**Assignment 4**

**Assignment Title:** "Multi-Agent Docker Container Deployment with Jenkins Pipeline"

**Brief Overview:**

This assignment demonstrates the automation of deploying web applications using Jenkins pipelines across multiple agents. It involves:

* **Docker Setup:** Installing Docker, starting the service, and managing containers and volumes.
* **Git Integration:** Cloning specific branches of a Git repository into designated directories.
* **Volume Management:** Creating Docker volumes, copying necessary files, and setting appropriate permissions.
* **Container Deployment:** Running Docker containers with mounted volumes to serve web content.

This process is executed across three agents (built-in, slave-1, slave-2), each handling different stages of the deployment, ensuring a modular and scalable CI/CD pipeline.

**Script-**

pipeline {

agent none

stages {

stage('emptying container') {

agent { label 'built-in' }

steps {

sh '''

sudo yum install docker -y

sudo service docker start

sudo docker kill C1 || true

sudo docker rm C1 || true

sudo rm -rf /mnt/Project\_2025 || true

sudo docker volume rm V1 || true

'''

}

}

stage('cloning git - built-in') {

agent { label 'built-in' }

steps {

sh '''

sudo yum install git -y

sudo git clone -b 2025-Q1 https://github.com/prasadpawar252/20205\_REPO.git /mnt/Project\_2025

'''

}

}

stage('creating volume - built-in') {

agent { label 'built-in' }

steps {

sh '''

sudo docker volume create V1

sudo cp /mnt/Project\_2025/index.html /var/lib/docker/volumes/V1/\_data/

sudo chmod -R 777 /var/lib/docker/volumes/V1/\_data/index.html

'''

}

}

stage('creating container - built-in') {

agent { label 'built-in' }

steps {

sh '''

sudo docker run -itd -p 60:80 -v V1:/usr/local/apache2/htdocs --name C1 httpd

'''

}

}

stage('Docker for slave-1') {

agent { label 'slave-1' }

steps {

sh '''

sudo yum install docker -y

sudo service docker start

sudo docker kill C2 || true

sudo docker rm C2 || true

sudo rm -rf /mnt/Project\_2025 || true

sudo docker volume rm V2 || true

'''

}

}

stage('cloning git - slave-1') {

agent { label 'slave-1' }

steps {

sh '''

sudo yum install git -y

sudo git clone -b 2025-Q2 https://github.com/prasadpawar252/20205\_REPO.git /mnt/Project\_2025

'''

}

}

stage('creating volume - slave-1') {

agent { label 'slave-1' }

steps {

sh '''

sudo docker volume create V2

sudo cp /mnt/Project\_2025/index.html /var/lib/docker/volumes/V2/\_data/

sudo chmod -R 777 /var/lib/docker/volumes/V2/\_data/index.html

'''

}

}

stage('creating container - slave-1') {

agent { label 'slave-1' }

steps {

sh '''

sudo docker run -itd -p 60:80 -v V2:/usr/local/apache2/htdocs --name C2 httpd

'''

}

}

stage('Docker for slave-2') {

agent { label 'slave-2' }

steps {

sh '''

sudo yum install docker -y

sudo service docker start

sudo docker kill C3 || true

sudo docker rm C3 || true

sudo rm -rf /mnt/Project\_2025 || true

sudo docker volume rm V3 || true

'''

}

}

stage('cloning git - slave-2') {

agent { label 'slave-2' }

steps {

sh '''

sudo yum install git -y

sudo git clone -b 2025-Q3 https://github.com/prasadpawar252/20205\_REPO.git /mnt/Project\_2025

'''

}

}

stage('creating volume - slave-2') {

agent { label 'slave-2' }

steps {

sh '''

sudo docker volume create V3

sudo cp /mnt/Project\_2025/index.html /var/lib/docker/volumes/V3/\_data/

sudo chmod -R 777 /var/lib/docker/volumes/V3/\_data/index.html

'''

}

}

stage('creating container - slave-2') {

agent { label 'slave-2' }

steps {

sh '''

sudo docker run -itd -p 60:80 -v V3:/usr/local/apache2/htdocs --name C3 httpd

'''

}

}

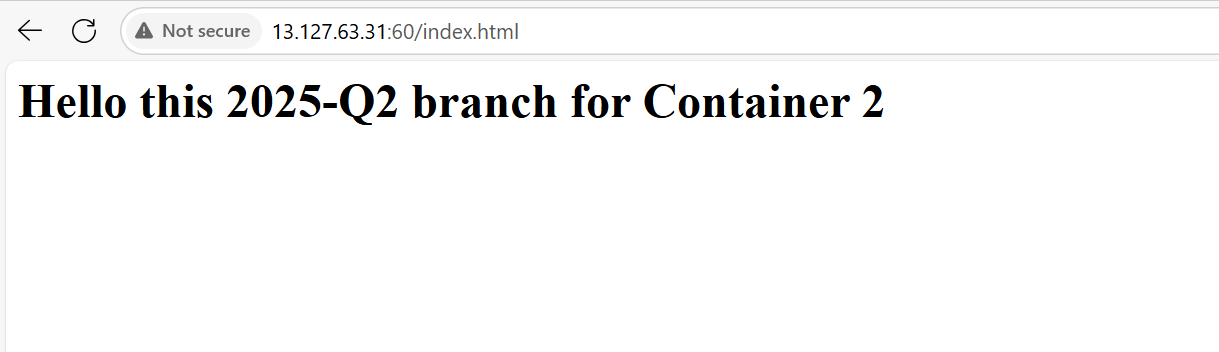
}

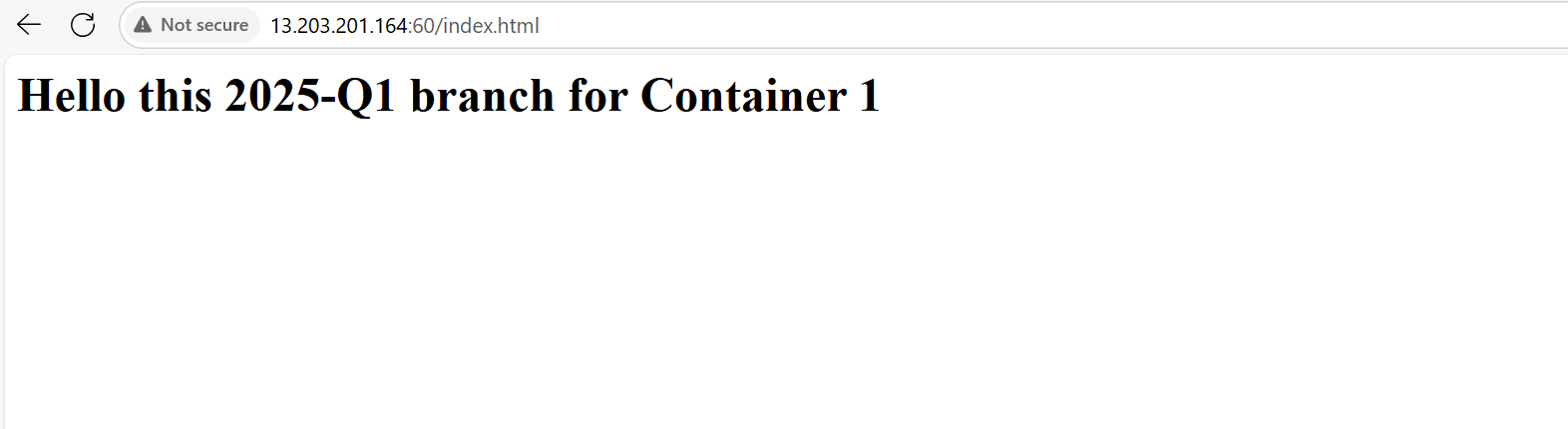
}

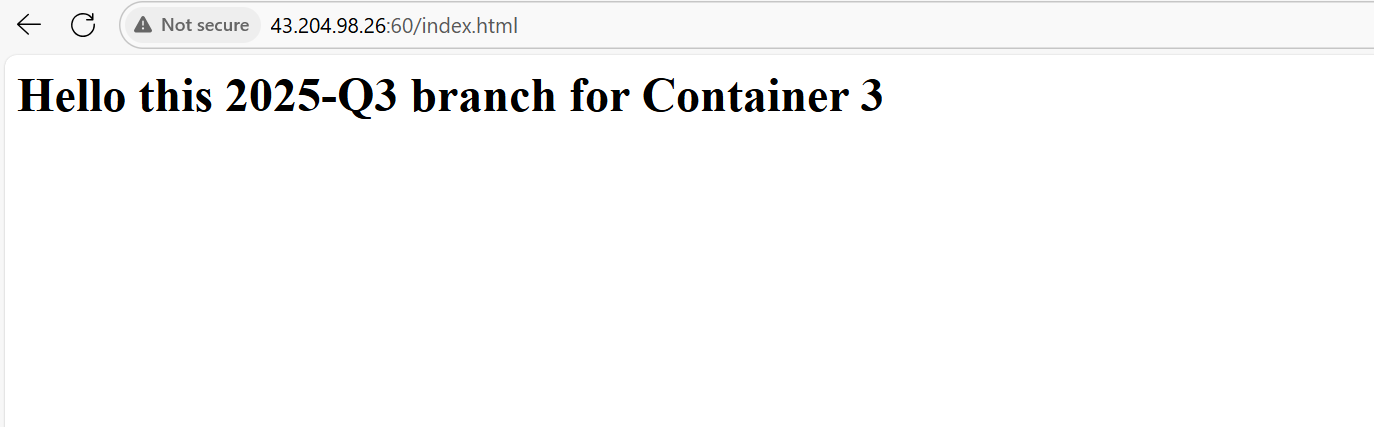
**Console outputs –**

****

**Final Outputs –**

****

****

****

**Pipeline Overview**

This Jenkins pipeline automates the deployment of web applications using Docker containers across three different agents: built-in, slave-1, and slave-2. Each agent handles a specific phase of the deployment, ensuring a modular and scalable CI/CD process.

**Stages Explained**

1. **Emptying Container (built-in agent)**
   * Installs Docker and starts the Docker service.
   * Terminates and removes any existing container named C1.
   * Deletes the /mnt/Project\_2025 directory and removes the Docker volume V1.
2. **Cloning Git Repository (built-in agent)**
   * Installs Git.
   * Clones the 2025-Q1 branch of the repository into /mnt/Project\_2025.
3. **Creating Volume (built-in agent)**
   * Creates a Docker volume named V1.
   * Copies the index.html file from the cloned repository into the volume.
   * Sets appropriate permissions for the index.html file.
4. **Creating Container (built-in agent)**
   * Runs a Docker container named C1 using the httpd image.
   * Maps port 60 on the host to port 80 in the container.
   * Mounts the V1 volume to serve the index.html file.
5. **Docker Setup for Slave-1**
   * Installs Docker and starts the Docker service on slave-1.
   * Terminates and removes any existing container named C2.
   * Deletes the /mnt/Project\_2025 directory and removes the Docker volume V2.
6. **Cloning Git Repository (slave-1)**
   * Installs Git on slave-1.
   * Clones the 2025-Q2 branch of the repository into /mnt/Project\_2025.
7. **Creating Volume (slave-1)**
   * Creates a Docker volume named V2.
   * Copies the index.html file from the cloned repository into the volume.
   * Sets appropriate permissions for the index.html file.
8. **Creating Container (slave-1)**
   * Runs a Docker container named C2 using the httpd image.
   * Maps port 60 on the host to port 80 in the container.
   * Mounts the V2 volume to serve the index.html file.
9. **Docker Setup for Slave-2**
   * Installs Docker and starts the Docker service on slave-2.
   * Terminates and removes any existing container named C3.
   * Deletes the /mnt/Project\_2025 directory and removes the Docker volume V3.
10. **Cloning Git Repository (slave-2)**
    * Installs Git on slave-2.
    * Clones the 2025-Q3 branch of the repository into /mnt/Project\_2025.
11. **Creating Volume (slave-2)**
    * Creates a Docker volume named V3.
    * Copies the index.html file from the cloned repository into the volume.
    * Sets appropriate permissions for the index.html file.
12. **Creating Container (slave-2)**
    * Runs a Docker container named C3 using the httpd image.
    * Maps port 60 on the host to port 80 in the container.
    * Mounts the V3 volume to serve the index.html file.