

- ② Persistent volume object
- ③ Persistent volume claim object.

## secret and config map:-

namespace → it is only a object.

## project Deployment

① launch 2 instances

- ⇒ one instance should be configured with jenkins
- ⇒ another one instance install tomcat server.

## configurations of tomcat server instance

⇒ \$ wget tomcat.html ⇒ \$ ls

⇒ \$ tar -zxvf apache-tomcat-9.0.85.tar.gz ⇒ \$ ls

⇒ \$ cd conf/ ⇒ \$ ls

⇒ \$ vi tomcat-users.xml

type at last before the closing tag.

```
<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"/>
```

!w2

⇒ \$ cd ..

⇒ \$ cd webapps/ ⇒ \$ cd manager/ ⇒ \$ cd META-INF/ ⇒ \$ ls

⇒ \$ vi context.xml

comment from cookies <!-- space cookies

3 lines. fill manager

space -->

⇒ \$ cd .. ⇒ \$ cd ..

⇒ \$ cd bin/ ⇒ sh startup.sh

⇒ copy this instance public IP ⇒ paste in new tab: 8080

⇒ you get tomcat webpage

## Configurations of jenkins server instance

① Install jenkins ⇒ Install suggested plugins ⇒ start jenkins

② manage jenkins ⇒ tool ⇒ add maven ⇒ provide name - Maven ⇒ apply save

③ manage jenkins ⇒ Available plugins ⇒ search deploy to containers ⇒ install

④ new item ⇒ provide name ⇒ freestyle project ⇒ ok



=> Source code management

select ● git

provide repository url

=> Build steps => Invoke top-level maven target

Select -> maven

goals -> clean install

=> Post build actions: deploy war/ear to a container

=> war/ear files

\*/\*/\*.war

=> context path: paste artifact-id <sup>realizing</sup> ~~of pom~~ pom.xml file

=> add containers

=> credentials -> add -> jenkins -> user name: admin

↓  
select this credential

password: admin

desc: tomcat

→ add

=> tomcat url => copy the tomcat <sup>tab</sup> url & paste here

=> apply save

=> Build now

after build is success then  
go to tomcat page there you get the your  
prod click on -> manager apps -> you have your project  
click on that (artifact id name)

⑤ launch 2 instances

one instance should be configured with jenkins

in another instance install tomcat server & do ~~conf~~

=> configuration of jenkins server instance

\* install jenkins -> install suggested plugins -> start jenkins

in terminal configure this instance as ansible master node

=> \$sudo amazon-linux-extras install ansible2 -y

=> \$vi /etc/ansible/hosts

=> \$vi /etc/ansible/ansible.cfg

=> \$vi /etc/ssh/sshd\_config

=> \$service sshd restart

=> \$vi sudo

=> \$useradd ansible

=> \$passwd ansible 1234 1234

=> \$su - ansible

=> \$ssh-keygen

=> \$ssh private ip

=> \$ssh-copy-id ansible@privateip tomcat instance

=> exit

=> ~~\$vi play.yml~~

configure this



=> \$ vi play.yml

---

- hosts: all

become: true

tasks:

- name: copy jar/war on to tomcat servers

copy:

src: /home/ansible/artifact-id.war  
(testfreshers)

~~apache-tomcat~~

dest: /root/apache-tomcat-9.0.85/webapps

:wq

=> \$ ansible-playbook play.yml

=> configuration of tomcat server instance

\$ wget tomcat-war => \$ls

\$ tar -zxvf apache-tomcat-9.0.85.tar.gz => \$ls

=> \$ cd apache-tomcat-9.0.85

=> \$ cd conf/

=> \$ vi tomcat-users.xml

paste the code -> <sup>it is in</sup> ~~code~~ previous task

:wq

=> \$ cd ..

=> \$ cd webapps/ => \$ cd manager/

=> \$ vi context.xml

comment the cookies

=> \$ cd .. => \$ cd .. => \$ cd bin/ => \$ sh startup.sh

=> \$ useradd ansible

=> \$ passwd ansible

1234

1234

=> \$ vi sudo

=> \$ vi /etc/ssh/sshd\_config

=> \$ service sshd restart

#

20

configure this instance as  
ansible worker node

## In Jenkins UI

- ⇒ manage jenkins → tools → add maven → Maven → apply & Save
- ⇒ manage jenkins → available plugins → Publish over ssh → install
- ⇒ manage jenkins → System → Passphrase : Password of ansible user  
1234

in SSH server → add

name: ansible

hostname: jenkins server private IP

user name: ansible

Advanced: ☒ use password authentication

provide password: 1234

Proxy password: 1234

Test configuration

Success

apply save

⇒ create freestyle project.

⇒ source code management

git  
↳ provide url

⇒ add build step

Invoke top level maven target

→ select Maven version

→ goals : clean install

⇒ add build step:

send files/execute over ssh

source files: \*\*/\*.war

Remove prefix: target

remote directory: /

exec command: ansible-playbook

/home/ansible/play.yml

apply save

⇒ build now

⇒ go to terminal page : manage apps →  
click on your project

### ③ launch one instance

- => install jenkins => connect to browser
- => install ansible -> `sudo amazon-linux-extras install ansible2 -y`
- => install docker -> `{sudo yum install docker -y`  
`{ systemctl start docker`
- => create one user -> `useradd dockeradmin`
- => passwd dockeradmin  
1234  
1234
- => add user to docker group <sup>default group when we start the docker</sup>  
`{usermod -a -G docker dockeradmin`
- => `{vi sudo` > after add permission  
dockeradmin ALL=(ALL) NOPASSWD: ALL  
:wq
- => `{vi /etc/ssh/sshd-config` -> PermitRootLogin <sup>uncomment</sup>  
PasswordAuthentication -> yes
- => install git -> `{sudo yum install git -y`
- => `{su - dockeradmin`
- => vi Dockerfile  
FROM tomcat:9-jre9  
MAINTAINER "sowmyapras@gmail.com"  
COPY ./<artifactId>.war /usr/local/tomcat/webapps
- => In Jenkins UI  
add tool -> maven  
add plugin -> publish over ssh -> install  
system -> Passphrase -> 1234 (dockeradmin <sup>pc</sup> password)  
SSH server -> add  
name: docker  
host name: private ip of jenkins server  
user name: dockeradmin  
advanced: click on password authentication  
passphrase : 1234  
proxy password: 1234  
click on test configuration -> apply save



=> create freestyle job

=> source code management

-> select -> git

=> provide url of your github repository that contains webapplication

=> add buildstep

invoke top level maven target

↳ select -> maven

goals -> clean install

add build step: send file @ execute command over SSH

~~from~~  
-> source file: \*\*/\*.war

remove prefix: target

remote directory: /

exec command:

```
docker stop 2spider;
```

```
docker rm -f 2spider;
```

```
docker image rm -f 2spider;
```

```
cd /home/dockeradmin;
```

```
docker build -t 2spider .
```

=> Post build action

build  
~~send~~ artifact over maven

exec - command : docker run -it -d --name 2spider -p 8090:8080 2spider

apply save

build now.

- ④ Task:- The application should be deployed in worker node containers

### Master node terminal

```
$ sudo su -
$ vi jen.sh
$ sh jen.sh
$ sudo yum install docker -y
$ systemctl start docker
$ sudo yum install git -y
$ sudo yum install java -y
$ systemctl start jenkins

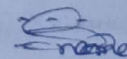
$ useradd ansible
$ passwd ansible
1234
1234
$ sudo usermod -a -G docker ansible
$ sudo amazon-linux-extras install ansible2
-y
$ vi /etc/ansible/hosts
$ vi /etc/ansible/ansible.cfg
$ visudo
$ vi /etc/ssh/sshd-config
$ service sshd restart
$ su - ansible
$ ssh-keygen
$ ssh-copy-id ansible@private1p3 worker node
$ ssh private1p3 slave
$ exit
$ vi Dockerfile
FROM tomcat:9-jre9
MAINTAINER "sowmyayps@gmail.com"
COPY ./testpreshers.war /usr/local/tomcat/webapps
```

### Worker node terminal

```
$ sudo su -
$ useradd ansible
$ passwd ansible
1234
1234
$ sudo yum install docker -y
$ systemctl start docker
$ usermod -a -G docker ansible
$ vi /etc/ssh/sshd-config
uncomment -> PasswordAuthentication
PermitRootLogin
$ service sshd restart
$ visudo
after root permissions
ansible ALL=(ALL) NOPASSWD: ALL
$ su - ansible
$ docker login
$ docker images
$ docker ps
```

images will not be pulled  
we need to give permission

```
$ docker login
$ vi play.yml
```

  
name

```
- name: create image
  hosts: demo
  user: ansible
  connection: ssh
  become: yes
  tasks:
    - name: run the container
      command: docker run
      -it -d --name
      my-container -p 8070:8080
      sowmyayps/my-image
```





=> manage jenkins -> tools -> add maven -> maven -> apply & save

=> manage jenkins -> plugins -> available plugin -> publish over ssh -> install

=> manage jenkins -> system ->

publish over ssh -> passphrase : 1234

add ssh server -> name: docker

hostname: private IP of master (jenkins installed server)

username: ansible

advanced : passphrase: 1234

proxy password: 1234

test configuration

apply save

=> new -> <sup>name of project</sup> deploy -> freestyle project -> ok

=> scm [source code management]

git -> provide url of repository that contains your web application

=> add build step -> Invoke top level maven target

↳ select maven version

↳ goals -> clean install

=> add build step -> send files @ execute command over ssh

name: docker

source files: \*/\*.war

remote prefix: target

remote directory: /

TD sum

copy public IP of workstation

new tab -> Paste public IP: 8070

test flasher

bash fact

=> add build step -> send files @ execute command over ssh

source files: Dockerfile

exec command : cd /home/ansible/

docker build -t soumyajps/my-image .

docker push soumyajps/my-image

docker rmi soumyajps/my-image

command over ssh

=> add build step -> send files @ execute

exec command: ansible-playbooks /home/ansible/play.yml

apply save

build now



for first build that play.yml playbook is ok → use that playbook for first build

⇒ If you give next build then the build is unstable because the previous containers and images are present so first we need to delete it & then build

2) for next build

for next build use this playbooks

we commands in the playbooks

tasks:

- name: stop container

command: docker stop my-container

- name: remove container

command: docker rm -f my-container

- name: remove image

command: docker image rm -f sawmyapp/image

- name: run the container

command: docker run -it -d --name my-container -p 8090:8090

sawmyapp/deploy

⑤ Same <sup>4<sup>th</sup></sup> task but we need to push the versions of images to Dockerhub and the for the appropriate version of image the containers need to run.

⇒ same <sup>4<sup>th</sup></sup> task configuration just change in jenkins job commands & playbooks.

⇒ same first two build steps

2) ~~on~~ third build step → add → send files @ execute over SSH

source files: Dockerfile

exec command:

cd /home/ansible

docker build -t \$JOB\_NAME:v1.\$BUILD\_ID .

docker tag \$JOB\_NAME:v1.\$BUILD\_ID sawmyapp/\$JOB\_NAME:v1.\$BUILD\_ID

docker tag \$JOB\_NAME:v1.\$BUILD\_ID sawmyapp/\$JOB\_NAME:latest

docker push sawmyapp/\$JOB\_NAME:latest

docker rmi \$JOB\_NAME:v1.\$BUILD\_ID sawmyapp/\$JOB\_NAME:v1.\$BUILD\_ID sawmyapp/\$JOB\_NAME:latest

⇒ add build step: send files over SSH

exec command: ansible-playbook /home/ansible/play.yml

vi play.yml

for first build.

---

- name: create image

hosts: demo

user: ansible

connection: ssh

become: ~~yes~~ yes

tasks:

- name:

command:

docker run -it -d --name my-container -p 8080:8080

sowmyaaps/deploy: latest

↳ this should be same as  
jenkins job  
freestyle project

vi play.yml → for next builds add 3 more tasks to remove the  
previous containers.

tasks:

- name: stop container

command: docker stop my-container

- name: remove container

command: docker rm -f my-container

- name: stop image

command: docker image rm -f sowmyaaps/deploy:latest

- name: run the container

command: docker run -it -d --name my-container -p 8080:8080  
sowmyaaps/deploy:latest

run: ansible node private\_ip:8080/askfactip.



## ⑥ Servlet project deployment [monolithic]

launch 3 instances

- 1 → <sup>for</sup> jenkins
- 2 → <sup>for</sup> tomcat
- 3 → <sup>for</sup> database

connect jenkins instance to terminal

- ⇒ sudo su -
- ⇒ vi jen.sh
- Paste scripts
- ⇒ \$ sh jen.sh
- ⇒ \$ sudo yum install java -y
- ⇒ \$ sudo yum install git -y
- ⇒ \$ systemctl start jenkins
- ⇒ ~~connect this instance to~~  
Install all the suggested plugins  
Start jenkins

- ⇒ manage jenkins ⇒ Tool ⇒ add maven ⇒ Maven ⇒ apply save
- ⇒ manage jenkins ⇒ plugins ⇒ available Plugins ⇒ Deploy to container ⇒ install
- ⇒ new item ⇒ provide name ⇒ freestyle project ⇒ OK
- ⇒ source code Management  
git ⇒ provide git url that contains your servlet project
- ⇒ Build step ⇒ add build step ⇒ Invoke top level maven target
  - ↳ select Maven version
  - ↳ goals ⇒ clean install

⇒ add post-build action ⇒ Deploy war/ear to a container

source files: \*/\*/p.war

context-path: todo-app (artifact id)

add container → add → <sup>credentials</sup> username: admin  
password: admin

⇒ select this credential add

⇒ Provide tomcat url that you posted in another tab

- ⇒ connect tomcat instance to terminal
- ⇒ wget tomcat -> \$15
- ⇒ tar -zxvf ~~tomcat~~ apache-tomcat-9.0.85.tar.gz
- ⇒ \$ ls
- ⇒ \$ cd ~~apache-tomcat~~ -9.0.85
- ⇒ \$ cd conf/
- ⇒ vi tomcat-users.xml
- last paste scripts  
<roles>
- ⇒ cd ..
- ⇒ cd webapps/ ⇒ cd manager/ ~~⇒ cd ..~~
- ⇒ \$ cd META-INF ⇒
- ⇒ vi context.xml
- comment from cookies 3 lines
- ⇒ cd .. ⇒ cd .. ⇒ cd ..
- ⇒ cd bin
- ⇒ sh startup.sh
- ⇒ copy the public IP of this instance: 8080  
paste in new tab you get tomcat page

- ⇒ In Amazon management console
- ⇒ search RDS → create database

⇒ select → Easy create

⇒ mysql

⇒ free-tier

⇒ provide Password :  Root admin  
confirm Password  Root admin

⇒ ~~select~~ ~~attach~~ ~~EC2 instance~~ → ~~database~~ Setup EC2 connection

↳ connect to an EC2 compute resource

↳ select the instance that you created for database

⇒ create database

wait until the database is created

⇒ goto github repository [securd project]

in Persistence.xml file provide the database details

⇒ In the place of localhost → Paste end point : 3306 / todo?createDatabaseIfNotExist=true

⇒ user : value admin

⇒ Password : value Root admin

⇒ commit changes

⇒ In RDS :- configure the Inbound & outbound rules for security group

⇒ ec2-rds Security group : Inbound rules : all-traffic anywhere IPv4

outbound rules : all-traffic anywhere IPv4

rds-ec2 Security group : Inbound rules : all-traffic anywhere IPv4

outbound rules : all-traffic anywhere IPv4

Jenkins dashboard

⇒ build now - if build is success

⇒ tomcat page ⇒ manager-app - username : admin  
password : admin

⇒ todo-app ⇒ click on this you will get your Project



connect the database instance to terminal

2) Install mysql => sudo yum install mysql -y

2) \$ mysql -h endpoint of your database -P 3306 -u admin -P  
Provide Password: rootadmin

2) => show databases;

2) use <sup>todo</sup> database-name;

2) \$ show tables;

2) select \* from <sup>customer</sup> table-name;

2)

⑦ Servlet - Project Deployment on Containers [microservice]