Edelivery Installation Documentation

We need to follow a few steps before our web app goes live on the internet. These steps are mentioned below:

1. Purchasing Server On AWS
2. Establishing SSH connection to connect server and installing environment for our code to run.
3. Installation steps and commands
4. Pointing Domain name and SSL certificate
5. Basic setup on admin panel for making application fully functional.

Now let's get started with purchasing the server.

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.

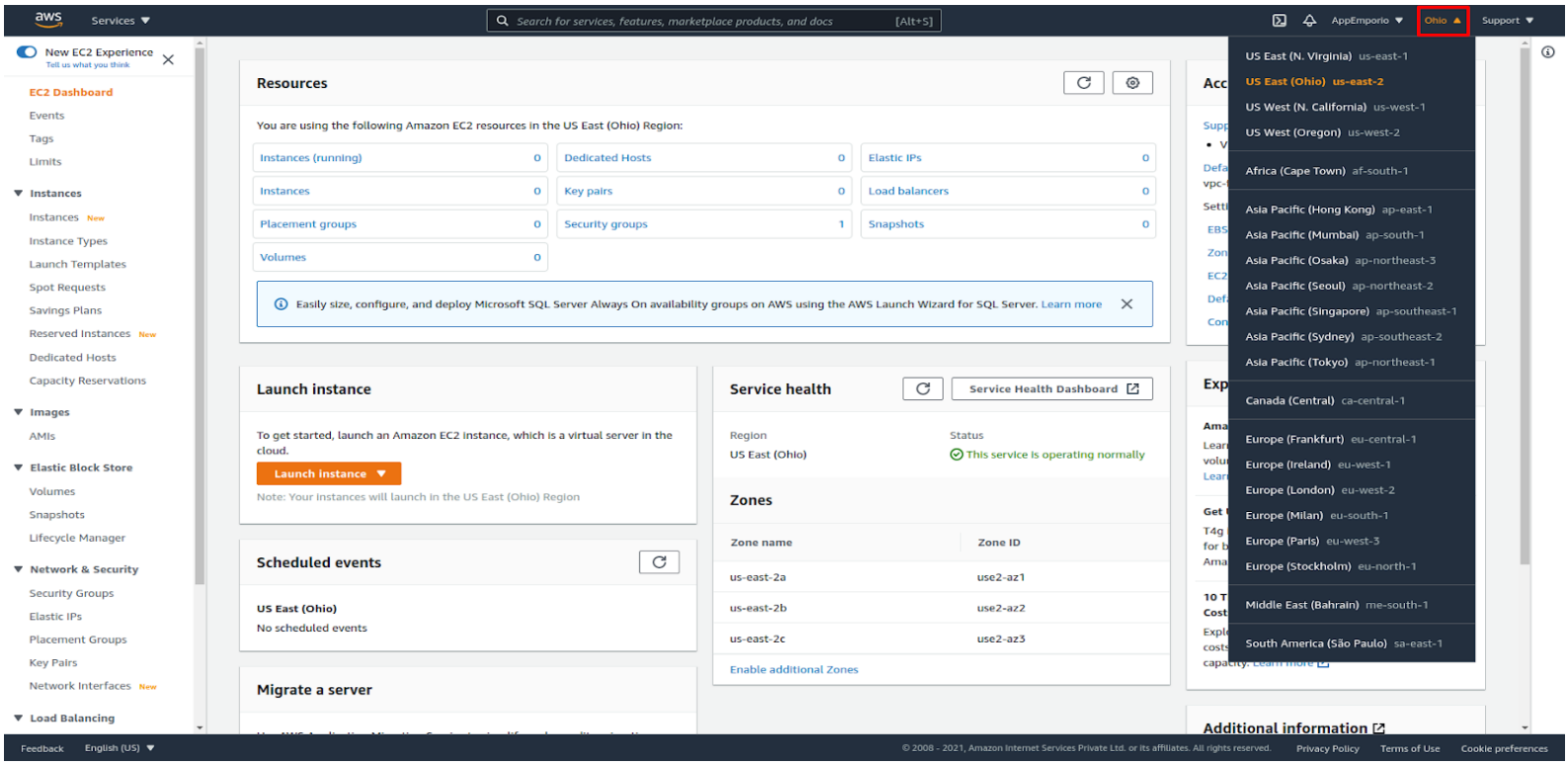
After registration now we need to purchase an instance(server) where we can install our code. In our code, we have 4 different repositories or directories which can be installed and run on, one instance or 4 different instances or we can install backend on one instance and other panels on another instance.

So for that, we need to purchase instances accordingly whether it's 1, 2, or 4.

Now let's see steps to purchase instance

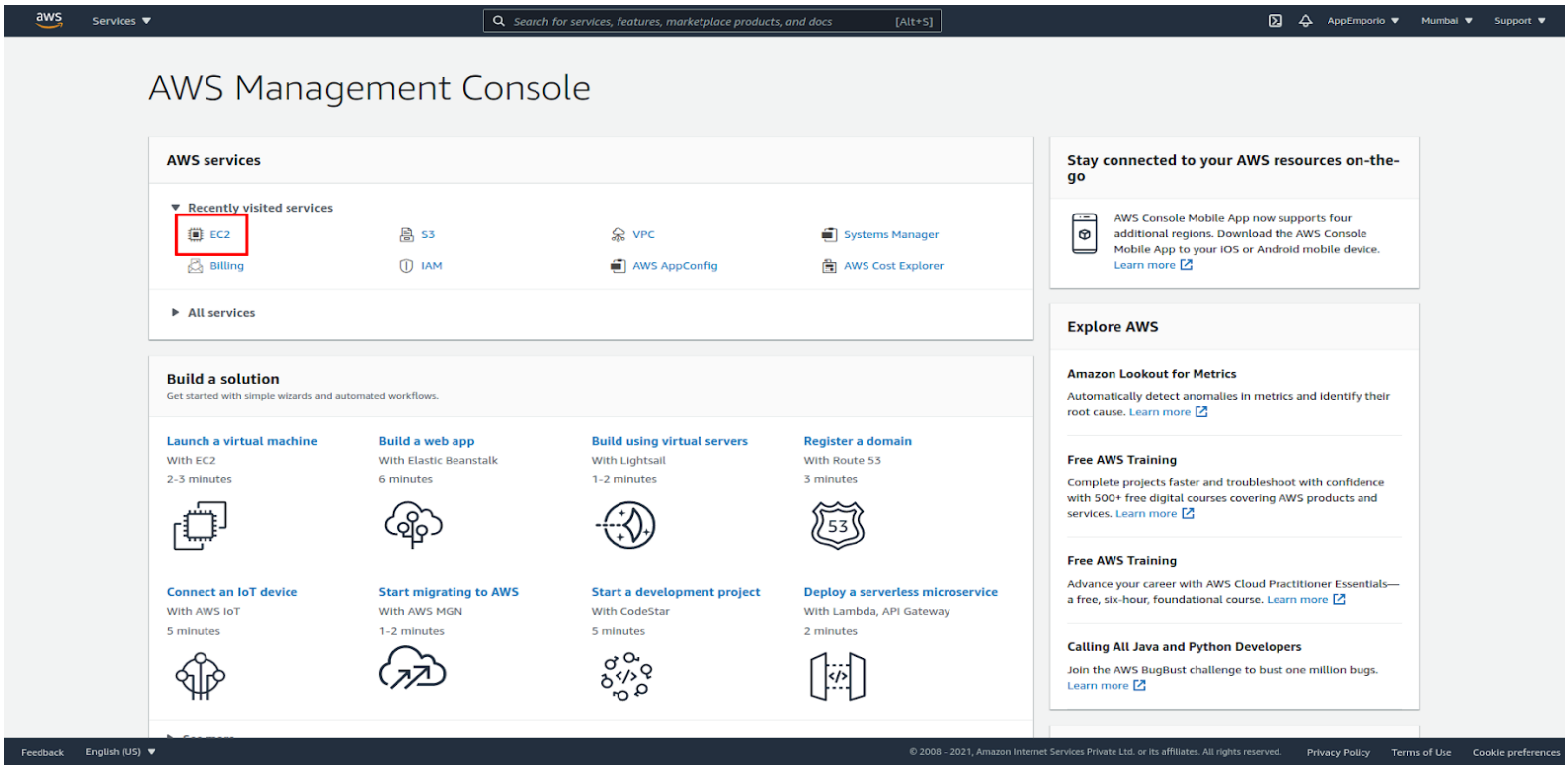
1.1 Selecting region for instance

- We need to select region (from 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.

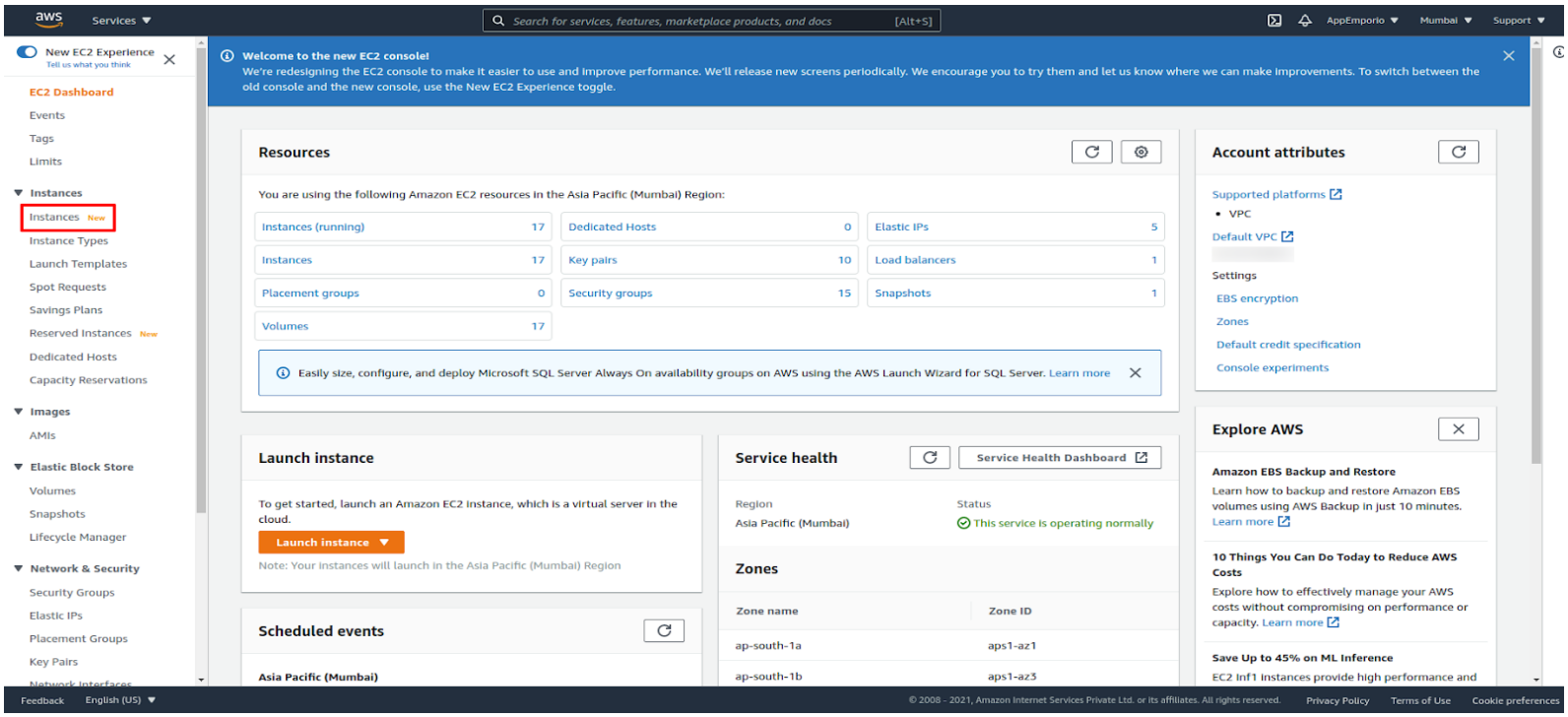


1.2 Purchasing Instance

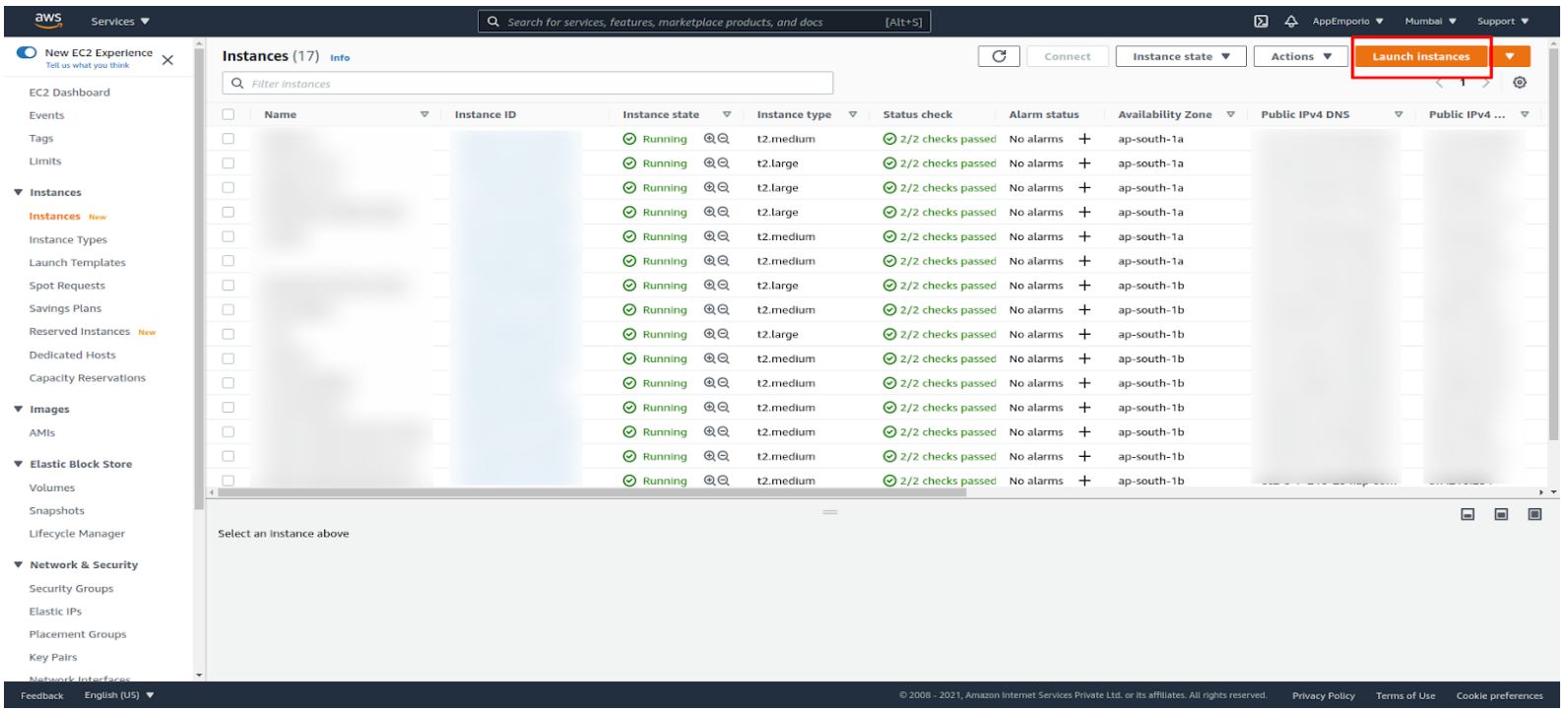
1.2.1: From AWS services select EC2.



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



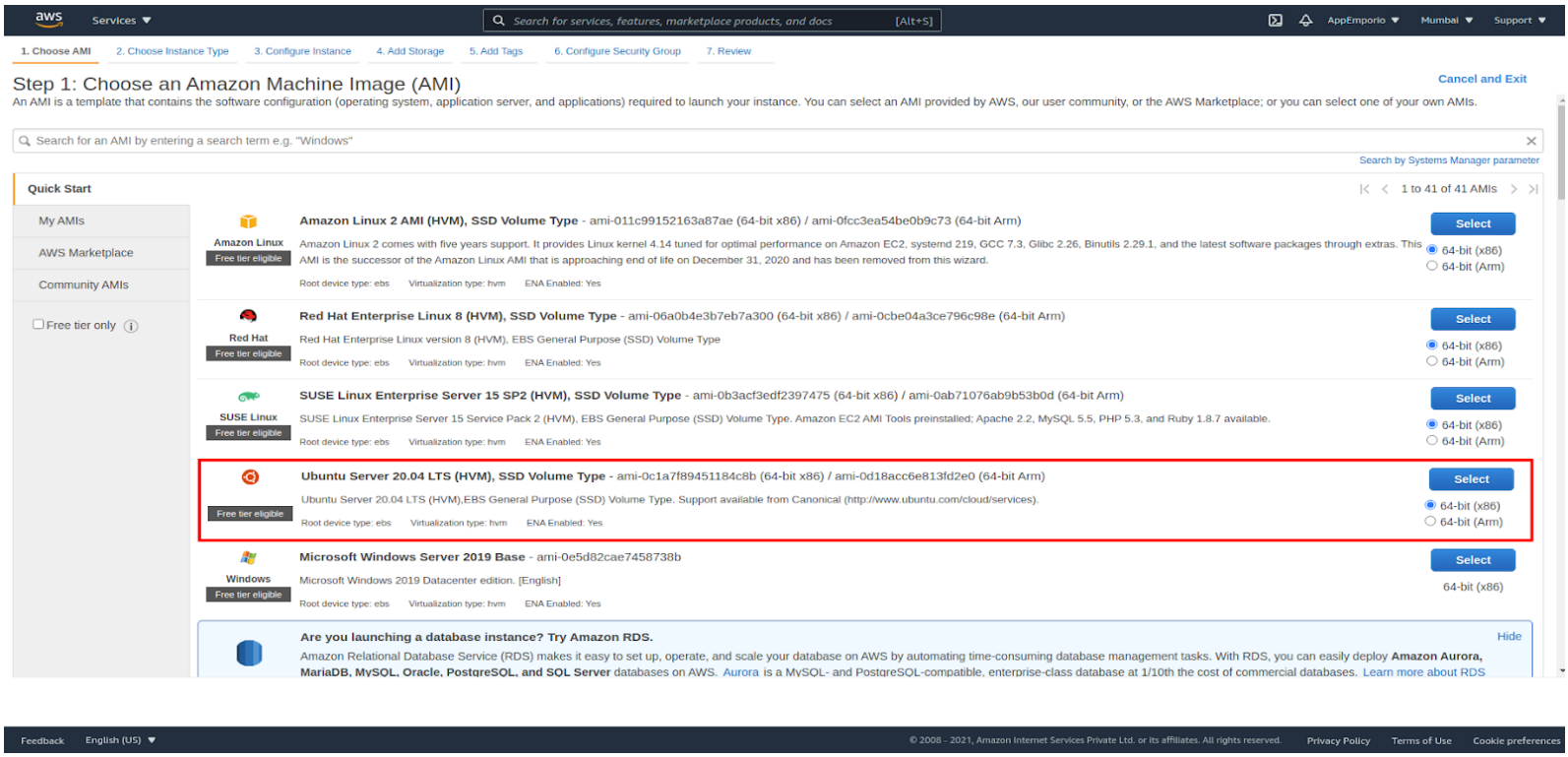
1.2.3: Select Launch from top right corner.



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

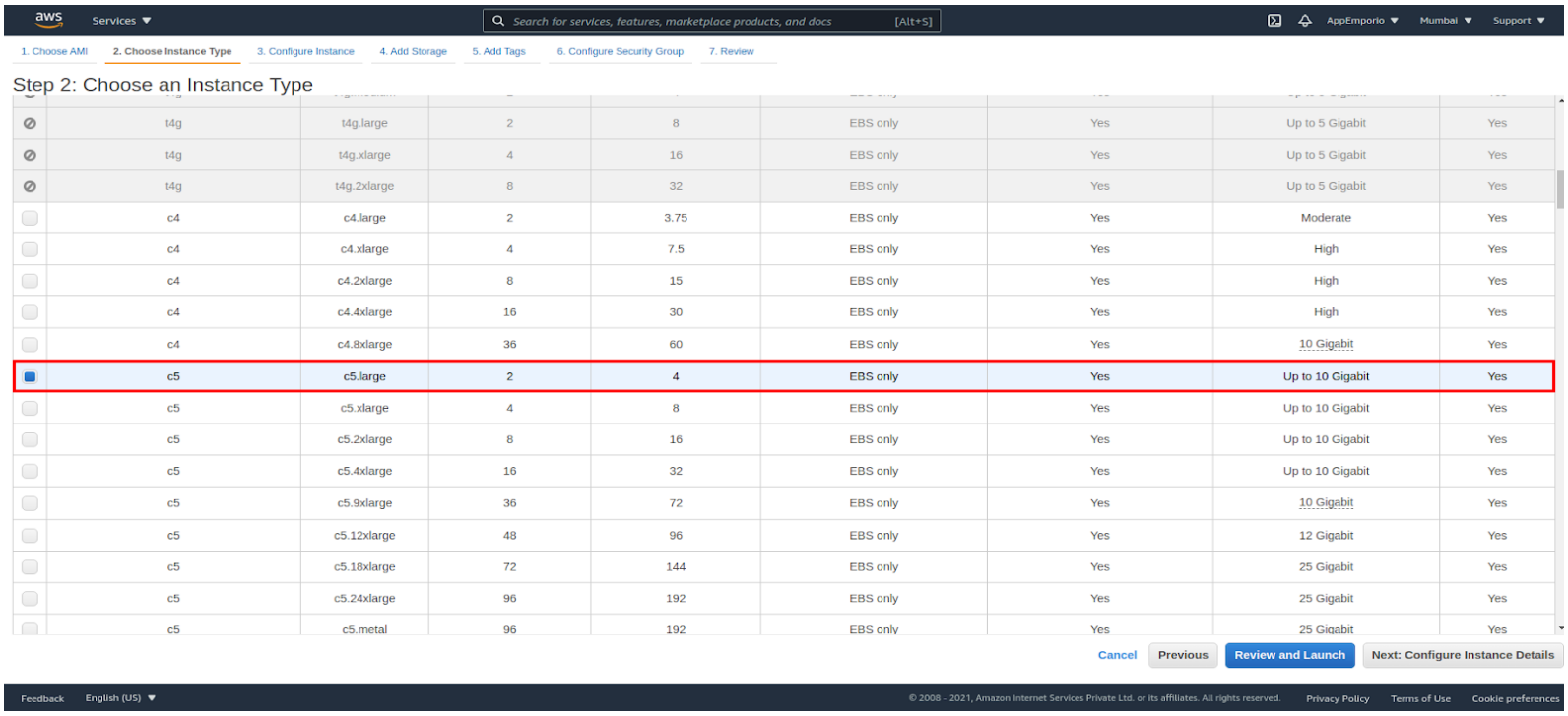
1.2.4(i) Choose an AMI

- 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 and t2.medium for panels.

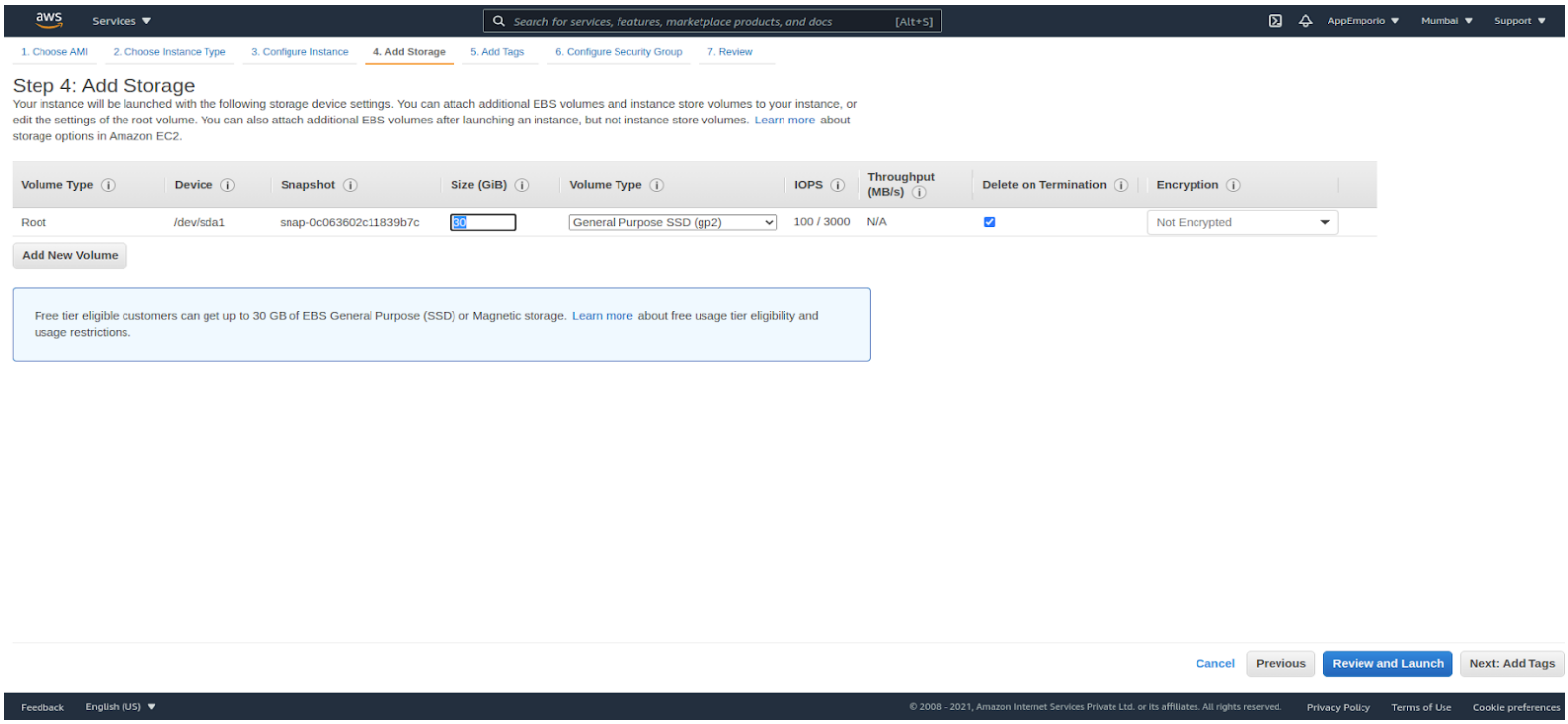


1.2.4(iii) Configure instance details

- 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.

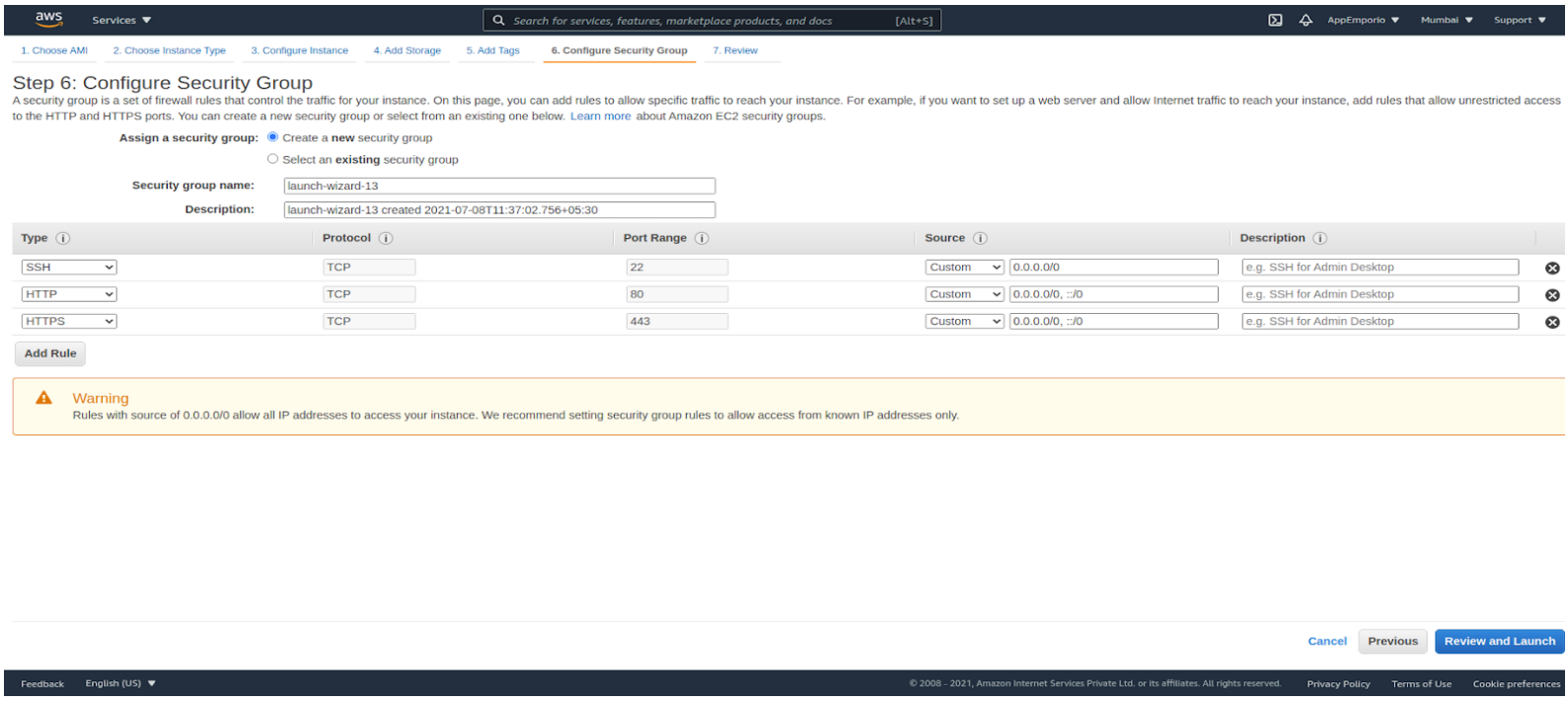


1.2.4(v) Add Tags(Optional)

- 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.

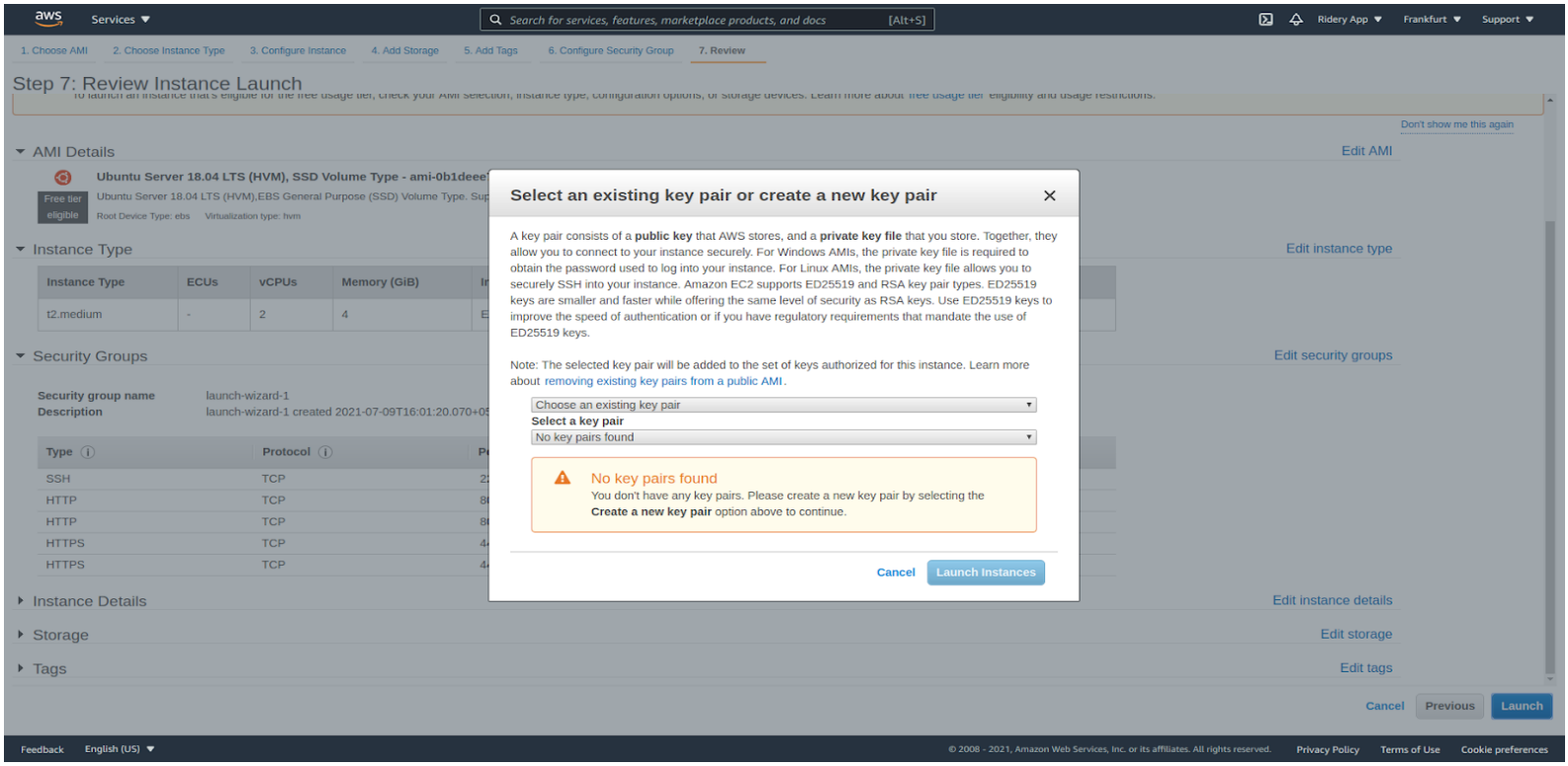


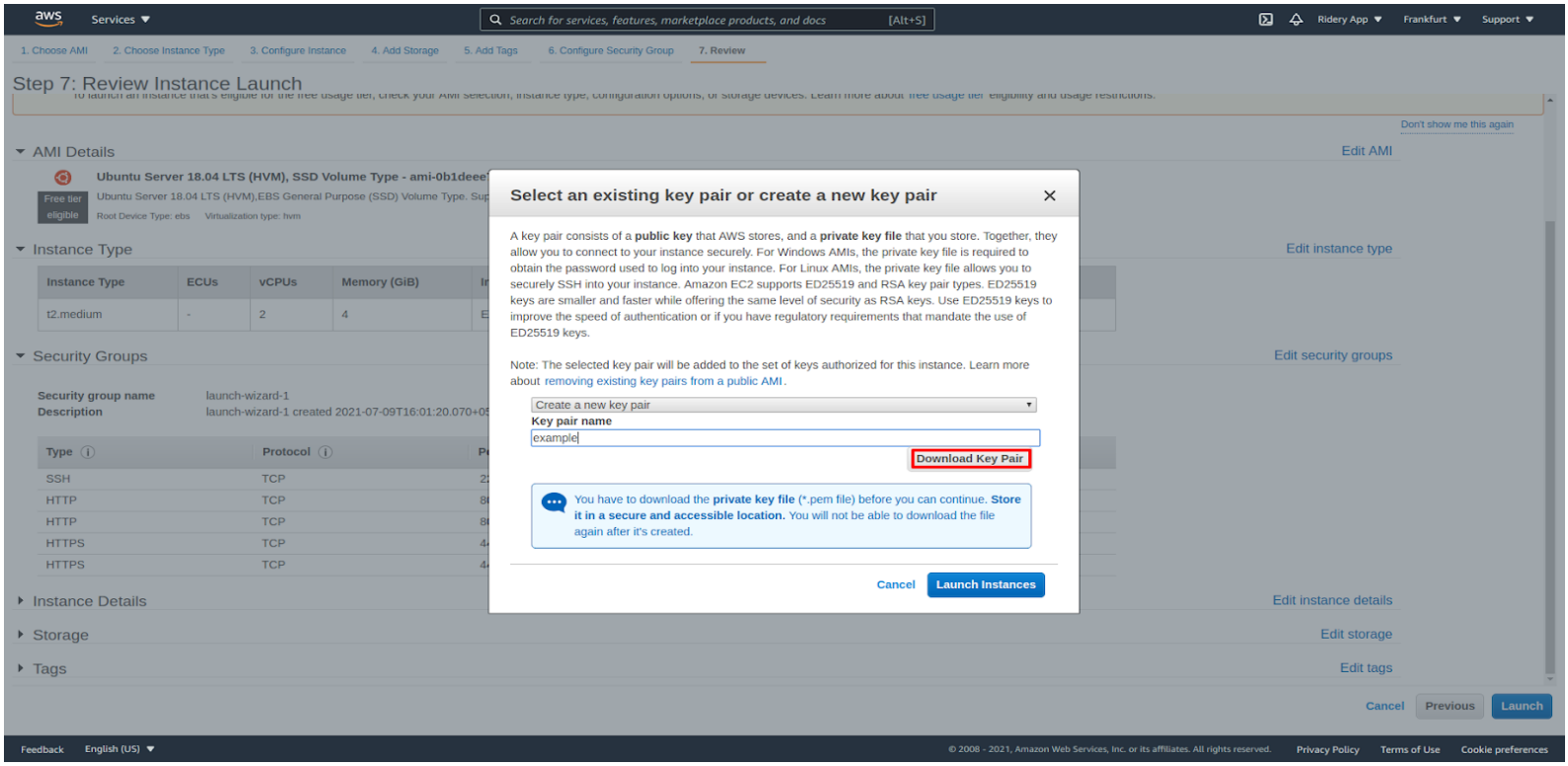
1.2.4(vii) Review

- 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.


Services
Search for services, features, marketplace products, and docs [Alt+S]
Ridery App
Frankfurt
Support
1. Choose AMI
2. Choose Instance Type
3. Configure Instance
4. Add Storage
5. Add Tags
6. Configure Security Group
7. Review
Step 7: Review Instance Launch
Don't show me this again
AMI Details
Edit AMI
Ubuntu Server 18.04 LTS (HVM), SSD Volume Type - ami-0b1deee
Free tier eligible
Ubuntu Server 18.04 LTS (HVM),EBS General Purpose (SSD) Volume Type. Sup
Root Device Type: ebs
Virtualization type: hvm
Instance Type
Edit instance type
Instance Type
ECUs
vCPUs
Memory (GiB)
t2.medium
Security Groups
Edit security groups
Security group name
launch-wizard-1
Description
launch-wizard-1 created 2021-07-09T16:01:20.070+05
Type
Protocol
SSH
TCP
HTTP
TCP
HTTP
TCP
HTTPS
TCP
HTTPS
TCP
Instance Details
Edit instance details
Storage
Edit storage
Tags
Edit tags
Cancel
Previous
Launch
Feedback
English (US)
© 2008 - 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy Policy
Terms of Use
Cookie preferences
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


Services
Search for services, features, marketplace products, and docs [Alt+S]
Ridery App
Frankfurt
Support
1. Choose AMI
2. Choose Instance Type
3. Configure Instance
4. Add Storage
5. Add Tags
6. Configure Security Group
7. Review
Step 7: Review Instance Launch
Don't show me this again
AMI Details
Edit AMI
Ubuntu Server 18.04 LTS (HVM), SSD Volume Type - ami-0b1deee
Free tier eligible
Ubuntu Server 18.04 LTS (HVM),EBS General Purpose (SSD) Volume Type. Sup
Root Device Type: ebs
Virtualization type: hvm
Instance Type
Edit instance type
Instance Type
ECUs
vCPUs
Memory (GiB)
t2.medium
Security Groups
Edit security groups
Security group name
launch-wizard-1
Description
launch-wizard-1 created 2021-07-09T16:01:20.070+05
Type
Protocol
SSH
TCP
HTTP
TCP
HTTP
TCP
HTTPS
TCP
HTTPS
TCP
Instance Details
Edit instance details
Storage
Edit storage
Tags
Edit tags
Cancel
Previous
Launch
Feedback
English (US)
© 2008 - 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy Policy
Terms of Use
Cookie preferences
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. 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 domain server.

2.1 Open PuTTY and setup connection

We can connect just with ip and using security key as well.

Method 1: Connection with IP and root user

- Enter ip address and press open. On the next screen we need to enter username and password.
- After validating now you have access to server files and your SSH connection is established.

Method 2: Connection with security file.

- Enter IP and now got AUTH > SSH from right side nav bar
- Browse and select your security file.
- After pressing open enter user name, if you have created a server from AWS then username would be "ubuntu".
- Now you will be able to access server files and your SSH connection is established.

2.2 Installing and basic environment setup

Now we need to install following environments

1. nodejs
2. MongoDB
3. Nginx
4. pm2

Step 1: Setup node environment on the server

- Following commands will install node.js version 14 in server

  sudo apt-get install python-software-properties
  curl -sL https://deb.nodesource.com/setup_14.x | sudo -E bash

sudo apt-get install -y nodejs
sudo apt-get update

- To check whether node js was installed properly or not, after entering below command in response it will return node version and that is 14, if node was installed properly.

  sudo node -v

Step 2: Installing MongoDB to server for DB environment

- Following commands will install MongoDB version 4.4 in server

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 -

- Use below command if selected AMI was Ubuntu 20.04 (Focal)
  echo "deb [arch=amd64,arm64] https://repo.mongodb.org/apt/ubuntu focal/mongodb-org/4.4 multiverse" | sudo tee /etc/apt/sources.list.d/mongodb-org-4.4.list

- Use below command if selected AMI was Ubuntu 18.04 (Bionic)
  echo "deb [arch=amd64,arm64] https://repo.mongodb.org/apt/ubuntu bionic/mongodb-org/4.4 multiverse" | sudo tee /etc/apt/sources.list.d/mongodb-org-4.4.list

- Use below command if selected AMI was Ubuntu 16.04 (Xenial)
  echo "deb [arch=amd64,arm64] https://repo.mongodb.org/apt/ubuntu xenial/mongodb-org/4.4 multiverse" | sudo tee /etc/apt/sources.list.d/mongodb-org-4.4.list

  sudo apt-get update
  sudo apt-get install -y mongodb-org

- To check whether MongoDB was installed properly or not, after entering the below command in response it will return the db version and that is 4.4, if node was installed properly.

  mongod -version

- Now if MongoDB was installed successfully then now we need to start mongod so that we can use it when our application is running.

  sudo service mongod start

- Below command will help you to check if mongodb is running or not, if it is running then it will return Active: active in green text.

  sudo service mongod status

Step 3: Setup Nginx

- Following commands will install Nginx
  sudo apt-get install -y nginx
  systemctl status nginx

- Below command will start Nginx
  sudo service start nginx

- Below command will help you to check if nginx is running or not, if it is running then it will return Active: active in green text.

  sudo service nginx status

Step 4: Installing pm2

- Following commands will install pm2

  sudo npm install pm2 -g

Step 5: installing Git and Cloning code from git repo to the server and giving some basic permissions to directories

- Below command will install git in our server

  sudo apt-get install git

- Below commands will create a directory at /var/www/html and will give some required permission for access to this directory when our application is live.

  cd /var/www/html
  sudo mkdir <projname>
  sudo chmod -R 777 <projname>
  cd <projname>

- Now in this project directory we will clone code from git repository to our server. If we are going to use multiple servers then clone code that is going to be hosted from this clone. For example, we are going to use this server only for backend then only clone backend and rest on their server.

  sudo git clone <admin panel git repository link>
  sudo git clone <store panel git repository link>
  sudo git clone <user panel git repository link>
  sudo git clone <backend git repository link>

3. Installation steps and commands

Now our code is on the server, we need to install some dependencies of our code.

3.1 Installing Dependencies

First we will move to directory where we cloned our code, and that is /var/www/html/<projectname>
If we have installed all panels and backend in the same server then we will find several directories. Move into any one of them. (let's say admin_panel)

- One of the below code will install all dependencies required for admin panel

    sudo npm install or *sudo npm i*

3.2 Setting up our database

Now we need to enter a basic Database to our db so we can make some functionalities ready to use.

For that we need to change the backend, so if you are on another server then skip this step.

- Now lets move to settings_data directory in the backend. Use the below commands.

    cd setting_data

- Before entering required data let's see what will be our db name. Below code will lead to that file and will open that file highlighted portion is our database name.

    *cd /var/www/html/**<projectname>**/config/db*
    nano development.js

- Below commands will enter some required data to our Database.

*sudo mongoimport --db **<dbName>** --collection email_details --file email_details.json*
*sudo mongoimport --db **<dbName>** --collection sms_details --file sms_details.json*
*sudo mongoimport --db **<dbName>** --collection delivery_types --file delivery_types.json*
*sudo mongoimport --db **<dbName>** --collection admins --file admins.json*
*sudo mongoimport --db **<dbName>** --collection image_settings --file image_settings.json*
*sudo mongoimport --db **<dbName>** --collection installation_settings --file installation_settings.json*
*sudo mongoimport --db **<dbName>** --collection payment_gateways --file payment_gateways.json*
*sudo mongoimport --db **<dbName>** --collection settings --file settings.json*
*sudo mongoimport --db **<dbName>** --collection sms_gateways --file sms_gateways.json*

4. Nginx Configurations, Pointing Domain name and SSL certificate

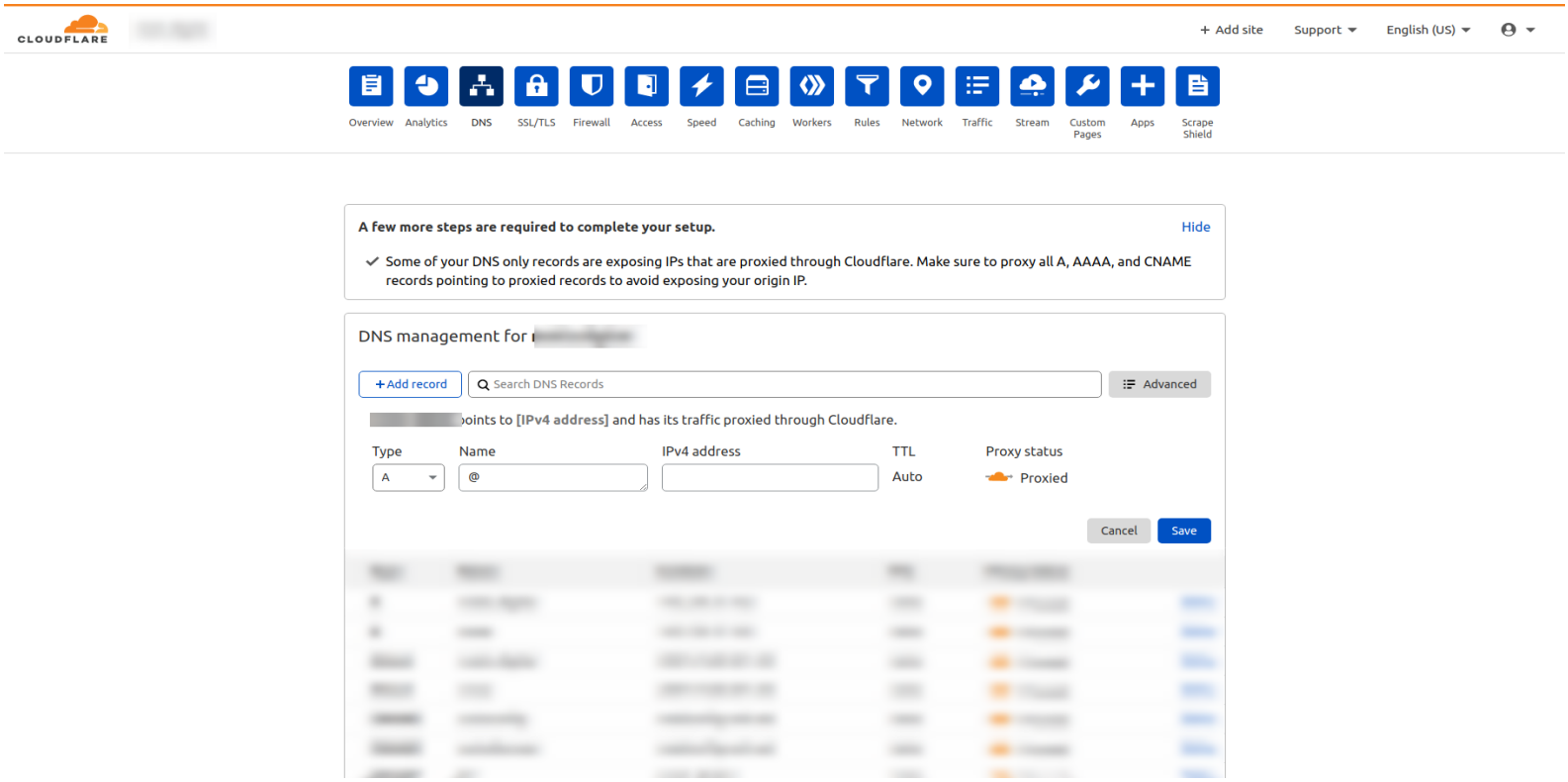
Now as our code is now running on the server, we can't open our application from search engines or web browsers using domain names.

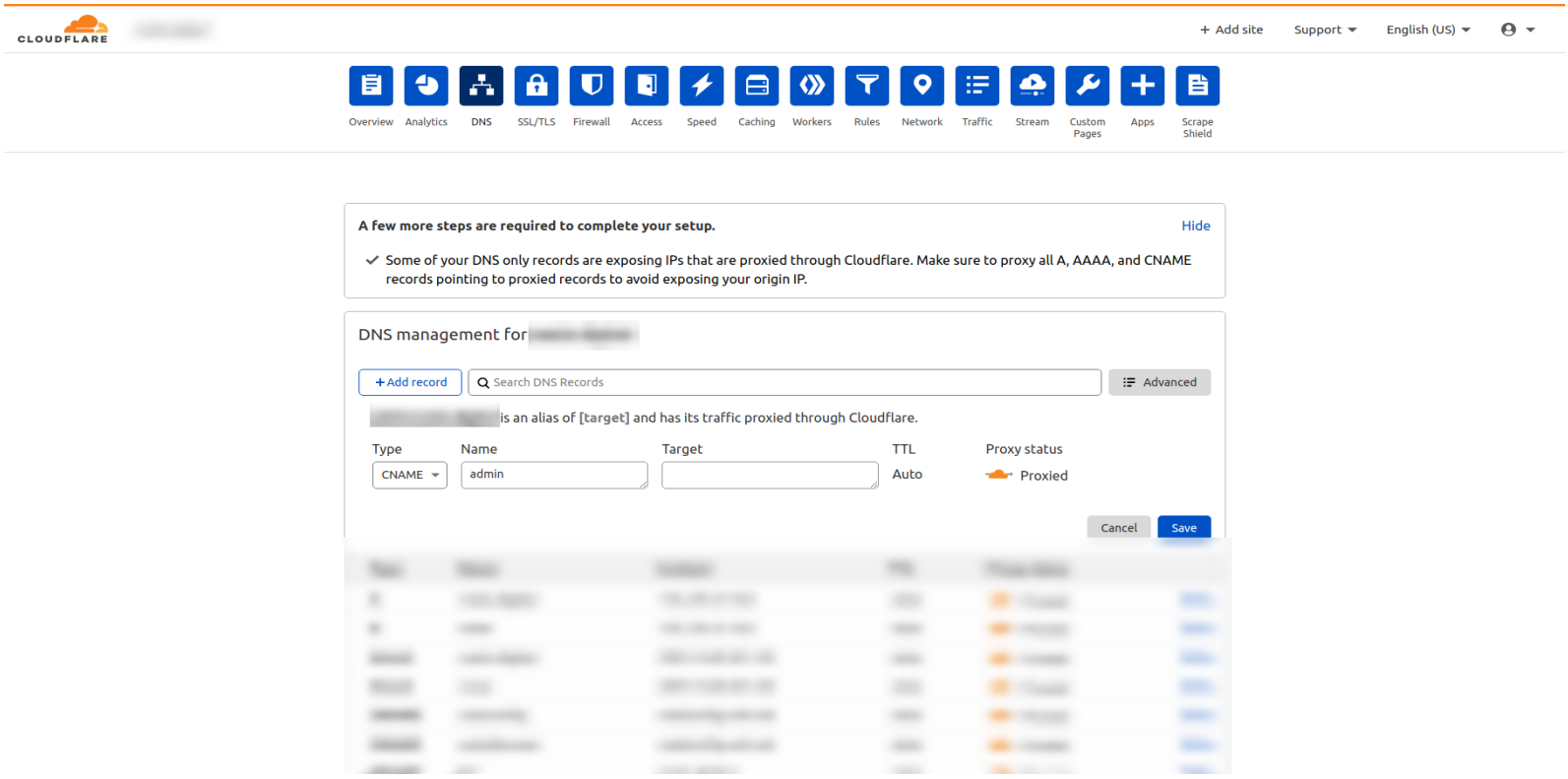
For that we need to setup few things and that are:

1. Domain Name
2. SSL certificate
3. NGINX

4.1 Domain Name(DNS)

For Domain configuration we are using Cloudflare (https://www.cloudflare.com). Register your account and got to DNS section and click Add Record. On given input field in type section select A (A recordes) and in name enter @ and in IPv4 enter IP address of our server, then click Save. For sub domains click add new record again and in type section select CNAME and in name section enter admin and in IPv4 enter @ (it indirectly point to our server ip address). Add 3 more sub domains for api, store and for user. Generally for our user panels we uses 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

- 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

4.3 Setting up NGINX configurations

Now before going further let's check if nginx was installed and running in our server properly or not. We have our server ip address, so let's use it and enter that on broweser.like http://<IPaddress>, and now we can see NGINX default server. If not then there is some issue, please follow Step 3 from Establishing SSH connection to connect server and installing environment for our code to run section.

Nginx by default allows you to transfer files with 1MB of size. In some cases we need to upload and use images/files which are larger than 1MB. So for that we need to configure that first.

- First we need to make changes in the nginx.config file. So now lets move to that files directory.

  cd /etc/nginx
  sudo nano nginx.config

- Now the nginx.config file is open and we need to add the below lines. This line will allow us to transfer 25MB files. We can set any number here.

  ```
  http {
     client_max_body_size 25M;

     //other lines...
  }
  ```

- Now let's configure our server and domain using nginx. First we need to head to the directory where our nginx files will be lying and that path is */etc/nginx/sites-available*.

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

- Now we have nging opened and some code lying there which is pointed to the screen we saw on the browser. So now let's point it to our code.

- First remove the whole code in that nginx file and then copy belowe code and paste to that nginx file.

  ```
  server {
     listen 80;
     server_name <domain name>;
     location / {
        proxy_pass http://localhost:8000;
        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; # managed by Certbot
    ssl_certificate "/etc/nginx/ssl/<originkey.pem>";
    ssl_certificate_key "/etc/nginx/ssl/<privatekey.pem>";

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

- Above File is for backend, we can use this same file for other panels also just change below things

  Admin panel
  http://localhost:8000 > http://localhost:3000
  domain name registered for admin panel

  Store panel
  http://localhost:8000 > http://localhost:5000
  domain name registered for store panel

  User panel
  http://localhost:8000 > http://localhost:4000
  domain name registered for user panel

5. Basic setup on admin panel for making application fully functional.

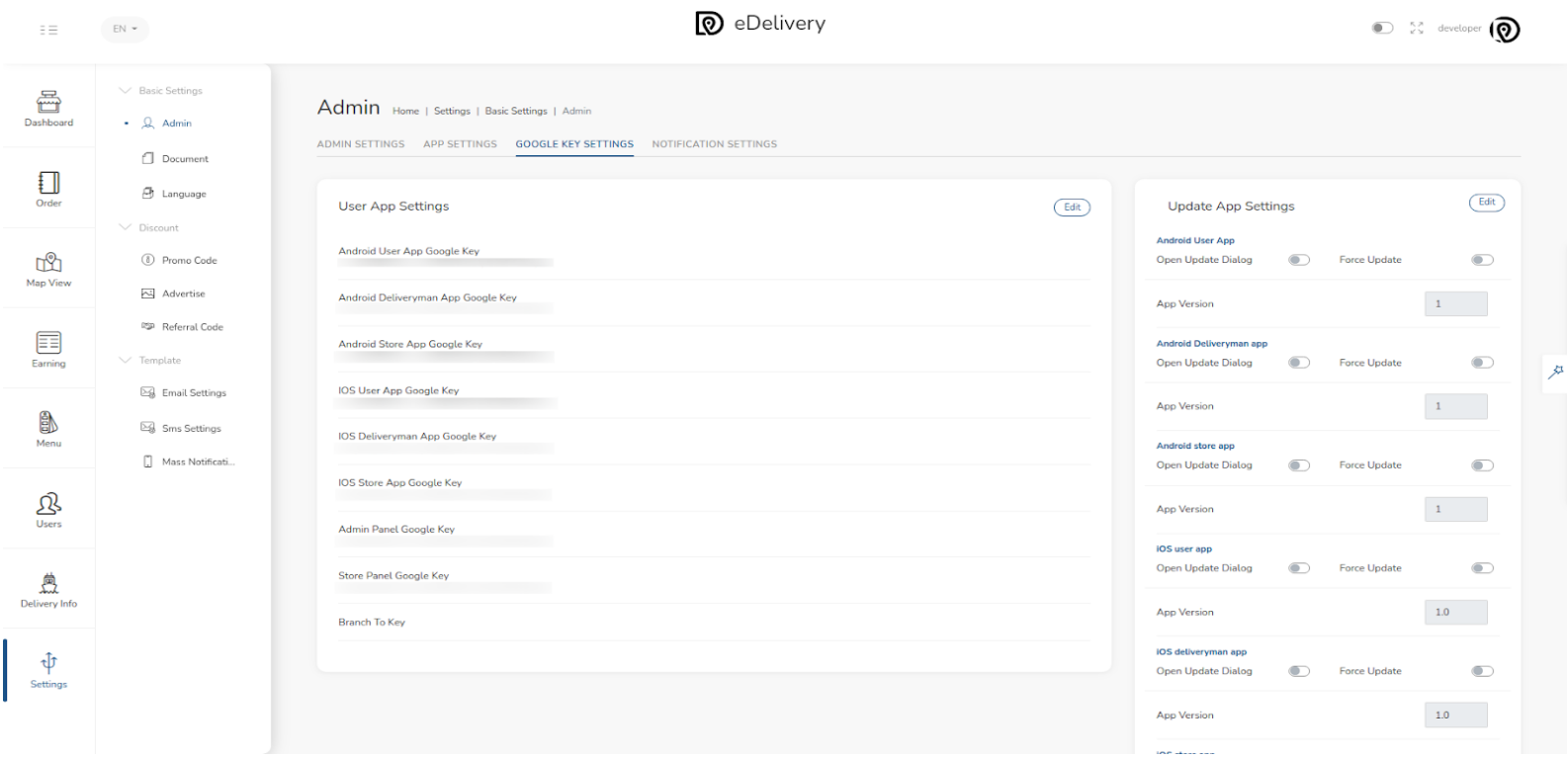
Now our applications are live so now we need to do some configurations to make our application fully functional

We need to set Google api keys to make maps working.

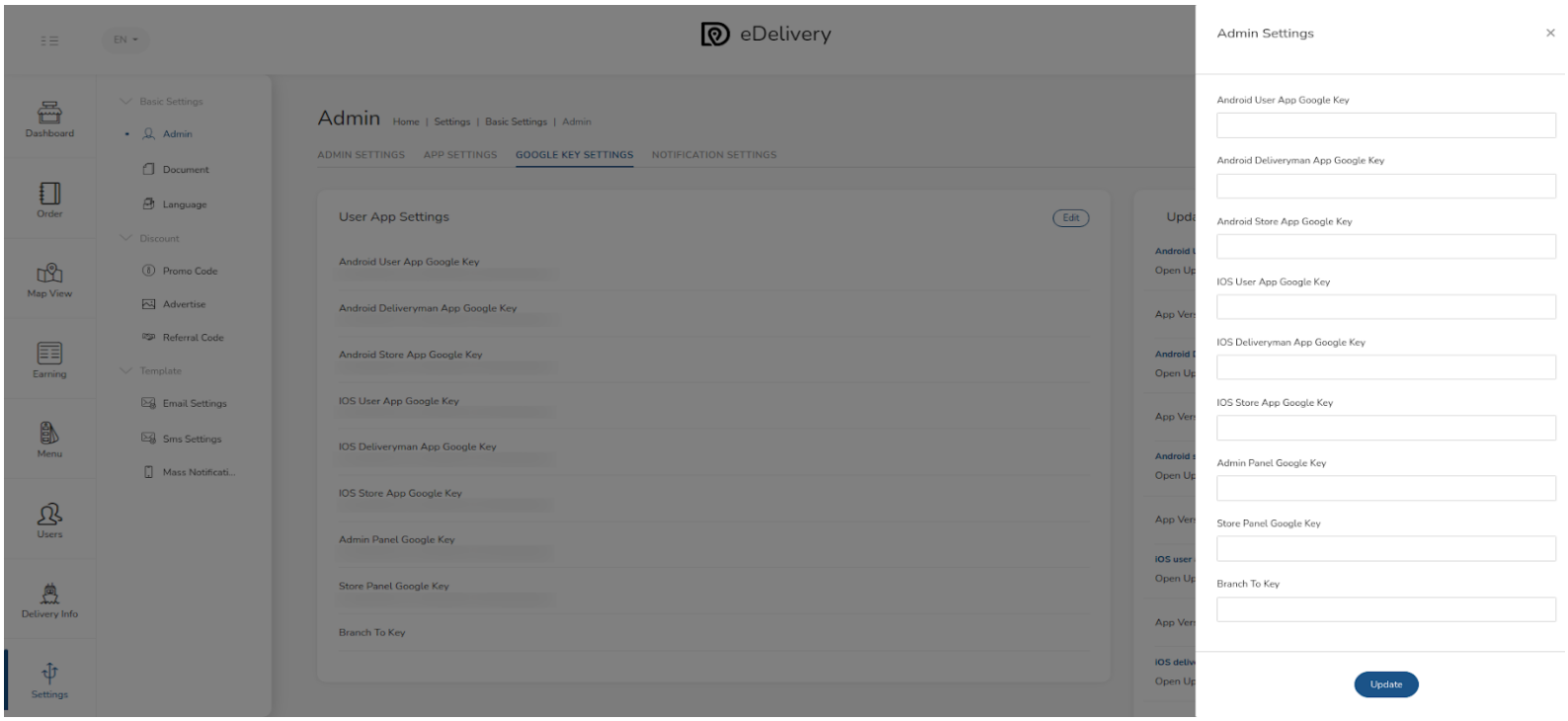
As shown in image head to app settings and press edit button to set you google api keys

Step 1 : Set up Google api keys

- In the admin panel, navigate to settings, admin and then google api key settings.

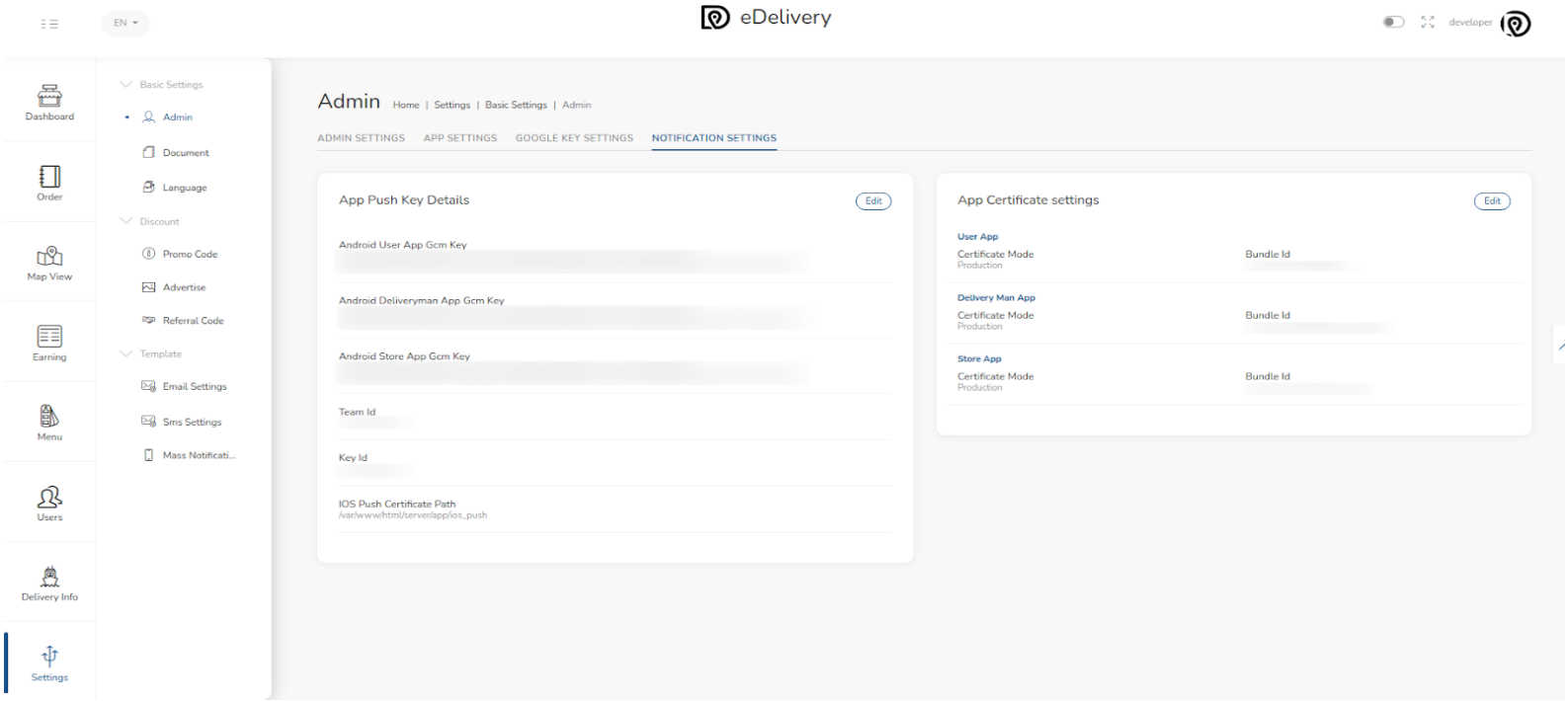


- Press edit button located at top-right corner in User App settings
- Place your google api keys in appropriate fields.



Step 2: Set up Notification Firebase Keys

- In the admin panel, navigate to settings, admin and then Notification settings



- Now press the edit button located at the top-right corner of the App Push Key Details section.
- Now enter your firebase keys in appropriate fields.
- Select your p8 file for ios push