goals

clean package -DskipTests

Advanced ▾

5. Executing Shell commands.

Execute shell ?

Command

See [the list of available environment variables](#)

```
docker build -t ekartimage -f docker/Dockerfile .
docker login -u balachandran2005 -p 19-May-05
docker tag ekartimage balachandran2005/task2:ekart
docker push balachandran2005/task2:ekart
kubectl create deployment ekartdep2 --image=balachandran2005/task2:ekart --port=8070
kubectl expose deployment ekartdep2 --type=NodePort --port=8070
kubectl get pods
```

6. Successful execution of the above commands and generation of .war file for Tomcat deployment.

✓ Console Output

Download Copy View as plain text

```
Started by user Balachandran
Running as SYSTEM
Building in workspace /var/lib/jenkins/workspace/task5
The recommended git tool is: NONE
No credentials specified
> git rev-parse --resolve-git-dir /var/lib/jenkins/workspace/task5/.git # timeout=10
Fetching changes from the remote Git repository
> git config remote.origin.url https://github.com/balachandrakk/Task5_ekart.git # timeout=10
Fetching upstream changes from https://github.com/balachandrakk/Task5_ekart.git
> git --version # timeout=10
> git --version # 'git version 2.43.0'
> git fetch --tags --force --progress -- https://github.com/balachandrakk/Task5_ekart.git +refs/heads/*:refs/remotes/origin/* # timeout=10
> git rev-parse refs/remotes/origin/main^{commit} # timeout=10
Checking out Revision c8318aa375b44364808f0017e9d61a75bef38f25 (refs/remotes/origin/main)
> git config core.sparsecheckout # timeout=10
> git checkout -f c8318aa375b44364808f0017e9d61a75bef38f25 # timeout=10
Commit message: "Delete 1.txt"
> git rev-list --no-walk c8318aa375b44364808f0017e9d61a75bef38f25 # timeout=10
[task5] $ "/mnt/c/Program Files/apache-maven-3.9.9-bin/apache-maven-3.9.9/bin/mvn" clean package -DskipTests
[INFO] Scanning for projects...
[INFO]
```

```
Login Succeeded
+ docker tag ekartimage balachandran2005/task2:ekart
+ docker push balachandran2005/task2:ekart
The push refers to repository [docker.io/balachandran2005/task2]
f34a9819c48c: Preparing
01bca7cb4826: Preparing
a8cc3712c14a: Preparing
cd7100a72410: Preparing
cd7100a72410: Layer already exists
a8cc3712c14a: Layer already exists
01bca7cb4826: Layer already exists
f34a9819c48c: Pushed
ekart: digest: sha256:38cc4f79c3d82162cdeae314a96da8147ed0f6f2843e8f2ad0016dfc96d19e1 size: 1159
+ kubectl create deployment ekartdep2 --image=balachandran2005/task2:ekart --port=8070
deployment.apps/ekartdep2 created
+ kubectl expose deployment ekartdep2 --type=NodePort --port=8070
service/ekartdep2 exposed
+ kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
ekartdep2-556cfdbc97-dg7zh          0/1     Pending   0           0s
Finished: SUCCESS
```

7. Managing deployment with Kubernetes.

```
jenkins@Toxin:~$ minikube start
🐳 minikube v1.35.0 on Ubuntu 24.04 (amd64)
🌟 Using the docker driver based on existing profile
👉 Starting "minikube" primary control-plane node in "minikube" cluster
📡 Pulling base image v0.0.46 ...
🔄 Updating the running docker "minikube" container ...
🔧 Preparing Kubernetes v1.32.0 on Docker 27.4.1 ...
🔍 Verifying Kubernetes components...
   ▪ Using image gcr.io/k8s-minikube/storage-provisioner:v5
🌟 Enabled addons: default-storageclass, storage-provisioner
🏁 Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
```

```
jenkins@Toxin:~$ kubectl get svc
NAME          TYPE          CLUSTER-IP    EXTERNAL-IP    PORT(S)          AGE
ekartdep2     NodePort      10.104.178.0  <none>         8070:32623/TCP   117s
kubernetes    ClusterIP     10.96.0.1     <none>         443/TCP          22m
```

```
jenkins@Toxin:~$ kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
ekartdep2-556cfdbc97-dg72h         1/1     Running   1 (85s ago)  2m3s
jenkins@Toxin:~$ minikube service ekartdep2
-----|-----|-----|-----|
| NAMESPACE | NAME      | TARGET PORT | URL                               |
|-----|-----|-----|-----|
| default   | ekartdep2 | 8070        | http://192.168.49.2:32623      |
|-----|-----|-----|-----|
🌐 Opening service default/ekartdep2 in default browser...
👉 http://192.168.49.2:32623
jenkins@Toxin:~$ kubectl port-forward svc/ekartdep2 8070:8070
Forwarding from 127.0.0.1:8070 -> 8070
Forwarding from [::1]:8070 -> 8070
Handling connection for 8070
Handling connection for 8070
Handling connection for 8070
Handling connection for 8070
|
```

8. Final Output in URL <http://127.0.0.1:8070>

