

Project 3 – To create a Docker ECS based solution service in AWS Public Cloud using managed container service

Name: Vishwanaath Gopalakrishnan

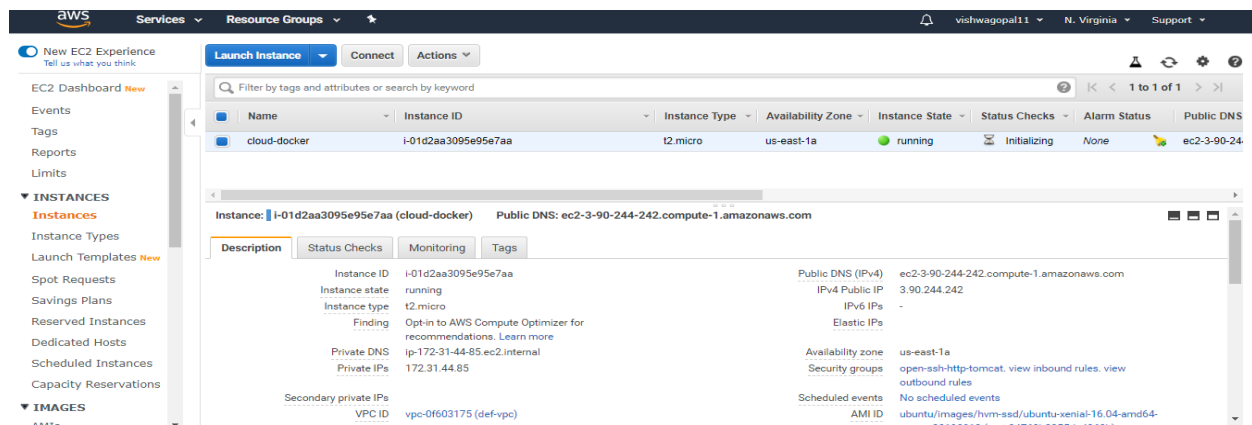
Batch: PGPCC_OCT19A

Solution steps:

- Download the WAR file from <https://storage.googleapis.com/skl-training/aws-codelabs/aws-intro/HelloWorld.war>
- Web application needs to be packaged as a Docker image running on Tomcat having JRE8 - you will have to write a Dockerfile
- Once the image is created, run and verify image by accessing web application using ec2 instance public-ip
- Sign up for docker hub and create public repository.
- Tag the image appropriately and push to Docker Hub Repository.
- Using ECS Fargate create a cluster, task and service(s)

Step 1: Create an EC2 instance and install docker files in it and ensure that the setup is working as expected

Create an Ubuntu EC2 instance using the 7 Step workflow and open the ports 22, 80 and 8080 to access ssh, http and tomcat servers



Pem file: gl-docker.pem

Ssh into the pem file to ensure that is connected from the terminal

```

ubuntu@ip-172-31-44-85: ~
ubuntu@DESKTOP-49DC5BB:/mnt/c/users/vishw/downloads$ pwd
/mnt/c/users/vishw/downloads
ubuntu@DESKTOP-49DC5BB:/mnt/c/users/vishw/downloads$ sudo su
[sudo] password for ubuntu:
root@DESKTOP-49DC5BB:/mnt/c/users/vishw/downloads# clear
root@DESKTOP-49DC5BB:/mnt/c/users/vishw/downloads# pwd
/mnt/c/users/vishw/downloads
root@DESKTOP-49DC5BB:/mnt/c/users/vishw/downloads# ls -l *.pem
-r-xr-xr-x 1 ubuntu ubuntu 1692 Dec 25 11:26 gl-docker.pem
-r-xr-xr-x 1 ubuntu ubuntu 1696 Dec 12 02:48 gl-vish3.pem
-r-xr-xr-x 1 ubuntu ubuntu 1692 Nov 17 15:09 gl.pem
-rwxrwxrwx 1 ubuntu ubuntu 1692 Nov 16 07:17 'glpem1 (1).pem'
-rwxrwxrwx 1 ubuntu ubuntu 1692 Nov 16 07:23 'glpem1 (2).pem'
-r-xr-xr-x 1 ubuntu ubuntu 1692 Nov 16 11:53 greatlearning.pem
-r-xr-xr-x 1 ubuntu ubuntu 1696 Nov 17 21:13 owncloud.pem
root@DESKTOP-49DC5BB:/mnt/c/users/vishw/downloads# chmod 400 gl-docker.pem
root@DESKTOP-49DC5BB:/mnt/c/users/vishw/downloads# ssh -i gl-docker.pem ubuntu@3.90.244.242
The authenticity of host '3.90.244.242 (3.90.244.242)' can't be established.
ECDSA key fingerprint is SHA256:D4hLkj28zHAiaFovCWldYy9p0cRPf8vISCNVYwSvvaw.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '3.90.244.242' (ECDSA) to the list of known hosts.
Welcome to Ubuntu 16.04.6 LTS (GNU/Linux 4.4.0-1092-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

0 packages can be updated.
0 updates are security updates.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-172-31-44-85:~$

```

Run the below commands

- `sudo apt update` - to all the repositories
- `sudo apt install docker.io` – to install docker repositories

ubuntu@ip-172-31-44-85: ~

```
ubuntu@ip-172-31-44-85:~$ sudo apt update
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu xenial InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu xenial-updates InRelease [109 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu xenial-backports InRelease [107 kB]
Get:4 http://us-east-1.ec2.archive.ubuntu.com/ubuntu xenial/universe amd64 Packages [7,532 kB]
Get:5 http://security.ubuntu.com/ubuntu xenial-security InRelease [109 kB]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu xenial/universe Translation-en [4,354 kB]
Get:7 http://us-east-1.ec2.archive.ubuntu.com/ubuntu xenial/multiverse amd64 Packages [144 kB]
Get:8 http://us-east-1.ec2.archive.ubuntu.com/ubuntu xenial/multiverse Translation-en [106 kB]
Get:9 http://us-east-1.ec2.archive.ubuntu.com/ubuntu xenial-updates/main amd64 Packages [1,082 kB]
Get:10 http://us-east-1.ec2.archive.ubuntu.com/ubuntu xenial-updates/main Translation-en [416 kB]
Get:11 http://us-east-1.ec2.archive.ubuntu.com/ubuntu xenial-updates/universe amd64 Packages [771 kB]
Get:12 http://us-east-1.ec2.archive.ubuntu.com/ubuntu xenial-updates/universe Translation-en [324 kB]
Get:13 http://us-east-1.ec2.archive.ubuntu.com/ubuntu xenial-updates/multiverse amd64 Packages [16.8 kB]
Get:14 http://us-east-1.ec2.archive.ubuntu.com/ubuntu xenial-updates/multiverse Translation-en [8,468 B]
Get:15 http://us-east-1.ec2.archive.ubuntu.com/ubuntu xenial-backports/main amd64 Packages [7,280 B]
Get:16 http://us-east-1.ec2.archive.ubuntu.com/ubuntu xenial-backports/main Translation-en [4,456 B]
Get:17 http://us-east-1.ec2.archive.ubuntu.com/ubuntu xenial-backports/universe amd64 Packages [8,064 B]
Get:18 http://us-east-1.ec2.archive.ubuntu.com/ubuntu xenial-backports/universe Translation-en [4,328 B]
Get:19 http://security.ubuntu.com/ubuntu xenial-security/main amd64 Packages [797 kB]
Get:20 http://security.ubuntu.com/ubuntu xenial-security/main Translation-en [306 kB]
Get:21 http://security.ubuntu.com/ubuntu xenial-security/universe amd64 Packages [466 kB]
Get:22 http://security.ubuntu.com/ubuntu xenial-security/universe Translation-en [192 kB]
Get:23 http://security.ubuntu.com/ubuntu xenial-security/multiverse amd64 Packages [5,728 B]
Get:24 http://security.ubuntu.com/ubuntu xenial-security/multiverse Translation-en [2,708 B]
Fetched 16.9 MB in 3s (5,010 kB/s)
Reading package lists... Done
Building dependency tree
Reading state information... Done
61 packages can be upgraded. Run 'apt list --upgradable' to see them.
ubuntu@ip-172-31-44-85:~$ sudo apt install docker.io
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  bridge-utils cgroupfs-mount containerd pigz runc ubuntu-fan
Suggested packages:
  mountall aufs-tools debootstrap docker-doc rinse zfs-fuse | zfsutils
The following NEW packages will be installed:
  bridge-utils cgroupfs-mount containerd docker.io pigz runc ubuntu-fan
0 upgraded, 7 newly installed, 0 to remove and 61 not upgraded.
Need to get 52.2 MB of archives.
After this operation, 257 MB of additional disk space will be used.
Do you want to continue? [Y/n] Y
```

Docker installed successfully and able to see the client and server details as below

Add ubuntu instance to the docker

```

REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
ubuntu@ip-172-31-44-85:/var/lib/docker$ sudo user mod -aG docker ubuntu
sudo: user: command not found
ubuntu@ip-172-31-44-85:/var/lib/docker$ sudo usermod -aG docker ubuntu
ubuntu@ip-172-31-44-85:/var/lib/docker$ sudo docker version
Client:
 Version:           18.09.7
 API version:        1.39
 Go version:         go1.10.4
 Git commit:         2d0083d
 Built:              Fri Aug 16 14:19:38 2019
 OS/Arch:            linux/amd64
 Experimental:        false

Server:
 Engine:
  Version:           18.09.7
  API version:        1.39 (minimum version 1.12)
  Go version:         go1.10.4
  Git commit:         2d0083d
  Built:              Thu Aug 15 15:12:41 2019
  OS/Arch:            linux/amd64
  Experimental:        false
ubuntu@ip-172-31-44-85:/var/lib/docker$ sudo chown ubuntu:ubuntu -R /opt
ubuntu@ip-172-31-44-85:/var/lib/docker$ ls -al /opt
total 12
drwxr-xr-x  3 ubuntu ubuntu 4096 Dec 25 16:52 .
drwxr-xr-x 23 root    root   4096 Dec 25 16:27 ..
drwx--x--x  4 ubuntu ubuntu 4096 Dec 25 16:52 containerd
ubuntu@ip-172-31-44-85:/var/lib/docker$

```

```

Last login: Wed Dec 25 16:47:12 2019 from 73.61.110.39
ubuntu@ip-172-31-44-85:~$ docker version
Client:
 Version:           18.09.7
 API version:        1.39
 Go version:         go1.10.4
 Git commit:         2d0083d
 Built:              Fri Aug 16 14:19:38 2019
 OS/Arch:            linux/amd64
 Experimental:        false

Server:
 Engine:
  Version:           18.09.7
  API version:        1.39 (minimum version 1.12)
  Go version:         go1.10.4
  Git commit:         2d0083d
  Built:              Thu Aug 15 15:12:41 2019
  OS/Arch:            linux/amd64
  Experimental:        false
ubuntu@ip-172-31-44-85:~$ ps -ef | grep [d]ocker
root      13341      1   0 16:52 ?        00:00:00 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock
ubuntu@ip-172-31-44-85:~$ docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
ubuntu@ip-172-31-44-85:~$ docker ps -a
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS              NAMES
ubuntu@ip-172-31-44-85:~$

```

Testing if the docker is connected:

```

ubuntu@ip-172-31-44-85:/$ docker run --rm busybox /bin/echo "Hello world"
Unable to find image 'busybox:latest' locally
latest: Pulling from library/busybox
322973677ef5: Pull complete
Digest: sha256:1828edd60c5efd34b2bf5dd3282ec0cc04d47b2ff9caa0b6d4f07a21d1c08084
Status: Downloaded newer image for busybox:latest
Hello world
ubuntu@ip-172-31-44-85:/$ docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
busybox              latest             b534869c81f0       3 weeks ago        1.22MB
ubuntu@ip-172-31-44-85:/$ docker ps -a
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS              NAMES
ubuntu@ip-172-31-44-85:/$

```

Step 2: Download Tomcat application image with GRE8 from docker hub into the ubuntu instance and create a container and run it in detached mode mapped to 8080 port

Run the command `docker run -d -p 80:8080 tomcat:jre8` and test by accessing the default tomcat page by accessing ipv4 address

The screenshot shows a terminal window on an Ubuntu instance with IP 172-31-44-85. The user runs the command `docker run -d -p 80:8080 tomcat:jre8`. The terminal output shows the Docker image being pulled from the library, with various layers being pulled and digested. The status indicates that a newer image for `tomcat:jre8` was downloaded. The user then runs `docker ps -a`, showing a container named `clever_khorana` with ID `69646449105a` in a 'Up' state. The user then runs `docker stop clever_khorana`, and the container status changes to 'Exited (143) 7 seconds ago'. Finally, the user runs `docker ps -a` again, showing the container in the 'Exited' state.

Below the terminal window, a web browser shows the Apache Tomcat 8.5.41 default page. The page has a navigation bar with links to Home, Documentation, Configuration, Examples, Wiki, and Mailing Lists. The main content area features a green banner with the message "If you're seeing this, you've successfully installed Tomcat. Congratulations!". Below this, there is a section for "Recommended Reading" with links to "Security Considerations HOW-TO", "Manager Application HOW-TO", and "Clustering/Session Replication HOW-TO". There are also buttons for "Server Status", "Manager App", and "Host Manager". A "Developer Quick Start" section provides links to "Tomcat Setup", "First Web Application", "Realms & AAA", "JDBC DataSources", "Examples", "Servlet Specifications", and "Tomcat Versions". At the bottom, there are three boxes: "Managing Tomcat" (with a note about security and a link to "Read more..."), "Documentation" (with links to "Tomcat 8.5 Documentation", "Tomcat 8.5 Configuration", and "Tomcat Wiki"), and "Getting Help" (with a link to "FAQ and Mailing Lists" and a note about mailing lists).

Tomcat default page works as expected from the IP address of the EC2 instance 3.90.244.242

Step 3: Inject the application HelloWorld.war and create an image

Download the WAR file from <https://storage.googleapis.com/skl-training/aws-codelabs/aws-intro/HelloWorld.war>

Web application needs to be packaged as a Docker image running on Tomcat having JRE8 - you will have to write into a Dockerfile

- Create a directory as helloworld
- Use **wget** command to copy the warfile into the directory
- Write a docker file for the modifications to the war file and name it as Dockerfile
- Use docker build command to build an image and check if the image has been created successfully

```
Status: Downloaded newer image for tomcat:jre8
69646449105a2765cc6a744196e392e9ed319599f8e8fb267cd586
ubuntu@ip-172-31-44-85:/$ docker ps -a
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS               NAMES
69646449105a        tomcat:jre8        "catalina.sh run"   About a minute ago   Up About a minute   0.0.0.0:80->8080/tcp   clever_khorana
ubuntu@ip-172-31-44-85:/$ docker stop clever_khorana
clever_khorana
ubuntu@ip-172-31-44-85:/$ docker ps -a
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS               NAMES
69646449105a        tomcat:jre8        "catalina.sh run"   5 minutes ago       Exited (143) 7 seconds ago                       clever_khorana
ubuntu@ip-172-31-44-85:/$ cd /opt/
ubuntu@ip-172-31-44-85:/opt$ ls -al
total 12
drwxr-xr-x  3 ubuntu ubuntu 4096 Dec 25 16:52 .
drwxr-xr-x 23 root    root   4096 Dec 25 16:27 ..
drwx--x--x  4 ubuntu ubuntu 4096 Dec 25 16:52 containerd
ubuntu@ip-172-31-44-85:/opt$
ubuntu@ip-172-31-44-85:/opt$ mkdir helloworld
ubuntu@ip-172-31-44-85:/opt/helloworld$ wget https://storage.googleapis.com/skl-training/aws-codelabs/aws-intro/HelloWorld.war
--2019-12-25 19:17:16--  https://storage.googleapis.com/skl-training/aws-codelabs/aws-intro/HelloWorld.war
Resolving storage.googleapis.com (storage.googleapis.com)... 172.217.15.80, 2607:f8b0:4004:809::2010
Connecting to storage.googleapis.com (storage.googleapis.com)|172.217.15.80|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 9852807 (9.4M) [application/octet-stream]
Saving to: 'HelloWorld.war'

HelloWorld.war                               100%[=====>]  9.40M  --.-KB/s  in 0.1s

2019-12-25 19:17:17 (92.9 MB/s) - 'HelloWorld.war' saved [9852807/9852807]

ubuntu@ip-172-31-44-85:/opt/helloworld$ ls -al
total 9632
drwxrwxr-x  2 ubuntu ubuntu  4096 Dec 25 19:17 .
drwxr-xr-x  4 ubuntu ubuntu  4096 Dec 25 19:16 ..
-rw-rw-r--  1 ubuntu ubuntu 9852807 Aug 11 09:08 HelloWorld.war
ubuntu@ip-172-31-44-85:/opt/helloworld$ nano Dockerfile
```

```
ubuntu@ip-172-31-44-85:/opt/helloworld$ nano Dockerfile
ubuntu@ip-172-31-44-85:/opt/helloworld$ cat Dockerfile
FROM tomcat:jre8
MAINTAINER Vishwanaath Gopalakrishnan
COPY HelloWorld.war /usr/local/tomcat/webapps/
ubuntu@ip-172-31-44-85:/opt/helloworld$ ls -l
total 9628
-rw-rw-r--  1 ubuntu ubuntu  102 Dec 25 19:27 Dockerfile
-rw-rw-r--  1 ubuntu ubuntu 9852807 Aug 11 09:08 HelloWorld.war
ubuntu@ip-172-31-44-85:/opt/helloworld$
```



```

ubuntu@ip-172-31-44-85:/opt/helloworld$ ls -l
total 9628
-rw-rw-r-- 1 ubuntu ubuntu 102 Dec 25 19:27 Dockerfile
-rw-rw-r-- 1 ubuntu ubuntu 9852807 Aug 11 09:08 HelloWorld.war
ubuntu@ip-172-31-44-85:/opt/helloworld$ docker build -t helloworld .
Sending build context to Docker daemon 9.855MB
Step 1/3 : FROM tomcat:jre8
--> 3639174793ba
Step 2/3 : MAINTAINER Vishwanaath Gopalakrishnan
--> Running in 6946293c45d7
Removing intermediate container 6946293c45d7
--> 51817502ae54
Step 3/3 : COPY HelloWorld.war /usr/local/tomcat/webapps/
--> 871be81c311a
Successfully built 871be81c311a
Successfully tagged helloworld:latest
ubuntu@ip-172-31-44-85:/opt/helloworld$ docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
helloworld           latest             871be81c311a       About a minute ago 473MB
busybox              latest             b534869c81f0       3 weeks ago        1.22MB
tomcat               jre8              3639174793ba       7 months ago       463MB
ubuntu@ip-172-31-44-85:/opt/helloworld$

```

The screenshot displays the AWS Management Console interface. The top navigation bar includes the AWS logo, 'Services', 'Resource Groups', and user information. The left sidebar shows the 'EC2 Dashboard' and various navigation links. The main content area shows a list of EC2 instances with one instance, 'cloud-docker', highlighted. Below the list, the details for this instance are shown, including its ID, type, and public IP address. At the bottom of the screenshot, a terminal window shows the command to run the 'helloworld' container using Docker.

Name	Instance ID	Instance Type	Availability Zone	Instance State
cloud-docker	i-01d2aa3095e95e7aa	t2.micro	us-east-1a	running

```

Instance: i-01d2aa3095e95e7aa (cloud-docker) Public DNS: ec2-3-90-244-242.compute-1.amazonaws.com
Description Status Checks Monitoring Tags
Instance ID i-01d2aa3095e95e7aa Public DNS (IPv4) ec2-3-90-244-242.compute-1.amazonaws.com
Instance state running IPv4 Public IP 3.90.244.242

```

```

ubuntu@ip-172-31-44-85:/opt/helloworld$ docker run -d -p 80:8080 helloworld
b5ba45f8070758334c4eabd152ea1c815b159b70aad0194bc80bb0592f99998
ubuntu@ip-172-31-44-85:/opt/helloworld$

```

Access the public IP address to see if the changes made are in effect

ubuntu@ip-172-31-44-85: /opt/helloworld


ubuntu@ip-172-31-44-85: /opt/helloworld\$ nm -f Dockerfile

Instances | EC2 Management Console console.aws.amazon.com

Home Documentation Configuration Examples Wiki Mailing Lists Find Help

Apache Tomcat/8.5.41

If you're seeing this, you've successfully installed Tomcat. Congratulations!

 Recommended Reading:

- [Security Considerations HOW-TO](#)
- [Manager Application HOW-TO](#)
- [Clustering/Session Replication HOW-TO](#)

Server Status
Manager App
Host Manager

Developer Quick Start

- [Tomcat Setup](#)
- [First Web Application](#)
- [Realms & AAA](#)
- [JDBC Data Sources](#)
- [Examples](#)
- [Servlet Specifications](#)
- [Tomcat Versions](#)

gl-docker.pem Project03_Deployi...pdf Padi-Kolam4-300x...jpg Show all

```
helloworld latest 871be81c311a About a minute ago 473MB
busybox latest b534869c81f0 3 weeks ago 1.22MB
tomcat jre8 3639174793ba 7 months ago 463MB
ubuntu@ip-172-31-44-85:/opt/helloworld$ docker run -d -p 80:8080 helloworld
fb5ba45f8070758334c4eabd152ea1c815b159b70aad0194bc80bb0592f99998
ubuntu@ip-172-31-44-85:/opt/helloworld$
```

Instances | Olympus - | Project03_ | dockerhub | docker/wh | Official Im | Welcome |

Not secure | 3.90.244.242/HelloWorld/#

Welcome!

If you are reading this message then the installation has gone well and the application is running. Congratulations!!
You may want to sign in using the credentials that you see below the text boxes to experience voice enabled services from Google.

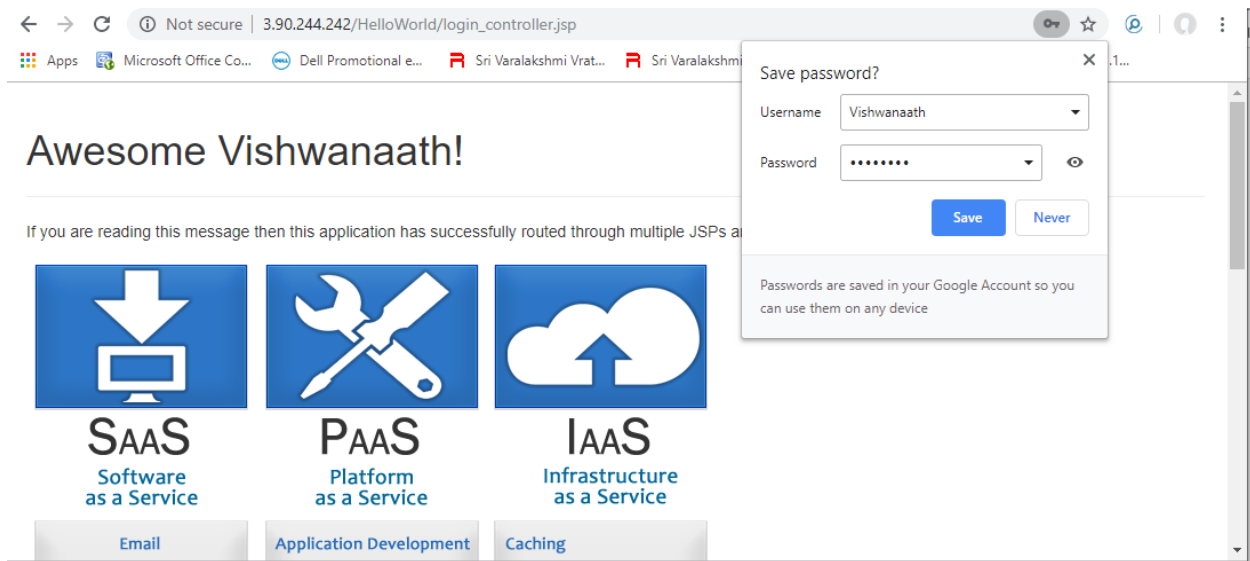
Login

Type in your first name

Password

The password is hard coded as admin123

[Go ahead, try it!](#)



Validation of the image and container instances and its working in CLI:

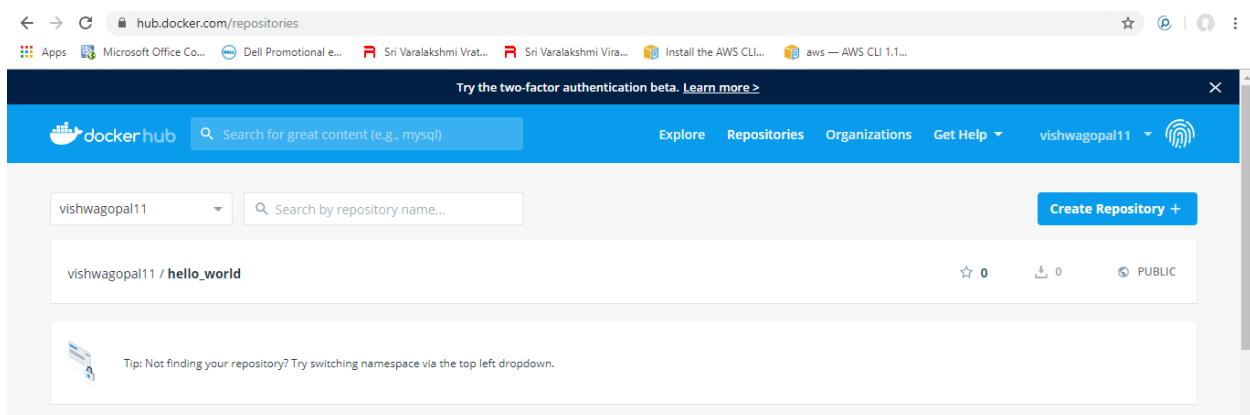
```
ubuntu@ip-172-31-44-85:/opt/helloworld$ docker run -d -p 80:8080 helloworld
db5ba45f8070758334c4eabd152ea1c815b159b70aad0194bc80bb0592f99998
ubuntu@ip-172-31-44-85:/opt/helloworld$ docker images
No command 'docker' found, did you mean:
  Command 'docker' from package 'docker.io' (universe)
docker: command not found
ubuntu@ip-172-31-44-85:/opt/helloworld$ docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
helloworld    latest    871be81c311a   16 minutes ago 473MB
busybox       latest    b534869c81f0   3 weeks ago   1.22MB
tomcat        jre8      3639174793ba   7 months ago  463MB
ubuntu@ip-172-31-44-85:/opt/helloworld$ docker ps -a
CONTAINER ID   IMAGE      COMMAND                  CREATED    STATUS      PORTS                               NAMES
db5ba45f8070   helloworld "catalina.sh run"        12 minutes ago Up 12 minutes 0.0.0.0:80->8080/tcp   vigorous_robinson
69646449105a   tomcat:jre8 "catalina.sh run"        40 minutes ago Exited (143) 34 minutes ago   clever_khorana
```

The application is working as expected from the EC2 container instance based on the changes amde

Step 4:

- Sign up for docker hub and create public repository
- Tag the image appropriately and push to Docker Hub Repository

Created a docker hub account and a public repository hello_world in it



Login into Docker through Ubuntu CLI and push the image file into the repository using docker tag and docker push commands

```
docker tag local-image:tagname new-repo:tagname
```

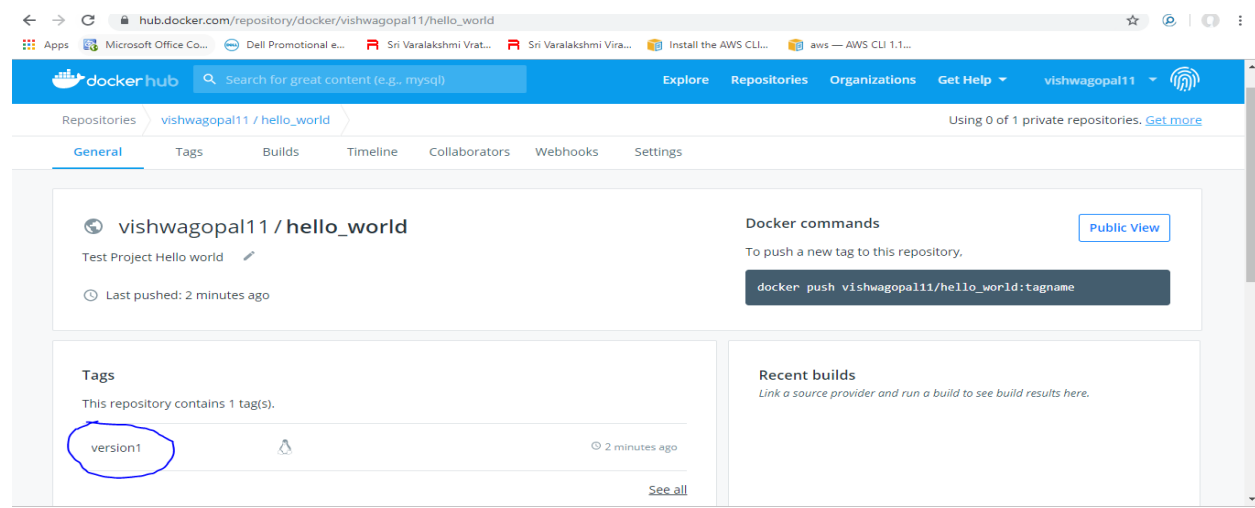
```
docker push new-repo:tagname
```

```
docker push vishwagopal11/hello_world:tagname
```

The image is tagged and pushed to docker hub repository and confirmed as per the validation in the docker hub repository

```
ubuntu@ip-172-31-44-85:/opt/helloworld$ docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
smartfin            latest             1b479b227b9a       About an hour ago  166MB
helloworld          latest             871be81c311a       2 hours ago       473MB
vishwagopal11/hello_world latest           b534869c81f0       3 weeks ago       1.22MB
busybox             latest             2ae34abc2ed0       3 weeks ago       165MB
httpd               latest             3639174793ba       7 months ago      463MB
tomcat              jre8
ubuntu@ip-172-31-44-85:/opt/helloworld$ docker login
Login with your Docker ID to push and pull images from Docker Hub. If you don't have a Docker ID, head over to https://hub.docker.com to create one.
Username: vishwagopal11
Password:
WARNING! Your password will be stored unencrypted in /home/ubuntu/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
ubuntu@ip-172-31-44-85:/opt/helloworld$ docker tag helloworld:latest vishwagopal11/hello_world:version1
ubuntu@ip-172-31-44-85:/opt/helloworld$ docker push vishwagopal11/hello_world:version1
No command 'docker' found, did you mean:
  Command 'docker' from package 'docker.io' (universe)
docker: command not found
ubuntu@ip-172-31-44-85:/opt/helloworld$ docker push vishwagopal11/hello_world:version1
The push refers to repository [docker.io/vishwagopal11/hello_world]
3bb79932b6ab: Pushed
f24d8b358bb1: Mounted from library/tomcat
8bcc49b9925: Mounted from library/tomcat
f0e1731fd286: Mounted from library/tomcat
2b6c38ff3137: Mounted from library/tomcat
b3bf4d5a39fb: Mounted from library/tomcat
fe60061c6c4e: Mounted from library/tomcat
7d63f8777ebf: Mounted from library/tomcat
1b958b53b256: Mounted from library/tomcat
2c719774c1e1: Mounted from library/tomcat
ec62f19bb3aa: Mounted from library/tomcat
f94641f1e1f: Mounted from library/tomcat
version1: digest: sha256:0431c0046339143427e26ad637db5ae3af8b2bbe01a4f5fbb0ff342a55085b0 size: 2837
ubuntu@ip-172-31-44-85:/opt/helloworld$
```



hub.docker.com/repository/registry-1.docker.io/vishwagopal11/hello_world/tags?page=1

dockerhub Search for great content (e.g., mysql) Explore Repositories Organizations Get Help vishwagopal11

Repositories vishwagopal11 / hello_world Using 0 of 1 private repositories. [Get more](#)

General Tags Builds Timeline Collaborators Webhooks Settings

Action Filter Tags Sort by Latest


version1
Last updated 3 minutes ago by vishwagopal11

docker pull vishwagopal11/hello_world

DIGEST	OS/ARCH	COMPRESSED SIZE
0431c0046339	linux/amd64	195.07 MB

hub.docker.com/layers/vishwagopal11/hello_world/version1/images/sha256-0431c0046339143427e26ad637db5ae3af8b2bbe01a4f5fbb07ff342a55085b0

dockerhub Search for great content (e.g., mysql) Explore Repositories Organizations Get Help vishwagopal11

 vishwagopal11 / hello_world : version1
DIGEST: sha256:0431c0046339143427e26ad637db5ae3af8b2bbe01a4f5fbb07ff342a55085b0

OS/ARCH	SIZE	LAST PUSHED
linux/amd64	195.07 MB	4 minutes ago by vishwagopal11

IMAGE HISTORY

Step	Command	Size
1	ADD file ... in /	43.24 MB
2	CMD ["bash"]	0 B
3	/bin/sh -c apt-get update &&	10.29 MB
4	/bin/sh -c set -ex; if	4.14 MB

Command: ADD file:caf91edab64f988bc24766c58ee66c00311c7c921296b8e5b51d7023422a1485 in /

```
ubuntu@ip-172-31-44-85:/opt/helloworld$ docker build -t helloworld .
Sending build context to Docker daemon 497.3MB
Step 1/3 : FROM tomcat:jre8
--> 3639174793ba
Step 2/3 : MAINTAINER Vishwanaath Gopalakrishnan
--> Using cache
--> 51817502ae54
Step 3/3 : COPY HelloWorld.war /usr/local/tomcat/webapps/
--> Using cache
--> 871be81c311a
Successfully built 871be81c311a
Successfully tagged helloworld:latest
ubuntu@ip-172-31-44-85:/opt/helloworld$ docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
smartfin             latest              1b479b227b9a       5 hours ago        166MB
helloworld           latest              871be81c311a       6 hours ago        473MB
vishwagopal11/hello_world latest              871be81c311a       6 hours ago        473MB
vishwagopal11/hello_world version1            871be81c311a       6 hours ago        473MB
busybox              latest              b534869c81f0       3 weeks ago        1.22MB
httpd                latest              2ae34abc2ed0       3 weeks ago        165MB
tomcat               jre8                3639174793ba       7 months ago       463MB
ubuntu@ip-172-31-44-85:/opt/helloworld$ docker tag helloworld:latest vishwagopal11/hello_world:version2
ubuntu@ip-172-31-44-85:/opt/helloworld$ docker push vishwagopal11/hello_world:version2
```

Execute the docker build again and look for new version so the image getting created in the dockerhub after tagging and pushing

```
ubuntu@ip-172-31-44-85:/opt/helloworld$ docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
smartfin             latest             1b479b227b9a       5 hours ago        166MB
helloworld           latest             871be81c311a       6 hours ago        473MB
vishwagopal11/hello_world latest             871be81c311a       6 hours ago        473MB
vishwagopal11/hello_world version1           871be81c311a       6 hours ago        473MB
busybox             latest             b534869c81f0       3 weeks ago        1.22MB
httpd               latest             2ae34abc2ed0       3 weeks ago        165MB
tomcat              jre8               3639174793ba       7 months ago       463MB

ubuntu@ip-172-31-44-85:/opt/helloworld$ docker tag helloworld:latest vishwagopal11/hello_world:version2
ubuntu@ip-172-31-44-85:/opt/helloworld$ docker push vishwagopal11/hello_world:version2
The push refers to repository [docker.io/vishwagopal11/hello_world]
8bb79932b6ab: Layer already exists
f24d8b358bb1: Layer already exists
c8bcc49b9925: Layer already exists
f0e1731fd286: Layer already exists
2b6c38ff3137: Layer already exists
d38f3d5a39fb: Layer already exists
fe60061c6c4e: Layer already exists
7d63f8777ebf: Layer already exists
1b958b53b256: Layer already exists
2c719774c1e1: Layer already exists
ec62f19bb3aa: Layer already exists
f94641f1fe1f: Layer already exists
version2: digest: sha256:0431c0046339143427e26ad637db5ae3af8b2bbe01a4f5fbb07ff342a55085b0 size: 2837
ubuntu@ip-172-31-44-85:/opt/helloworld$ docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
smartfin             latest             1b479b227b9a       5 hours ago        166MB
helloworld           latest             871be81c311a       6 hours ago        473MB
vishwagopal11/hello_world latest             871be81c311a       6 hours ago        473MB
vishwagopal11/hello_world version1           871be81c311a       6 hours ago        473MB
vishwagopal11/hello_world version2           871be81c311a       6 hours ago        473MB
busybox             latest             b534869c81f0       3 weeks ago        1.22MB
httpd               latest             2ae34abc2ed0       3 weeks ago        165MB
tomcat              jre8               3639174793ba       7 months ago       463MB
ubuntu@ip-172-31-44-85:/opt/helloworld$
```

hub.docker.com/repository/docker/vishwagopal11/hello_world

dockerhub Search for great content (e.g., mysql) Explore Repositories Organizations Get Help vishwagopal11

Repositories vishwagopal11 / hello_world Using 0 of 1 private repositories. [Get more](#)

General Tags Builds Timeline Collaborators Webhooks Settings

vishwagopal11 / hello_world

Test Project Hello world

Last pushed: a minute ago

Docker commands [Public View](#)

To push a new tag to this repository.

```
docker push vishwagopal11/hello_world:tagname
```

Tags

This repository contains 2 tag(s).

version1		4 hours ago
version2		2 minutes ago

Recent builds

[Link a source provider and run a build to see build results here.](#)

Repositories **vishwagopal11 / hello_world** Using 0 of 1 private repositories. [Get more](#)

General **Tags** Builds Timeline Collaborators Webhooks Settings

☐ Action Sort by Latest

Tag	DIGEST	OS/ARCH	COMPRESSED SIZE
version2 Last updated 9 minutes ago by vishwagopal11	0431c0046339	linux/amd64	195.07 MB
version1 Last updated 4 hours ago by vishwagopal11	0431c0046339	linux/amd64	195.07 MB

Step 5: Using ECS FARGATE create clusters, service and tasks pointing to the docker hub repository

Using create cluster option and provide the link to the Dockerhub repository using define container and task edit options

us-east-1.console.aws.amazon.com/ecs/home?region=us-east-1#/firstRun

Getting Started with Amazon Elastic Container Service (Amazon ECS) using Fargate

Step 1: Container and Task
 Step 2: Service
 Step 3: Cluster
 Step 4: Review

Diagram of ECS objects and how they relate

Container definition


Choose an image for your container below to get started quickly or define the container image to use.

Container	Image	Memory	CPU
sample-app	httpd:2.4	0.5GB (512)	0.25 vCPU (256)
nginx	nginx:latest	0.5GB (512)	0.25 vCPU (256)
tomcat-webserver	tomcat	2GB (2048)	1 vCPU (1024)
helloworld-container	vishwagopal11/hello_world:version2	0.5GB (512)	0.25 vCPU (256)





Task definition

A task definition is a blueprint for your application, and describes one or more containers through attributes. Some attributes are

us-east-1.console.aws.amazon.com/ecs/home?region=us-east-1#/firstRun

Task definition 

A task definition is a blueprint for your application, and describes one or more containers through attributes. Some attributes are configured at the task level but the majority of attributes are configured per container.

Task definition name	helloworld-task	
Network mode	awsvpc	
Task execution role	Create new	
Compatibilities	FARGATE	
Task memory	0.5GB (512)	
Task CPU	0.25 vCPU (256)	

*Required

Cancel Next

Update the Service details and helloworld-container-service and select the application load balancer option as per the requirement

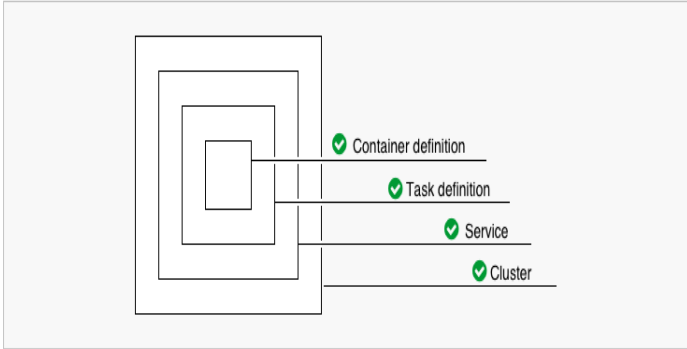
Final step would be to configure the cluster and below is the review page

us-east-1.console.aws.amazon.com/ecs/home?region=us-east-1#/firstRun

Getting Started with Amazon Elastic Container Service (Amazon ECS) using Fargate

Step 1: Container and Task
Step 2: Service
Step 3: Cluster
Step 4: Review

Diagram of ECS objects and how they relate



Review

Review the configuration you've set up before creating your task definition, service, and cluster.

us-east-1.console.aws.amazon.com/ecs/home?region=us-east-1#/firstRun

Task definition

Task definition name: helloworld-task

Network mode: awsvpc

Task execution role: Create new

Container name: helloworld-container

Image: vishwagopal11/hello_world:version2

Memory: 512

Port: 8080

Protocol: HTTP

Service

Service name: helloworld-container-service

Number of desired tasks: 1

Load balancer listener port: 8080

Load balancer listener protocol: HTTP

us-east-1.console.aws.amazon.com/ecs/home?region=us-east-1#/firstRun

Protocol: HTTP

Service

Service name: helloworld-container-service

Number of desired tasks: 1

Load balancer listener port: 8080

Load balancer listener protocol: HTTP

Cluster

Cluster name: helloworld-cluster

VPC ID: Automatically create new

Subnets: Automatically create new

*Required

Cancel Previous Create

Click on create to create the cluster, tasks and services by ECS FARGATE

us-east-1.console.aws.amazon.com/ecs/home?region=us-east-1#/firstRun

Back View service Enabled after service creation completes successfully

Additional features that you can add to your service after creation

Scale based on metrics
You can configure scaling rules based on CloudWatch metrics

Preparing service : 2 of 10 complete

ECS resource creation

Cluster helloworld-cluster	complete
Task definition helloworld-task:1	complete
Service	pending

Additional AWS service integrations

Log group /ecs/helloworld-task	complete
CloudFormation stack	pending
VPC	pending
Subnet 1	pending
Subnet 2	pending
Security group	pending
Load balancer	pending

Ensure that all tasks are complete and click on View service to view the details

us-east-1.console.aws.amazon.com/ecs/home?region=us-east-1#/firstRun

Back View service

Additional features that you can add to your service after creation

Scale based on metrics
You can configure scaling rules based on CloudWatch metrics

Preparing service : 10 of 10 complete

ECS resource creation

Cluster helloworld-cluster	complete ✓
Task definition helloworld-task:1	complete ✓
Service helloworld-container-service	complete ✓

Additional AWS service integrations

Log group /ecs/helloworld-task	complete ✓
CloudFormation stack EC2ContainerService-helloworld-cluster	complete ✓
VPC vpc-0877a3b906f7e03b6	complete ✓
Subnet 1 subnet-07a9ea2cc9a2431b5	complete ✓
Subnet 2 subnet-0847289f6415c9f63	complete ✓
Security group sg-04ea63a29a008cc00	complete ✓
Load balancer arn:aws:elasticloadbalancing:us-east-1:018894105410:loadbalancer/app/EC2Co-EcsEI-QQIV48LVWL3/af128edb3a5a6e6c	complete ✓

ECS Review Page:

us-east-1.console.aws.amazon.com/ecs/home?region=us-east-1#/clusters/helloworld-cluster/services/helloworld-container-service/details

aws Services Resource Groups

vishwagopal11 N. Virginia Support

Amazon ECS

Clusters

Task Definitions

Account Settings

Amazon EKS

Clusters

Amazon ECR

Repositories

AWS Marketplace

Discover software

Subscriptions

Clusters > [helloworld-cluster](#) > Service: [helloworld-container-service](#)

Service : [helloworld-container-service](#) Update Delete

Cluster	helloworld-cluster	Desired count	1
Status	ACTIVE	Pending count	0
Task definition	helloworld-task:1	Running count	1
Service type	REPLICA		
Launch type	FARGATE		
Platform version	LATEST(1.3.0)		
Service role	AWSServiceRoleForECS		

Details Tasks Events Auto Scaling Deployments Metrics Tags Logs

Load Balancing

Target Group Name	Container Name	Container Port
EC2Co-Defau-1Y19GFCPI6801	helloworld-container	8080

Network Access

Amazon ECR
Repositories
AWS Marketplace
Discover software
Subscriptions

Service type: FARGATE
Launch type: FARGATE
Platform version: LATEST(1.3.0)
Service role: AWSServiceRoleForECS

Details Tasks Events Auto Scaling Deployments Metrics Tags Logs

Load Balancing

Target Group Name	Container Name	Container Port
EC2Co-Defau-1Y19GFCPI6801	helloworld-contaner	8080

Network Access

Health check grace period: 0

Allowed VPC: vpc-0877a3b906f7e03b6

Allowed subnets: subnet-07a9ea2cc9a2431b5, subnet-0847289f6415c9f63

Security groups*: sg-04ea63a29a008cc00

Auto-assign public IP: ENABLED

Clusters
Task Definitions
Account Settings
Amazon EKS
Clusters
Amazon ECR
Repositories
AWS Marketplace
Discover software
Subscriptions

Service : helloworld-contaner-service

Update Delete

Cluster: helloworld-cluster
Status: ACTIVE
Task definition: helloworld-task:1
Service type: REPLICA
Launch type: FARGATE
Platform version: LATEST(1.3.0)
Service role: AWSServiceRoleForECS

Desired count: 1
Pending count: 0
Running count: 1

Details Tasks Events Auto Scaling Deployments Metrics Tags Logs

Last updated on December 25, 2019 9:07:48 PM (0m ago)

Task status: **Running** Stopped

Filter in this page

Task	Task Definition	Last status	Desired status	Group	Launch type	Platform version ...
5bda8c33-f0c0-411...	helloworld-task:1	RUNNING	RUNNING	service:helloworld-...	FARGATE	1.3.0

us-east-1.console.aws.amazon.com/ecs/home?region=us-east-1#/clusters/helloworld-cluster/services/helloworld-contaner-service/events

Apps Microsoft Office Co... Dell Promotional e... Sri Varalakshmi Vrat... Sri Varalakshmi Vira... Install the AWS CLI... aws - AWS CLI 1.1...

Clusters
Task Definitions
Account Settings
Amazon EKS
Clusters
Amazon ECR
Repositories
AWS Marketplace
Discover software
Subscriptions

Service : helloworld-contaner-service

Update Delete

Cluster: helloworld-cluster
Status: ACTIVE
Task definition: helloworld-task:1
Service type: REPLICA
Launch type: FARGATE
Platform version: LATEST(1.3.0)
Service role: AWSServiceRoleForECS

Desired count: 1
Pending count: 0
Running count: 1

Details Tasks Events Auto Scaling Deployments Metrics Tags Logs

Last updated on December 25, 2019 9:08:17 PM (0m ago)

Filter in this page

Event Id	Event Time	Message
d13731ee-553d-4fe3-a2b3-2192aa69fa50	2019-12-25 21:07:23 -0500	service helloworld-contaner-service has reached a steady state.
d6c6be262-5a90-4538-a3d7-338c62ca5db5	2019-12-25 21:06:34 -0500	service helloworld-contaner-service registered 1 targets in target-group EC2Co-Defau-1Y19GFCPI6801
ec32b004-7000-40a5-ed62-a5b151a04ada	2019-12-25 21:05:33 -0500	service helloworld-contaner-service has started 1 tasks: task-5bda8c33-f0c0-411...

Subscriptions [↗](#)

Details	Tasks	Events	Auto Scaling	Deployments	Metrics	Tags	Logs
Task status RUNNING STOPPED							
Last updated on December 25, 2019 9:09:03 PM (0m ago) ↻							
Filter logs ✕ All 30s 5m 1h 6h 1d 1w < 1-76 >							
Timestamp (UTC+00:00) ▾	Message	Task					
2019-12-25 21:07:08	26-Dec-2019 02:07:08.472 INFO [main] org.apache.catalina...	5bda8c33-f0c0-411a-92a9-774050832641					
2019-12-25 21:07:08	26-Dec-2019 02:07:08.468 INFO [main] org.apache.coyote.A...	5bda8c33-f0c0-411a-92a9-774050832641					
2019-12-25 21:07:08	26-Dec-2019 02:07:08.171 INFO [main] org.apache.coyote.A...	5bda8c33-f0c0-411a-92a9-774050832641					
2019-12-25 21:07:08	26-Dec-2019 02:07:08.162 INFO [localhost-startStop-1] org...	5bda8c33-f0c0-411a-92a9-774050832641					
2019-12-25 21:07:08	26-Dec-2019 02:07:08.069 INFO [localhost-startStop-1] org...	5bda8c33-f0c0-411a-92a9-774050832641					
2019-12-25 21:07:08	26-Dec-2019 02:07:08.069 INFO [localhost-startStop-1] org...	5bda8c33-f0c0-411a-92a9-774050832641					
2019-12-25 21:07:07	26-Dec-2019 02:07:07.968 INFO [localhost-startStop-1] org...	5bda8c33-f0c0-411a-92a9-774050832641					
2019-12-25 21:07:07	26-Dec-2019 02:07:07.968 INFO [localhost-startStop-1] org...	5bda8c33-f0c0-411a-92a9-774050832641					
2019-12-25 21:07:06	26-Dec-2019 02:07:06.859 INFO [localhost-startStop-1] org...	5bda8c33-f0c0-411a-92a9-774050832641					
2019-12-25 21:07:06	26-Dec-2019 02:07:06.859 INFO [localhost-startStop-1] org...	5bda8c33-f0c0-411a-92a9-774050832641					

Task - Summary:

us-east-1.console.aws.amazon.com/ecs/home?region=us-east-1#/taskDefinitions

Apps Microsoft Office Co... Dell Promotional e... Sri Varalakshmi Vrat... Sri Varalakshmi Vira... Install the AWS CLI... aws — AWS CLI 1.1...

aws Services Resource Groups

Amazon ECS Clusters Task Definitions Account Settings Amazon EKS Clusters Amazon ECR Repositories AWS Marketplace Discover software Subscriptions [↗](#)

Task Definitions

Task definitions specify the container information for your application, such as how many containers are part of your task, what resources they will use, how they are linked together, and which host ports they will use. [Learn more](#)

Create new Task Definition Create new revision Actions

Last updated on December 25, 2019 9:11:49 PM (1m ago) [↻](#) [?](#)

Status: **ACTIVE** INACTIVE 1 selected

Filter in this page < 1-1 > Page size 50

Task Definition	Latest revision status
<input checked="" type="checkbox"/> helloworld-task	ACTIVE

Cluster – Summary

console.aws.amazon.com/ecs/home?region=us-east-1#/clusters

Apps Microsoft Office Co... Dell Promotional e... Sri Varalakshmi Vrat... Sri Varalakshmi Vira... Install the AWS CLI... aws — AWS CLI 1.1...

aws Services Resource Groups

Amazon ECS Clusters Task Definitions Account Settings Amazon EKS Clusters Amazon ECR Repositories AWS Marketplace Discover software Subscriptions [↗](#)

Clusters

An Amazon ECS cluster is a regional grouping of one or more container instances on which you can run task requests. Each account receives a default cluster the first time you use the Amazon ECS service. Clusters may contain more than one Amazon EC2 instance type.

For more information, see the [ECS documentation](#).

Create Cluster Get Started

View list card [view all](#) [↻](#)

1-1 of 1

helloworld-cluster >

FARGATE

1 Services	1 Running tasks	0 Pending tasks
------------	-----------------	-----------------

0 Services	0 Pending tasks	0 Pending tasks	No data CPU utilization	No data Memory utilization	0 Pending tasks
------------	-----------------	-----------------	-------------------------	----------------------------	-----------------

- EC2 – Load balancing and target groups created by FARGATE for the EC2 instance

console.aws.amazon.com/ec2/v2/home?region=us-east-1#Instances:sort=securityGroupNames

Services Resource Groups

New EC2 Experience

Launch Instance Connect Actions

EC2 Dashboard

Events Tags Reports Limits

INSTANCES

Instances Instance Types Launch Templates Spot Requests Savings Plans Reserved Instances Dedicated Hosts Scheduled Instances Capacity Reservations

IMAGES

Filter by tags and attributes or search by keyword

Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS (IP)
cloud-docker	i-01d2aa3095e95e7aa	t2.micro	us-east-1a	running	2/2 checks ...	None	ec2-3-90-244-242

Instance: i-01d2aa3095e95e7aa (cloud-docker) Public DNS: ec2-3-90-244-242.compute-1.amazonaws.com

Description Status Checks Monitoring Tags

Instance ID i-01d2aa3095e95e7aa Public DNS (IPv4) ec2-3-90-244-242.compute-1.amazonaws.com

Instance state running IPv4 Public IP 3.90.244.242

Instance type t2.micro IPv6 IPs -

Finding Opt-in to AWS Compute Optimizer for recommendations. Learn more Elastic IPs

Private DNS ip-172-31-44-85.ec2.internal Availability zone us-east-1a

Private IPs 172.31.44.85 Security groups open-ssh-http-tomcat. view inbound rules. view outbound rules

Secondary private IPs Scheduled events No scheduled events

VPC ID vpc-0f603175 (def-vpc) AMI ID ubuntu/images/hvm-ssd/ubuntu-xenial-16.04-amd64-server-20190913 (ami-04763b3055de4860b)

Subnet ID subnet-12b5b34e (def-subnet-1a) Platform -

Feedback English (US) © 2008 - 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use

console.aws.amazon.com/ec2/v2/home?region=us-east-1#LoadBalancers:sort=loadBalancerName

Services Resource Groups

New EC2 Experience

Create Load Balancer Actions

EC2 Dashboard

Events Tags Reports Limits

INSTANCES

Instances Instance Types Launch Templates Spot Requests Savings Plans Reserved Instances Dedicated Hosts Scheduled Instances Capacity Reservations

IMAGES

Filter by tags and attributes or search by keyword

Name	DNS name	State	VPC ID	Availability Zones	Type	Create
EC2Co-EcsEI-QQIV48LVWL3	EC2Co-EcsEI-QQIV48LVWL...	active	vpc-0877a3b9067e03b6	us-east-1a, us-east-1b	application	Deceml

Load balancer: EC2Co-EcsEI-QQIV48LVWL3

Description Listeners Monitoring Integrated services Tags

Basic Configuration

Name EC2Co-EcsEI-QQIV48LVWL3

ARN arn:aws:elasticloadbalancing:us-east-1:018894105410:loadbalancer/app/EC2Co-EcsEI-QQIV48LVWL3/af128edb3a5a6e6c

DNS name EC2Co-EcsEI-QQIV48LVWL3-74163045.us-east-1.elb.amazonaws.com (A Record)

State active

Type application

Feedback English (US) © 2008 - 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use

← → ↻ console.aws.amazon.com/ec2/v2/home?region=us-east-1#TargetGroups:sort=targetGroupName

Apps Microsoft Office Co... Dell Promotional e... Sri Varalakshmi Vrat... Sri Varalakshmi Vira... Install the AWS CLI... aws — AWS CLI 1.1...

aws Services Resource Groups

New EC2 Experience Tell us what you think

STORE Volumes Snapshots Lifecycle Manager

NETWORK & SECURITY Security Groups Elastic IPs New Placement Groups New Key Pairs New Network Interfaces

LOAD BALANCING Load Balancers Target Groups

AUTO SCALING Launch Configurations Auto Scaling Groups

Create target group Actions

Filter by tags and attributes or search by keyword

Name	Port	Protocol	Target type	Load Balanc	VPC ID	Monitoring
EC2Co-Defau-1Y19GFCPI6...	8080	HTTP	ip	EC2Co-Ec...	vpc-0877a3b906f7e03b6	
web-autoscale-tg	80	HTTP	instance		vpc-0f603175	

Description Targets Health checks Monitoring Tags

Basic Configuration

Name EC2Co-Defau-1Y19GFCPI6801

ARN arn:aws:elasticloadbalancing:us-east-1:018894105410:targetgroup/EC2Co-Defau-1Y19GFCPI6801/7307d0bb808a0673

Protocol HTTP

Port 8080

Target type ip

VPC vpc-0877a3b906f7e03b6

Load balancer EC2Co-EcsEI-QQIV48LVWL3

Feedback English (US) © 2008 - 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use

← → ↻ console.aws.amazon.com/ec2/v2/home?region=us-east-1#TargetGroups:sort=targetGroupName

Apps Microsoft Office Co... Dell Promotional e... Sri Varalakshmi Vrat... Sri Varalakshmi Vira... Install the AWS CLI... aws — AWS CLI 1.1...

aws Services Resource Groups

New EC2 Experience Tell us what you think

STORE Volumes Snapshots Lifecycle Manager

NETWORK & SECURITY Security Groups Elastic IPs New Placement Groups New Key Pairs New Network Interfaces

LOAD BALANCING Load Balancers Target Groups

AUTO SCALING Launch Configurations Auto Scaling Groups

Create target group Actions

Filter by tags and attributes or search by keyword

Name	Port	Protocol	Target type	Load Balanc	VPC ID	Monitoring
EC2Co-Defau-1Y19GFCPI6...	8080	HTTP	ip	EC2Co-Ec...	vpc-0877a3b906f7e03b6	
web-autoscale-tg	80	HTTP	instance		vpc-0f603175	

Description Targets Health checks Monitoring Tags

The load balancer starts routing requests to a newly registered target as soon as the registration process completes and the target passes the initial health checks. If demand on your targets increases, you can register additional targets. If demand on your targets decreases, you can deregister targets.

Edit

Registered targets

IP address	Port	Availability Zone	Status	Description
10.0.0.116	8080	us-east-1a	healthy	This target is currently passing target group's health checks.

Availability Zones

Availability Zone	Target count	Healthy?
us-east-1a	1	Yes

aws Services Resource Groups

New EC2 Experience Tell us what you think

STORE Volumes Snapshots Lifecycle Manager

NETWORK & SECURITY Security Groups Elastic IPs New Placement Groups New Key Pairs New Network Interfaces

LOAD BALANCING Load Balancers Target Groups

AUTO SCALING Launch Configurations Auto Scaling Groups

Create target group Actions

Filter by tags and attributes or search by keyword

Name	Port	Protocol	Target type	Load Balanc	VPC ID	Monitoring
EC2Co-Defau-1Y19GFCPI6...	8080	HTTP	ip	EC2Co-Ec...	vpc-0877a3b906f7e03b6	
web-autoscale-tg	80	HTTP	instance		vpc-0f603175	

Target group: EC2Co-Defau-1Y19GFCPI6801

Description Targets Health checks Monitoring Tags

Add/Edit Tags

Key	Value
Description	Created for ECS cluster helloworld-cluster
Name	ECS helloworld-cluster - TargetGroup

Setup Elasticity rules using Auto scaling group if needed by editing the Task

Set Auto Scaling (optional)

Edit

Minimum number of tasks 1

Maximum number of tasks 2

upper-bound: CPUUtilization >= 80

Policy type: Step scaling

For alarm: [alarm-upper](#)

Take the action:

Add 1 tasks when 80 <= CPUUtilization

lower-bound: CPUUtilization <= 30

Policy type: Step scaling

For alarm: [alarm-lower](#)

Take the action:

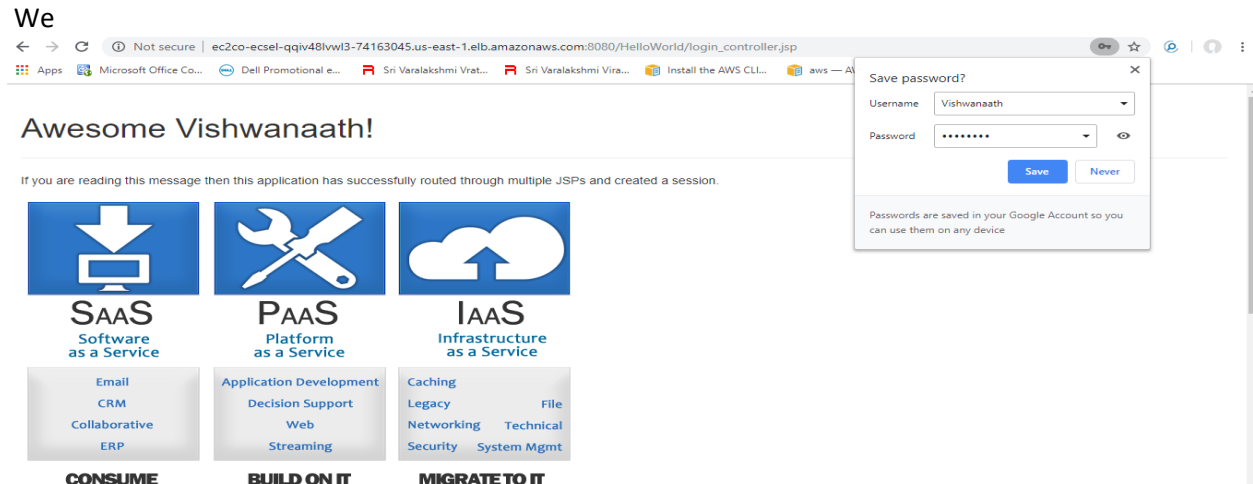
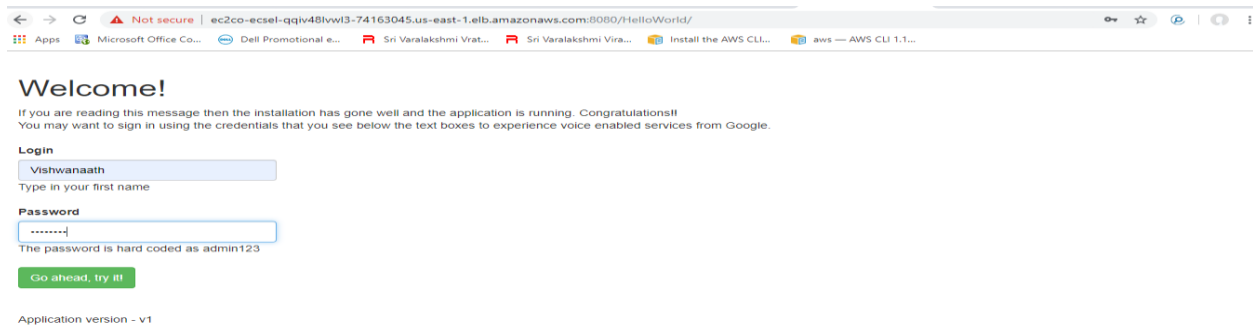
Remove 1 tasks when 30 >= CPUUtilization

Check the DNS name of the Load balancer and try to access the service with port 8080

<http://ec2co-ecsel-qgiv48lvwl3-74163045.us-east-1.elb.amazonaws.com:8080/>

- Access the HelloWorld application from the EC2 instance

<http://ec2co-ecsel-qgiv48lvwl3-74163045.us-east-1.elb.amazonaws.com:8080/HelloWorld/>



Lessons and Observations:

- Created an EC2 instance and installed docker files in it and ensured that the setup is working as expected
- Downloaded Tomcat application image with GRE8 from docker hub into the ubuntu instance and create a container and run it in detached mode mapped to 8080 port
- Injected the application HelloWorld.war and built the docker image and accessed it from the EC2 instance to check if it is working
- Sign up for docker hub and create public repository and tagged the image appropriately and pushed to Docker Hub Repository and ensured that the further upgrades and builds to the application are being pushed to the docker hub repository
- Using ECS FARGATE, configured clusters, created tasks and services and managed the container through ECS
- Also created the ASG, Load balancers and Target groups were setup the ECS and defined the auto scaling policies
- Validated if the Tomcat application is working from the Target groups created