Installation of EBER

1). Purchasing Server On AWS

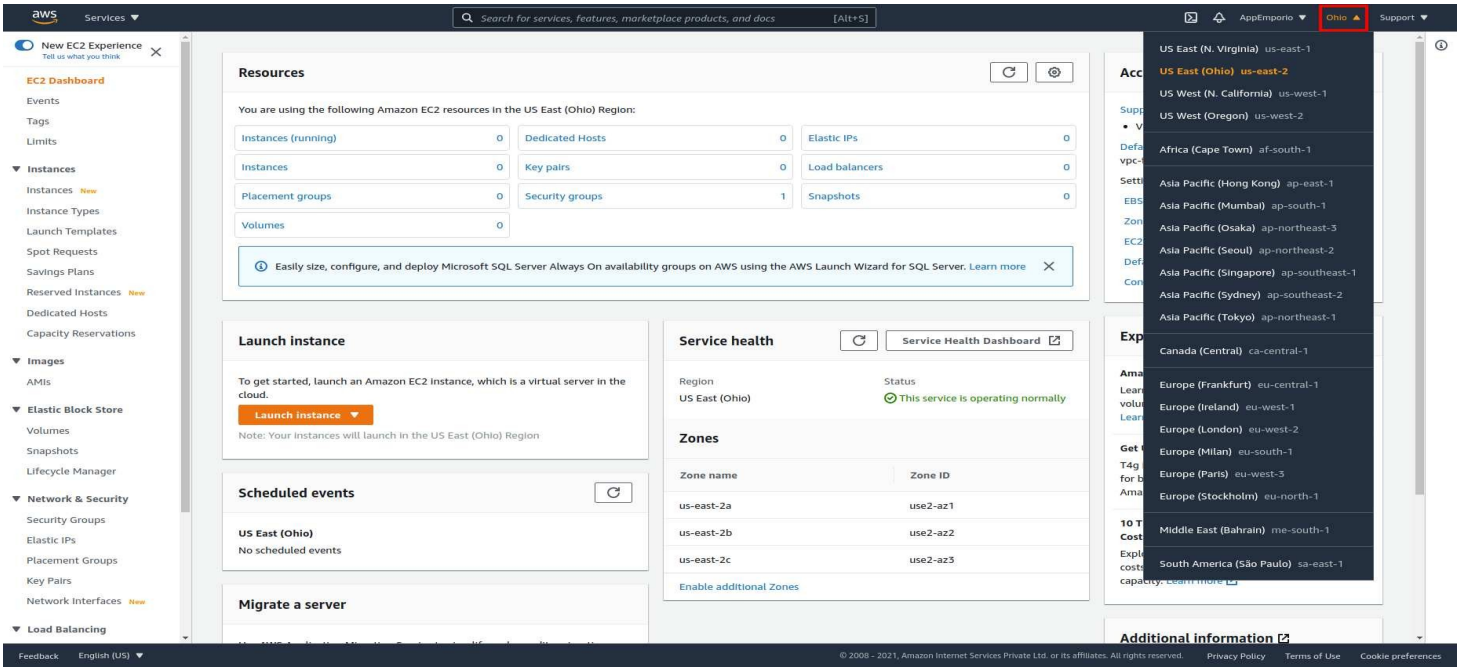
Create an account on AWS and complete basic registration steps that are required to get done before we can access AWS benefits.

After registration now we need to purchase an instance(server) where we can install our code. So for that, we need to purchase instances

steps to purchase instance

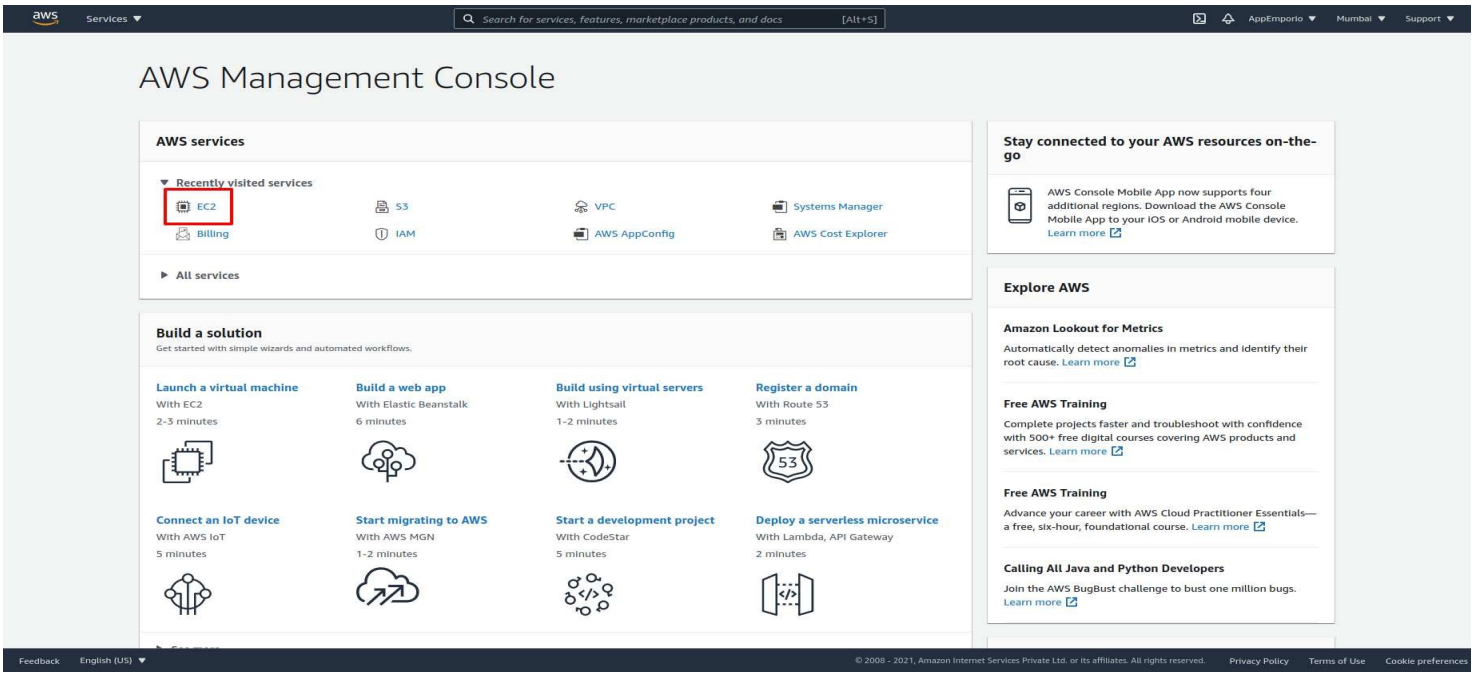
1.1 Selecting region for instance

- We need to select the region (from the top right corner) wisely so that we get the best latency and API responses in less time. Select the region where this application is going to be used most of the time.
- If a region is not available in a particular country or state, then use the region which is nearest.

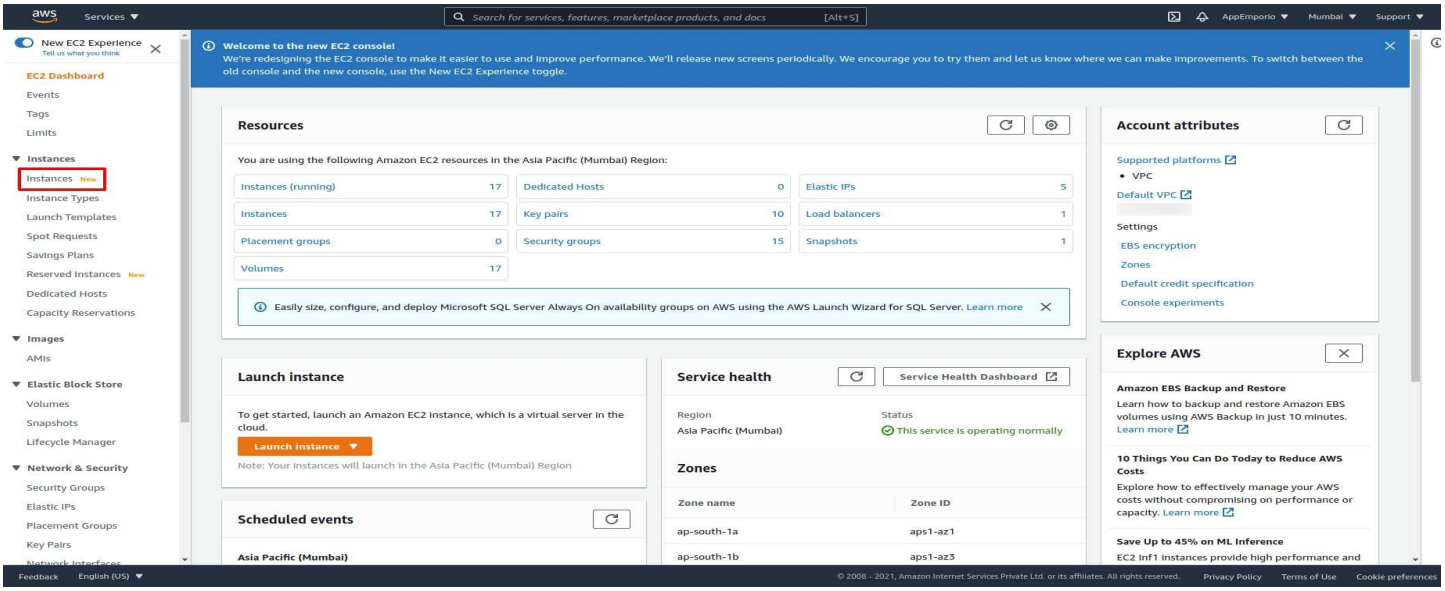


1.2 Purchasing Instance

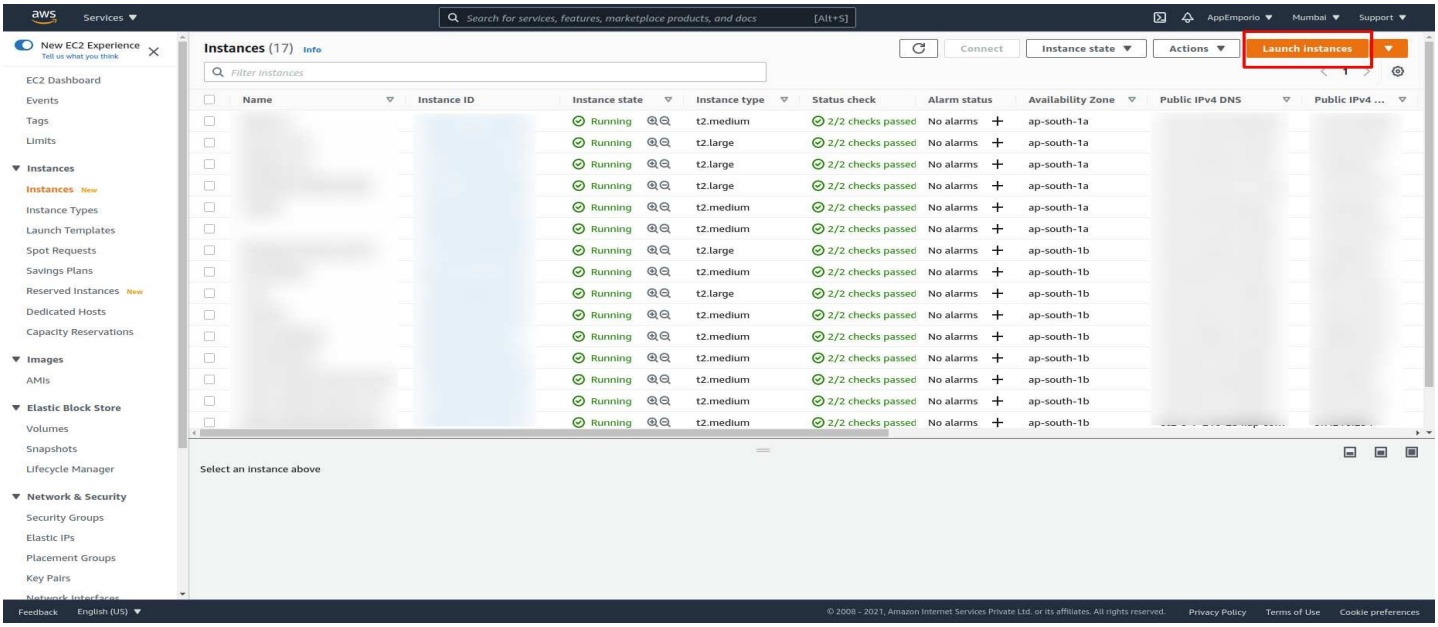
1.2.1: From AWS services select EC2.



1.2.2: Now, from the side navigation bar select instances.



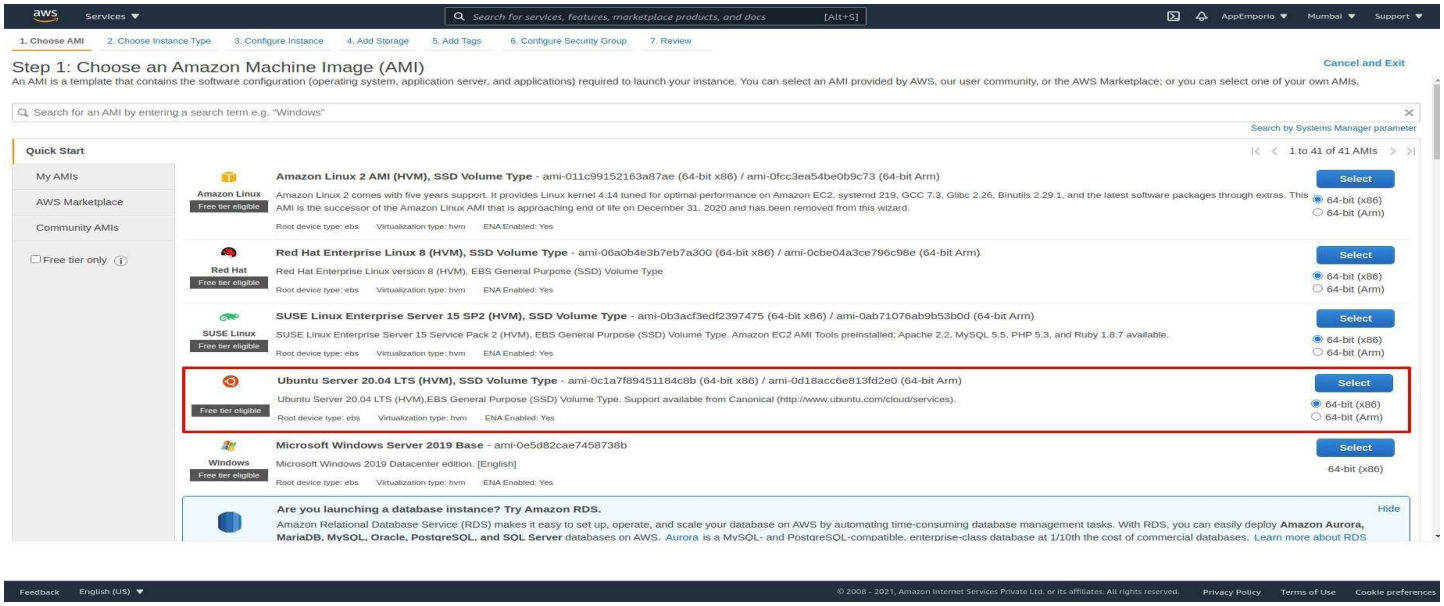
1.2.3: Select Launch from the top right corner.



1.2.4: Now we few steps to follow for launching our instance

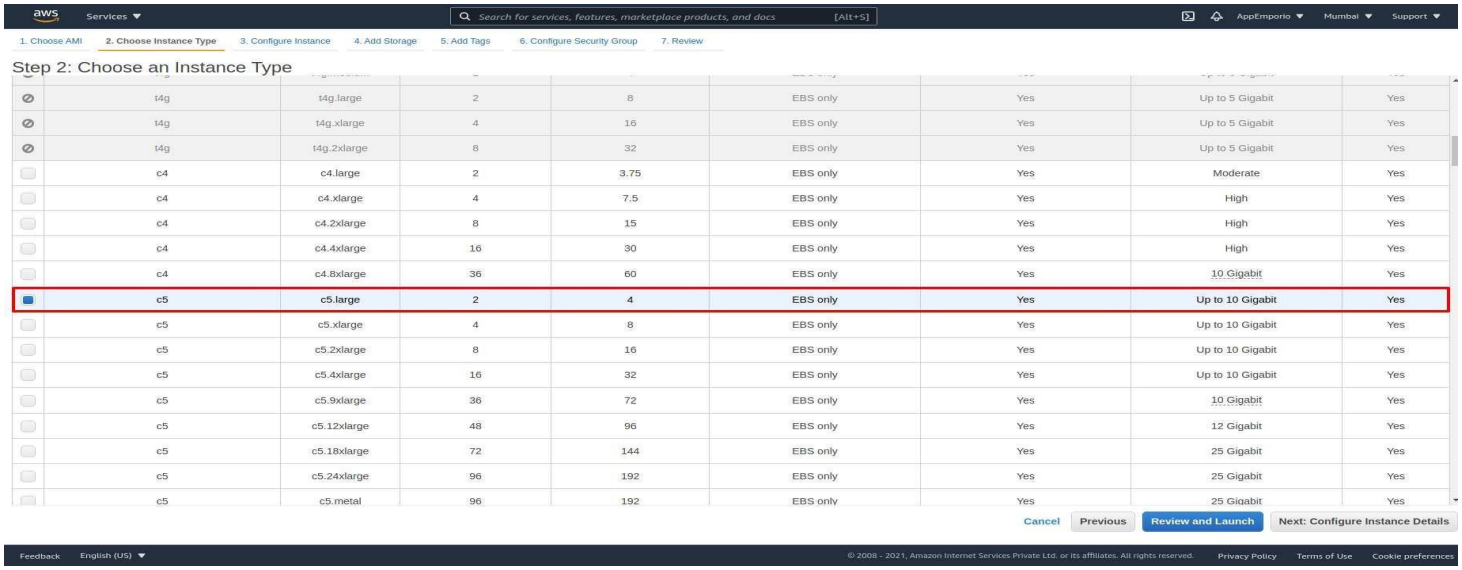
1.2.4(i) Choose an AMI

- AMI is a bunch of basic preloaded software configurations like Operating Systems.
- Select Ubuntu which is most preferable according to our use case.



1.2.4(ii) Choose instance type

- Amazon EC2 provides a wide selection of instance types optimized to fit different use cases. They have varying combinations of CPU, memory, storage, and networking capacity, and give you the flexibility to choose the appropriate mix of resources for your applications.
- Preferable selection c5.large for backend

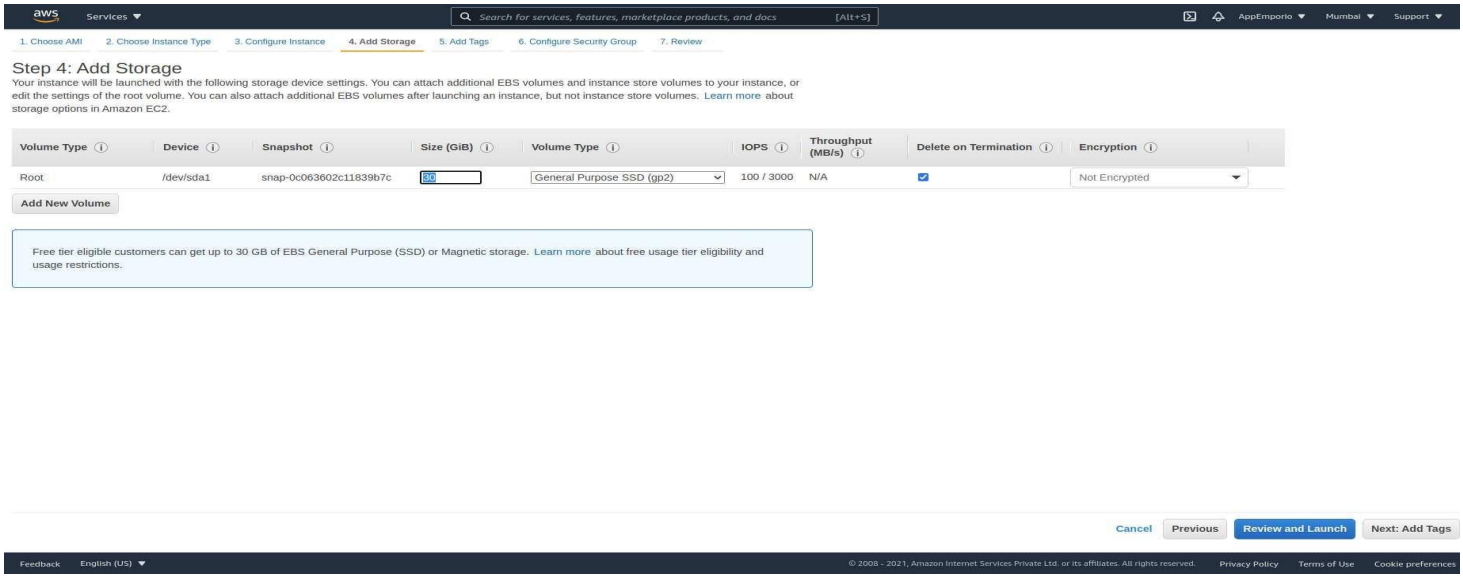


1.2.4(iii) Configure instance details

- Configure the instance to suit your requirements. You can launch multiple instances from the same AMI, request Spot instances to take advantage of the lower pricing, assign an access management role to the instance, and more.
- Configure your server according to your requirements. Let's head to the next step.

1.2.4(iv) Add storage

- Your instance will be launched with the following storage device settings. You can attach additional EBS volumes and instance store volumes to your instance, or edit the settings of the root volume. You can also attach additional EBS volumes after launching an instance, but not instance store volumes.
- Select 30 GiB.

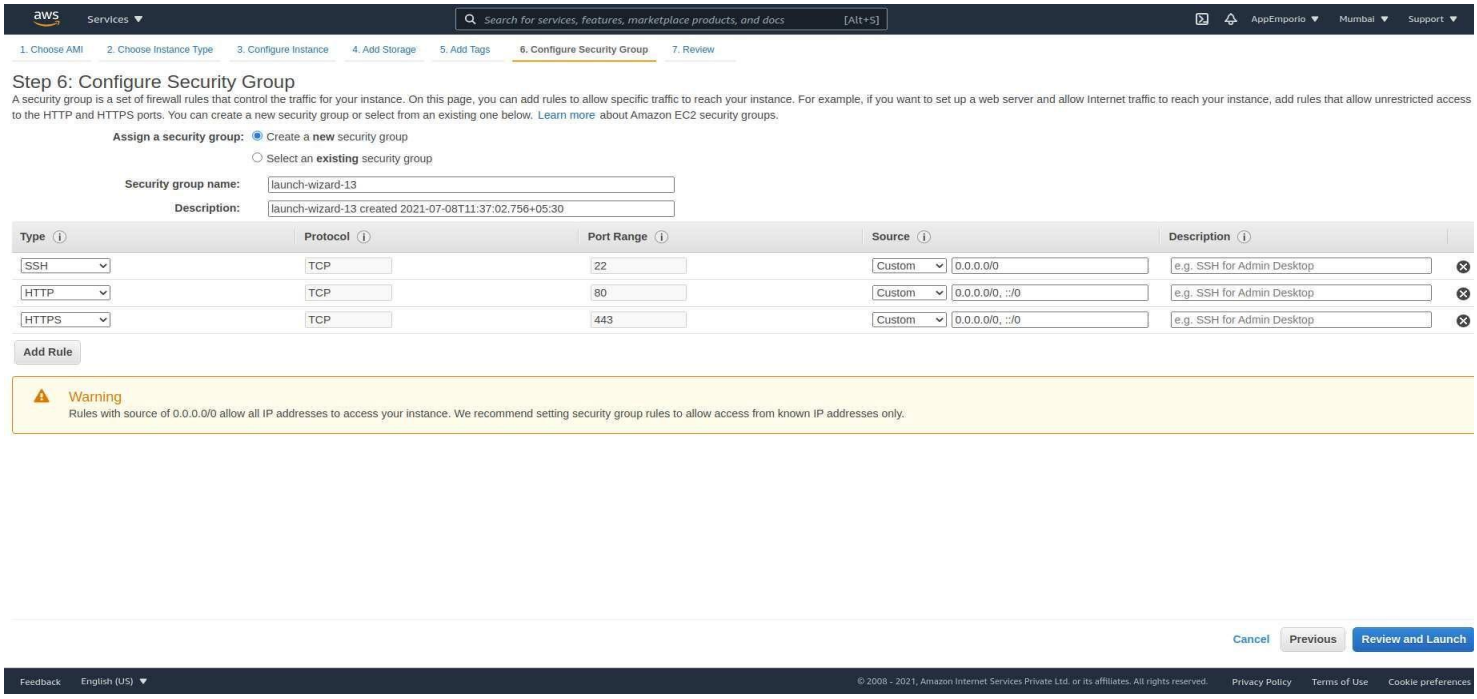


1.2.4(v) Add Tags(Optional)

- A tag consists of a case-sensitive key-value pair. For example, you could define a tag with key = Name and value = Webserver. A copy of a tag can be applied to volumes, instances or both.
- This step is optional for us.

1.2.4(vi) Configure security group

- A security group is a set of firewall rules that control the traffic for your instance. On this page, you can add rules to allow specific traffic to reach your instance.
- We will requires SSH, HTTP, and HTTPS security groups to have ssh connection and http and https connections from web browsers.

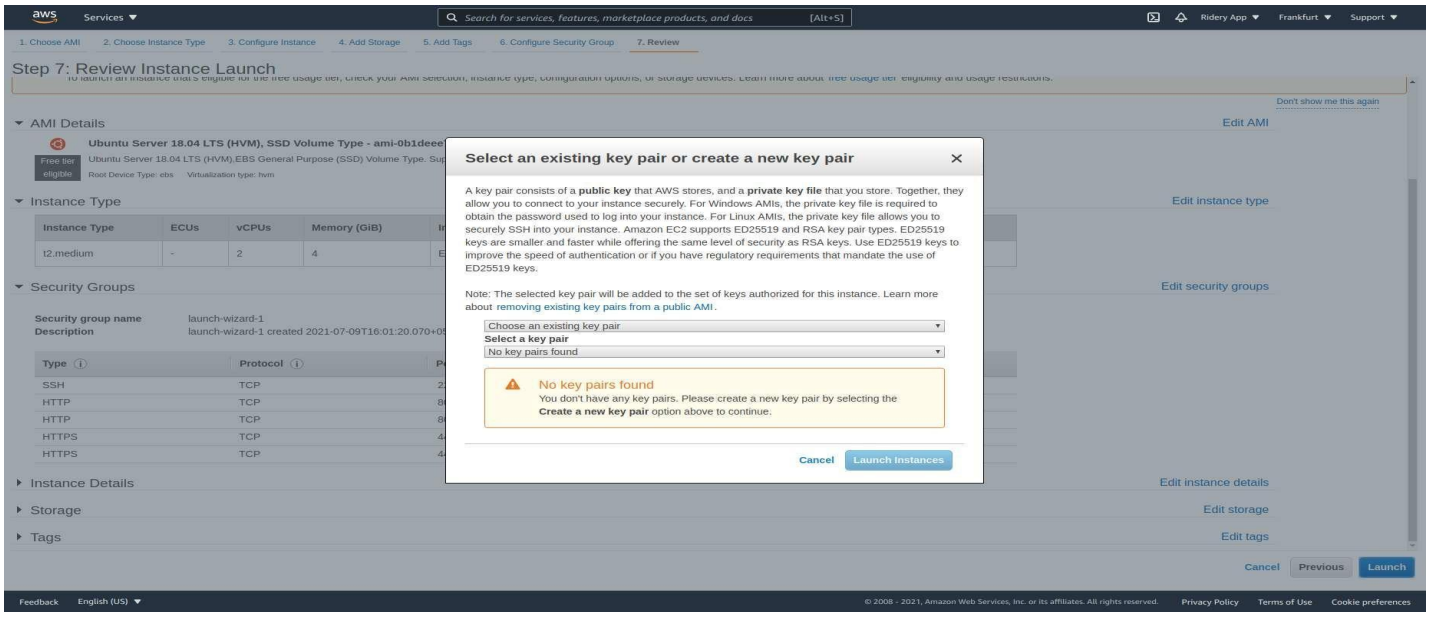


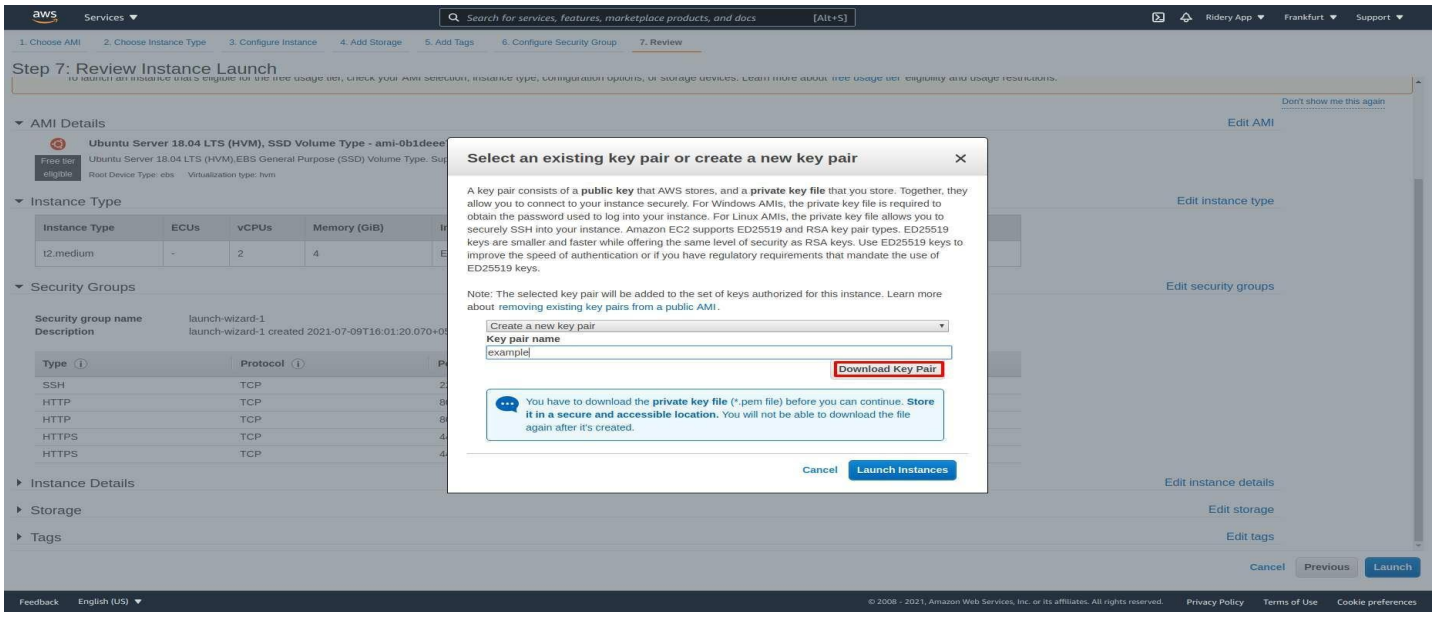
1.2.4(vii) Review

- Please review your instance launch details. You can go back to edit changes for each section.
- Click Launch to assign a key pair to your instance and complete the launch process.

1.2.4(viii) Security Keys

- As shown in the image below while reviewing we can see a popup for security. If you have any security keys then select that or generate a new one for our server.
- For creating new security keys we need to select Create a new key pair and enter the appropriate name for your key and press download key pair.


Select an existing key pair or create a new key pair
A key pair consists of a public key that AWS stores, and a private key file that you store. Together, they allow you to connect to your instance securely. For Windows AMIs, the private key file is required to obtain the password used to log into your instance. For Linux AMIs, the private key file allows you to securely SSH into your instance. Amazon EC2 supports ED25519 and RSA key pair types. ED25519 keys are smaller and faster while offering the same level of security as RSA keys. Use ED25519 keys to improve the speed of authentication or if you have regulatory requirements that mandate the use of ED25519 keys.
Note: The selected key pair will be added to the set of keys authorized for this instance. Learn more about removing existing key pairs from a public AMI.
Choose an existing key pair
Select a key pair
No key pairs found
No key pairs found
You don't have any key pairs. Please create a new key pair by selecting the Create a new key pair option above to continue.
Cancel
Launch Instances


Select an existing key pair or create a new key pair
A key pair consists of a public key that AWS stores, and a private key file that you store. Together, they allow you to connect to your instance securely. For Windows AMIs, the private key file is required to obtain the password used to log into your instance. For Linux AMIs, the private key file allows you to securely SSH into your instance. Amazon EC2 supports ED25519 and RSA key pair types. ED25519 keys are smaller and faster while offering the same level of security as RSA keys. Use ED25519 keys to improve the speed of authentication or if you have regulatory requirements that mandate the use of ED25519 keys.
Note: The selected key pair will be added to the set of keys authorized for this instance. Learn more about removing existing key pairs from a public AMI.
Create a new key pair
Key pair name
example
Download Key Pair
You have to download the private key file (*.pem file) before you can continue. Store it in a secure and accessible location. You will not be able to download the file again after it's created.
Cancel
Launch Instances

2). Add Elastic IP address in AWS

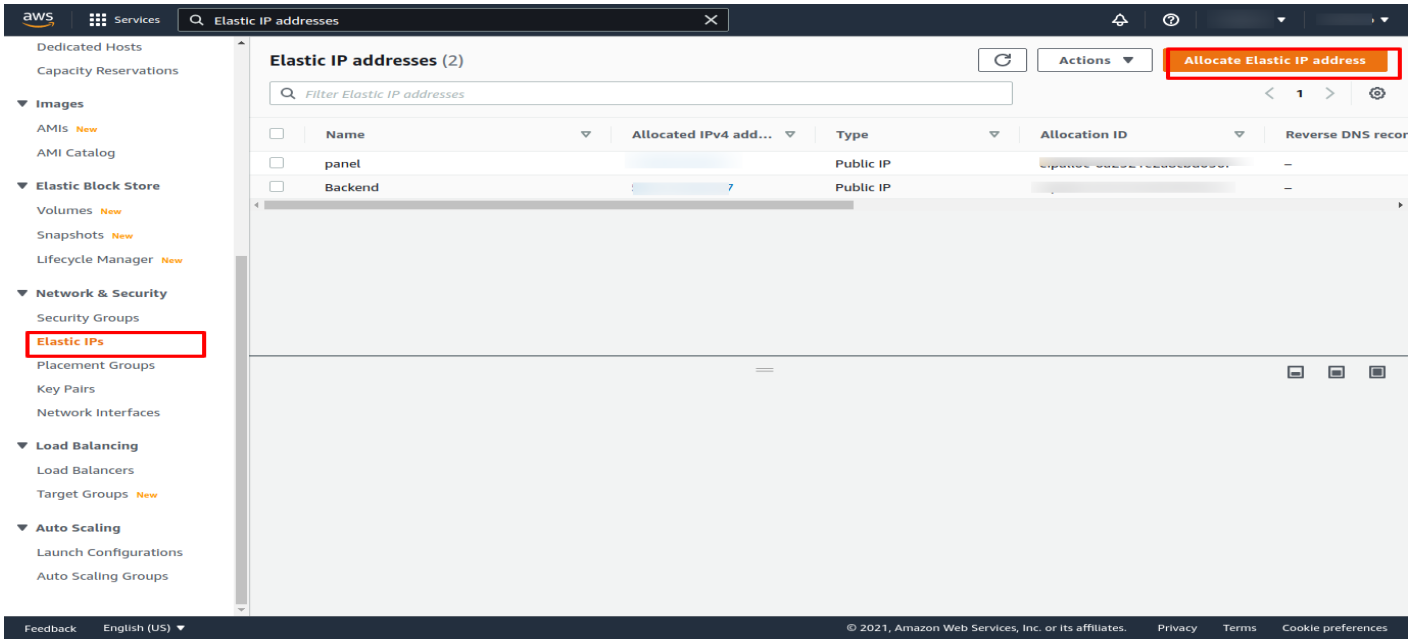
2.1) What is Elastic Ip and why it is useful?

The Elastic IP address is a public **static IPv4 address** that is reachable from the Internet. Basically, Elastic IP addresses are used by AWS to manage its dynamic cloud computing services. Within the AWS infrastructure, customers have virtual private clouds (VPC), within the VPCs, users have instances. So when you launch an EC2 instance, you receive a Public IP address by which that instance is reachable from the internet. Once you stop that instance and restart the instance you get a new Public IP for the same instance. So it's basically a problem to connect your instance from the internet for not having a static IP. To overcome this problem, we attach an Elastic IP to an Instance that doesn't change after you stop/start the instance.

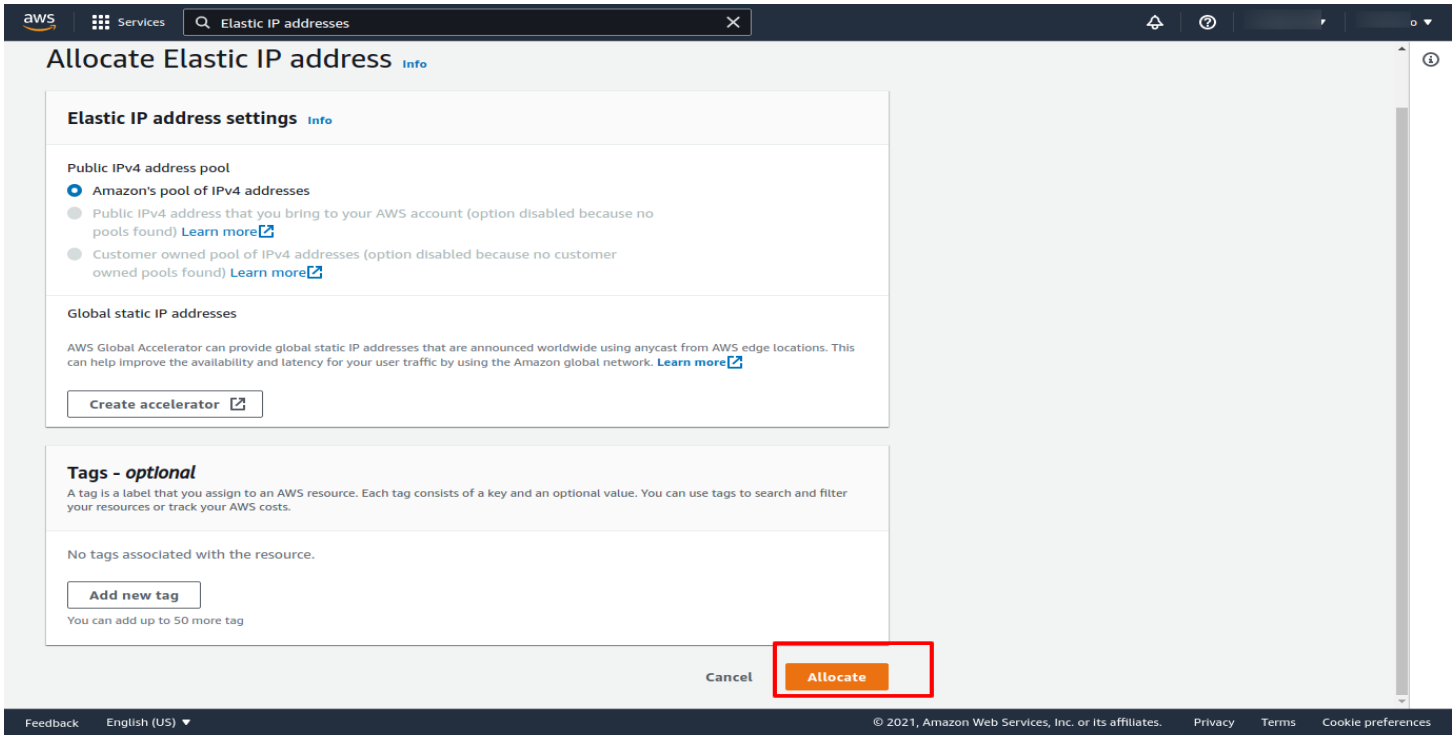
2.2) Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/

In the navigation pane, choose Network & Security, Elastic IPs.

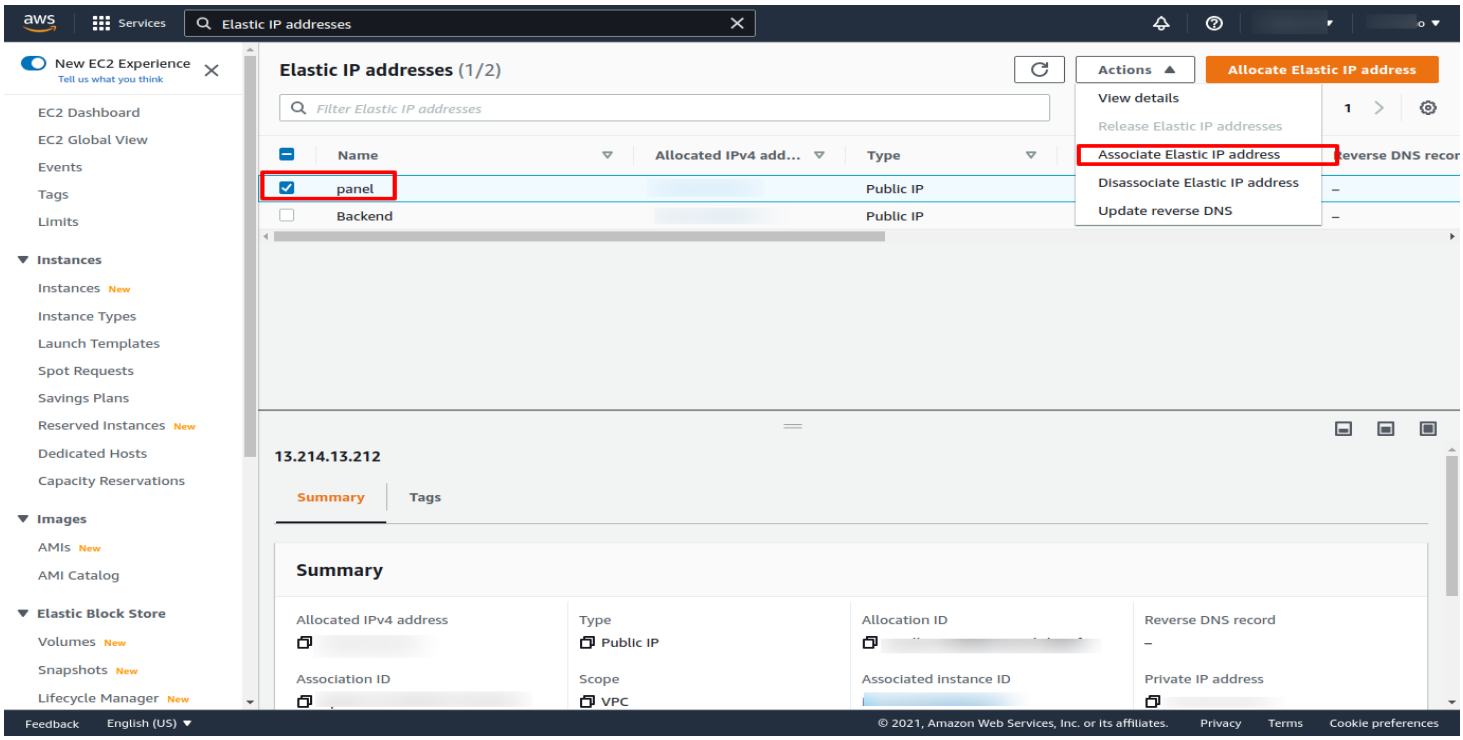
2.3) Choose Allocate Elastic IP address.



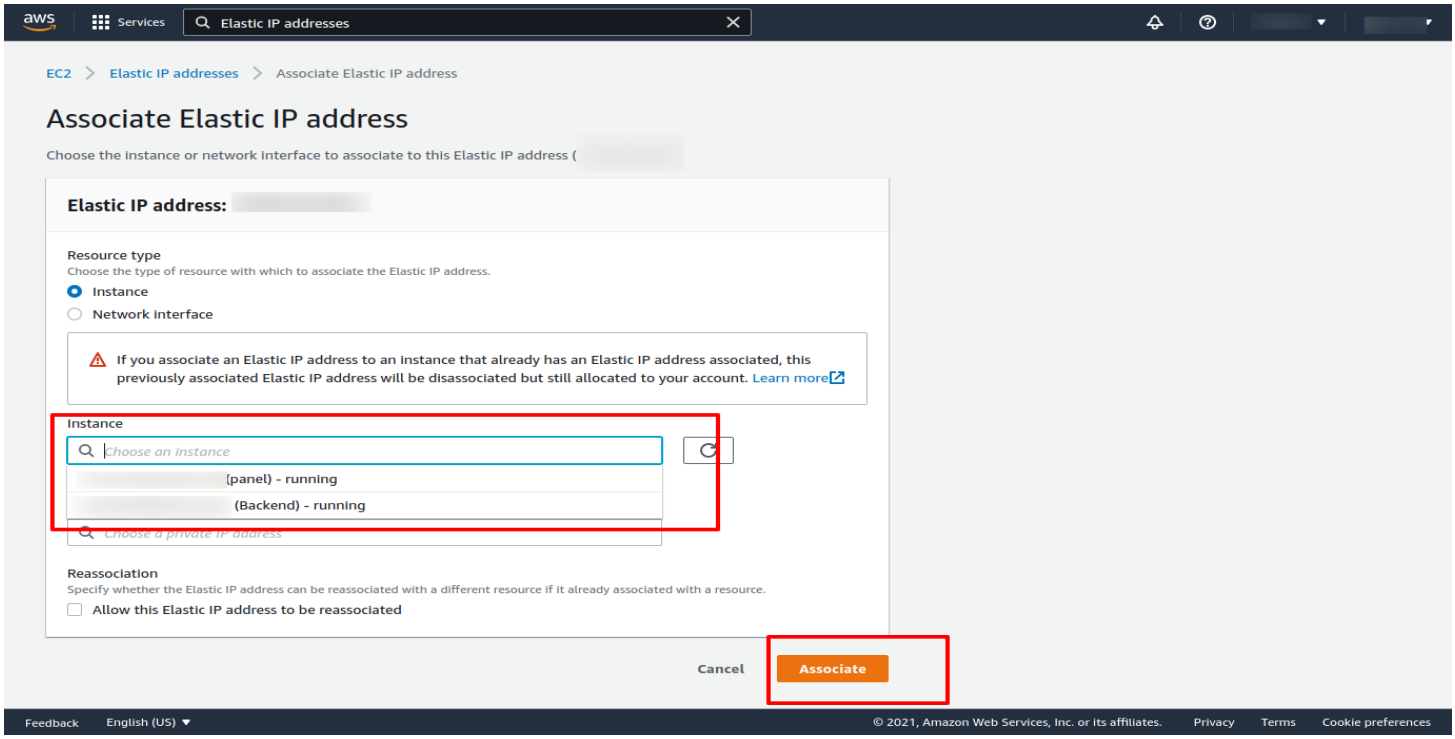
2.4) For the Public IPv4 address pool, choose one of the following and then allocate



2.5) Now Select IP and click on allocate an elastic IP address.



2.6) Select your project Instance and allocate.



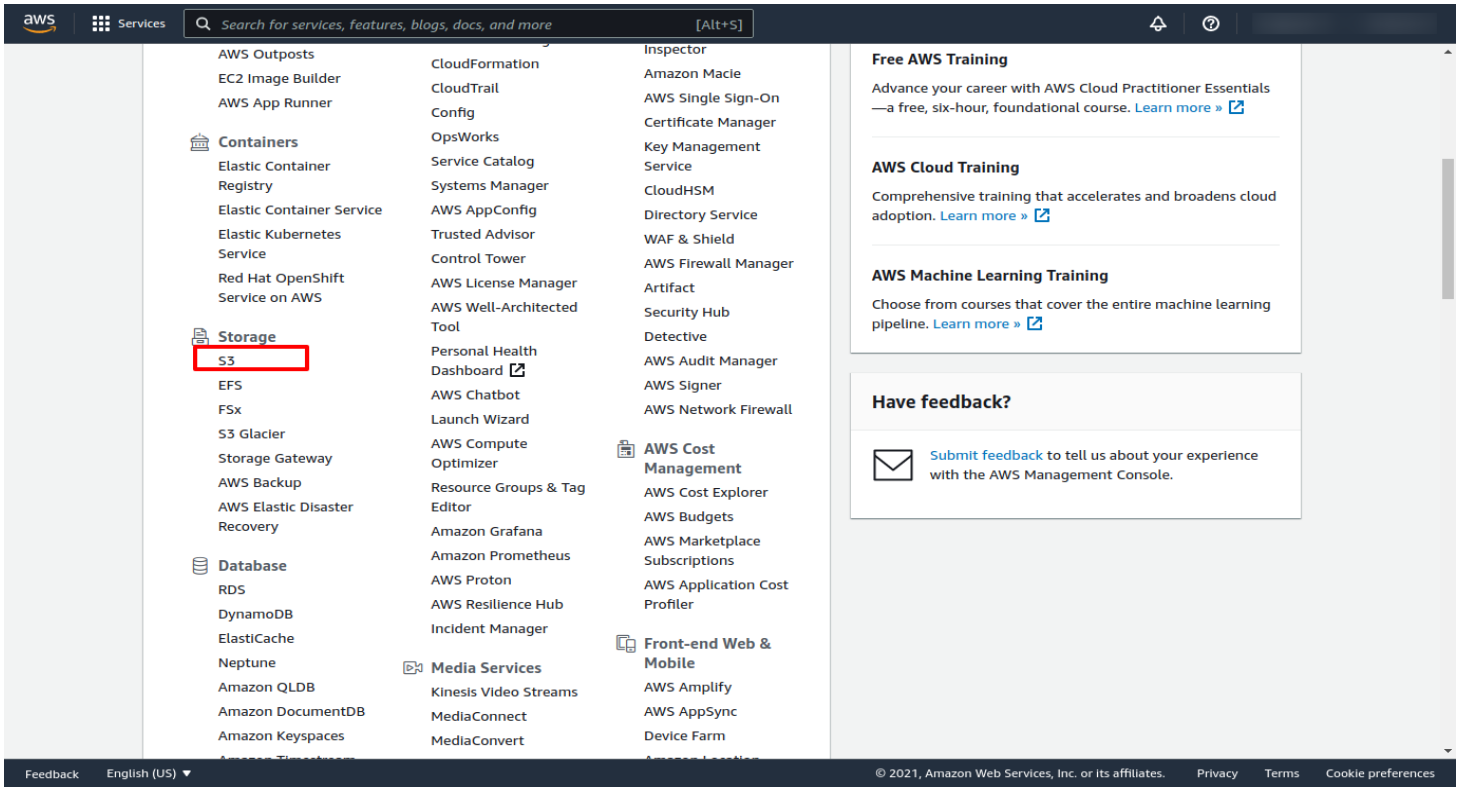
3). Create s3 Bucket in AWS.

3.1) What is S3 Bucket and why do we use it?

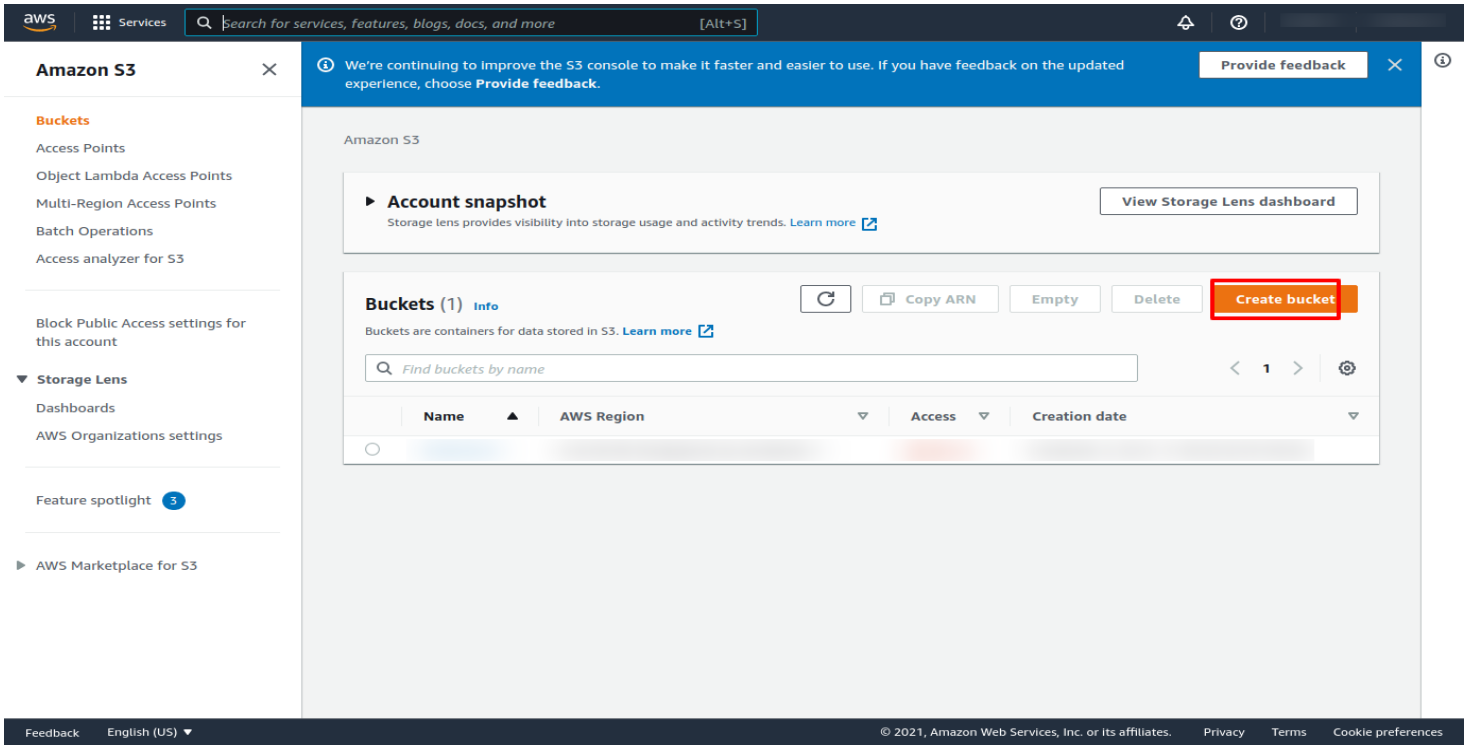
An Amazon S3 bucket is a public cloud storage resource available in Amazon Web Services' (AWS) Simple Storage Service (S3), an object storage offering. Amazon S3 buckets, which are similar to file folders, store objects, which consist of data and its descriptive metadata.

Amazon S3 is a program that's built to store, protect, and retrieve data from “buckets” at anytime from anywhere on any device. ... Use cases include websites, mobile apps, archiving, data backups and restorations, IoT devices, enterprise application storage, and providing the underlying storage layer for your data lake.

3.2) Click on s3 from all services

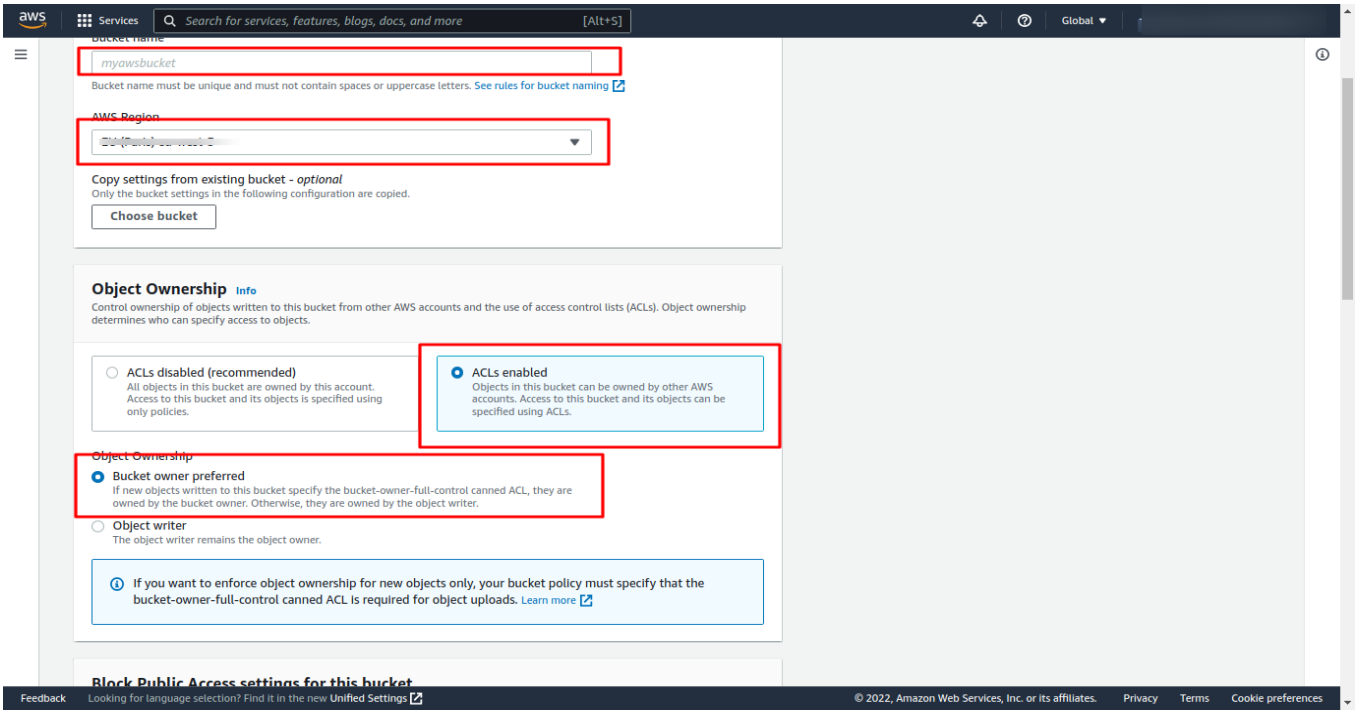


3.3) Click on create a bucket



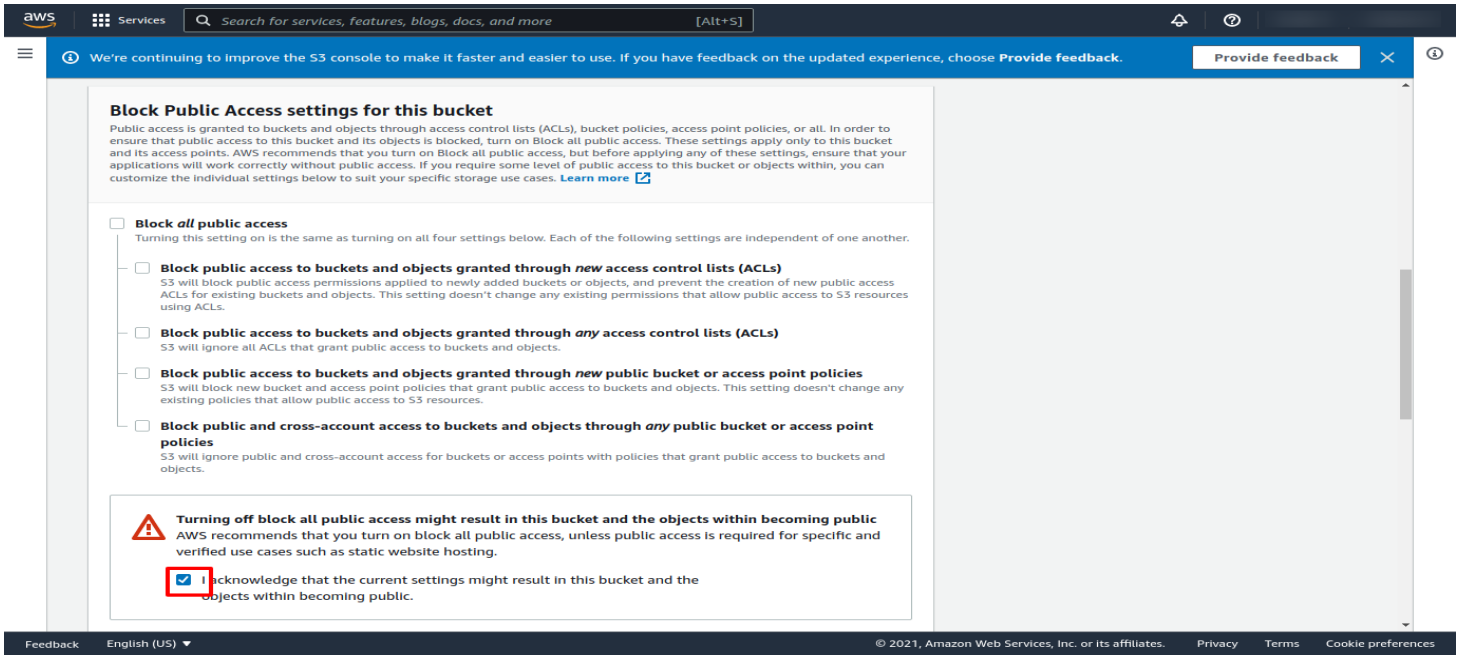
3.4) Add bucket name and select region according to country.

- In Bucket name, enter a unique name for your bucket.
- Choose a Region close to you to minimize latency and costs and address regulatory requirements. Objects stored in a Region never leave that Region unless you explicitly transfer them to another Region.

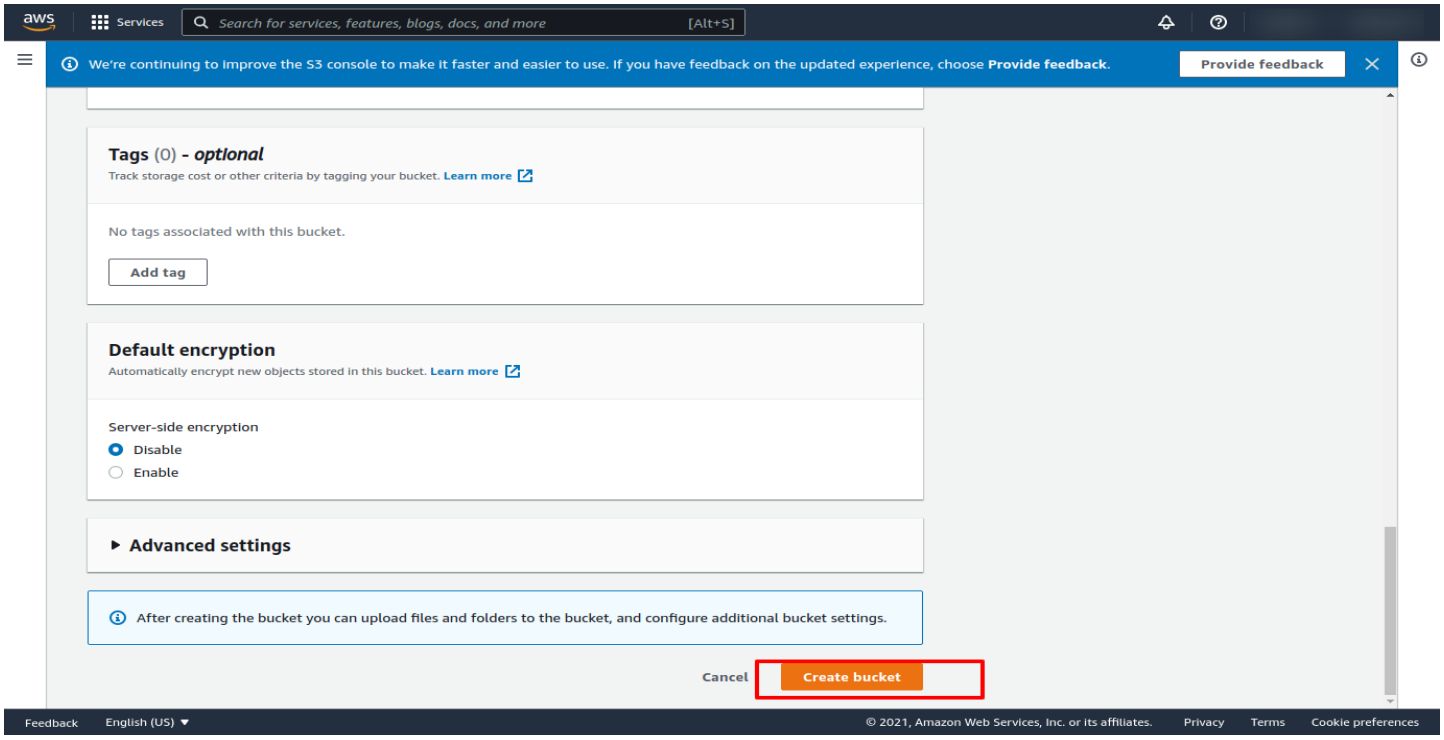


3.5) Unblock public access.

- We need to disable the block off public access
- Accept the Acknowledge because our bucket gives the public access.

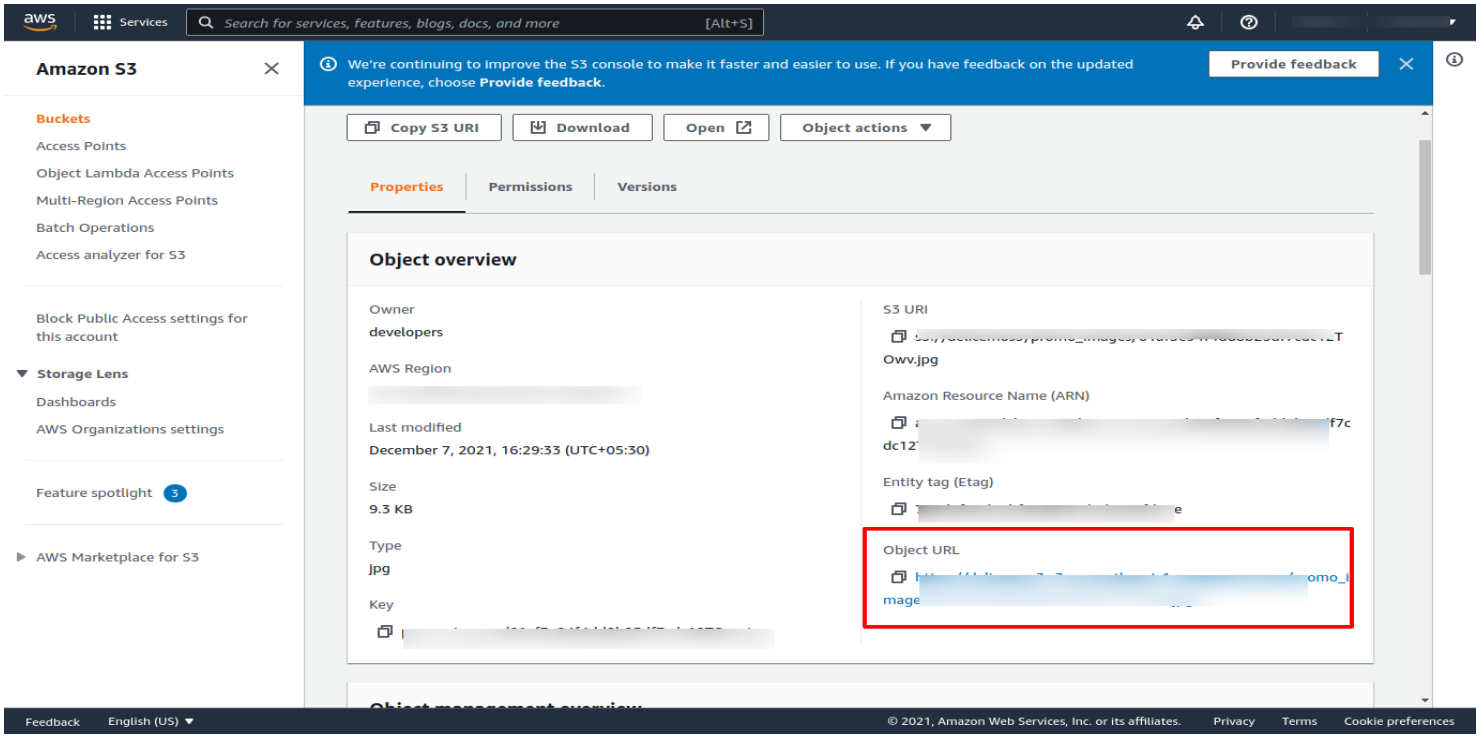


3.6) Now Click on Create Bucket.



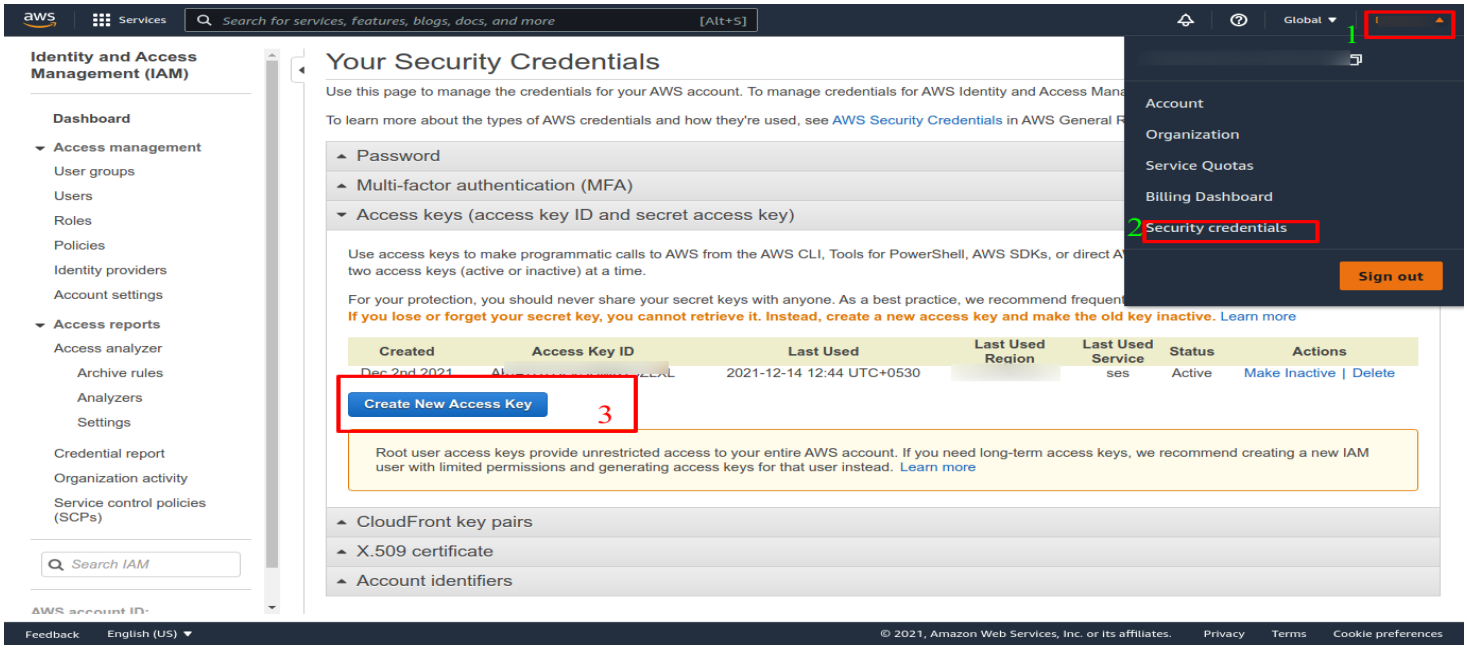
3.7) how to get an S3 bucket image URL

1) Upload image in s3 bucket and select that image and open properties.
2) In object overview copy that Object URI .
3) It will look something like - “http://your-bucket.s3-website-us-east-1.amazonaws.com”



3.8) S3 bucket Access key and secret key download and save it.

1) Save this key & Add this Acces key & Secret key in the Database..
2) Add bucket name in Database.



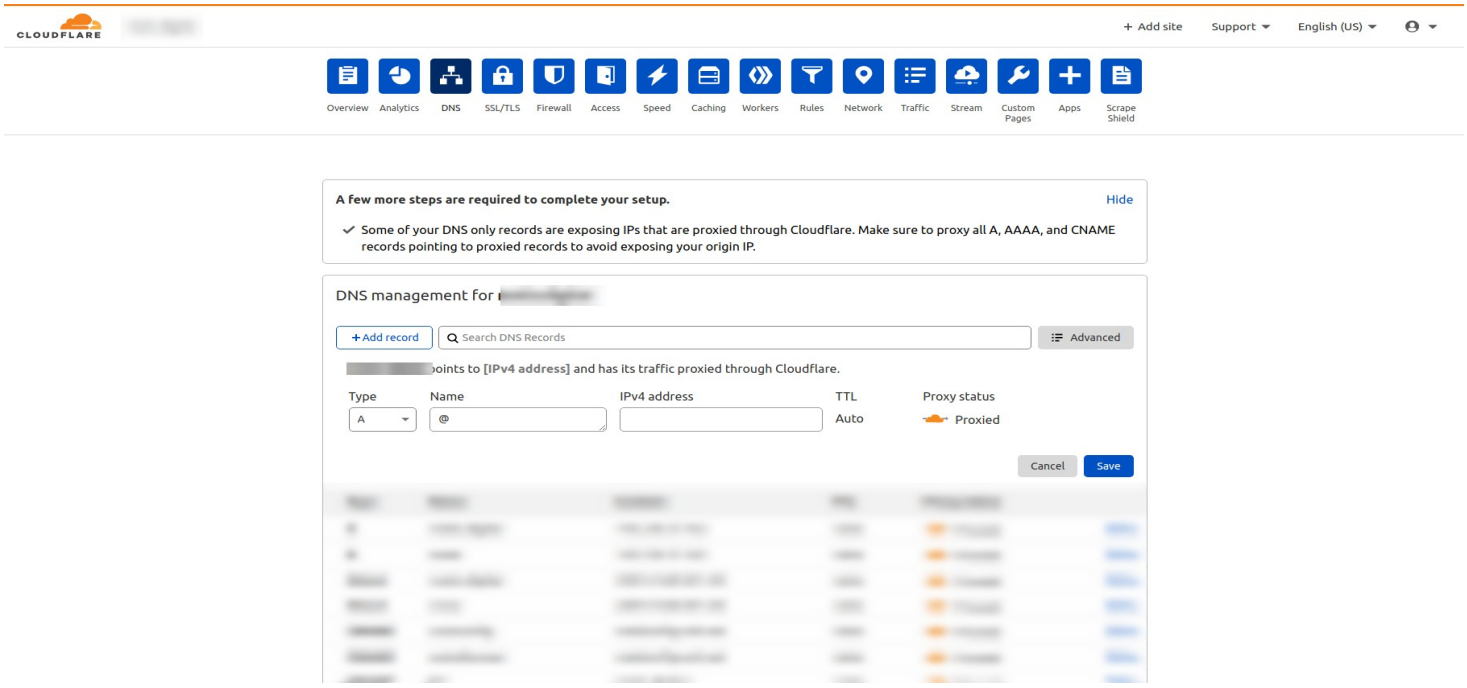
4). Pointing Domain name, and SSL certificate

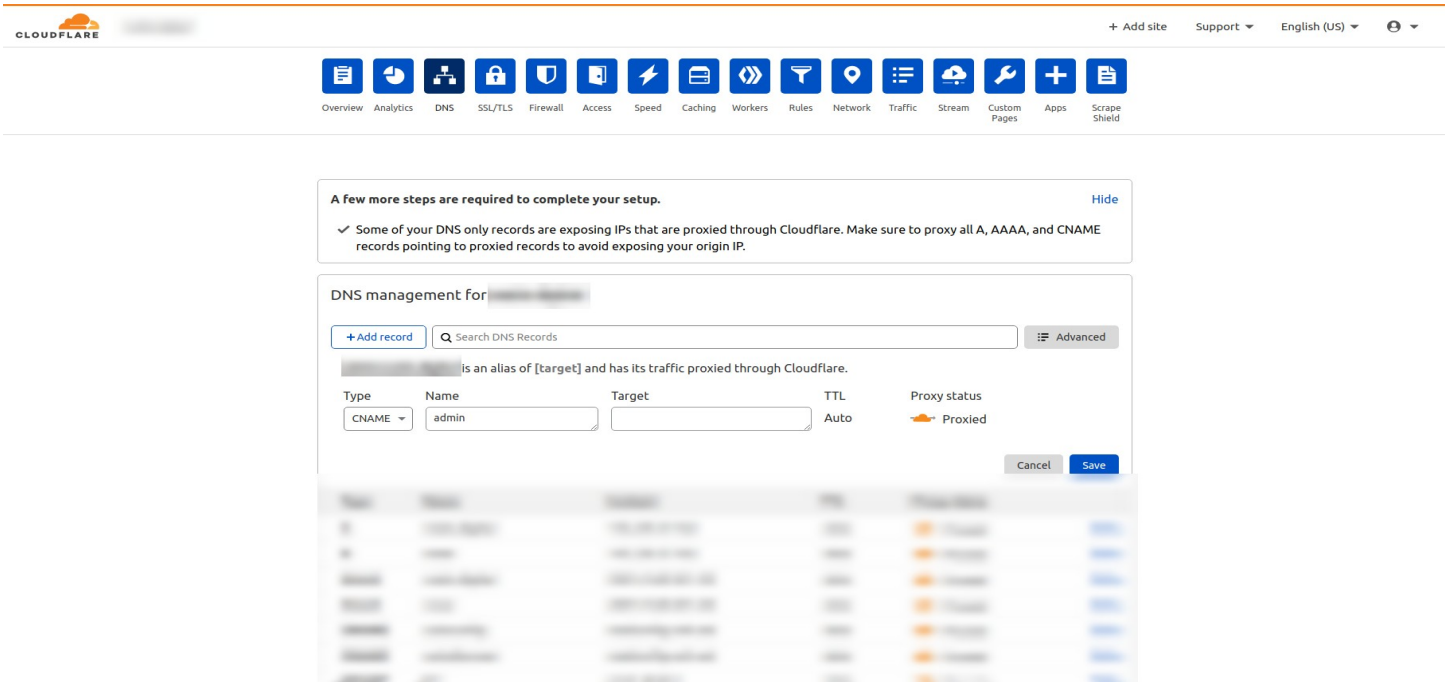
Now as our code is now running on the server, we can't open our application from search engines or web browsers using domain names.

For that, we need to setup a few things and that are:
1. Domain Name
2. SSL certificate

4.1 Domain Name(DNS)

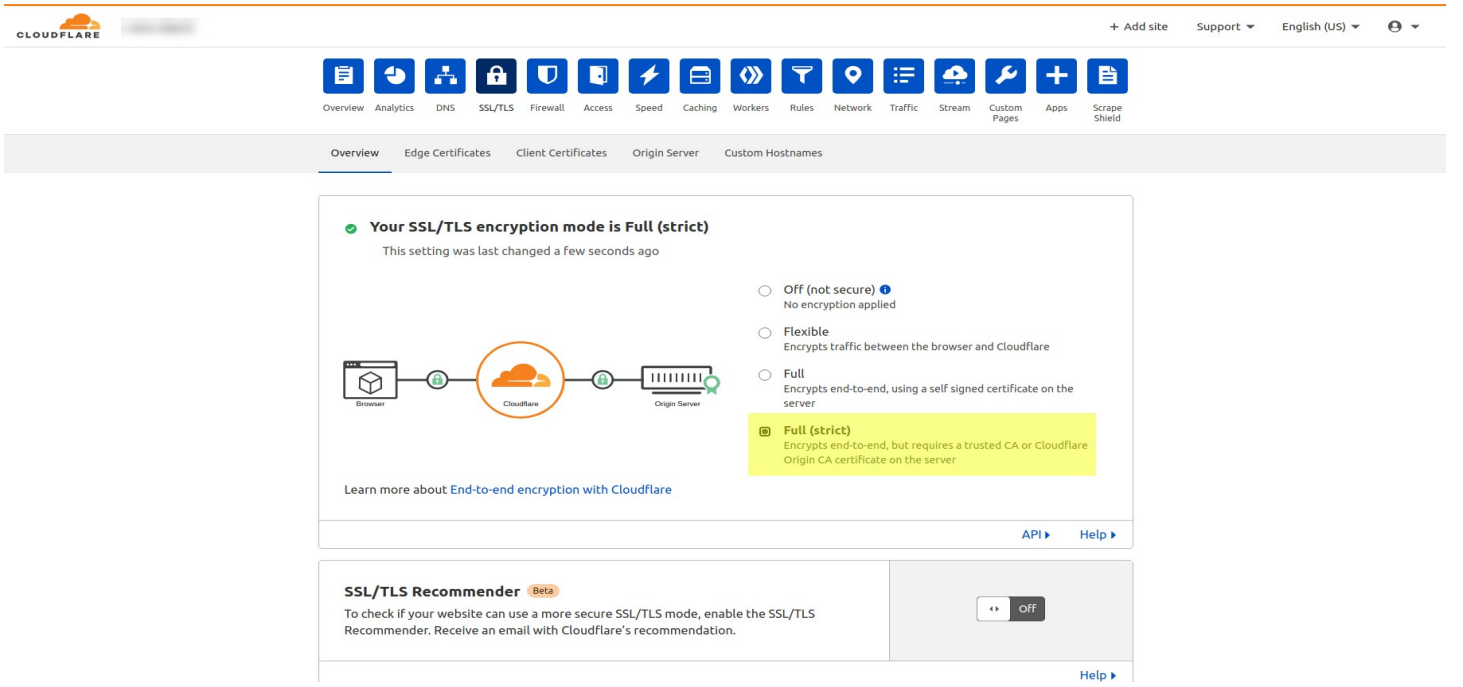
For Domain configuration we are using Cloudflare (https://www.cloudflare.com). Register your account and go to the DNS section and click Add Record. On the given input field in the type section select A (A records) and in name enter @ and in IPv4 enter the IP address of our server, then click Save. For subdomains click add new record again and in the type section select CNAME and in name section enter admin and in IPv4 enter @ (it indirectly points to our server IP address). Add 3 more subdomains for API, store, and for user.
Generally, for our user panels, we use our domain name instead of sud domains, And now we are done with our Domain pointing.

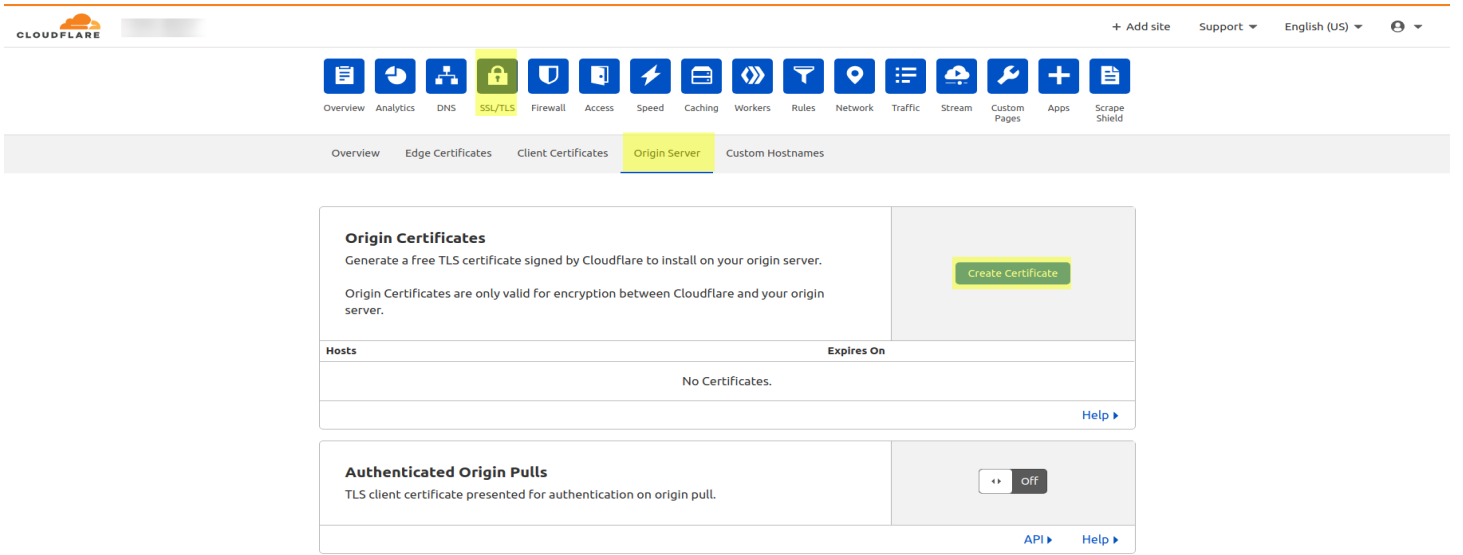




4.2 SSL certificate

Now we have a domain pointed to our IP address, let’s create an SSL certificate to secure our domain. Let’s head towards Cloudflare for SSL certificates (https://www.cloudflare.com). Head towards the SSL/TLS section(SSL\TLS > Overview) and select your plan(prefered full strict). For certificates go to SSL/TLS > Origin Server to generate certificates. Please check screenshots for References.


CLOUDFLARE
+ Add site
Support
English (US)
Overview
Analytics
DNS
SSL/TLS
Firewall
Access
Speed
Caching
Workers
Rules
Network
Traffic
Stream
Custom Pages
Apps
Scrape Shield
Overview
Edge Certificates
Client Certificates
Origin Server
Custom Hostnames
Your SSL/TLS encryption mode is Full (strict)
This setting was last changed a few seconds ago
Browser
Cloudflare
Origin Server
Off (not secure)
No encryption applied
Flexible
Encrypts traffic between the browser and Cloudflare
Full
Encrypts end-to-end, using a self signed certificate on the server
Full (strict)
Encrypts end-to-end, but requires a trusted CA or Cloudflare Origin CA certificate on the server
Learn more about End-to-end encryption with Cloudflare
API
Help
SSL/TLS Recommender
Beta
To check if your website can use a more secure SSL/TLS mode, enable the SSL/TLS Recommender. Receive an email with Cloudflare's recommendation.
Off
Help


CLOUDFLARE
+ Add site
Support
English (US)
Overview
Analytics
DNS
SSL/TLS
Firewall
Access
Speed
Caching
Workers
Rules
Network
Traffic
Stream
Custom Pages
Apps
Scrape Shield
Overview
Edge Certificates
Client Certificates
Origin Server
Custom Hostnames
Origin Certificates
Generate a free TLS certificate signed by Cloudflare to install on your origin server.
Origin Certificates are only valid for encryption between Cloudflare and your origin server.
Create Certificate
Hosts
Expires On
No Certificates.
Help
Authenticated Origin Pulls
TLS client certificate presented for authentication on origin pull.
Off
API
Help

5) Establishing SSH connection to connect server and installing environment for our code to run.

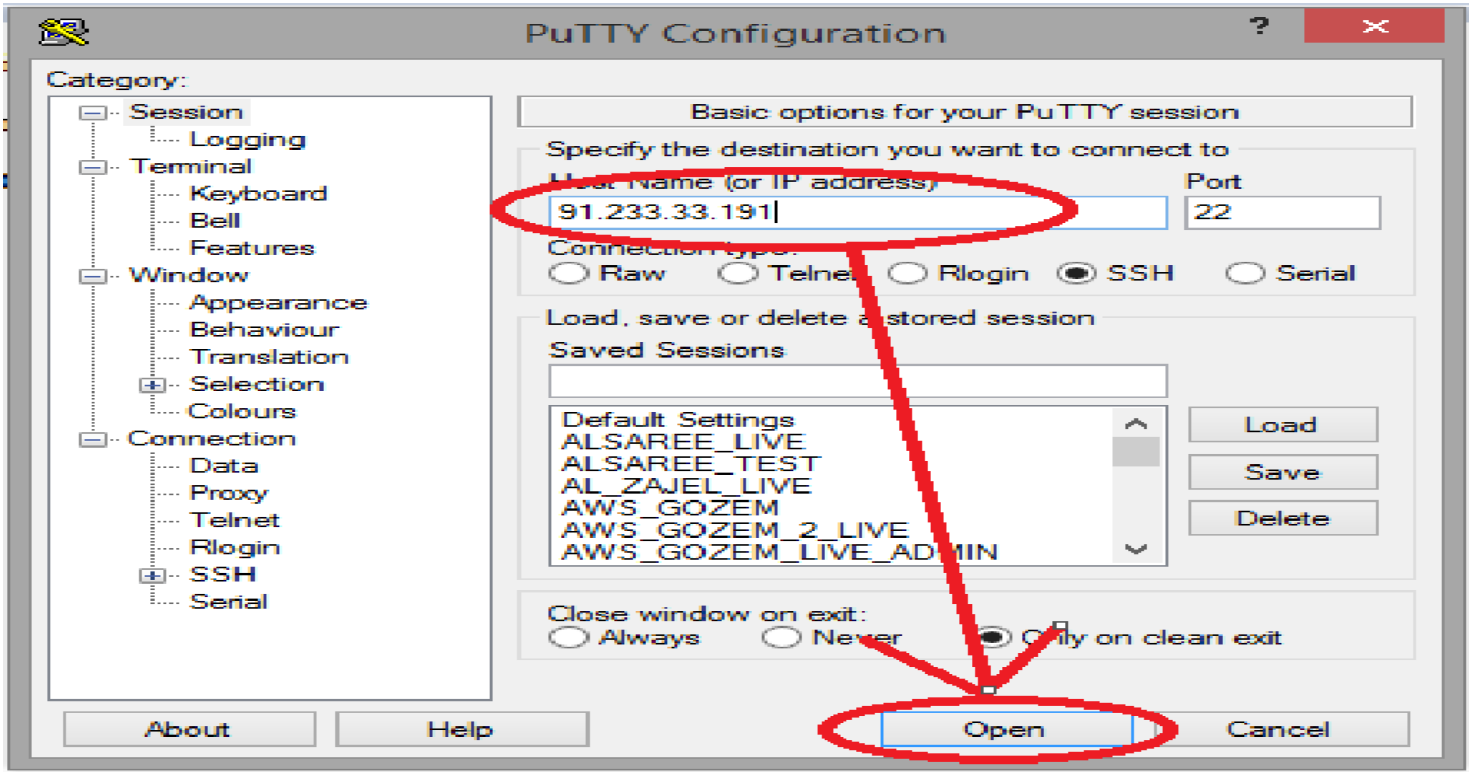
We will be Using putty for establishing SSH connection between our system

and server.

What is PuTTY ?

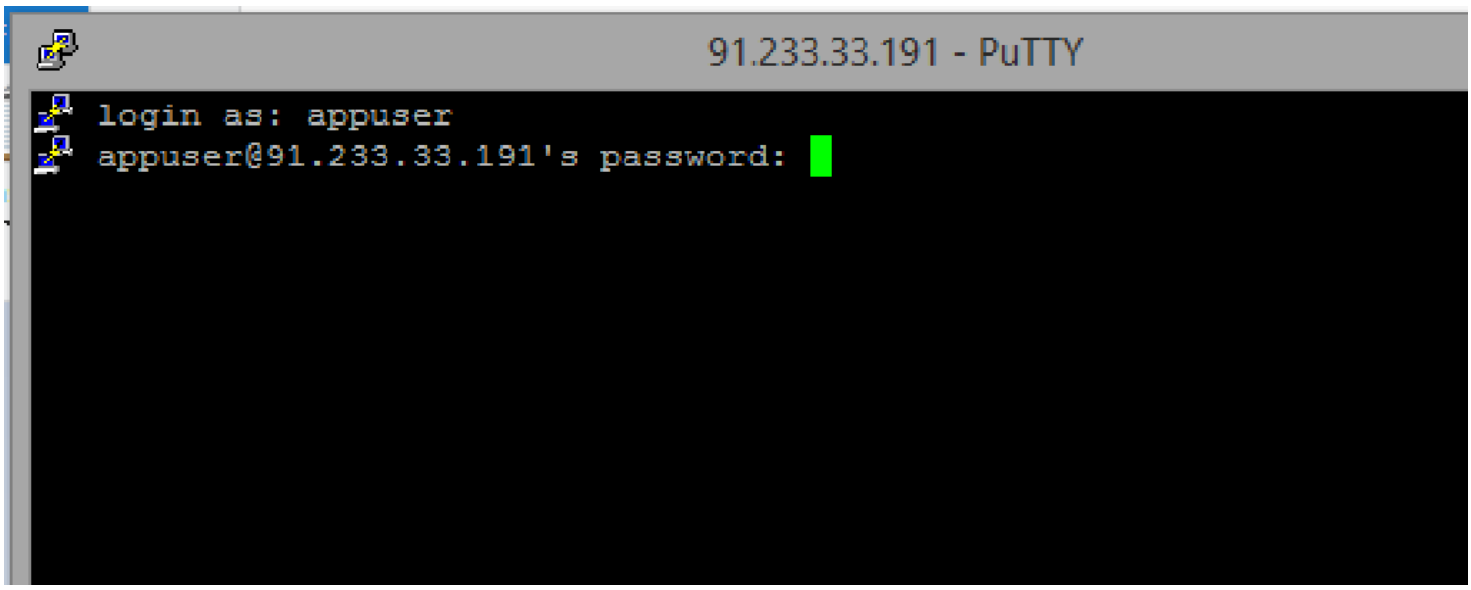
PuTTY is a free and open-source terminal emulator, serial console, and network file transfer application. It supports several network protocols, including SCP, SSH, Telnet, rlogin, and raw socket connection. It can also connect to a serial port. The name "PuTTY" has no official meaning.in short, it use for connect the ubuntu server in any operating system for application installation on the domain server.

Step 1 : Open PuTTY using ip



Step 2 : Enter username and password in the PuTTY console :

(if you connect with the key and ip then no need to provide a user name
and password at that time hostname be like : username@ip
for ex : ubuntu@91.233.33.191 or root@91.233.33.191)



Step 3 : Perform the below steps to install all dependancies or important server domain:

Command to change permission from root (Used in AWS) :

sudo chown <hostname>

hostname = root(for digitalocean domain) /

ubuntu(mostly in all) /

ur root name

Step 1 : Installing NodeJS Server by PPA Method (version 16)

sudo apt-get install python-software-properties

curl -sL https://deb.nodesource.com/setup_16.x | sudo -E bash

sudo apt-get install nodejs

(Reference : https://tecadmin.net/install-latest-nodejs-npm-on-ubuntu/)(always Install LTS release)

Step 2 : Installing NGINX Server

sudo apt-get update

sudo apt-get install nginx

sudo ufw app list

sudo ufw allow 'Nginx Full'

systemctl status nginx

sudo systemctl start nginx

(Reference : https://www.digitalocean.com/community/tutorials/how-to-install-nginx-on-ubuntu-16-04)

Step 3 : Installing MongoDB Server (4.4)

wget -qO - https://www.mongodb.org/static/pgp/server-4.4.asc | sudo apt-key add

sudo apt-get install gnupg

wget -qO - https://www.mongodb.org/static/pgp/server-4.4.asc | sudo apt-key add -

```
sudo apt-get update
```

Then select the appropriate tab below based on the result:

- `systemd` - select the **systemd (systemctl)** tab below.
- `init` - select the **System V Init (service)** tab below.

sudo systemctl start mongod

If you receive an error similar to the following when starting mongod:
`Failed to start mongod.service: Unit mongod.service not Found.`

sudo systemctl daemon-reload

sudo systemctl status mongod

sudo systemctl start mongod

(Reference :
https://www.mongodb.com/docs/v4.4/tutorial/install-mongodb-on-ubuntu/)

Step-4 : Installing Redis Session Management Server

sudo apt-get update

sudo apt-get install build-essential

sudo apt-get install tcl 8.6

wget https://download.redis.io/releases/redis-7.0.0.tar.gz

tar xzf redis-7.0.0.tar.gz

cd redis-7.0.0

make

make test

sudo make install

cd utils

sudo nano ./install_server.sh

Comment it →

```
#bail if this system is managed by systemd
#_pid_1_exe="$(readlink -f /proc/1/exe)"
#if [ "${_pid_1_exe##*/}" = systemd ]
#then
# echo "This systems seems to use systemd."
# echo "Please take a look at the provided example  service unit files in the
directory, and adept and install them.  sorry!"
# exit 1
#fi
```

sudo ./install_server.sh
then (press enter 5 times)

https://stackoverflow.com/questions/61694459/installing-redis-use-install-server-s

sudo service redis start

—--> if not installed command this 2 line
1> sudo apt-get install redis-server
2> sudo service redis-server status

Step-5 : Installing Git & Checkout Code

sudo apt-get install git

(command to give permission)
sudo chmod -R 777 /var/www/html

Change path :
cd /var/www/html (for ubuntu)
cd /root (for root)

git clone <Your URL>
(ex.: git clone https://gitlab.com/elluminatiEber/eberBackend.git)

Step-6 : Installing required modules & Configure database for production environment

cd backend (if root, else go to apropreate path)

sudo chmod -R 777 /var/www/html/ OR
sudo chmod -R 777 /root

rm -rf package-lock.json

sudo npm install

sudo npm install nodemon -g

sudo npm i pm2 -g

goto FileZilla and change DB name

//CONFIGURE DATABASE NAME IN PRODUCTION AND DEVELOPMENT ENVIRONMENT
sudo nano /root/eberBackend/config/env/production.js (for root)
sudo nano /var/www/html/backend/config/env/production.js (for ubuntu)

//Edit below line with your database name db:
'mongodb://localhost:27017/<YOUR_DATABASE_NAME>'

(permission of the folder if needed :)

sudo chmod -R 777 /var/www/html/web/new_user_panel/node_modules

sudo npm install socket.io-redis

nodemon server.js

Step-7 : Configure your NGINX Server config file

sudo nano /etc/nginx/sites-available/default

(default file is use to final configuration of our domain to ip we are appplying ssl in this file for security of our domain.
here, we are adding two server function one for admin panel and onother one is for user panel add your domain in this and related port in the below file and copy it in the default file.)

//Add below lines in the file

```
server {
listen 80;
server_name localhost;
root <path to server file>;

#Load configuration files for the default server block.
include /etc/nginx/default.d/*.conf;

location / {
proxy_pass http://localhost:5000;
proxy_http_version 1.1;
proxy_set_header Upgrade $http_upgrade;
proxy_set_header Connection 'upgrade';
proxy_set_header Host $host;
proxy_cache_bypass $http_upgrade;
}

error_page 404 /404.html;
location = /errorPage.html {
}
}
```

(**ctrl+o** to write out the file; **ctrl+x** to exit the file; then **y** to save the containt and give **enter**)

(Most IMP : STOP NGINX SERVER
sudo systemctl stop nginx)

Step-8 : Removing SSL comments from NGINX config file and restarting NGINX

sudo nano /etc/nginx/sites-available/default

//File after removing comments

```
server {
    server_name eberdeveloper.appemporio.net;
    location / {
        proxy_pass http://localhost:5000;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection 'upgrade';
        proxy_set_header Host $host;
        proxy_cache_bypass $http_upgrade;
    }

    listen 443 ssl;
    underscores_in_headers on;
    ssl_certificate /var/www/html/eber/ssl/eber_origin.pem;
    ssl_certificate_key /var/www/html/eber/ssl/eber_private.pem ;

    location /gmapsapi {
        proxy_pass https://maps.googleapis.com/;
        proxy_pass_request_body on;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    }

    error_page 404 /404.html;
          location = /errorPage.html {
    }


        if ($scheme = http) {
              return 301 https://$server_name$request_uri;
        }
}
```

//Save the file above and restart NGINX Server
sudo systemctl restart nginx
OR
sudo systemctl start nginx

Step 9 : Start Forever Server(mostly we are using PM2):

change Path :
cd /var/www/html/**<path to server.js of admin>** (for ubuntu)
cd /root /**<path to server.js of user>** (for root)

forever start server.js
OR
if you install pm2 then;
pm2 restart server.js

cd /var/www/html/**<path to server.js of user>** (for ubuntu)
cd /root /**<path to server.js of user>** (for root)
forever start server.js
OR
if you install pm2 then;
pm2 restart server.js

Step 10 : Add database from Setting_data folder :

cd /backend/settingsdata

node initial_data.js

After few second type ctrl + c to exit

nodemon server.js

Step 11: Once its start on the domain then change api keys :

first of all go to the gitlab acc of client and enable the api keys:

for backend ;

1) Road map API key

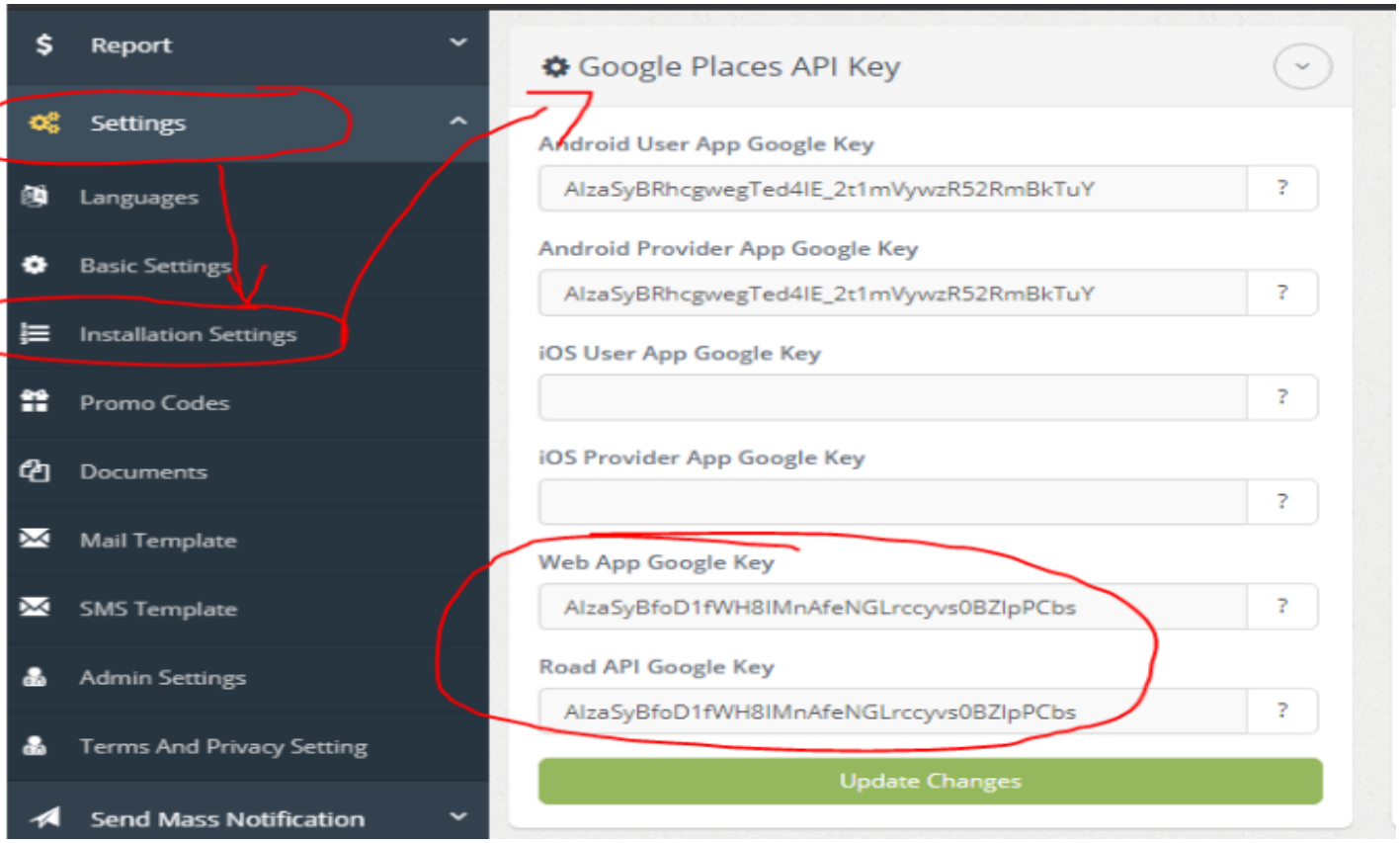
2)JavaScript API key

or you can do all api enable if you want to do ;)

-> Add Browser Key in the;

admin panel -> settings -> installation settings ->

1. Web App google key
2. Road Map google key



 # EXTRA :
sudo zip -r web.zip web/

Step 12 : connecting to db

go in to the puttygen and generate the private key public key and db key to use db.
go to the robo3t .
add ssh connection and enter details relate then test and connect.

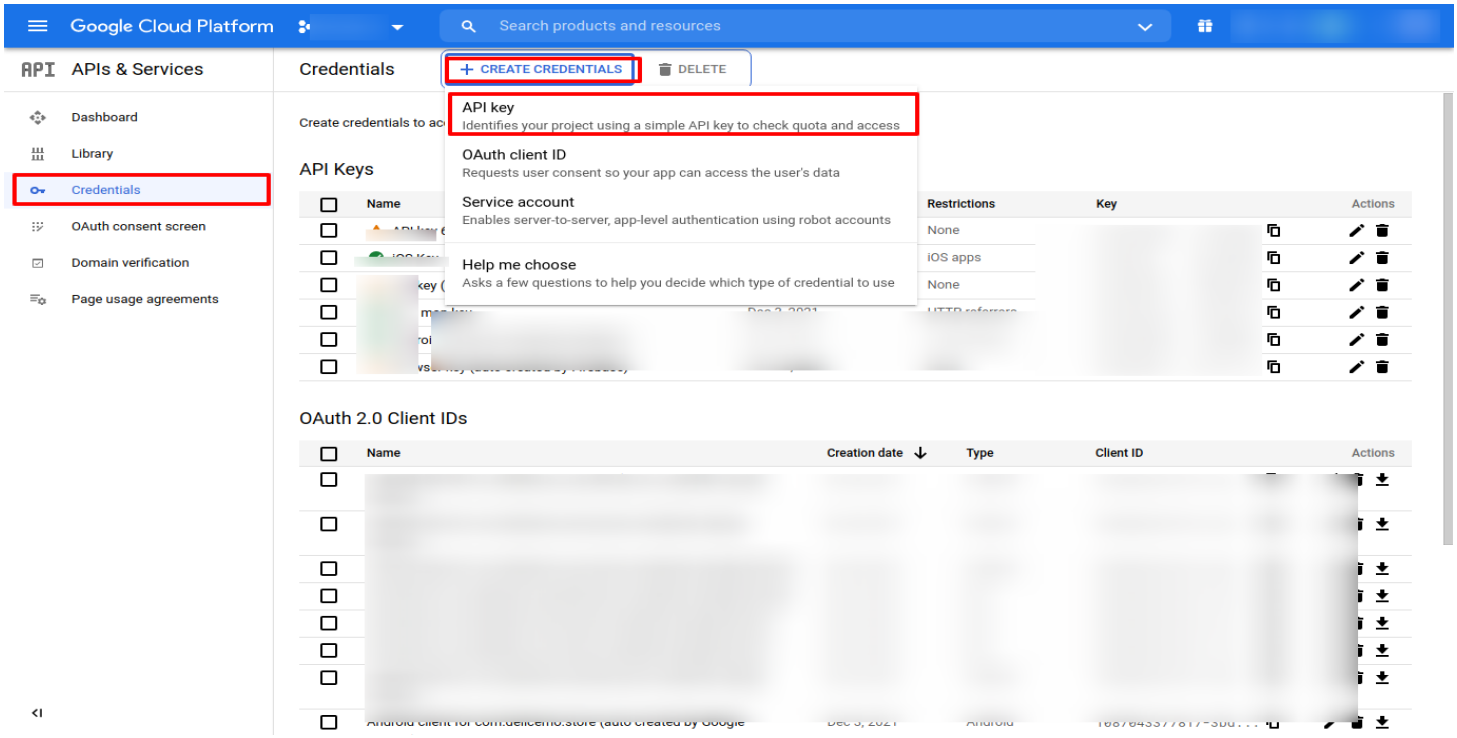
twilio ml :
twilio bin ->
<Response>
<Dial>
<Number>{{to}}</Number>
</Dial>
</Response>

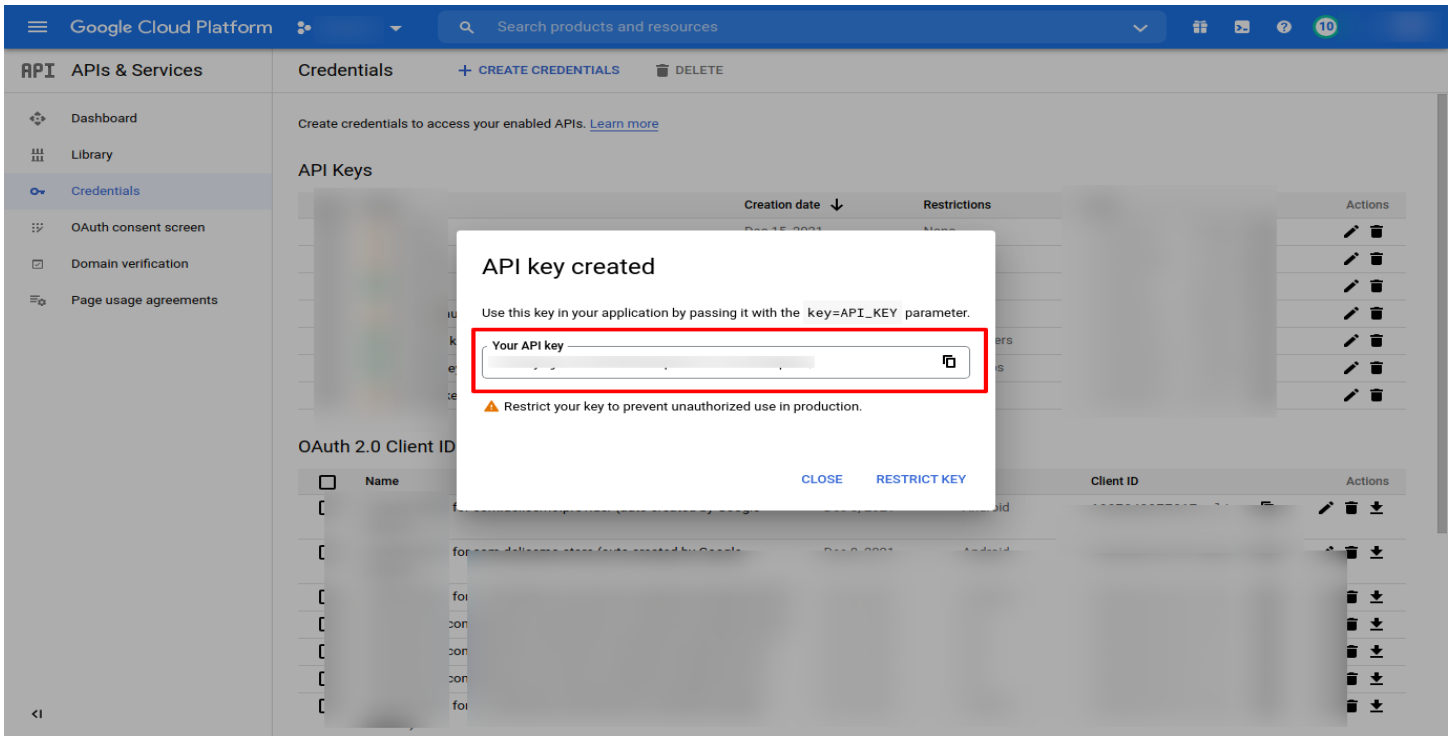
6). Google API key

1) Open Google cloud console.

2) Copy that API key and save it in a text file and use it in Your project

3) Set this API Key in the given files.





4) --Google Api key----

-> Turn on this apis

Directions API
Distance Matrix API
Geocoding API
Geolocation API
Maps JavaScript API
Maps SDK for Android
Maps SDK for iOS
Maps Static API
Places API
Roads API

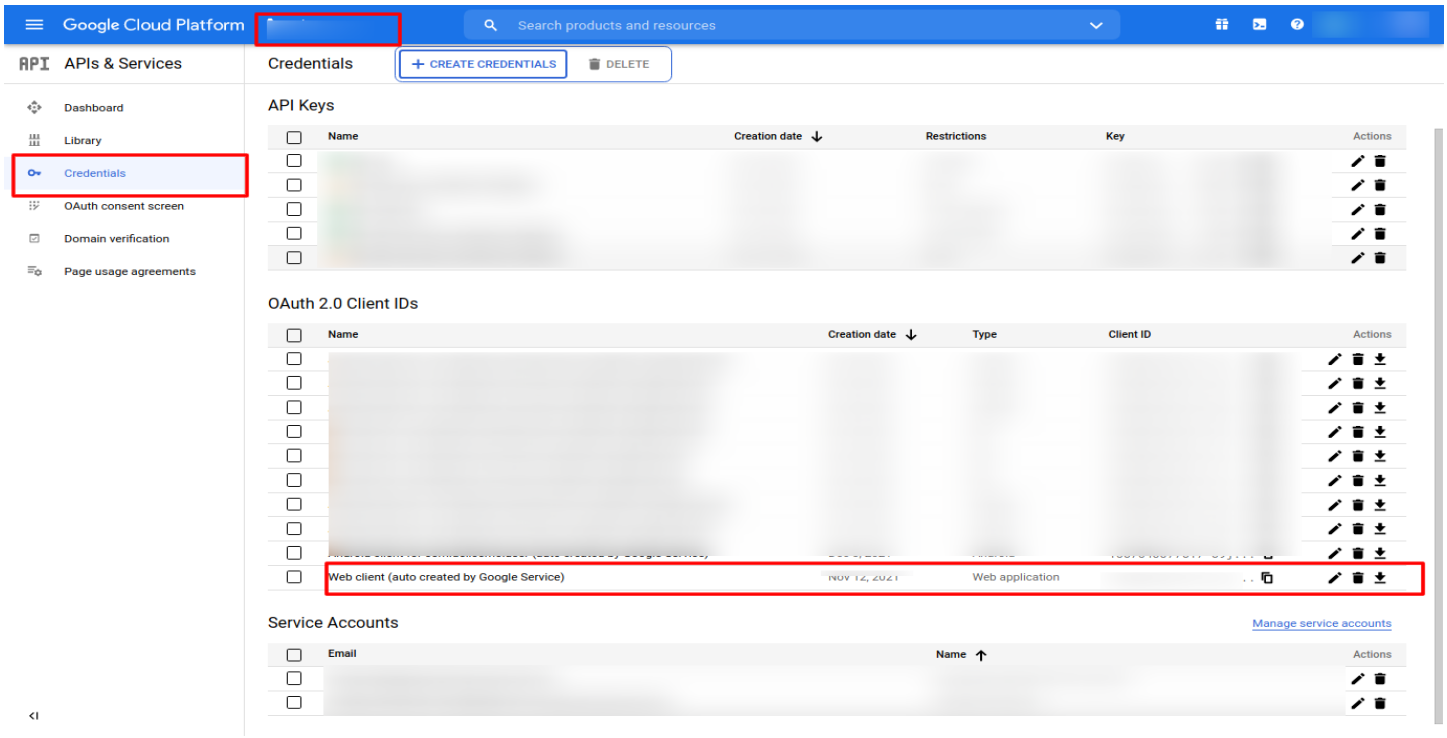
*Please Restrict the Google API Key

7). Social login using Gmail

1) Open google console and select your project.

2) Select Web client auto service.

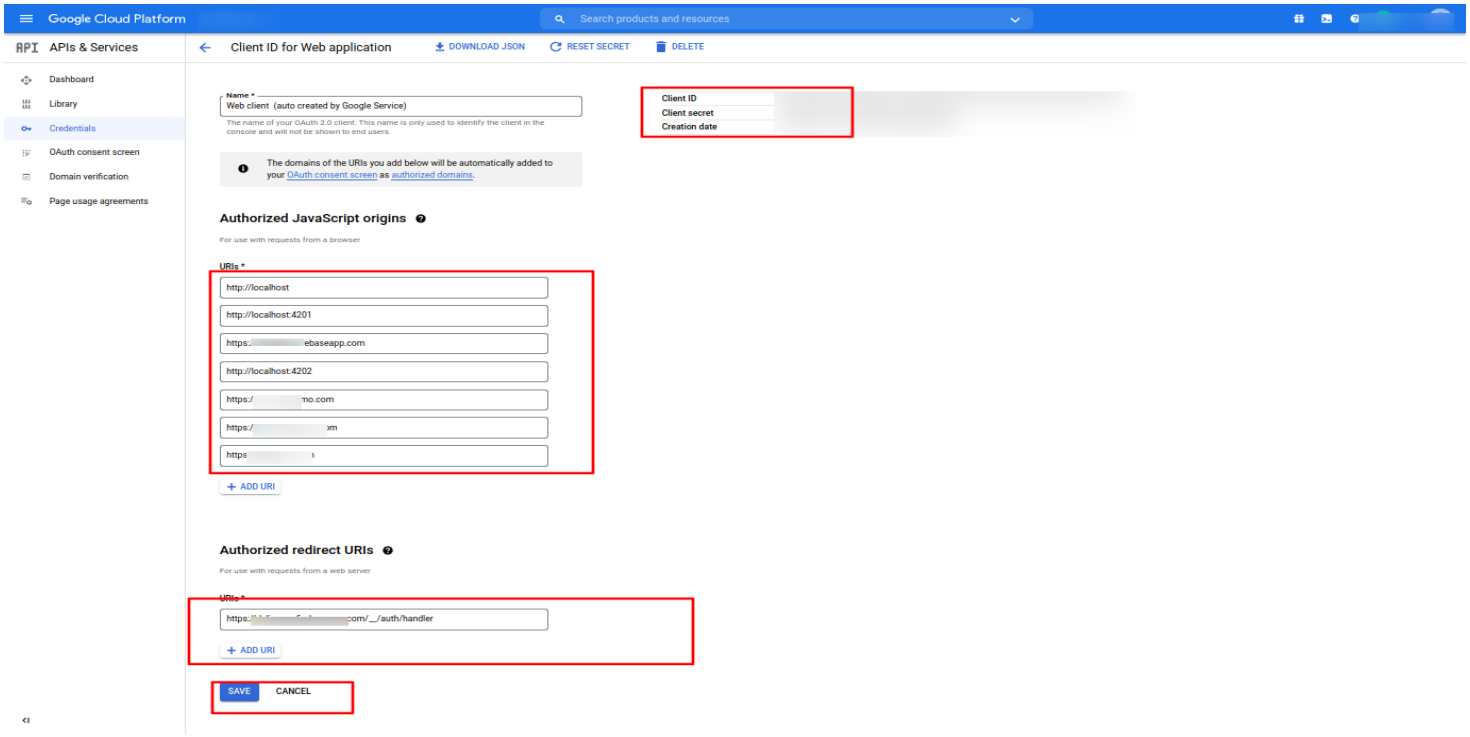
* Note -> Set google app id in the project after generating.



3) Add your domain name to URI

4) And use Client Id & Secret in your project Database.

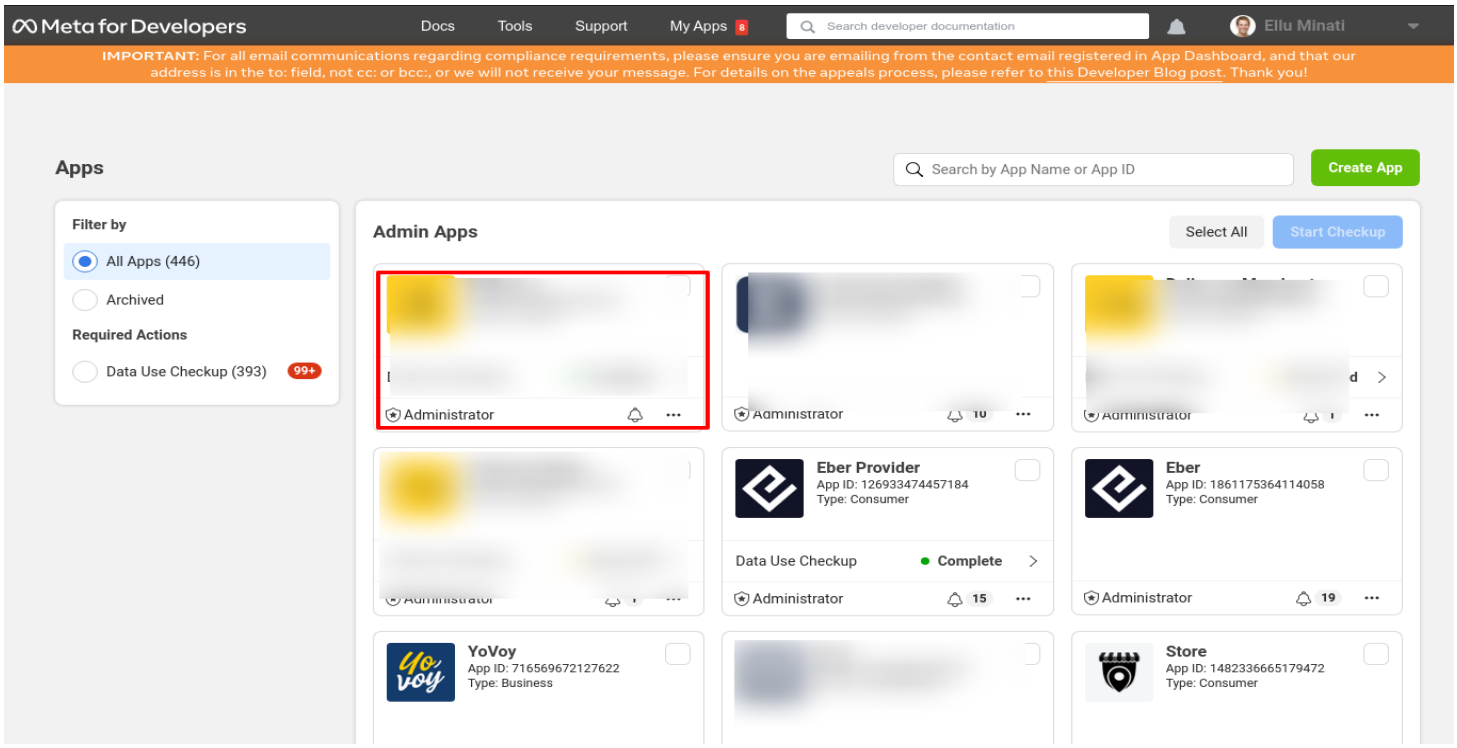
5) you have to add client_id in constant.json collection.



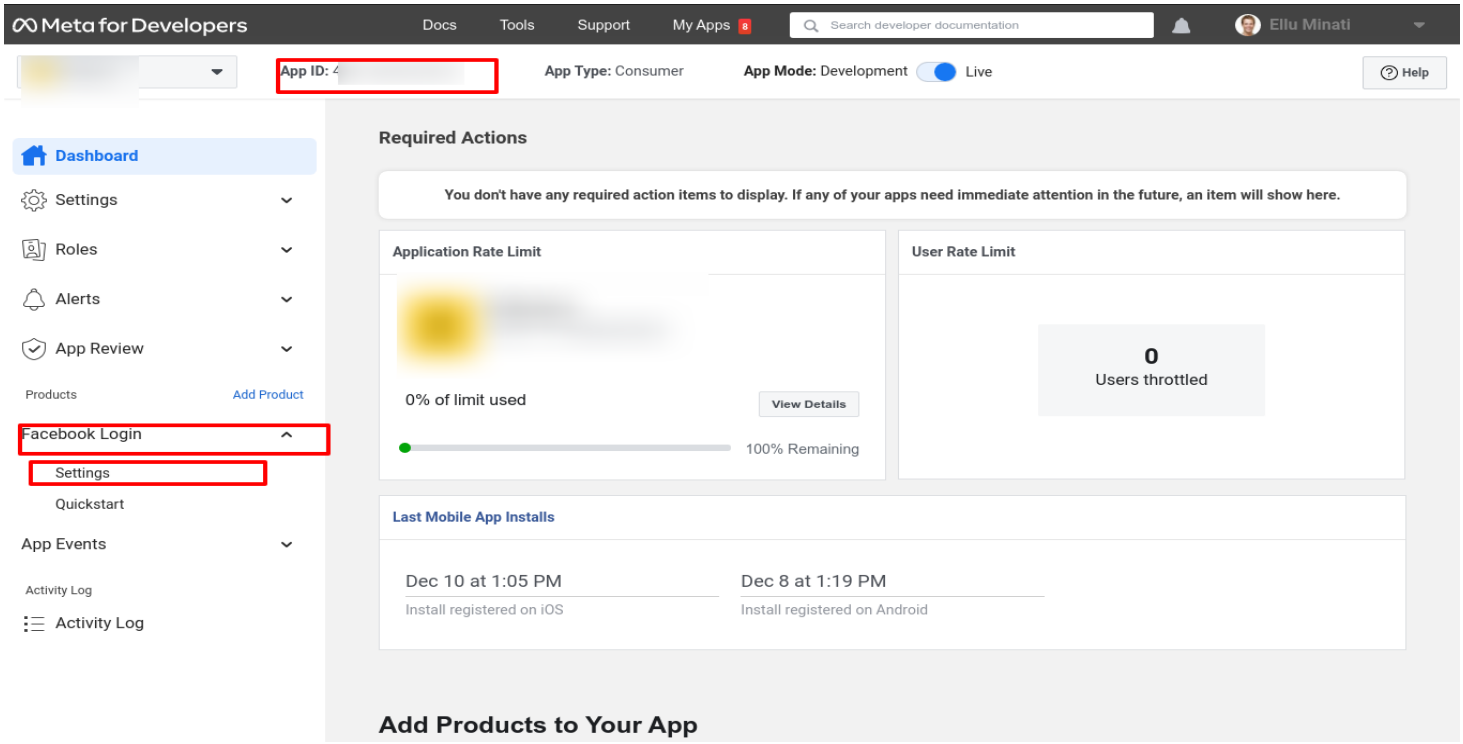
8). Facebook Login.

1) Open Facebook developer console.

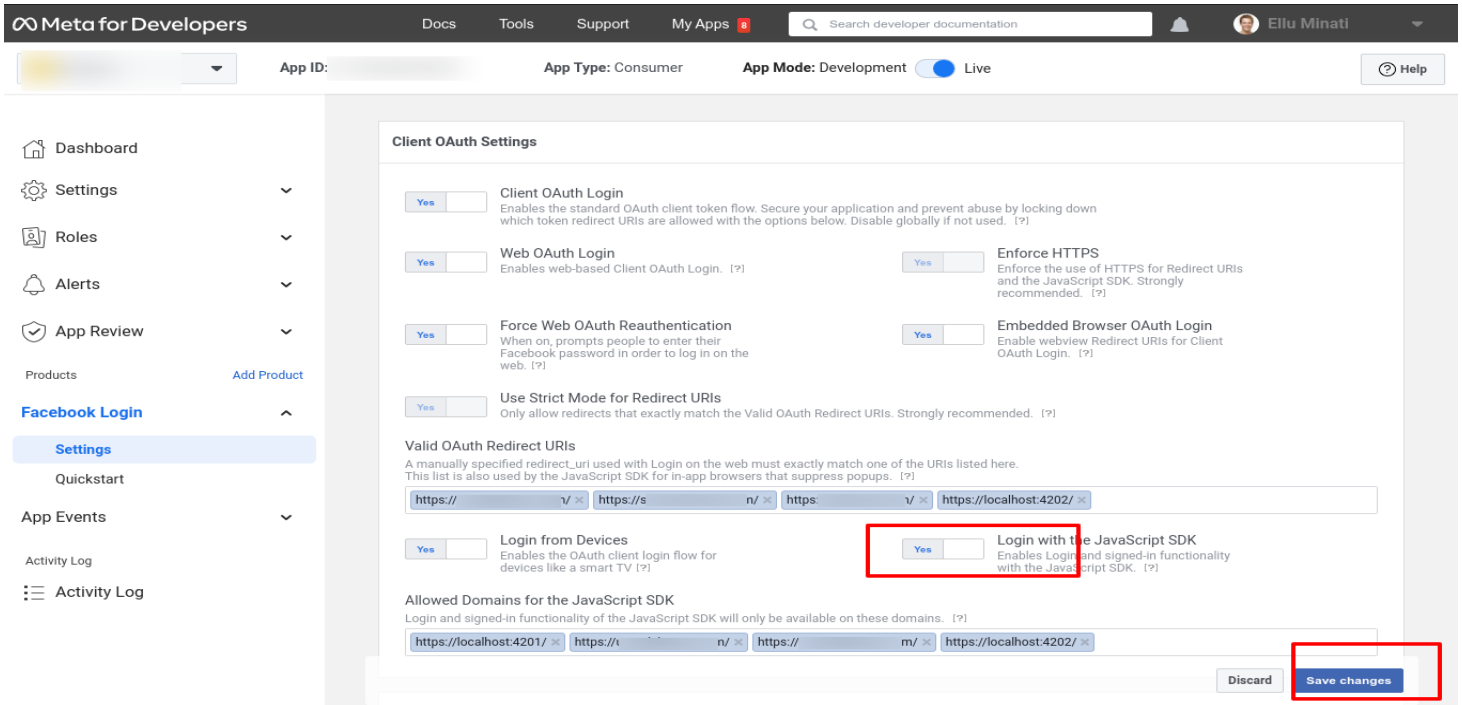
2) Select your project app.



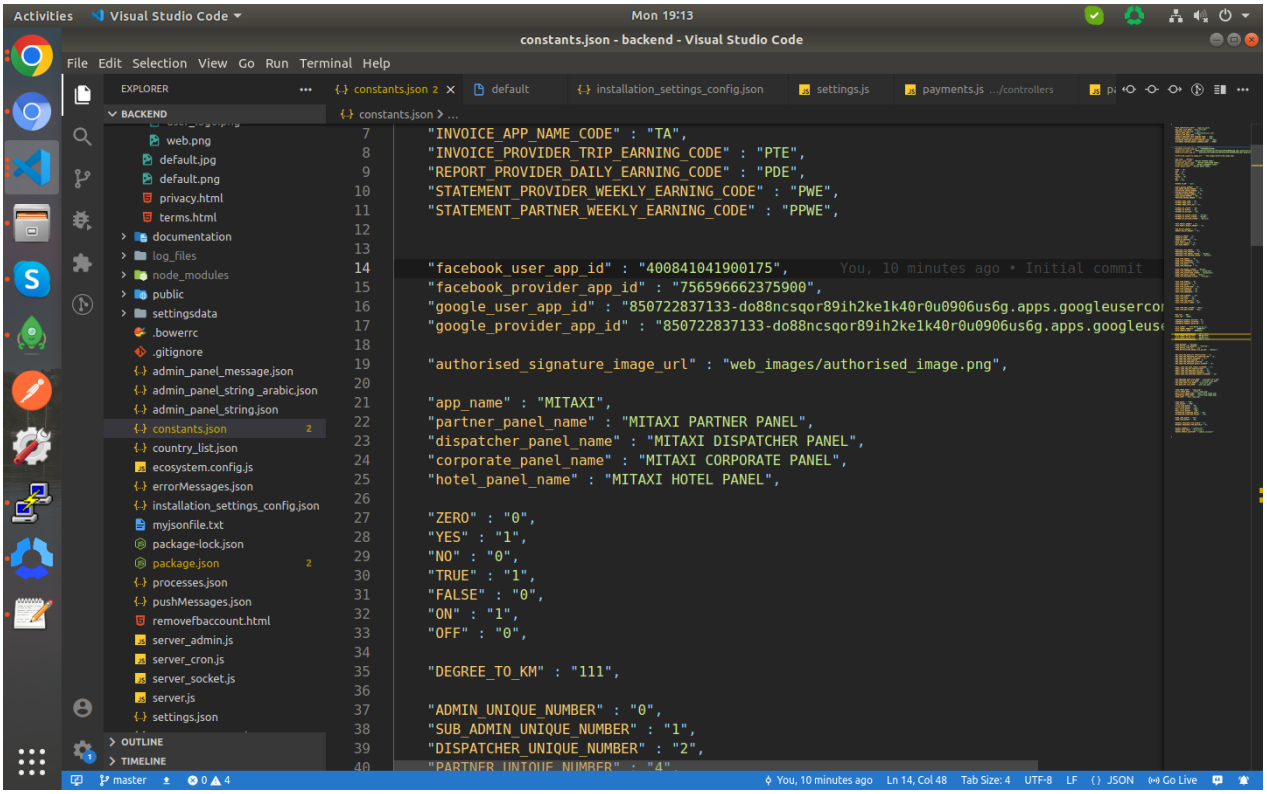
3) Click on FB Login Setting.



4) Add Domains of your project



5) Add user fb id and driver fb id in constant.json file



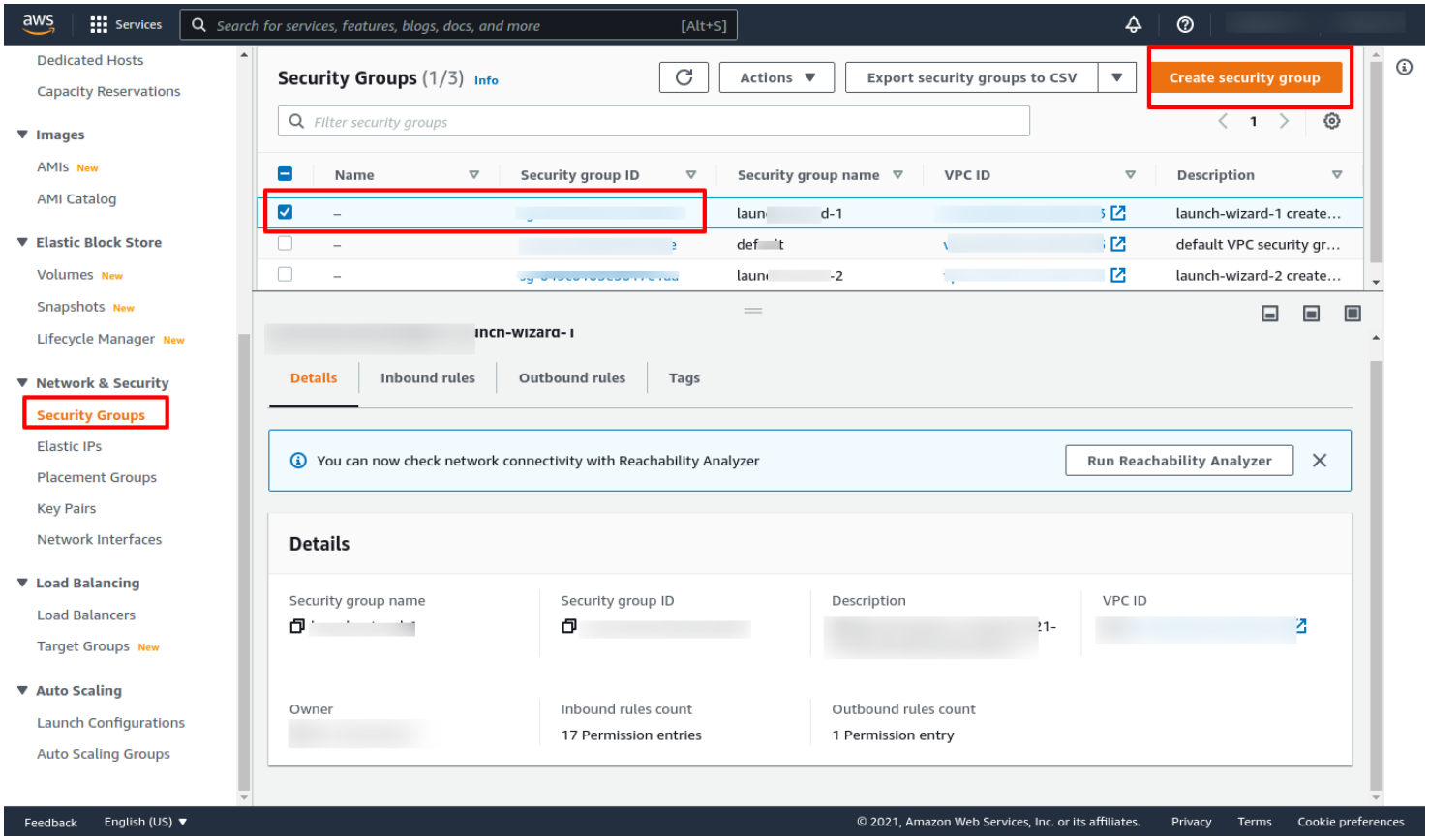
9). Security Groups in AWS.

1) Why security groups are used in AWS?

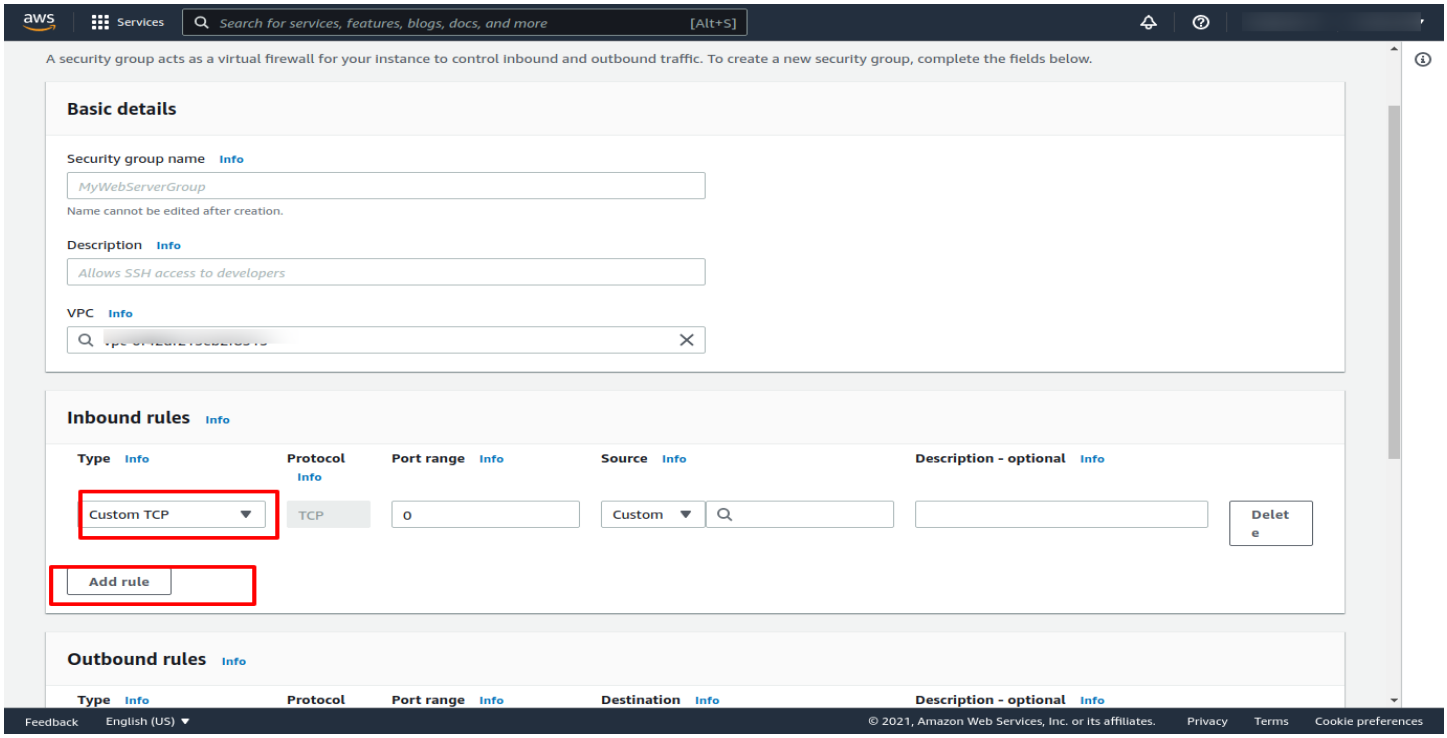
A security group acts as a virtual firewall for your EC2 instances to control incoming and outgoing traffic. Inbound rules control the incoming traffic to your instance, and outbound rules control the outgoing traffic from your instance. If you don't specify a security group, Amazon EC2 uses the default security group.

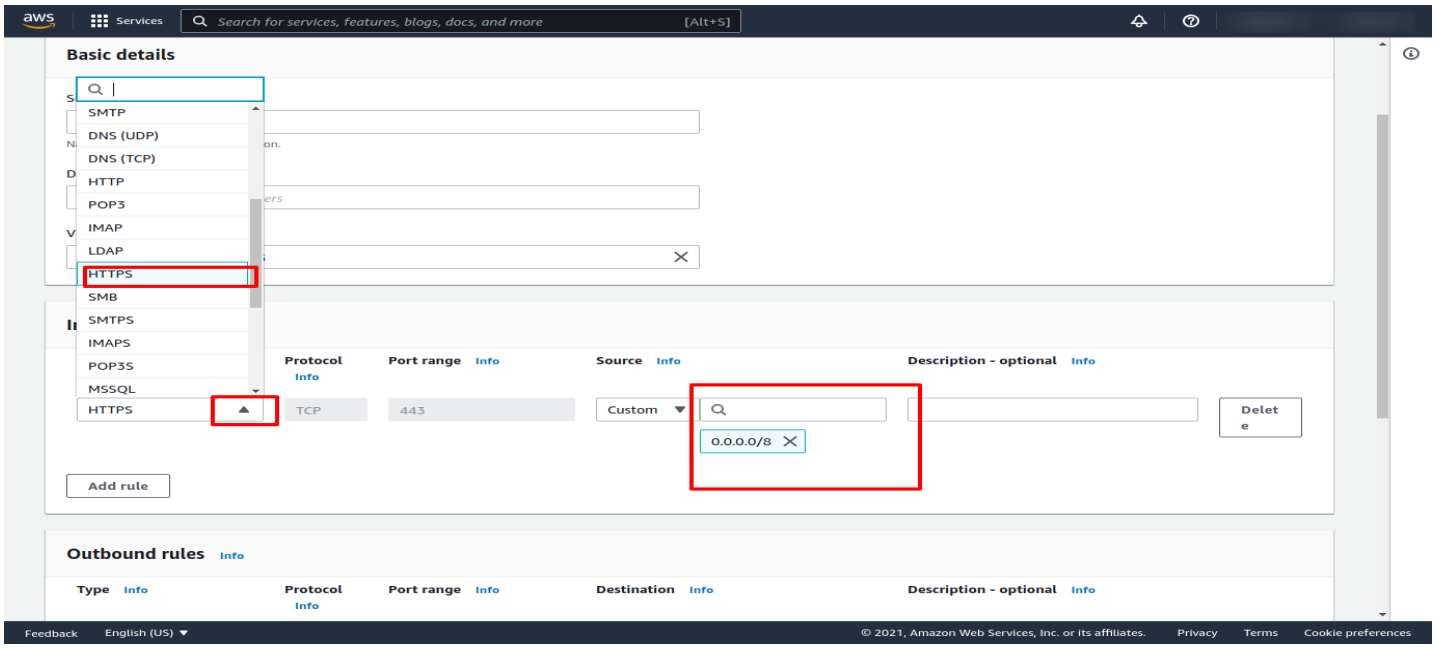
2) In the navigation pane, choose Security Groups.

3) Choose your instance and create a security group.

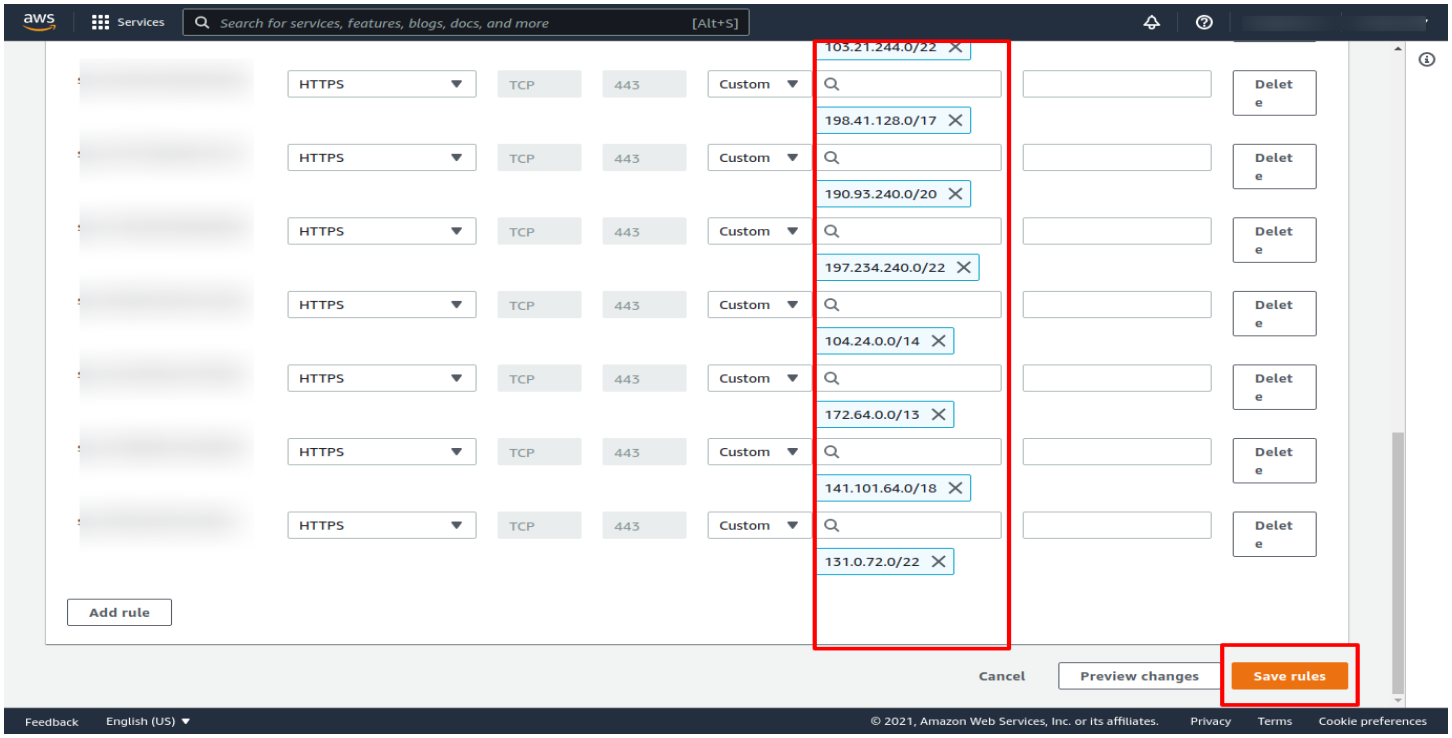


4) First click on add rule and then click on the custom button and select the type HTTPS.





5) Add all custom IPv4 addresses with HTTPS, which are given in this link https://www.cloudflare.com/en-gb/ips/ ,and Save Rules.



10). Note For Folder Structure.

In the delivered code you might not get following files/folders because it will be generated or added after installation.
Some of the folders contain dynamic files/folders.

1) List of empty folders in clone:

- data/partner_document : Uploaded Images For Partner Document
- data/partner_profile : Uploaded Images For Partner Profile
- data/provider_document : Uploaded Images For Driver Document
- data/provider_profile : Uploaded Images For Driver Profile
- data/service_type_images : Uploaded Images For Vehicle Image
- data/service_type_map_pin_images : Uploaded Images For Vehicle Map Pin
- data/user_document : Uploaded Images For User Document
- data/user_profile : Uploaded Images For User Profile

2) The following file/folder will get auto generated after the installation:

- node_modules : Auto Generate After Installation Of Node Dependencies
- package-lock.json : Auto Generate After Installation Of Node Dependencies
- log_files : Auto Generate Files For Logs
- data/xlsheet : Auto Generate xl sheets
- config/data : Auto Generate For Temporary Image Upload Cache

3) The following keys/file are removed that need to be configured from client account that needs to be push in the repository:

- app/ios_push/push_file.p8 - Will be added by admin from client account