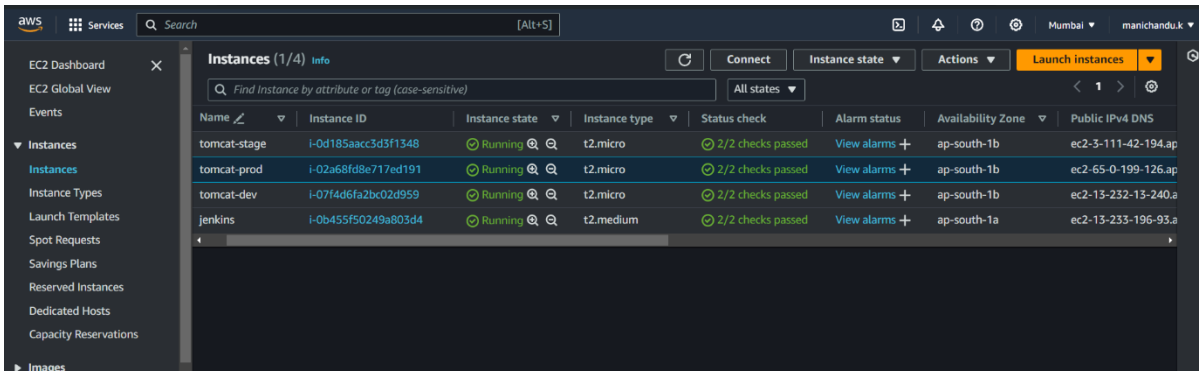
Deployment of an application into three Tomcat servers in different environments with Condition Yes (or) No using CI/CD

By

Manichandu K.

- First launch an instance for our Jenkins server of instance type “t2.medium” and AMI type ubuntu.
- Now launch three free-tier instances for the Tomcat server for three different environments.



- Now SSH into the Jenkins server and give these commands.

sudo -i
apt update && apt install default-jdk -y
apt install maven -y
*sudo wget -O /usr/share/keyrings/jenkins-keyring.asc *
https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key
*echo "deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc]" *
*https://pkg.jenkins.io/debian-stable binary/ | sudo tee *
/etc/apt/sources.list.d/jenkins.list > /dev/null
sudo apt-get update
sudo apt-get install Jenkins -y

- Now the Jenkins server is ready and to be sure give these commands as well.

systemctl enable jenkins

systemctl start jenkins

systemctl status jenkins

- Jenkins will only start if it shows running.
- Now add port 8080 to the security group of Jenkins instance as it is the default port for Jenkins.
- Now copy the public IP / DNS of the instance and add :8080 at the end to access Jenkins.

http://<public-ip>:8080

- Install the necessary plugins and give the user credentials.
- Now the Jenkins dashboard will open.

Configuring Tomcat Servers:

- Now SSH into one Tomcat instance and give the command as follows:

sudo -i

apt update && apt install default-jdk -y

cd /opt

wget https://dlcdn.apache.org/tomcat/tomcat-9/v9.0.91/bin/apache-tomcat-9.0.91.tar.gz

tar -xvzf apache-tomcat-9.0.91.tar.gz

mv apache-tomcat-9.0.91 tomcat

- Now we should configure our tomcat server to access it.

cd tomcat

vi webapps/host-manager/META-INF/context.xml

- Here an editor will open and we must comment on this part.

```
<Valve className="org.apache.catalina.valves.RemoteAddrValve"
       allow="127\.\d+\.\d+\.\d+|::1|0:0:0:0:0:0:0:1" />
  <Manager sessionAttributeValueClassNameFilter="java\.lang\.(?:Boolean|Int
eger|Long|Number|String)|org\.apache\.catalina\.filters\.CsrfPreventionFilte
r\$LruCache(?:\$1)?|java\.util\.(?:Linked)?HashMap"/>
            </Context>
```

```
<!--  <Valve className="org.apache.catalina.valves.RemoteAddrValve"
       allow="127\.\d+\.\d+\.\d+|::1|0:0:0:0:0:0:0:1" /> -->
  <Manager sessionAttributeValueClassNameFilter="java\.lang\.(?:Boolean|Int
eger|Long|Number|String)|org\.apache\.catalina\.filters\.CsrfPreventionFilte
r\$LruCache(?:\$1)?|java\.util\.(?:Linked)?HashMap"/>
            </Context>
```

- As shown the first two lines are commented out, now we have to edit the same part in another file.

vi webapps/manager/META-INF/context.xml

- Now add users to the Tomcat server to access it.

vi conf/tomcat-users.xml

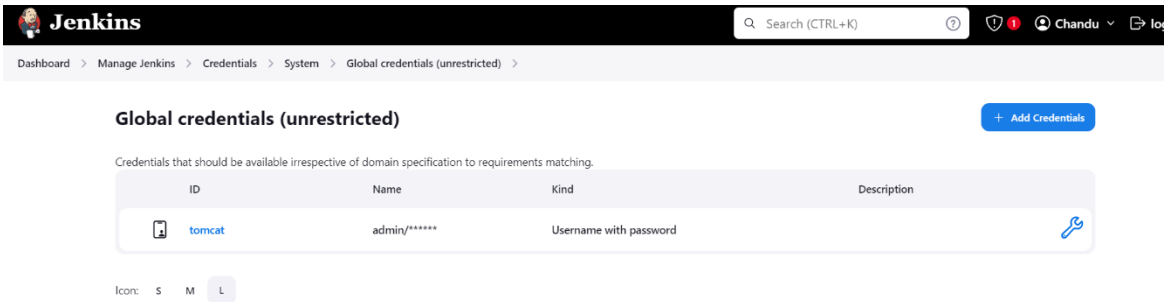
- Here an editor will open and go to the end of the file and add these lines above <tomcat-users>

<role rolename="manager-gui"/>
<role rolename="manager-script"/>
<role rolename="manager-jmx"/>
<role rolename="manager-status"/>
<user username="admin" password="admin" roles="manager-gui, manager-script, manager-jmx, manager-status"/>
<user username="deployer" password="deployer" roles="manager-script"/>
<user username="tomcat" password="s3cret" roles="manager-gui"/>

- Repeat the same process for the other two tomcat instances. And open port number 8080 in security group of tomcat servers.
- As of now our Jenkins and Tomcat servers are ready to use.
- Now move to the Jenkins server and go to

Manage Jenkins > Credentials > System > Global Credentials

- Click on Add credentials and select username with password.
- To deploy war files directly to Tomcat using the pipeline the tomcat role should be <manager-script>, so give username and password credentials as “admin / admin”.
- Give id as ‘tomcat’ before creating and it should look like



- Now go to the dashboard and click on the new item, give whatever you like as the project name, and click on the pipeline.

```
pipeline {
    agent any

    environment {
        GIT_REPO = '<your-git-hub-repo>'
        DEV_TOMCAT_URL = 'http://<DEV-TOMCAT-PUBLIC-IP>:8080/manager/text/deploy?path=/petclinic&update=true'
        TEST_TOMCAT_URL = 'http://<TEST-TOMCAT-PUBLIC-IP>:8080/manager/text/deploy?path=/petclinic1&update=true'
        PROD_TOMCAT_URL = 'http://<PROD-TOMCAT-PUBLIC-IP>/manager/text/deploy?path=/petclinic2&update=true'
        TOMCAT_CREDENTIALS_ID = 'tomcat' // ID for Jenkins credential for Tomcat
    }

    stages {
        stage('Checkout') {
            steps {
                git branch: 'master', url: "${env.GIT_REPO}"
            }
        }

        stage('Build') {
            steps {
                sh 'mvn clean package'
            }
        }

        stage('Deploy to Development') {
            steps {
                script {
                    if (askYesNo("Deploy to Development?")) {
                        deployToTomcat(env.DEV_TOMCAT_URL)
                    } else {
                        deployToTomcat(env.DEV_TOMCAT_URL)
                    }
                }
            }
        }

        stage('Deploy to Testing') {
            steps {
                script {
                    if (askYesNo("Deploy to Testing?")) {
```

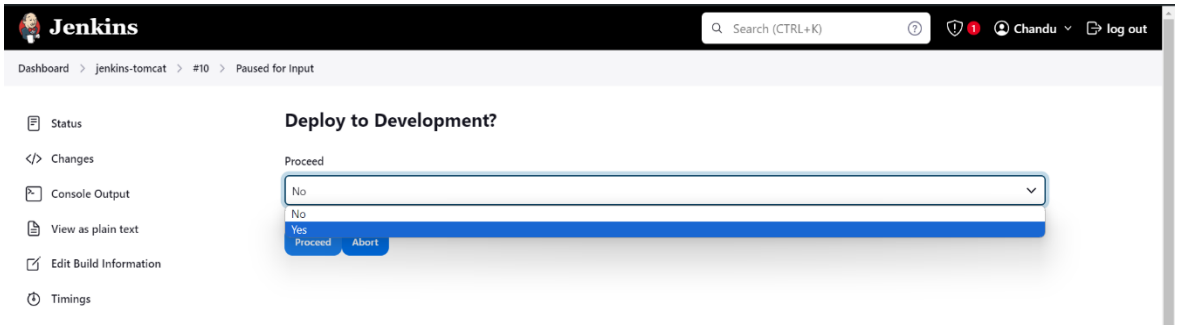
Ln 8, Col 40 2,395 characters

```
                    if (askYesNo("Deploy to Testing?")) {
                        deployToTomcat(env.TEST_TOMCAT_URL)
                    } else {
                        deployToTomcat(env.TEST_TOMCAT_URL)
                    }
                }
            }
        }

        stage('Deploy to Production') {
            steps {
                script {
                    if (askYesNo("Deploy to Production?")) {
                        deployToTomcat(env.PROD_TOMCAT_URL)
                    } else {
                        deployToTomcat(env.PROD_TOMCAT_URL)
                    }
                }
            }
        }
    }

    post {
        always {
            cleanWs()
        }
    }
}

def deployToTomcat(tomcatUrl) {
    withCredentials([usernamePassword(credentialsId: "${env.TOMCAT_CREDENTIALS_ID}", passwordVariable: 'TOMCAT_PASSWORD', usernameVariable: 'TOMCAT_USERNAME')]) {
        sh """
        curl -u "$TOMCAT_USERNAME:$TOMCAT_PASSWORD" -T target/*.war "$tomcatUrl"
        """
    }
}

def askYesNo(String message) {
    input message: message, parameters: [choice(choices: ['No', 'Yes'], description: '', name: 'Proceed')]
    return params.Proceed == 'Yes'
}
```

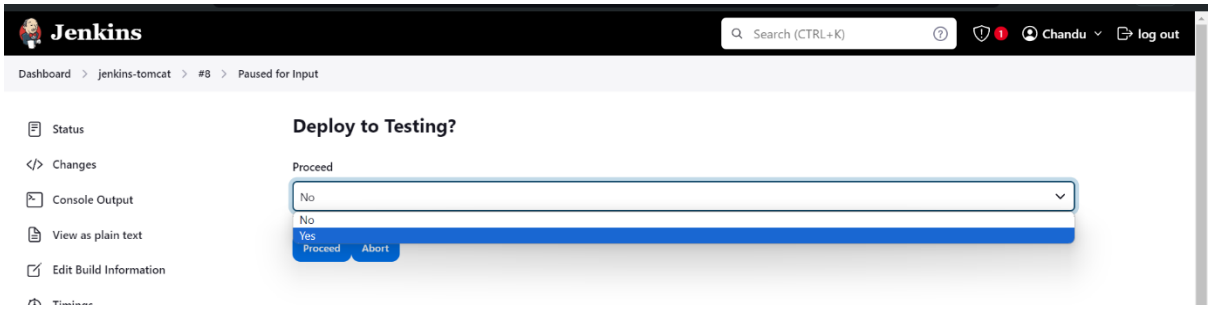
- Click on apply and save. Click on build now.

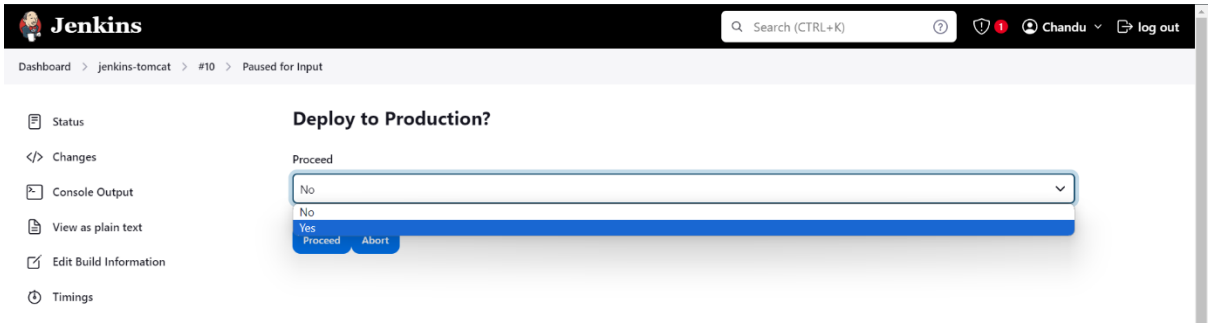
- Now click on the console output of the build and you will see a prompt like this:

- Click on input requested and it will show like this



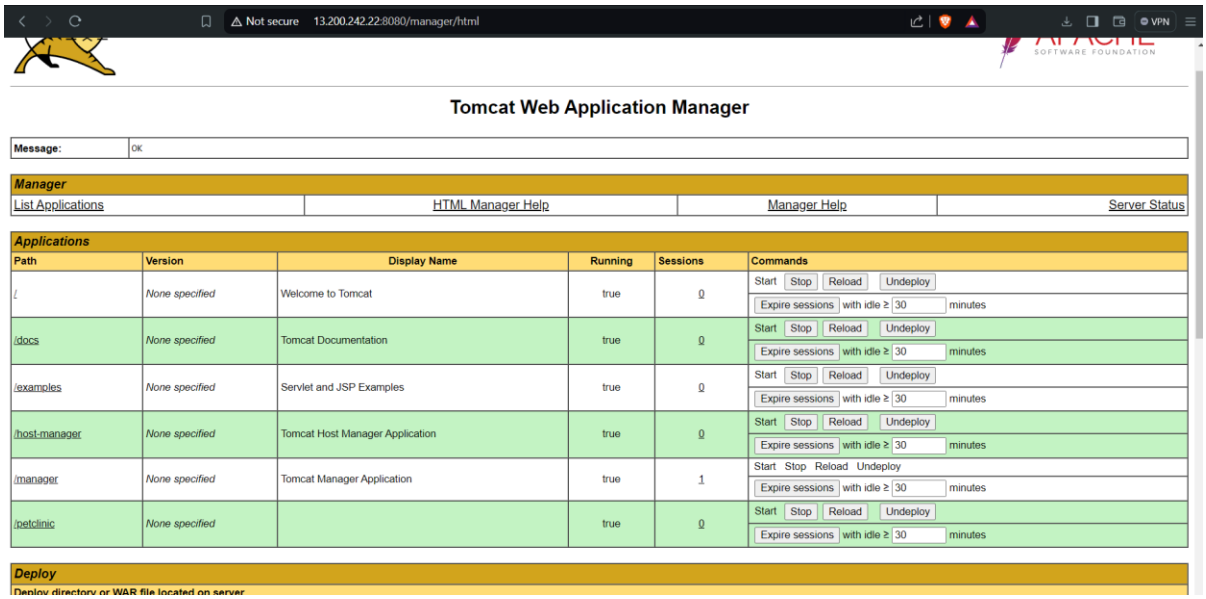
- Click on yes and proceed and we will get two prompts like this for test and prod environments.

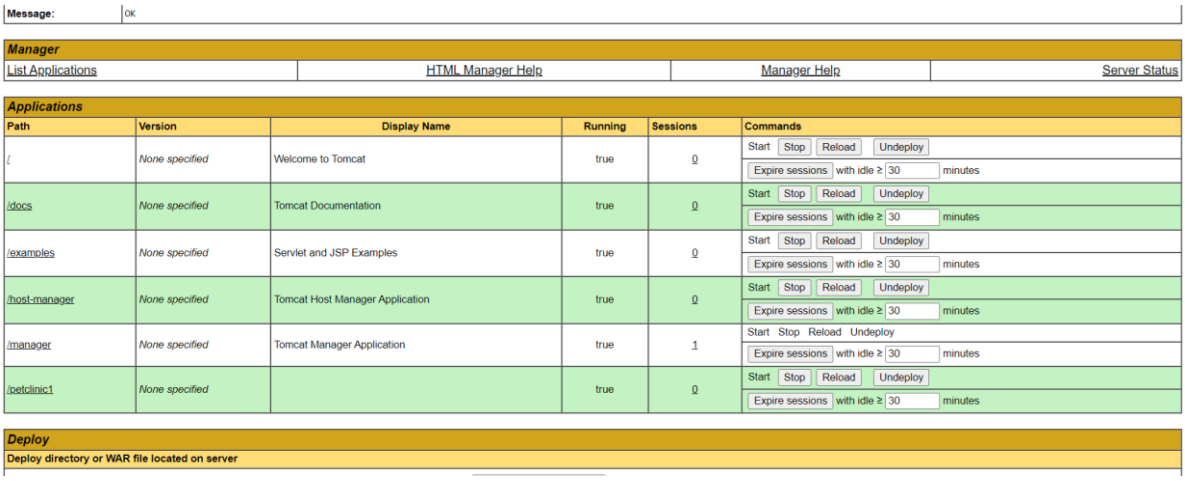


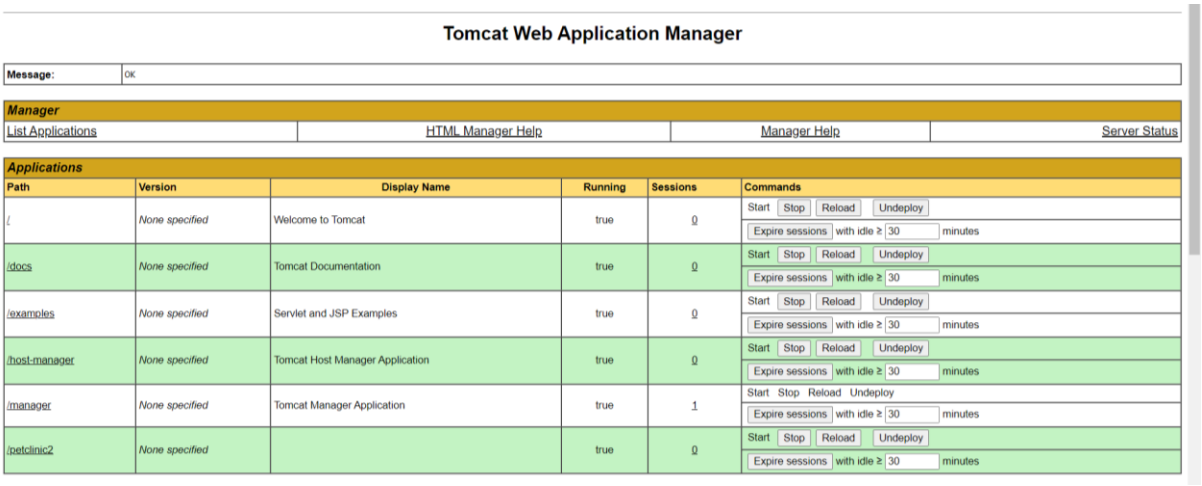


- Now it shows as FINISHED. If logged into tomcat servers using

http://<public-ip-tomcat-instance>:8080







Tomcat Web Application Manager

Message:	OK

Manager

List Applications	HTML Manager Help	Manager Help	Server Status

Applications

Path	Version	Display Name	Running	Sessions	Commands
/	None specified	Welcome to Tomcat	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/docs	None specified	Tomcat Documentation	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/examples	None specified	Servlet and JSP Examples	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/host-manager	None specified	Tomcat Host Manager Application	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/manager	None specified	Tomcat Manager Application	true	1	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/petclinic2	None specified		true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes

- Three applications are deployed into three different servers using pipeline.