

Assignment:22/09/2022

TO BUILD .WAR ON MASTER AND DEPLOY IT ON TWO NODES FOR TWO DIFFERENT PORTS (8080 AND 8090) USING DOCKER-COMPOSE FILE.

STEP1) launch Master and two slave

<input type="checkbox"/>	Master	i-04833e845d3f5fe77		Running	QQ	t2.micro		2/2 checks passed	No alarms	+	us-east-2b	ec2-18-117-135-140.us...	18.117.135.140	-
<input type="checkbox"/>	node1	i-0d436620c5e53cbc3		Running	QQ	t2.micro		2/2 checks passed	No alarms	+	us-east-2b	ec2-3-135-64-139.us-e...	3.135.64.139	-
<input type="checkbox"/>	node2	i-04ca9620782b10932		Running	QQ	t2.micro		2/2 checks passed	No alarms	+	us-east-2b	ec2-3-14-66-72.us-east...	3.14.66.72	-
<input type="checkbox"/>	test2	i-062735b5a834acf57		Stopped	QQ	t2.micro	-	-	No alarms	+	us-east-2b	-	-	-

Step2) write code for stage one

Clone the code from Git repo

Build gameoflife.war on Master and also copy it on both the nodes simultaneously

Script ?

```
1 pipeline {
2   agent none
3   stages {
4     stage ('git-clone-to-master'){
5       agent {
6         node{
7           label 'built-in'
8           customWorkspace '/root/Assignment/'
9         }
10      }
11
12      tools {
13        maven 'maven3.0'
14        jdk 'java1.8'
15      }
16      environment {
17        PATH = "/root/maven/apache-maven-3.8.6/bin:$PATH"
18      }
19      steps {
20        git 'https://github.com/ashrayp18/game-of-life.git'
21        sh "mvn install"
22        sh "cp /root/Assignment/gameoflife-web/target/gameoflife.war /root/tommy/apache-tomcat-9.0.65/webapps/"
23        sh "sudo scp -i /root/test.pem /root/Assignment/gameoflife-web/target/gameoflife.war ec2-user@172.31.20.150:/home/ec2-user/docker/"
24        sh "sudo scp -i /root/test.pem /root/Assignment/gameoflife-web/target/gameoflife.war ec2-user@172.31.17.248:/home/ec2-user/docker/"
25      }
26    }
27  }
```

STEP3)

COPY THE GAMEOFLIFE.WAR to Node1 (DEV ENV)

CLONE THE GIT REPO (DOCKER-COMPOSE FILE) to NODE1

```
27     }
28
29     stage ('running-docker-compose-on-node-1'){
30         agent {
31             node{
32                 label '172.31.20.150'
33                 customWorkspace '/home/ec2-user/docker/'
34             }
35         }
36         steps {
37             git 'https://github.com/ashrayp18/Project-docker-compose.git'
38             sh "sudo systemctl start docker"
39             sh "sudo docker-compose down"
40             sh "sudo docker-compose up -d"
41         }
42     }
```

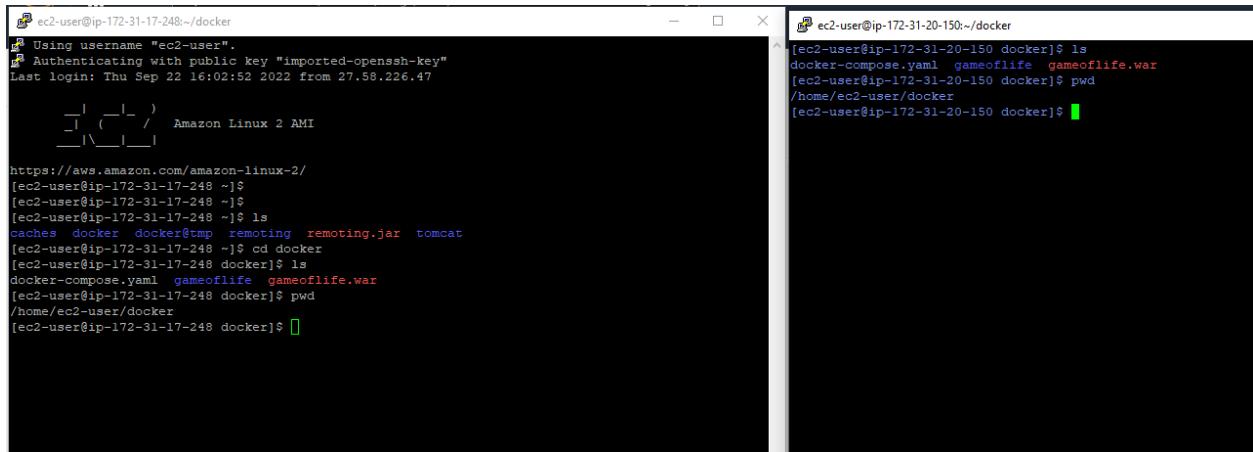
STEP4)

COPY THE GAMEOFLIFE.WAR to Node2 (QA ENV)

CLONE THE GIT REPO (DOCKER-COMPOSE FILE) to NODE2

```
39         sh "sudo docker-compose down"
40         sh "sudo docker-compose up -d"
41     }
42
43 }
44 stage ('running-docker-compose-on-node-2'){
45     agent {
46         node{
47             label '172.31.17.248'
48             customWorkspace '/home/ec2-user/docker/'
49         }
50     }
51     steps {
52         git 'https://github.com/ashrayp18/Project-docker-compose.git'
53         sh "sudo systemctl start docker"
54         sh "sudo docker-compose down"
55         sh "sudo docker-compose up -d"
```

STEP5) LOGIN AND CHECK ON NODES IF FILES ARE PRESENT



The image shows two terminal windows side-by-side. The left window is titled 'ec2-user@ip-172-31-17-248:~/docker' and shows an SSH login session for 'ec2-user' on an Amazon Linux 2 AMI. The user runs 'ls' and lists files: 'docker-compose.yaml', 'gameoflife', and 'gameoflife.war'. The right window is titled 'ec2-user@ip-172-31-20-150:~/docker' and shows the same user running 'ls' and 'pwd' to confirm the file presence and current directory.

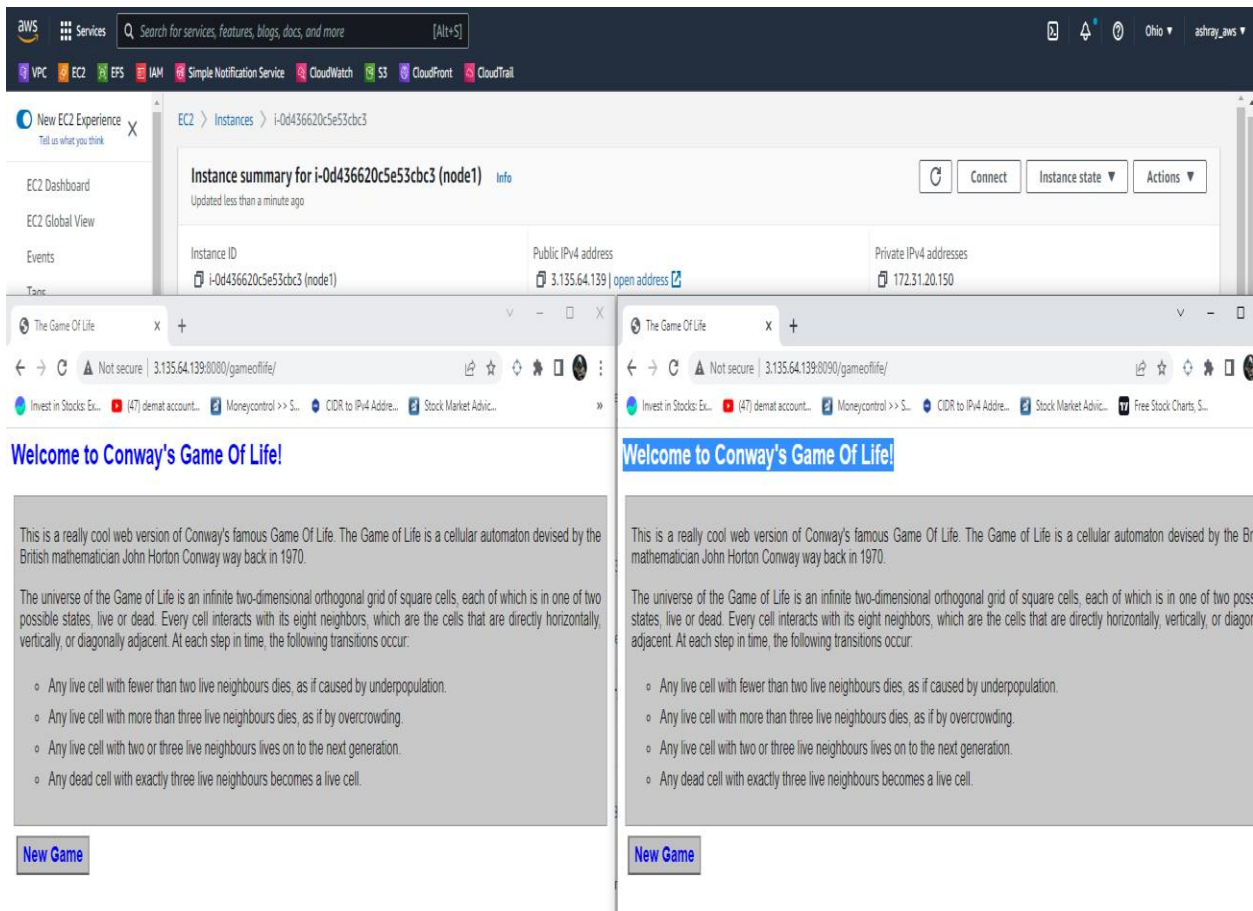
```
ec2-user@ip-172-31-17-248:~/docker
Using username "ec2-user".
Authenticating with public key "imported-openssh-key"
Last login: Thu Sep 22 16:02:52 2022 from 27.58.226.47

 _ _ _ _ _
| | | | |
|_|_|_|_|_|_ Amazon Linux 2 AMI

https://aws.amazon.com/amazon-linux-2/
[ec2-user@ip-172-31-17-248 ~]$
[ec2-user@ip-172-31-17-248 ~]$
[ec2-user@ip-172-31-17-248 ~]$ ls
cache docker docker@tmp remotimg.jar tomcat
[ec2-user@ip-172-31-17-248 ~]$ cd docker
[ec2-user@ip-172-31-17-248 docker]$ ls
docker-compose.yaml gameoflife gameoflife.war
[ec2-user@ip-172-31-17-248 docker]$ pwd
/home/ec2-user/docker
[ec2-user@ip-172-31-17-248 docker]$

ec2-user@ip-172-31-20-150:~/docker
[ec2-user@ip-172-31-20-150 docker]$ ls
docker-compose.yaml gameoflife gameoflife.war
[ec2-user@ip-172-31-20-150 docker]$ pwd
/home/ec2-user/docker
[ec2-user@ip-172-31-20-150 docker]$
```

STEP6) CHECK THE PUBLIC IP FOR NODE ONE ON PORT 8080 and 8090



The image displays the AWS Management Console for an EC2 instance named 'i-0d436620c5e53cbc3 (node1)'. The instance's public IPv4 address is 3.135.64.139. Below the console, two browser windows show the 'Welcome to Conway's Game Of Life!' page. The left browser window is at the address 3.135.64.139:8080/gameoflife/, and the right browser window is at 3.135.64.139:8090/gameoflife/. Both pages show the same content, including a description of the Game of Life and a 'New Game' button.

AWS Management Console:

- Instance ID: i-0d436620c5e53cbc3 (node1)
- Public IPv4 address: 3.135.64.139
- Private IPv4 addresses: 172.31.20.150

Browser Windows:

- Left window: 3.135.64.139:8080/gameoflife/
- Right window: 3.135.64.139:8090/gameoflife/

Page Content:

Welcome to Conway's Game Of Life!

This is a really cool web version of Conway's famous Game Of Life. The Game of Life is a cellular automaton devised by the British mathematician John Horton Conway way back in 1970.

The universe of the Game of Life is an infinite two-dimensional orthogonal grid of square cells, each of which is in one of two possible states, live or dead. Every cell interacts with its eight neighbors, which are the cells that are directly horizontally, vertically, or diagonally adjacent. At each step in time, the following transitions occur:

- Any live cell with fewer than two live neighbours dies, as if caused by underpopulation.
- Any live cell with more than three live neighbours dies, as if by overcrowding.
- Any live cell with two or three live neighbours lives on to the next generation.
- Any dead cell with exactly three live neighbours becomes a live cell.

[New Game](#)

STEP7) CHECK THE PUBLIC IP FOR NODE ONE ON PORT 8080 and 8090

The screenshot displays the AWS Management Console interface. The top navigation bar shows the AWS logo, a search bar, and the user's profile. The left sidebar contains navigation links for various AWS services. The main content area shows the 'Instances' page for the EC2 instance 'i-04ca9620782b10932'. The instance summary includes the Instance ID, Public IPv4 address (3.14.66.72), and Private IPv4 addresses (172.31.17.248). Below the console, two browser windows are open, both displaying 'The Game Of Life' web application. The left browser window shows the application running on the public IP 3.14.66.72 on port 8080, and the right browser window shows it running on the public IP 3.14.66.72 on port 8090. Both applications display a welcome message and a 'New Game' button.

The screenshot displays the AWS CodePipeline console interface. The top navigation bar shows the AWS logo, a search bar, and the user's profile. The left sidebar contains navigation links for various AWS services. The main content area shows the 'DOCKER-COMPOSE-ON-TWO-NODE-PIPELINE' pipeline. The pipeline status is 'Succeeded' and it was started by user 'ashray'. The pipeline diagram shows four stages: 'Start', 'git-clone-to-master', 'running-docker-compose-on-no...', and 'running-docker-compose-on-no...'. The 'running-docker-compose-on-no...' stage is currently running. Below the pipeline diagram, the execution details for the 'running-docker-compose-on-no-2' stage are shown, including the command 'https://github.com/ashray18/Project-docker-compose.git' and the output of the 'sudo docker-compose up -d' command.

