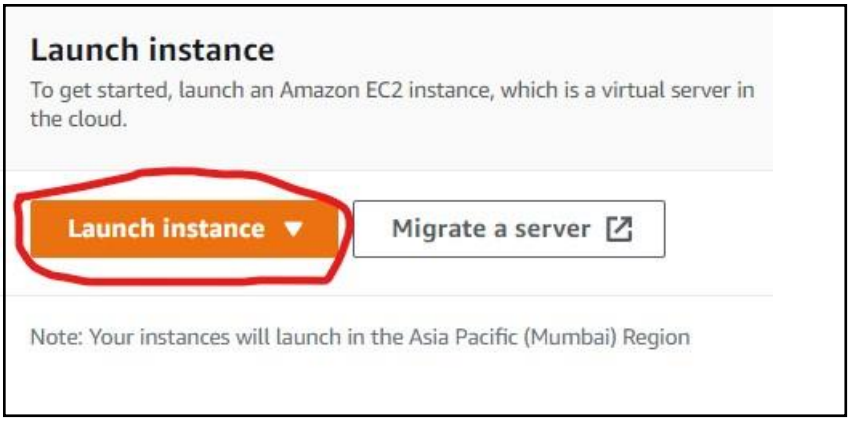
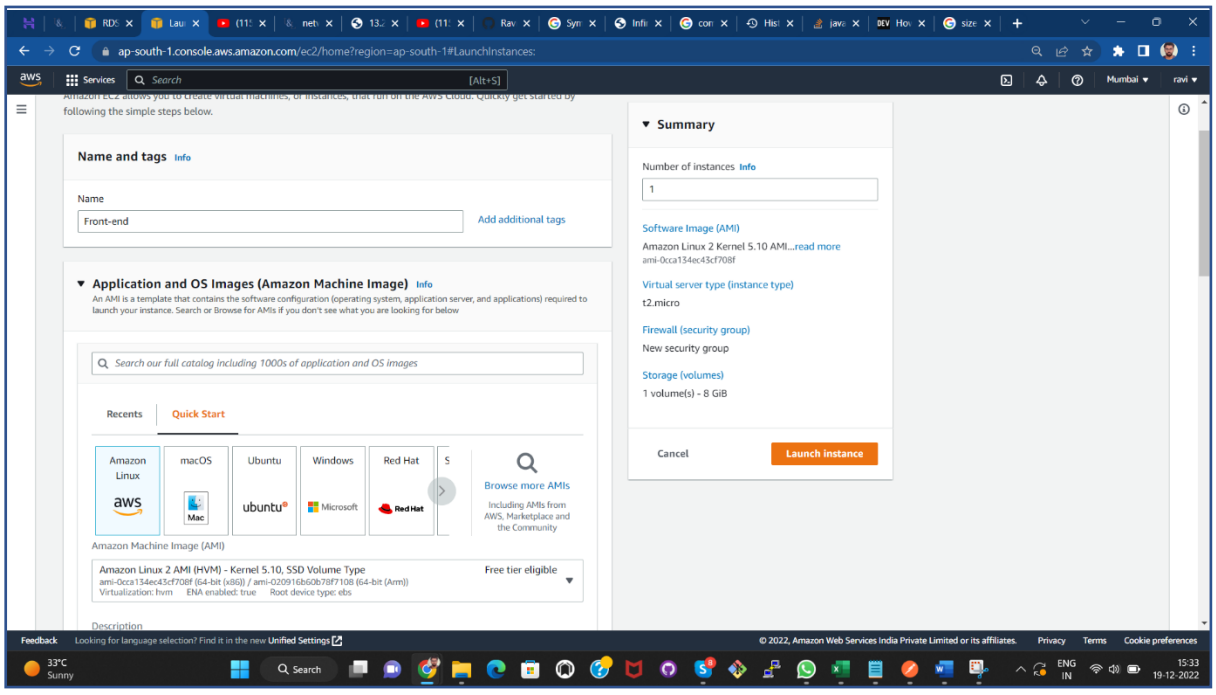
Title: Live Laravel back-end api(serve project) project in Aws Ec2-Instance.

Step1: Open Ec2 and launch instance using option “Launch Instance.”



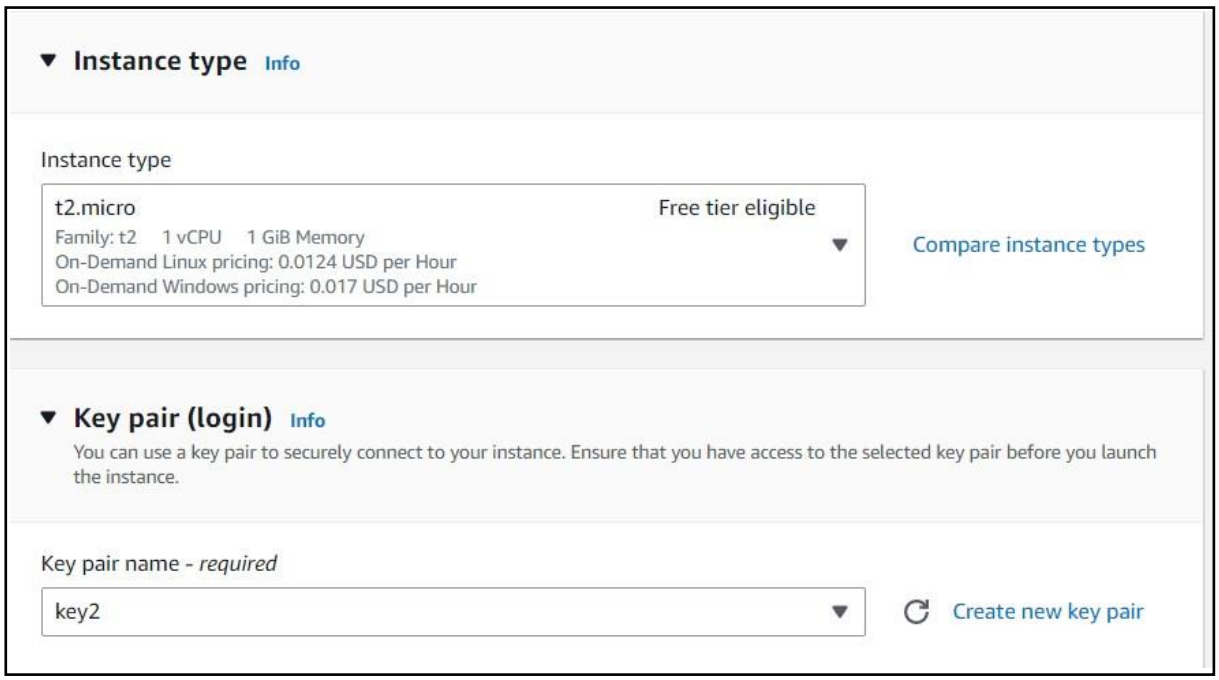
Step2: Give name that instance what you want and select ‘Ubuntu’ OS in “**Application and OS Images (Amazon Machine Image)**” option.



Step3: Select Instance Type (Instance types comprise varying combinations of CPU, memory, storage, and networking capacity and **give you the flexibility to choose the appropriate mix of resources for your applications**.) as per your website requirement, then select “key pair(login)” if you have one or create new one using “create new key pair” option.

- In some project there is t2.small or t2.medium is required, for now set it on t2.micro and later if there is cmd “php artisan db:seed” is not working and hang the process then just stop the ec2 change the instance from

t2.micro to t2.small or t2.medium and start ec2 then apply that cmd it's works fine.

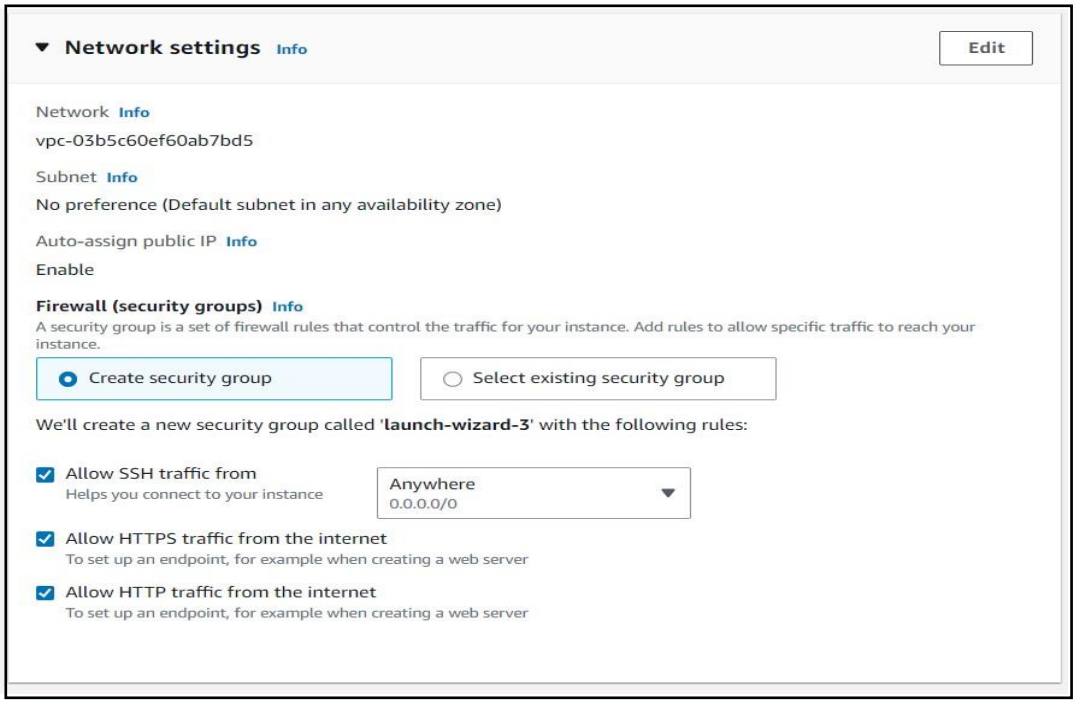


- For how to create keypair see documentation of "Front-end quasar project live" after step3 in it.

Step4:

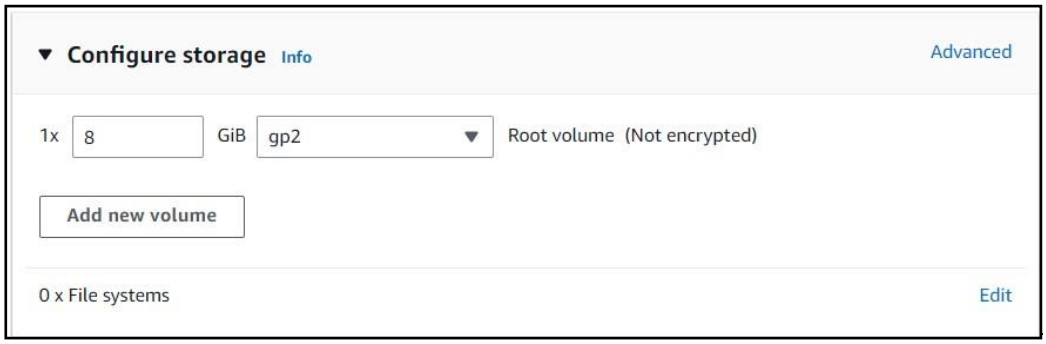
Network setting> Firewall (security groups) select "**Create security group**" if you don't have one or if you already created security group select "**Select existing security group**" and then select the name of security group you created.

For "**Create security group**" option select check-box of "**Allow SSH traffic from, Allow HTTPS traffic from the internet, Allow HTTP traffic from the internet**".



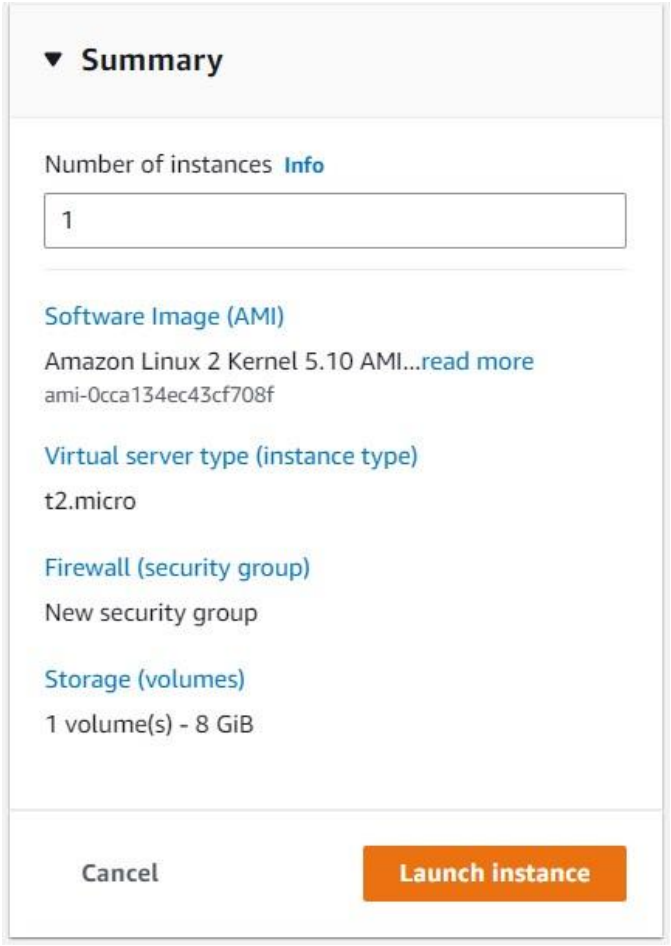
Step5:

In “**Configure Storage”** select size of storage as per limitation of EBS volume type. (Here I take 8gb in **gb2** EBS volume type, which’s limitation is up to 30gb for free tier eligible account.)



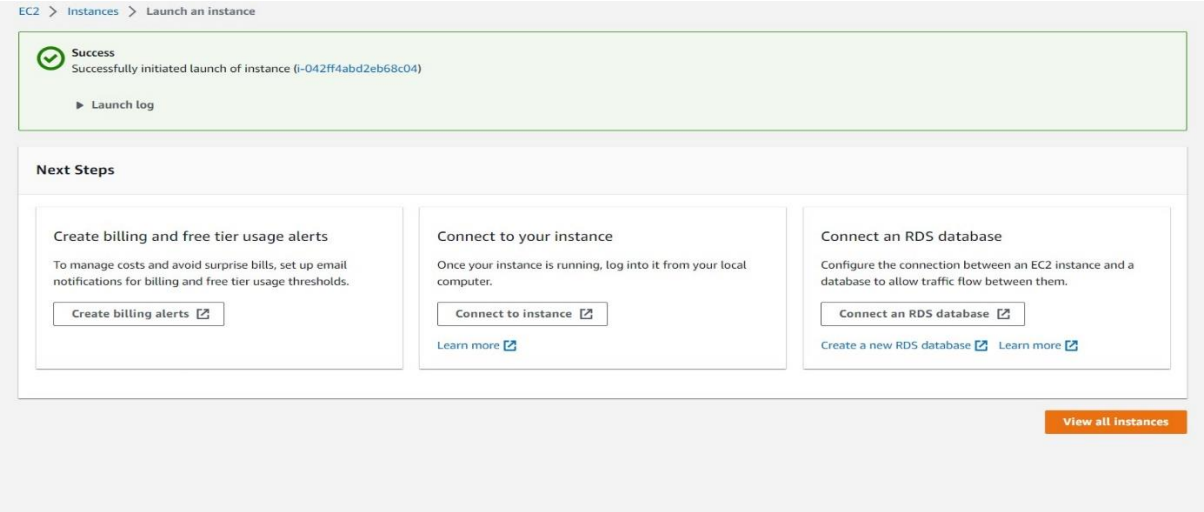
Step6:

Select **“Launch instance”** and launch it.



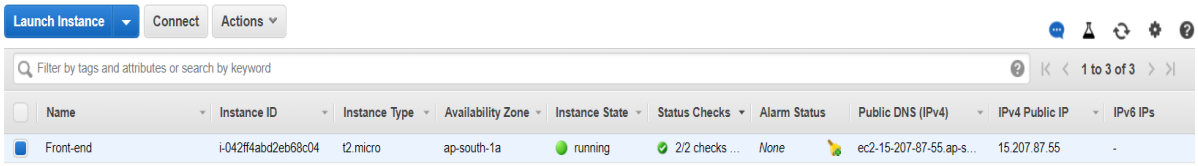
Step7:

Choose “**View all Instances**”



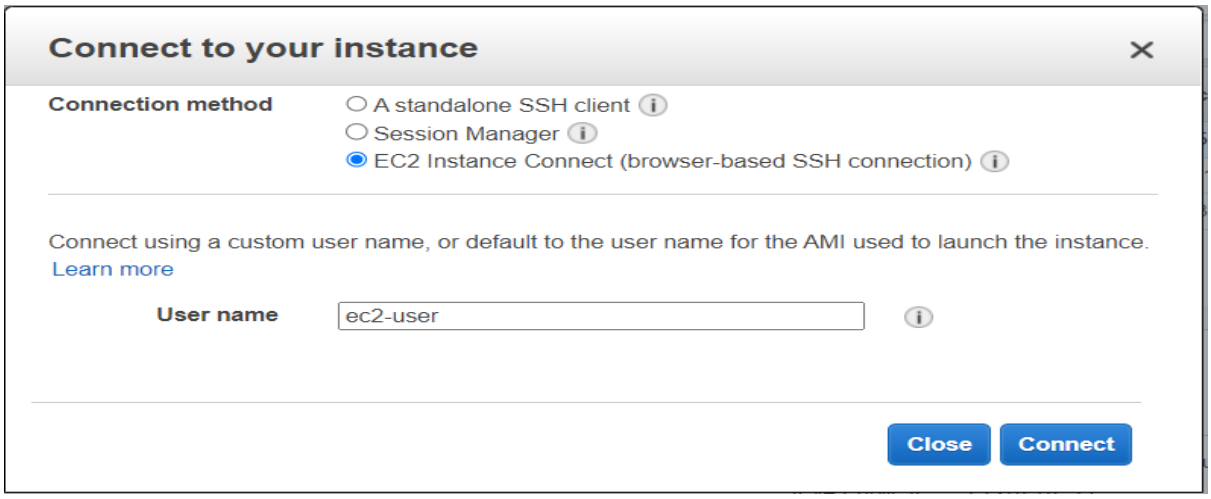
Step8:

Choose Option “**Connect**” after the “**Status Checks**” shows “**2/2checks**”.



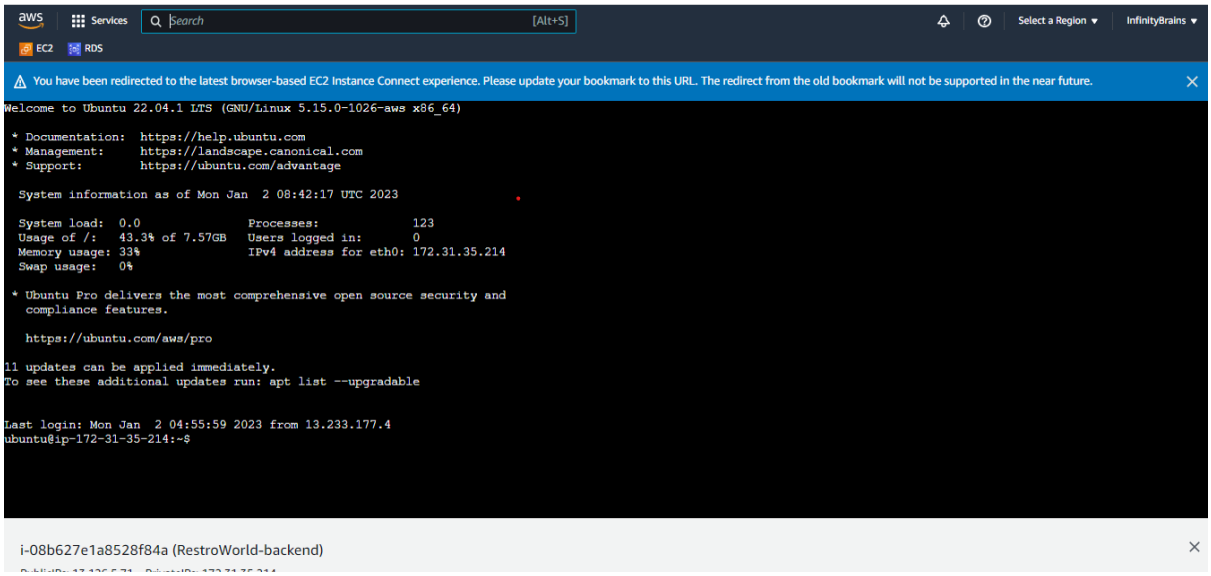
Step9:

Choose “**EC2 Instance Connect (browser-based SSH connection)**” and select “Connect” option.



Step 10:

This type of interface is open.



Now, first give following cmds :

sudo su

sudo apt-get update

- Active git using following cmd:

git init

- Create ssh key using following cmd in terminal

ssh-keygen

or

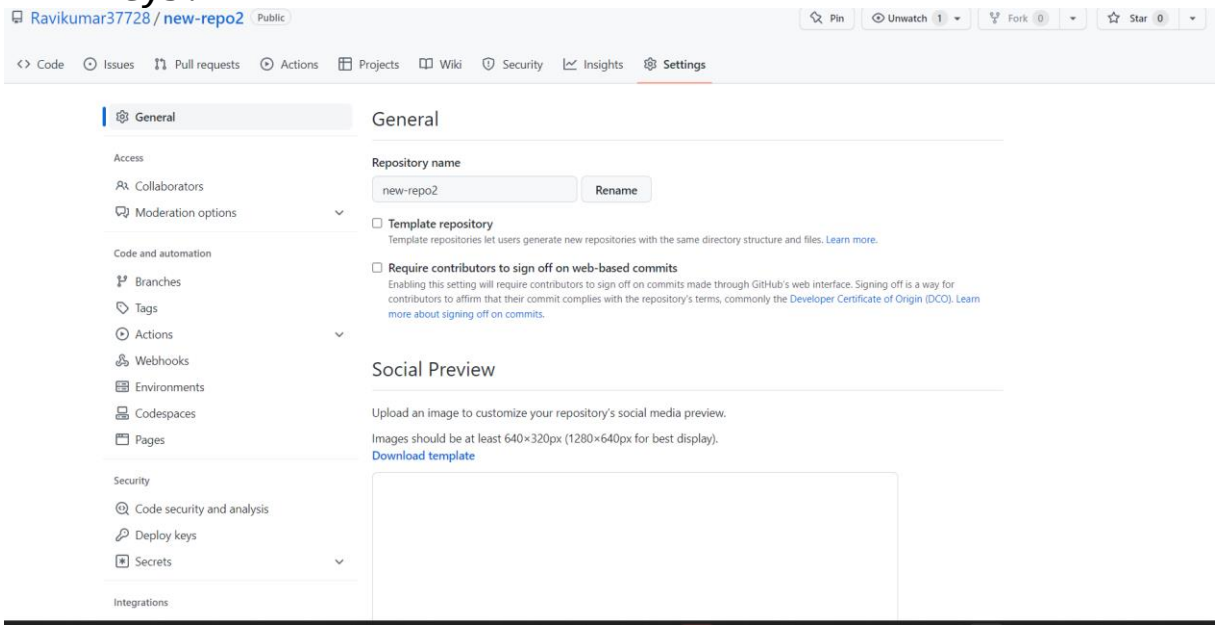
ssh-keygen -t ed25519 -C "mailto:youremailaddress@domain.tld DAY-MONTH-YEAR" -f ~/.ssh/my_key

- Use below shown cmd and then copy “id_rsa.pub” text data which have been shown below:

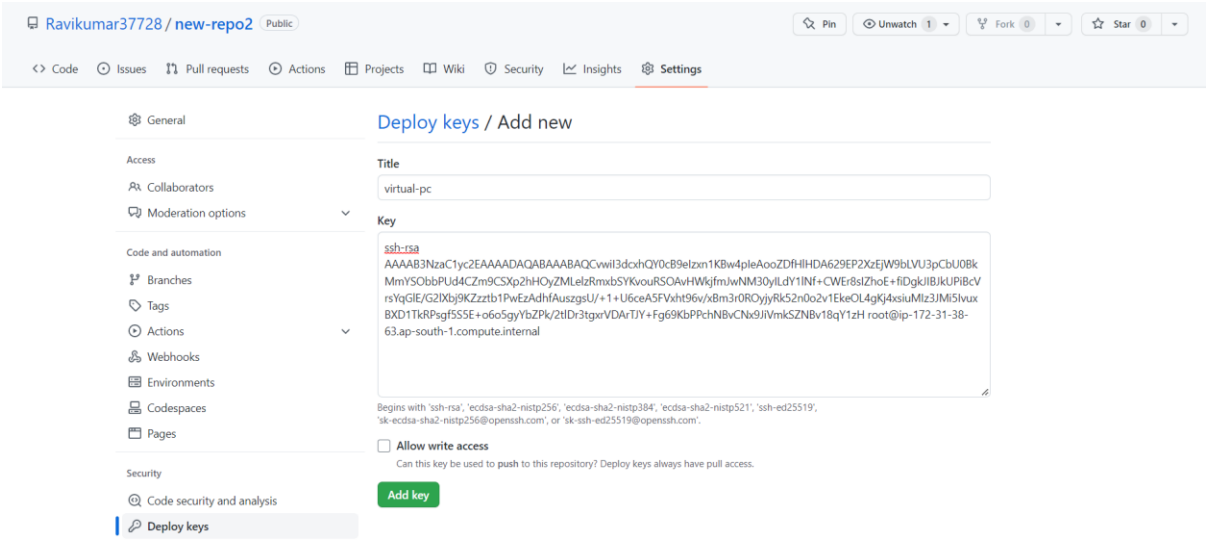
cat ~/.ssh/id_rsa.pub

```
       __|  __|_  )
       _|  (     /   Amazon Linux 2 AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-2/
[ec2-user@ip-172-31-38-63 ~]$ sudo su
[root@ip-172-31-38-63 ec2-user]# ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_rsa.
Your public key has been saved in /root/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:gDPb5H2WZqRzrm6wt3vhx/85mAoyfDvRFC3+HXATbDU root@ip-172-31-38-63.ap-south-1.compute.internal
The key's randomart image is:
+---[RSA 2048]----+
|          . ..E.|
|     .    o o = .|
|    + o o. o + . |
|     B = oo   .  |
|    . = So.. . . |
|    .. *o.. . .  |
|     o+.++   o   |
|    . o=+oo o .. |
|     +=+.o.o..o. |
+----[SHA256]-----+
[root@ip-172-31-38-63 ec2-user]# cat ~/root/.ssh/id_rsa.pub
cat: /root/root/.ssh/id_rsa.pub: No such file or directory
[root@ip-172-31-38-63 ec2-user]# cat ~/.ssh/id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQCvwiI3dcxhQY0cB9eIzxn1KBw4pIeAooZDfHlHDA629EP2XzEjW9bLVU3pCbU0BkMmYSObbPUd4CZm9CSXp2hHOyZMLelzRmxbSYKvouRSOAvHWkjfmJwNM30yILdY1lNf+CWEr8sIZhoE+fiDgkJIBJ
kUPiBcVrsYqGlE/G2lXbj9KZzztb1PwEzAdhfAuszgsU/+1+U6ceA5FVxht96v/xBm3r0R0yjyRk52n0o2v1EkeOL4gKj4xsiuMIz3Mi5IvuxBXD1TkRPsgf5S5E+o6o5gyYbZPk/2tlDr3tgxrVDArTJY+Fg69KbPPchNBvCNx9JiVmkSZNBv18qY1zH
 root@ip-172-31-38-63.ap-south-1.compute.internal
[root@ip-172-31-38-63 ec2-user]#
```

- Paste it in git-hub source code repo.>settings>Deploy Keys.



- Paste copied text in there and named it. Then press “Add key”



Your key is ssh key is now added in git hub repo.

Step 11:

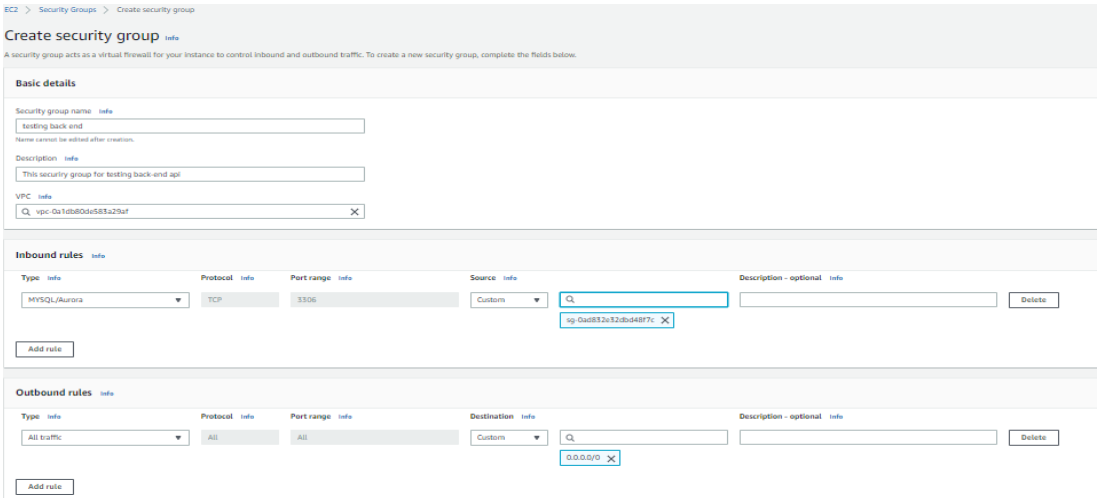
- Give following cmd for activate github repo in ec2.

```
ssh git@github.com
```

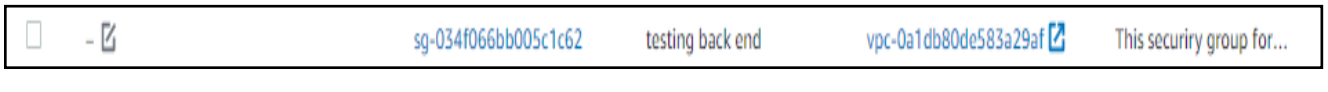
- Type “yes” in connecting
- Now git hub is successfully connected to your ec2 insatnces.

Step 12:

- Search “Security Groups” open it.
- Select “Create Security group” give name it, add description for that if you want it.
- Select “Add rule” in Inbound rules, in Type select “MYSQL/Aurora” don’t change “Source” and after it select that security group, which is used by ec2 backend-api



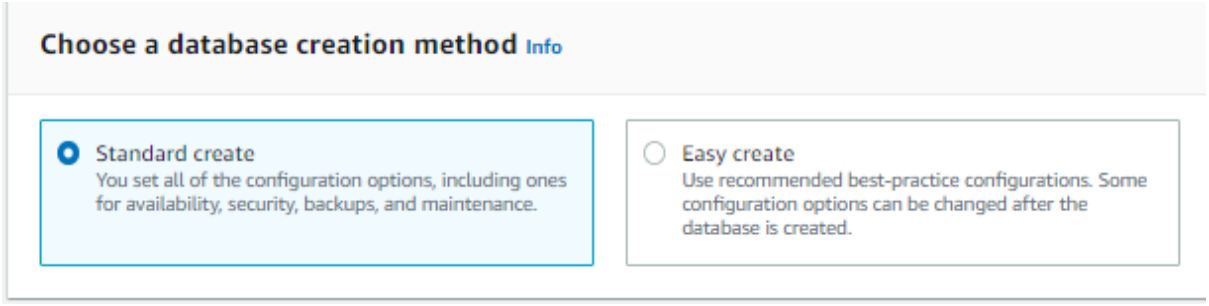
- Then select “Create security group”, your group is created.

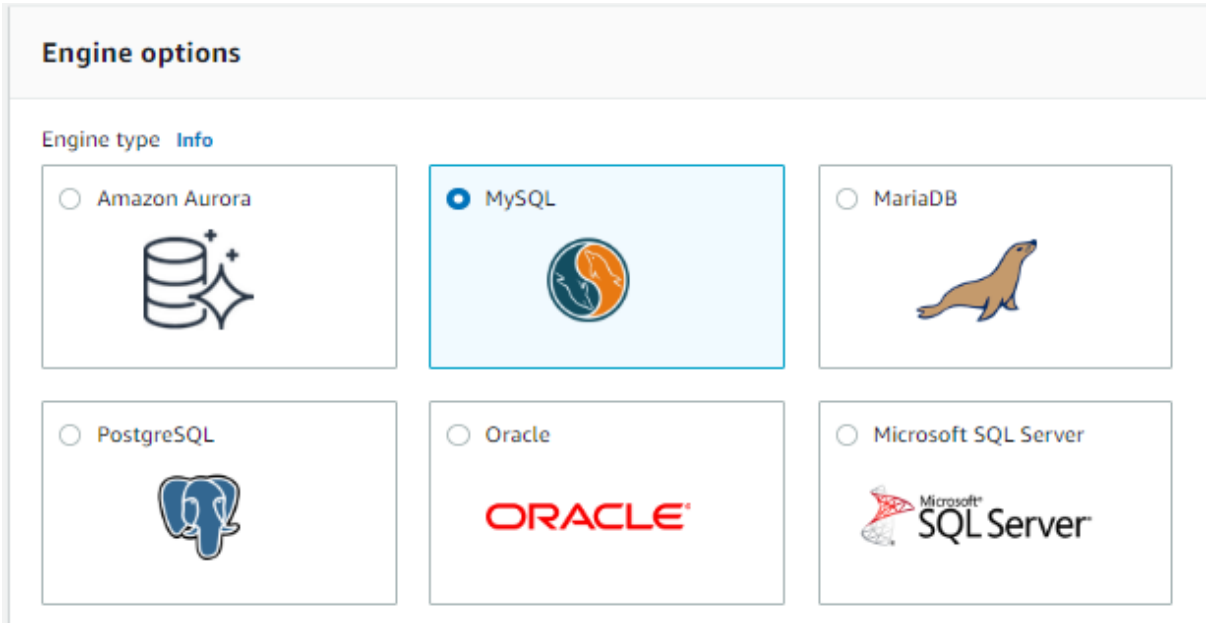


- Now give it name for your ease of remembering it.

Step13:

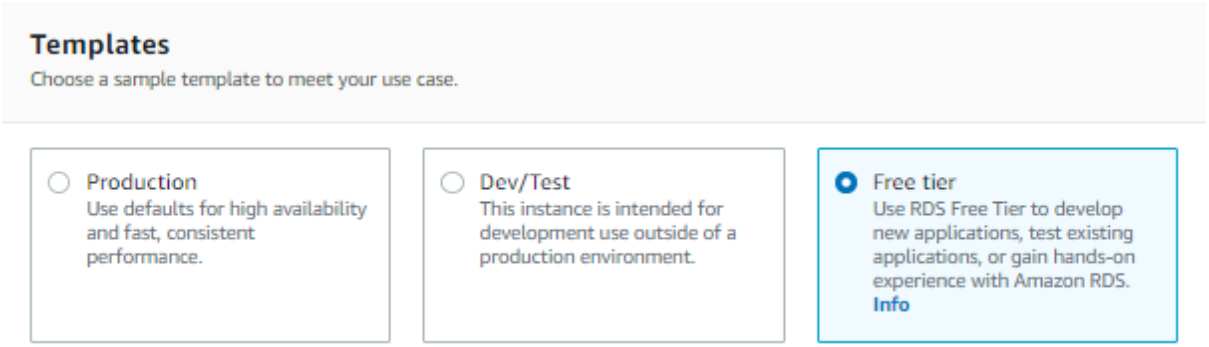
- Search “Rds” and open it, select “Create database”.
- Select “Standard create” in first option, then select “MySQL” in Engine Option.



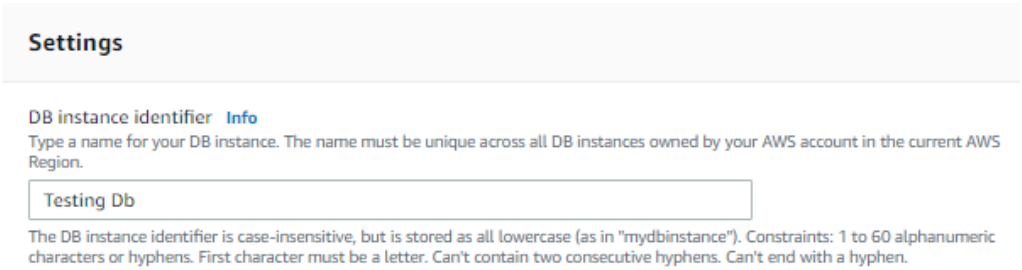


- In “Engine Version” Select latest version of it, in my case it’s “MySQL 8.0.31”

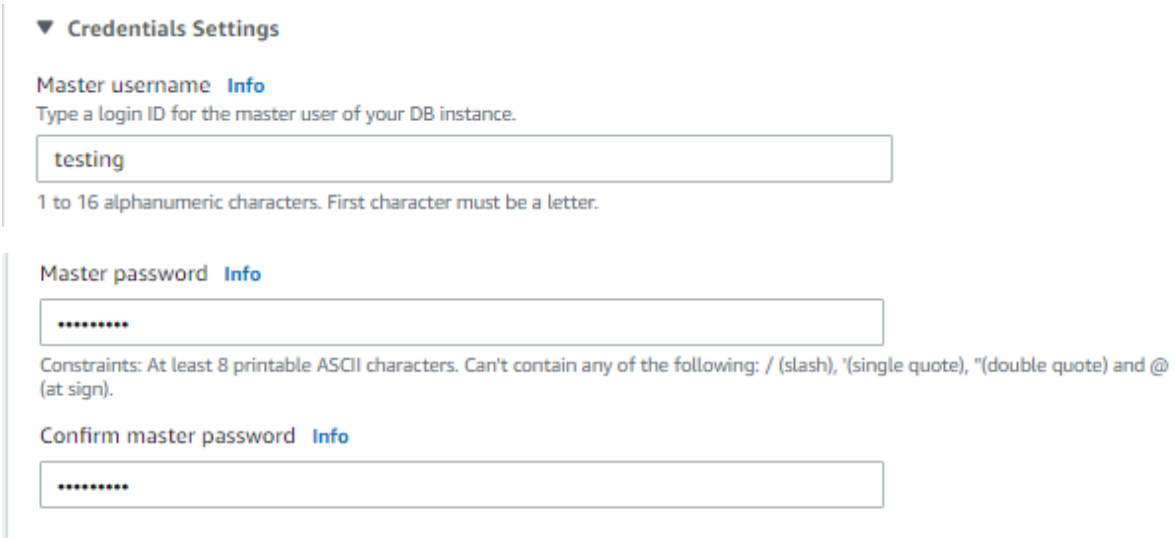
- In Templates select “Free tier”.



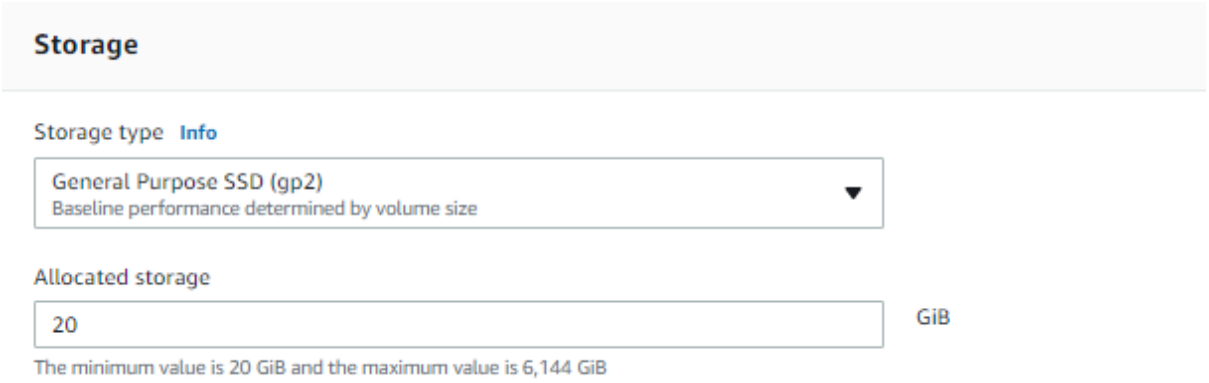
- In “Settings” give name to the database in option “DB instance identifier”.



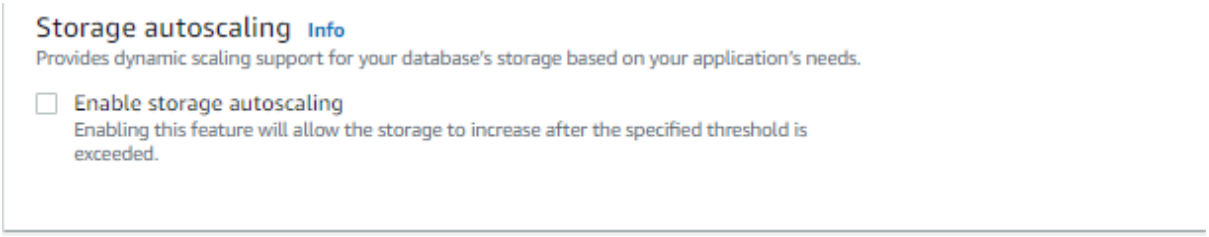
- Give “Master username” and “Master password” also note it down somewhere because we need it later.



- Change “DB instance class” if it’s required for project or leave it as it is.
- Change “Storage” according to your project requirement for me its 20Gib.

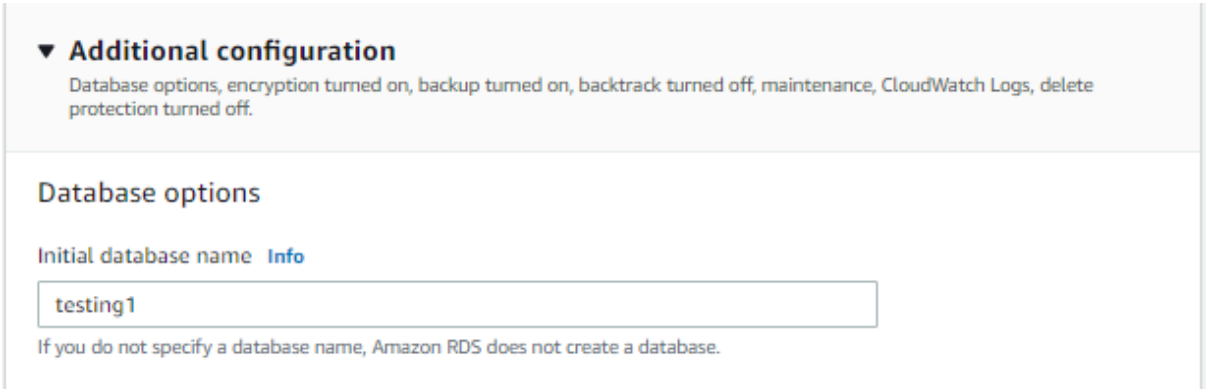


- Unchecked “Storage autoscaling”.

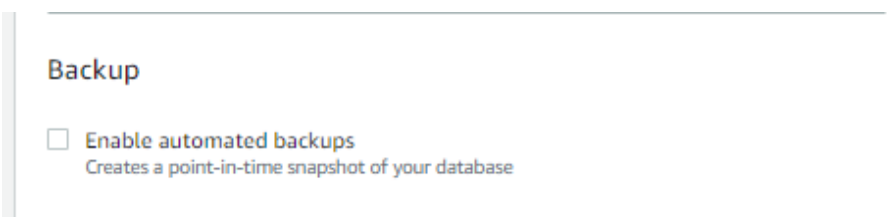


- Select security group which one you created on step12 in “Existing VPC Security groups” after removing “default” security group.

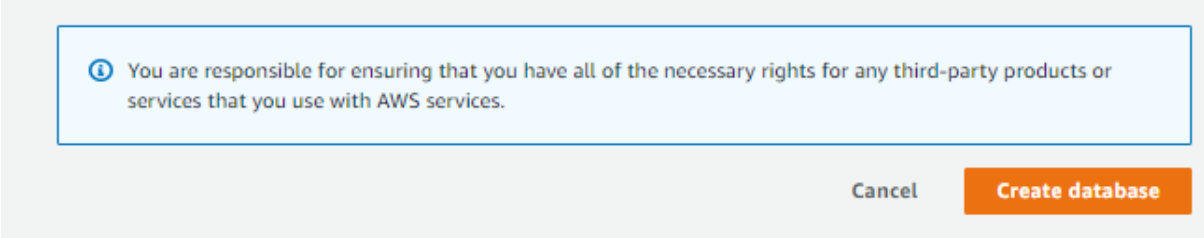
- In “Additional configuration” give Initial Database name and also note it down somewhere for later it will be required.



- If you want to store automated backup of your database then don’t unchecked “Backup”, in my case I Unchecked it.



- Now, select “Create database” your data-base in Rds is created.

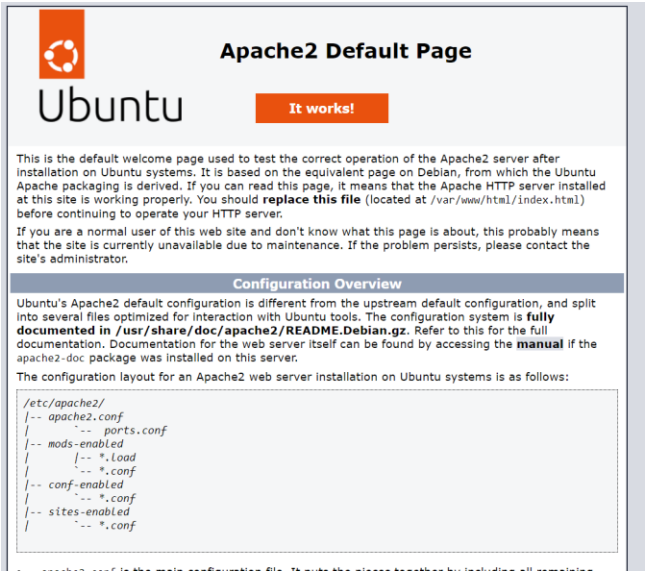


Step 14:Import Project in ec2 and connect it with Rds.

- Open Your ec2 instance
- Fire following cmds:

```
sudo su

sudo apt-get update
sudo apt-get install apache2
sudo service apache2 restart
sudo apt-get install mysql-server
sudo apt-get install php8.1 libapache2-mod-php8.1 php8.1-mysql

sudo add-apt-repository ppa:ondrej/php8.1
sudo apt-get install php8.1-mcrypt

sudo curl -O https://getcomposer.org/composer.phar
sudo mv composer.phar composer
sudo chmod +x composer
sudo mv composer /usr/local/bin
composer
sudo apt-get install php8.1-mbstring
sudo apt-get install php8.1-xml
sudo apt-get install zip unzip
composer clearcache

cd /var/www/html
```

- Run your IPv4 address in browser this type of interface is shown.



- Now in your ec2 instance remove index.html file from location /var/www/html and import your working project there by follow following cmds:

  cd /var/www/html

  rm -rf index.html

  git init

  git remote add origin (ssh link)
  e.g.
  git remote add origin git@github.com:Ravikumar37728/new-repo2.git

- Now use following cmd for pull codes in ec2 on path var/www/html.

  git pull origin Master

  or

  git pull origin main

  NOTE: main or master or other branch is depended upon which branch code you want to pull from connected git-hub repo.

- Check your code is there or not by using cmd:
  ls or ls -la
- Now give following cmd for installing and updating composer

  composer install --ignore-platform-reqs

  composer update --ignore-platform-reqs

- Change and add data in 000-default.conf for open it, use following cmd:

  sudo nano /etc/apache2/sites-available/000-default.conf

- Change "DocumentRoot" /var/www/html to "/var/www/html/index.php" or in some project it will be "/var/www/html/public" use one of it, by which one of it project is running (generally second one path is used when in postman,uploaded img's url gives blank screen in browser.)
- Also add given below data just above the </VirtualHost> in 000-default.conf.

<Directory "/var/www/html">

    Options Indexes FollowSymLinks MultiViews

    AllowOverride All

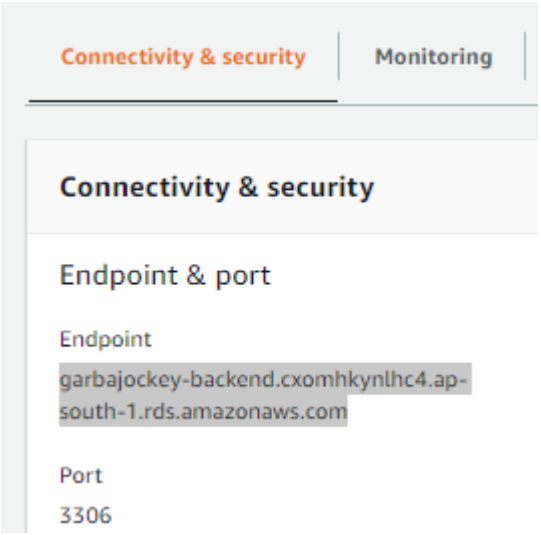
    Require all granted

</Directory>

- After it restart apache 2 using cmd:
  sudo service apache2 restart
- Rename .env.example and named it .env using cmd:
  mv .env.example .env
- Fire following cmd:

  php artisan cach:clear
  php artisan config:clear
  php artisan key:generate

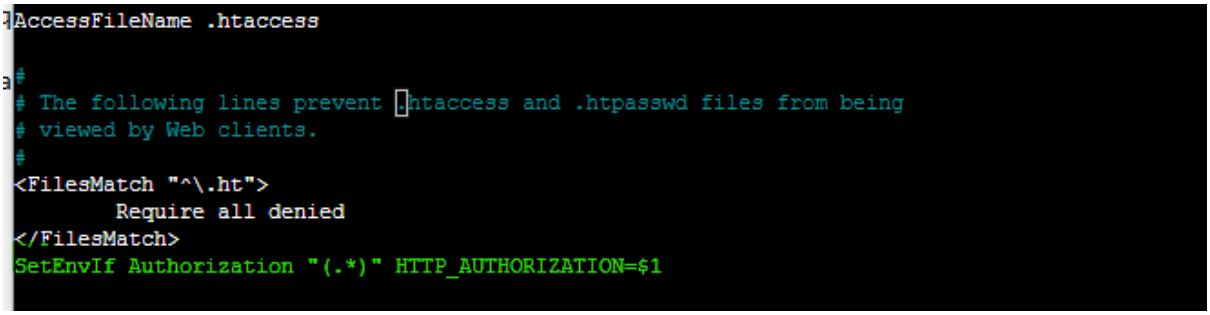
- Give read-wright permission to the storage and bootstrap
  sudo chmod 777 -R bootstrap/
  sudo chmod 777 -R storage/
  sudo service apache2 restart
- Open .env file and enter DB_HOST, DB_DATABASE, DB_USERNAME, DB_PASSWORD of Rds in there.
- For DB_HOST ,open rds click on your db and open it init in "Connectivity & Security" copy the "Endpoint" link and paste it in DB_Host



- In DB_DATABASE paste name of data base which we save previously in step13,in DB_USERNAME and DB_PASSWORD give Master Username and Master Password which we save previously in step13,and save it. then again one time restart apache2.
- Fire following cmd:

  nano /etc/php/8.1/apache2/php.ini

- Change three things in there by finding it in file using Ctrl+Q .
  upload_max_filesize = 512M
  post_max_size = 558M
  max_execution_time = 300
- Now fire following cmds:
  php artisan migrate:fresh
  php artisan passport:install
  php artisan key:generate
  php artisan storage:link
  php artisan route:cache
  php artisan route:clear
  php artisan db:seed
- Now, add one line in apache2.conf file for that fire cmd:
  sudo nano /etc/apache2/apache2.conf
- Find .htaccess using Ctrl+Q and add given below line on that place which is shown in given figure.
  SetEnvIf Authorization "(.*)" HTTP_AUTHORIZATION=$1



- Restart your apache2
  sudo service apache2 restart
- Now run your IPv4 address if its redirect to Laravel page it's means it is live and give path it to api (e.g. 65.24.3.16/api or in some case it is 65.24.3.16/index.php/api) if its redirect to api that means it is working fine.