**Jenkins + Docker Pipeline Project Documentation**

**Objective**

Automate the build, test, and deployment of a Dockerized web application using Jenkins Pipeline.

**Prerequisites**

Ensure the following are installed on your Jenkins server:

✅ Jenkins (Latest version)  
✅ Docker (For containerization)  
✅ Git (For version control)  
✅ SSH (For remote deployments)  
✅ Jenkins Plugins:

* Pipeline Plugin
* Docker Pipeline Plugin
* Git Plugin

**Step 1: Install Docker on Jenkins Server**

1. **Update system packages and install Docker:**

sudo apt update

sudo apt install docker.io -y

1. **Start and enable Docker:**

sudo systemctl start docker

sudo systemctl enable docker

1. **Add Jenkins user to Docker group (to allow Jenkins to run Docker commands):**

sudo usermod -aG docker jenkins

1. **Restart Jenkins to apply changes:**

sudo systemctl restart jenkins

1. **Verify Docker installation:**

docker --version

**Step 2: Enable Password Authentication (If Needed)**

If SSH key authentication is not set up, enable password login:

1. **Connect to the remote server and edit the SSH configuration file:**

sudo nano /etc/ssh/sshd\_config

1. **Modify these lines:**

PasswordAuthentication yes

PermitRootLogin yes

1. **Save the file and restart SSH:**

sudo systemctl restart ssh

1. **Test SSH login:**

ssh master@192.168.203.128

**Step 3: Create a Simple Web Application**

1. **Clone the sample app repository:**

git clone https://github.com/KyathamRohith/jenkins-docker.git

cd jenkins-docker

1. **Create a Dockerfile in the project directory:**

FROM nginx:latest – to run static web application

COPY index.html /usr/share/nginx/html/

EXPOSE 80

CMD ["nginx", "-g", "daemon off;"]

1. **Create an index.html file:**

<!DOCTYPE html>

<html>

<head>

<title>Jenkins + Docker Pipeline</title>

</head>

<body>

<h1>Deployment Successful with Jenkins and Docker!</h1>

</body>

</html>

**Step 4: Create a Jenkins Pipeline**

1. **Open Jenkins → New Item → Pipeline → OK**
2. **Copy and paste the following Jenkinsfile:**

pipeline {

agent any

environment {

DOCKER\_IMAGE = "app-image"

DOCKER\_TAG = "latest"

DOCKER\_REPO = "default-repo/app-image"

DOCKER\_CREDENTIALS\_ID = "docker-credentials-id"

CONTAINER\_NAME = "local-container"

CONTAINER\_NAME1 = "server-container"

}

stages {

stage('Clone Repository') {

steps {

git 'https://github.com/default-user/repo.git'

}

}

stage('Docker Login') {

steps {

script {

docker.withRegistry('https://index.docker.io/v1/', DOCKER\_CREDENTIALS\_ID) {

echo "Logged into Docker Hub"

}

}

}

}

stage('Build Docker Image') {

steps {

script {

sh "docker build -t ${DOCKER\_IMAGE}:${DOCKER\_TAG} ."

}

}

}

stage('Run Container Locally') {

steps {

script {

sh """

docker ps -a -q --filter name=${CONTAINER\_NAME} | xargs -r docker stop || true

docker ps -a -q --filter name=${CONTAINER\_NAME} | xargs -r docker rm || true

docker run -d -p 8093:80 --name ${CONTAINER\_NAME} ${DOCKER\_IMAGE}:${DOCKER\_TAG}

"""

}

}

}

stage('Push to Docker Hub') {

steps {

script {

docker.withRegistry('https://index.docker.io/v1/', DOCKER\_CREDENTIALS\_ID) {

sh "docker tag ${DOCKER\_IMAGE}:${DOCKER\_TAG} ${DOCKER\_REPO}:${DOCKER\_TAG}"

sh "docker push ${DOCKER\_REPO}:${DOCKER\_TAG}"

}

}

}

}

stage('Deploy to Server') {

steps {

script {

sh """

sshpass -p "password" ssh -o StrictHostKeyChecking=no user@server-ip '

docker pull ${DOCKER\_REPO}:${DOCKER\_TAG} &&

docker ps -a -q --filter name=${CONTAINER\_NAME1} | xargs -r docker stop || true &&

docker ps -a -q --filter name=${CONTAINER\_NAME1} | xargs -r docker rm || true &&

docker run -d -p 80:80 --name ${CONTAINER\_NAME1} ${DOCKER\_REPO}:${DOCKER\_TAG}'

"""

}

}

}

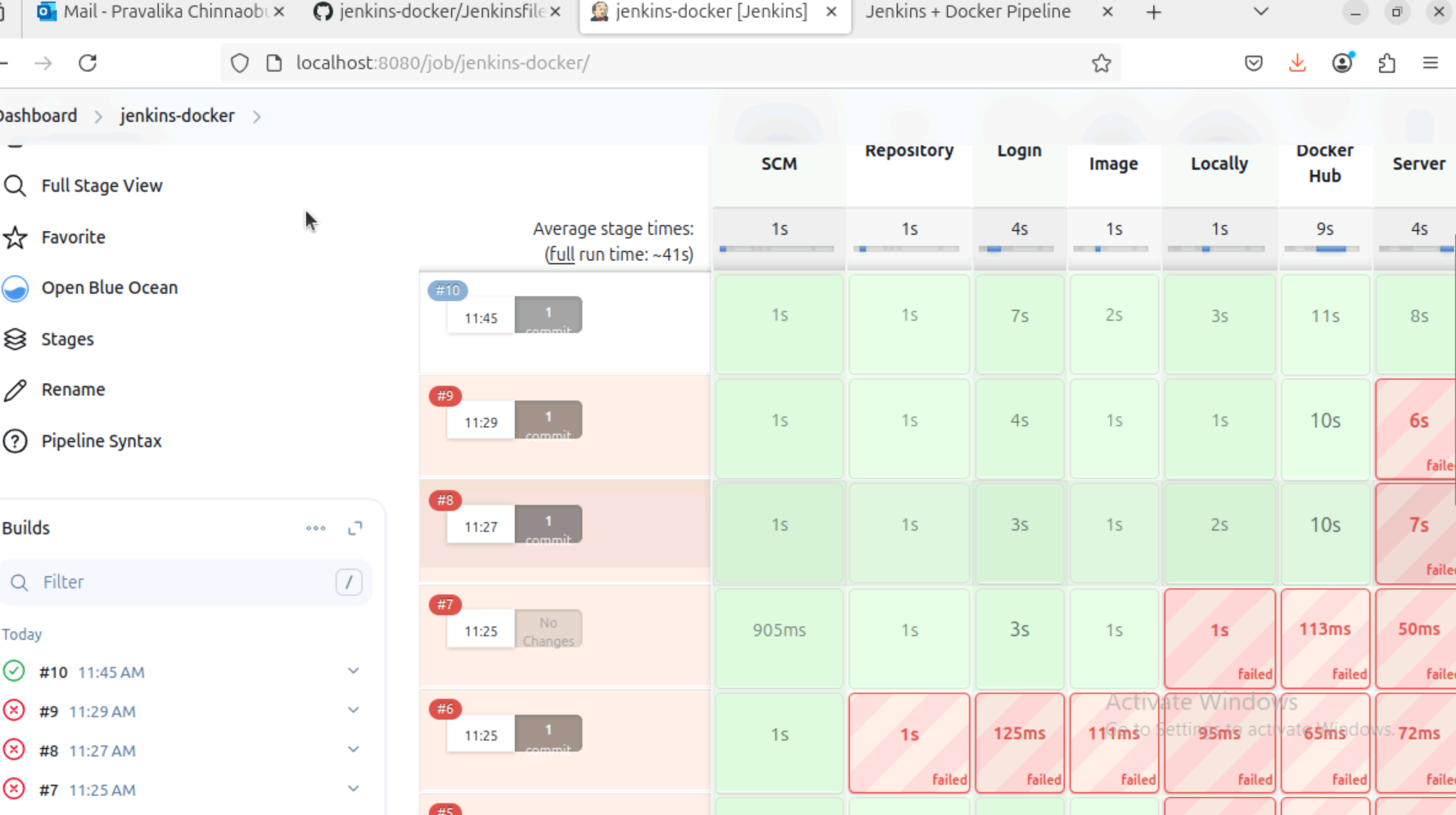
}

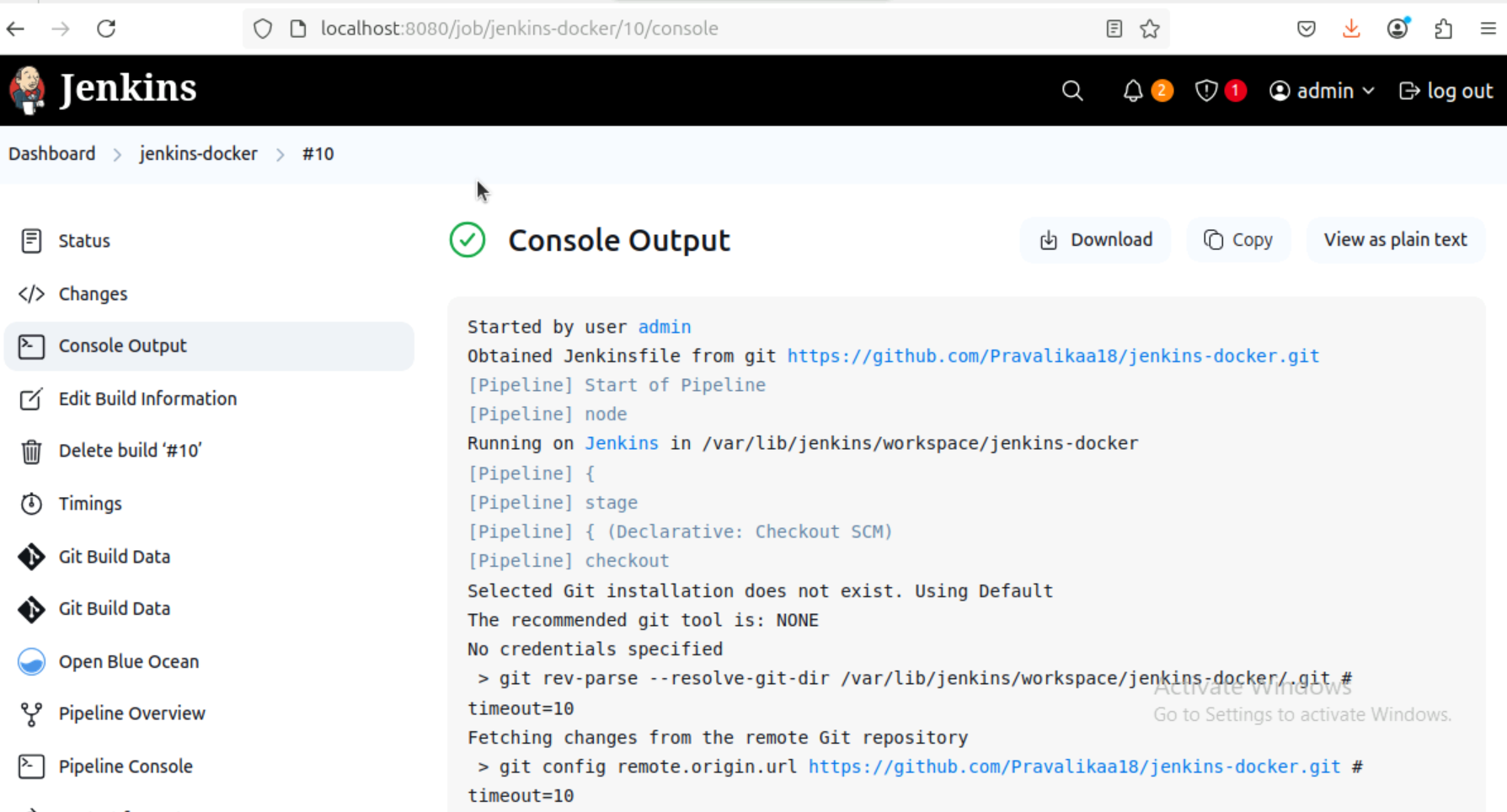
}

**Step 5: Run the Jenkins Pipeline**

1. **Go to Jenkins Dashboard → Click on your pipeline job**
2. **Click "Build Now"**
3. **Check console output to verify the build process**

OUTPUT





Localhost with portn0.

