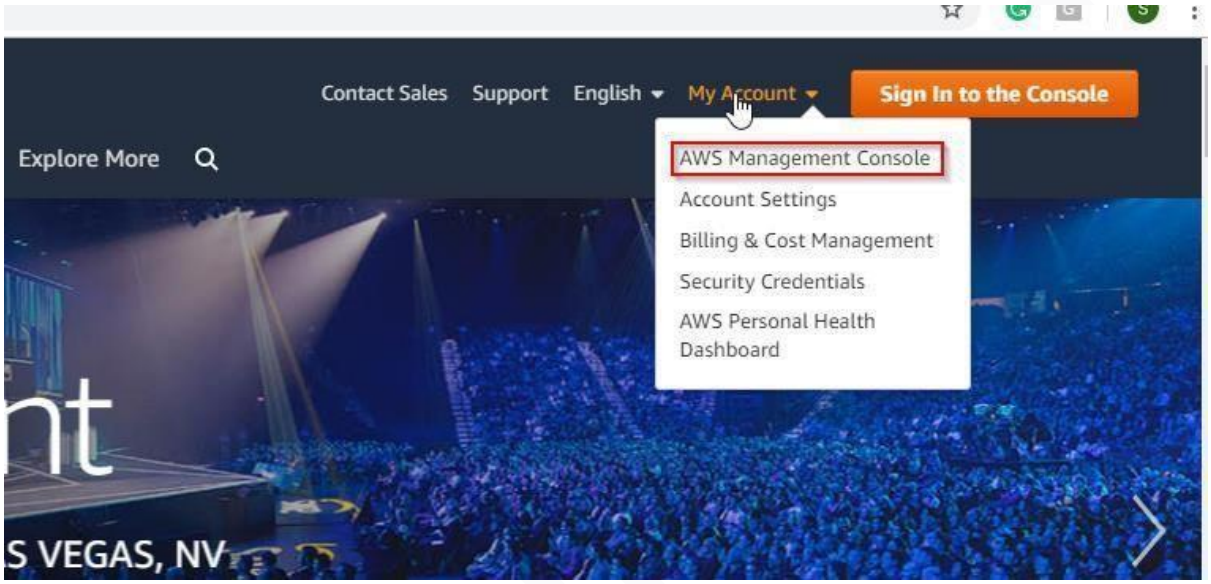
Project- 1: Solution

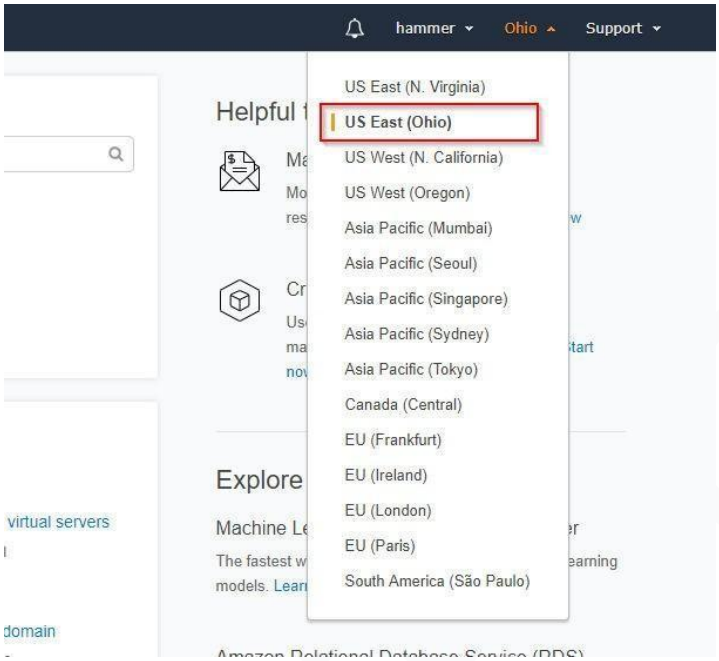
Connect your system with your EC2 instance:

1. First you need to install PuTTY on your system and then connect it with your EC2 instance.
2. Below are the steps for it:

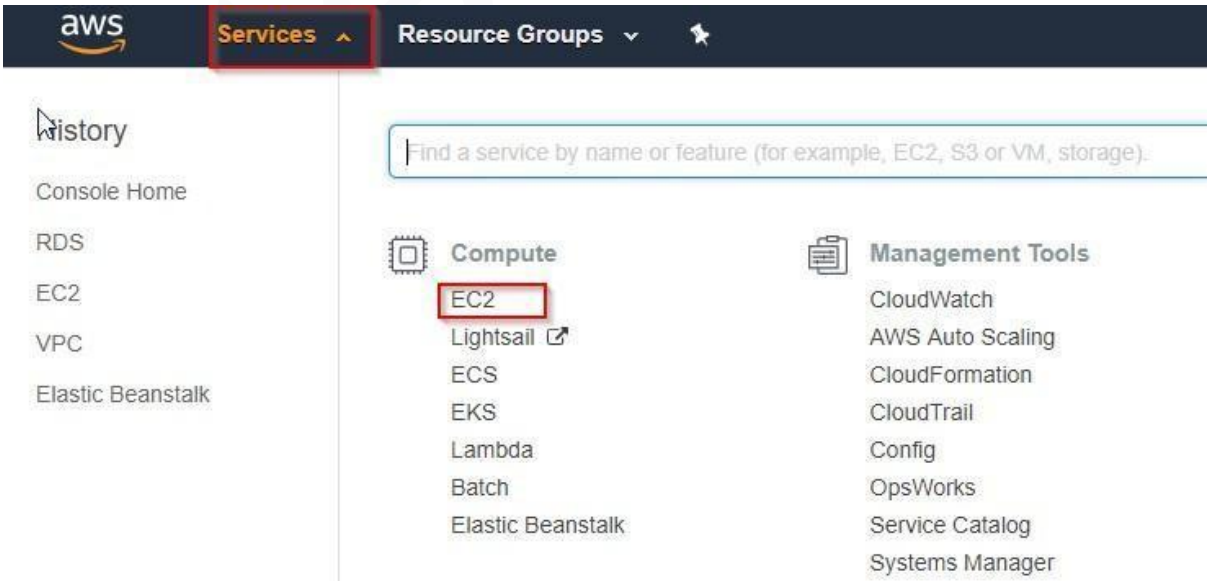
    a. First sign into the AWS Management Console



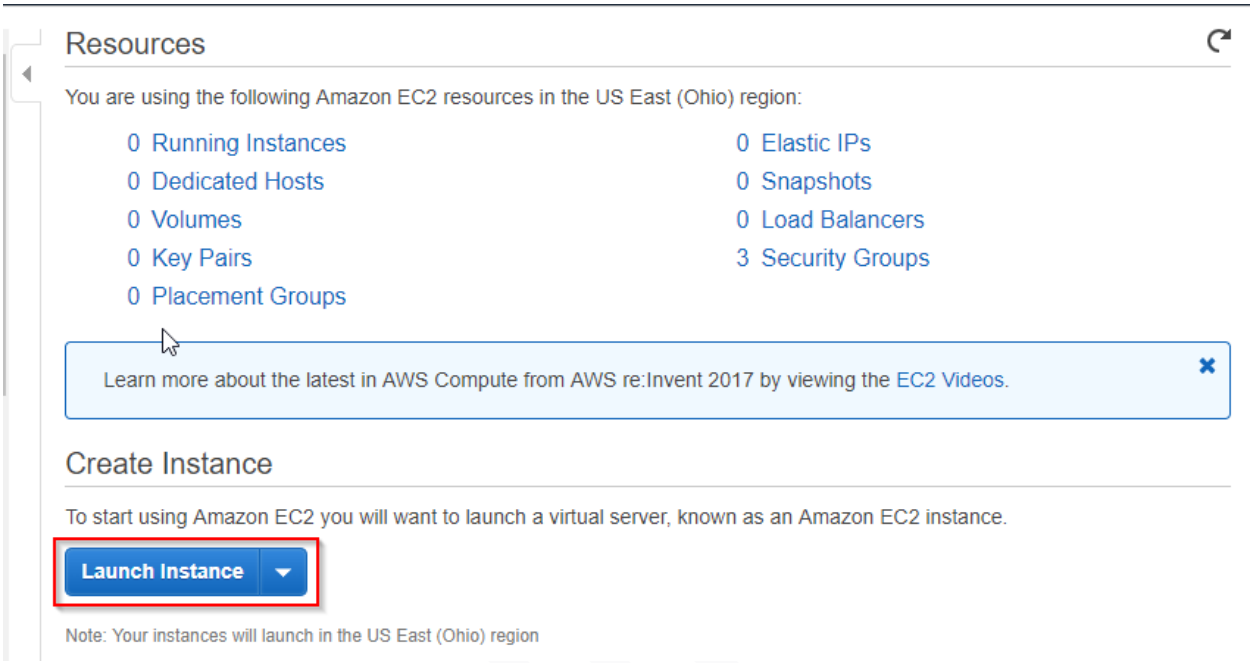
b. Select any region you want. We've selected Ohio here



c. In the **Services** section, you'll see **Compute** where you need to choose **EC2**

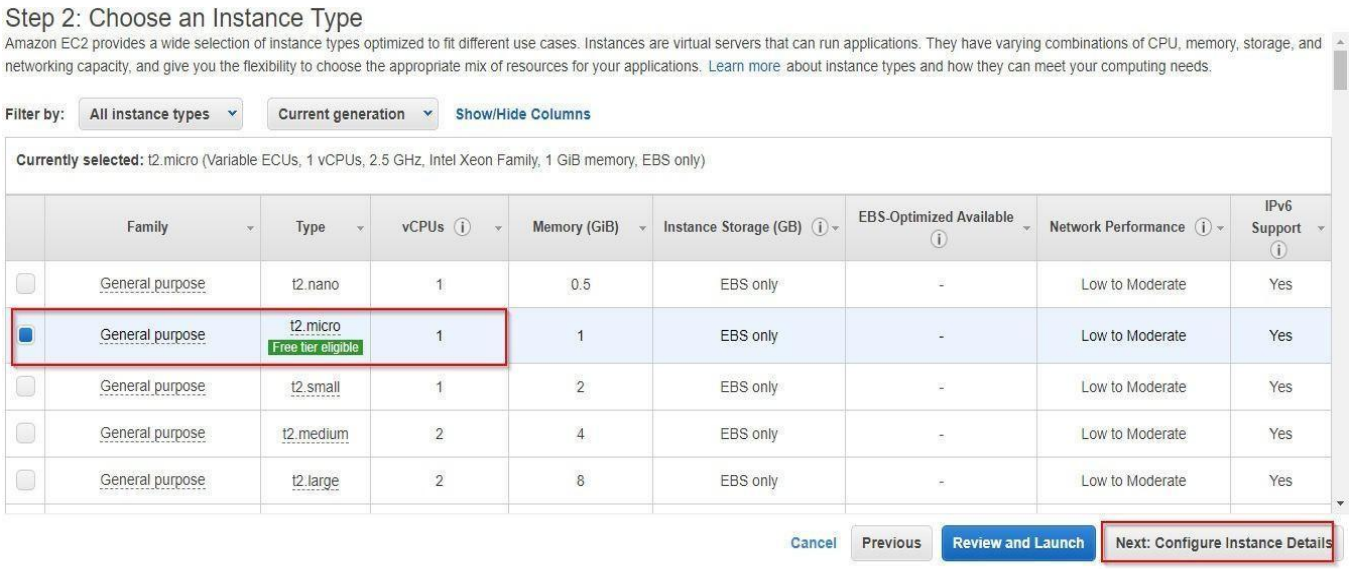


d. Then in the Create Instance section, select the **Launch Instance** option

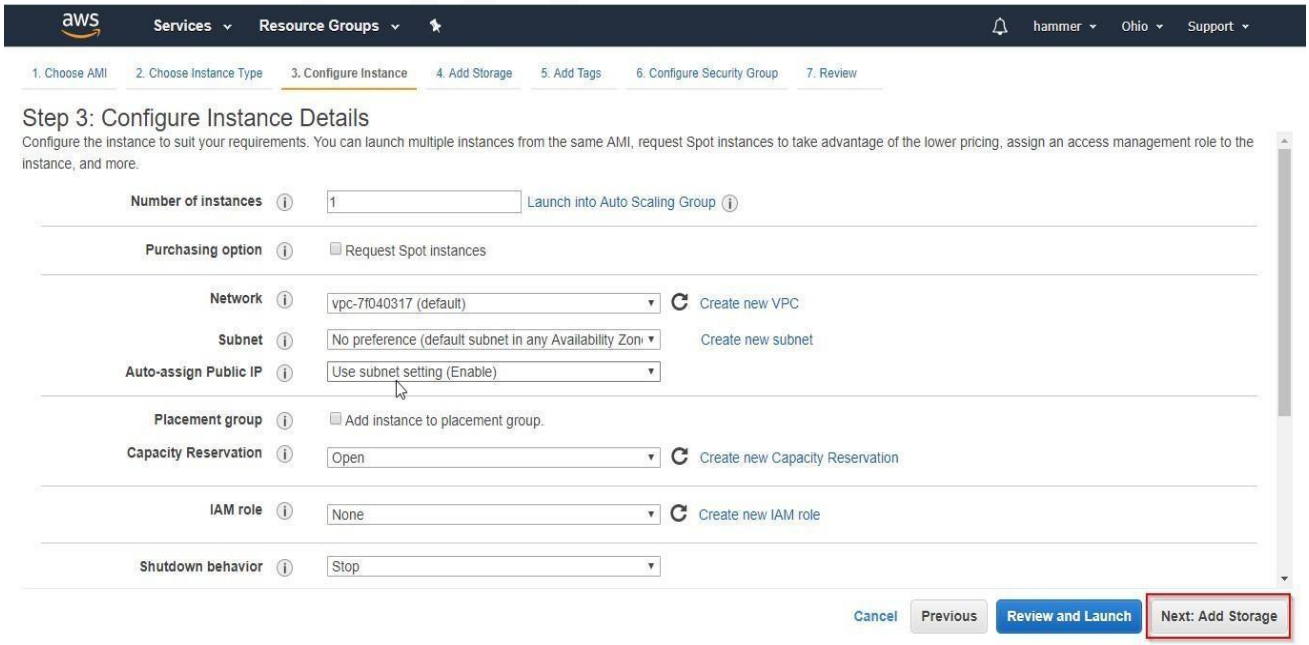


e. Then Select an **AMI** or **Amazon Machine Image**

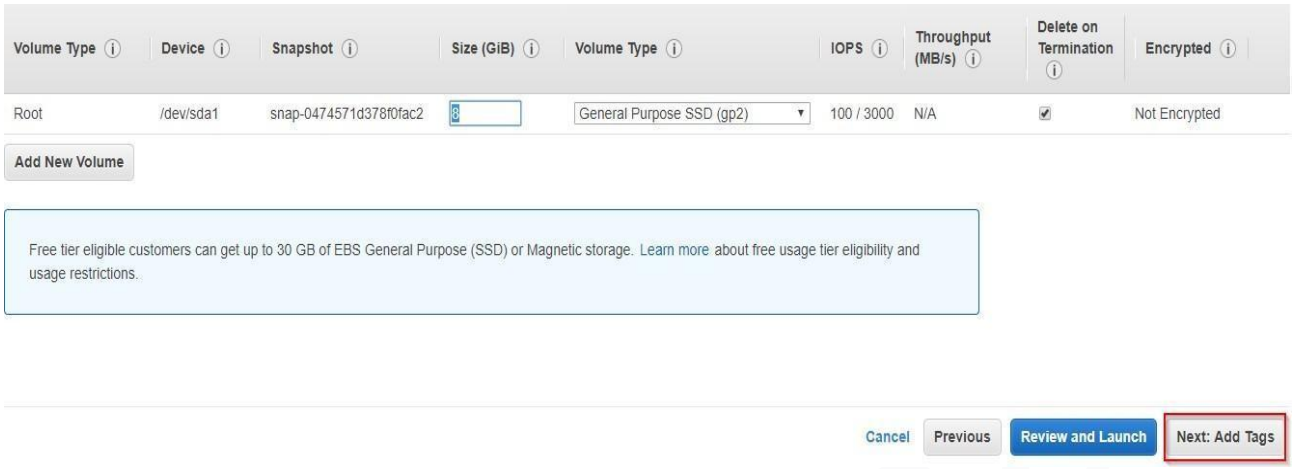
f. Choose your instance type. We're choosing Free tier for demo purposes



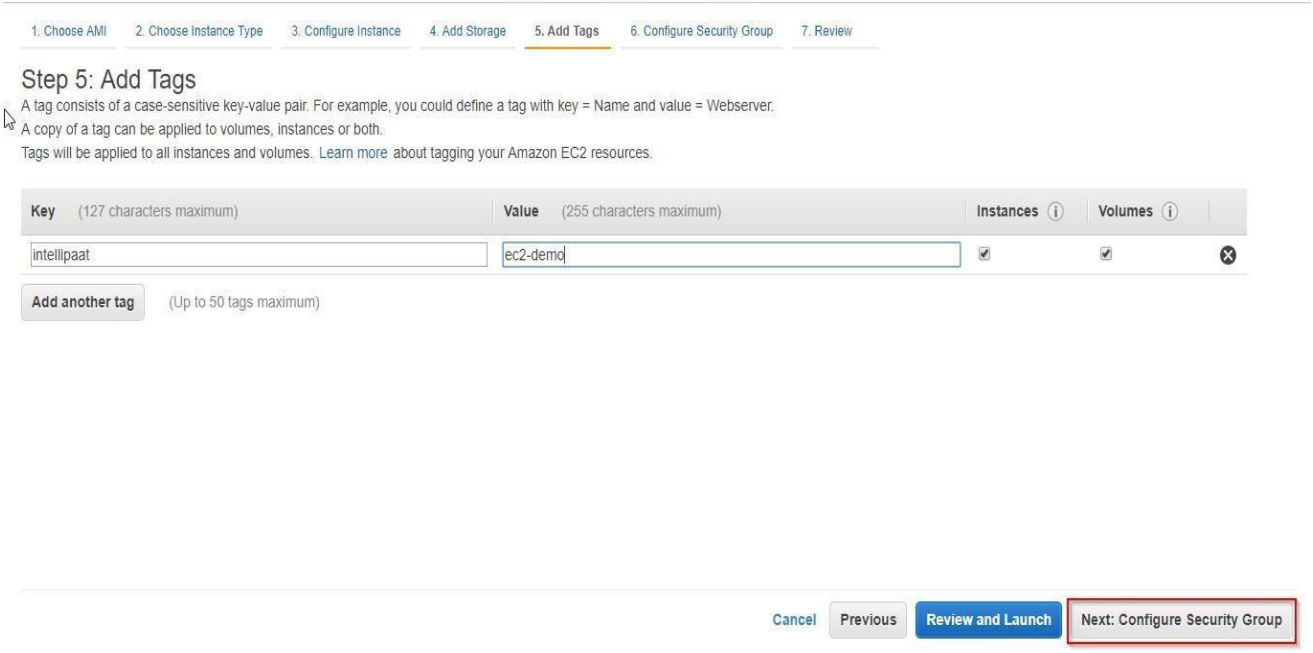
g. Configure your instance details and then select the **Add storage** option



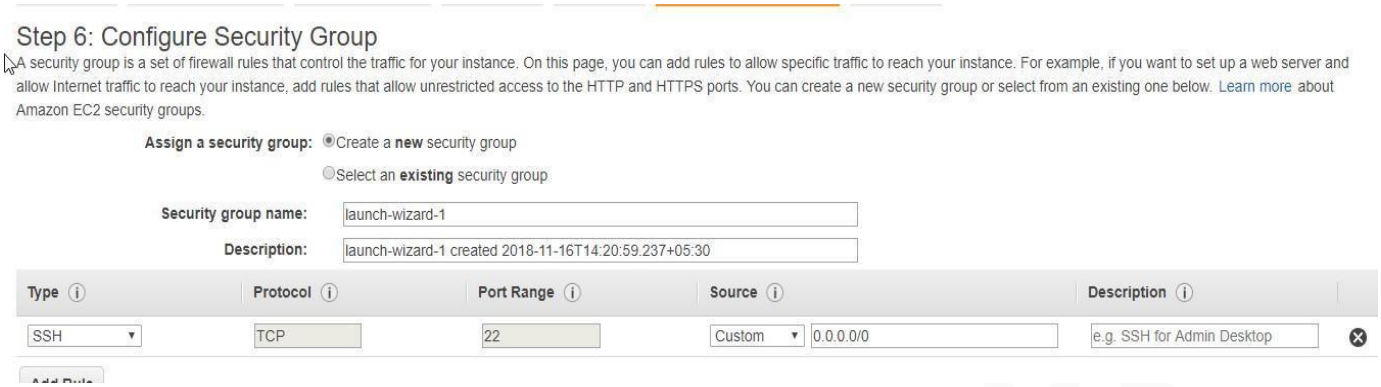
h. Then click on Add Tags



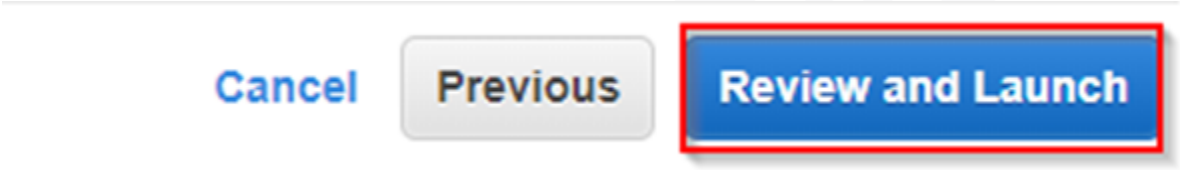
i. Add Tags, name the Key and Value then click Configure Security Group



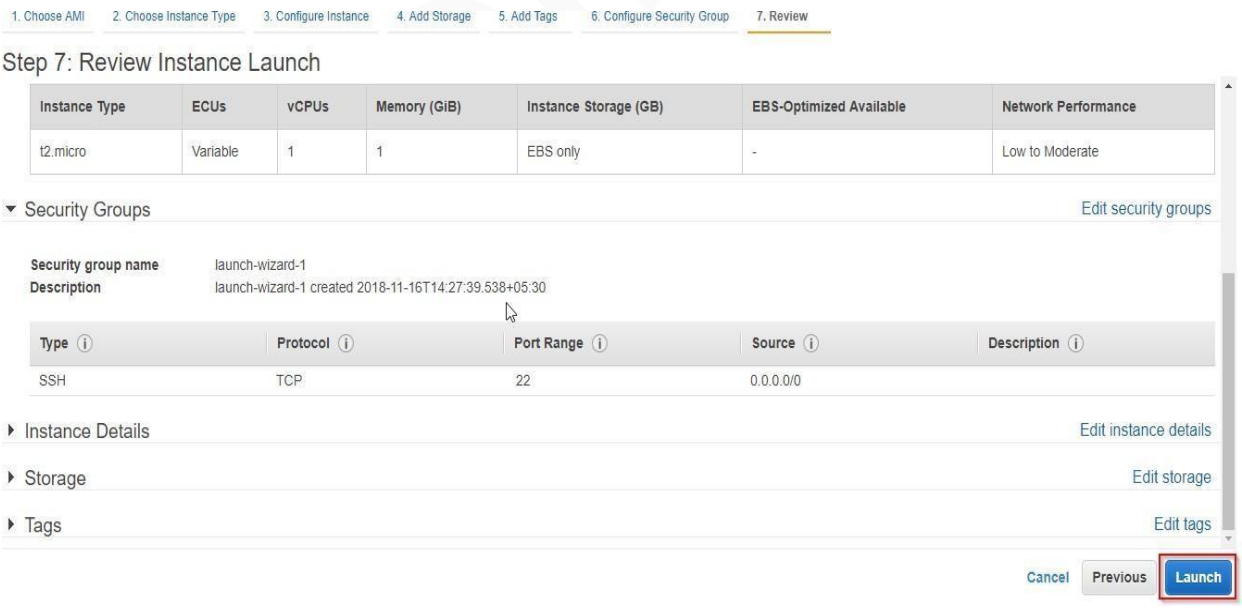
j. Keep the configuration of security group as it



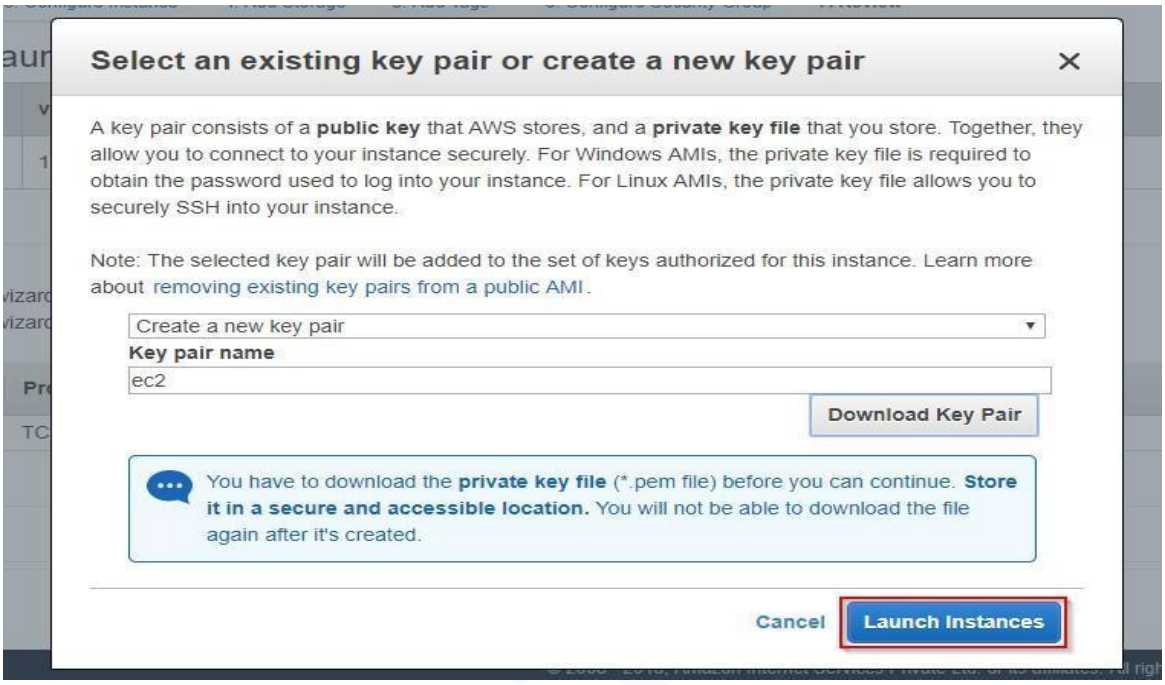
k. Then click Review & Launch



l. Then directly Launch it

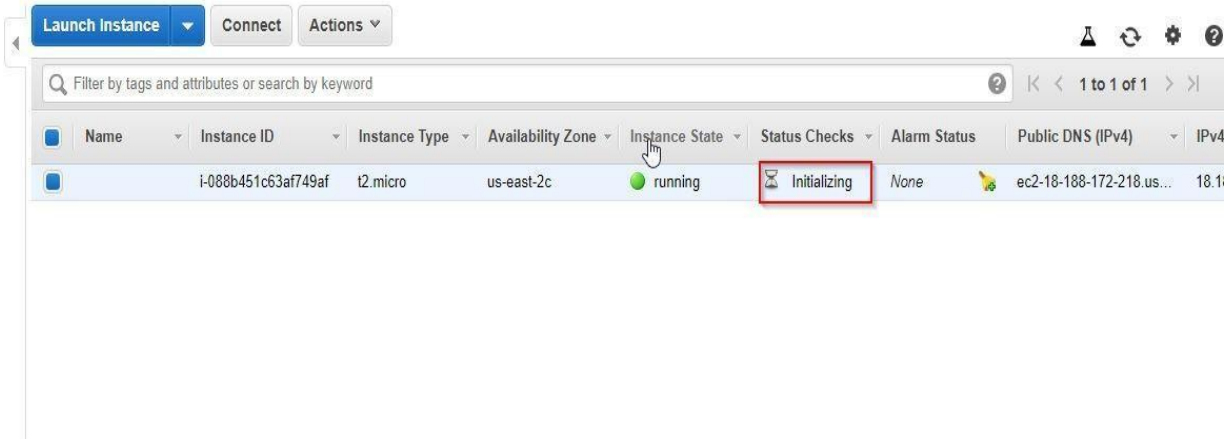


m. Then Create a key pair, download it and then Launch your instance

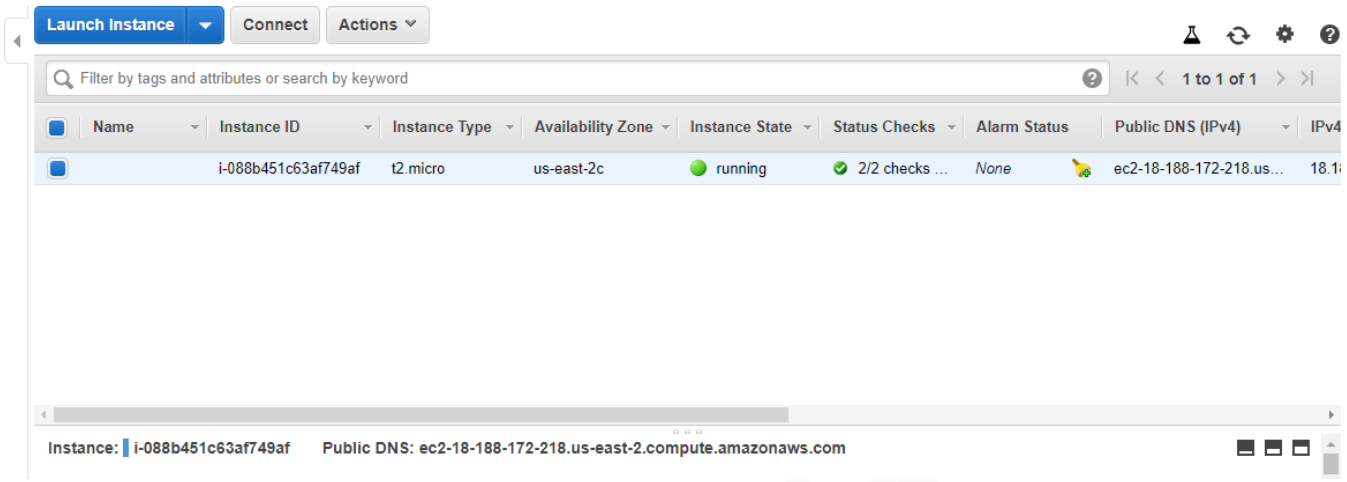


n. Status

o. You will be able to see in your status that your Instance is on Initializing stage



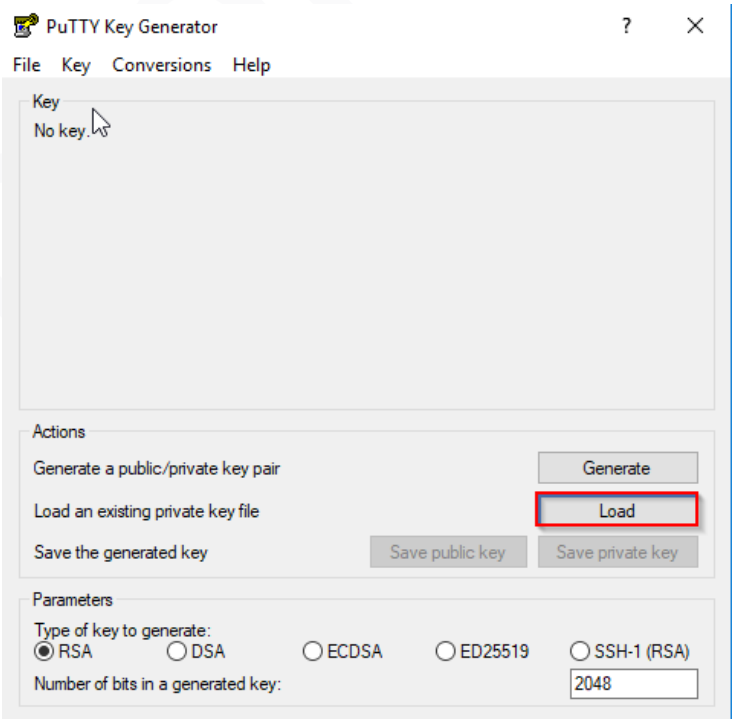
p. Then after few minutes, you will see that now your instance is in running stage



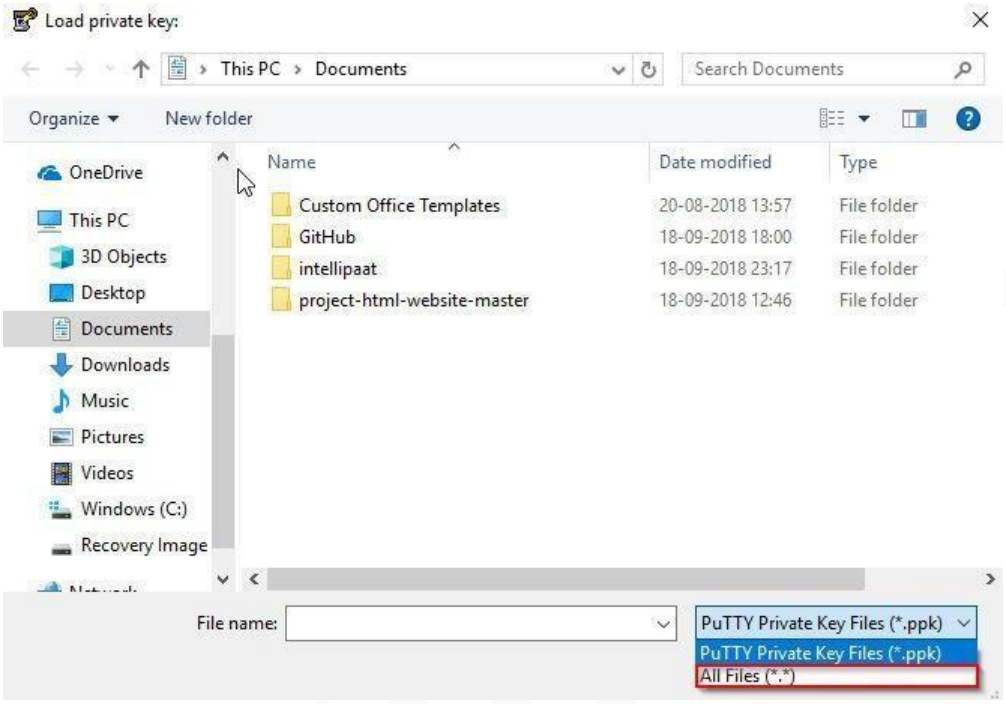
q. Now it's time to convert your private key using PuTTYgen

PuTTY won't be able to support this .pem file, so you'd require a PuTTY gen tool which can convert your .pem file into .ppk format, because you need a .ppk file in order to connect it with your instance.

r. Click Load in your PuTTY gen

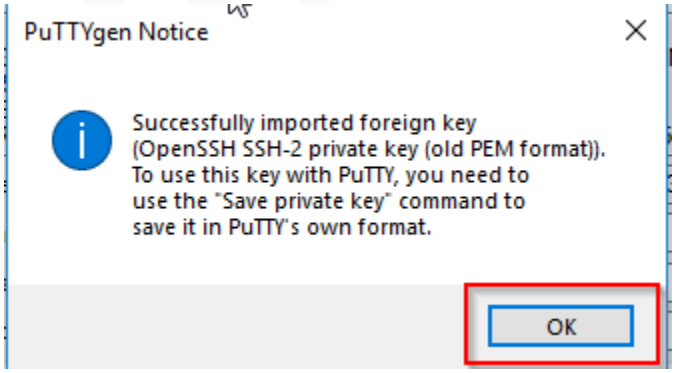


s. PuTTYgen key always shows the .ppk format file, so go to the right bottom bar and select the All files option as shown below

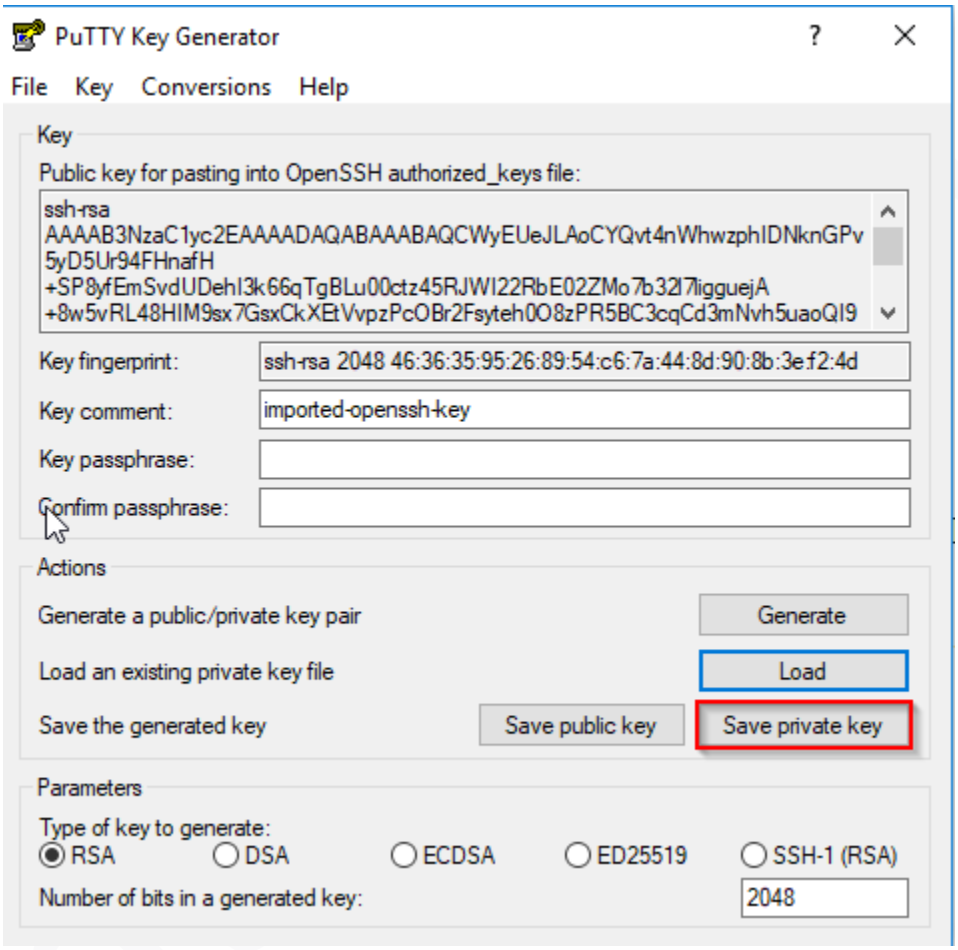


t. Then select the folder where you downloaded this keypair and load it there

u. You will see this option then click OK



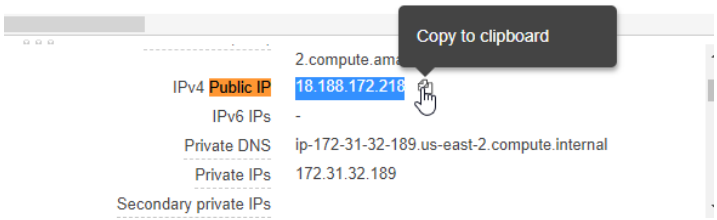
v. Then click on Save the Private key, PuTTY gen will give a warning about saving the key without Key passphrase, click Yes and specify the same name for your file that you gave it in the key pair



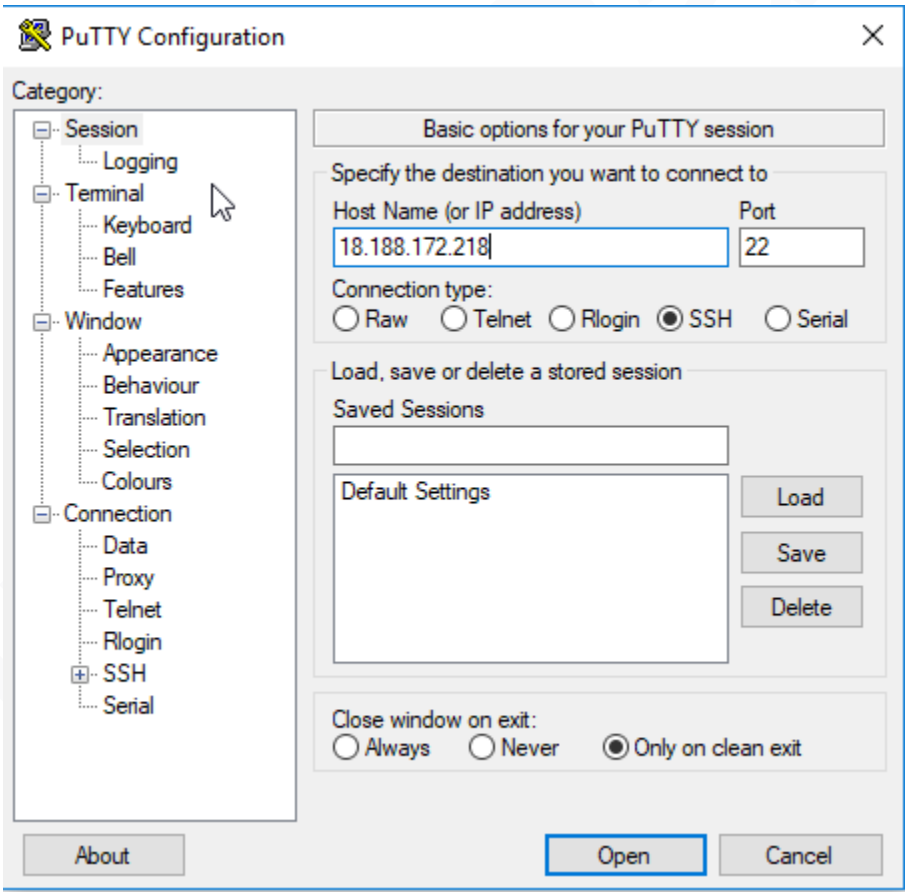
w. Now you will see that in your folder, the .ppk file is already added with that name you had given (in our case, it's ec2)

Connecting to your EC2 Instance using SSH & PuTTY:

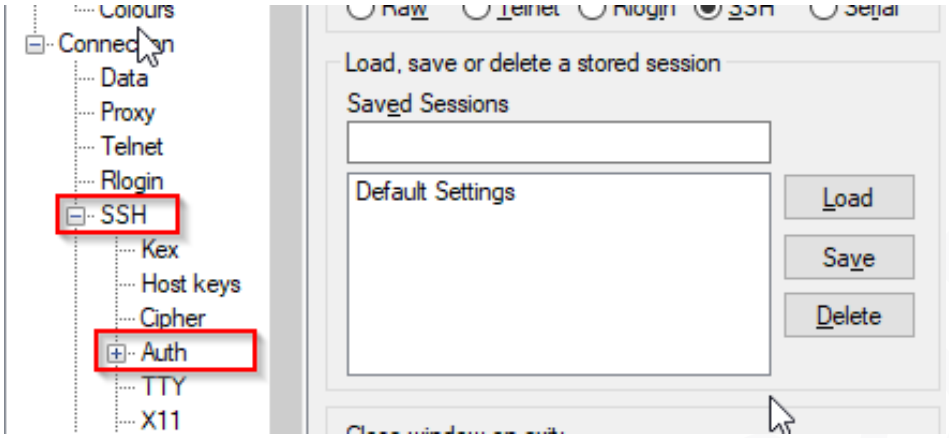
- First open PuTTY.exe then in the Host Name box, add the Public IP of your Instance



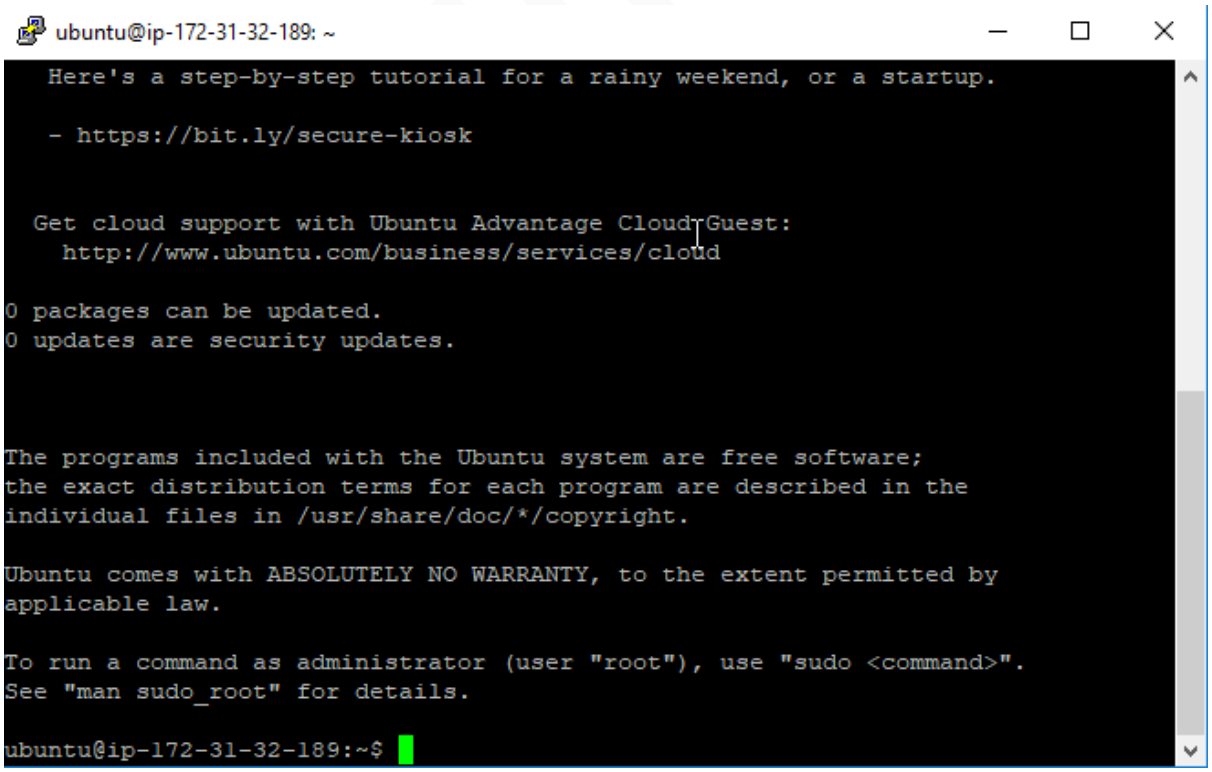
- Copy paste this Public IP in your PuTTY Hostname



- Then in the category list, expand the SSH and Click on AUTH (but don't expand it



- Then Click Open
  - Login as per your OS, in our case it is ubuntu, so we will Login as: Ubuntu



- First Update your system using the command
  sudo apt-get update

- Then use this command in PuTTY to install Apache2
  sudo apt-get install apache2

- Then install php-mysql using the following command
  sudo add-apt-repository -y ppa:ondrej/php
  sudo apt install php5.6 mysql-client php5.6-mysqli

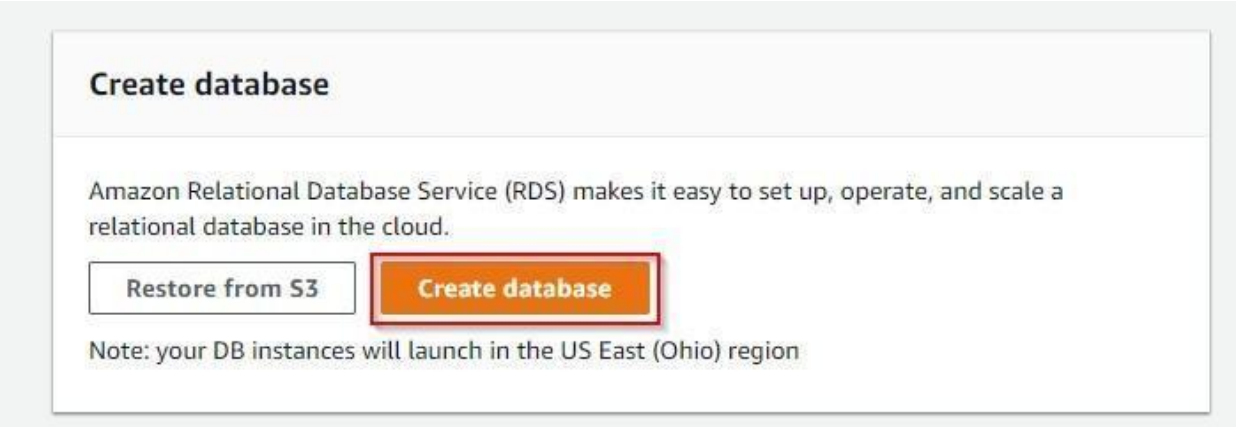
Now everything is updated in your system

```
Creating config file /etc/php/5.6/mods-available/pdo_mysql.ini with new version

Creating config file /etc/php/5.6/mods-available/mysql.ini with new version
Setting up php5.6-json (5.6.38-3+ubuntu18.04.1+deb.sury.org+1) ...

Creating config file /etc/php/5.6/mods-available/json.ini with new version
Setting up mysql-client (5.7.24-0ubuntu0.18.04.1) ...
Setting up php5.6-cli (5.6.38-3+ubuntu18.04.1+deb.sury.org+1) ...
update-alternatives: using /usr/bin/php5.6 to provide /usr/bin/php (php) in auto mode
update-alternatives: using /usr/bin/phar5.6 to provide /usr/bin/phar (phar) in auto mode
update-alternatives: using /usr/bin/phar.phar5.6 to provide /usr/bin/phar.phar (phar.phar) in auto mode

Creating config file /etc/php/5.6/cli/php.ini with new version
Setting up libapache2-mod-php5.6 (5.6.38-3+ubuntu18.04.1+deb.sury.org+1) ...

Creating config file /etc/php/5.6/apache2/php.ini with new version
Module mpm_event disabled.
Enabling module mpm_prefork.
apache2_switch_mpm Switch to prefork
apache2_invoke: Enable module php5.6
Setting up php5.6 (5.6.38-3+ubuntu18.04.1+deb.sury.org+1) ...
Processing triggers for libc-bin (2.27-3ubuntu1) ...
ubuntu@ip-172-31-32-189:~$
```

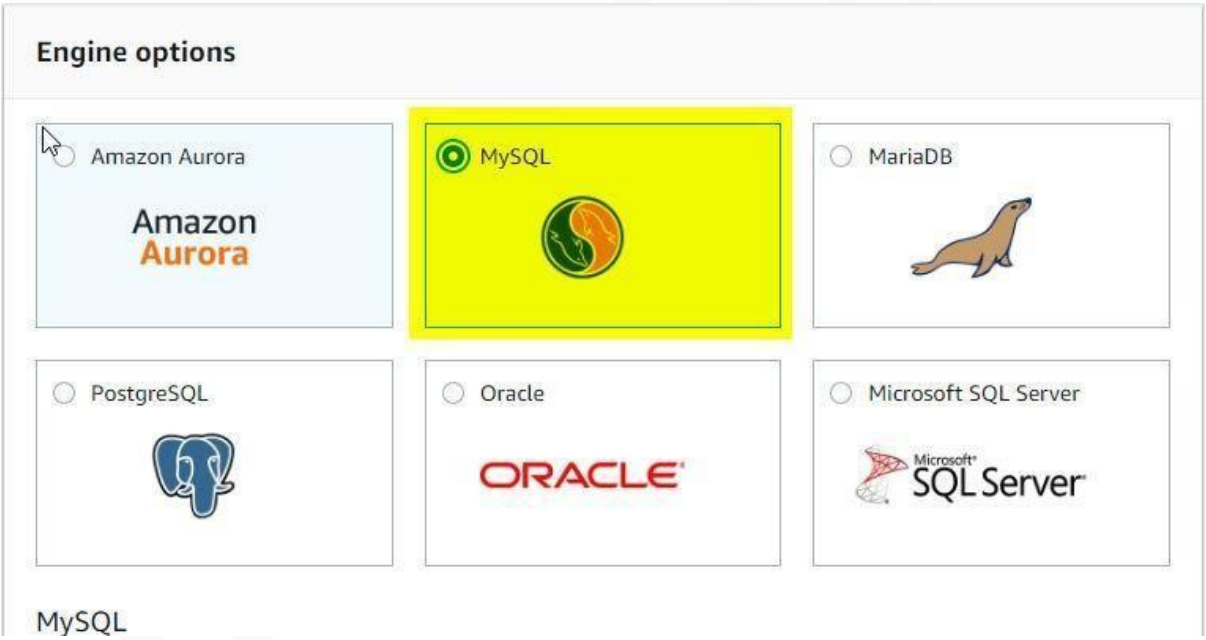
Now we connect mysql with the RDS:

- Go to your AWS Management Console
- Select RDS

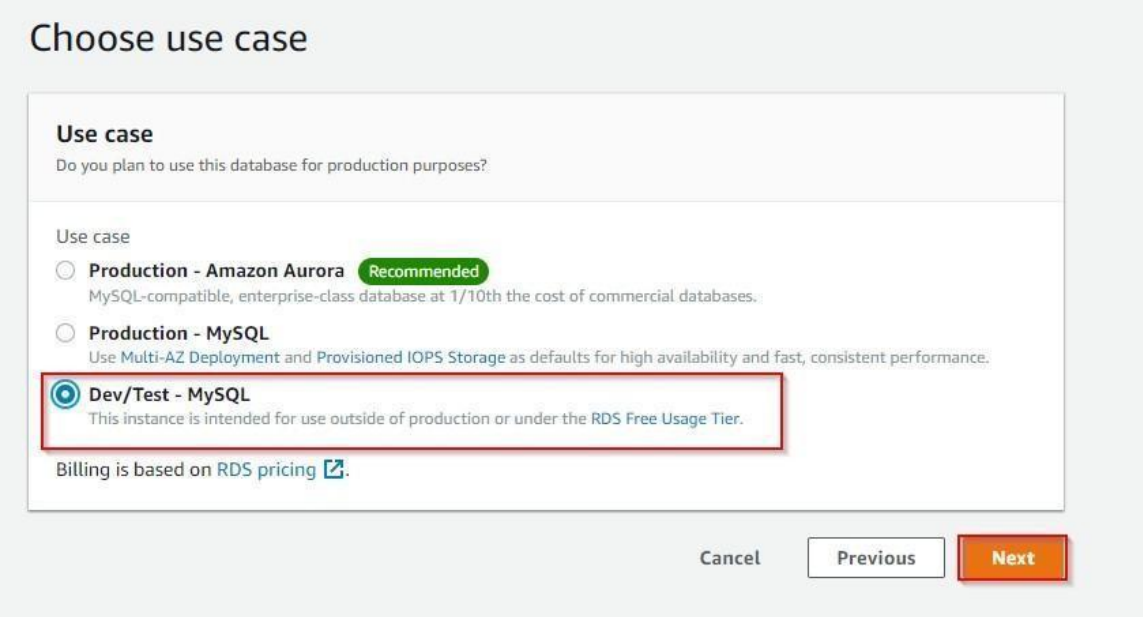
- Then click on Create Database



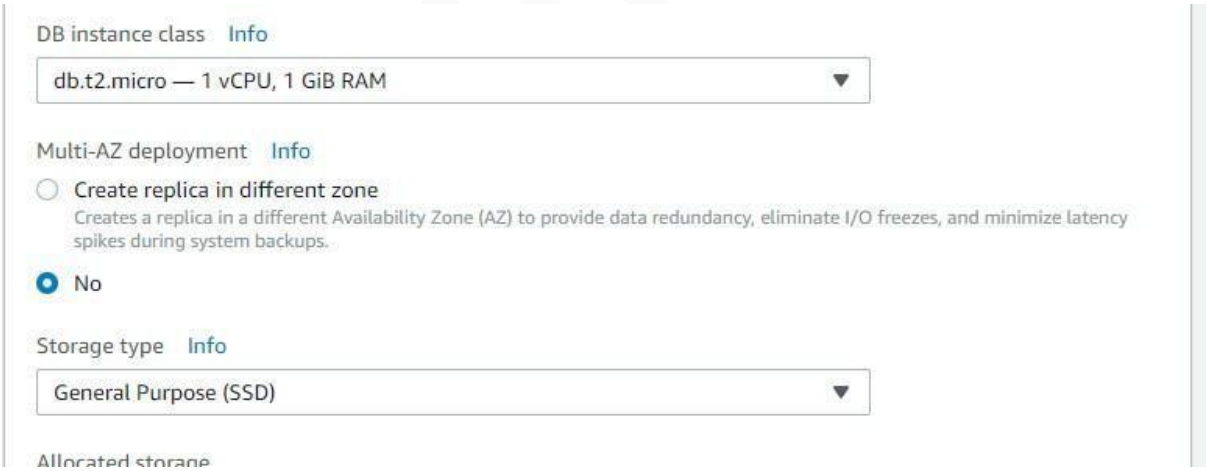
- Select the MySQL Engine and click Next



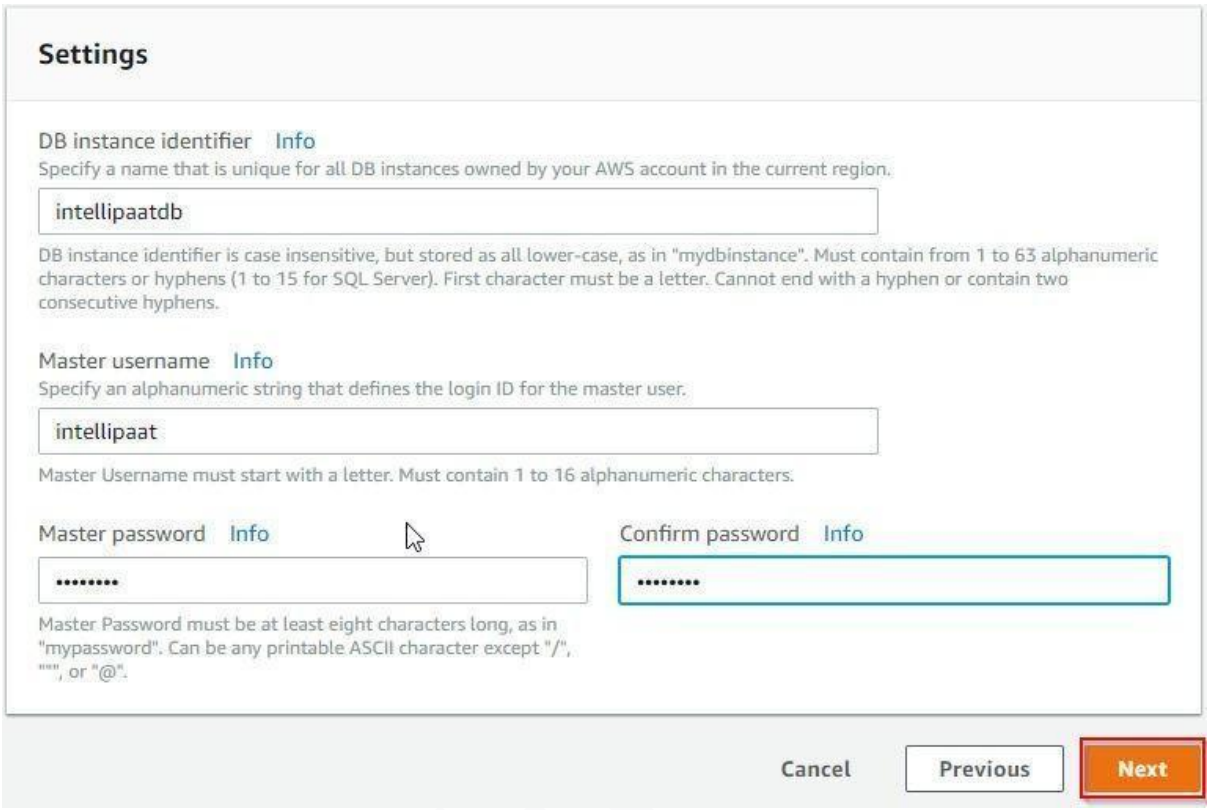
- Since we're using it for the demo purpose, so we'll choose the Dev/Test -MySQL option only and then click Next



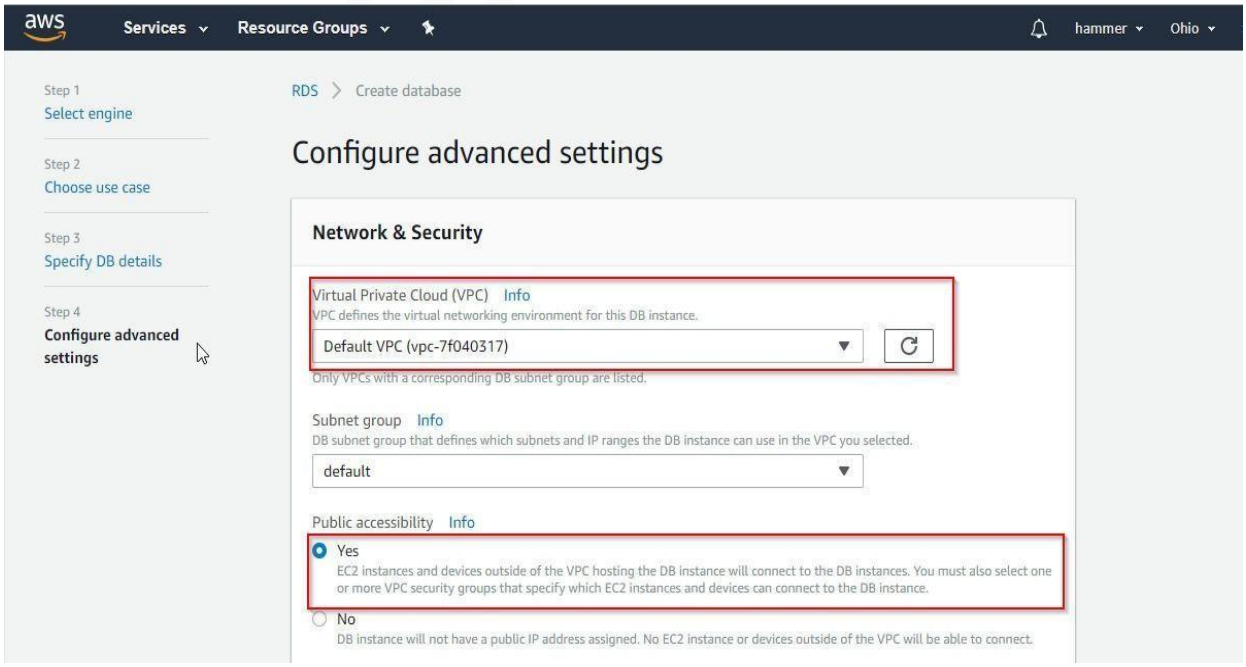
- Specify DB Details, make sure to choose only db.t2.micro in DB Instance Class



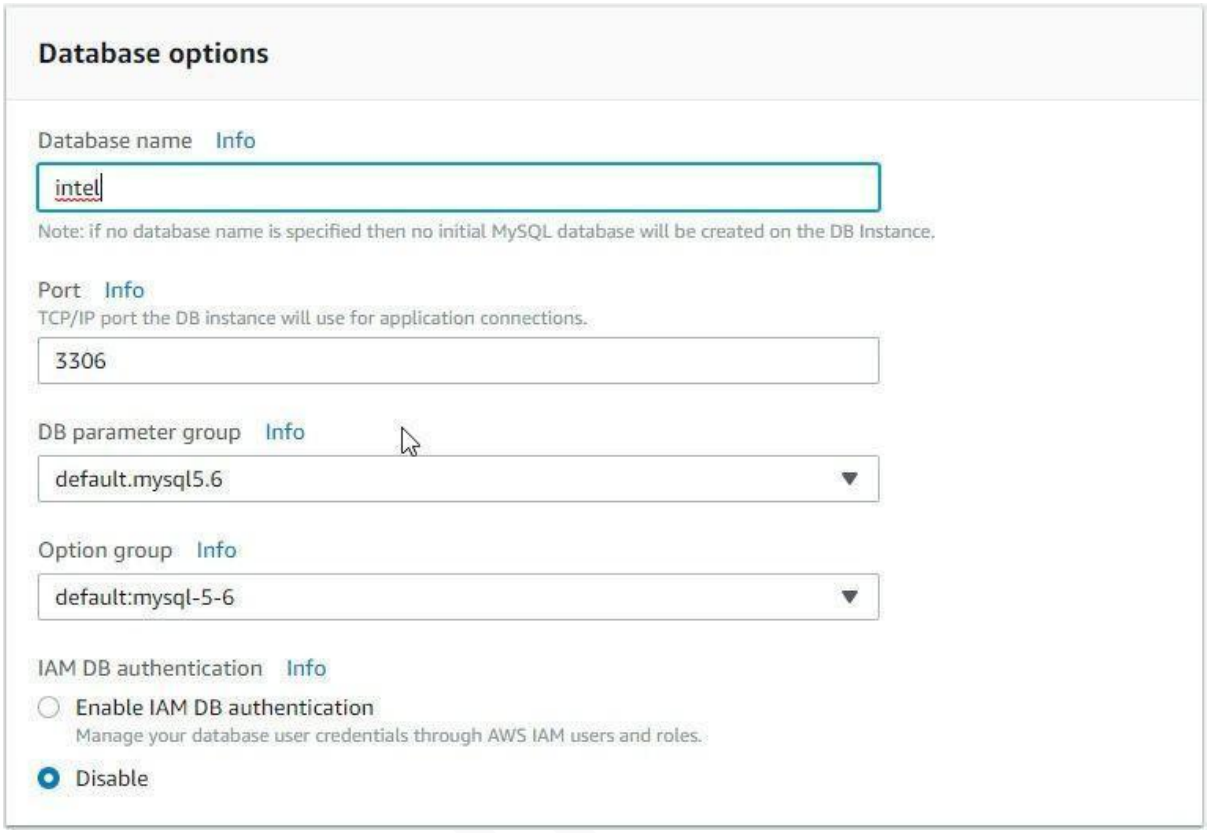
- Enter these credentials (Note: Make sure you remember these credentials, as they will be required for connecting the RDS with your PuTTY



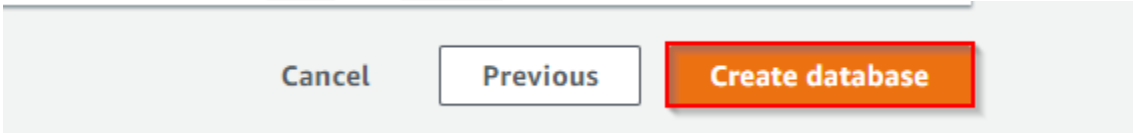
- Then in the Configure Advanced Option, make sure to keep the VPC as default, along with the Public Accessibility as Yes



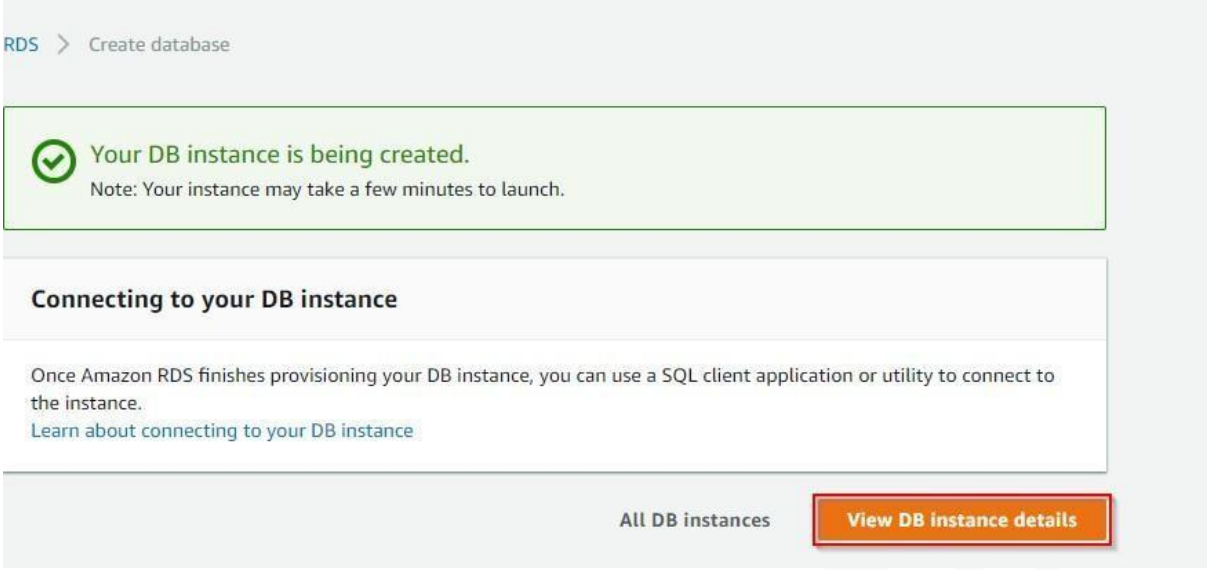
- In the Database Options, name the Database and keep the other artifacts as it is



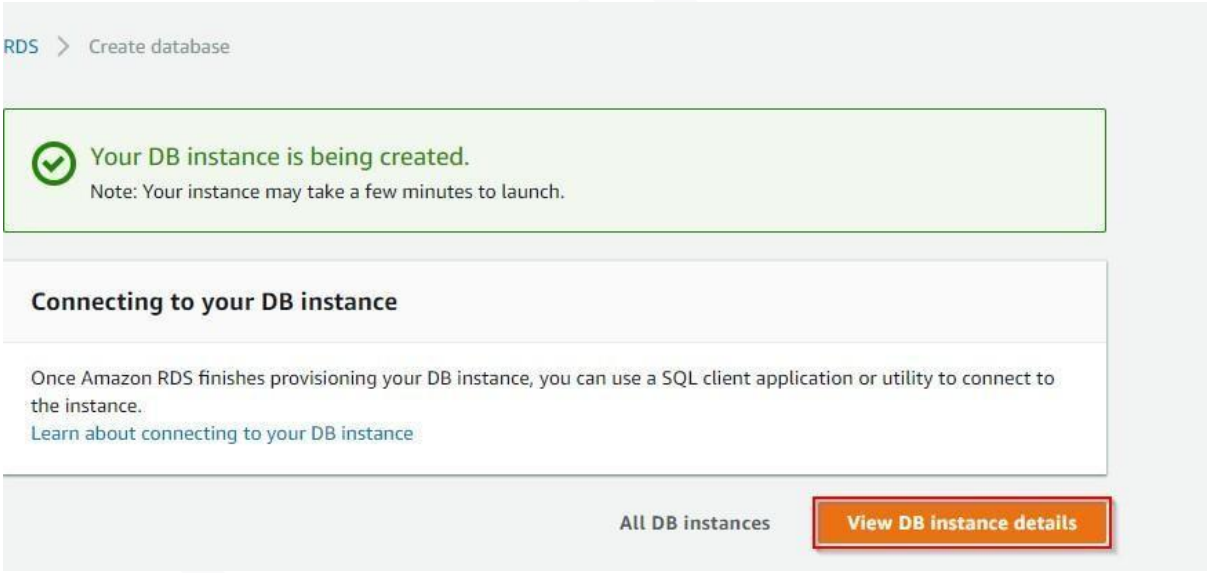
- Then click on Create database



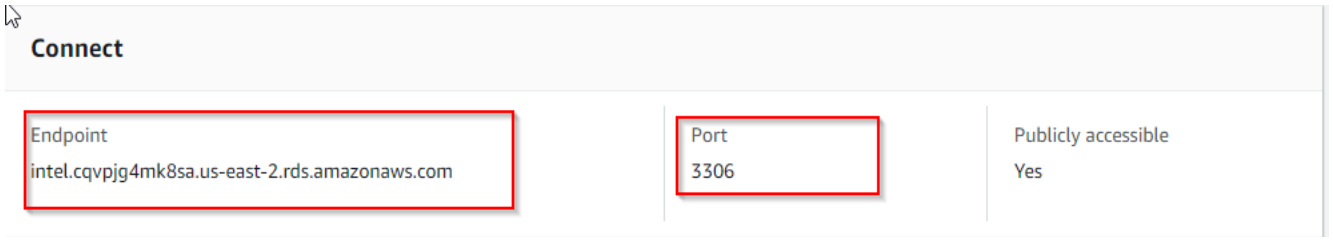
- Then you can check your instance status



- It may take few minutes for RDS to go from Initial to Running stage, you will observe that Endpoint and Port are not yet available (wait for few minutes)



- In few minutes, you will be able to see the Endpoint and Port



- Also, make sure to change some security configuration in the RDS
- Go to your EC2 Instance Security Groups and select your group ID

- Then go to RDS Security groups and select the Inbound rules panel there and click on Add Rule

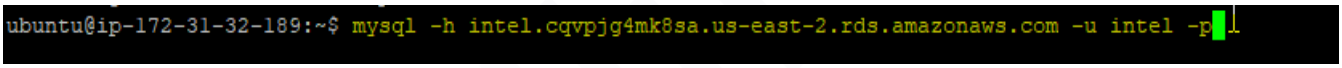
- Then paste the EC2 Security ID in Source> Custom > Security Group by keeping the Type as MYSQL/Aurora

- Now go back to your PuTTY and use this command as shown below

mysql -h hostname -u username -p

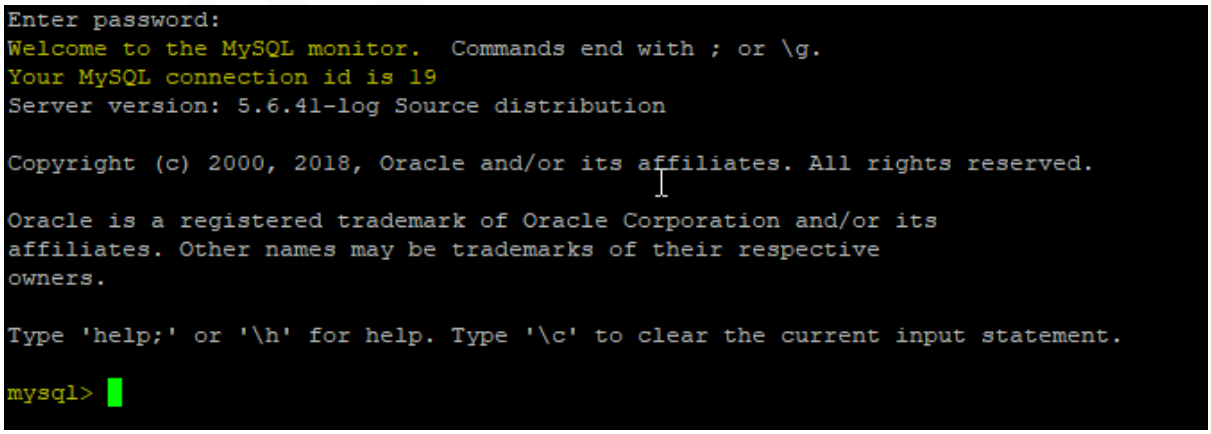
NOTE: In place of hostname, make sure to use your Endpoint from RDS Username which you created

Here, we're using our own Endpoint and username and password used



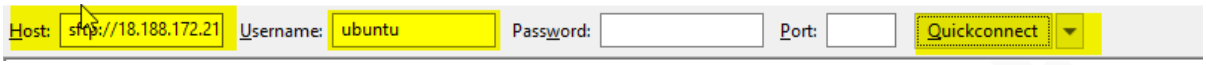
Use the command as shown below:

- After this, it will ask for your password, in our case, password is: intel123
- Then it will show that you're connected to the mysql



Filezilla

- Now install Filezilla
- In order to connect it, enter hostname as the Endpoint of EC2 and Username as Ubuntu and no need to keep the password, then quickconnect



- Now your Filezilla is connected with your EC2 instance
- Create a 'New Folder' of your website in your Desktop
- And copy paste it in your Filezilla Remote Site path: /home/ubuntu

```
ubuntu@ip-172-31-32-189:~$ sudo cp -r New\ folder/ /var/www/html
ubuntu@ip-172-31-32-189:~$ cd /var/www/html
ubuntu@ip-172-31-32-189:/var/www/html$ ls
'New folder'   index.html
```

- Now go back to your PuTTY, where you will see that it contains the index.html file
- Now you need to remove this 'index.html' file and add 'index.php' in its place
- For that you need to use "sudo su" and remove this file using remove command

```
ubuntu@ip-172-31-32-189:/var/www/html$ sudo su
root@ip-172-31-32-189:/var/www/html# rm index.html
root@ip-172-31-32-189:/var/www/html# cd New\ folder/
root@ip-172-31-32-189:/var/www/html/New folder# ls
images  index.php
```

- Also, before running this website, you need to create a table in it (its database)

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| innodb             |
| intel              |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
6 rows in set (0.00 sec)

mysql> use intel
Database changed
mysql> select * from data;
ERROR 1146 (42S02): Table 'intel.data' doesn't exist
mysql> show tables;
```

- Now go to the path where website files are kept and run the index.php file by using sudo nano index.php

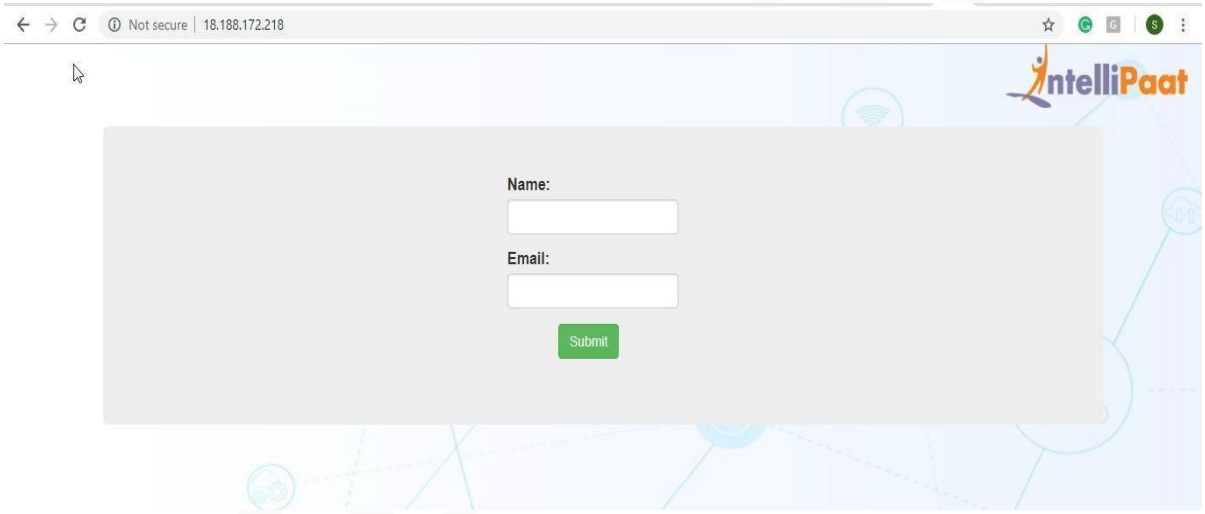
```
ubuntu@ip-172-31-32-189:/var/www/html$ sudo nano index.php
```

- Now after this, GNU nano will pop up where you have to make changes in your code, you have to check if in your server name, the endpoint of your RDS is there along with username, password and db name

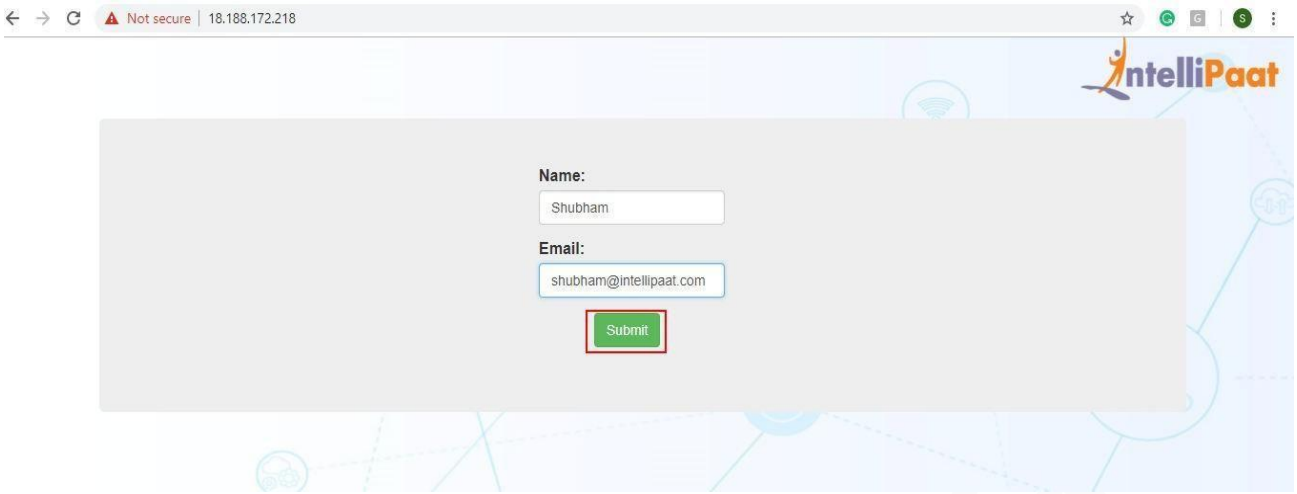
```
                        <butto
</form></td>
  <td colspan="4"></td>
</tr>
</table>
</div>
</div>
<?php
$firstname=$_POST['firstname'];
$email=$_POST['email'];
$servername = "intellipaatdb.cqvpjg4mk8sa.us-east-2.rds.amazonaws.com";
$username = "intellipaat";
$password = "intell23";
$db = "intel";
// Create connection
$conn = new mysqli($servername, $username, $password, $db);

// Check connection
```

- Now when you will try, and copy paste the Public IP of your EC2 Instance

- After copying this IP to your browser, you will observe that your website is working on it

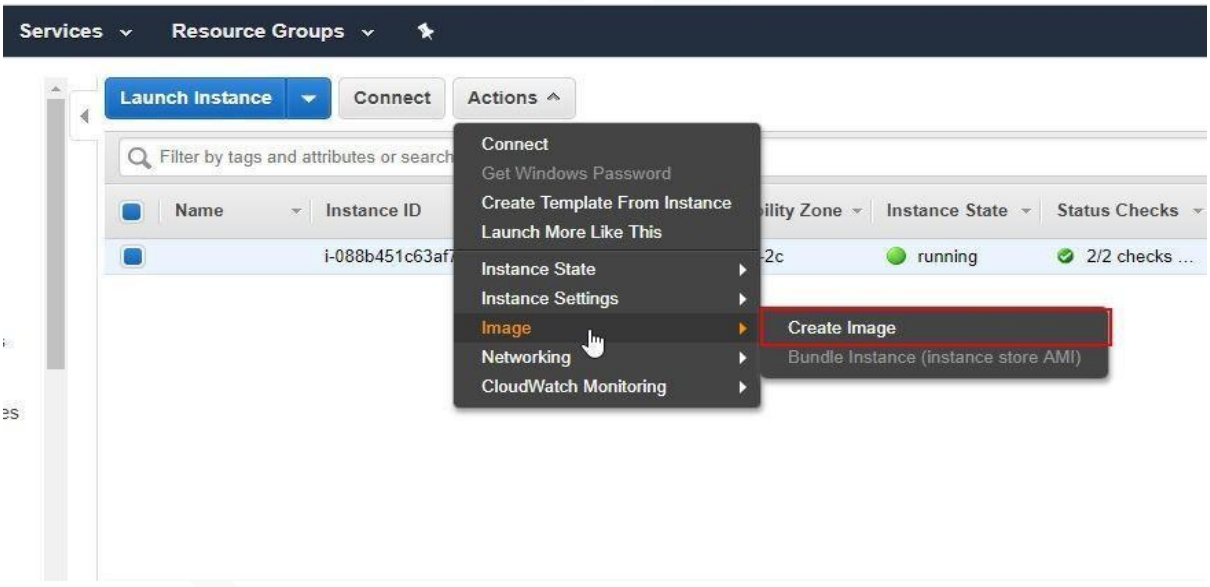


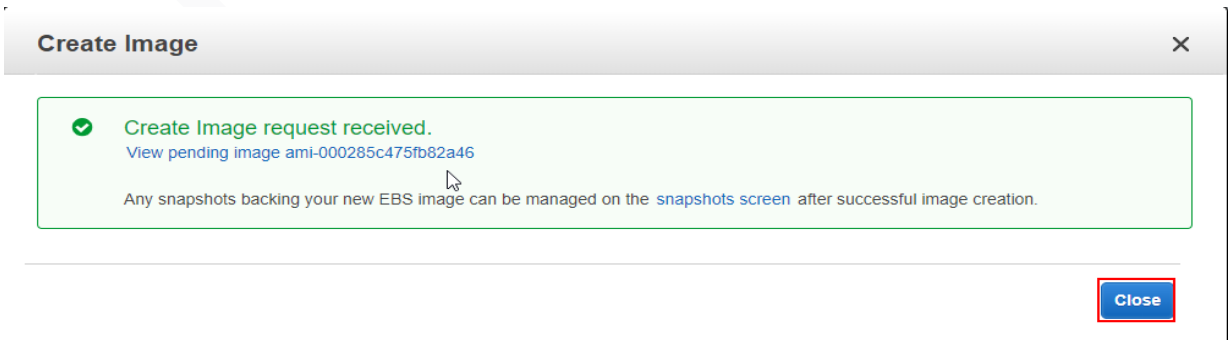
- Now when you enter these details in this website, you will see the following result

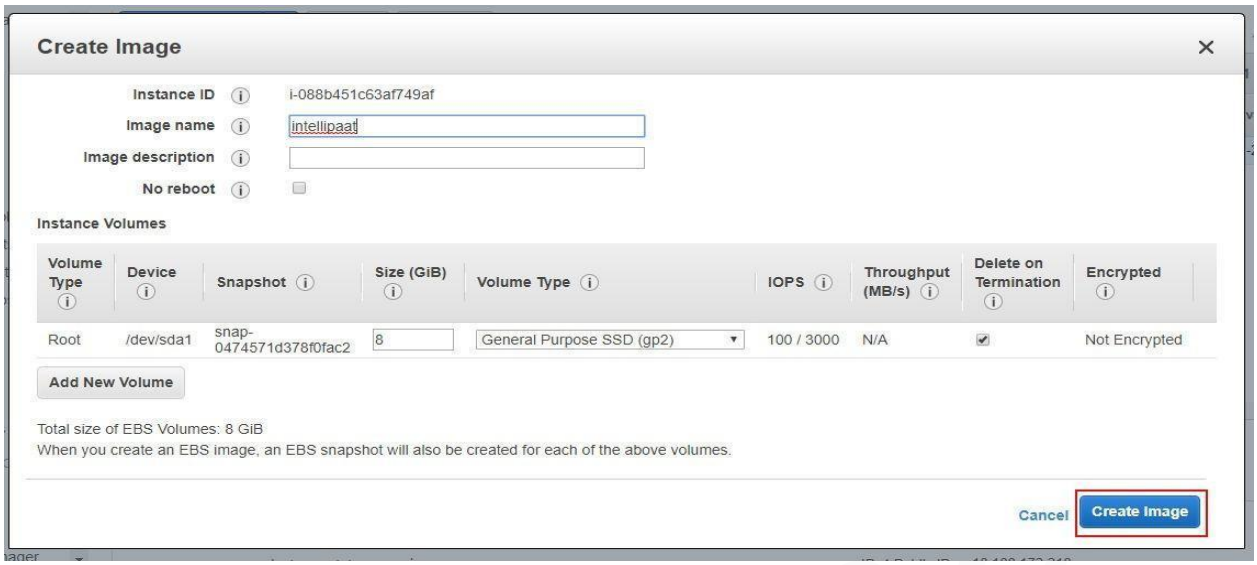


Auto Scaling:

Now, we'll do the autoscaling of our website by going to our EC2 Instance and then click on Actions and Create Image







Then further, activate its autoscaling and then its classic load balancer which directs the traffic to your website directly.