FACULTY OF INFORMATION SCIENCE AND TECHNOLOGY

MULTIMEDIA UNIVERSITY

MALAYSIA

TCC3141 CLOUD COMPUTING

TRIMESTER 1, 2022/202

PROJECT REPORT

No	Student Name	Student ID	Programme
1	GRAYSON GOH JIN YI (GROUP LEADER)	1191101340	BIT ST
2	SIAH KAH CHUAN	1191100577	BIT ST
3	FOO HAW LIANG	1191101497	BIT ST
4	ALVIN KUA CHEE SHERN	1181103380	BIT ST
5	GOH JIN JET	1191101375	BIT ST

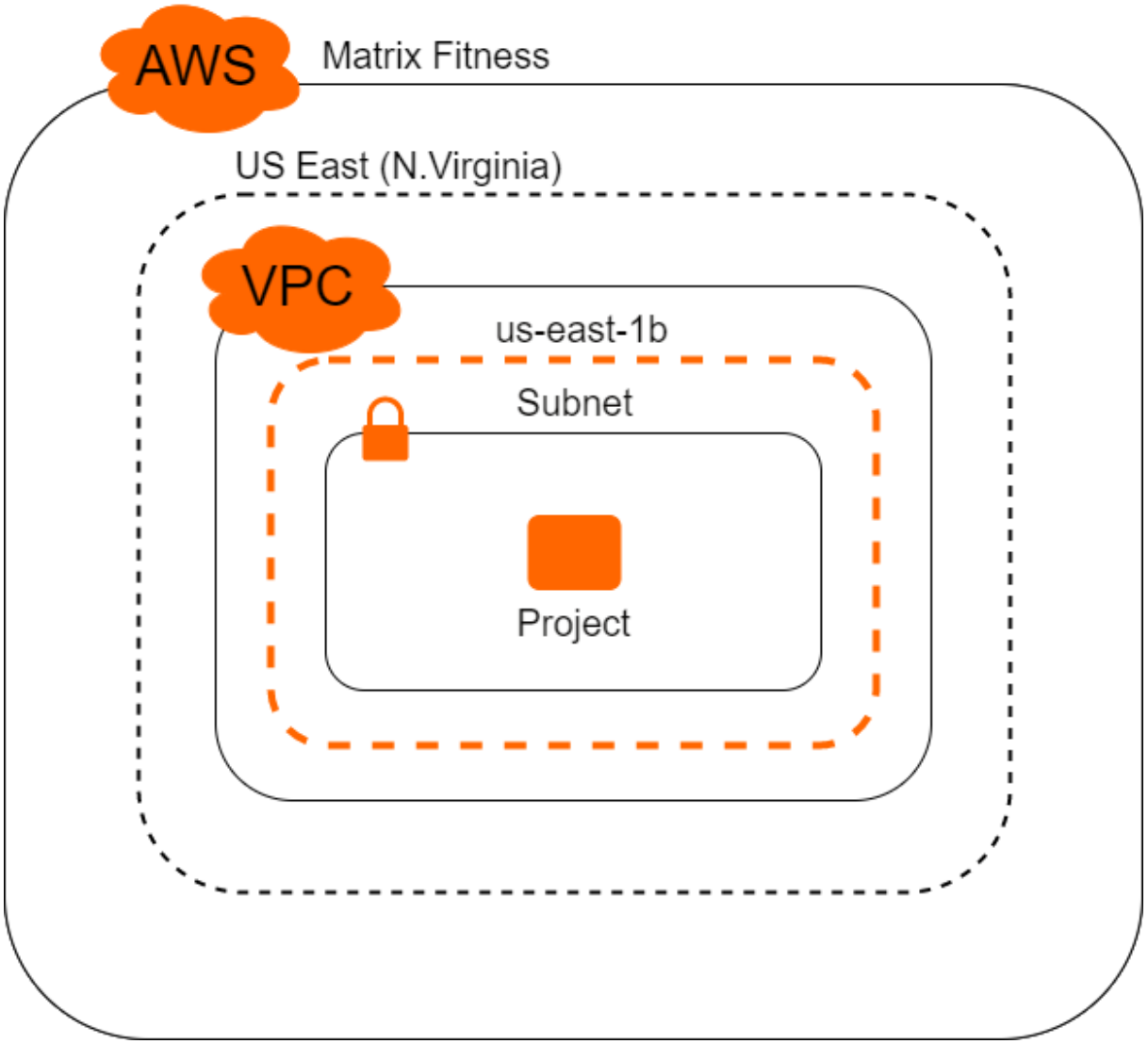
Table of Content

Phase 1

Introduction of Website

The website we implemented is mainly focused for the admin to do management on the products. The products are all related to fitness equipment. The company name that we applied to is Matrix Fitness. Admin will be required to insert the name and password in order to access the system. The website is designed to be ease of use, user-friendly and simple for admins to proceed any interaction and manipulation. The IP of the website is **projectlb-1004533238.us-east-1.elb.amazonaws.com**

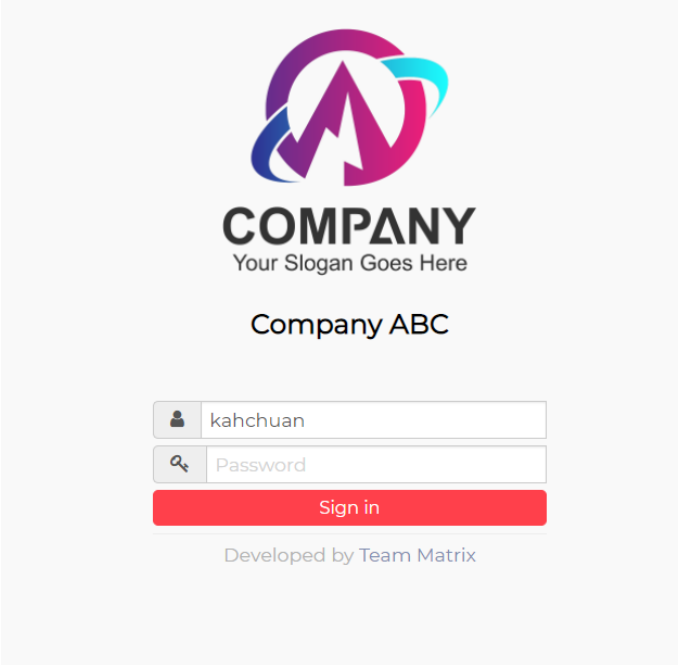
Layout Diagram of the networks



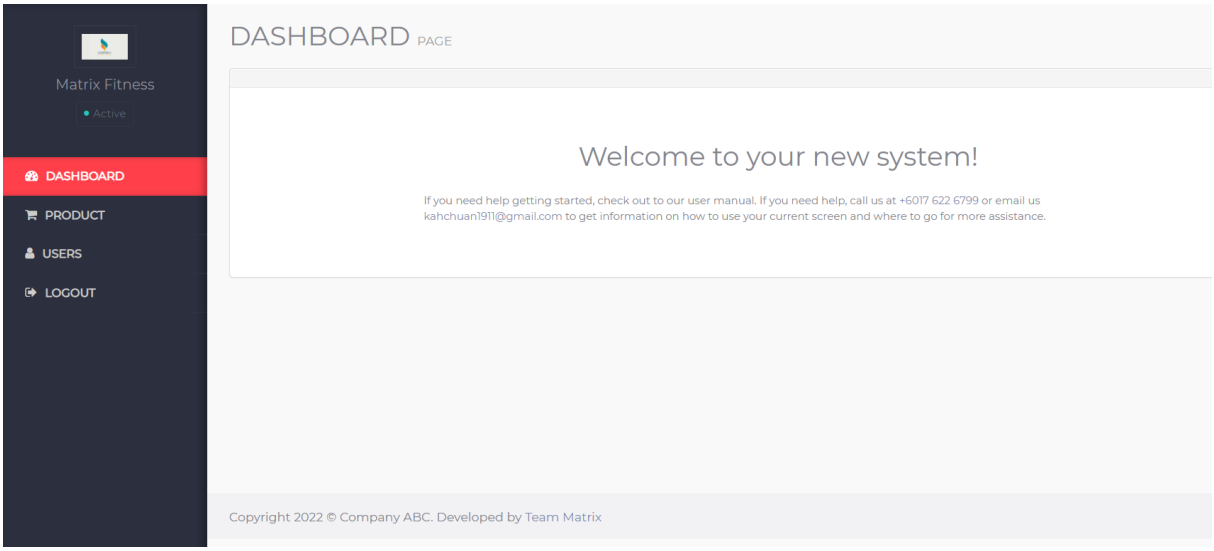
Phase 1 Implementation steps

First, prepare the e-commerce website template as shown below:

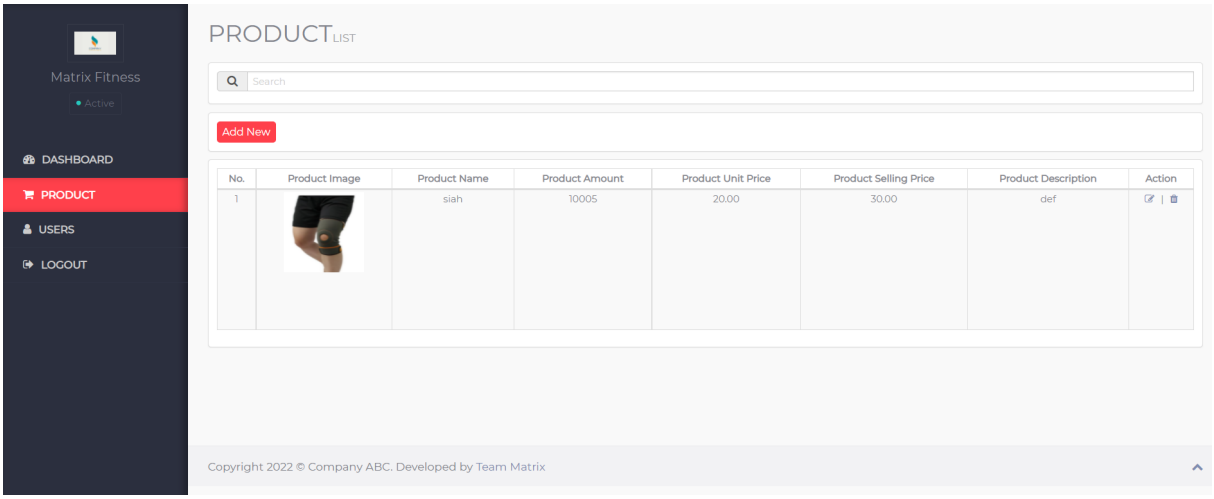
Login Page



Home Page

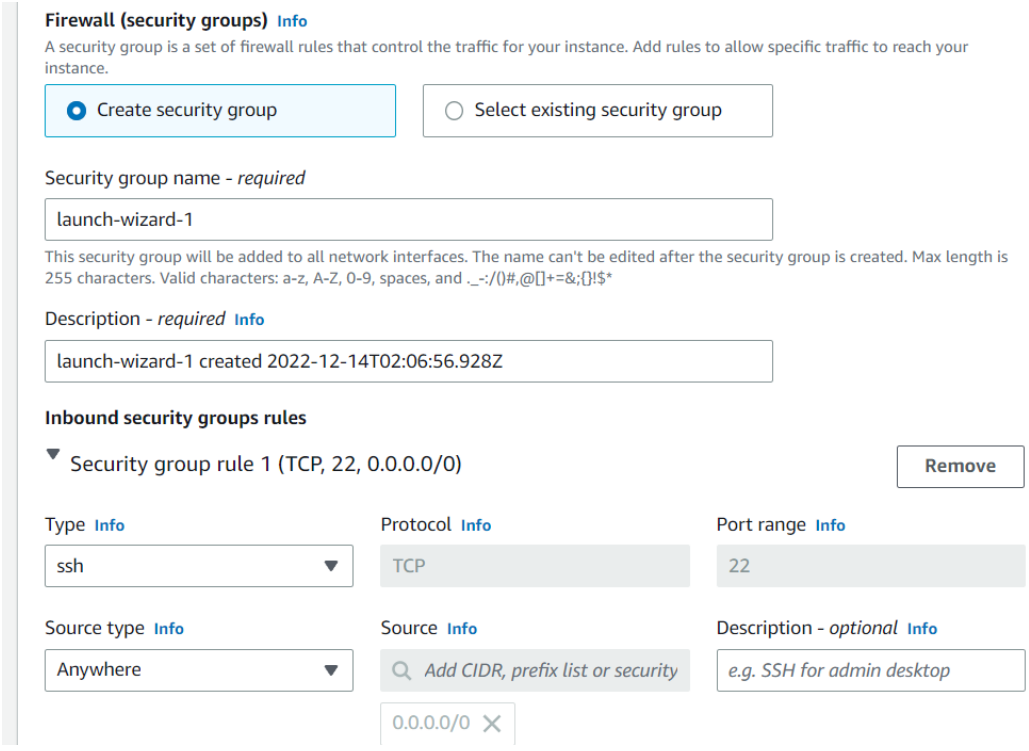


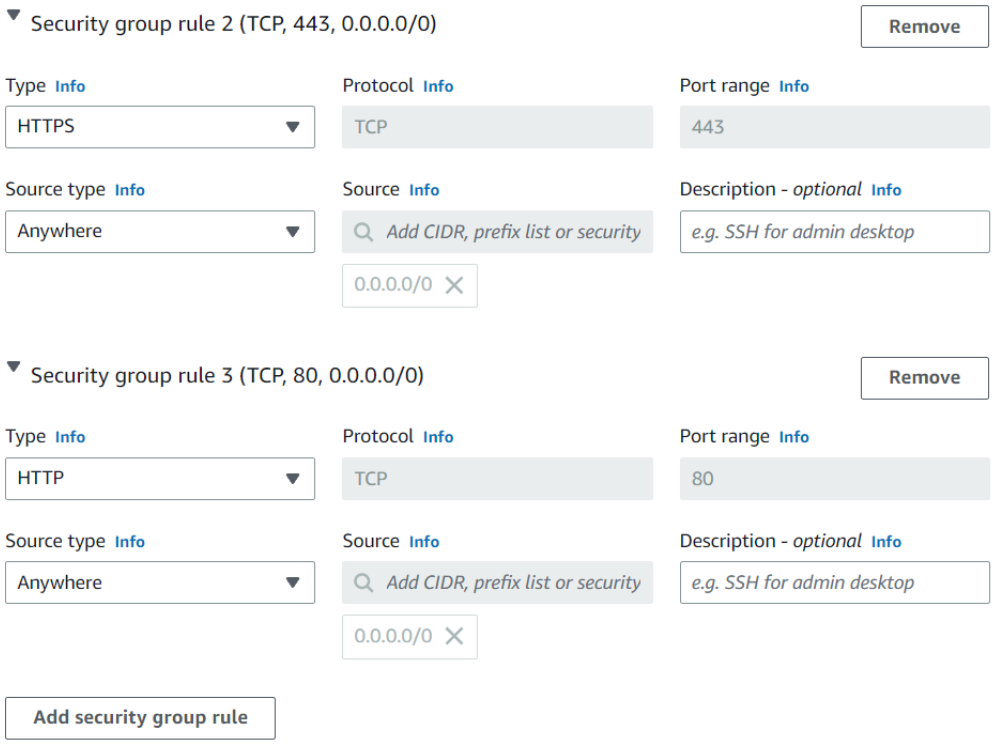
Product Page



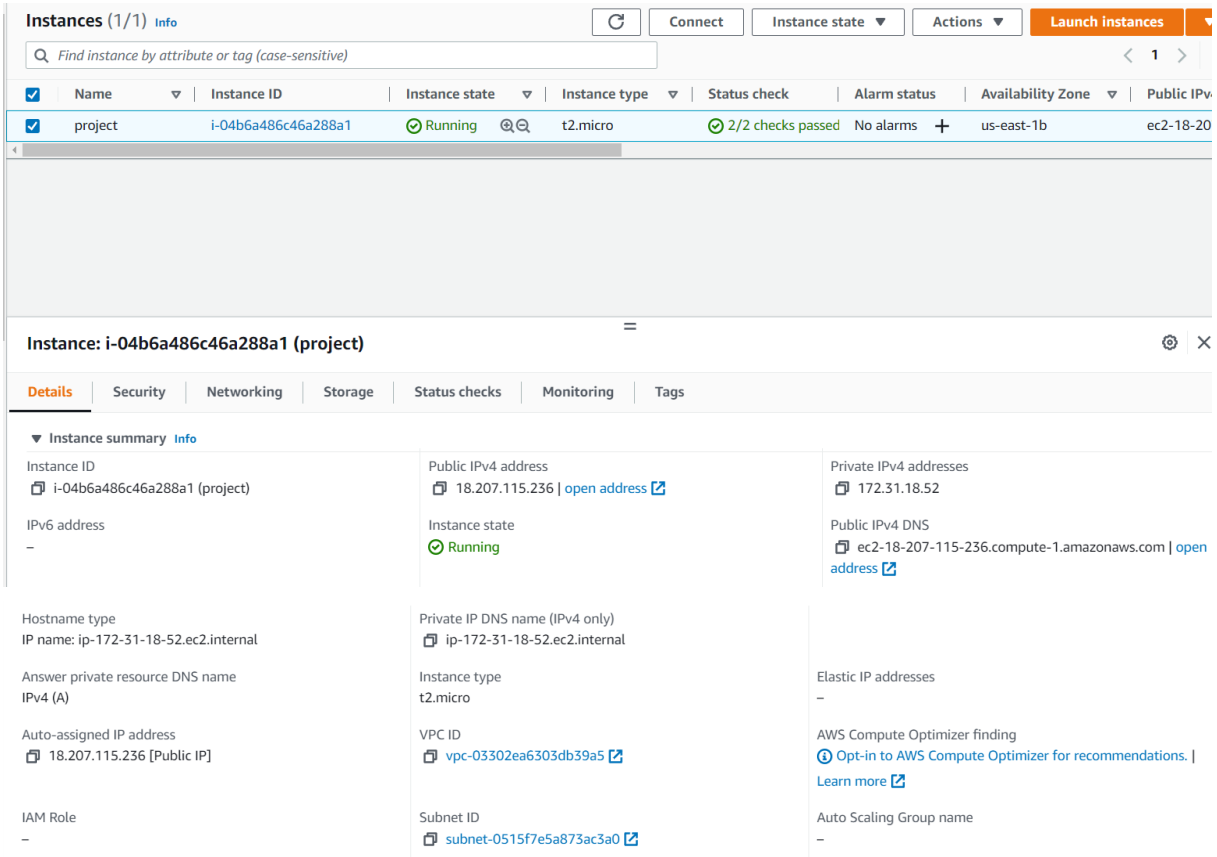
Next, launch the EC2 with the following setting:

- Name the instance as project
- Choose the Amazon Linux 2 AMI (HVM) - Kernel 5.10, SSD Volume Type
- For the instance type, choose the t2.micro
- For the Firewall (security group), choose create security group and with the following setting:

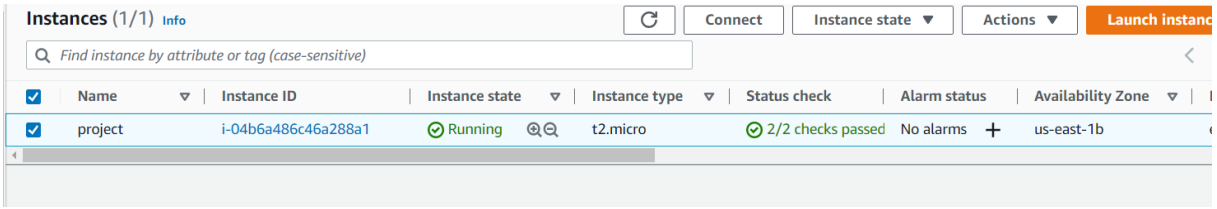




- For the key pair, choose create new key pair, name with project and key pair type select RSA, the private key file format select ppk and save the **project.ppk.**
- After all the settings are done, press Launch Instance. Below show the launched EC2 instances:



After launching the EC2 instances, choose the Connect button as shown below to move to the CLI.



After that, install the packages as shown as below:

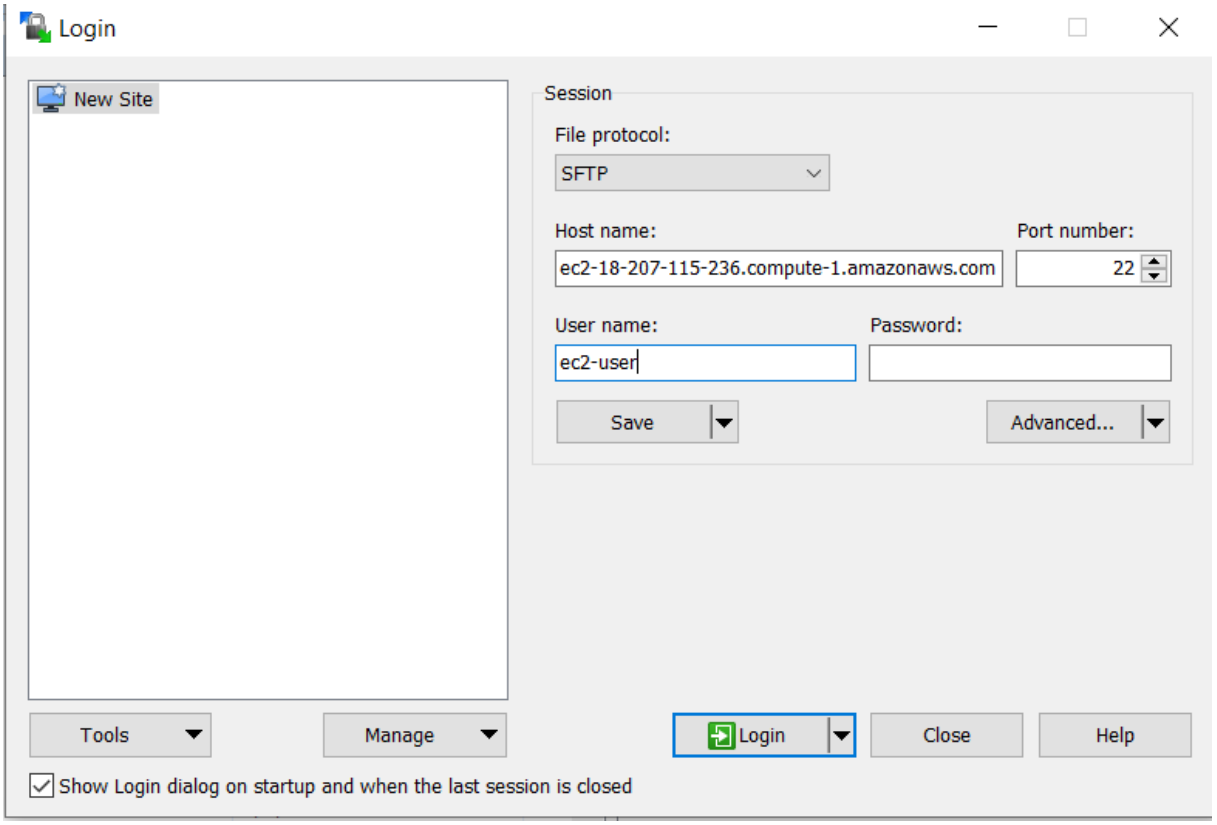
```
[ec2-user@ip-172-31-18-52 ~]$ sudo yum list installed httpd mariadb-server php-mysqlnd
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Installed Packages
httpd.x86_64                                          2.4.54-1.amzn2
mariadb-server.x86_64                                 3:10.2.38-1.amzn2.0.1
php-mysqlnd.x86_64                                    7.2.34-1.amzn2
[ec2-user@ip-172-31-18-52 ~]$
```

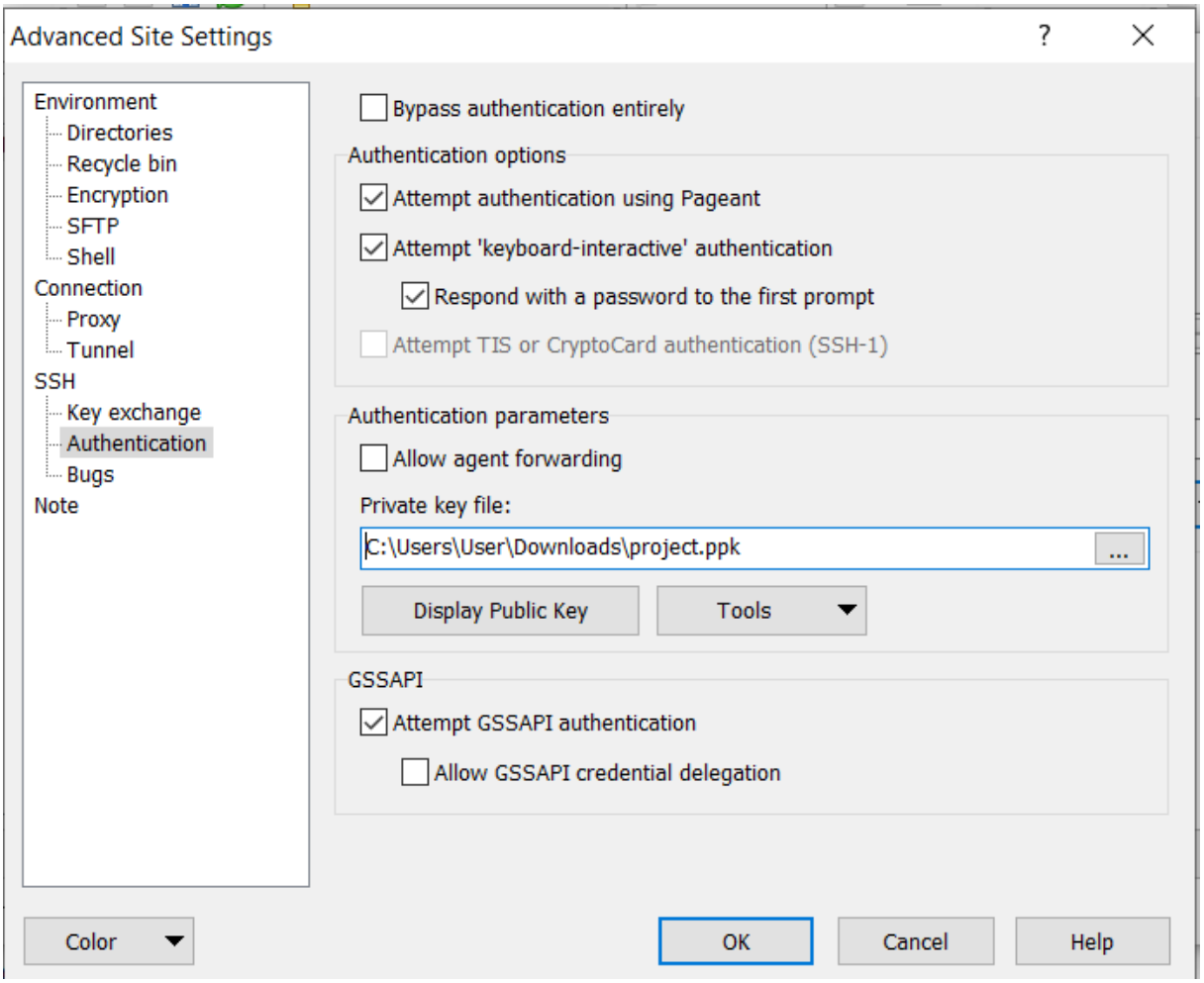
After installed all the above packages, enabled it by using the following command:

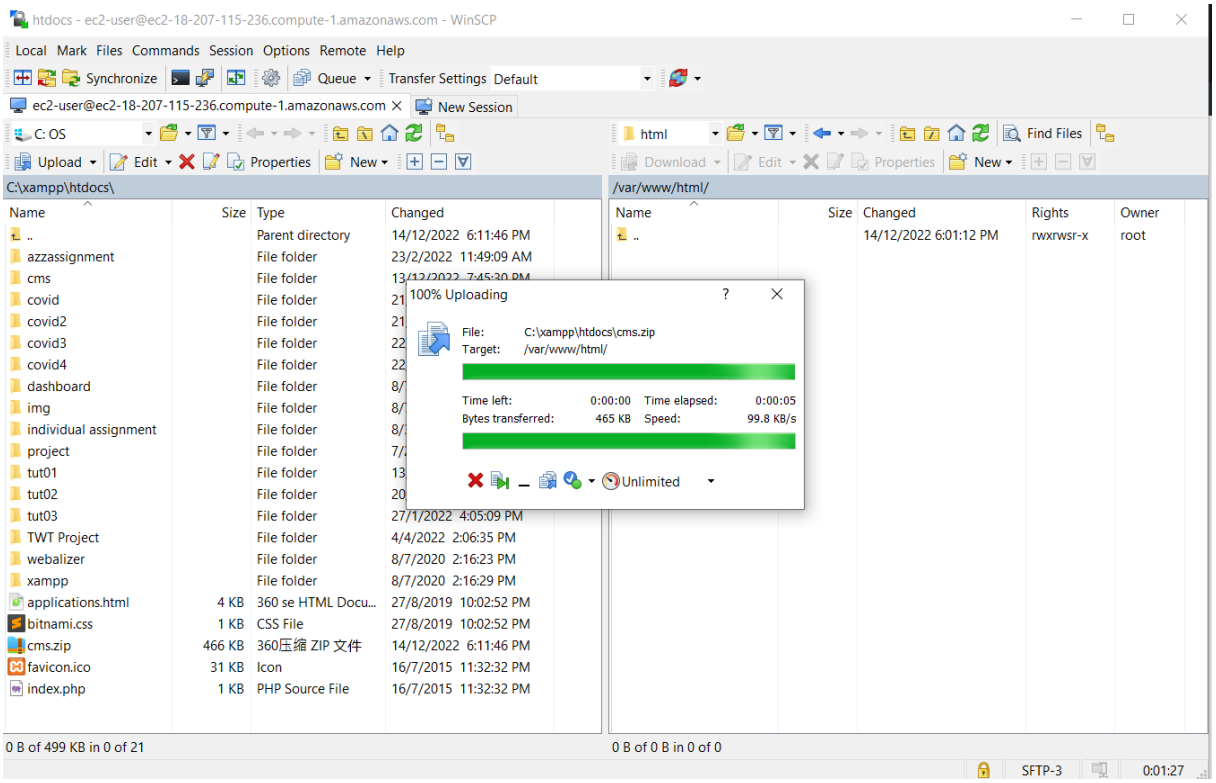
- sudo systemctl start httpd
- sudo systemctl enable httpd

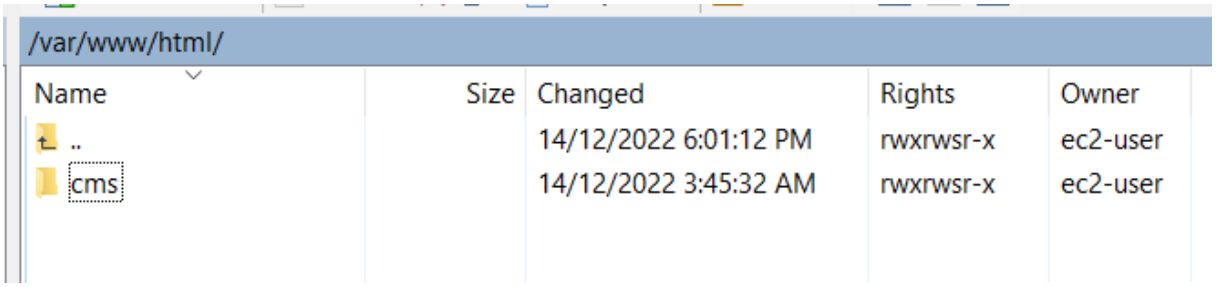
After opening the httpd, we need to login into WinSCP using the private key **project.ppk** we have created just now. Then we need to transfer the web template to the EC2 instances by using WinSCP

The following steps have been shown below:




Advanced Site Settings
Environment
Directories
Recycle bin
Encryption
SFTP
Shell
Connection
Proxy
Tunnel
SSH
Key exchange
Authentication
Bugs
Note
Bypass authentication entirely
Authentication options
Attempt authentication using Pageant
Attempt 'keyboard-interactive' authentication
Respond with a password to the first prompt
Attempt TIS or CryptoCard authentication (SSH-1)
Authentication parameters
Allow agent forwarding
Private key file:
C:\Users\User\Downloads\project.ppk
Display Public Key
Tools
GSSAPI
Attempt GSSAPI authentication
Allow GSSAPI credential delegation
Color
OK
Cancel
Help


htdocs - ec2-user@ec2-18-207-115-236.compute-1.amazonaws.com - WinSCP
Local Mark Files Commands Session Options Remote Help
Synchronize
Queue
Transfer Settings Default
ec2-user@ec2-18-207-115-236.compute-1.amazonaws.com
New Session
C: OS
Upload
Edit
Properties
New
C:\xampp\htdocs\
html
Download
Find Files
/var/www/html/
Name
Size
Type
Changed
Rights
Owner
100% Uploading
File: C:\xampp\htdocs\cms.zip
Target: /var/www/html/
Time left: 0:00:00
Time elapsed: 0:00:05
Bytes transferred: 465 KB
Speed: 99.8 KB/s
Unlimited
0 B of 499 KB in 0 of 21
0 B of 0 B in 0 of 0
SFTP-3
0:01:27



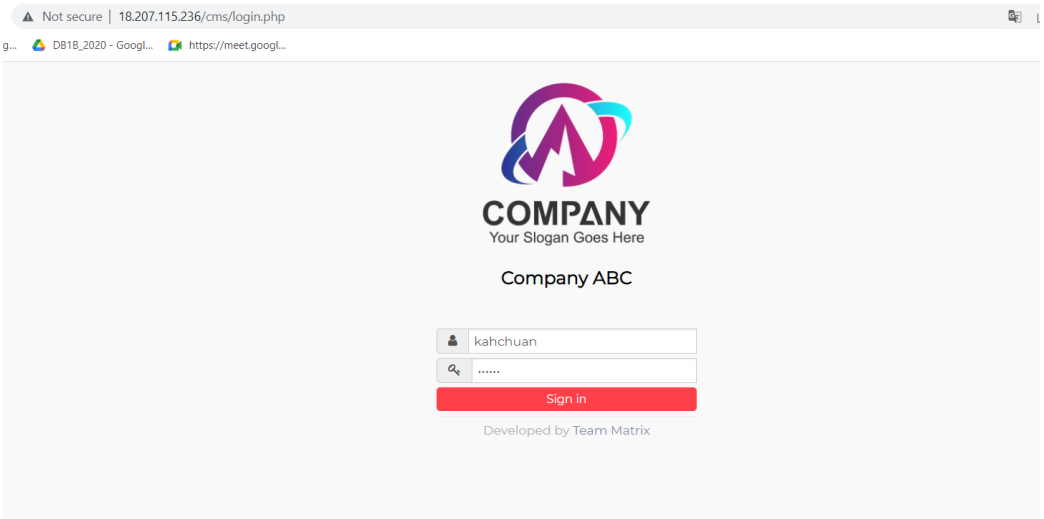
After transferring the web template to the EC2 instances, we move back to CLI to configure the database. The steps have been shown below:

```
[ec2-user@ip-172-31-18-52 ~]$ sudo mysql -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 22
Server version: 10.2.38-MariaDB MariaDB Server

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

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

MariaDB [(none)]> CREATE DATABASE system_cms;
Query OK, 1 row affected (0.00 sec)
```

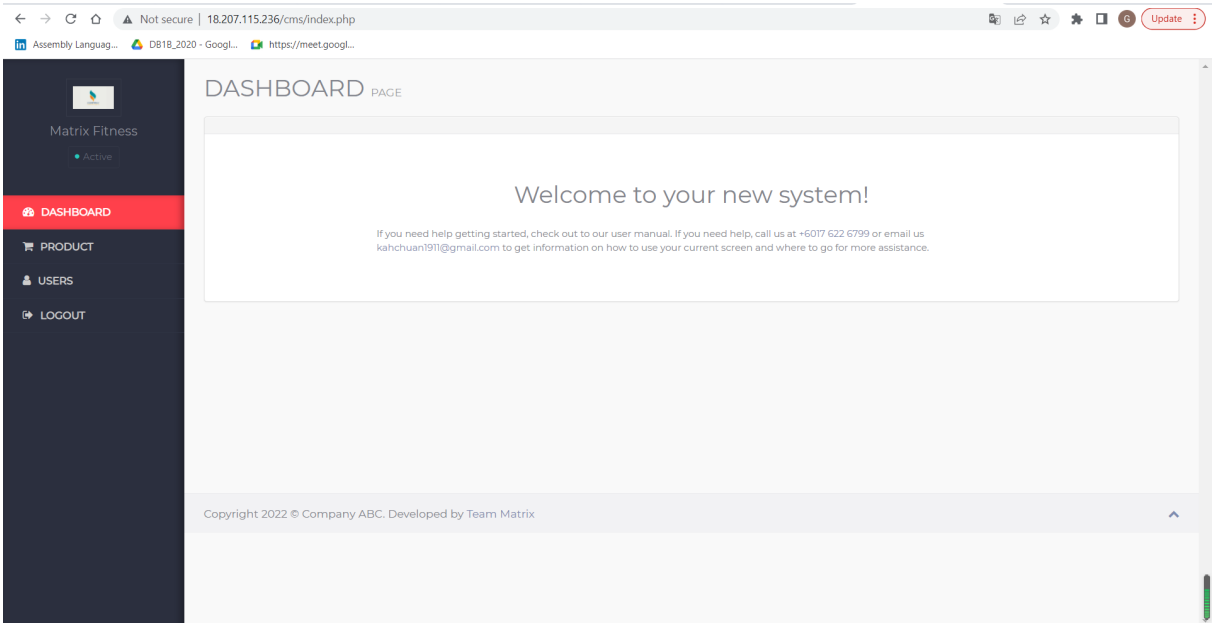
```
https://aws.amazon.com/amazon-linux-2/
[ec2-user@ip-172-31-18-52 ~]$ cd /var/www/html/cms/Connect
[ec2-user@ip-172-31-18-52 Connect]$ sudo mysql -u root -p system_cms < system_user.sql
Enter password:
[ec2-user@ip-172-31-18-52 Connect]$ sudo mysql -u root -p system_cms < system_product.sql
Enter password:
[ec2-user@ip-172-31-18-52 Connect]$
```

After all the set up the website can be fully functional.
IP of the website: http://18.207.115.236/cms/login.php

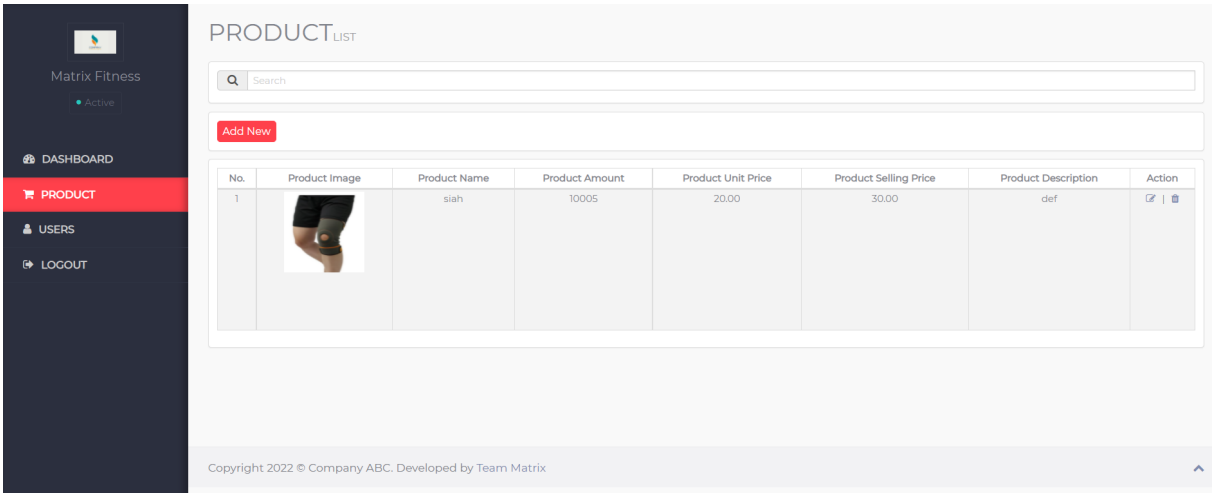
Login Page



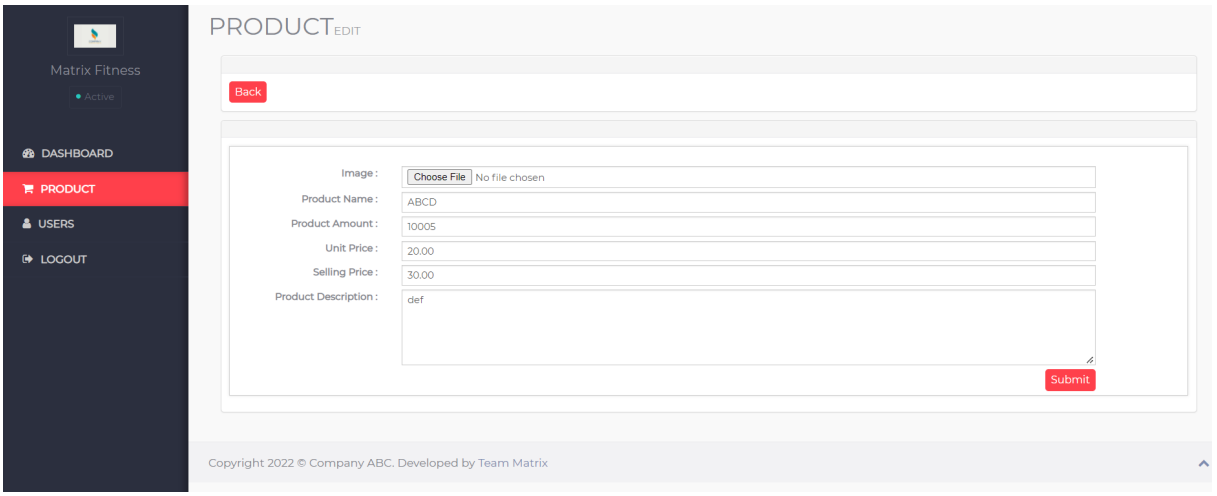
Home Page



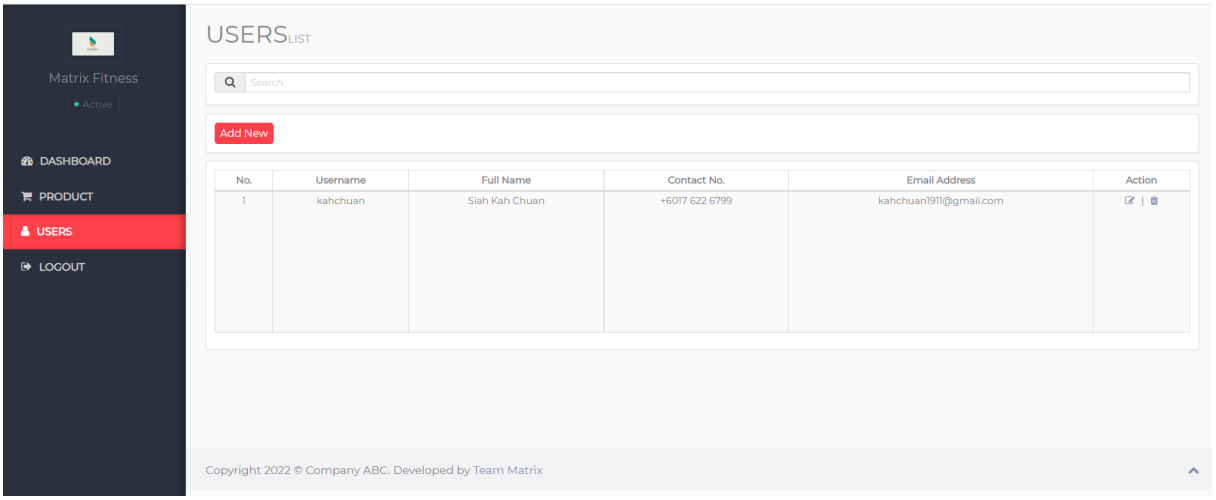
Product Page



Product Editing

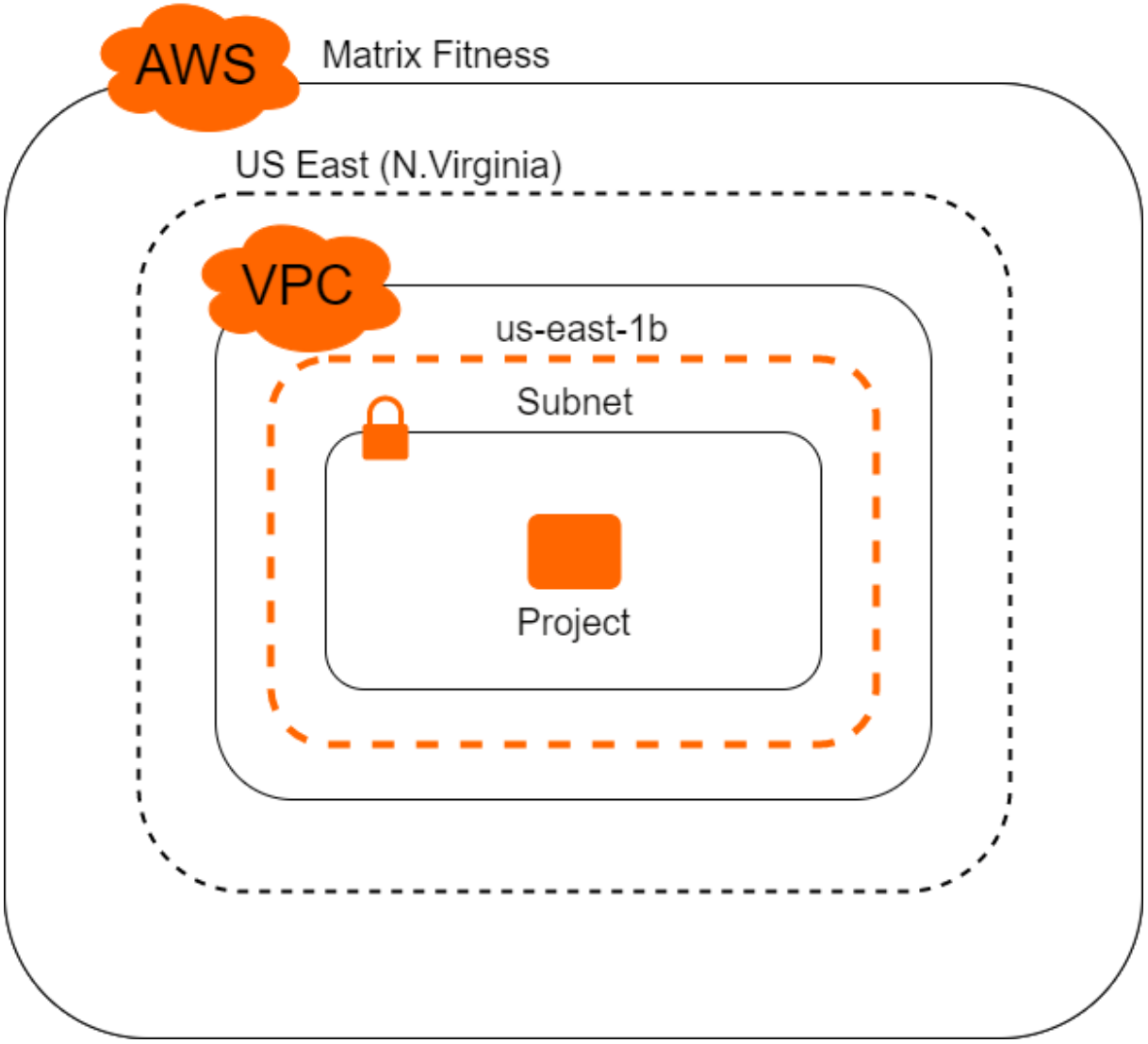


User Admin Page



Phase 2

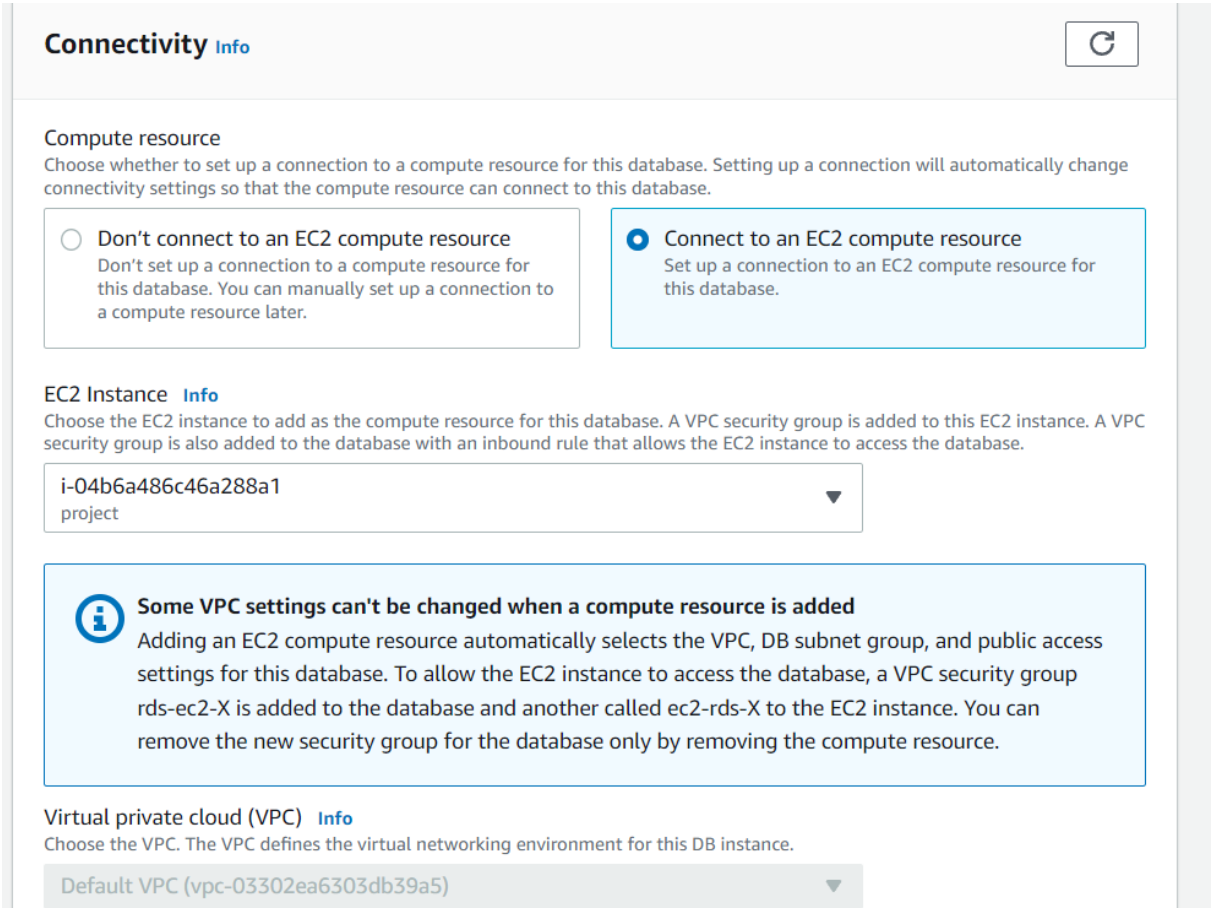
Layout diagram of the revised network



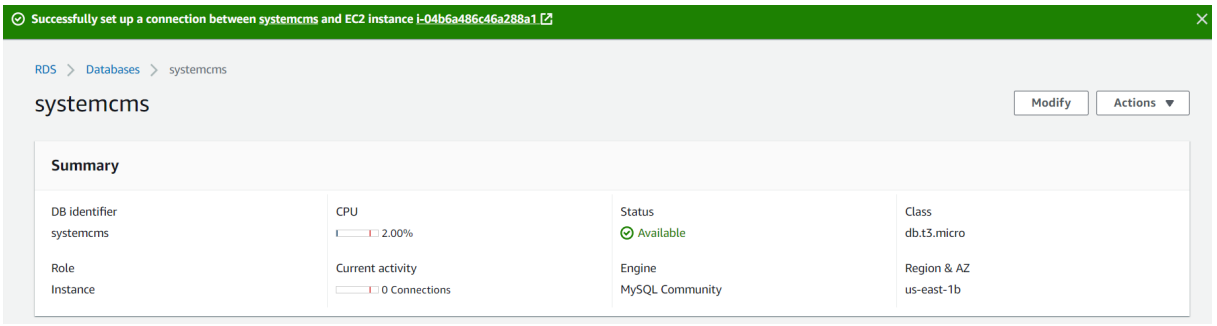
Phase 2 Implementation steps

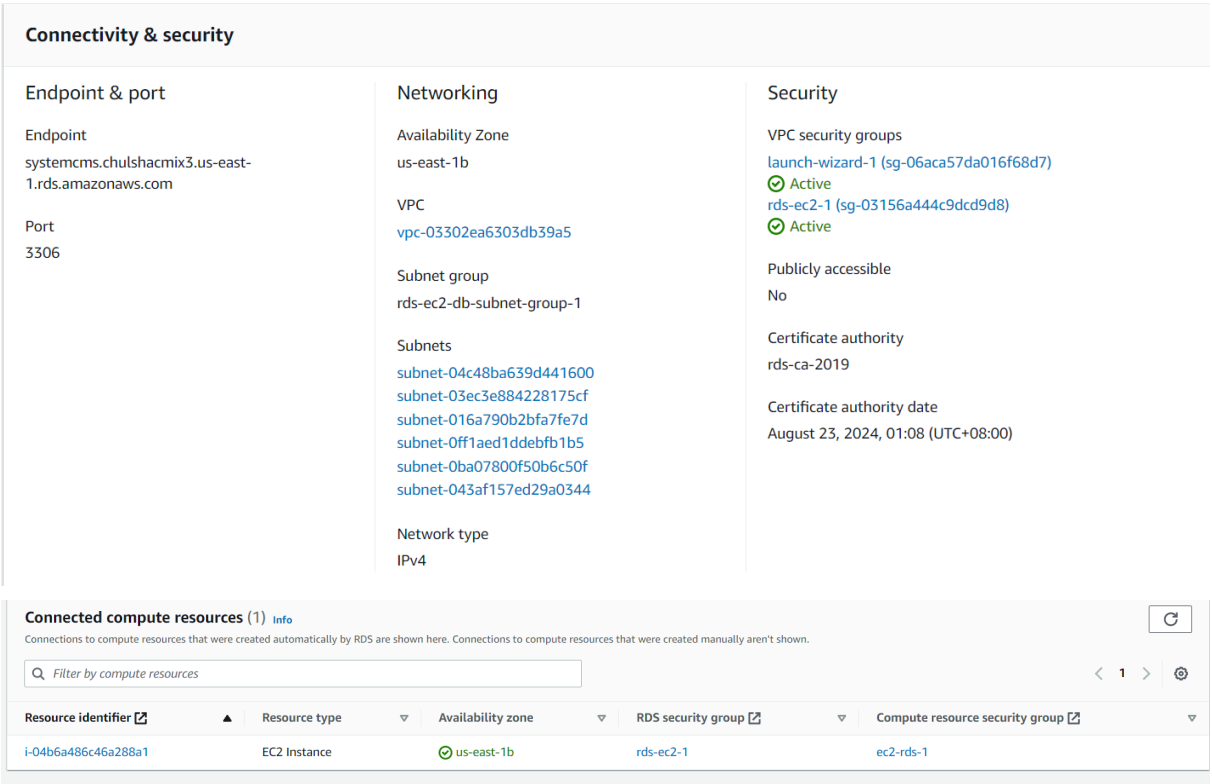
In phase 2, we need to separate the database from the web application EC2 instance we have created in phase 1. The first step is to open a new RDS instance. With the following set up

- Database creation method choose **standard create**
- Engin option choose **MySql**
- Template choose **free tier**
- Name the RDS instances as **system_cms**
- And for the connectivity, choose **Connect to an EC2 compute resource and select the EC2** instance we have create before
- And press the **Create Database**



Below shows the created RDS,





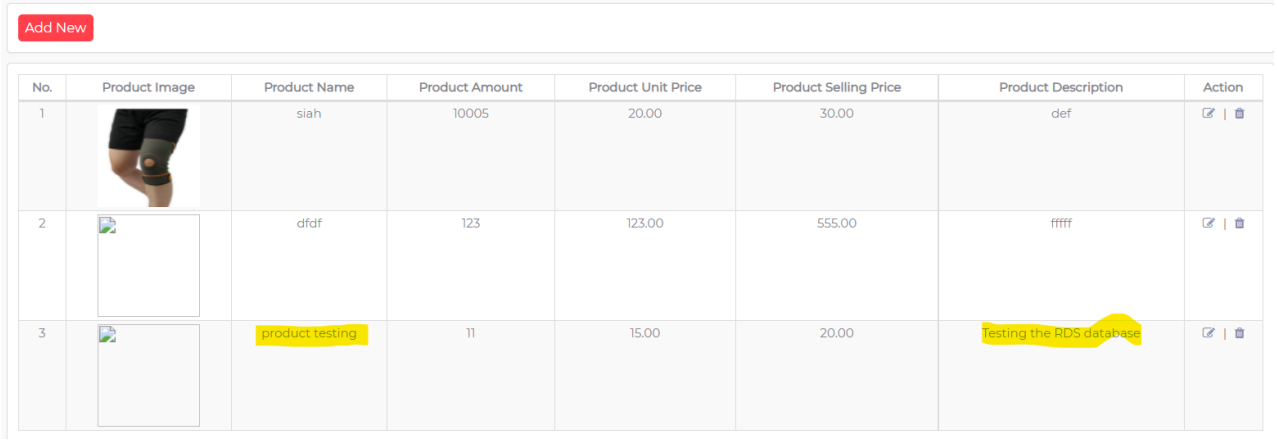
After creating the RDS database, we move to the EC2 CLI to log in to the RDS database and import the MYSQL dump file.

```
[ec2-user@ip-172-31-18-52 ~]$ mysql -h systemcms.chulshacmix3.us-east-1.rds.amazonaws.com -P 3306 -u admin -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MySQL connection id is 17
Server version: 8.0.28 Source distribution

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

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

MySQL [(none)]> CREATE DATABASE system_cms
    -> ;
Query OK, 1 row affected (0.01 sec)
```

```
[ec2-user@ip-172-31-18-52 Connect]$ mysql -h systemcms.chulshacmix3.us-east-1.rds.amazonaws.com -u admin -p system_cms < system_product.sql
Enter password:
[ec2-user@ip-172-31-18-52 Connect]$ mysql -h systemcms.chulshacmix3.us-east-1.rds.amazonaws.com -u admin -p system_cms < system_user.sql
Enter password:
```

And now we have the separate RDS database **(endpoint:systemcms.chulshacmix3.us-east-1.rds.amazonaws.com)** connected with the EC2 instances . For testing purposes, we try to add a new product to the website and compare it with the RDS database.

We have added a testing product to our website and check whether it updates to our RDS database.



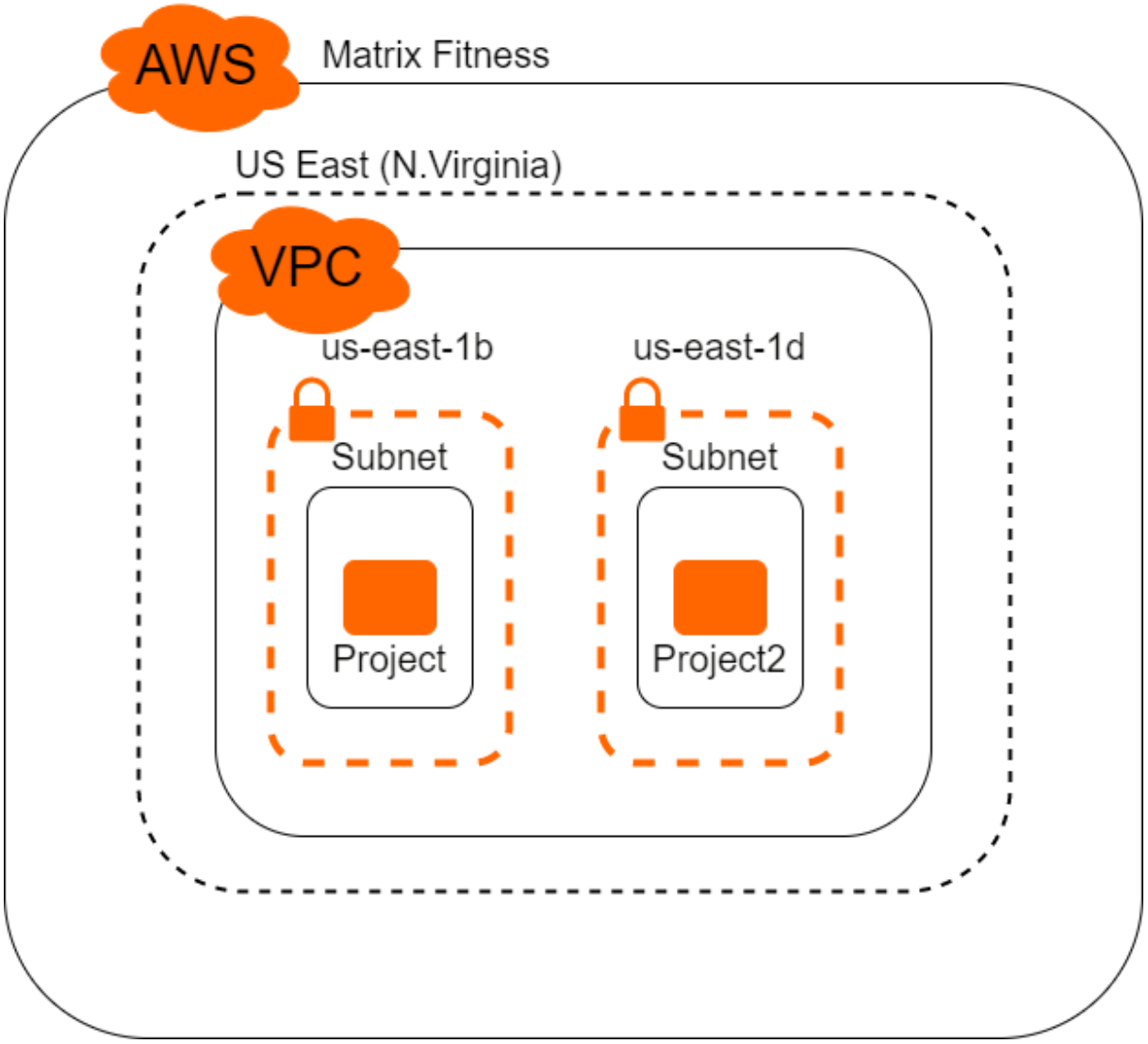

```
[ec2-user@ip-172-31-18-52 Connect]$ mysql -h systemcms.chulshacmix3.us-east-1.rds.amazonaws.com -P 3306 -u admin -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MySQL connection id is 50
Server version: 8.0.28 Source distribution

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

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

MySQL [(none)]> USE system_cms
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
```

```
MySQL [system_cms]> SELECT * FROM system_product;
+------------+------------------+-------------+-------------+--------------------+--------------------+--------------------------+-------------
| product_id | product_name     | product_pic | product_amt | product_unit_price | product_sell_price | product_description      | product_tras
+------------+------------------+-------------+-------------+--------------------+--------------------+--------------------------+-------------
|          1 | siah             | P2.png      |       10005 |              20.00 |              30.00 | def                      | 1
|          2 | siah             | P2.png      |       10005 |              20.00 |              30.00 | def                      | 0
|          3 | dfdf             | NULL        |         123 |             123.00 |             555.00 | fffff                    | 0
|          4 | product testing  | NULL        |          11 |              15.00 |              20.00 | Testing the RDS database | 0
+------------+------------------+-------------+-------------+--------------------+--------------------+--------------------------+-------------
```

With this, we have completed the Phase 2 implementation.

Phase 3

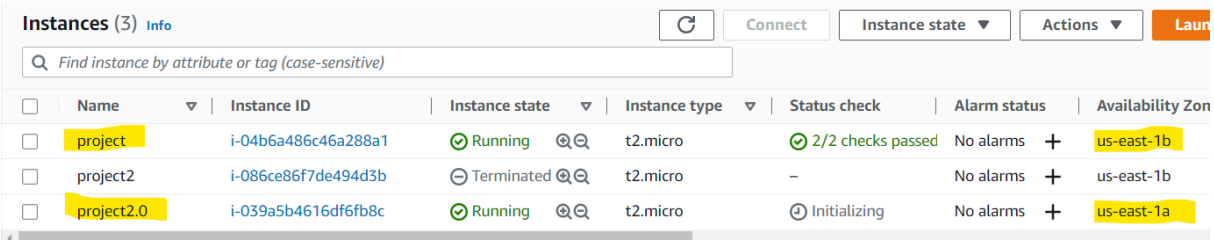
Layout diagram of the revised network



Phase 3 Implementation steps

1) Create a second ec2 instance with different availability zone, and different key pair

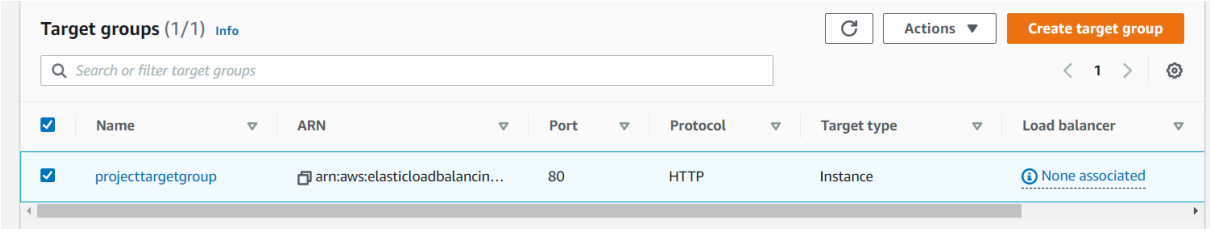
- For a quicker creation, can go to your first EC2 instance, click image and template and select launch more like this and then change the key pair and availability zone
- After that, open up the cli under the connect option
- Perform the same setting you used during first EC2 setup
- Go to /var/www/html folder to unzip the cms.zip
- Test your second EC2 instance



2) Load balancing

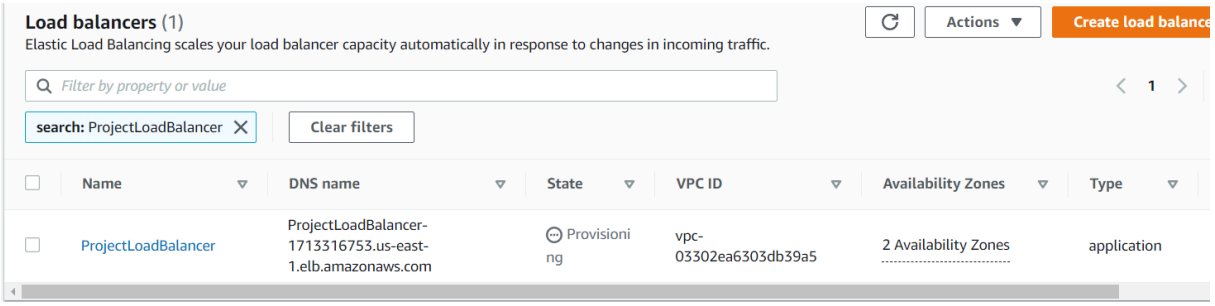
a) Create a target group

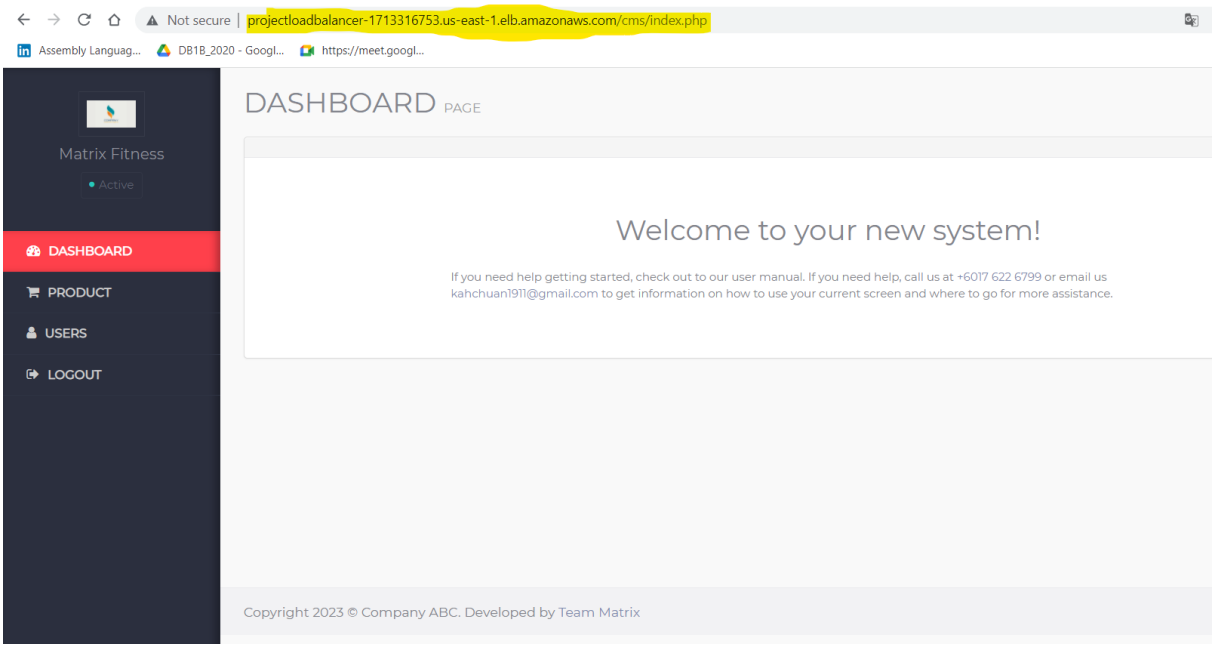
- Go to "Target Groups" at the sidebar of EC2 dashboard
- Click create target groups
- Keep instance target groups selected
- Create a new target group name
- Keep default setting for the rest of section
- Click next
- Select the two EC2 instance that you have created earlier
- Include as pending below



b) Create a load balancer

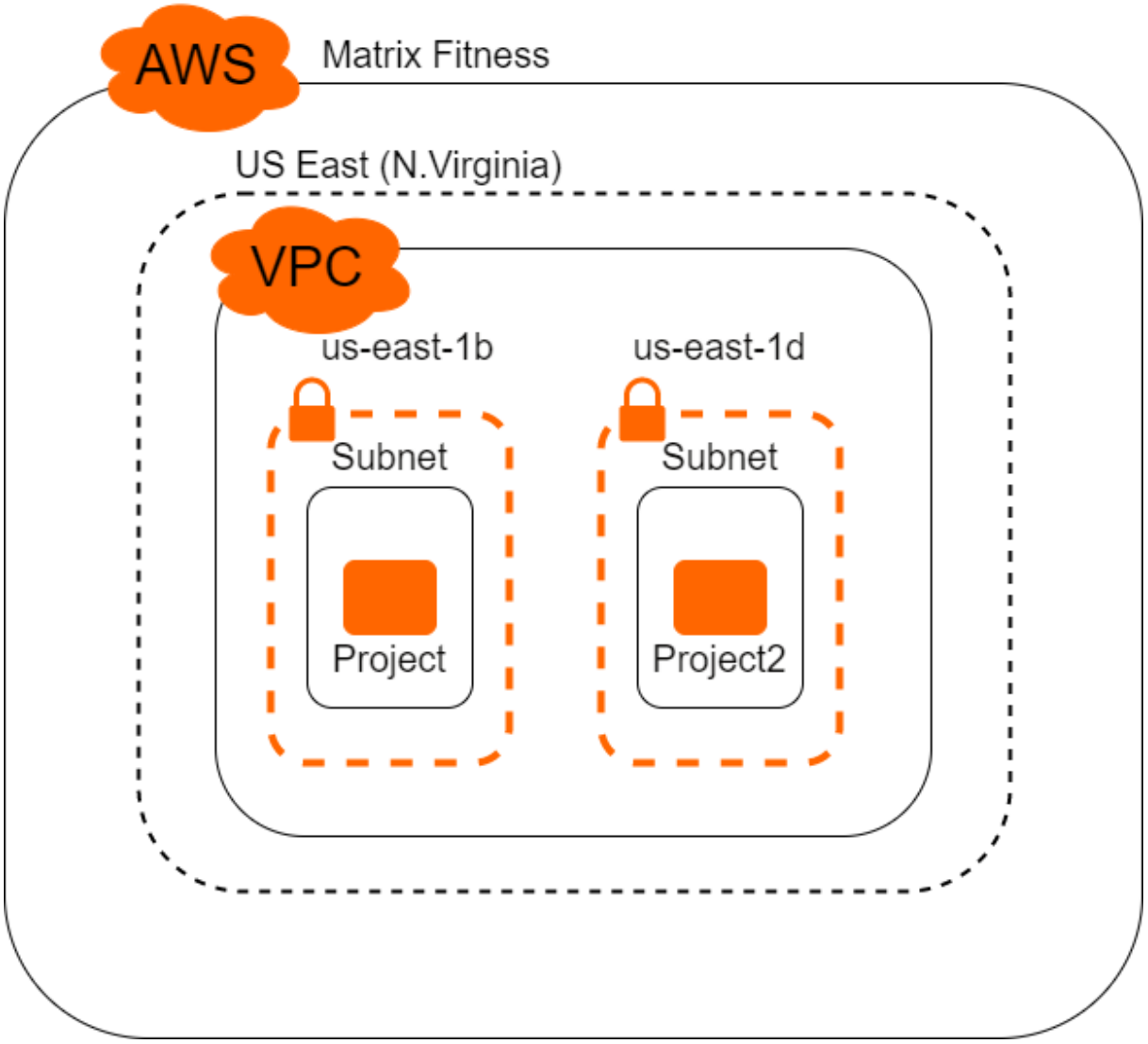
- Go to the sidebar and find load balancer and create load balancer
- Choose application load balancer
- Create a new load balancer name
- Keep default setting for scheme, ip address type and VPC
- Choose the availability zone for your two EC2 instance
- Select the security groups as what you applied for your EC2 instance
- Keep default setting for the rest, select the target group created before and create load balancer
- After creation, We can use the Elastic Load Balancer DNS to access the system.
- The DNS name is shown below:


Load balancers (1)
Elastic Load Balancing scales your load balancer capacity automatically in response to changes in incoming traffic.
Actions
Create load balance
Filter by property or value
search: ProjectLoadBalancer
Clear filters
1
Name
DNS name
State
VPC ID
Availability Zones
Type
ProjectLoadBalancer
ProjectLoadBalancer-1713316753.us-east-1.elb.amazonaws.com
Provisioning
vpc-03302ea6303db39a5
2 Availability Zones
application


Not secure | projectloadbalancer-1713316753.us-east-1.elb.amazonaws.com/cms/index.php
Assembly Languag...
DB1B_2020 - Googl...
https://meet.googl...
Matrix Fitness
Active
DASHBOARD
PRODUCT
USERS
LOGOUT
DASHBOARD PAGE
Welcome to your new system!
If you need help getting started, check out to our user manual. If you need help, call us at +6017 622 6799 or email us kahchuan1911@gmail.com to get information on how to use your current screen and where to go for more assistance.
Copyright 2023 © Company ABC. Developed by Team Matrix

Phase 4

Layout diagram of the revised network



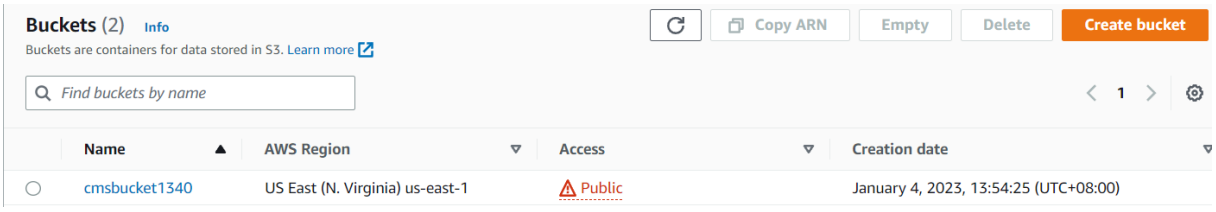
Phase 4 Implementation steps

1) First, we need to create a S3 bucket to store our web template

- Go to S3 bucket and choose create bucket
- Give the bucket a new name
- Keep default settings for ACLs disabled
- Untick the block all public access option
- Keep the default settings for the rest of section
- Press create bucket option
- After the bucket has been created
- Go to the details page and upload file
- Upload the "cms.zip" file into S3 bucket
- After that go to permission tab and scroll down to bucket policy and edit it
- Paste the following command:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "PublicReadGetObject",
            "Effect": "Allow",
            "Principal": "*",
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::files-str/*" <- (change this to your own arn)
        }
    ]
}
```

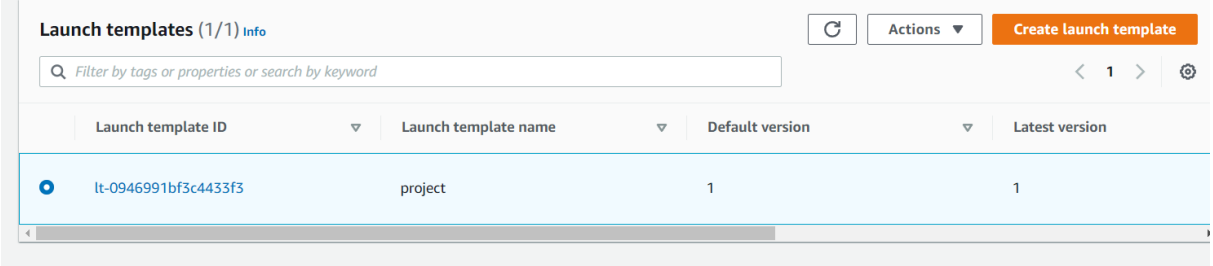
- After that press save changes



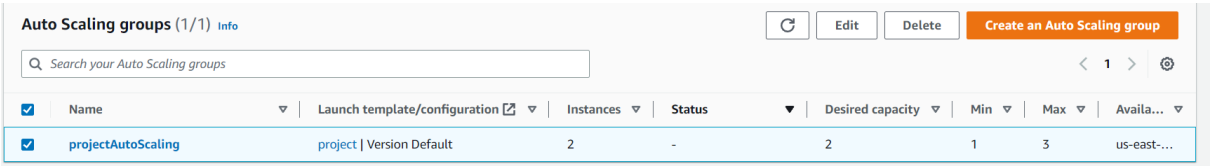
2) After creating the S3 bucket, we need to setup the launch template

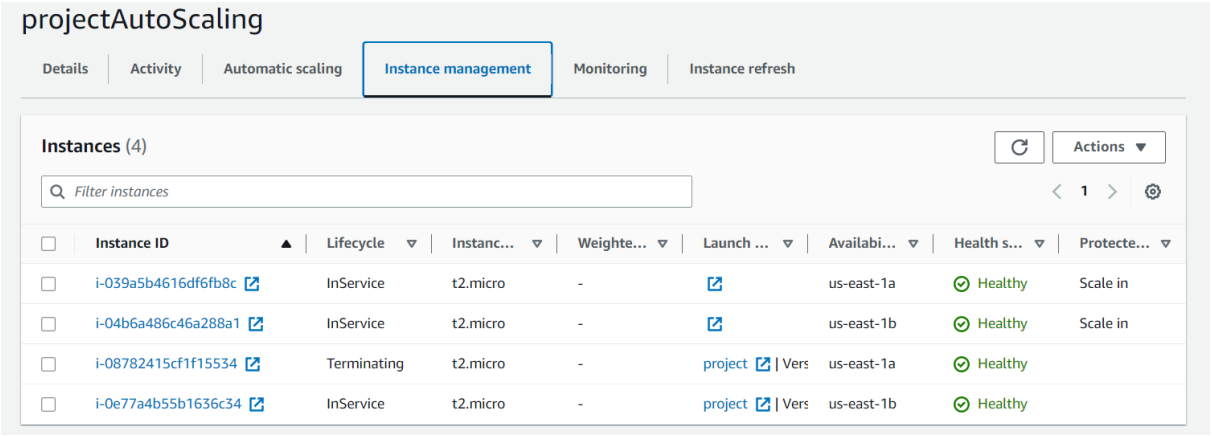
- Goto launch templates
- Choose create launch template
- Create a new launch template name
- Give brief description for template version description (optional)
- Tick on auto scaling guidance
- Choose your favourite application and os images (mine is amazon linux 2 x86)
- Choose t2.micro for instance type

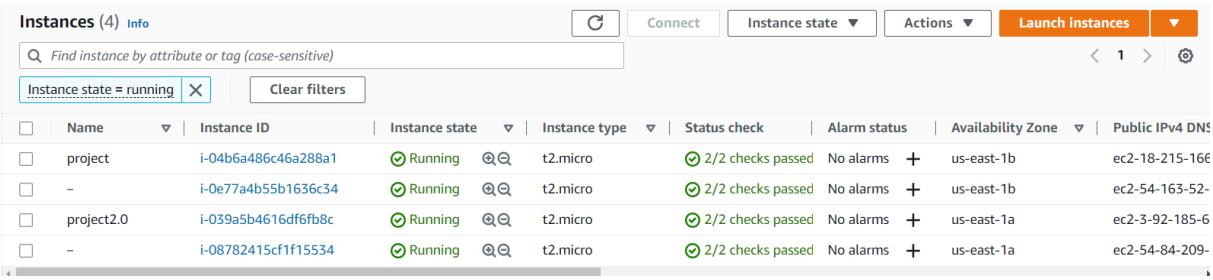
- Choose the security groups for the one you applied for your EC2 instance
- Open up advanced detail, go to user data and fill up the command you use in your EC2 and create the launch template
- The following command:
  #!/bin/bash
  yum update -y
  amazon-linux-extras install php8.0 mariadb10.5
  yum -y install httpd
  systemctl enable httpd
  systemctl start httpd
  usermod -a -G apache ec2-user
  chown -R ec2-user:apache /var/www
  chmod 2775 /var/www
  find /var/www -type d -exec sudo chmod 2775 {} \;
  find /var/www -type f -exec sudo chmod 0664 {} \;
  systemctl restart httpd
  wget https://cmsbucket1340.s3.amazonaws.com/cms.zip -P /var/www/html
  unzip /var/www/html/cms.zip -d /var/www/html/
  echo "<?php header('Location:cms/index.php');?>" > /var/www/html/index.php



3) Lastly, we need to setup the auto scaling groups
- Go to auto scaling groups
- Choose create auto scaling groups
- Create a new auto scaling groups name
- Select the template we created earlier
- Choose VPC and the two availability zone for the two EC2 we created earlier
- Choose attach to an existing load balancer which you created earlier
- Keep default setting for the rest of the page
- Keep desired capacity as 2, minimum capacity as 1, maximum capacity as 4
- Choose target tracking scaling policy select average cpu utilization for give 80 (or any number you like) for target value and create the auto scaling group
- After that go to the EC2 instance list and click on the EC2 instance respectively and select action, go to instance setting and click attach to auto scaling groups
- Later on, go to the list of the auto scaling groups and select the auto scaling groups created just now
- Go to the instance management and select the two EC2 instances that you created and click on action to set scale in protection to avoid the EC2 instances getting deleted.


Auto Scaling groups (1/1) Info
Edit
Delete
Create an Auto Scaling group
Search your Auto Scaling groups
1
Name
Launch template/configuration
Instances
Status
Desired capacity
Min
Max
Availa...
projectAutoScaling
project | Version Default
2
-
2
1
3
us-east-...


projectAutoScaling
Details
Activity
Automatic scaling
Instance management
Monitoring
Instance refresh
Instances (4)
Actions
Filter instances
1
Instance ID
Lifecycle
Instanc...
Weighte...
Launch ...
Availabi...
Health s...
Protecte...
i-039a5b4616df6fb8c
InService
t2.micro
-
us-east-1a
Healthy
Scale in
i-04b6a486c46a288a1
InService
t2.micro
-
us-east-1b
Healthy
Scale in
i-08782415cf1f15534
Terminating
t2.micro
-
project | Vers
us-east-1a
Healthy
i-0e77a4b55b1636c34
InService
t2.micro
-
project | Vers
us-east-1b
Healthy


Instances (4) Info
Connect
Instance state
Actions
Launch instances
Find instance by attribute or tag (case-sensitive)
1
Instance state = running
Clear filters
Name
Instance ID
Instance state
Instance type
Status check
Alarm status
Availability Zone
Public IPv4 DNS
project
i-04b6a486c46a288a1
Running
t2.micro
2/2 checks passed
No alarms
us-east-1b
ec2-18-215-166
-
i-0e77a4b55b1636c34
Running
t2.micro
2/2 checks passed
No alarms
us-east-1b
ec2-54-163-52-
project2.0
i-039a5b4616df6fb8c
Running
t2.micro
2/2 checks passed
No alarms
us-east-1a
ec2-3-92-185-6
-
i-08782415cf1f15534
Running
t2.micro
2/2 checks passed
No alarms
us-east-1a
ec2-54-84-209-