

SETUP CONTINUES DEPLOYMENT

CREATE & PUBLISH DOCKER IMAGES

✓ Create Docker file Instructions.

- Create a new file **Dockerfile** in local repository.
- Add the following instructions

```
FROM adoptopenjdk/openjdk11

EXPOSE 8080

ENV APP_HOME /usr/src/app

COPY target/*.jar $APP_HOME/app.jar

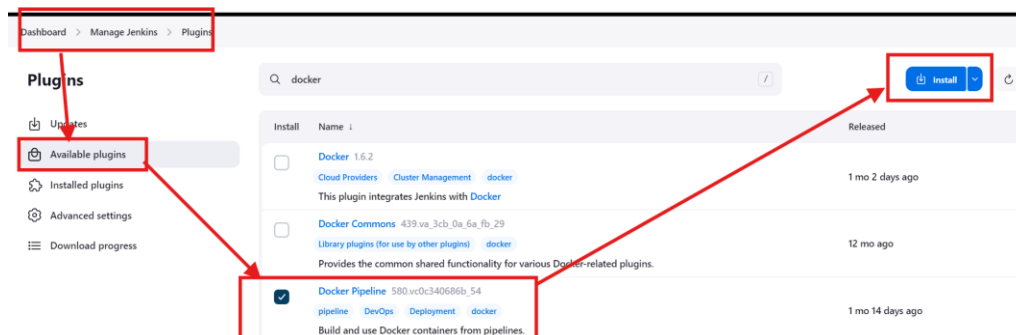
WORKDIR $APP_HOME

CMD ["java", "-jar", "app.jar"]
```

- And commit to the repository and push to GITHUB.

✓ Install Docker Plugin for Jenkins

- In Jenkins Dashboard head over to **Manage Jenkins > Plugins > Available Plugins**.
- Search for **Docker Pipeline > Install > Restart** (*if needed).



✓ Setup Docker Hub credentials

- In Jenkins Dashboard head over to **Manage Jenkins > Credentials > Global Credentials > Create Credentials**.
- Select type as **Username and Password > Enter the username and password > Give unique ID > Add**.

The screenshot shows the 'Create Credentials' form in Jenkins. The 'Kind' dropdown is set to 'Username with password'. The 'Scope' dropdown is set to 'Global (Jenkins, nodes, items, all child items, etc)'. The 'Username' field contains 'vnsharsha7999'. The 'Treat username as secret' checkbox is unchecked. The 'Password' field is masked with dots. The 'ID' field contains 'Docker-Pass'.

✓ Generate pipeline script.

- In Jenkins Dashboard head over to **New Item > Select Pipeline > Give a name > Create**
- Scroll down to the bottom of the page **Select Pipeline Syntax > Declarative Snippet Generator**.
- Select the **withDockerRepo step > Give Repo URL > Select docker credential from dropdown > Generate pipeline syntax**.

The screenshot shows the 'Declarative Snippet Generator' page in Jenkins. The 'Snippet Generator' tab is selected in the left sidebar. The 'Steps' section shows a 'Sample Step' dropdown set to 'withDockerRegistry: Sets up Docker registry endpoint'. Below this, the 'withDockerRegistry' configuration is shown with a 'Docker registry URL' field and a 'Registry credentials' dropdown set to 'vnsharsha7999/***** (Docker-Pass)'. At the bottom, there is a '+ Add' button and a 'Generate Pipeline Script' button.

- Copy the step, and add it to the pipeline.

```
stage ("CREATE AND PUSH DOCKER IMAGES"){
  steps{
    withDockerRegistry(credentialsId: 'Docker-Pass') {
      sh "docker build -t <Docker-registry-name>/App ."
      sh "docker push <Docker-registry-name> /App"
    }
  }
}
```

SETUP KUBERNETES DEPLOYMENT

✓ Create Kubernetes Manifest.

- Create a new file **deployment-service.yaml** in local repository.
- Add the following contents.

```
apiVersion: apps/v1
kind: Deployment # Kubernetes resource kind we are creating
metadata:
  name: boardgame-deployment
spec:
  selector:
    matchLabels:
      app: boardgame
  replicas: 2 # Number of replicas that will be created for this
deployment
  template:
    metadata:
      labels:
        app: boardgame
    spec:
      containers:
        - name: boardgame
          image: adijaiswal/boardgame:latest # Image that will be
used to containers in the cluster
          imagePullPolicy: Always
          ports:
            - containerPort: 8080 # The port that the container is
running on in the cluster
```

```

---

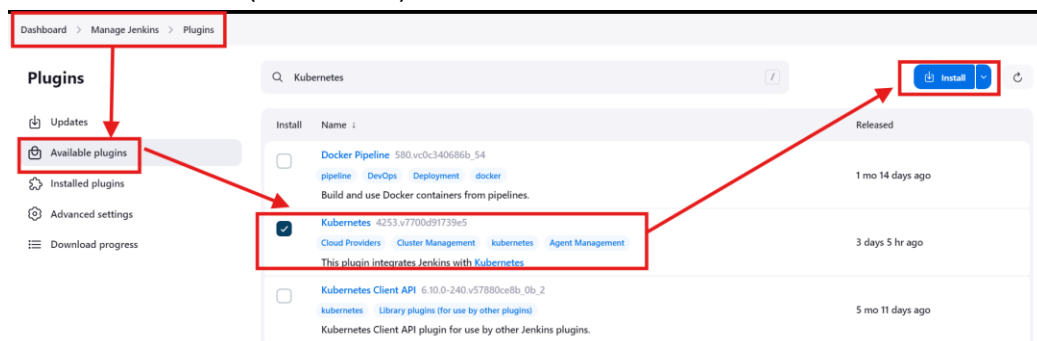
apiVersion: v1 # Kubernetes API version
kind: Service # Kubernetes resource kind we are creating
metadata: # Metadata of the resource kind we are creating
  name: boardgame-ssvc
spec:
  selector:
    app: boardgame
  ports:
    - protocol: "TCP"
      port: 8080
      targetPort: 8080
  type: LoadBalancer # type of the service.

```

- Add and commit to GITHUB.

✓ Install Kubernetes Plugin

- In Jenkins Dashboard head over to **Manage Jenkins > Plugins > Available Plugins**.
- Search for the following plugins
 - **Kubernetes**
 - **Kubernetes CLI**
 - **Kubernetes Credential**
- **Install > Restart** (*if needed).



✓ Setup Cluster credentials

- SSH to the minikube server, and use the following command to extract the cluster credentials file.

Cat .kube/config

```
ubuntu@ip-172-31-91-57:~$ cat .kube/config
apiVersion: v1
clusters:
- cluster:
  certificate-authority: /home/ubuntu/.minikube/ca.crt
  extensions:
  - extension:
    last-update: Thu, 11 Jul 2024 11:58:42 UTC
    provider: minikube.sigs.k8s.io
    version: v1.33.1
    name: cluster_info
  server: https://192.168.49.2:8443
  name: minikube
contexts:
- context:
  cluster: minikube
  extensions:
  - extension:
    last-update: Thu, 11 Jul 2024 11:58:42 UTC
    provider: minikube.sigs.k8s.io
    version: v1.33.1
    name: context_info
  namespace: default
  user: minikube
  name: minikube
current-context: minikube
kind: Config
preferences: {}
users:
- name: minikube
  user:
    client-certificate: /home/ubuntu/.minikube/profiles/minikube/client.crt
    client-key: /home/ubuntu/.minikube/profiles/minikube/client.key
```

- Get the CA file details and get the credentials.

```
Last login: Thu Jul 11 12:11:12 2024 from 40.205.84.211
ubuntu@ip-172-31-91-57:~$ cat /home/ubuntu/.minikube/ca.crt
-----BEGIN CERTIFICATE-----
MIIDBjCCAE6gAwIBAgIBATANBgkqhkiG9w0BAQsFADAVMRMwEQYDVQDEwptaW5p
a3ViZUNBMB4XDTE0MDcxMDExNTgyN1oXDTE0MDcwOTExNTgyN1owFTETMBEGA1UE
AxMKbWluaWt1YmVDQTCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBAM0c
qcNWC/YqLEj fYDHXwmUfj1L1B0tIhPj4PHInr5E8eCx2T0D6w/JXFPbBYI0cmL9v
6x5xTxSq71BYHil2oZLHnA+qajcb3VxHvWKjRh0Uz f6E7X8EE0zSpr0Aiml5UfeC
yJQPcZQUZgcAagTay6iA3AokyK5y6sidqfLP2CQEKXWKPACmduPxdUBARMHnpouc
PS8gtPZEI7GYLYD5PKqhrDqdmFSMFahQnIkDZewMx840dGWbr14o+DZd7BGe1VN
ockg0gDMfn5T2Wtgz16gPiDuTuSXSeLHb7RYyIR3jKZ7UcV6fKARVQPJvoR4cYn
Ee1l+EBzwFDg16faftMCAwEAANhMF8wDgYDVR0PAQH/BAQDAgKkMB0GA1UdJQQw
MBQGCCsGAQUFBwMCBggrBgEFBQcDATAPBgNVHRMBAf8EBTADAQH/MB0GA1UdDgQw
BBT0V52DZD+Dnbh0bM8bREIm2Kyf7TANBgkqhkiG9w0BAQsFAA0CAQEATytWhlTD
qCTOM1Ak+D0JQ7H0Vzo1zQI+XyM+AT8gG4j4P9mmbjiid7yxYQM5LK5KQ7vmjml
3fKdUxeVrRzwZXbjvvnGDuG4mJ+aq7SZ0ib29wcR8Woe6qrPW99Un12oamHHowx0
NVVX0lj35wwLqrZZDScQHBkNqV7yP66pDzUBNlPUh2onAm+v+LWgUpz16E5+yR
tPLX9ZFUtpZaIzqlBh8Xt+15PftWReVkg/K82Eusu4vUaGdL8vvXhs8yBXoBI0Tc
hVmWZ05X5c/h0e/ScIHP85PsIuo1p+1hL9prCVqqYcWVh0poFWira74B05Tzfu3u
QG66HufpVMQmug==
-----END CERTIFICATE-----
```

- **Copy the cluster certificate contents**
- In Jenkins Dashboard head over to **Manage Jenkins > Credentials > Global Credentials > Create Credentials**.

- Select type as **Kubernetes Configuration** > **Paste the CA cert** > **Give unique ID** > **Save**.

✓ Install Kubectl

- SSH to the Jenkins server
- Run the following commands.

```
sudo apt-get update
sudo apt-get install -y apt-transport-https ca-certificates curl
gnupg
curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.30/deb/Release.key |
sudo gpg --dearmor -o /etc/apt/keyrings/kubernetes-apt-keyring.gpg
sudo chmod 644 /etc/apt/keyrings/kubernetes-apt-keyring.gpg
sudo apt-get update
sudo apt-get install -y kubectl
```

✓ Generate pipeline steps

- In Jenkins Dashboard head over to **New Item** > **Select Pipeline** > **Give a name** > **Create**
- Scroll down to the bottom of the page **Select Pipeline Syntax** > **Declarative Snippet Generator**.
- Select the **withKubeConfig** step > **Give the details** > **Generate pipeline Syntax**.

The screenshot shows the Jenkins Declarative Snippet Generator interface. At the top, the breadcrumb navigation is 'Dashboard > test > Pipeline Syntax'. On the left, there is a sidebar with links: 'Global Variables Reference', 'Online Documentation', 'Examples Reference', and 'IntelliJ IDEA GDSL'. The main area is titled 'Sample Step' and shows a search bar with 'withKubeConfig: Configure Kubernetes CLI (kubectl)'. Below this, the 'withKubeConfig' step is selected, and its configuration details are shown. The 'Credentials' dropdown is set to '- none -'. The 'Kubernetes server endpoint' is set to 'https://192.168.49.2:8443'. The 'Cluster name' is set to 'minikube'. Red boxes and arrows highlight the search bar, the 'withKubeConfig' step, and the configuration details section.

Dashboard > test > Pipeline Syntax

Global Variables Reference
Online Documentation
Examples Reference
IntelliJ IDEA GDSL

Sample Step

withKubeConfig: Configure Kubernetes CLI (kubectl)

withKubeConfig ?

Credentials

- none -
+ Add

Kubernetes server endpoint ?
https://192.168.49.2:8443

Cluster name ?
minikube

Certificate of certificate authority

-----BEGIN CERTIFICATE-----
MIIDBgCCAdgwwIBAgIBATANBgkqhkiG9w0BAQwFADwvMRMwEEDQVQCDewerkWSp
a3VZUNBMBA4DT0NDQwMDkxNTg1N1xDTM0MDowOTUxNTg1N1xFTETMBEGA1UE
AuMKbWwuaW11mVVDQTCASiNDQV/K6ZlhcNAQE88QDggEPADCCAQoCggEBAMQq
qcNWC/qLqFfDhXwmUfj1L1B0thPJ4Phn5E8eCz2TOD6w/DXFPbY10cmL9v
-----END CERTIFICATE-----

The certificate of the certificate authority (CA). It's used to validate the API server certificate.
Leaving this field empty will skip the certificate verification. (from Kubernetes CLI Plugin)

☒ Restrict access to kubeconfig file

Generate Pipeline Script

Q66HufpVMQmug==
-----END CERTIFICATE-----
{ "clusterName": "minikube", "contextName": "", "credentialsId": "", "namespace": "", "restrictKubeConfigAccess": false, "serverUrl": "https://192.168.49.2:8443" }
// some block
}

➤ Copy the step and add to the pipeline.

```
stage ("DEPLOY TO KUBERNETES"){
    steps{
        withKubeConfig(clusterName: 'minikube', contextName:
'', credentialsId: '', namespace: '', restrictKubeConfigAccess:
false, serverUrl: 'https://192.168.49.2:8443') {
            sh "kubectl apply -f deployment-service.yaml"
        }
    }
}
```