Manage Your Jenkins Multi Agents Cluster With Kubernetes

📘 Overview

This document outlines the complete setup of a Jenkins CI/CD cluster using Docker and Kubernetes (Minikube). The Jenkins Master is manually installed from a WAR file in a custom Docker image, and multiple Jenkins Agents (slaves) are manually deployed using Docker and later as Kubernetes pods. The communication between master and agents is secured using SSH.

🚀 Objective

- Manually install Jenkins in Docker using a WAR file
- Create Docker containers for both Jenkins Master and Agents (Slaves)
- Deploy the same setup using Kubernetes on Minikube
- Configure SSH-based communication between Master and Agents
- Set up and test a Jenkins pipeline using agents

🔧 What I Did (Brief Overview)

🐳 Docker Setup (Part 1)

1. **Created two separate Dockerfiles:**
    - One for **Jenkins Master** with WAR installation
    - One for **Jenkins Agent (Slave)** with SSH and Java
2. **Built both images manually**
3. **Ran containers locally:**
    - One master container
    - Two slave containers

4. **Generated SSH keys inside Jenkins Master container**
5. **Copied the public key to slave containers using `ssh-copy-id`**
6. **Started Jenkins UI, configured SSH-based agents**
7. **Tested with sample Pipeline job**

☸️ Kubernetes Setup (Part 2)

1. **Used same Docker images for Kubernetes pods**
2. **Created:**
    - Jenkins Master Deployment & Service
    - Two Jenkins Agent Deployments
3. **Exposed Jenkins Master via NodePort service**
4. **Accessed Jenkins UI via Minikube IP + Port**
5. **Logged into Master pod, generated SSH keys again**
6. **Copied public key into both Agent pods**
7. **Configured Agents in Jenkins using private key**
8. **Ran a working Pipeline job to verify Jenkins cluster setup**

🔧 Implementation Steps

🐳 Docker Setup (Part 1)

1. **Created two Dockerfiles:**
    - `Dockerfile.master` for Jenkins Master (Ubuntu base + Jenkins WAR setup)
    - `Dockerfile.agent` for Jenkins Agent (Ubuntu base + SSH + Java)

```
FROM redhat/ubi9:latest

RUN dnf clean all && \
    dnf -y update && \
    dnf install -y \
        dnf-plugins-core \
        sudo \
        git \
        wget \
        unzip \
        yum-utils \
        openssh \
        openssh-server \
        openssh-clients && \
    dnf clean all

RUN dnf config-manager --add-repo https://download.docker.com/linux/rhel/
docker-ce.repo && \
    dnf install -y docker-ce docker-ce-cli containerd.io docker-buildx-plugin do
cker-compose-plugin && \
    dnf clean all

RUN useradd -m -s /bin/bash -d /var/lib/jenkins jenkins && \
    echo "jenkins:jenkins123" | chpasswd && \
    echo 'jenkins ALL=(ALL) NOPASSWD:ALL' >> /etc/sudoers && \
    usermod -aG docker jenkins

RUN wget -O /etc/yum.repos.d/jenkins.repo https://pkg.jenkins.io/redhat/jenki
ns.repo && \
    rpm --import https://pkg.jenkins.io/redhat/jenkins.io-2023.key && \
    dnf upgrade -y && \
    dnf install -y fontconfig java-21-openjdk jenkins  && \
    dnf clean all

RUN chown jenkins:jenkins /usr/share/java/jenkins.war

RUN ssh-keygen -A && \
    mkdir -p /etc/ssh/sshd_config.d && \
    echo "PermitRootLogin no" > /etc/ssh/sshd_config.d/custom.conf && \
    echo "PasswordAuthentication yes" >> /etc/ssh/sshd_config.d/custom.con
```

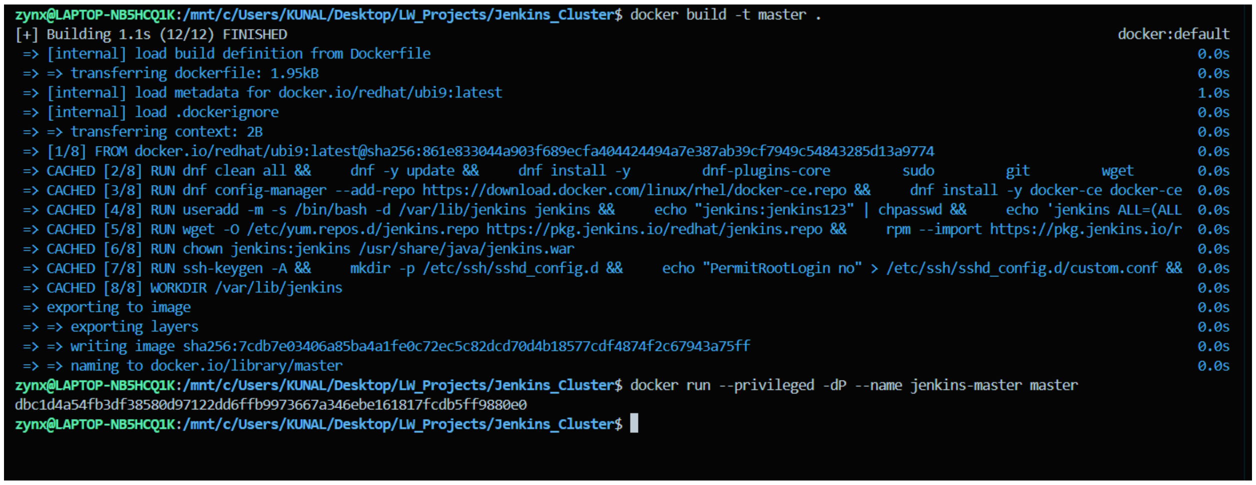
```
f && \
    echo "ChallengeResponseAuthentication no" >> /etc/ssh/sshd_config.d/cu
stom.conf && \
    echo "UsePAM yes" >> /etc/ssh/sshd_config.d/custom.conf

USER jenkins

WORKDIR /var/lib/jenkins

EXPOSE 8080 50000 22



ENTRYPOINT ["bash", "-c", "sudo dockerd & sleep 3 && sudo /usr/sbin/sshd &
& exec java -jar /usr/share/java/jenkins.war"]
```

1. **Built Docker Images Locally:**

   ```
   docker build -t master -f Dockerfile.master .
   docker build -t slave -f Dockerfile.slave .
   ```

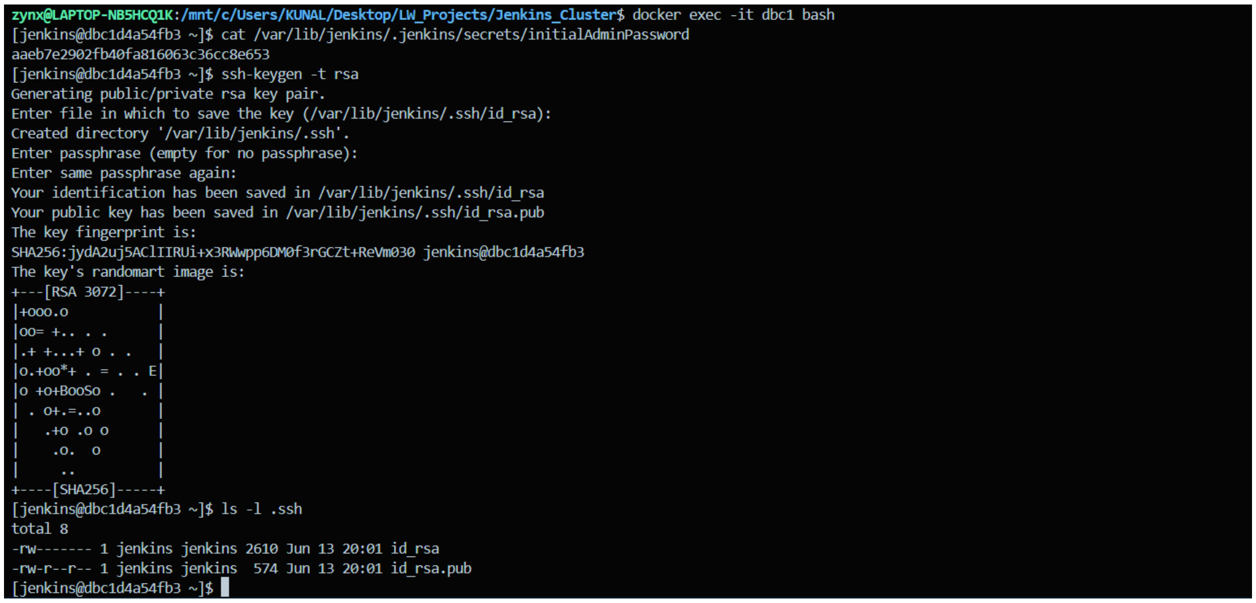
2. **Ran Containers Locally:**

   ```
   docker run -d --name master -p 8080:8080 master
   docker run -d --name agent1 slave
   docker run -d --name agent2 slave
   ```

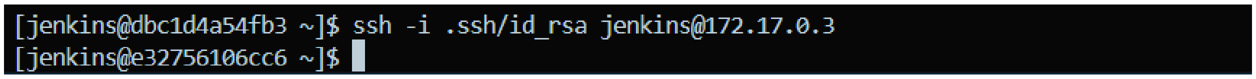


```
zynx@LAPTOP-NB5HCQ1K:/mnt/c/Users/KUNAL/Desktop/LW_Projects/Jenkins_Cluster$ docker build -t master .
[+] Building 1.1s (12/12) FINISHED                                                                     docker:default
 => [internal] load build definition from Dockerfile                                                            0.0s
 => => transferring dockerfile: 1.95kB                                                                          0.0s
 => [internal] load metadata for docker.io/redhat/ubi9:latest                                                   1.0s
 => [internal] load .dockerignore                                                                               0.0s
 => => transferring context: 2B                                                                                 0.0s
 => [1/8] FROM docker.io/redhat/ubi9:latest@sha256:861e833044a903f689ecfa404424494a7e387ab39cf7949c54843285d13a9774  0.0s
 => CACHED [2/8] RUN dnf clean all &&     dnf -y update &&     dnf install -y         dnf-plugins-core         sudo         git         wget  0.0s
 => CACHED [3/8] RUN dnf config-manager --add-repo https://download.docker.com/linux/rhel/docker-ce.repo &&     dnf install -y docker-ce docker-ce  0.0s
 => CACHED [4/8] RUN useradd -m -s /bin/bash -d /var/lib/jenkins jenkins &&     echo "jenkins:jenkins123" | chpasswd &&     echo 'jenkins ALL=(ALL  0.0s
 => CACHED [5/8] RUN wget -O /etc/yum.repos.d/jenkins.repo https://pkg.jenkins.io/redhat/jenkins.repo &&     rpm --import https://pkg.jenkins.io/r  0.0s
 => CACHED [6/8] RUN chown jenkins:jenkins /usr/share/java/jenkins.war                                          0.0s
 => CACHED [7/8] RUN ssh-keygen -A &&     mkdir -p /etc/ssh/sshd_config.d &&     echo "PermitRootLogin no" > /etc/ssh/sshd_config.d/custom.conf &&  0.0s
 => CACHED [8/8] WORKDIR /var/lib/jenkins                                                                       0.0s
 => exporting to image                                                                                          0.0s
 => => exporting layers                                                                                         0.0s
 => => writing image sha256:7cdb7e03406a85ba4a1fe0c72ec5c82dcd70d4b18577cdf4874f2c67943a75ff                     0.0s
 => => naming to docker.io/library/master                                                                       0.0s
zynx@LAPTOP-NB5HCQ1K:/mnt/c/Users/KUNAL/Desktop/LW_Projects/Jenkins_Cluster$ docker run --privileged -dP --name jenkins-master master
dbc1d4a54fb3df38580d97122dd6ffb9973667a346ebe161817fcdb5ff9880e0
zynx@LAPTOP-NB5HCQ1K:/mnt/c/Users/KUNAL/Desktop/LW_Projects/Jenkins_Cluster$
```

1. **SSH Setup:**
    - Generated SSH key inside master container: `ssh-keygen -t rsa`
    - Used `ssh-copy-id` to copy the public key to agents
    - Verified passwordless SSH connection

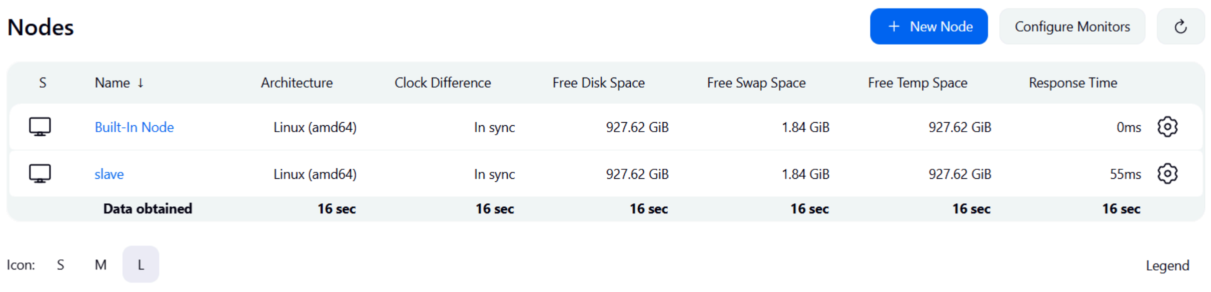


```
zynx@LAPTOP-NB5HCQ1K:/mnt/c/Users/KUNAL/Desktop/LW_Projects/Jenkins_Cluster$ docker exec -it dbc1 bash
[jenkins@dbc1d4a54fb3 ~]$ cat /var/lib/jenkins/.jenkins/secrets/initialAdminPassword
aaeb7e2902fb40fa816063c36cc8e653
[jenkins@dbc1d4a54fb3 ~]$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/var/lib/jenkins/.ssh/id_rsa):
Created directory '/var/lib/jenkins/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /var/lib/jenkins/.ssh/id_rsa
Your public key has been saved in /var/lib/jenkins/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:jydA2uj5AClIIRUi+x3RWwpp6DM0f3rGCZt+ReVm030 jenkins@dbc1d4a54fb3
The key's randomart image is:
+---[RSA 3072]----+
|+ooo.o           |
|oo= +.. . .      |
|.+ +...+ o . .   |
|o.+oo*+ . = . . E|
|o +o+BooSo .   . |
| . o+.=..o       |
|   .+o .o o      |
|    .o.  o       |
|     ..          |
+----[SHA256]-----+
[jenkins@dbc1d4a54fb3 ~]$ ls -l .ssh
total 8
-rw------- 1 jenkins jenkins 2610 Jun 13 20:01 id_rsa
-rw-r--r-- 1 jenkins jenkins  574 Jun 13 20:01 id_rsa.pub
[jenkins@dbc1d4a54fb3 ~]$
```



```
[jenkins@dbc1d4a54fb3 ~]$ ssh -i .ssh/id_rsa jenkins@172.17.0.3
[jenkins@e32756106cc6 ~]$
```

1. **Jenkins UI Setup:**
    - Opened Jenkins at `localhost:8080`
    - Installed required plugins (SSH Build Agents)
    - Added agent nodes using SSH credentials (private key method)
    - Labels assigned as `agent1`, `agent2`



Nodes

S	Name ↓	Architecture	Clock Difference	Free Disk Space	Free Swap Space	Free Temp Space	Response Time
🖥	Built-In Node	Linux (amd64)	In sync	927.62 GiB	1.84 GiB	927.62 GiB	0ms
🖥	slave	Linux (amd64)	In sync	927.62 GiB	1.84 GiB	927.62 GiB	55ms
	Data obtained	16 sec	16 sec	16 sec	16 sec	16 sec	16 sec

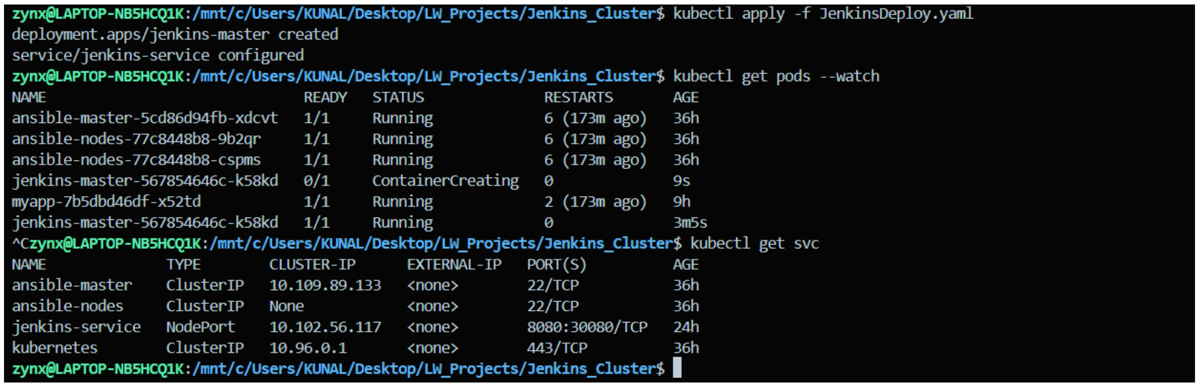
☸️ Kubernetes Setup (Part 2)

1. **Created YAML Deployments:**
    - One deployment for Jenkins master with NodePort service
    - Two deployments for agents (no service needed)

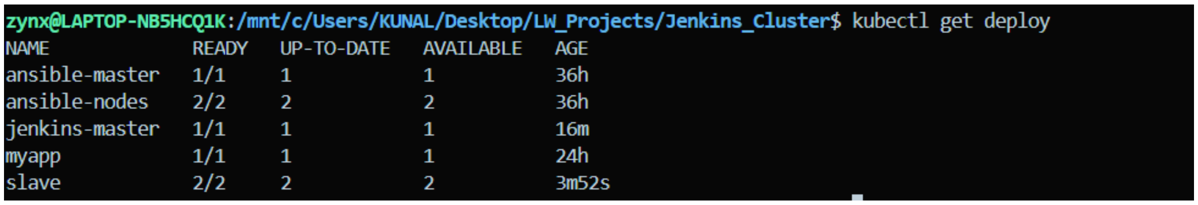
```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: jenkins-master
spec:
  replicas: 1
  selector:
    matchLabels:
      app: jenkins
  template:
    metadata:
      labels:
        app: jenkins
    spec:
      containers:
        - name: jenkins
          image: zynx01/master:v1
          ports:
            - containerPort: 8080
          volumeMounts:
            - name: jenkins-data
              mountPath: /var/jenkins_home
      volumes:
        - name: jenkins-data
          emptyDir: {}

---
apiVersion: v1
kind: Service
metadata:
  name: jenkins-service
spec:
  type: NodePort
  ports:
    - port: 8080
      targetPort: 8080
      nodePort: 30080
  selector:
    app: jenkins

 ---

 apiVersion: apps/v1
kind: Deployment
metadata:
  name: slave
spec:
  replicas: 2
  selector:
    matchLabels:
      app: slave
  template:
    metadata:
      labels:
        app: slave
    spec:
      containers:
        - name: slave
          image: zynx01/slave:v1
          resources:
          securityContext:
            privileged: true
          ports:
            - containerPort: 22
```

1. **Deployed to Minikube:**

   ```
   kubectl apply -f jenkins-master-deployment.yaml
   kubectl apply -f jenkins-agent-deployment-1.yaml
   ```



```
zynx@LAPTOP-NB5HCQ1K:/mnt/c/Users/KUNAL/Desktop/LW_Projects/Jenkins_Cluster$ kubectl apply -f JenkinsDeploy.yaml
deployment.apps/jenkins-master created
service/jenkins-service configured
zynx@LAPTOP-NB5HCQ1K:/mnt/c/Users/KUNAL/Desktop/LW_Projects/Jenkins_Cluster$ kubectl get pods --watch
NAME                              READY   STATUS              RESTARTS       AGE
ansible-master-5cd86d94fb-xdcvt   1/1     Running             6 (173m ago)   36h
ansible-nodes-77c8448b8-9b2qr     1/1     Running             6 (173m ago)   36h
ansible-nodes-77c8448b8-cspms     1/1     Running             6 (173m ago)   36h
jenkins-master-567854646c-k58kd   0/1     ContainerCreating   0              9s
myapp-7b5dbd46df-x52td            1/1     Running             2 (173m ago)   9h
jenkins-master-567854646c-k58kd   1/1     Running             0              3m5s
^Czynx@LAPTOP-NB5HCQ1K:/mnt/c/Users/KUNAL/Desktop/LW_Projects/Jenkins_Cluster$ kubectl get svc
NAME              TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)          AGE
ansible-master    ClusterIP   10.109.89.133   <none>        22/TCP           36h
ansible-nodes     ClusterIP   None            <none>        22/TCP           36h
jenkins-service   NodePort    10.102.56.117   <none>        8080:30080/TCP   24h
kubernetes        ClusterIP   10.96.0.1       <none>        443/TCP          36h
zynx@LAPTOP-NB5HCQ1K:/mnt/c/Users/KUNAL/Desktop/LW_Projects/Jenkins_Cluster$
```



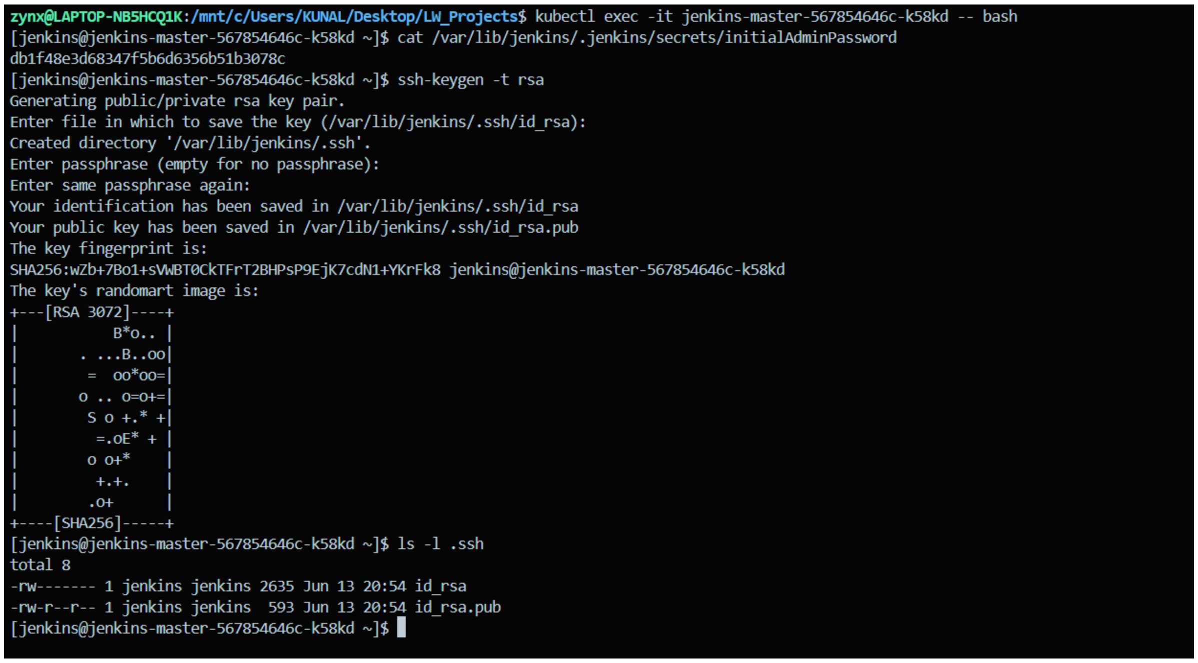
```
zynx@LAPTOP-NB5HCQ1K:/mnt/c/Users/KUNAL/Desktop/LW_Projects/Jenkins_Cluster$ kubectl get deploy
NAME             READY   UP-TO-DATE   AVAILABLE   AGE
ansible-master   1/1     1            1           36h
ansible-nodes    2/2     2            2           36h
jenkins-master   1/1     1            1           16m
myapp            1/1     1            1           24h
slave            2/2     2            2           3m52s
```

2. **Accessed Jenkins UI:**

```
minikube service jenkins-service --url
```

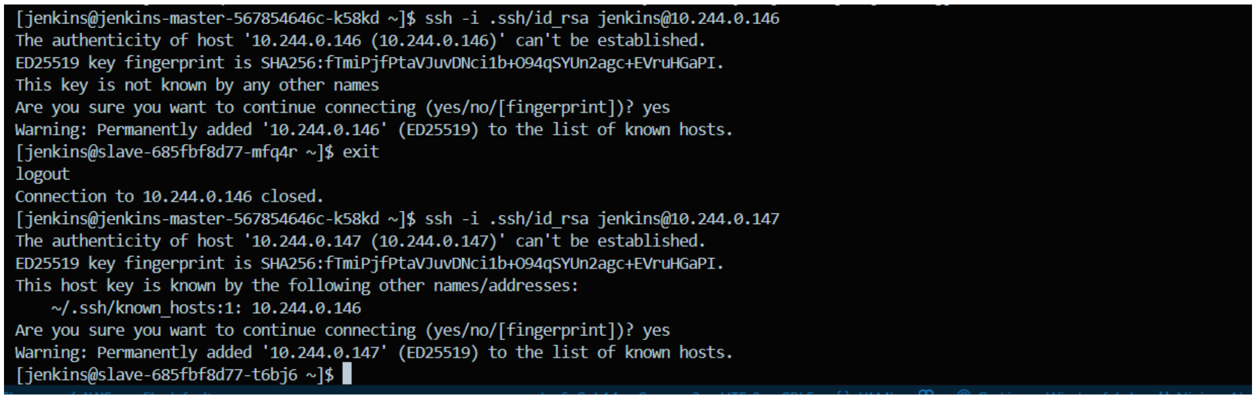
3. **SSH Setup in K8s:**

- Executed bash in master pod: `kubectl exec -it master -- bash`
- Generated SSH key: `ssh-keygen`
- Fetched IPs of agent pods: `kubectl get pods -o wide`
- Used `ssh-copy-id` to copy public key to agents



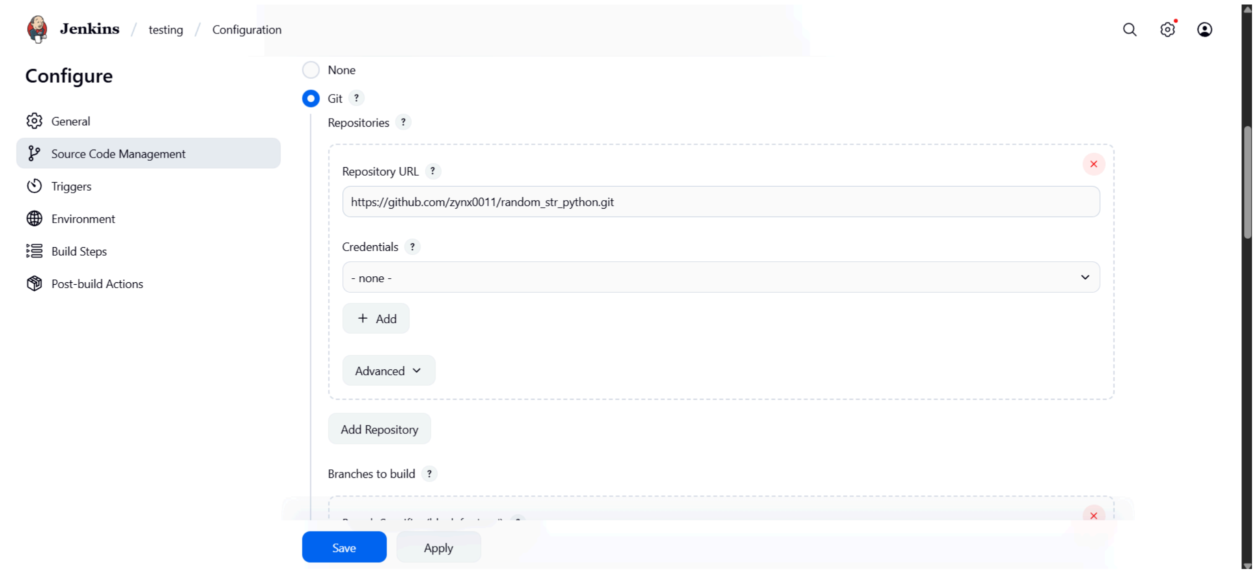
```
zynx@LAPTOP-NB5HCQ1K:/mnt/c/Users/KUNAL/Desktop/LW_Projects$ kubectl exec -it jenkins-master-567854646c-k58kd -- bash
[jenkins@jenkins-master-567854646c-k58kd ~]$ cat /var/lib/jenkins/.jenkins/secrets/initialAdminPassword
db1f48e3d68347f5b6d6356b51b3078c
[jenkins@jenkins-master-567854646c-k58kd ~]$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/var/lib/jenkins/.ssh/id_rsa):
Created directory '/var/lib/jenkins/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /var/lib/jenkins/.ssh/id_rsa
Your public key has been saved in /var/lib/jenkins/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:wZb+7Bo1+sVWBT0CkTFrT2BHPsP9EjK7cdN1+YKrFk8 jenkins@jenkins-master-567854646c-k58kd
The key's randomart image is:
+---[RSA 3072]----+
|            B*o.. |
|        . ...B..oo|
|         =  oo*oo=|
|        o .. o=o+=|
|         S o +.* +|
|          =.oE* + |
|         o o+*    |
|          +.+.    |
|         .o+      |
+----[SHA256]-----+
[jenkins@jenkins-master-567854646c-k58kd ~]$ ls -l .ssh
total 8
-rw------- 1 jenkins jenkins 2635 Jun 13 20:54 id_rsa
-rw-r--r-- 1 jenkins jenkins  593 Jun 13 20:54 id_rsa.pub
[jenkins@jenkins-master-567854646c-k58kd ~]$ 
```

4. **Configured Jenkins UI with agent pod IPs and same private key**

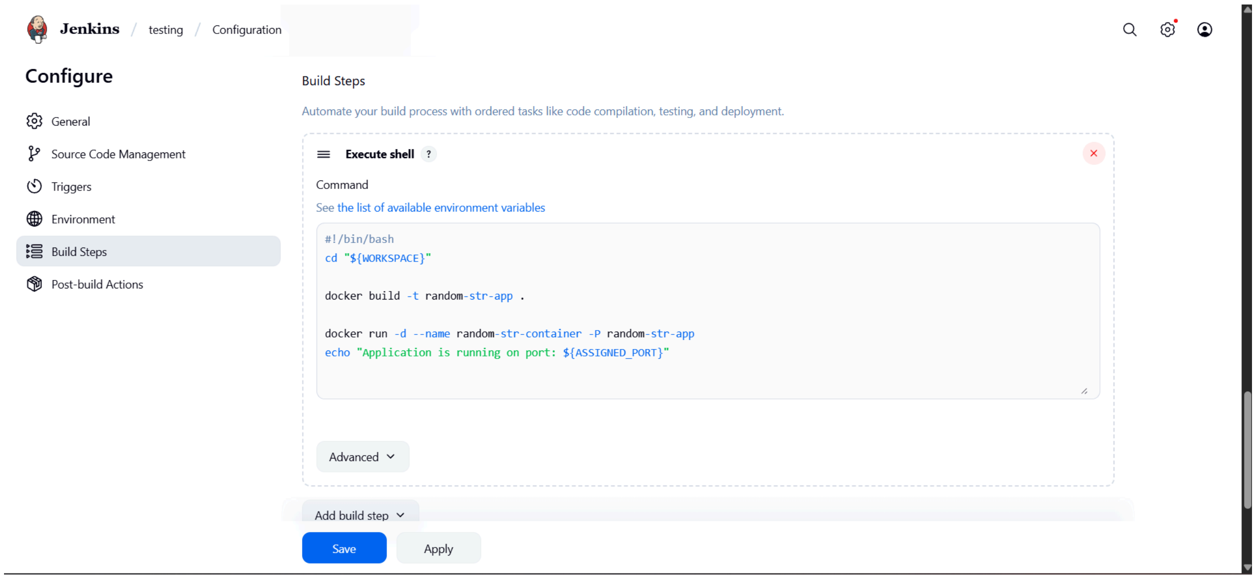


```
[jenkins@jenkins-master-567854646c-k58kd ~]$ ssh -i .ssh/id_rsa jenkins@10.244.0.146
The authenticity of host '10.244.0.146 (10.244.0.146)' can't be established.
ED25519 key fingerprint is SHA256:fTmiPjfPtaVJuvDNci1b+O94qSYUn2agc+EVruHGaPI.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.244.0.146' (ED25519) to the list of known hosts.
[jenkins@slave-685fbf8d77-mfq4r ~]$ exit
logout
Connection to 10.244.0.146 closed.
[jenkins@jenkins-master-567854646c-k58kd ~]$ ssh -i .ssh/id_rsa jenkins@10.244.0.147
The authenticity of host '10.244.0.147 (10.244.0.147)' can't be established.
ED25519 key fingerprint is SHA256:fTmiPjfPtaVJuvDNci1b+O94qSYUn2agc+EVruHGaPI.
This host key is known by the following other names/addresses:
    ~/.ssh/known_hosts:1: 10.244.0.146
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.244.0.147' (ED25519) to the list of known hosts.
[jenkins@slave-685fbf8d77-t6bj6 ~]$ 
```

5. Created pipeline and ran it on agent pods successfully

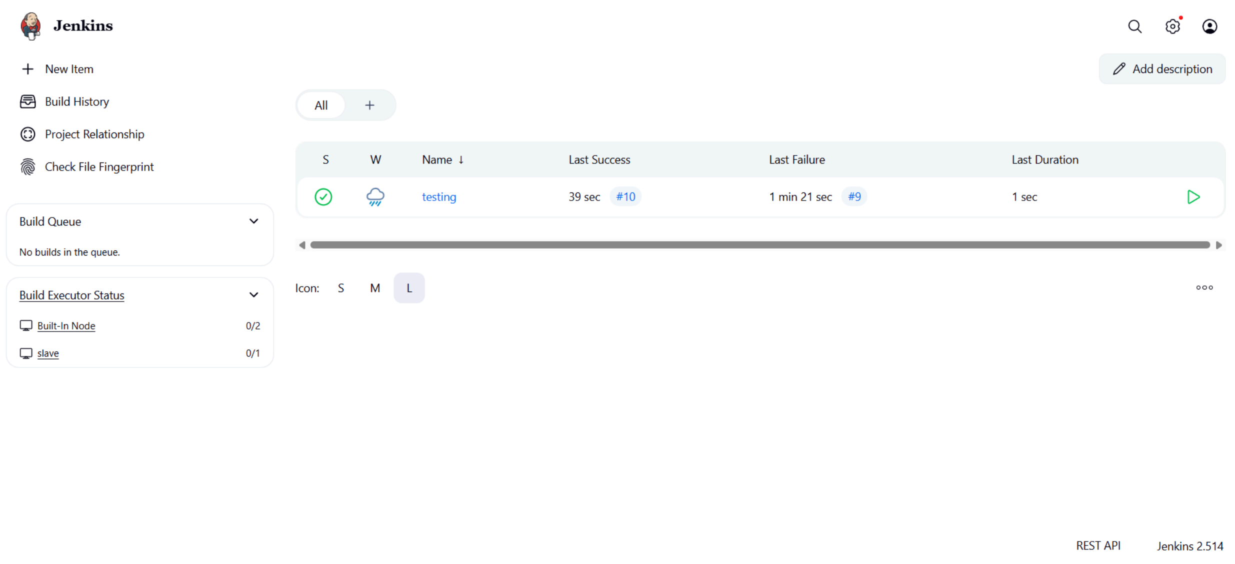


None
[✓ Git]
Repositories
Repository URL: https://github.com/zynx0011/random_str_python.git
Credentials: - none -
Add
Advanced
Add Repository
Branches to build

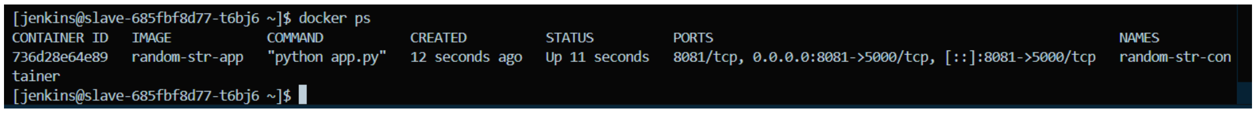


Build Steps
Automate your build process with ordered tasks like code compilation, testing, and deployment.
Execute shell
Command
See the list of available environment variables

```
#!/bin/bash
cd "${WORKSPACE}"

docker build -t random-str-app .

docker run -d --name random-str-container -P random-str-app
echo "Application is running on port: ${ASSIGNED_PORT}"
```



S	W	Name ↓	Last Success	Last Failure	Last Duration
✓		testing	39 sec #10	1 min 21 sec #9	1 sec



```
[jenkins@slave-685fbf8d77-t6bj6 ~]$ docker ps
CONTAINER ID   IMAGE            COMMAND           CREATED          STATUS          PORTS                                            NAMES
736d28e64e89   random-str-app   "python app.py"   12 seconds ago   Up 11 seconds   8081/tcp, 0.0.0.0:8081->5000/tcp, [::]:8081->5000/tcp   random-str-con
tainer
[jenkins@slave-685fbf8d77-t6bj6 ~]$
```

Final Thoughts

This project demonstrates how to **modernize Jenkins** by combining **Docker for custom deployments** and **Kubernetes for dynamic scaling**. The setup ensures **efficient CI/CD pipelines** with on-demand resource allocation, making it ideal for cloud-native environments.