Task 1:

- Create a VPC with a public subnet
- Create a new phpmyadmin EC2
- Create a Web proxy for caching

What helped me build this
https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/AWS_EC2.html
https://docs.aws.amazon.com/codebuild/latest/userguide/cloudformation-vpc-template.html

Description: This template deploys a VPC, with a pair of public and private subnets spread across two Availability Zones. It deploys an internet gateway, with a default route on the public subnets. It deploys a pair of NAT gateways (one in each AZ), and default routes for them in the private subnets.

Parameters:
EnvironmentName:
Description: An environment name that is prefixed to resource names
Type: String

VpcCIDR:
Description: Please enter the IP range (CIDR notation) for this VPC
Type: String
Default: 192.168.0.0/16

PublicSubnet1CIDR:
Description: Please enter the IP range (CIDR notation) for the public subnet in the first Availability Zone
Type: String
Default: 192.168.0.0/18

PublicSubnet2CIDR:
Description: Please enter the IP range (CIDR notation) for the public subnet in the second Availability Zone
Type: String
Default: 192.168.64.0/18

PrivateSubnet1CIDR:
Description: Please enter the IP range (CIDR notation) for the private subnet in the first Availability Zone
Type: String

```
    Default: 192.168.128.0/18

  PrivateSubnet2CIDR:
    Description: Please enter the IP range (CIDR notation) for the private subnet in the second
Availability Zone
    Type: String
    Default: 192.168.192.0/18

Resources:
  VPC:
    Type: AWS::EC2::VPC
    Properties:
      CidrBlock: !Ref VpcCIDR
      EnableDnsSupport: true
      EnableDnsHostnames: true
      Tags:
        - Key: Name
          Value: !Ref EnvironmentName

  InternetGateway:
    Type: AWS::EC2::InternetGateway
    Properties:
      Tags:
        - Key: Name
          Value: !Ref EnvironmentName

  InternetGatewayAttachment:
    Type: AWS::EC2::VPCGatewayAttachment
    Properties:
      InternetGatewayId: !Ref InternetGateway
      VpcId: !Ref VPC

  PublicSubnet1:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId: !Ref VPC
      AvailabilityZone: !Select [ 0, !GetAZs '' ]
      CidrBlock: !Ref PublicSubnet1CIDR
      MapPublicIpOnLaunch: true
      Tags:
        - Key: Name
          Value: !Sub ${EnvironmentName} Public Subnet (AZ1)

  PublicSubnet2:
```

```
  Type: AWS::EC2::Subnet
  Properties:
    VpcId: !Ref VPC
    AvailabilityZone: !Select [ 1, !GetAZs  '' ]
    CidrBlock: !Ref PublicSubnet2CIDR
    MapPublicIpOnLaunch: true
    Tags:
      - Key: Name
        Value: !Sub ${EnvironmentName} Public Subnet (AZ2)

PrivateSubnet1:
  Type: AWS::EC2::Subnet
  Properties:
    VpcId: !Ref VPC
    AvailabilityZone: !Select [ 0, !GetAZs  '' ]
    CidrBlock: !Ref PrivateSubnet1CIDR
    MapPublicIpOnLaunch: false
    Tags:
      - Key: Name
        Value: !Sub ${EnvironmentName} Private Subnet (AZ1)

PrivateSubnet2:
  Type: AWS::EC2::Subnet
  Properties:
    VpcId: !Ref VPC
    AvailabilityZone: !Select [ 1, !GetAZs  '' ]
    CidrBlock: !Ref PrivateSubnet2CIDR
    MapPublicIpOnLaunch: false
    Tags:
      - Key: Name
        Value: !Sub ${EnvironmentName} Private Subnet (AZ2)

NatGateway1EIP:
  Type: AWS::EC2::EIP
  DependsOn: InternetGatewayAttachment
  Properties:
    Domain: vpc

NatGateway2EIP:
  Type: AWS::EC2::EIP
  DependsOn: InternetGatewayAttachment
  Properties:
    Domain: vpc
```

```
NatGateway1:
  Type: AWS::EC2::NatGateway
  Properties:
    AllocationId: !GetAtt NatGateway1EIP.AllocationId
    SubnetId: !Ref PublicSubnet1

NatGateway2:
  Type: AWS::EC2::NatGateway
  Properties:
    AllocationId: !GetAtt NatGateway2EIP.AllocationId
    SubnetId: !Ref PublicSubnet2

PublicRouteTable:
  Type: AWS::EC2::RouteTable
  Properties:
    VpcId: !Ref VPC
    Tags:
      - Key: Name
        Value: !Sub ${EnvironmentName} Public Routes

DefaultPublicRoute:
  Type: AWS::EC2::Route
  DependsOn: InternetGatewayAttachment
  Properties:
    RouteTableId: !Ref PublicRouteTable
    DestinationCidrBlock: 0.0.0.0/0
    GatewayId: !Ref InternetGateway

PublicSubnet1RouteTableAssociation:
  Type: AWS::EC2::SubnetRouteTableAssociation
  Properties:
    RouteTableId: !Ref PublicRouteTable
    SubnetId: !Ref PublicSubnet1

PublicSubnet2RouteTableAssociation:
  Type: AWS::EC2::SubnetRouteTableAssociation
  Properties:
    RouteTableId: !Ref PublicRouteTable
    SubnetId: !Ref PublicSubnet2


PrivateRouteTable1:
  Type: AWS::EC2::RouteTable
  Properties:
```

```
    VpcId: !Ref VPC
    Tags:
      - Key: Name
        Value: !Sub ${EnvironmentName} Private Routes (AZ1)

  DefaultPrivateRoute1:
    Type: AWS::EC2::Route
    Properties:
      RouteTableId: !Ref PrivateRouteTable1
      DestinationCidrBlock: 0.0.0.0/0
      NatGatewayId: !Ref NatGateway1

  PrivateSubnet1RouteTableAssociation:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      RouteTableId: !Ref PrivateRouteTable1
      SubnetId: !Ref PrivateSubnet1

  PrivateRouteTable2:
    Type: AWS::EC2::RouteTable
    Properties:
      VpcId: !Ref VPC
      Tags:
        - Key: Name
          Value: !Sub ${EnvironmentName} Private Routes (AZ2)

  DefaultPrivateRoute2:
    Type: AWS::EC2::Route
    Properties:
      RouteTableId: !Ref PrivateRouteTable2
      DestinationCidrBlock: 0.0.0.0/0
      NatGatewayId: !Ref NatGateway2

  PrivateSubnet2RouteTableAssociation:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      RouteTableId: !Ref PrivateRouteTable2
      SubnetId: !Ref PrivateSubnet2

  NoIngressSecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupName: "no-ingress-sg"
      GroupDescription: "Security group with no ingress rule"
```

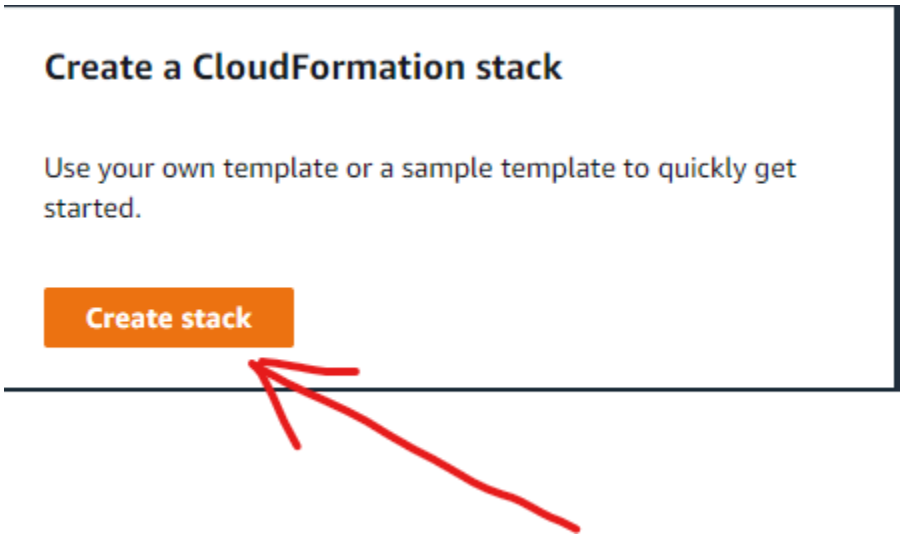
```
    VpcId: !Ref VPC

Ec2instance1:
  Type: AWS::EC2::Instance
  Properties:
    ImageId: ami-0629230e074c580f2
    InstanceType: t2.micro
    SubnetId: !Ref PublicSubnet1
    KeyName: "Python"
    SecurityGroupIds:
      - !Ref BastionSecurityGroup
    Tags:
      - Key: "Name"
        Value: "Bastion-Host"

BastionSecurityGroup:
  Type: AWS::EC2::SecurityGroup
  Properties:
    GroupDescription: "Security group that allows SSH from anywhere"
    GroupName: "Bastion"
    SecurityGroupIngress:
      - IpProtocol: tcp
        FromPort: 22
        ToPort: 22
        CidrIp: 0.0.0.0/0
    VpcId: !Ref VPC

Ec2instance2:
  Type: AWS::EC2::Instance
  Properties:
    ImageId: ami-0629230e074c580f2
    InstanceType: t2.micro
    SubnetId: !Ref PublicSubnet1
    KeyName: "Python"
    SecurityGroupIds:
      - !Ref BastionSecurityGroup
    Tags:
      - Key: "Name"
        Value: "Nginx-Host"

NginxSecurityGroup:
  Type: AWS::EC2::SecurityGroup
  Properties:
```

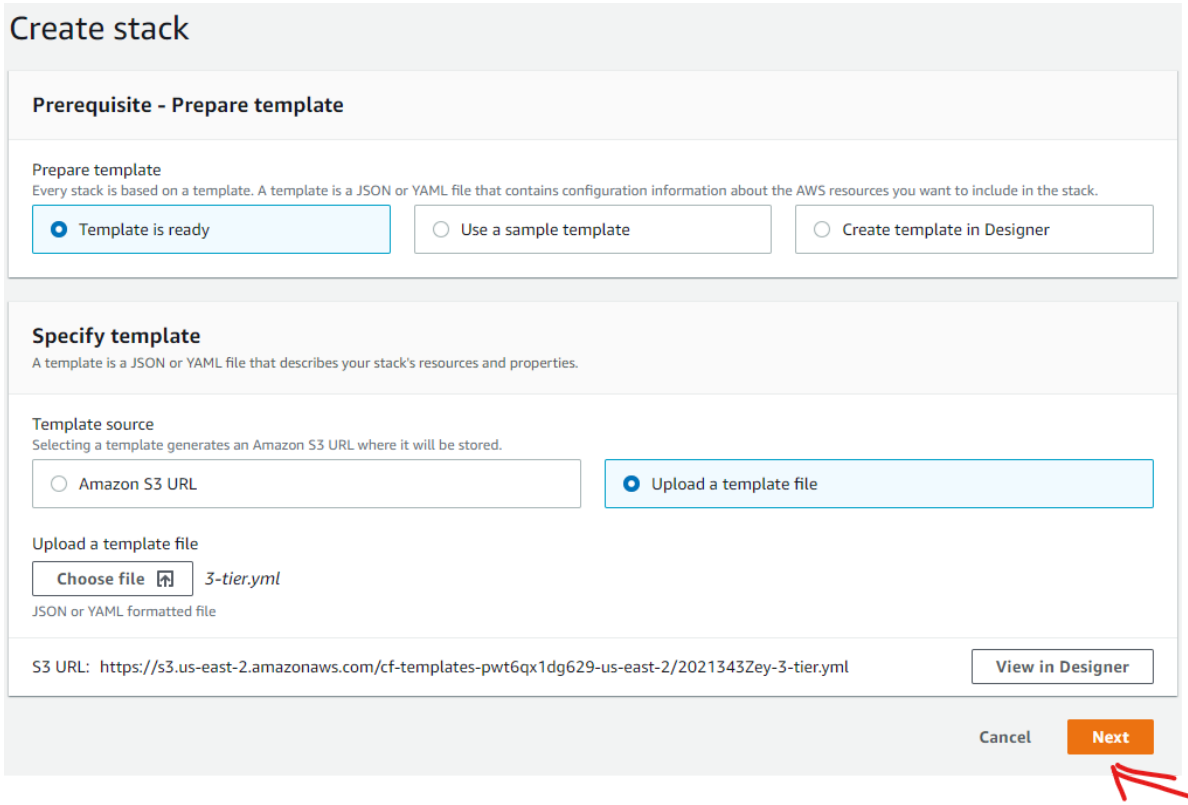
```
      GroupDescription: "Security group that allows SSH from bastion host only. It also allows
client access on HTTP/HTTPS"
      GroupName: "Nginx"
      SecurityGroupIngress:
        - IpProtocol: tcp
          FromPort: 22
          ToPort: 22
          SourceSecurityGroupId:
            Fn::GetAtt:
            - BastionSecurityGroup
            - GroupId
        - IpProtocol: tcp
          FromPort: 80
          ToPort: 80
          CidrIp: 0.0.0.0/0
        - IpProtocol: tcp
          FromPort: 443
          ToPort: 443
          CidrIp: 0.0.0.0/0
      VpcId: !Ref VPC


  Ec2instance3:
    Type: AWS::EC2::Instance
    Properties:
      ImageId: ami-0629230e074c580f2
      InstanceType: t2.micro
      SubnetId: !Ref PublicSubnet1
      KeyName: "Python"
      SecurityGroupIds:
        - !Ref BastionSecurityGroup
      Tags:
        - Key: "Name"
          Value: "PhpMYAdmin"

  PhpMyAdminSecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupDescription: "Security group that only allows SSH from the 1st instance (bastion host)
"
      GroupName: "phpMyAdmin"
      SecurityGroupIngress:
        - IpProtocol: tcp
          FromPort: 22
```

```
        ToPort: 22
        SourceSecurityGroupId:
          Fn::GetAtt:
          - BastionSecurityGroup
          - GroupId
      - IpProtocol: tcp
        FromPort: 80
        ToPort: 80
        SourceSecurityGroupId:
          Fn::GetAtt:
          - NginxSecurityGroup
          - GroupId
    VpcId: !Ref VPC


Outputs:
  VPC:
    Description: A reference to the created VPC
    Value: !Ref VPC

  PublicSubnets:
    Description: A list of the public subnets
    Value: !Join [ ",", [ !Ref PublicSubnet1, !Ref PublicSubnet2 ]]

  PrivateSubnets:
    Description: A list of the private subnets
    Value: !Join [ ",", [ !Ref PrivateSubnet1, !Ref PrivateSubnet2 ]]

  PublicSubnet1:
    Description: A reference to the public subnet in the 1st Availability Zone
    Value: !Ref PublicSubnet1

  PublicSubnet2:
    Description: A reference to the public subnet in the 2nd Availability Zone
    Value: !Ref PublicSubnet2

  PrivateSubnet1:
    Description: A reference to the private subnet in the 1st Availability Zone
    Value: !Ref PrivateSubnet1

  PrivateSubnet2:
    Description: A reference to the private subnet in the 2nd Availability Zone
    Value: !Ref PrivateSubnet2
```

NoIngressSecurityGroup:
  Description: Security group with no ingress rule
  Value: !Ref NoIngressSecurityGroup

CreATE THE STACK IN CLOUD FORMATION





Stack failure options

Behavior on provisioning failure

Specify the roll back behavior for a stack failure. **Learn more**

- (•) Roll back all stack resources
  Roll back the stack to the last known stable state.
- ( ) Preserve successfully provisioned resources
  Preserves the state of successfully provisioned resources, while rolling back failed resources to the last known stable state. Resources without a last known stable state will be deleted upon the next stack operation.

Advanced options

You can set additional options for your stack, like notification options and a stack policy. Learn more

▸ Stack policy

Defines the resources that you want to protect from unintentional updates during a stack update.

▸ Rollback configuration

Specify alarms for CloudFormation to monitor when creating and updating the stack. If the operation breaches an alarm threshold, CloudFormation rolls it back. **Learn more**

▸ Notification options

▸ Stack creation options

Cancel | Previous | **Next**

Specify stack details

Stack name

Stack name

3t

Stack name can include letters (A-Z and a-z), numbers (0-9), and dashes (-).

Parameters

Parameters are defined in your template and allow you to input custom values when you create or update a stack.

EnvironmentName
An environment name that is prefixed to resource names

env-3t

PrivateSubnet1CIDR
Please enter the IP range (CIDR notation) for the private subnet in the first Availability Zone

192.168.128.0/18

PrivateSubnet2CIDR
Please enter the IP range (CIDR notation) for the private subnet in the second Availability Zone

192.168.192.0/18

PublicSubnet1CIDR
Please enter the IP range (CIDR notation) for the public subnet in the first Availability Zone

192.168.0.0/18

PublicSubnet2CIDR
Please enter the IP range (CIDR notation) for the public subnet in the second Availability Zone

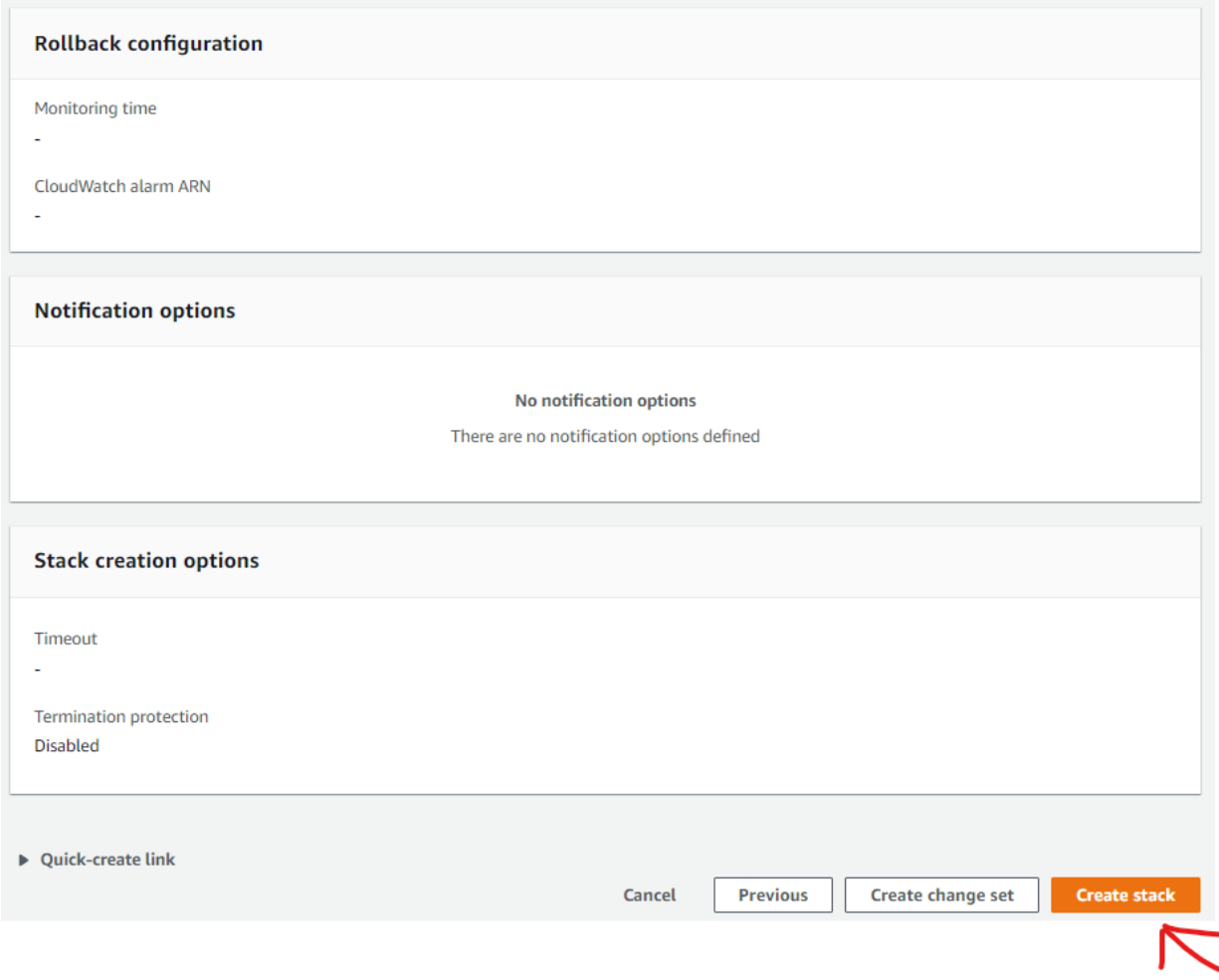
192.168.64.0/18

VpcCIDR
Please enter the IP range (CIDR notation) for this VPC

192.168.0.0/16

Cancel Previous Next

Review→create stack



3.21.43.17
18.222.110.36
192.168.180.20

ssh -i "Python.pem" ubuntu@3.21.43.17
Sudo nano linux.pem —>copy paste you key inside linux pem
sudo chmod 400 linux.pem
sudo ssh into Nginx and php using the new key
sudo ssh -i "linux.pem" ubuntu@192.168.162.62
sudo ssh -i "linux.pem" ubuntu@192.168.180.20

Task 2:

Create an AWS Application Load Balancer to connect to your reverse proxy.

Target Groups New

Basic configuration

Settings in this section cannot be changed after the target group is created.

Choose a target type

- (•) Instances
  - Supports load balancing to instances within a specific VPC.
- ( ) IP addresses
  - Supports load balancing to VPC and on-premises resources.
  - Facilitates routing to multiple IP addresses and network interfaces on the same instance.
  - Offers flexibility with microservice based architectures, simplifying inter-application communication.
- ( ) Lambda function
  - Facilitates routing to a single Lambda function.
  - Accessible to Application Load Balancers only.
- ( ) Application Load Balancer
  - Offers the flexibility for a Network Load Balancer to accept and route TCP requests within a specific VPC.
  - Facilitates using static IP addresses and PrivateLink with an Application Load Balancer.

Target group name

A maximum of 32 alphanumeric characters including hyphens are allowed, but the name must not begin or end with a hyphen.

Protocol : Port

HTTP : 80

VPC

Select the VPC with the instances that you want to include in the target group.

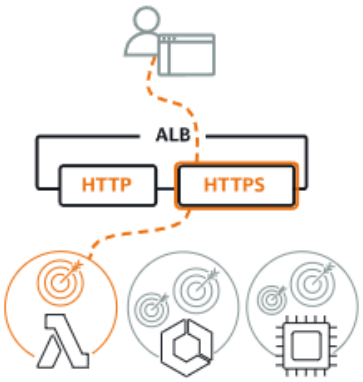
ENV-3tier
vpc-0656ae7cad5009785
IPv4: 192.168.0.0/16

Protocol version

- (•) HTTP1
  Send requests to targets using HTTP/1.1. Supported when the request protocol is HTTP/1.1 or HTTP/2.
- ( ) HTTP2
  Send requests to targets using HTTP/2. Supported when the request protocol is HTTP/2 or gRPC, but gRPC-specific features are not available.
- ( ) gRPC
  Send requests to targets using gRPC. Supported when the request protocol is gRPC.

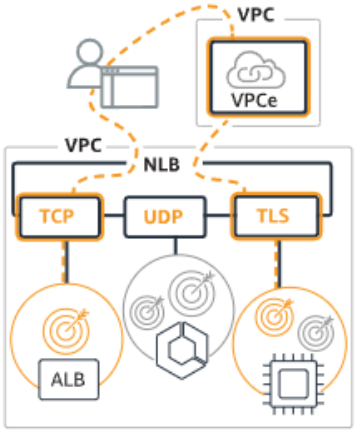
Load balancer types

Application Load Balancer Info



Choose an Application Load Balancer when you need a flexible feature set for your applications with HTTP and HTTPS traffic. Operating at the request level, Application Load Balancers provide advanced routing and visibility features targeted at application architectures, including microservices and containers.

Network Load Balancer Info



Choose a Network Load Balancer when you need ultra-high performance, TLS offloading at scale, centralized certificate deployment, support for UDP, and static IP addresses for your applications. Operating at the connection level, Network Load Balancers are capable of handling millions of requests per second securely while maintaining ultra-low

Gateway Load Balancer Info

Choose a Gateway Load Balancer when you need to deploy and manage a fleet of third-party virtual appliances that support GENEVE. These appliances enable you to improve security, compliance, and policy controls.

Create

Basic configuration

Load balancer name

Name must be unique within your AWS account and cannot be changed after the load balancer is created.

Threetier

A maximum of 32 alphanumeric characters including hyphens are allowed, but the name must not begin or end with a hyphen.

Scheme Info

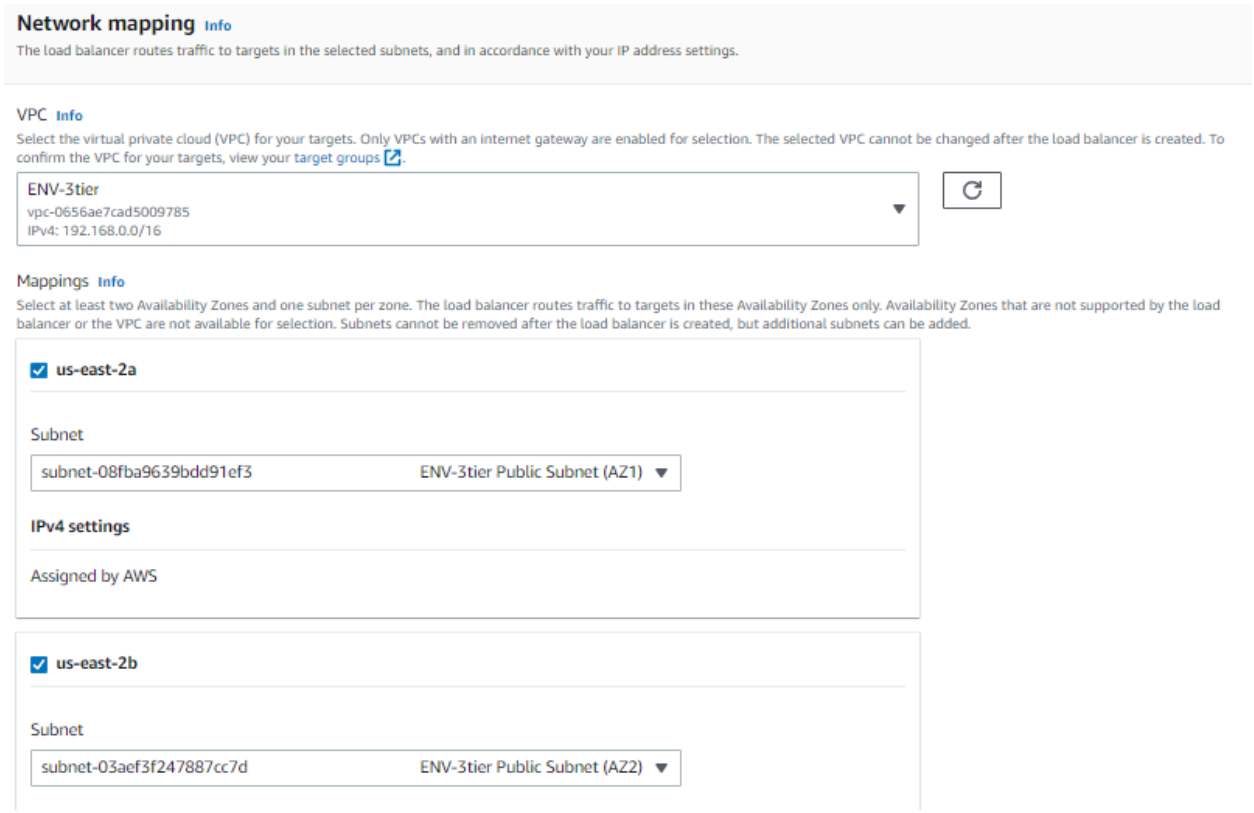
Scheme cannot be changed after the load balancer is created.

- (•) Internet-facing
  An internet-facing load balancer routes requests from clients over the internet to targets. Requires a public subnet. Learn more
- ( ) Internal
  An internal load balancer routes requests from clients to targets using private IP addresses.

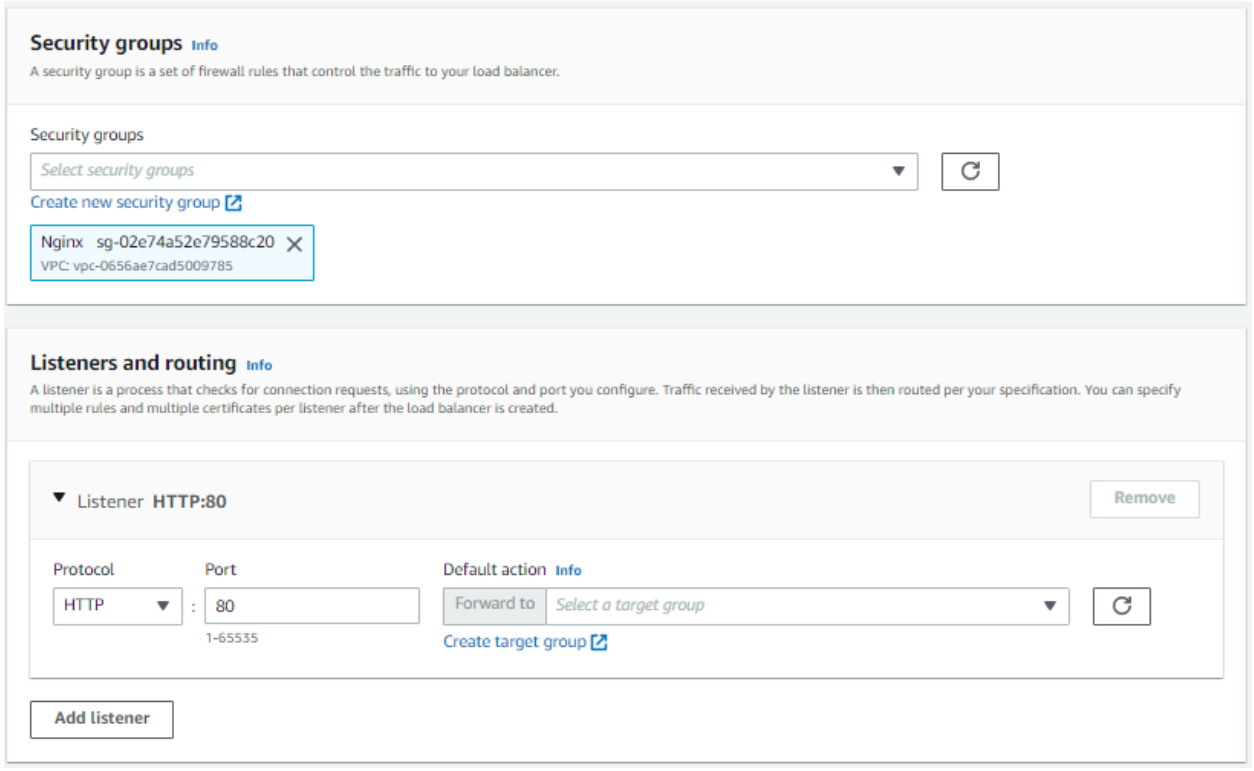
IP address type Info

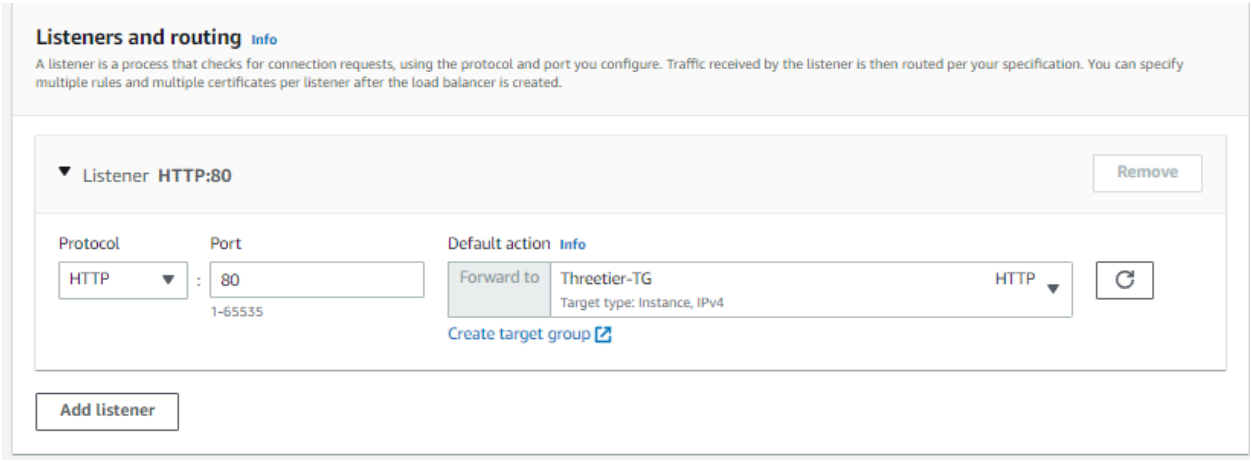
Select the type of IP addresses that your subnets use.

- (•) IPv4
  Recommended for internal load balancers.
- ( ) Dualstack
  Includes IPv4 and IPv6 addresses.



You just need a security group that take HTTP and HTTPS. So you could use Nginx` NGIUNX

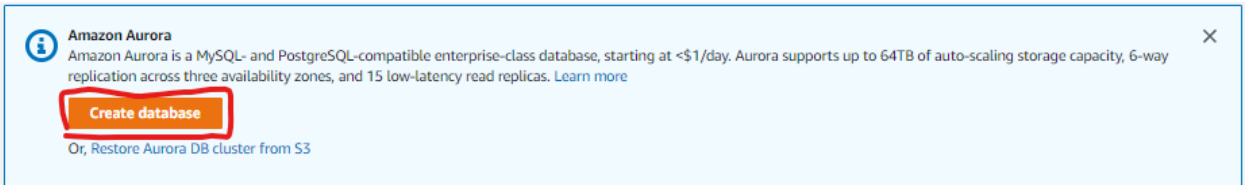




3.21.43.17
18.222.110.36
192.168.180.20

Task 3:

- Create a MySQL database on AWS by first creating a subnet group



Choose a database creation method Info

- (●) **Standard create**
  You set all of the configuration options, including ones for availability, security, backups, and maintenance.
- ( ) **Easy create**
  Use recommended best-practice configurations. Some configuration options can be changed after the database is created.

Engine options

Engine type Info

- ( ) Amazon Aurora

- (●) MySQL

- ( ) MariaDB

- ( ) PostgreSQL

- ( ) Oracle

- ( ) Microsoft SQL Server

Templates

Choose a sample template to meet your use case.

- ( ) **Production**
  Use defaults for high availability and fast, consistent performance.
- ( ) **Dev/Test**
  This instance is intended for development use outside of a production environment.
- (●) **Free tier**
  Use RDS Free Tier to develop new applications, test existing applications, or gain hands-on experience with Amazon RDS. **Info**

Settings

DB instance identifier **Info**

Type a name for your DB instance. The name must be unique across all DB instances owned by your AWS account in the current AWS Region.

three-tier-db

The DB instance identifier is case-insensitive, but is stored as all lowercase (as in "mydbinstance"). Constraints: 1 to 60 alphanumeric characters or hyphens. First character must be a letter. Can't contain two consecutive hyphens. Can't end with a hyphen.

▼ **Credentials Settings**

Master username **Info**

Type a login ID for the master user of your DB instance.

admin

1 to 16 alphanumeric characters. First character must be a letter.

☐ Auto generate a password

Amazon RDS can generate a password for you, or you can specify your own password.

Master password **Info**

••••••••••••

Constraints: At least 8 printable ASCII characters. Can't contain any of the following: / (slash), '(single quote), "(double quote) and @ (at sign).

Confirm password **Info**

••••••••••••

Keep default

DB instance class

DB instance class **Info**

○ Standard classes (includes m classes)

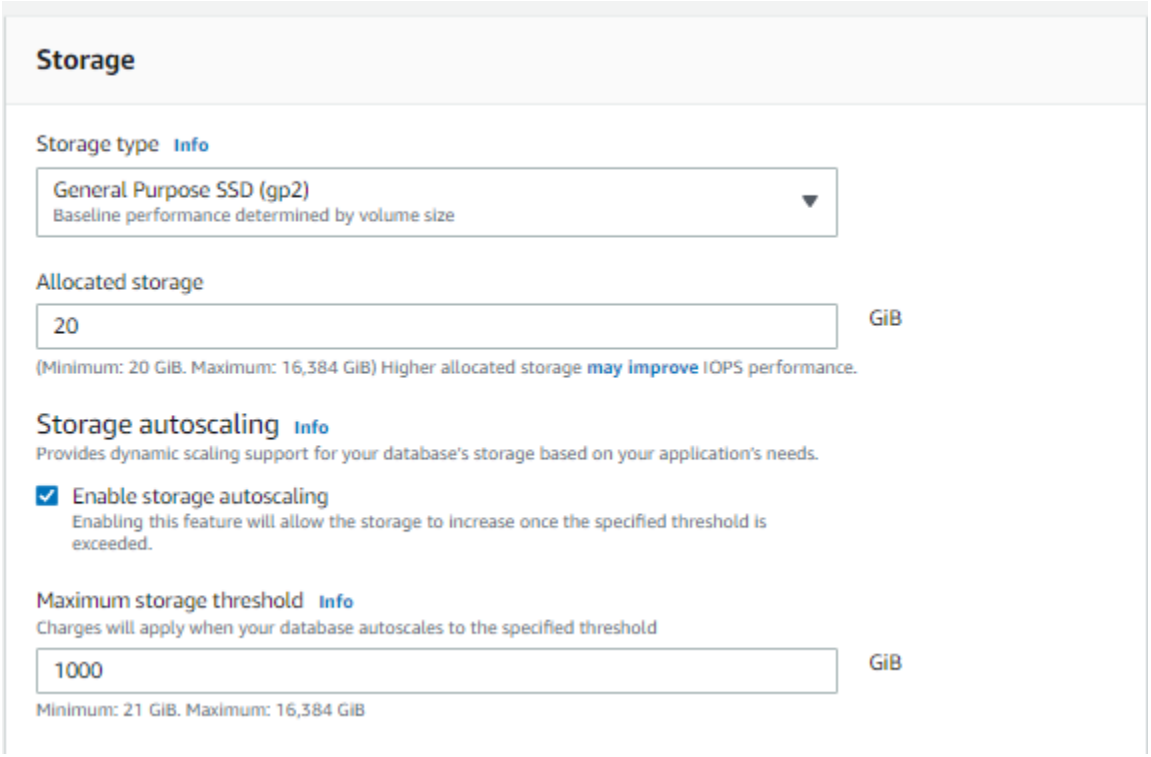
○ Memory optimized classes (includes r and x classes)

◉ Burstable classes (includes t classes)

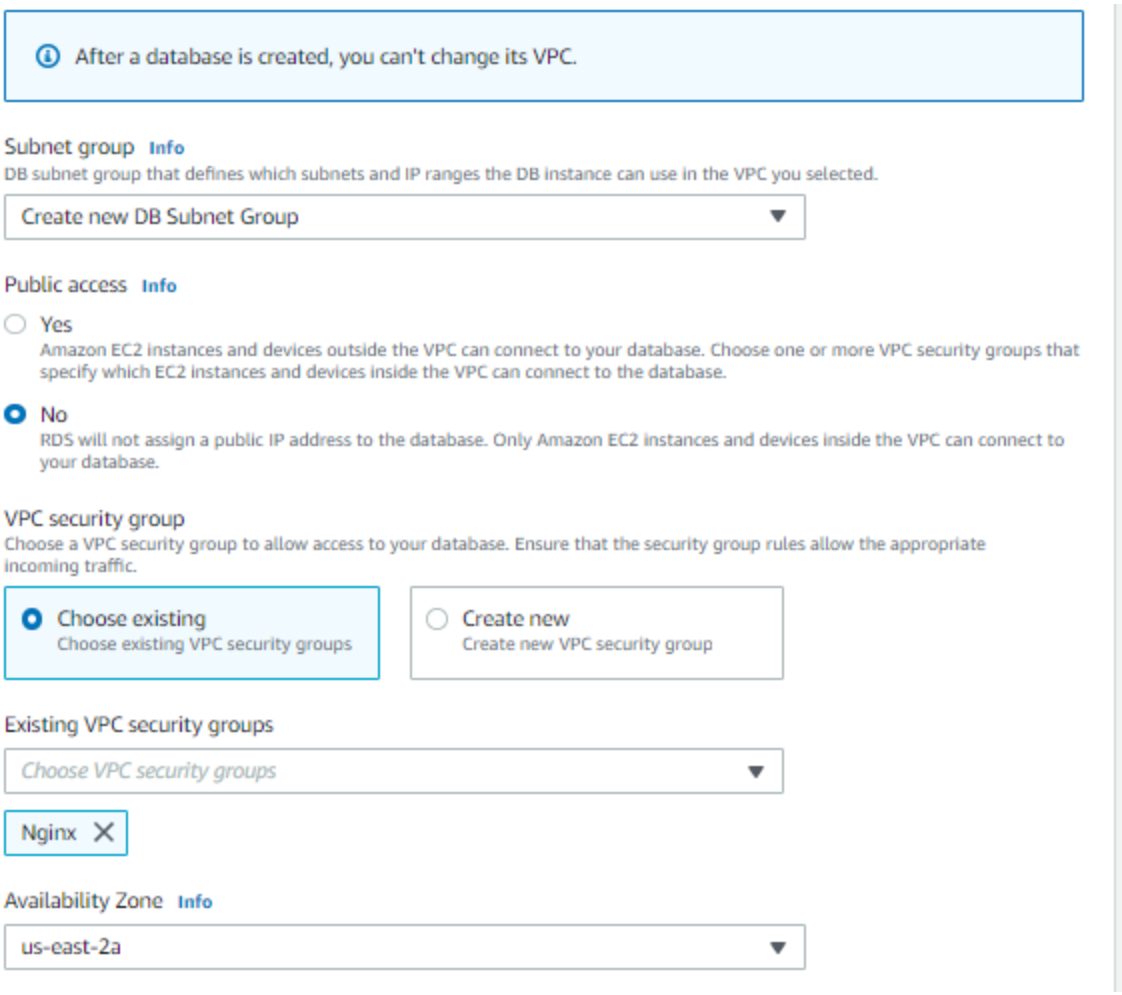
db.t2.micro
1 vCPUs 1 GiB RAM Not EBS Optimized ▼

◯ Include previous generation classes

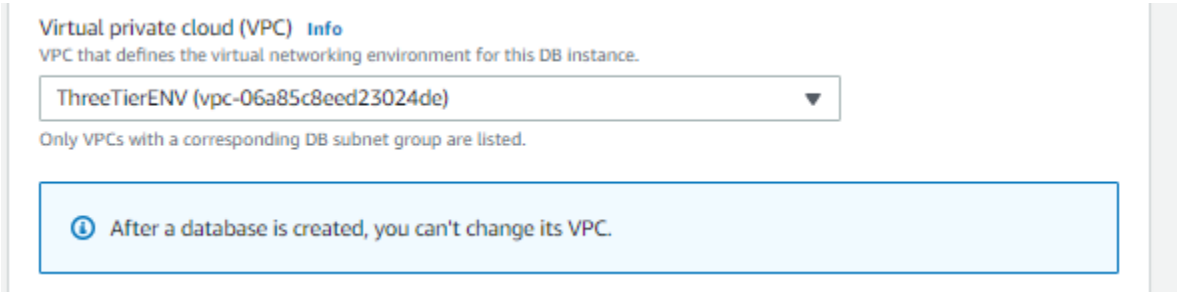
Keep Default



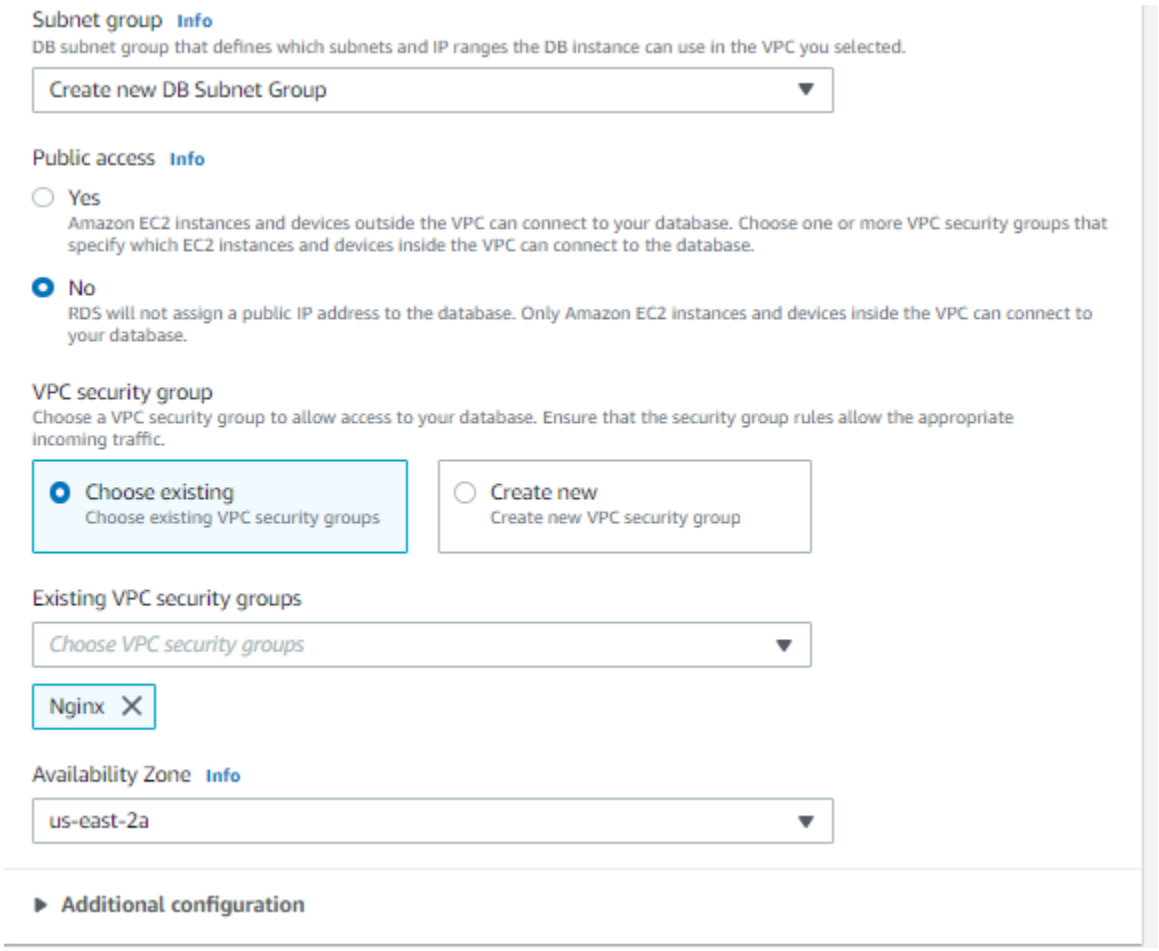
Pick



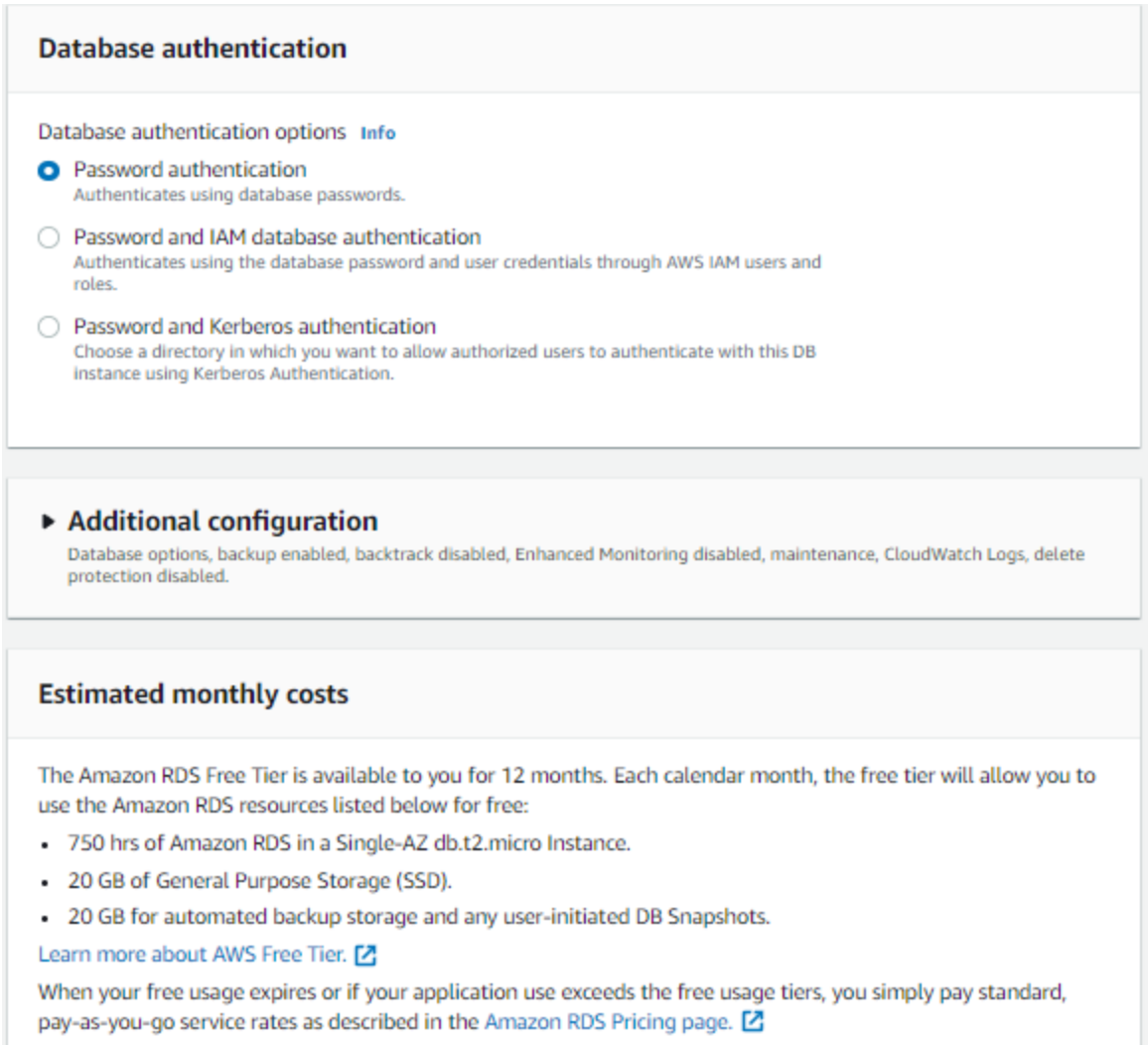
Pick your vpc – which is the three-tier



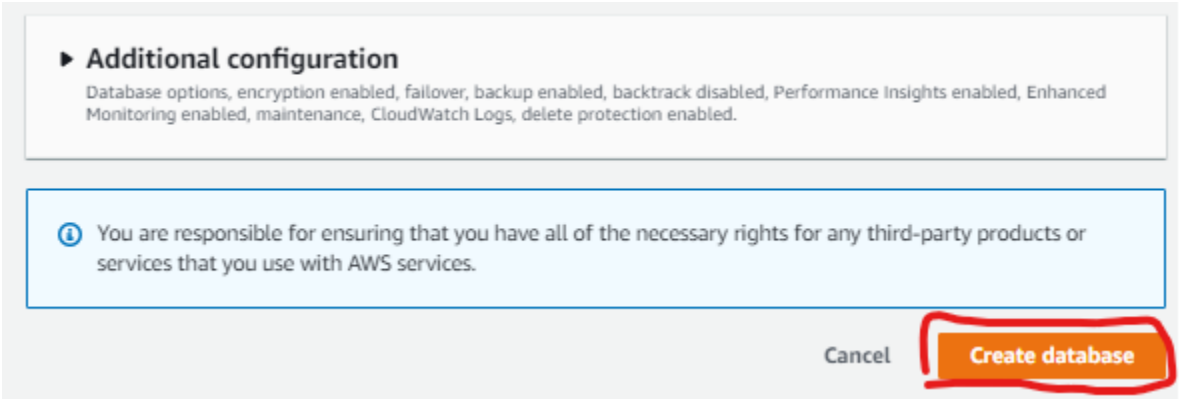
Pick no public access and use nginx security group because you want to allow Http and https



Everything else default



Go below and pick create database, it will take you some time



Task 4:

SSH into your phpMyadmin. If you aren already connected go into bastion and then connect to phpmyadmin with the new pem key.

sudo apt-get update && sudo apt-get upgrade -y

Instal apache
sudo apt-get install apache2 -y

Install PHP and a module that will have php connect to my sql server
sudo apt install php libapache2-mod-php php-mysql -y

Now we have to make sure our web page is working so change directory to where apache host web pages
cd /var/www/html

Create a PHP file
sudo nano test.php

Paste this inside test.php
<?php phpinfo();

Install MySQL server
sudo apt install mysql-server -y

Run MYSQL installation
sudo mysql_secure_installation
Yes
1
Same passwords as the password for rds
Yes
<ENTERKEY>
<ENTERKEY>
<ENTERKEY>
<ENTERKEY>

Enter the following command inside the interactive shell

show databases;

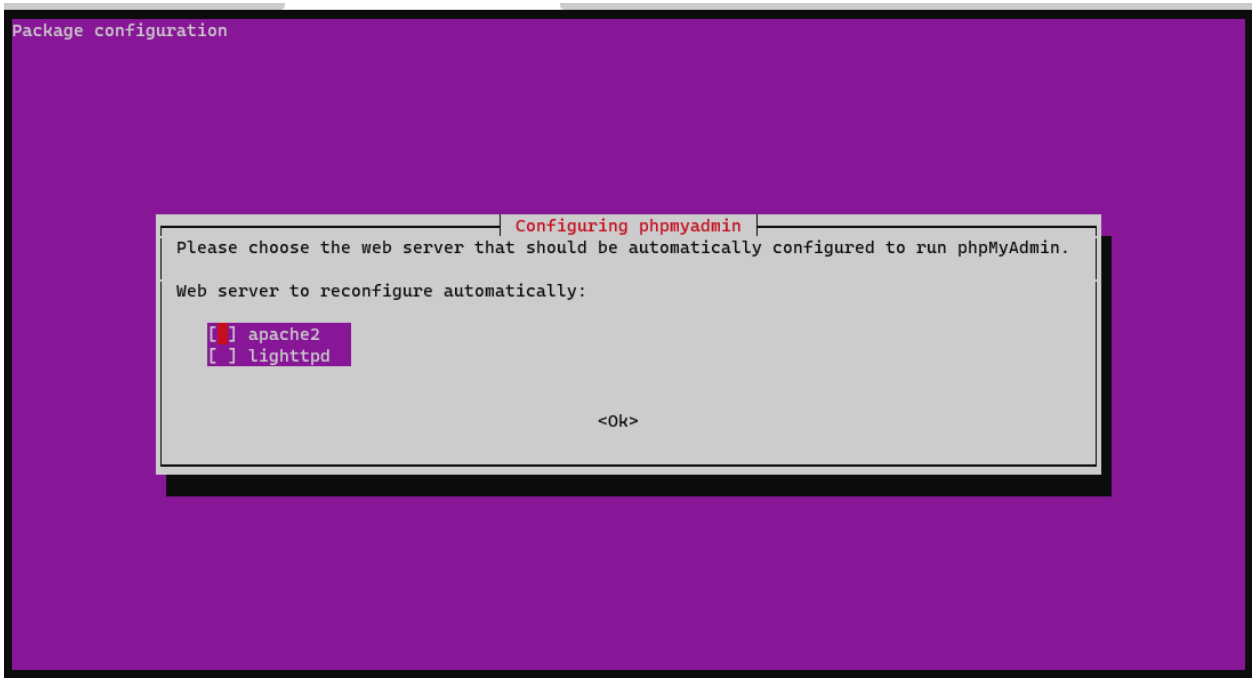
```
ubuntu@ip-192-168-180-20:/var/www/html$ show databases;

Command 'show' not found, but can be installed with:

sudo apt install mailutils-mh  # version 1:3.7-2.1, or
sudo apt install mmh           # version 0.4-2
sudo apt install nmh           # version 1.7.1-6
```

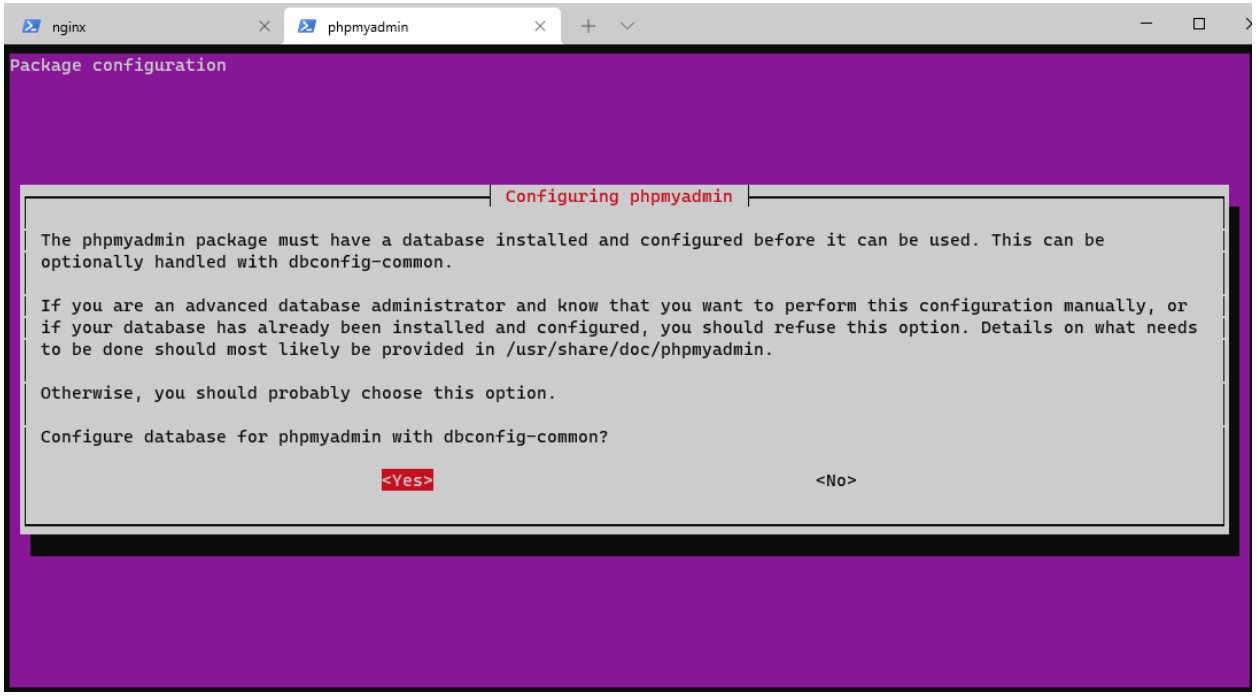
Logout using the command:exit

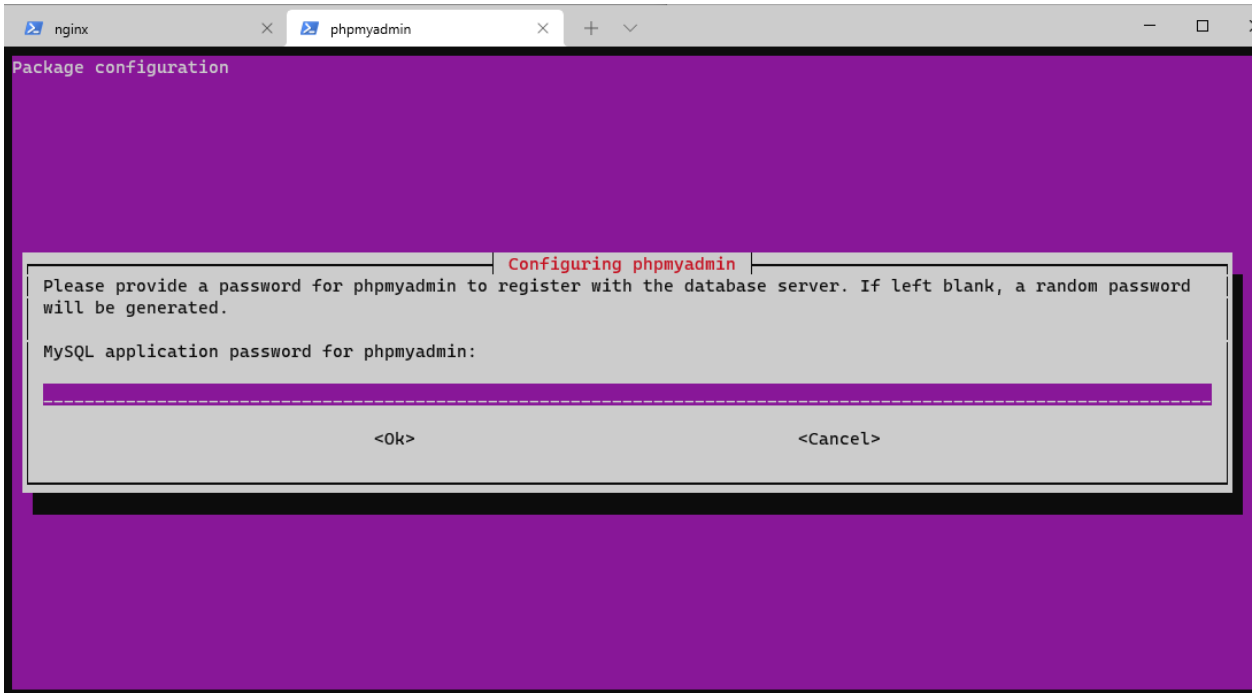
```
ubuntu@ip-192-168-180-20:/var/www/html$ exit
logout
There are stopped jobs.
```

Download packages for phpmyadmin

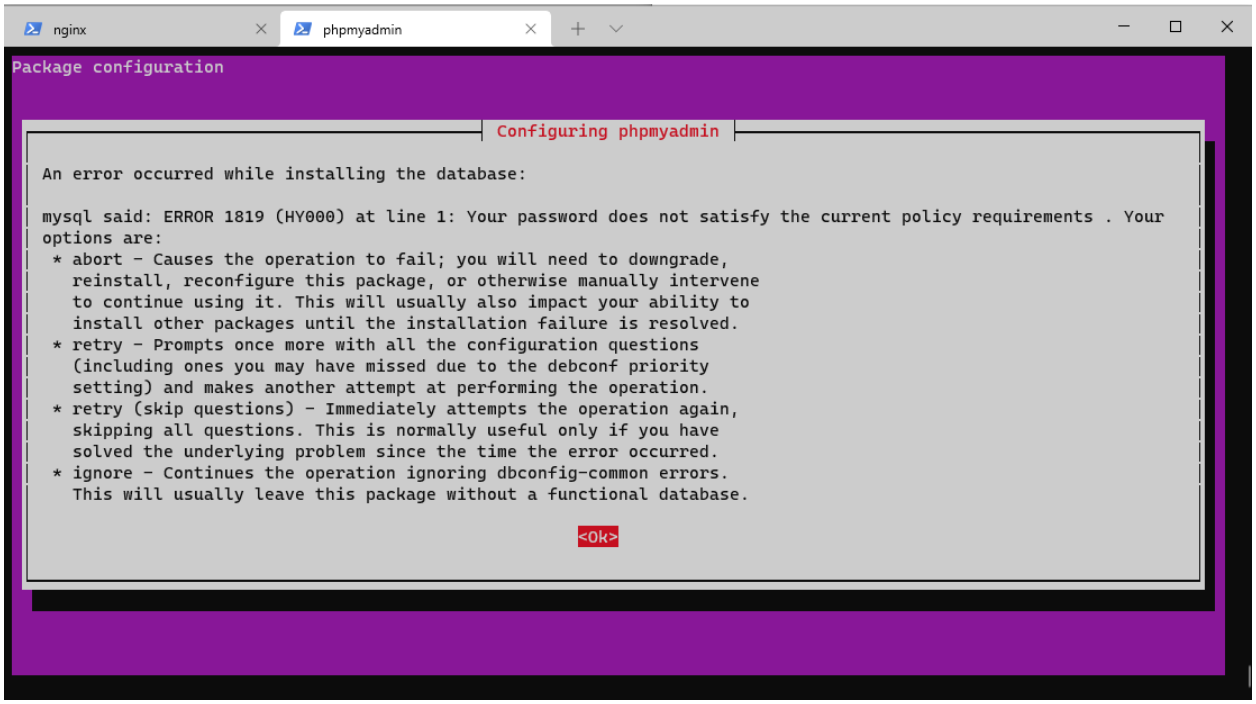
sudo apt install phpmyadmin php-mbstring php-zip php-gd php-json php-curl -y

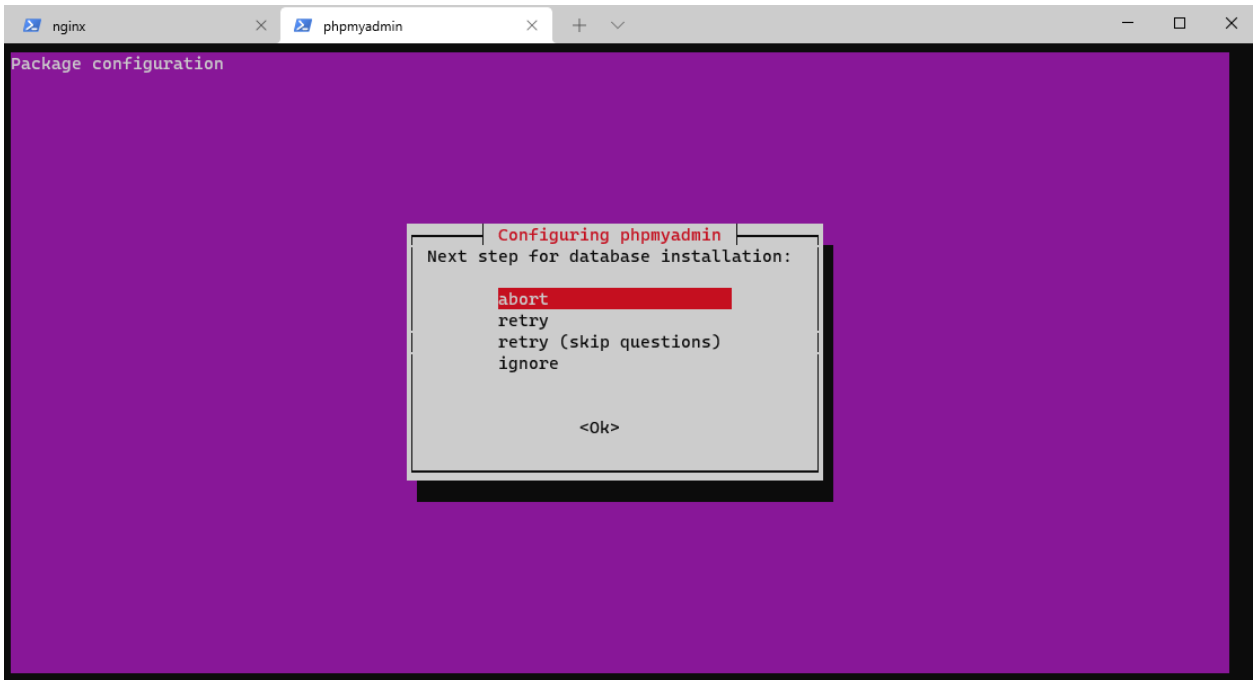






Password is same as rds
Reconform password





Log back into mysql
sudo mysql

Paste the following inside the MySQL interactive shell
SELECT user,authentication_string,plugin,host FROM mysql.user;

Root empty

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

mysql> SELECT user,authentication_string,plugin,host FROM mysql.user;
+------------------+------------------------------------------------------------------------+-----------------------+---
--------+
| user             | authentication_string                                                  | plugin                | ho
st      |
+------------------+------------------------------------------------------------------------+-----------------------+---
--------+
| debian-sys-maint | $A$005$&?OFDTj□:L□zS6□F□□ .sInWxYxz3ZKmru7rFqxAhPI9u/hVdCthpmrR9vH8oiA | caching_sha2_password | lo
calhost |
| mysql.infoschema | $A$005$THISISACOMBINATIONOFINVALIDSALTANDPASSWORDTHATMUSTNEVERBRBEUSED | caching_sha2_password | lo
calhost |
| mysql.session    | $A$005$THISISACOMBINATIONOFINVALIDSALTANDPASSWORDTHATMUSTNEVERBRBEUSED | caching_sha2_password | lo
calhost |
| mysql.sys        | $A$005$THISISACOMBINATIONOFINVALIDSALTANDPASSWORDTHATMUSTNEVERBRBEUSED | caching_sha2_password | lo
calhost |
| root             |                                                                        | auth_socket           | lo
calhost |
+------------------+------------------------------------------------------------------------+-----------------------+---
--------+
5 rows in set (0.01 sec)

mysql>
```

Run the following command inside the MySQL interactive shell
UNINSTALL COMPONENT "file://component_validate_password";

```
mysql> UNINSTALL COMPONENT "file://component_validate_password";
Query OK, 0 rows affected (0.00 sec)
```

We can then run the next command inside the MySQL interactive shell
INSTALL COMPONENT "file://component_validate_password";

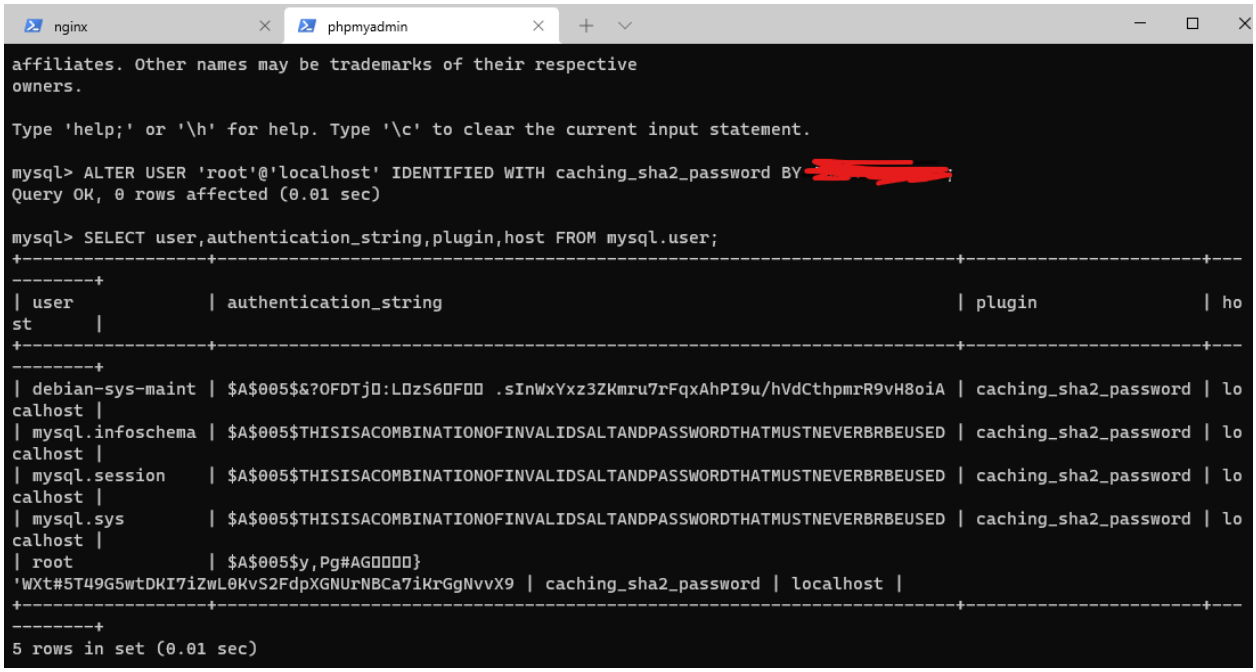
Exit ysql
Exit

Install the following packages
sudo phpenmod mbstring

Go back into the MySQL
sudo mysql

We can use the following command which will use a hashing algorithm to encrypt our password and store it into the root localhost field.
ALTER USER 'root'@'localhost' IDENTIFIED WITH caching_sha2_password BY 'Bkl.com1384';

We can check if the changes were made using the following command
SELECT user,authentication_string,plugin,host FROM mysql.user;



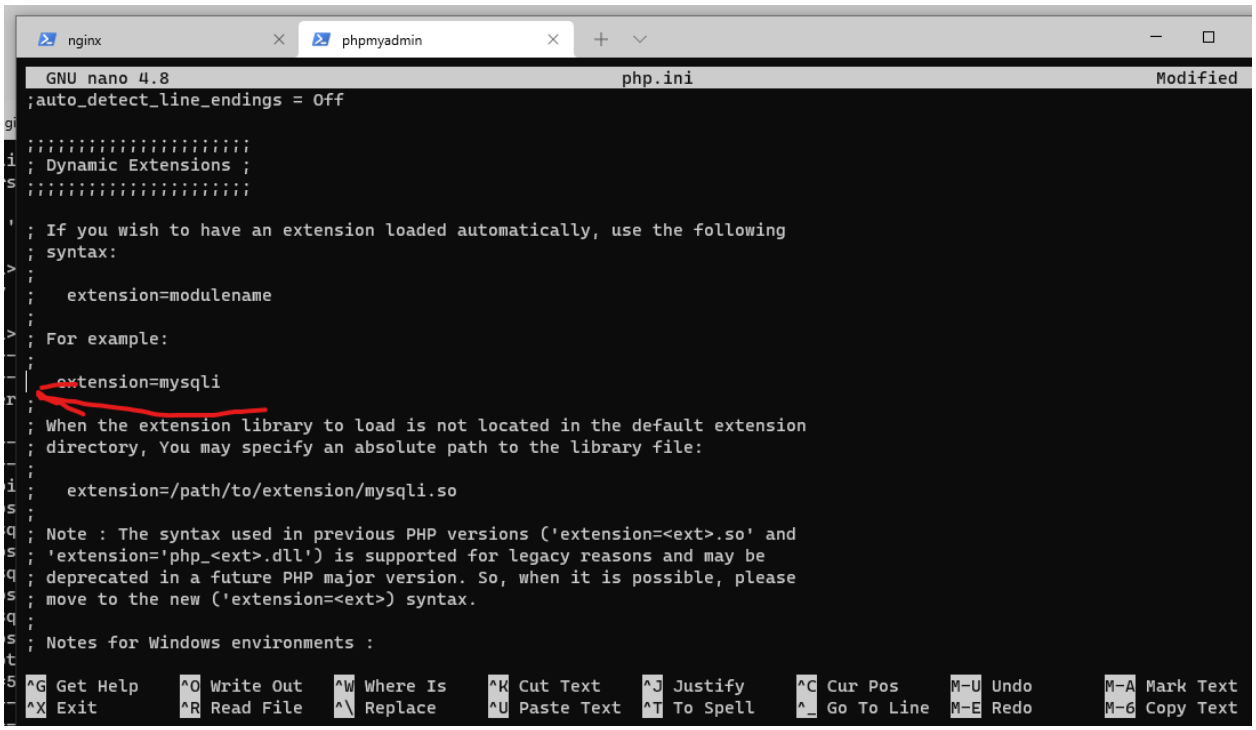
Exit the MySQL interactive shell

Exit

Change directory to the following
cd /etc/php/7.4/apache2/

We will need to edit a file…
sudo nano php.ini

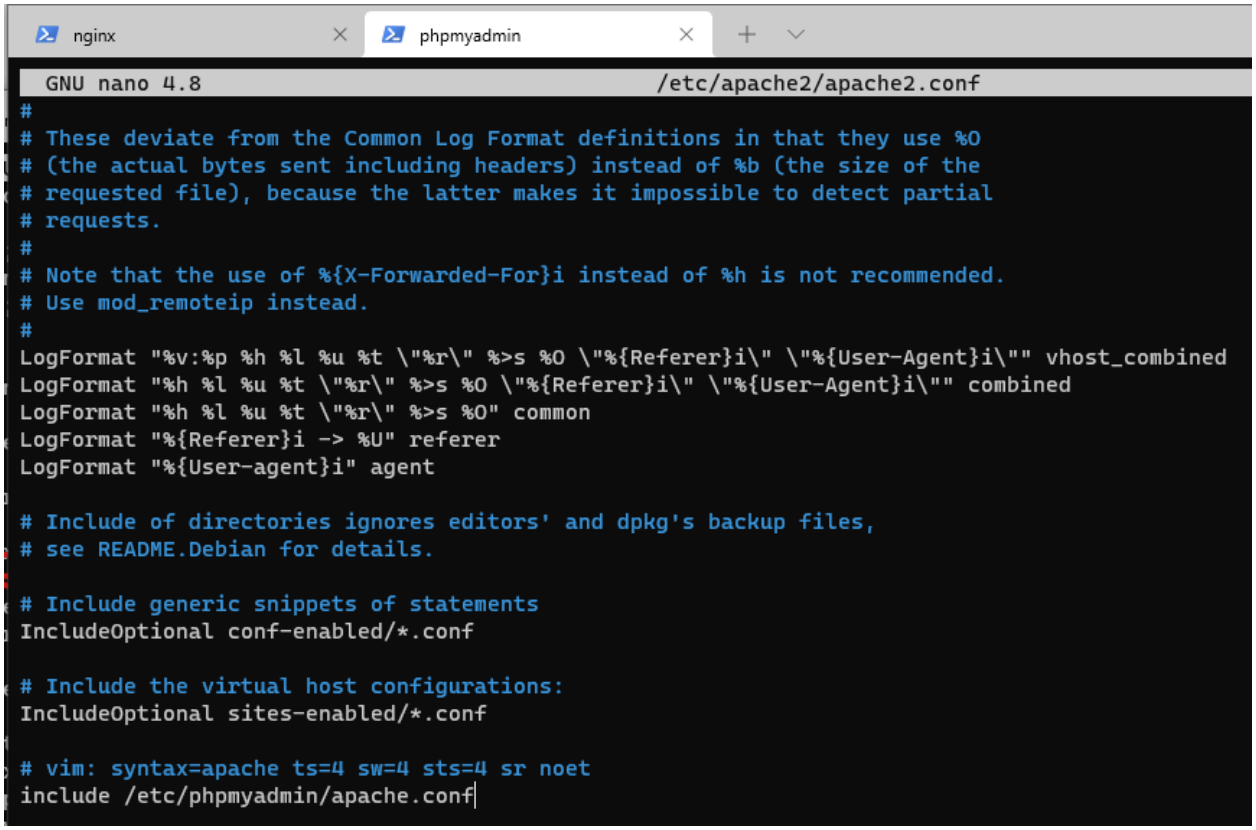
Inside Nano select ALT + G. This will allow us to go to a line. Go to line 895 and remove the semicolon ;



Save the file and exit it
CTRL + O
CTRL + X

We will edit the apache2 config file
sudo nano /etc/apache2/apache2.conf

Scroll all the way to the bottom and paste the following
include /etc/phpmyadmin/apache.conf



Save the file and exit it
CTRL + O
CTRL + X

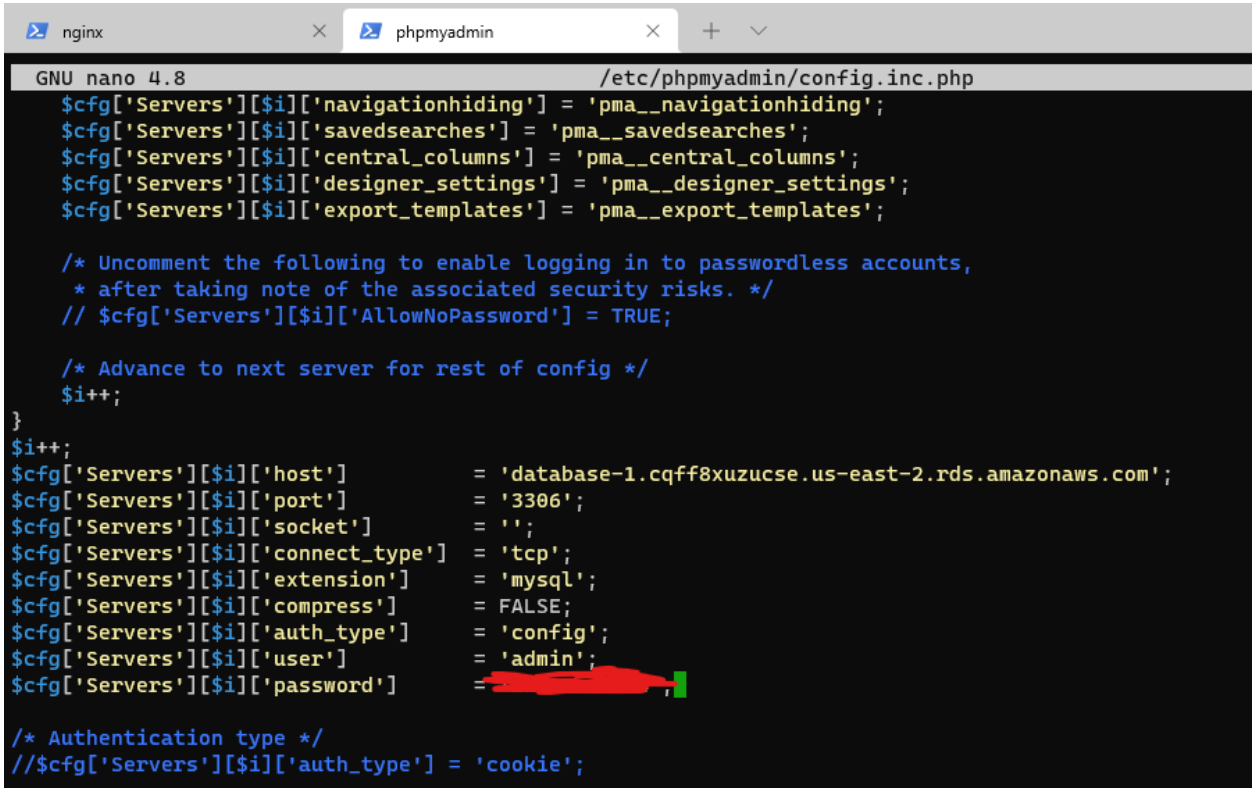
We will need to restart apache
sudo systemctl restart apache2

Edit the config file
sudo nano /etc/phpmyadmin/config.inc.php

Inside nano select ALT + G. This will allow us to go to a line. Go to line 102 and paste the following below

$cfg['Servers'][$i]['host'] = '**__FILL_IN_DETAILS__**';
$cfg['Servers'][$i]['port'] = '3306';
$cfg['Servers'][$i]['socket'] = '';
$cfg['Servers'][$i]['connect_type'] = 'tcp';

$cfg['Servers'][$i]['extension'] = 'mysql';
$cfg['Servers'][$i]['compress'] = FALSE;
$cfg['Servers'][$i]['auth_type'] = 'config';
$cfg['Servers'][$i]['user'] = '**__FILL_IN_DETAILS__**';
$cfg['Servers'][$i]['password'] = '**__FILL_IN_DETAILS__**';



Crtl o vs certl x

We can now test our connection to the new database. We should be able to log into your AWS RDS

curl localhost:80/phpmyadmin/

Restart nginx

sudo systemctl restart apache2

We can now configure the NGINX Proxy. Exit the phpMyAdmin EC2

exit

Connect to your NGINX EC2 instance

ssh into the bastion EC2 -> ssh into the NGINX EC2

Run the following command
sudo apt-get update && sudo apt-get upgrade -y

Install NGINX
sudo apt-get install nginx -y

Change directories to Sites available
sites-available are conf files that tell NGINX where to look for.
cd /etc/nginx/sites-available/

We need to unlink the default sites-enabled file
sudo unlink /etc/nginx/sites-enabled/default
sudo unlink /etc/nginx/sites-enabled/reverse-proxy.conf

Unlinking the reverse-proxy.conf will say there is no file. We need to create a configuration file for the reverse proxy
sudo nano reverse-proxy.conf

Paste the following inside the reverse-proxy configuration file (The proxy_pass IP is the phpMyAdmin private IPv4)

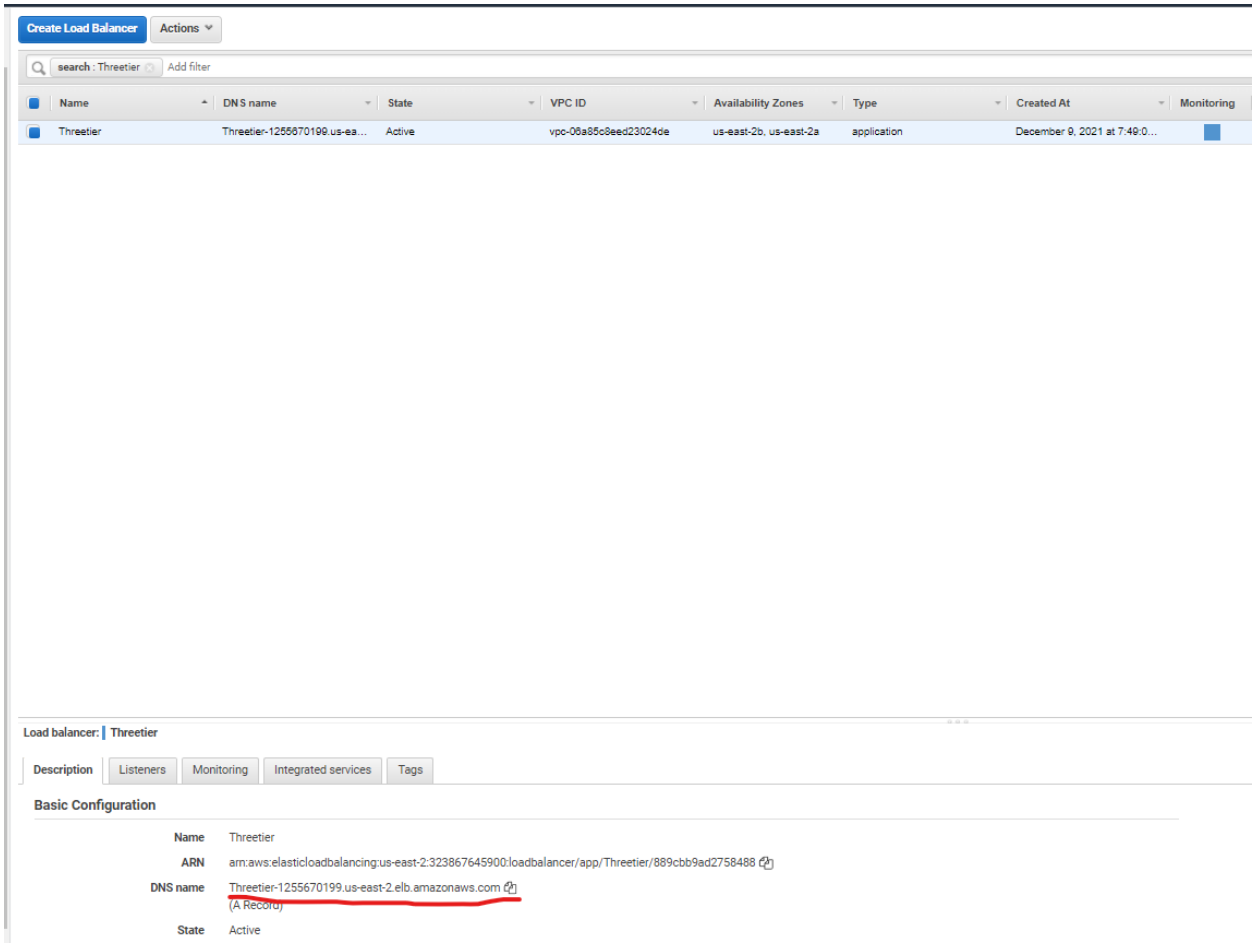
```
server {
   listen 80;
   location / {
      proxy_pass http:/192.168.180.20/;
   }
}
```

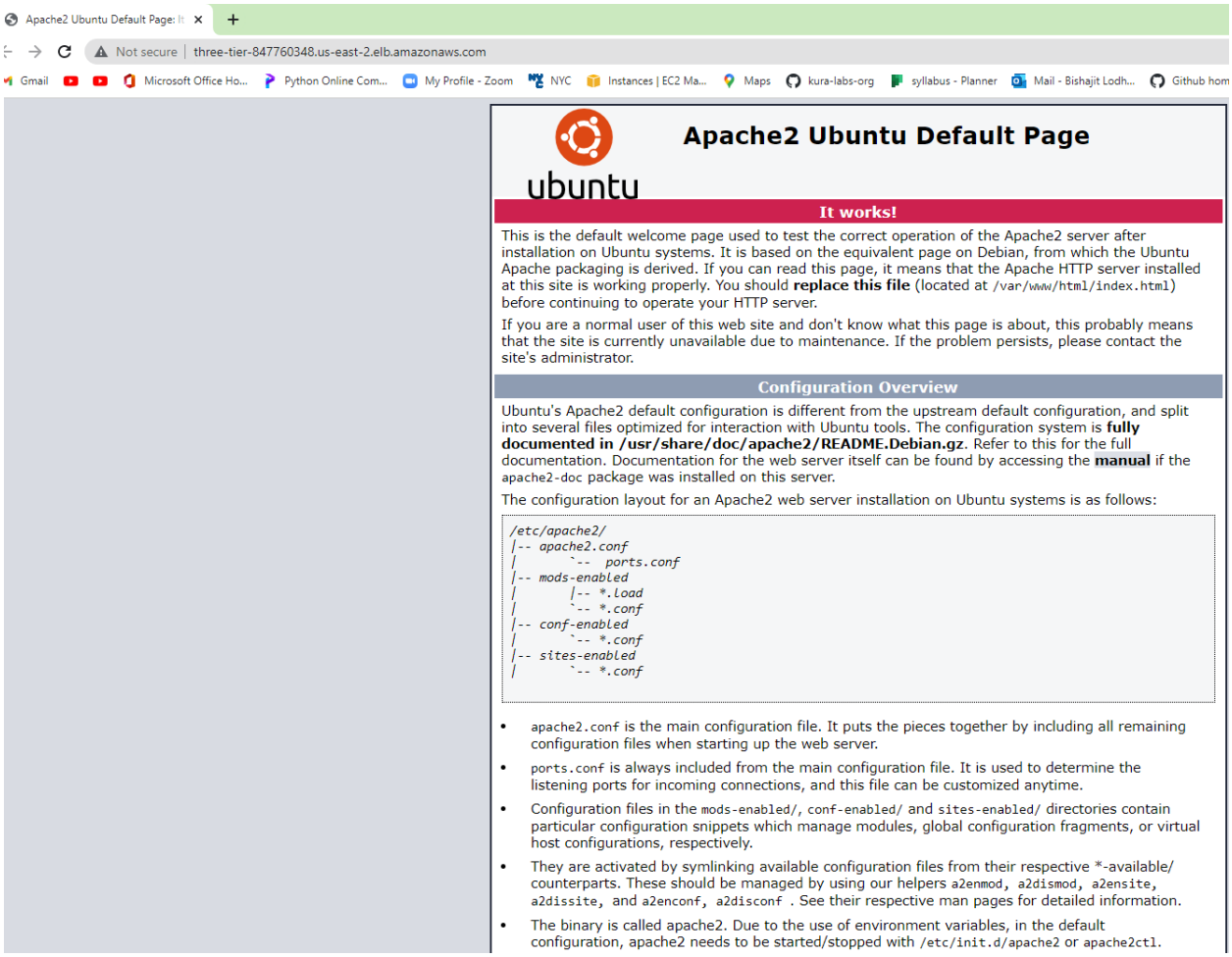
Check if the following directory is empty
ls /etc/nginx/sites-enabled/

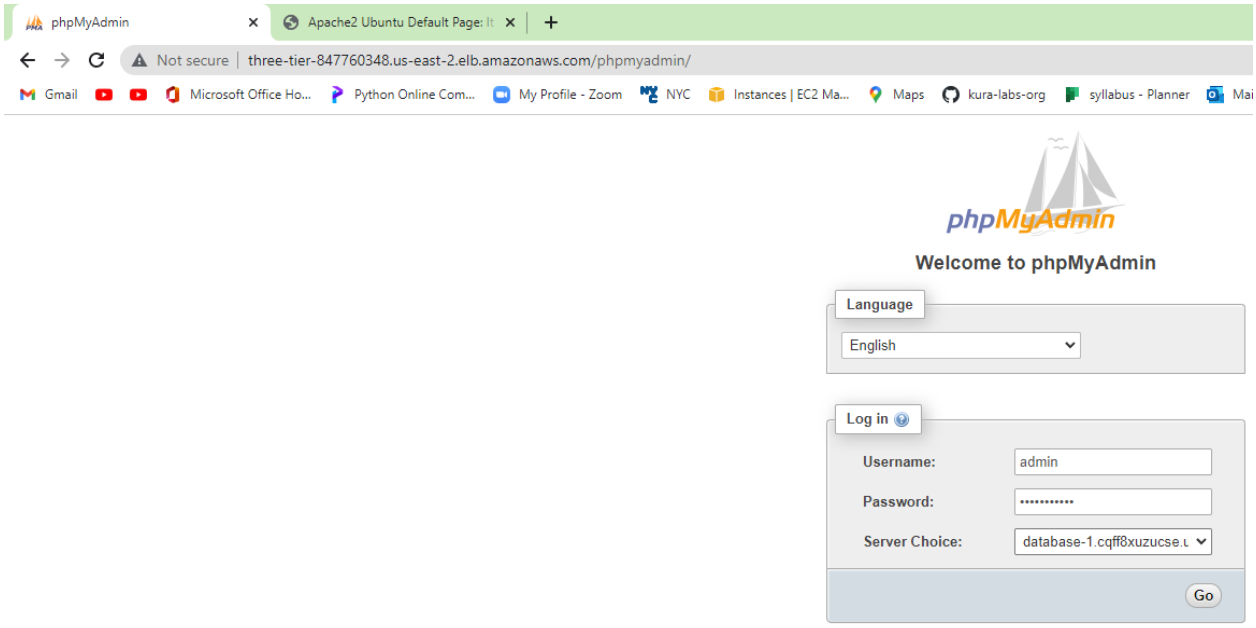
We will link reverse-proxy to sites enabled so that apache can read it and use it. (ONE LINE COMMAND)
sudo ln -s /etc/nginx/sites-available/reverse-proxy.conf
/etc/nginx/sites-enabled/reverse-proxy.conf

We will need to restart NGINX
sudo systemctl restart nginx

Now go to you load balancer


Create Load Balancer
Actions
search : Threetier
Add filter
Name
DNS name
State
VPC ID
Availability Zones
Type
Created At
Monitoring
Threetier
Threetier-1255670199.us-ea...
Active
vpc-06a85c8eed23024de
us-east-2b, us-east-2a
application
December 9, 2021 at 7:49:0...
Load balancer: Threetier
Description
Listeners
Monitoring
Integrated services
Tags
Basic Configuration
Name
Threetier
ARN
arn:aws:elasticloadbalancing:us-east-2:323867645900:loadbalancer/app/Threetier/889cbb9ad2758488
DNS name
Threetier-1255670199.us-east-2.elb.amazonaws.com
(A Record)
State
Active


Apache2 Ubuntu Default Page: It
Not secure | three-tier-847760348.us-east-2.elb.amazonaws.com
Gmail
Microsoft Office Ho...
Python Online Com...
My Profile - Zoom
NYC
Instances | EC2 Ma...
Maps
kura-labs-org
syllabus - Planner
Mail - Bishajit Lodh...
Github hom
ubuntu
Apache2 Ubuntu Default Page
It works!
This is the default welcome page used to test the correct operation of the Apache2 server after installation on Ubuntu systems. It is based on the equivalent page on Debian, from which the Ubuntu Apache packaging is derived. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should replace this file (located at /var/www/html/index.html) before continuing to operate your HTTP server.
If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.
Configuration Overview
Ubuntu's Apache2 default configuration is different from the upstream default configuration, and split into several files optimized for interaction with Ubuntu tools. The configuration system is fully documented in /usr/share/doc/apache2/README.Debian.gz. Refer to this for the full documentation. Documentation for the web server itself can be found by accessing the manual if the apache2-doc package was installed on this server.
The configuration layout for an Apache2 web server installation on Ubuntu systems is as follows:
/etc/apache2/
|-- apache2.conf
| `-- ports.conf
|-- mods-enabled
| |-- *.load
| `-- *.conf
|-- conf-enabled
| `-- *.conf
|-- sites-enabled
| `-- *.conf
apache2.conf is the main configuration file. It puts the pieces together by including all remaining configuration files when starting up the web server.
ports.conf is always included from the main configuration file. It is used to determine the listening ports for incoming connections, and this file can be customized anytime.
Configuration files in the mods-enabled/, conf-enabled/ and sites-enabled/ directories contain particular configuration snippets which manage modules, global configuration fragments, or virtual host configurations, respectively.
They are activated by symlinking available configuration files from their respective *-available/ counterparts. These should be managed by using our helpers a2enmod, a2dismod, a2ensite, a2dissite, and a2enconf, a2disconf . See their respective man pages for detailed information.
The binary is called apache2. Due to the use of environment variables, in the default configuration, apache2 needs to be started/stopped with /etc/init.d/apache2 or apache2ctl.


phpMyAdmin
Apache2 Ubuntu Default Page: It
Not secure | three-tier-847760348.us-east-2.elb.amazonaws.com/phpmyadmin/
Gmail
Microsoft Office Ho...
Python Online Com...
My Profile - Zoom
NYC
Instances | EC2 Ma...
Maps
kura-labs-org
syllabus - Planner
phpMyAdmin
Welcome to phpMyAdmin
Language
English
Log in
Username:
admin
Password:
Server Choice:
database-1.cqff8xuzucse.
Go