PGPCC - PROJECT

Deploying a data entry application on Containers

Scenario :

This deployment would require you to host the provided PHP application on an ECS cluster with the backend database being hosted on an EC2 instance running a MySQL container on Docker.
The database container will need to be exposed for remote connections and logins.
The PHP application should be configured with the database parameters and credentials and have the required libraries enabled in the container.

What are you expected to do?

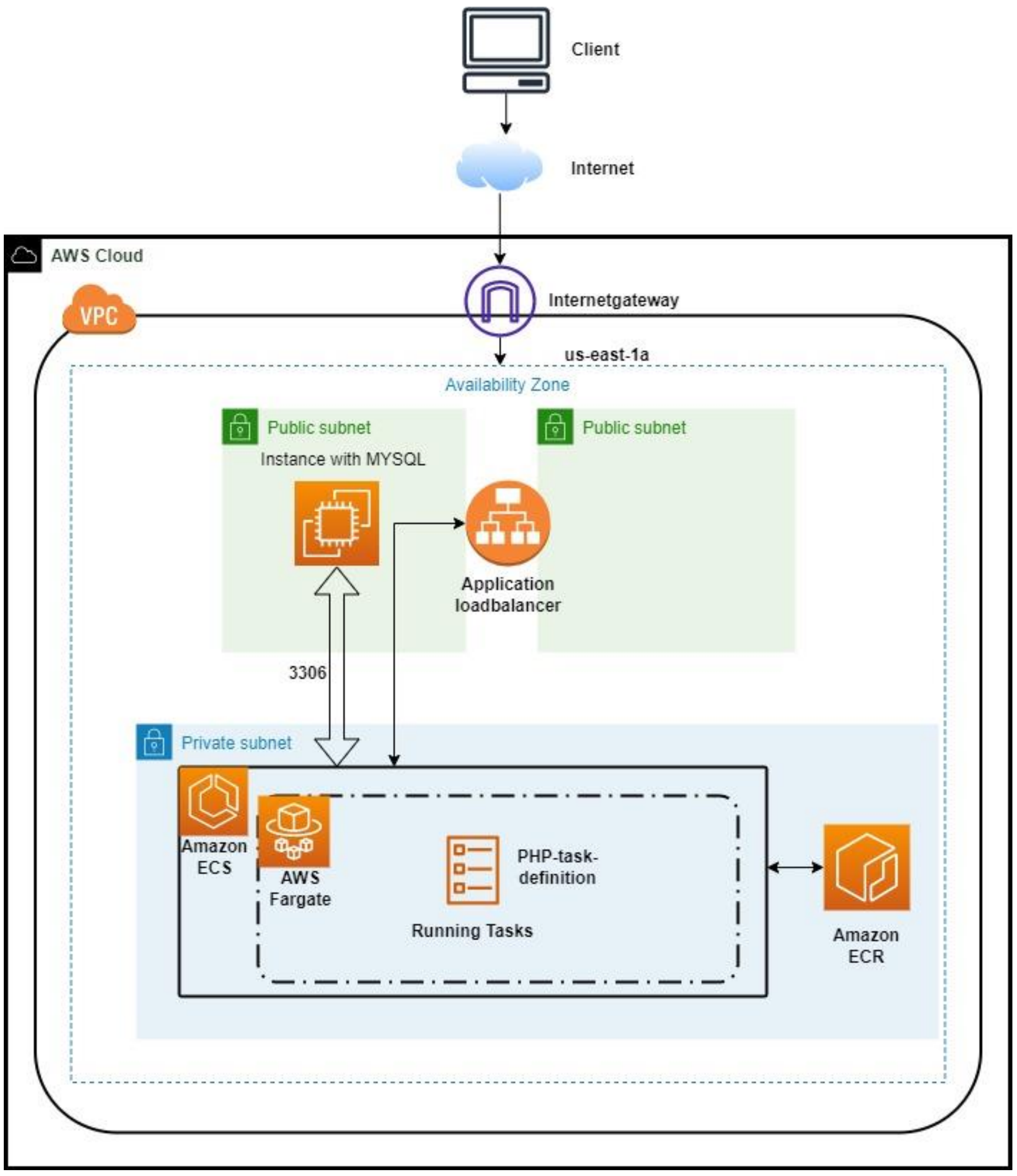
1. Phase 1 – Architecture

Create an architecture diagram for the final implementation

2. Phase 2 – Implementation

A. Download the zip file crud.zip provided with this project document on the Olympus portal
B. Create an EC2 instance using Amazon Linux 2
C. Deploy a MySQL container on the EC2 instance
D. Package the provided PHP application into a Docker Image
E. Push this Docker image into an ECR repository
F. Deploy this image on ECS Fargate with a Load Balancer attached
G. Access the PHP application on the web browser using the port configured in ECS

1. Phase 1 – Architecture:



2. Phase 2 – Implementation:

Creation of Database Server

Step name : Creation of **Database server**

Instructions : 1) Navigate to EC2 using the Services button at the top of the screen

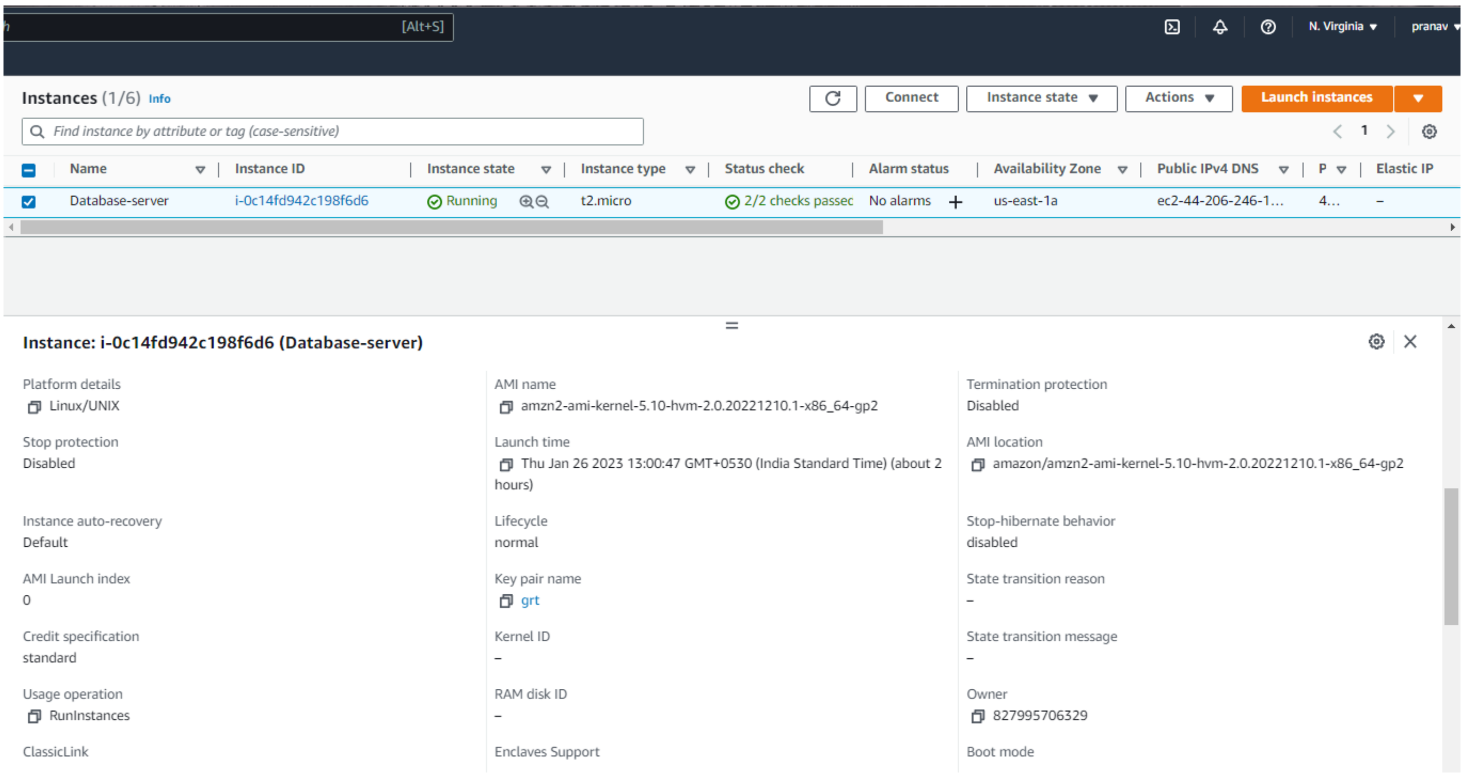
2) Select Instances at the left side of the screen

3) Click on Launch Instance

- Select the Amazon Linux 2
- Select the instance type **t2.micro**
- Select Network as "**default vpc**" and subnet as "**public-subnet**
- For the security group, open the **ports 80,443,22 and 3306** for source set to **"Anywhere"**

4) Launch the instance after creating a new pem file and downloading it.

NOTE:pem file created was grt.pem



Instances (1/6) Info

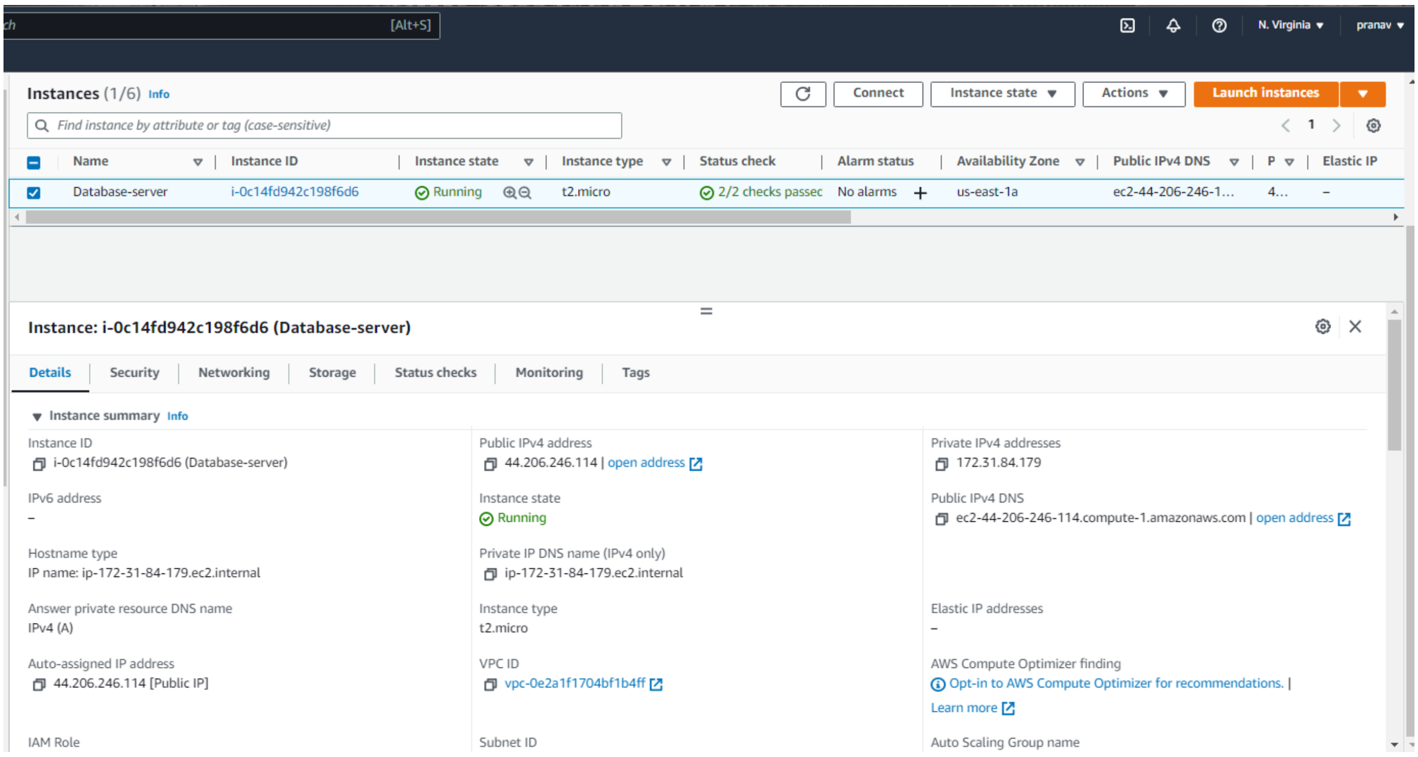
Find instance by attribute or tag (case-sensitive)

	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	P	Elastic IP
✓	Database-server	i-0c14fd942c198f6d6	Running	t2.micro	2/2 checks passec	No alarms	us-east-1a	ec2-44-206-246-1...	4...	–

Instance: i-0c14fd942c198f6d6 (Database-server)

Platform details Linux/UNIX	AMI name amzn2-ami-kernel-5.10-hvm-2.0.20221210.1-x86_64-gp2	Termination protection Disabled
Stop protection Disabled	Launch time Thu Jan 26 2023 13:00:47 GMT+0530 (India Standard Time) (about 2 hours)	AMI location amazon/amzn2-ami-kernel-5.10-hvm-2.0.20221210.1-x86_64-gp2
Instance auto-recovery Default	Lifecycle normal	Stop-hibernate behavior disabled
AMI Launch index 0	Key pair name grt	State transition reason –
Credit specification standard	Kernel ID –	State transition message –
Usage operation RunInstances	RAM disk ID –	Owner 827995706329
ClassicLink	Enclaves Support	Boot mode

<u>AMI used Amazon Linux 2</u>



Instances (1/6) Info

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	P	Elastic IP
Database-server	i-0c14fd942c198f6d6	Running	t2.micro	2/2 checks passec	No alarms	us-east-1a	ec2-44-206-246-1…	4…	–

Instance: i-0c14fd942c198f6d6 (Database-server)

Instance summary Info

Instance ID	Public IPv4 address	Private IPv4 addresses
i-0c14fd942c198f6d6 (Database-server)	44.206.246.114 | open address	172.31.84.179

IPv6 address: –

Instance state: Running

Public IPv4 DNS: ec2-44-206-246-114.compute-1.amazonaws.com | open address

Hostname type: IP name: ip-172-31-84-179.ec2.internal

Private IP DNS name (IPv4 only): ip-172-31-84-179.ec2.internal

Answer private resource DNS name: IPv4 (A)

Instance type: t2.micro

Elastic IP addresses: –

Auto-assigned IP address: 44.206.246.114 [Public IP]

VPC ID: vpc-0e2a1f1704bf1b4ff

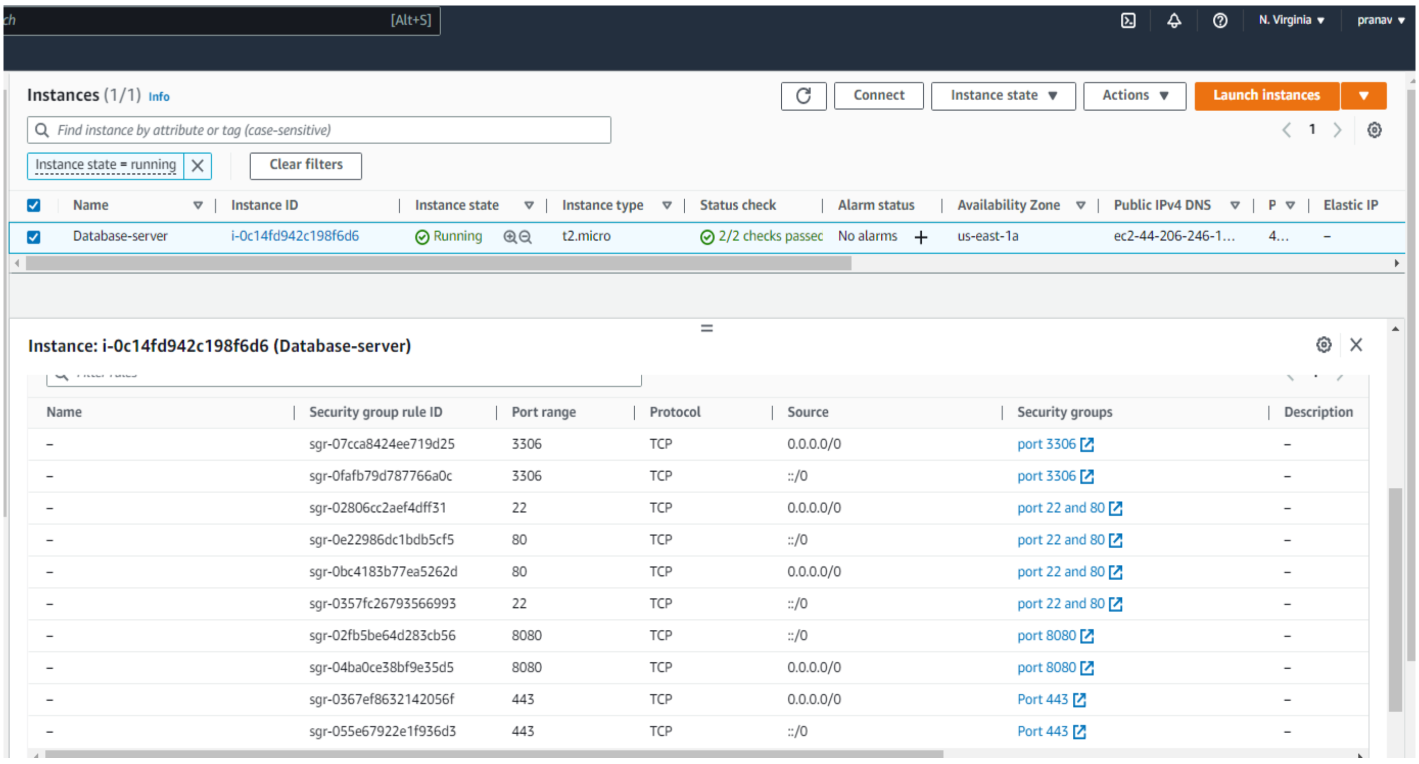
AWS Compute Optimizer finding: Opt-in to AWS Compute Optimizer for recommendations. | Learn more

IAM Role

Subnet ID

Auto Scaling Group name

<u>Instance configuration screen with public IP 44.206.246.114</u>



Instances (1/1) Info

Instance state = running

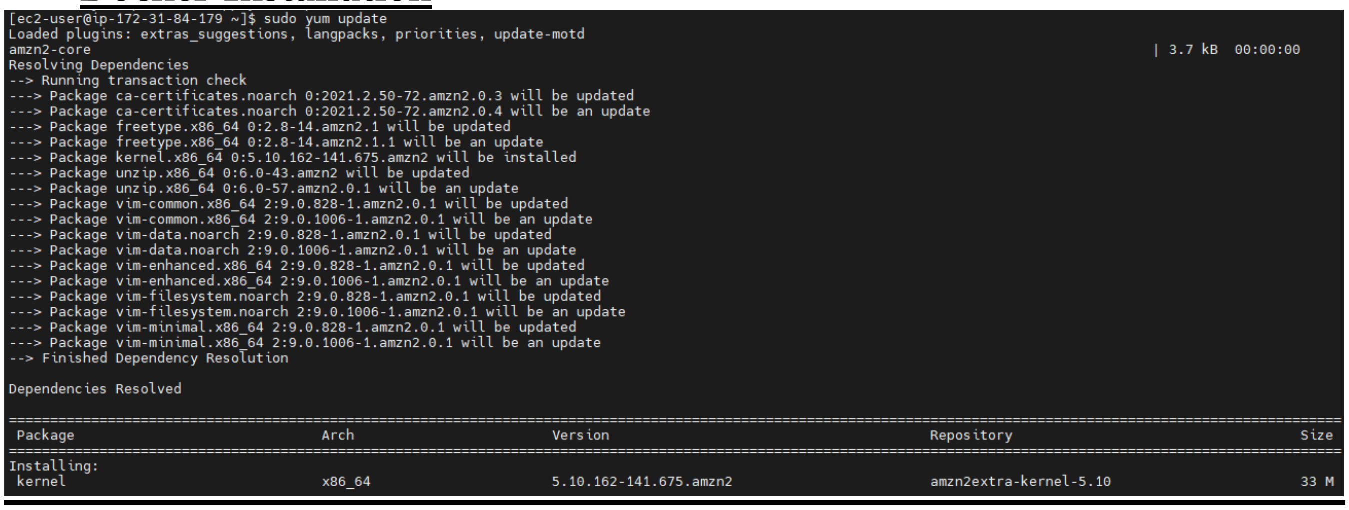
Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	P	Elastic IP
Database-server	i-0c14fd942c198f6d6	Running	t2.micro	2/2 checks passec	No alarms	us-east-1a	ec2-44-206-246-1…	4…	–

Instance: i-0c14fd942c198f6d6 (Database-server)

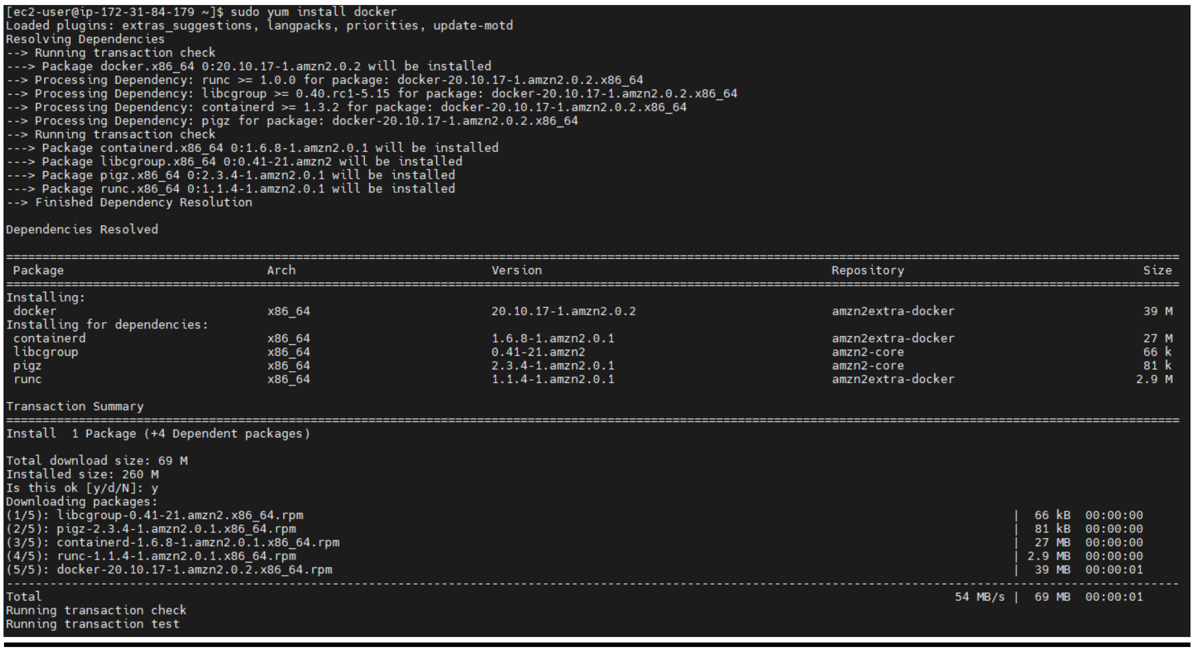
Name	Security group rule ID	Port range	Protocol	Source	Security groups	Description
–	sgr-07cca8424ee719d25	3306	TCP	0.0.0.0/0	port 3306	–
–	sgr-0fafb79d787766a0c	3306	TCP	::/0	port 3306	–
–	sgr-02806cc2aef4dff31	22	TCP	0.0.0.0/0	port 22 and 80	–
–	sgr-0e22986dc1bdb5cf5	80	TCP	::/0	port 22 and 80	–
–	sgr-0bc4183b77ea5262d	80	TCP	0.0.0.0/0	port 22 and 80	–
–	sgr-0357fc26793566993	22	TCP	::/0	port 22 and 80	–
–	sgr-02fb5be64d283cb56	8080	TCP	::/0	port 8080	–
–	sgr-04ba0ce38bf9e35d5	8080	TCP	0.0.0.0/0	port 8080	–
–	sgr-0367ef8632142056f	443	TCP	0.0.0.0/0	Port 443	–
–	sgr-055e67922e1f936d3	443	TCP	::/0	Port 443	–

This will be assigned to Database server EC2 instance in public subnet.
It opens ports **80,443,22,8080 and 3306**.This opens unrestricted access to above ports for the world source set to **"Anywhere"**

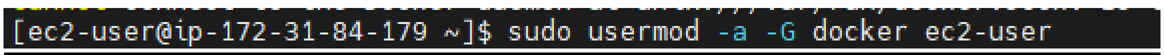
Docker Installation



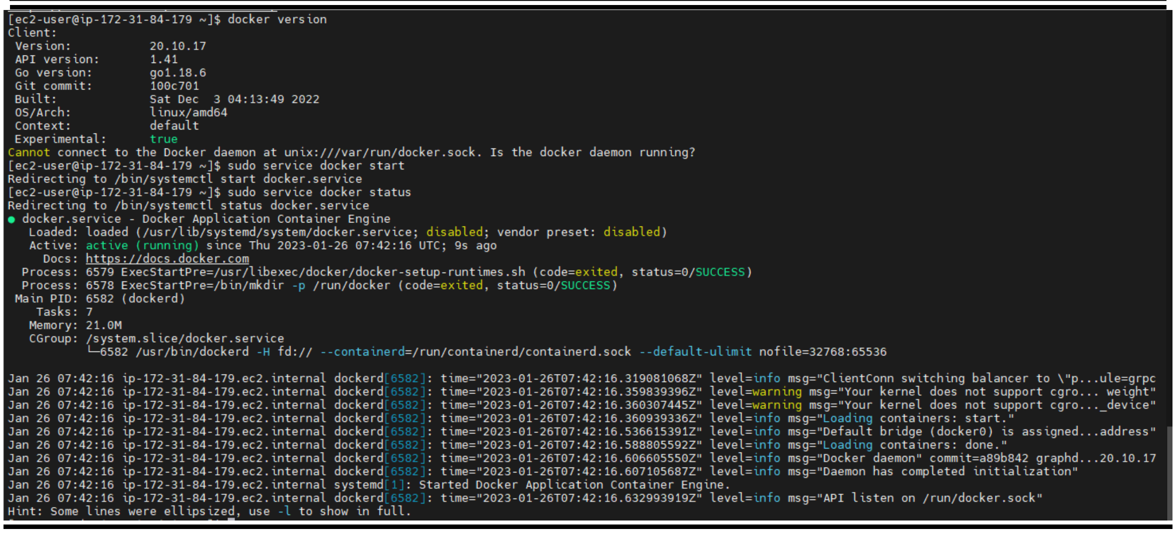
```
[ec2-user@ip-172-31-84-179 ~]$ sudo yum update
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
amzn2-core                                                                                                                     | 3.7 kB  00:00:00
Resolving Dependencies
--> Running transaction check
---> Package ca-certificates.noarch 0:2021.2.50-72.amzn2.0.3 will be updated
---> Package ca-certificates.noarch 0:2021.2.50-72.amzn2.0.4 will be an update
---> Package freetype.x86_64 0:2.8-14.amzn2.1 will be updated
---> Package freetype.x86_64 0:2.8-14.amzn2.1.1 will be an update
---> Package kernel.x86_64 0:5.10.162-141.675.amzn2 will be installed
---> Package unzip.x86_64 0:6.0-43.amzn2 will be updated
---> Package unzip.x86_64 0:6.0-57.amzn2.0.1 will be an update
---> Package vim-common.x86_64 2:9.0.828-1.amzn2.0.1 will be updated
---> Package vim-common.x86_64 2:9.0.1006-1.amzn2.0.1 will be an update
---> Package vim-data.noarch 2:9.0.828-1.amzn2.0.1 will be updated
---> Package vim-data.noarch 2:9.0.1006-1.amzn2.0.1 will be an update
---> Package vim-enhanced.x86_64 2:9.0.828-1.amzn2.0.1 will be updated
---> Package vim-enhanced.x86_64 2:9.0.1006-1.amzn2.0.1 will be an update
---> Package vim-filesystem.noarch 2:9.0.828-1.amzn2.0.1 will be updated
---> Package vim-filesystem.noarch 2:9.0.1006-1.amzn2.0.1 will be an update
---> Package vim-minimal.x86_64 2:9.0.828-1.amzn2.0.1 will be updated
---> Package vim-minimal.x86_64 2:9.0.1006-1.amzn2.0.1 will be an update
--> Finished Dependency Resolution

Dependencies Resolved

======================================================================================================================================================
 Package                             Arch                        Version                                      Repository                         Size
======================================================================================================================================================
Installing:
 kernel                              x86_64                      5.10.162-141.675.amzn2                       amzn2extra-kernel-5.10             33 M
```



```
[ec2-user@ip-172-31-84-179 ~]$ sudo yum install docker
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Resolving Dependencies
--> Running transaction check
---> Package docker.x86_64 0:20.10.17-1.amzn2.0.2 will be installed
--> Processing Dependency: runc >= 1.0.0 for package: docker-20.10.17-1.amzn2.0.2.x86_64
--> Processing Dependency: libcgroup >= 0.40.rc1-5.15 for package: docker-20.10.17-1.amzn2.0.2.x86_64
--> Processing Dependency: containerd >= 1.3.2 for package: docker-20.10.17-1.amzn2.0.2.x86_64
--> Processing Dependency: pigz for package: docker-20.10.17-1.amzn2.0.2.x86_64
--> Running transaction check
---> Package containerd.x86_64 0:1.6.8-1.amzn2.0.1 will be installed
---> Package libcgroup.x86_64 0:0.41-21.amzn2 will be installed
---> Package pigz.x86_64 0:2.3.4-1.amzn2.0.1 will be installed
---> Package runc.x86_64 0:1.1.4-1.amzn2.0.1 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

==============================================================================================================================================================
 Package                             Arch                              Version                                        Repository                        Size
==============================================================================================================================================================
Installing:
 docker                              x86_64                            20.10.17-1.amzn2.0.2                           amzn2extra-docker                 39 M
Installing for dependencies:
 containerd                          x86_64                            1.6.8-1.amzn2.0.1                              amzn2extra-docker                 27 M
 libcgroup                           x86_64                            0.41-21.amzn2                                  amzn2-core                        66 k
 pigz                                x86_64                            2.3.4-1.amzn2.0.1                              amzn2-core                        81 k
 runc                                x86_64                            1.1.4-1.amzn2.0.1                              amzn2extra-docker                2.9 M

Transaction Summary
==============================================================================================================================================================
Install  1 Package (+4 Dependent packages)

Total download size: 69 M
Installed size: 260 M
Is this ok [y/d/N]: y
Downloading packages:
(1/5): libcgroup-0.41-21.amzn2.x86_64.rpm                                                                                         |  66 kB  00:00:00
(2/5): pigz-2.3.4-1.amzn2.0.1.x86_64.rpm                                                                                          |  81 kB  00:00:00
(3/5): containerd-1.6.8-1.amzn2.0.1.x86_64.rpm                                                                                    |  27 MB  00:00:00
(4/5): runc-1.1.4-1.amzn2.0.1.x86_64.rpm                                                                                          | 2.9 MB  00:00:00
(5/5): docker-20.10.17-1.amzn2.0.2.x86_64.rpm                                                                                     |  39 MB  00:00:01
--------------------------------------------------------------------------------------------------------------------------------------------------------------
Total                                                                                                                    54 MB/s |  69 MB  00:00:01
Running transaction check
Running transaction test
```

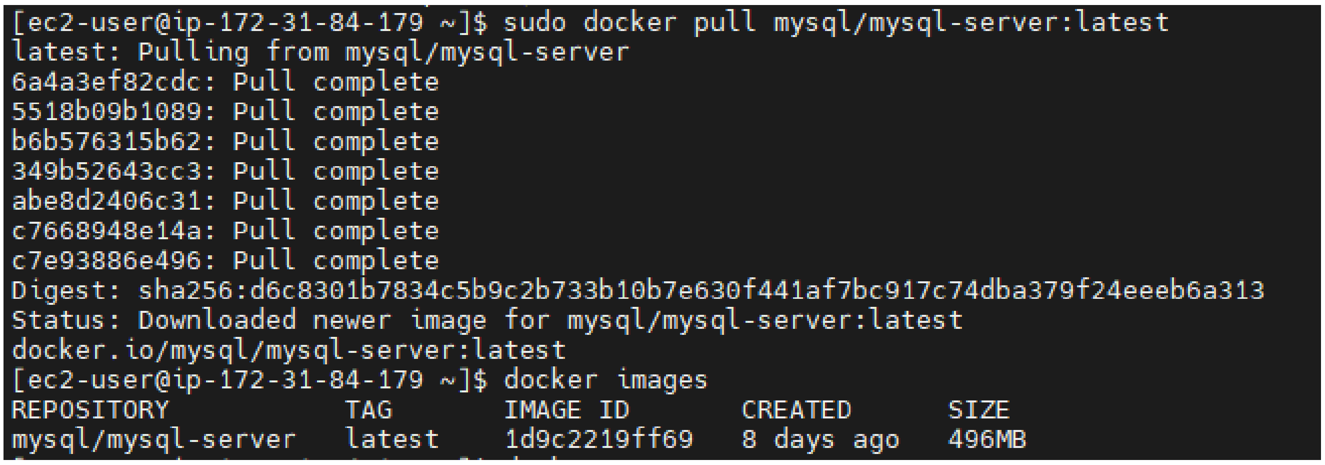


```
[ec2-user@ip-172-31-84-179 ~]$ sudo usermod -a -G docker ec2-user
```

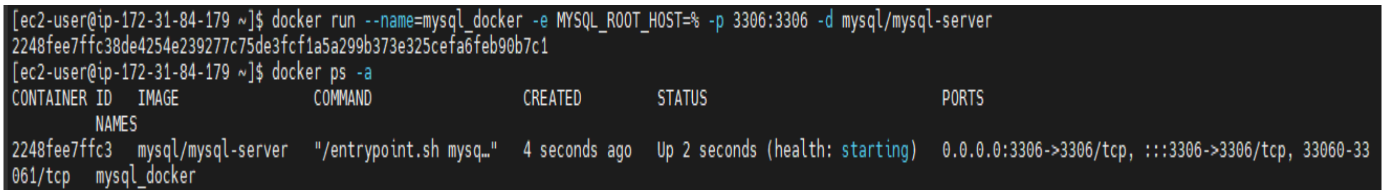


```
[ec2-user@ip-172-31-84-179 ~]$ docker version
Client:
 Version:           20.10.17
 API version:       1.41
 Go version:        go1.18.6
 Git commit:        100c701
 Built:             Sat Dec  3 04:13:49 2022
 OS/Arch:           linux/amd64
 Context:           default
 Experimental:      true
Cannot connect to the Docker daemon at unix:///var/run/docker.sock. Is the docker daemon running?
[ec2-user@ip-172-31-84-179 ~]$ sudo service docker start
Redirecting to /bin/systemctl start docker.service
[ec2-user@ip-172-31-84-179 ~]$ sudo service docker status
Redirecting to /bin/systemctl status docker.service
● docker.service - Docker Application Container Engine
   Loaded: loaded (/usr/lib/systemd/system/docker.service; disabled; vendor preset: disabled)
   Active: active (running) since Thu 2023-01-26 07:42:16 UTC; 9s ago
     Docs: https://docs.docker.com
  Process: 6579 ExecStartPre=/usr/libexec/docker/docker-setup-runtimes.sh (code=exited, status=0/SUCCESS)
  Process: 6578 ExecStartPre=/bin/mkdir -p /run/docker (code=exited, status=0/SUCCESS)
 Main PID: 6582 (dockerd)
    Tasks: 7
   Memory: 21.0M
   CGroup: /system.slice/docker.service
           └─6582 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock --default-ulimit nofile=32768:65536

Jan 26 07:42:16 ip-172-31-84-179.ec2.internal dockerd[6582]: time="2023-01-26T07:42:16.319081068Z" level=info msg="ClientConn switching balancer to \"p...ule=grpc
Jan 26 07:42:16 ip-172-31-84-179.ec2.internal dockerd[6582]: time="2023-01-26T07:42:16.359839396Z" level=warning msg="Your kernel does not support cgro... weight"
Jan 26 07:42:16 ip-172-31-84-179.ec2.internal dockerd[6582]: time="2023-01-26T07:42:16.360307445Z" level=warning msg="Your kernel does not support cgro..._device"
Jan 26 07:42:16 ip-172-31-84-179.ec2.internal dockerd[6582]: time="2023-01-26T07:42:16.360939336Z" level=info msg="Loading containers: start."
Jan 26 07:42:16 ip-172-31-84-179.ec2.internal dockerd[6582]: time="2023-01-26T07:42:16.536615391Z" level=info msg="Default bridge (docker0) is assigned...address"
Jan 26 07:42:16 ip-172-31-84-179.ec2.internal dockerd[6582]: time="2023-01-26T07:42:16.588805592Z" level=info msg="Loading containers: done."
Jan 26 07:42:16 ip-172-31-84-179.ec2.internal dockerd[6582]: time="2023-01-26T07:42:16.606605550Z" level=info msg="Docker daemon" commit=a89b842 graphd...20.10.17
Jan 26 07:42:16 ip-172-31-84-179.ec2.internal dockerd[6582]: time="2023-01-26T07:42:16.607105687Z" level=info msg="Daemon has completed initialization"
Jan 26 07:42:16 ip-172-31-84-179.ec2.internal systemd[1]: Started Docker Application Container Engine.
Jan 26 07:42:16 ip-172-31-84-179.ec2.internal dockerd[6582]: time="2023-01-26T07:42:16.632993919Z" level=info msg="API listen on /run/docker.sock"
Hint: Some lines were ellipsized, use -l to show in full.
```

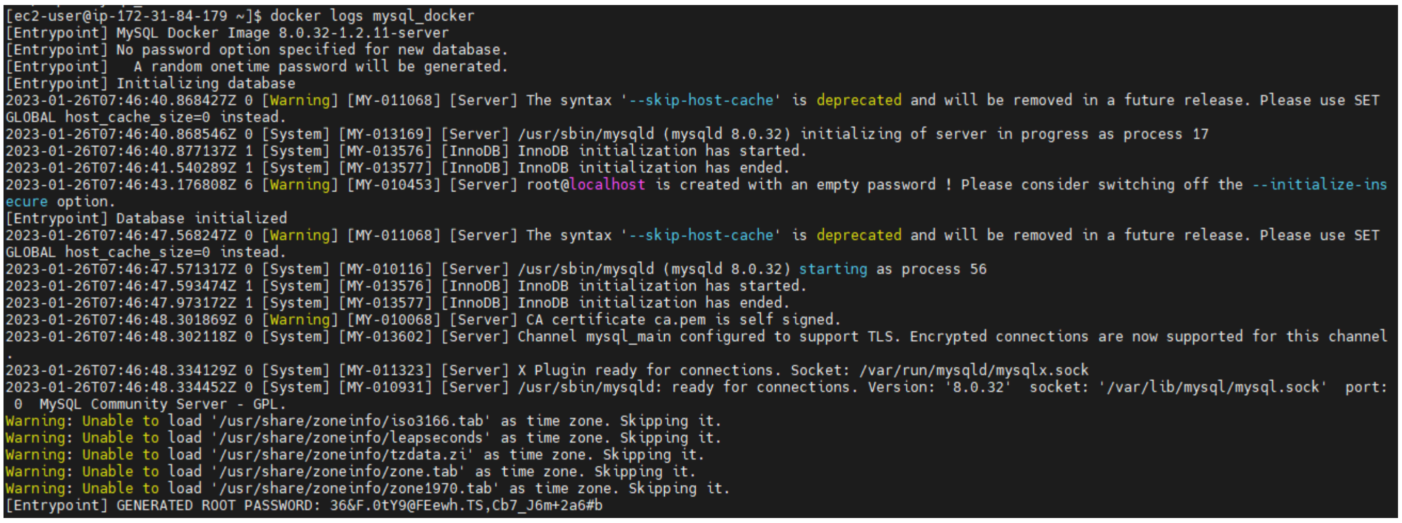
MYSQL SETUP AS A CONTAINER



```
[ec2-user@ip-172-31-84-179 ~]$ sudo docker pull mysql/mysql-server:latest
latest: Pulling from mysql/mysql-server
6a4a3ef82cdc: Pull complete
5518b09b1089: Pull complete
b6b576315b62: Pull complete
349b52643cc3: Pull complete
abe8d2406c31: Pull complete
c7668948e14a: Pull complete
c7e93886e496: Pull complete
Digest: sha256:d6c8301b7834c5b9c2b733b10b7e630f441af7bc917c74dba379f24eeeb6a313
Status: Downloaded newer image for mysql/mysql-server:latest
docker.io/mysql/mysql-server:latest
[ec2-user@ip-172-31-84-179 ~]$ docker images
REPOSITORY           TAG       IMAGE ID       CREATED       SIZE
mysql/mysql-server   latest    1d9c2219ff69   8 days ago    496MB
```

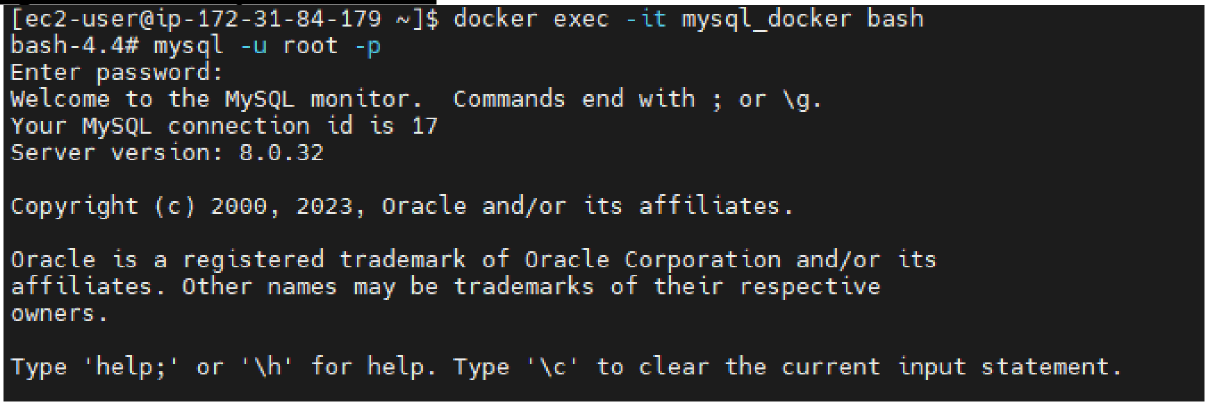


```
[ec2-user@ip-172-31-84-179 ~]$ docker run --name=mysql_docker -e MYSQL_ROOT_HOST=% -p 3306:3306 -d mysql/mysql-server
2248fee7ffc38de4254e239277c75de3fcf1a5a299b373e325cefa6feb90b7c1
[ec2-user@ip-172-31-84-179 ~]$ docker ps -a
CONTAINER ID   IMAGE                COMMAND                  CREATED         STATUS                            PORTS
          NAMES
2248fee7ffc3   mysql/mysql-server   "/entrypoint.sh mysq…"   4 seconds ago   Up 2 seconds (health: starting)   0.0.0.0:3306->3306/tcp, :::3306->3306/tcp, 33060-33
061/tcp   mysql_docker
```

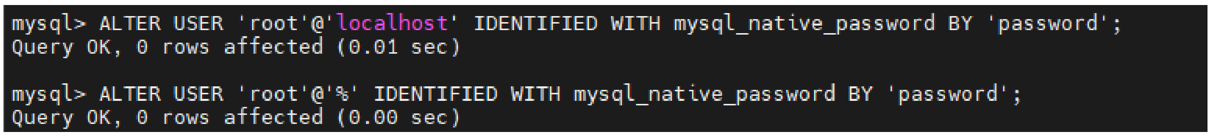


```
[ec2-user@ip-172-31-84-179 ~]$ docker logs mysql_docker
[Entrypoint] MySQL Docker Image 8.0.32-1.2.11-server
[Entrypoint] No password option specified for new database.
[Entrypoint]   A random onetime password will be generated.
[Entrypoint] Initializing database
2023-01-26T07:46:40.868427Z 0 [Warning] [MY-011068] [Server] The syntax '--skip-host-cache' is deprecated and will be removed in a future release. Please use SET
GLOBAL host_cache_size=0 instead.
2023-01-26T07:46:40.868546Z 0 [System] [MY-013169] [Server] /usr/sbin/mysqld (mysqld 8.0.32) initializing of server in progress as process 17
2023-01-26T07:46:40.877137Z 1 [System] [MY-013576] [InnoDB] InnoDB initialization has started.
2023-01-26T07:46:41.540289Z 1 [System] [MY-013577] [InnoDB] InnoDB initialization has ended.
2023-01-26T07:46:43.176808Z 6 [Warning] [MY-010453] [Server] root@localhost is created with an empty password ! Please consider switching off the --initialize-ins
ecure option.
[Entrypoint] Database initialized
2023-01-26T07:46:47.568247Z 0 [Warning] [MY-011068] [Server] The syntax '--skip-host-cache' is deprecated and will be removed in a future release. Please use SET
GLOBAL host_cache_size=0 instead.
2023-01-26T07:46:47.571317Z 0 [System] [MY-010116] [Server] /usr/sbin/mysqld (mysqld 8.0.32) starting as process 56
2023-01-26T07:46:47.593474Z 1 [System] [MY-013576] [InnoDB] InnoDB initialization has started.
2023-01-26T07:46:47.973172Z 1 [System] [MY-013577] [InnoDB] InnoDB initialization has ended.
2023-01-26T07:46:48.301869Z 0 [Warning] [MY-010068] [Server] CA certificate ca.pem is self signed.
2023-01-26T07:46:48.302118Z 0 [System] [MY-013602] [Server] Channel mysql_main configured to support TLS. Encrypted connections are now supported for this channel
.
2023-01-26T07:46:48.334129Z 0 [System] [MY-011323] [Server] X Plugin ready for connections. Socket: /var/run/mysqld/mysqlx.sock
2023-01-26T07:46:48.334452Z 0 [System] [MY-010931] [Server] /usr/sbin/mysqld: ready for connections. Version: '8.0.32'  socket: '/var/lib/mysql/mysql.sock'  port:
 0  MySQL Community Server - GPL.
Warning: Unable to load '/usr/share/zoneinfo/iso3166.tab' as time zone. Skipping it.
Warning: Unable to load '/usr/share/zoneinfo/leapseconds' as time zone. Skipping it.
Warning: Unable to load '/usr/share/zoneinfo/tzdata.zi' as time zone. Skipping it.
Warning: Unable to load '/usr/share/zoneinfo/zone.tab' as time zone. Skipping it.
Warning: Unable to load '/usr/share/zoneinfo/zone1970.tab' as time zone. Skipping it.
[Entrypoint] GENERATED ROOT PASSWORD: 36&F.0tY9@FEewh.TS,Cb7_J6m+2a6#b
```

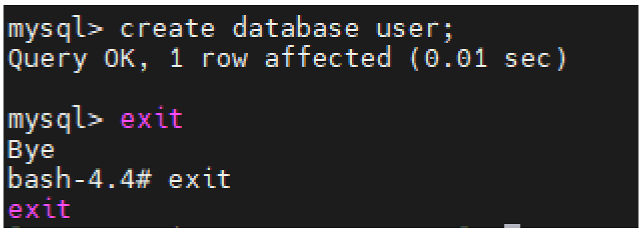
Login to the MySQL shell



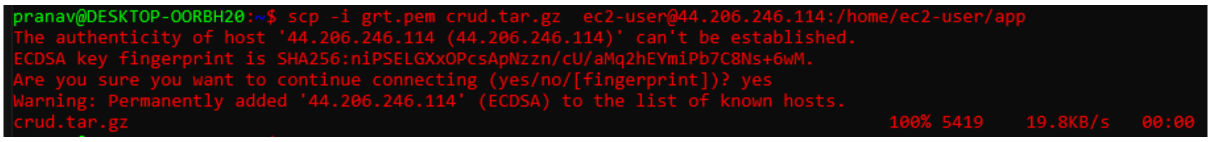
```
[ec2-user@ip-172-31-84-179 ~]$ docker exec -it mysql_docker bash
bash-4.4# mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 17
Server version: 8.0.32

Copyright (c) 2000, 2023, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
```



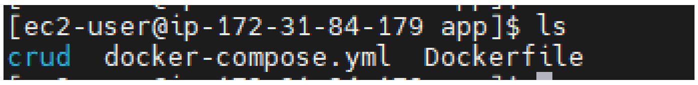
```
mysql> ALTER USER 'root'@'localhost' IDENTIFIED WITH mysql_native_password BY 'password';
Query OK, 0 rows affected (0.01 sec)

mysql> ALTER USER 'root'@'%' IDENTIFIED WITH mysql_native_password BY 'password';
Query OK, 0 rows affected (0.00 sec)
```



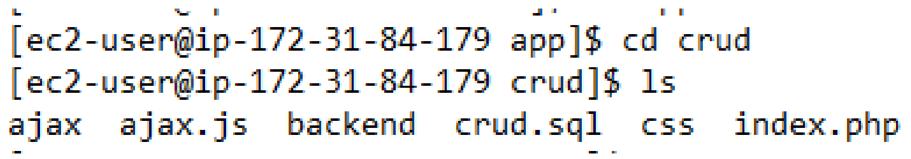
```
mysql> create database user;
Query OK, 1 row affected (0.01 sec)

mysql> exit
Bye
bash-4.4# exit
exit
```



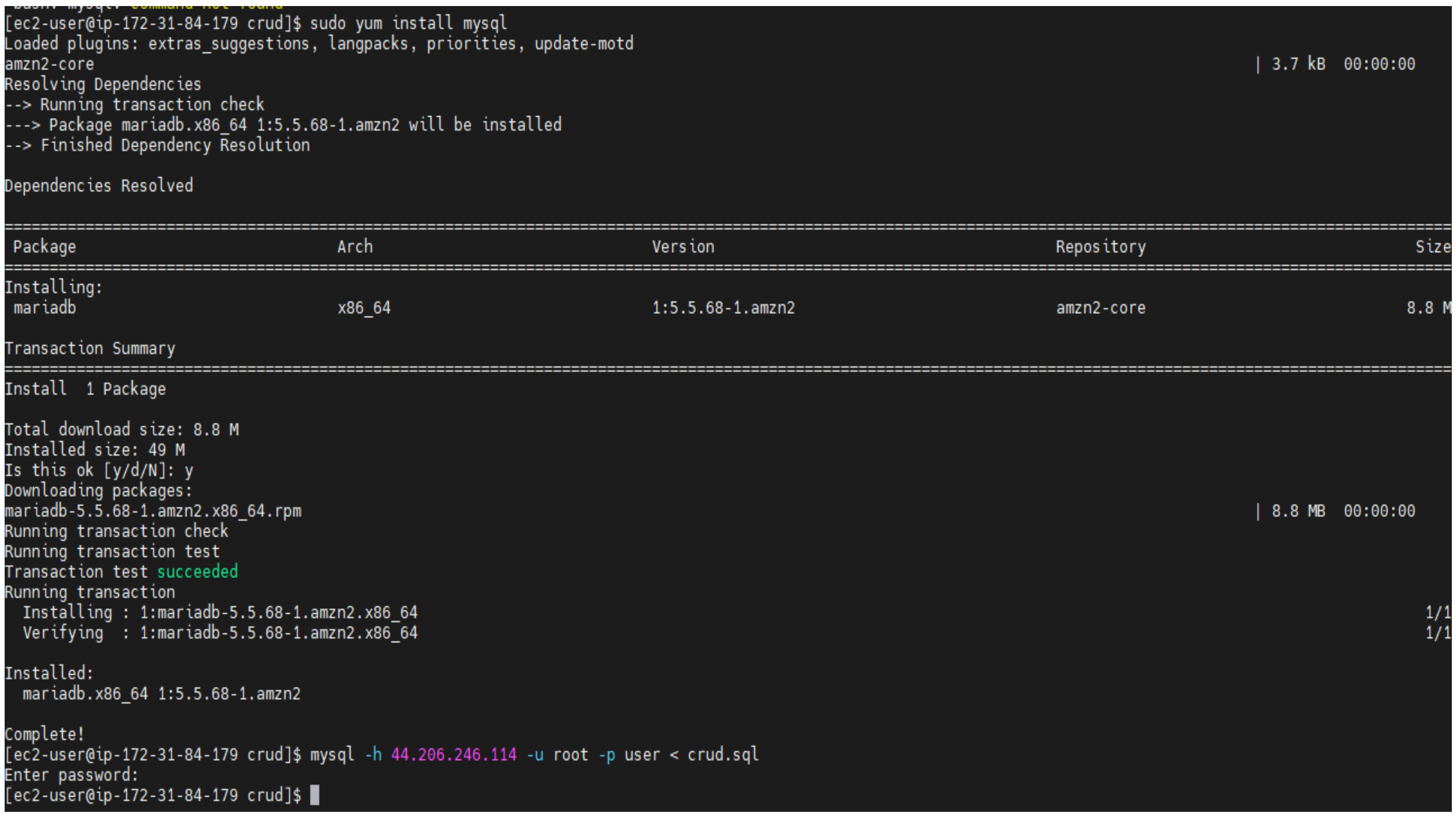
```
pranav@DESKTOP-OORBH20:~$ scp -i grt.pem crud.tar.gz  ec2-user@44.206.246.114:/home/ec2-user/app
The authenticity of host '44.206.246.114 (44.206.246.114)' can't be established.
ECDSA key fingerprint is SHA256:niPSELGXxOPcsApNzzn/cU/aMq2hEYmiPb7C8Ns+6wM.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '44.206.246.114' (ECDSA) to the list of known hosts.
crud.tar.gz                                                                 100% 5419    19.8KB/s   00:00
```



```
[ec2-user@ip-172-31-84-179 app]$ ls
crud  docker-compose.yml  Dockerfile
```

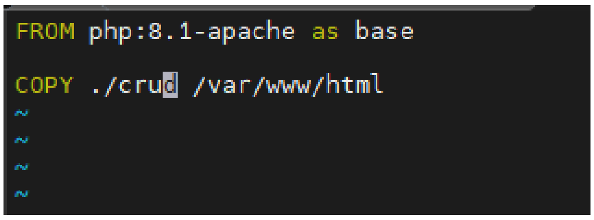


```
[ec2-user@ip-172-31-84-179 app]$ cd crud
[ec2-user@ip-172-31-84-179 crud]$ ls
ajax  ajax.js  backend  crud.sql  css  index.php
```



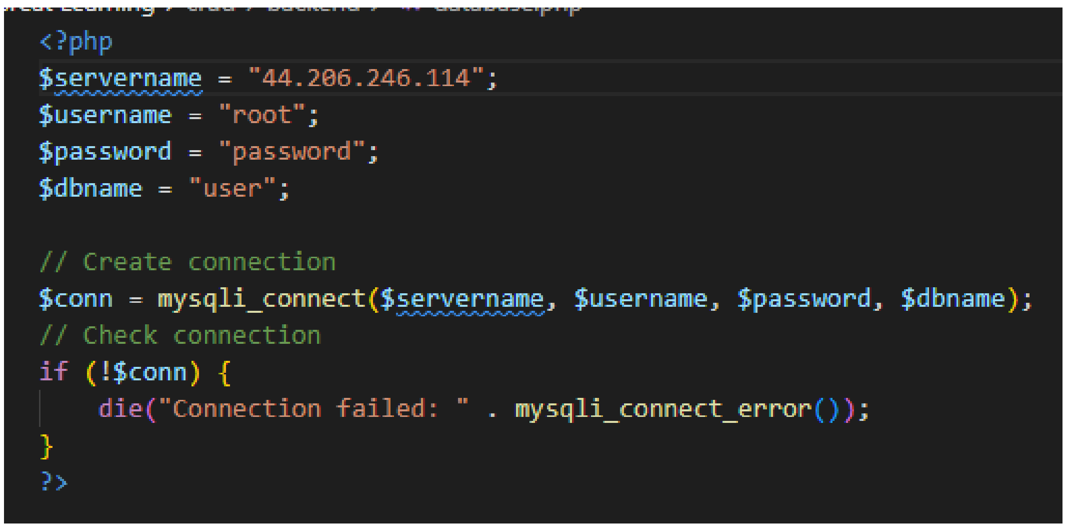
```
[ec2-user@ip-172-31-84-179 crud]$ sudo yum install mysql
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
amzn2-core                                                                                                                            | 3.7 kB  00:00:00
Resolving Dependencies
--> Running transaction check
---> Package mariadb.x86_64 1:5.5.68-1.amzn2 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

==============================================================================================================================================================
 Package                           Arch                              Version                                      Repository                              Size
==============================================================================================================================================================
Installing:
 mariadb                           x86_64                            1:5.5.68-1.amzn2                             amzn2-core                              8.8 M

Transaction Summary
==============================================================================================================================================================
Install  1 Package

Total download size: 8.8 M
Installed size: 49 M
Is this ok [y/d/N]: y
Downloading packages:
mariadb-5.5.68-1.amzn2.x86_64.rpm                                                                                                     | 8.8 MB  00:00:00
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Installing : 1:mariadb-5.5.68-1.amzn2.x86_64                                                                                                             1/1
  Verifying  : 1:mariadb-5.5.68-1.amzn2.x86_64                                                                                                             1/1

Installed:
  mariadb.x86_64 1:5.5.68-1.amzn2

Complete!
[ec2-user@ip-172-31-84-179 crud]$ mysql -h 44.206.246.114 -u root -p user < crud.sql
Enter password:
[ec2-user@ip-172-31-84-179 crud]$
```

Setting up the application as container



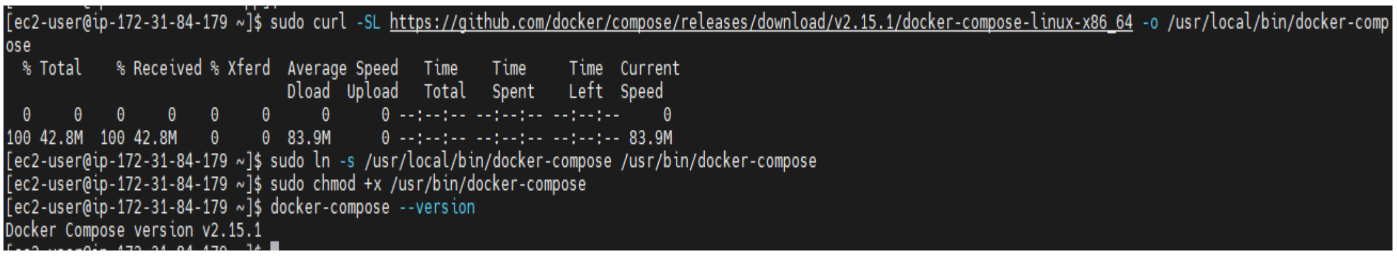
```
FROM php:8.1-apache as base

COPY ./crud /var/www/html
~
~
~
~
```

Docker file



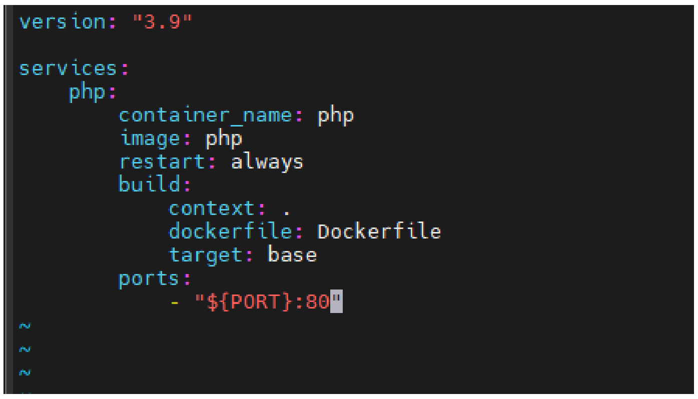
```
<?php
$servername = "44.206.246.114";
$username = "root";
$password = "password";
$dbname = "user";

// Create connection
$conn = mysqli_connect($servername, $username, $password, $dbname);
// Check connection
if (!$conn) {
    die("Connection failed: " . mysqli_connect_error());
}
?>
```

database.php



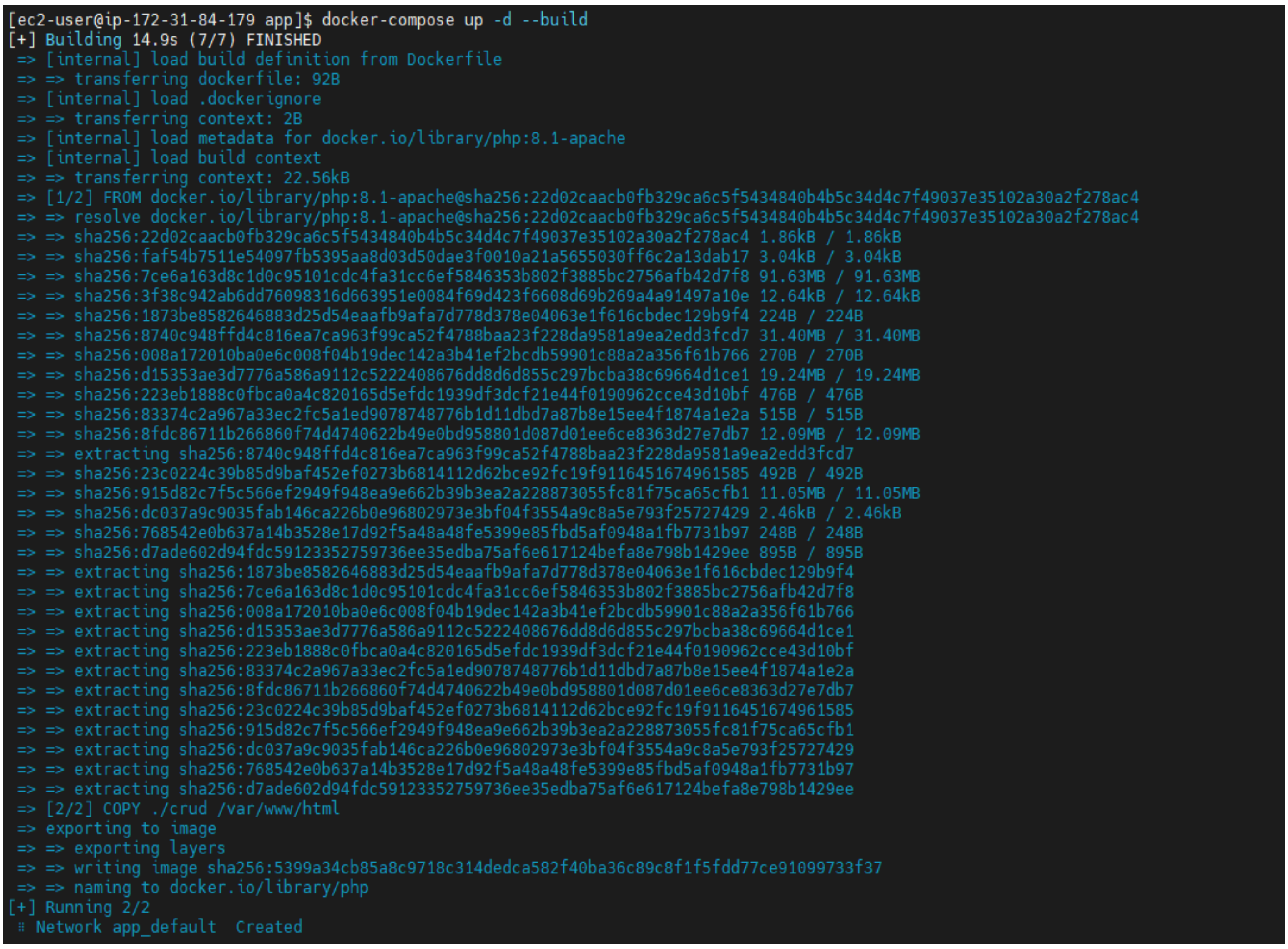
```
[ec2-user@ip-172-31-84-179 ~]$ sudo curl -SL https://github.com/docker/compose/releases/download/v2.15.1/docker-compose-linux-x86_64 -o /usr/local/bin/docker-comp
ose
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0
100 42.8M  100 42.8M    0     0  83.9M      0 --:--:-- --:--:-- --:--:-- 83.9M
[ec2-user@ip-172-31-84-179 ~]$ sudo ln -s /usr/local/bin/docker-compose /usr/bin/docker-compose
[ec2-user@ip-172-31-84-179 ~]$ sudo chmod +x /usr/bin/docker-compose
[ec2-user@ip-172-31-84-179 ~]$ docker-compose --version
Docker Compose version v2.15.1
```

Installing docker compose



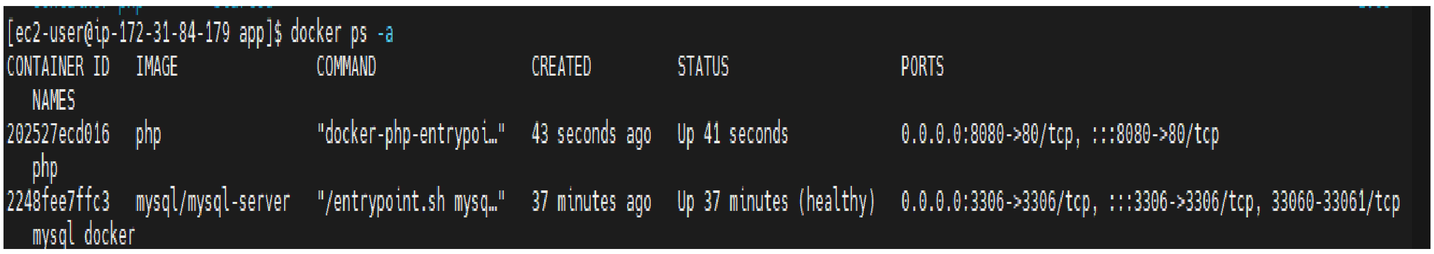
```
version: "3.9"

services:
    php:
        container_name: php
        image: php
        restart: always
        build:
            context: .
            dockerfile: Dockerfile
            target: base
        ports:
            - "${PORT}:80"
~
~
~
```

docker-compose.yml



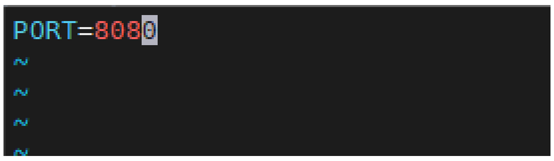
```
[ec2-user@ip-172-31-84-179 app]$ docker-compose up -d --build
[+] Building 14.9s (7/7) FINISHED
 => [internal] load build definition from Dockerfile
 => => transferring dockerfile: 92B
 => [internal] load .dockerignore
 => => transferring context: 2B
 => [internal] load metadata for docker.io/library/php:8.1-apache
 => [internal] load build context
 => => transferring context: 22.56kB
 => [1/2] FROM docker.io/library/php:8.1-apache@sha256:22d02caacb0fb329ca6c5f5434840b4b5c34d4c7f49037e35102a30a2f278ac4
 => => resolve docker.io/library/php:8.1-apache@sha256:22d02caacb0fb329ca6c5f5434840b4b5c34d4c7f49037e35102a30a2f278ac4
 => => sha256:22d02caacb0fb329ca6c5f5434840b4b5c34d4c7f49037e35102a30a2f278ac4 1.86kB / 1.86kB
 => => sha256:faf54b7511e54097fb5395aa8d03d50dae3f0010a21a5655030ff6c2a13dab17 3.04kB / 3.04kB
 => => sha256:7ce6a163d8c1d0c95101cdc4fa31cc6ef5846353b802f3885bc2756afb42d7f8 91.63MB / 91.63MB
 => => sha256:3f38c942ab6dd76098316d663951e0084f69d423f6608d69b269a4a91497a10e 12.64kB / 12.64kB
 => => sha256:1873be8582646883d25d54eaafb9afa7d778d378e04063e1f616cbdec129b9f4 224B / 224B
 => => sha256:8740c948ffd4c816ea7ca963f99ca52f4788baa23f228da9581a9ea2edd3fcd7 31.40MB / 31.40MB
 => => sha256:008a172010ba0e6c008f04b19dec142a3b41ef2bcdb59901c88a2a356f61b766 270B / 270B
 => => sha256:d15353ae3d7776a586a9112c5222408676dd8d6d855c297bcba38c69664d1ce1 19.24MB / 19.24MB
 => => sha256:223eb1888c0fbca0a4c820165d5efdc1939df3dcf21e44f0190962cce43d10bf 476B / 476B
 => => sha256:83374c2a967a33ec2fc5a1ed9078748776b1d11dbd7a87b8e15ee4f1874a1e2a 515B / 515B
 => => sha256:8fdc86711b266860f74d4740622b49e0bd958801d087d01ee6ce8363d27e7db7 12.09MB / 12.09MB
 => => extracting sha256:8740c948ffd4c816ea7ca963f99ca52f4788baa23f228da9581a9ea2edd3fcd7
 => => sha256:23c0224c39b85d9baf452ef0273b6814112d62bce92fc19f9116451674961585 492B / 492B
 => => sha256:915d82c7f5c566ef2949f948ea9e662b39b3ea2a228873055fc81f75ca65cfb1 11.05MB / 11.05MB
 => => sha256:dc037a9c9035fab146ca226b0e96802973e3bf04f3554a9c8a5e793f25727429 2.46kB / 2.46kB
 => => sha256:768542e0b637a14b3528e17d92f5a48a48fe5399e85fbd5af0948a1fb7731b97 248B / 248B
 => => sha256:d7ade602d94fdc59123352759736ee35edba75af6e617124befa8e798b1429ee 895B / 895B
 => => extracting sha256:1873be8582646883d25d54eaafb9afa7d778d378e04063e1f616cbdec129b9f4
 => => extracting sha256:7ce6a163d8c1d0c95101cdc4fa31cc6ef5846353b802f3885bc2756afb42d7f8
 => => extracting sha256:008a172010ba0e6c008f04b19dec142a3b41ef2bcdb59901c88a2a356f61b766
 => => extracting sha256:d15353ae3d7776a586a9112c5222408676dd8d6d855c297bcba38c69664d1ce1
 => => extracting sha256:223eb1888c0fbca0a4c820165d5efdc1939df3dcf21e44f0190962cce43d10bf
 => => extracting sha256:83374c2a967a33ec2fc5a1ed9078748776b1d11dbd7a87b8e15ee4f1874a1e2a
 => => extracting sha256:8fdc86711b266860f74d4740622b49e0bd958801d087d01ee6ce8363d27e7db7
 => => extracting sha256:23c0224c39b85d9baf452ef0273b6814112d62bce92fc19f9116451674961585
 => => extracting sha256:915d82c7f5c566ef2949f948ea9e662b39b3ea2a228873055fc81f75ca65cfb1
 => => extracting sha256:dc037a9c9035fab146ca226b0e96802973e3bf04f3554a9c8a5e793f25727429
 => => extracting sha256:768542e0b637a14b3528e17d92f5a48a48fe5399e85fbd5af0948a1fb7731b97
 => => extracting sha256:d7ade602d94fdc59123352759736ee35edba75af6e617124befa8e798b1429ee
 => [2/2] COPY ./crud /var/www/html
 => exporting to image
 => => exporting layers
 => => writing image sha256:5399a34cb85a8c9718c314dedca582f40ba36c89c8f1f5fdd77ce91099733f37
 => => naming to docker.io/library/php
[+] Running 2/2
 ⠿ Network app_default  Created
```

Creating an image using application files and docker files



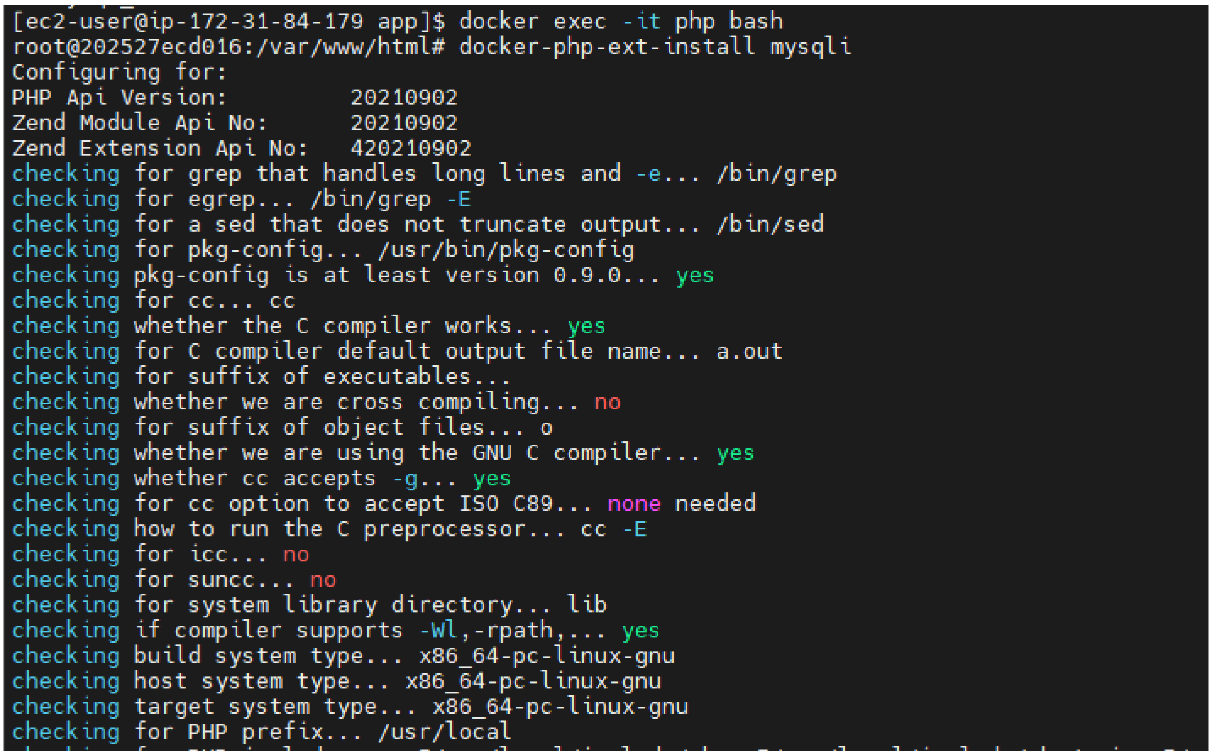
```
[ec2-user@ip-172-31-84-179 app]$ docker ps -a
CONTAINER ID   IMAGE               COMMAND                  CREATED          STATUS                    PORTS
  NAMES
202527ecd016   php                 "docker-php-entrypoi…"   43 seconds ago   Up 41 seconds             0.0.0.0:8080->80/tcp, :::8080->80/tcp
  php
2248fee7ffc3   mysql/mysql-server  "/entrypoint.sh mysq…"   37 minutes ago   Up 37 minutes (healthy)   0.0.0.0:3306->3306/tcp, :::3306->3306/tcp, 33060-33061/tcp
  mysql docker
```

displaying containers

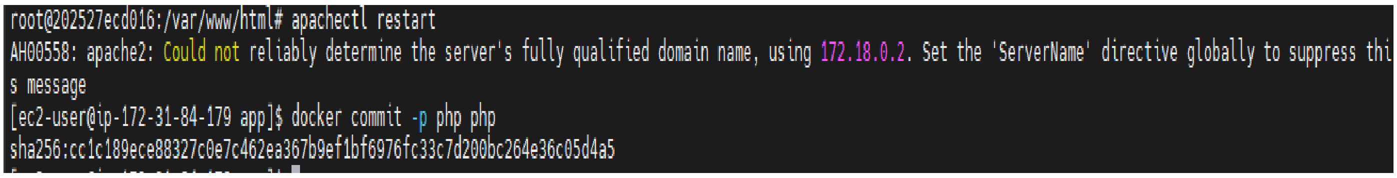


```
PORT=8080
~
~
~
~
```

.env file



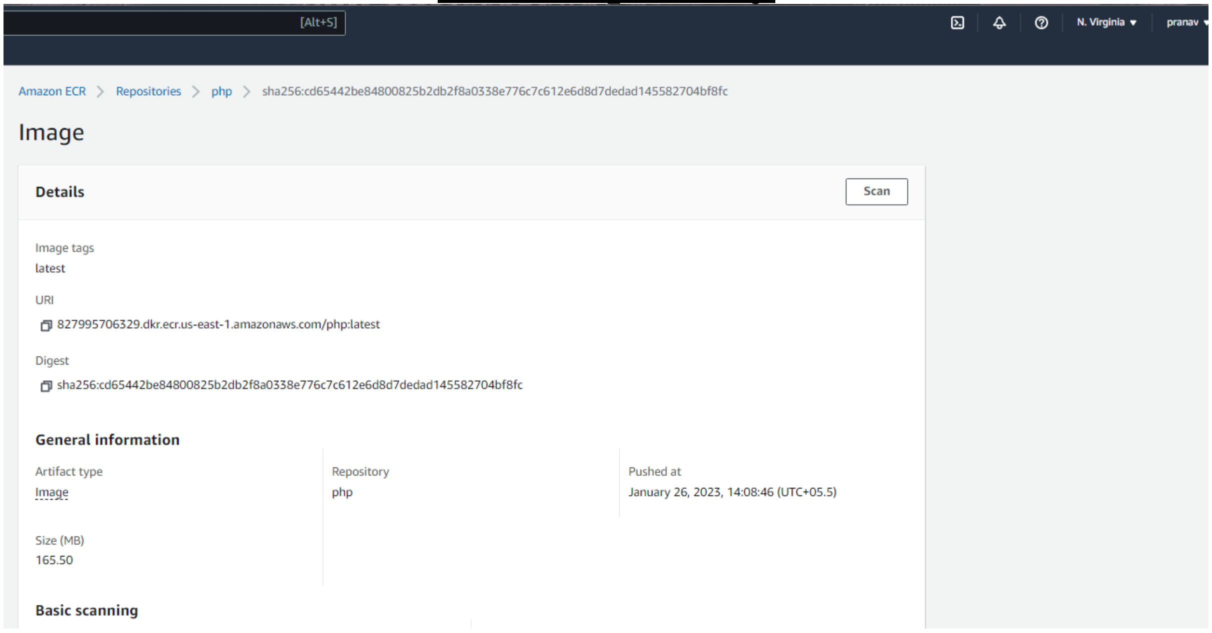
```
[ec2-user@ip-172-31-84-179 app]$ docker exec -it php bash
root@202527ecd016:/var/www/html# docker-php-ext-install mysqli
Configuring for:
PHP Api Version:         20210902
Zend Module Api No:      20210902
Zend Extension Api No:   420210902
checking for grep that handles long lines and -e... /bin/grep
checking for egrep... /bin/grep -E
checking for a sed that does not truncate output... /bin/sed
checking for pkg-config... /usr/bin/pkg-config
checking pkg-config is at least version 0.9.0... yes
checking for cc... cc
checking whether the C compiler works... yes
checking for C compiler default output file name... a.out
checking for suffix of executables...
checking whether we are cross compiling... no
checking for suffix of object files... o
checking whether we are using the GNU C compiler... yes
checking whether cc accepts -g... yes
checking for cc option to accept ISO C89... none needed
checking how to run the C preprocessor... cc -E
checking for icc... no
checking for suncc... no
checking for system library directory... lib
checking if compiler supports -Wl,-rpath,... yes
checking build system type... x86_64-pc-linux-gnu
checking host system type... x86_64-pc-linux-gnu
checking target system type... x86_64-pc-linux-gnu
checking for PHP prefix... /usr/local
```



```
root@202527ecd016:/var/www/html# apachectl restart
AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using 172.18.0.2. Set the 'ServerName' directive globally to suppress thi
s message
[ec2-user@ip-172-31-84-179 app]$ docker commit -p php php
sha256:cc1c189ece88327c0e7c462ea367b9ef1bf6976fc33c7d200bc264e36c05d4a5
```

Enable mysqli() inside the container

ECR Repository



Amazon ECR > Repositories > php > sha256:cd65442be84800825b2db2f8a0338e776c7c612e6d8d7dedad145582704bf8fc

Image

Details

Scan

Image tags
latest

URI
827995706329.dkr.ecr.us-east-1.amazonaws.com/php:latest

Digest
sha256:cd65442be84800825b2db2f8a0338e776c7c612e6d8d7dedad145582704bf8fc

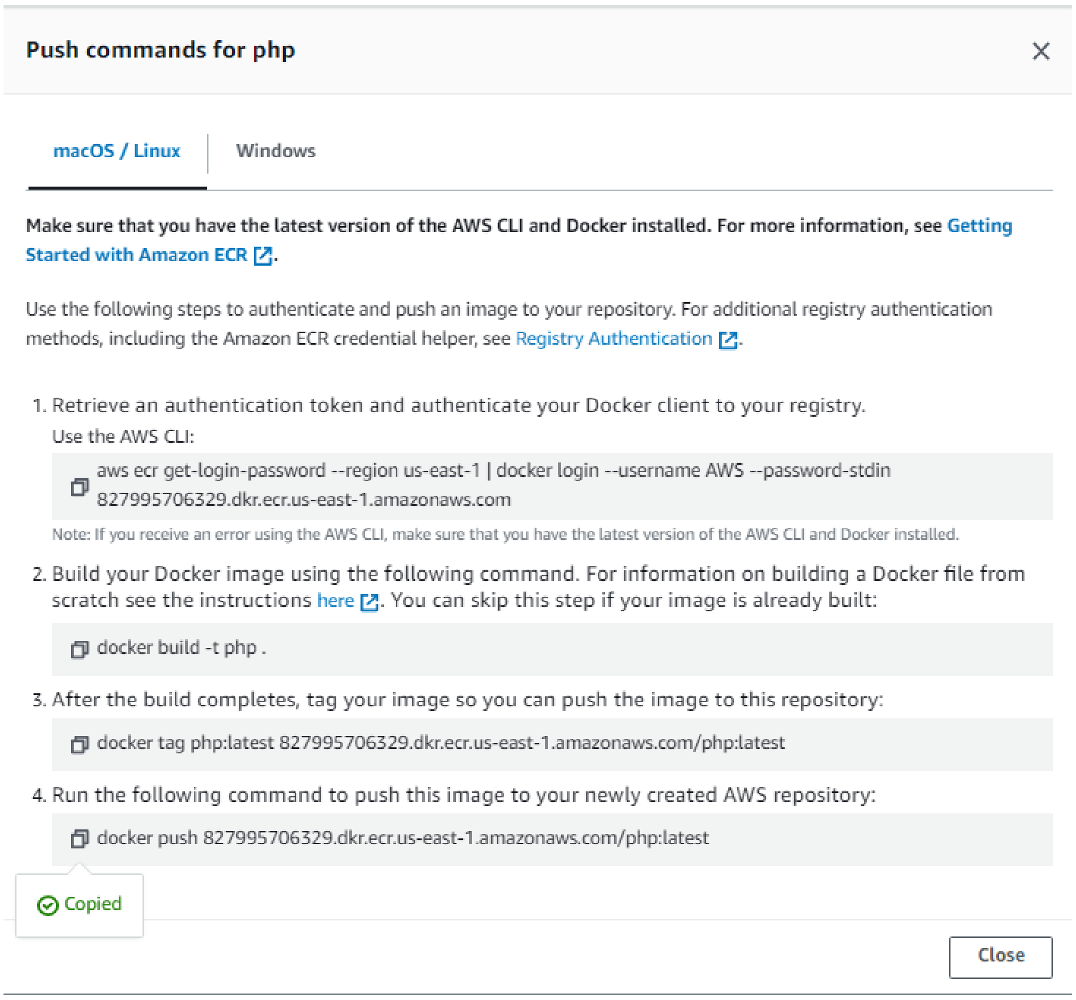
General information

Artifact type	Repository	Pushed at
Image	php	January 26, 2023, 14:08:46 (UTC+05.5)

Size (MB)
165.50

Basic scanning

Pushing the image to the ECR repository



Push commands for php

macOS / Linux | Windows

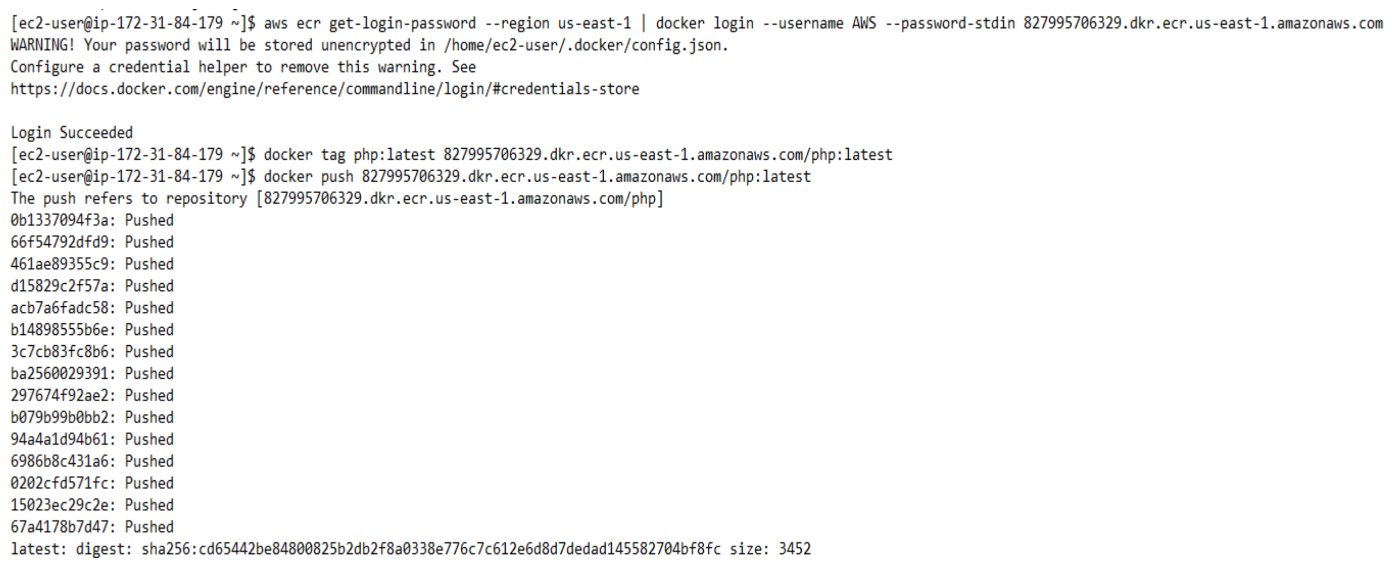
Make sure that you have the latest version of the AWS CLI and Docker installed. For more information, see Getting Started with Amazon ECR.

Use the following steps to authenticate and push an image to your repository. For additional registry authentication methods, including the Amazon ECR credential helper, see Registry Authentication.

1. Retrieve an authentication token and authenticate your Docker client to your registry.
   Use the AWS CLI:
   ```
   aws ecr get-login-password --region us-east-1 | docker login --username AWS --password-stdin
   827995706329.dkr.ecr.us-east-1.amazonaws.com
   ```
   Note: If you receive an error using the AWS CLI, make sure that you have the latest version of the AWS CLI and Docker installed.
2. Build your Docker image using the following command. For information on building a Docker file from scratch see the instructions here. You can skip this step if your image is already built:
   ```
   docker build -t php .
   ```
3. After the build completes, tag your image so you can push the image to this repository:
   ```
   docker tag php:latest 827995706329.dkr.ecr.us-east-1.amazonaws.com/php:latest
   ```
4. Run the following command to push this image to your newly created AWS repository:
   ```
   docker push 827995706329.dkr.ecr.us-east-1.amazonaws.com/php:latest
   ```

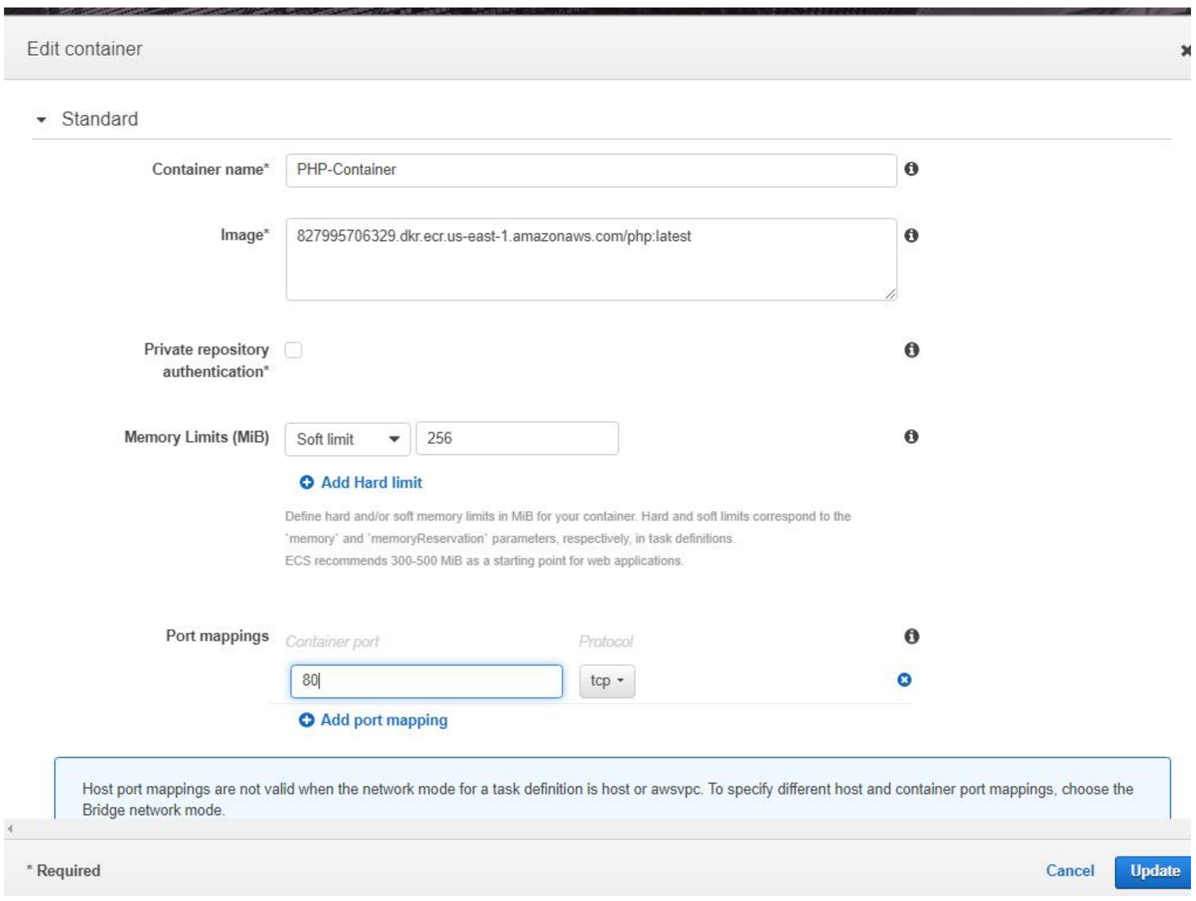
Copied

Close



```
[ec2-user@ip-172-31-84-179 ~]$ aws ecr get-login-password --region us-east-1 | docker login --username AWS --password-stdin 827995706329.dkr.ecr.us-east-1.amazonaws.com
WARNING! Your password will be stored unencrypted in /home/ec2-user/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
[ec2-user@ip-172-31-84-179 ~]$ docker tag php:latest 827995706329.dkr.ecr.us-east-1.amazonaws.com/php:latest
[ec2-user@ip-172-31-84-179 ~]$ docker push 827995706329.dkr.ecr.us-east-1.amazonaws.com/php:latest
The push refers to repository [827995706329.dkr.ecr.us-east-1.amazonaws.com/php]
0b1337094f3a: Pushed
66f54792dfd9: Pushed
461ae89355c9: Pushed
d15829c2f57a: Pushed
acb7a6fadc58: Pushed
b14898555b6e: Pushed
3c7cb83fc8b6: Pushed
ba2560029391: Pushed
297674f92ae2: Pushed
b079b99b0bb2: Pushed
94a4a1d94b61: Pushed
6986b8c431a6: Pushed
0202cfd571fc: Pushed
15023ec29c2e: Pushed
67a4178b7d47: Pushed
latest: digest: sha256:cd65442be84800825b2db2f8a0338e776c7c612e6d8d7dedad145582704bf8fc size: 3452
```

Configuring the cluster



Edit container

Standard

Container name* PHP-Container

Image* 827995706329.dkr.ecr.us-east-1.amazonaws.com/php:latest

Private repository authentication* ☐

Memory Limits (MiB) Soft limit 256

Add Hard limit

Define hard and/or soft memory limits in MiB for your container. Hard and soft limits correspond to the `memory` and `memoryReservation` parameters, respectively, in task definitions.
ECS recommends 300-500 MiB as a starting point for web applications.

Port mappings

Container port	Protocol
80	tcp

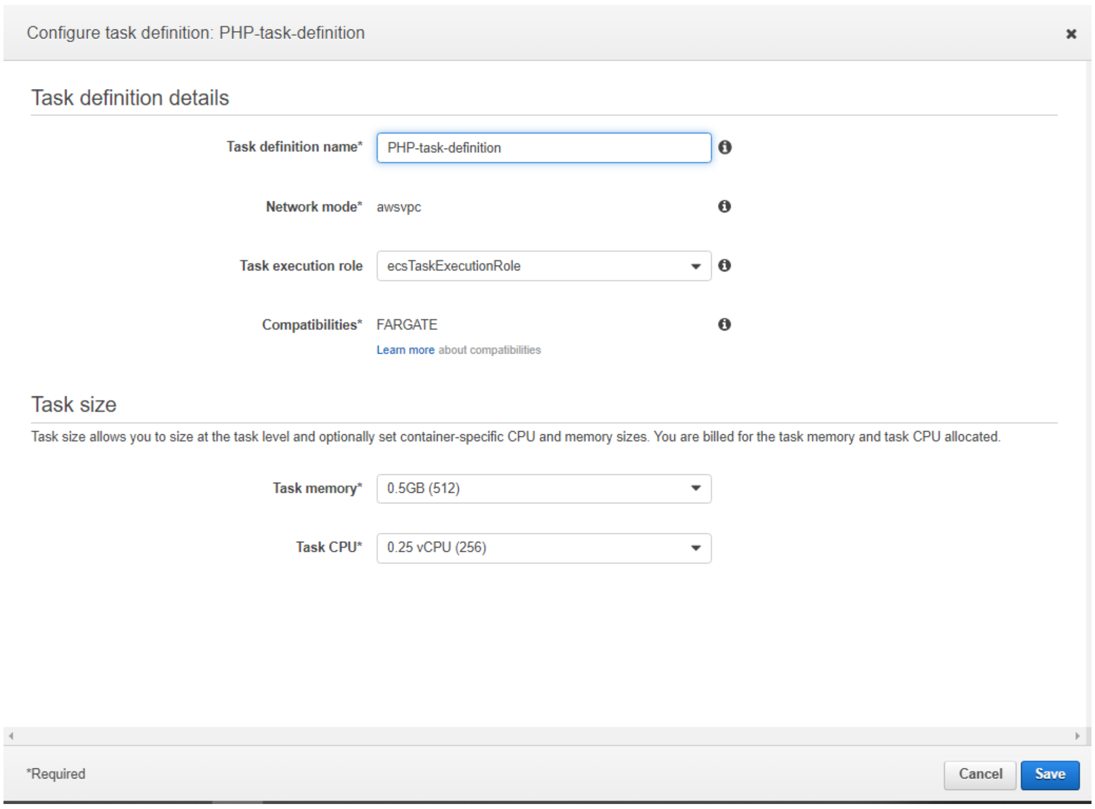
Add port mapping

Host port mappings are not valid when the network mode for a task definition is host or awsvpc. To specify different host and container port mappings, choose the Bridge network mode.

* Required

Cancel Update

Creating container



Configure task definition: PHP-task-definition

Task definition details

Task definition name*	PHP-task-definition
Network mode*	awsvpc
Task execution role	ecsTaskExecutionRole
Compatibilities*	FARGATE

Learn more about compatibilities

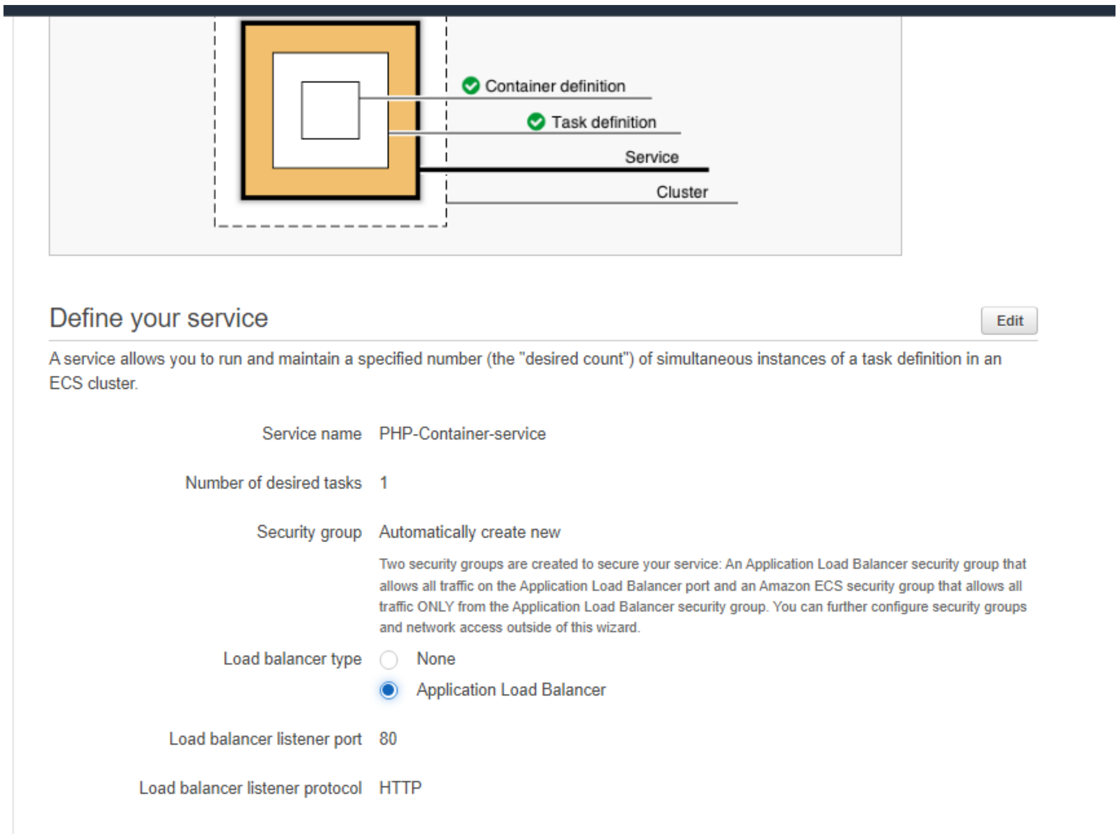
Task size

Task size allows you to size at the task level and optionally set container-specific CPU and memory sizes. You are billed for the task memory and task CPU allocated.

Task memory*	0.5GB (512)
Task CPU*	0.25 vCPU (256)

*Required

Creating task-definition



Define your service

A service allows you to run and maintain a specified number (the "desired count") of simultaneous instances of a task definition in an ECS cluster.

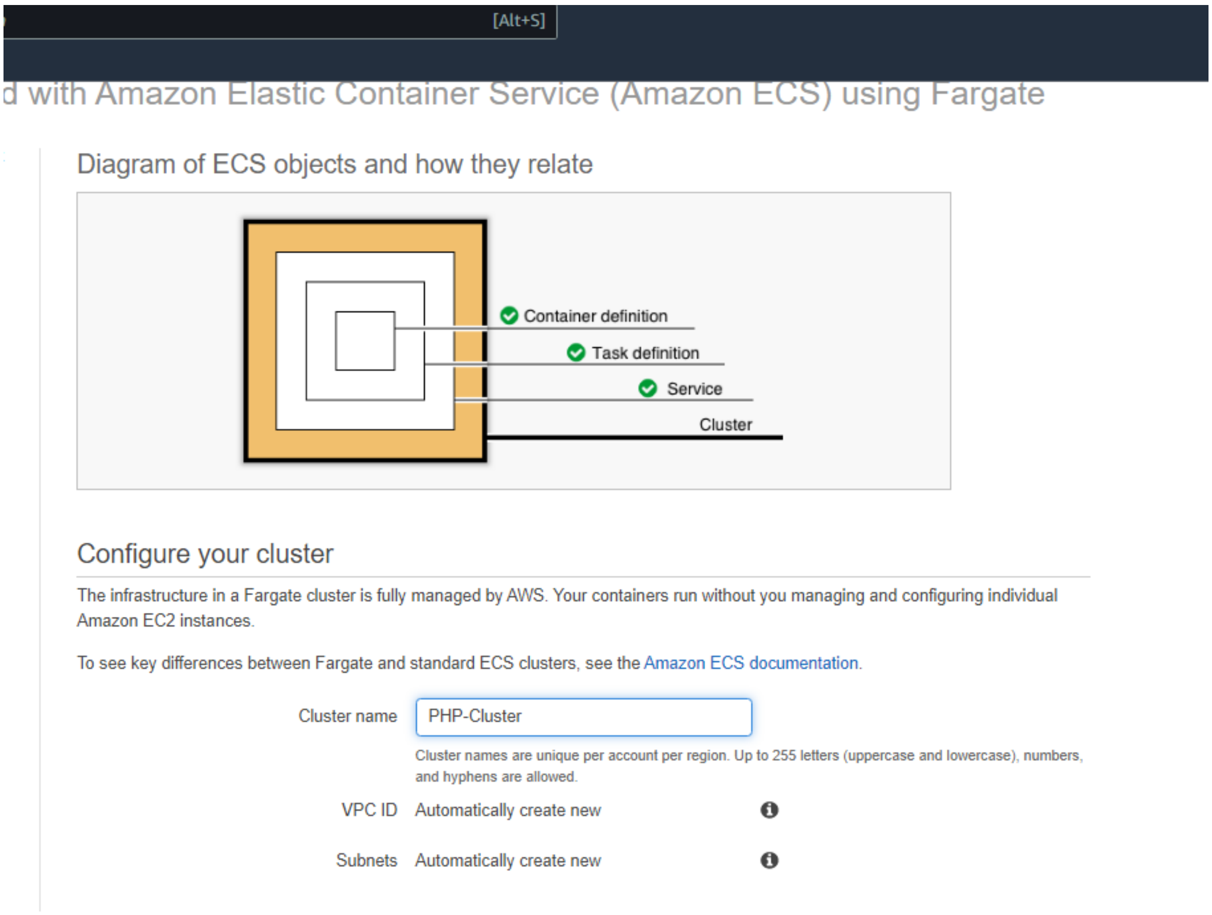
Service name	PHP-Container-service
Number of desired tasks	1
Security group	Automatically create new

Two security groups are created to secure your service: An Application Load Balancer security group that allows all traffic on the Application Load Balancer port and an Amazon ECS security group that allows all traffic ONLY from the Application Load Balancer security group. You can further configure security groups and network access outside of this wizard.

Load balancer type: None [✓ Application Load Balancer]

Load balancer listener port	80
Load balancer listener protocol	HTTP

Creating service


[Alt+S]

d with Amazon Elastic Container Service (Amazon ECS) using Fargate

Diagram of ECS objects and how they relate

Configure your cluster

The infrastructure in a Fargate cluster is fully managed by AWS. Your containers run without you managing and configuring individual Amazon EC2 instances.

To see key differences between Fargate and standard ECS clusters, see the Amazon ECS documentation.

Cluster name: PHP-Cluster

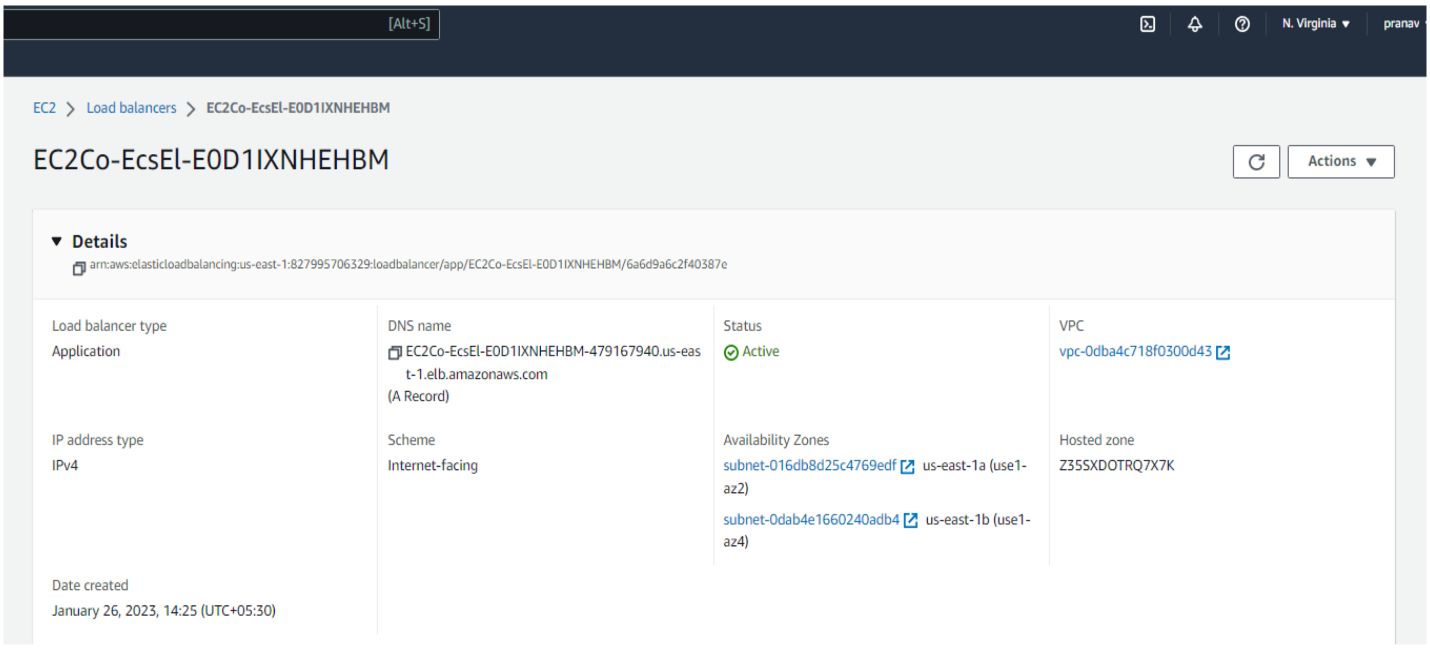
Cluster names are unique per account per region. Up to 255 letters (uppercase and lowercase), numbers, and hyphens are allowed.

VPC ID: Automatically create new

Subnets: Automatically create new

Creating Cluster

Final Output



EC2 > Load balancers > EC2Co-EcsEl-E0D1IXNHEHBM

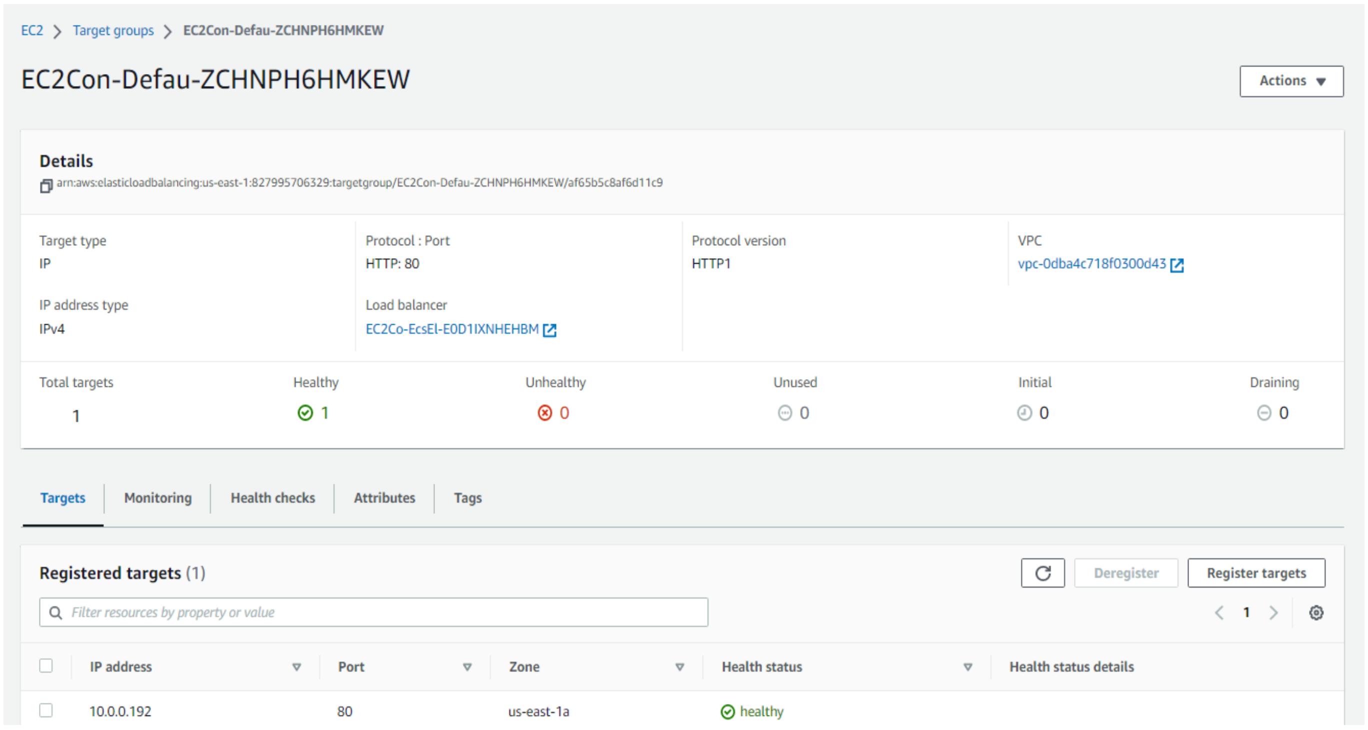
EC2Co-EcsEl-E0D1IXNHEHBM

Details

arn:aws:elasticloadbalancing:us-east-1:827995706329:loadbalancer/app/EC2Co-EcsEl-E0D1IXNHEHBM/6a6d9a6c2f40387e

Load balancer type Application	DNS name EC2Co-EcsEl-E0D1IXNHEHBM-479167940.us-east-1.elb.amazonaws.com (A Record)	Status Active	VPC vpc-0dba4c718f0300d43
IP address type IPv4	Scheme Internet-facing	Availability Zones subnet-016db8d25c4769edf us-east-1a (use1-az2) subnet-0dab4e1660240adb4 us-east-1b (use1-az4)	Hosted zone Z35SXDOTRQ7X7K
Date created January 26, 2023, 14:25 (UTC+05:30)			

Load Balancer created



EC2 > Target groups > EC2Con-Defau-ZCHNPH6HMKEW

EC2Con-Defau-ZCHNPH6HMKEW

Details

arn:aws:elasticloadbalancing:us-east-1:827995706329:targetgroup/EC2Con-Defau-ZCHNPH6HMKEW/af65b5c8af6d11c9

Target type IP	Protocol : Port HTTP: 80	Protocol version HTTP1	VPC vpc-0dba4c718f0300d43
IP address type IPv4	Load balancer EC2Co-EcsEl-E0D1IXNHEHBM		

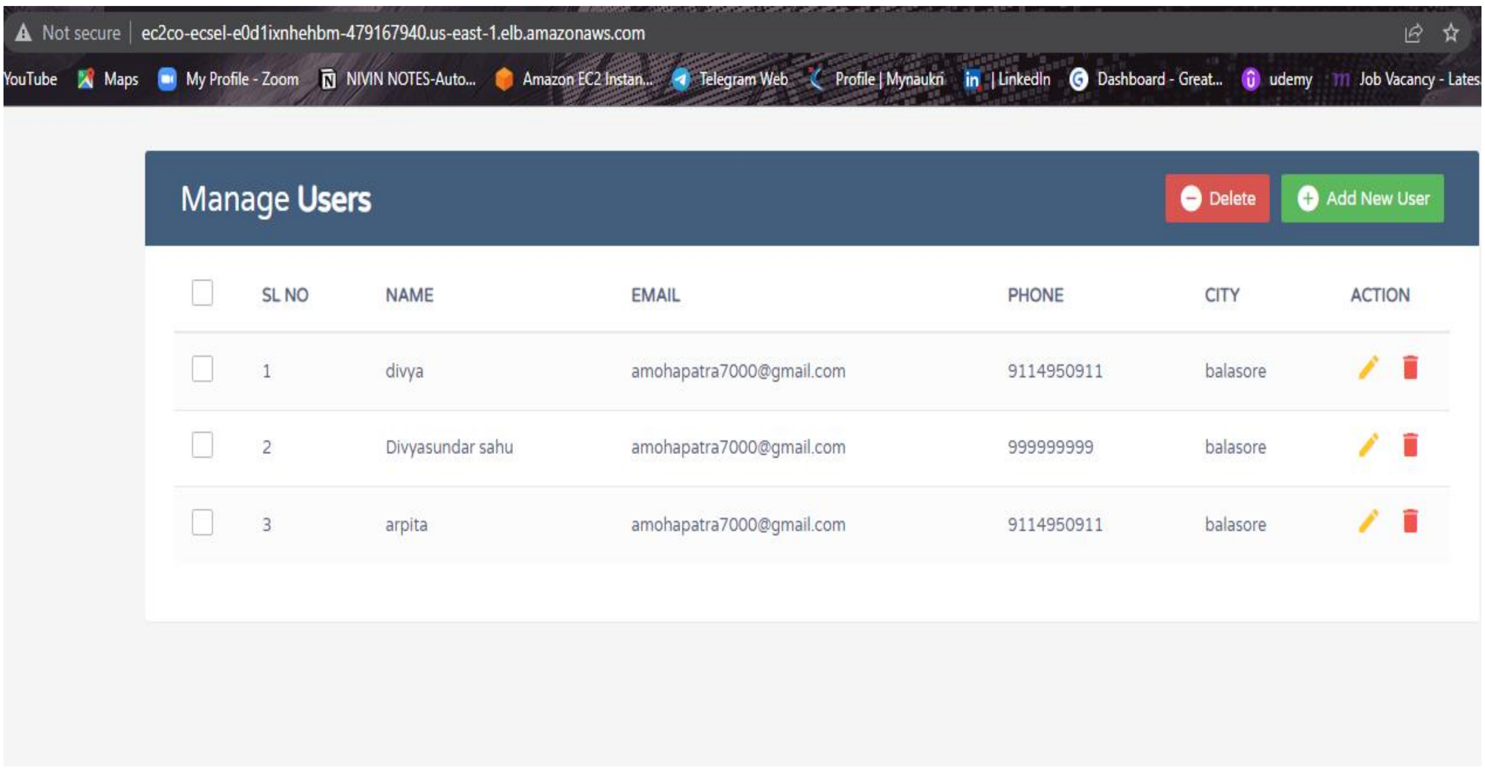
Total targets	Healthy	Unhealthy	Unused	Initial	Draining
1	✓ 1	⊗ 0	0	0	0

Registered targets (1)

IP address	Port	Zone	Health status	Health status details
10.0.0.192	80	us-east-1a	✓ healthy	

Target Group created

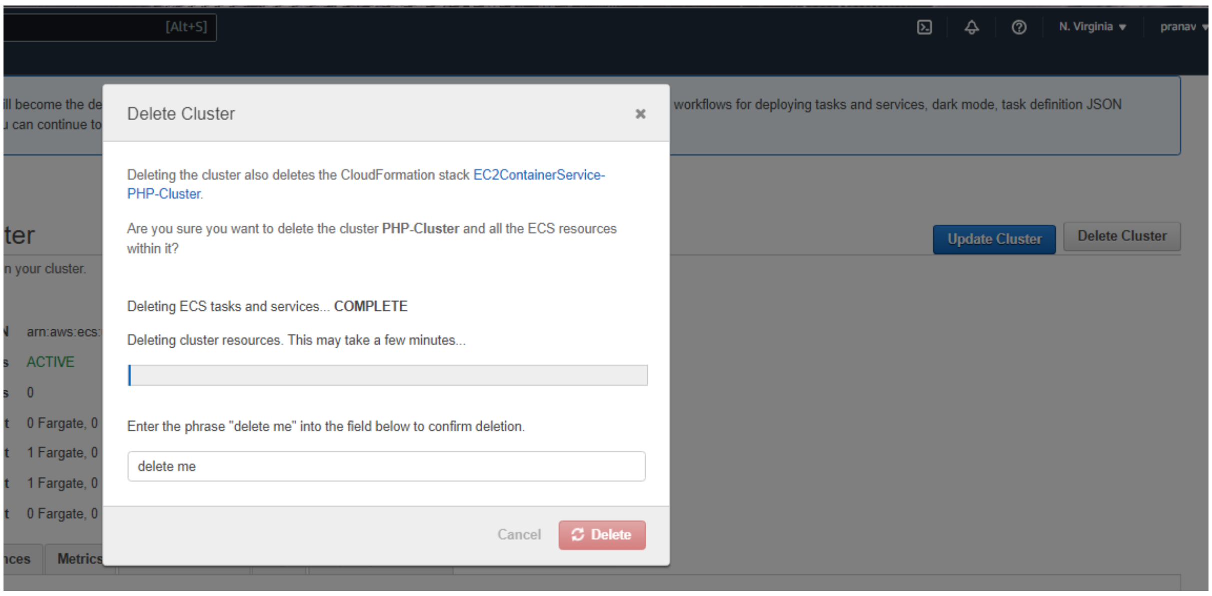
Accessing the PHP application on the web browser using the port configured in ECS



Manage Users

	SL NO	NAME	EMAIL	PHONE	CITY	ACTION
☐	1	divya	amohapatra7000@gmail.com	9114950911	balasore	✎ 🗑
☐	2	Divyasundar sahu	amohapatra7000@gmail.com	999999999	balasore	✎ 🗑
☐	3	arpita	amohapatra7000@gmail.com	9114950911	balasore	✎ 🗑

Resource Cleaning up!:


Delete Cluster

Deleting the cluster also deletes the CloudFormation stack EC2ContainerService-PHP-Cluster.

Are you sure you want to delete the cluster **PHP-Cluster** and all the ECS resources within it?

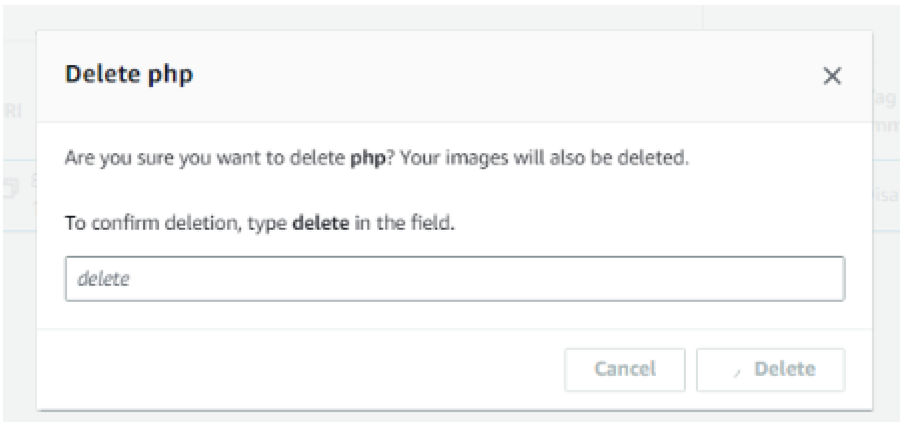
Deleting ECS tasks and services... COMPLETE

Deleting cluster resources. This may take a few minutes...

Enter the phrase "delete me" into the field below to confirm deletion.

delete me

Deleting Cluster

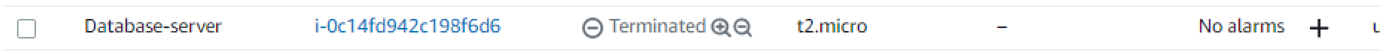

Delete php

Are you sure you want to delete **php**? Your images will also be deleted.

To confirm deletion, type **delete** in the field.

delete

Deleting ECR repository


Database-server	i-0c14fd942c198f6d6	Terminated	t2.micro	–	No alarms	+

Terminating instance