

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.

Resources

You are using the following Amazon EC2 resources in the US East (Ohio) Region:

Instances (running)	0	Dedicated Hosts	0	Elastic IPs	0
Instances	0	Key pairs	0	Load balancers	0
Placement groups	0	Security groups	1	Snapshots	0
Volumes	0				

Launch instance

To get started, launch an Amazon EC2 instance, which is a virtual server in the cloud.

[Launch instance](#)

Note: Your instances will launch in the US East (Ohio) Region

Scheduled events

US East (Ohio)
No scheduled events

Migrate a server

Service health

Region: US East (Ohio) Status: ✔ This service is operating normally

Zones

Zone name	Zone ID
us-east-2a	use2-az1
us-east-2b	use2-az2
us-east-2c	use2-az3

[Enable additional Zones](#)

Additional information

1.2 Purchasing Instance

1.2.1: From AWS services select EC2.

AWS services

Recently visited services

- EC2**
- Billing
- S3
- IAM
- VPC
- AWS AppConfig
- Systems Manager
- AWS Cost Explorer

Build a solution

Get started with simple wizards and automated workflows.

- Launch a virtual machine**
With EC2
2-3 minutes
- Build a web app**
With Elastic Beanstalk
6 minutes
- Build using virtual servers**
With Lightsail
1-2 minutes
- Register a domain**
With Route 53
3 minutes
- Connect an IoT device**
With AWS IoT
5 minutes
- Start migrating to AWS**
With AWS MGN
1-2 minutes
- Start a development project**
With CodeStar
5 minutes
- Deploy a serverless microservice**
With Lambda, API Gateway
2 minutes

Stay connected to your AWS resources on-the-go

AWS Console Mobile App now supports four additional regions. Download the AWS Console Mobile App to your iOS or Android mobile device. [Learn more](#)

Explore AWS

Amazon Lookout for Metrics
Automatically detect anomalies in metrics and identify their root cause. [Learn more](#)

Free AWS Training
Complete projects faster and troubleshoot with confidence with 500+ free digital courses covering AWS products and services. [Learn more](#)

Free AWS Training
Advance your career with AWS Cloud Practitioner Essentials—a free, six-hour, foundational course. [Learn more](#)

Calling All Java and Python Developers
Join the AWS BugBust challenge to bust one million bugs. [Learn more](#)

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

The screenshot shows the AWS Management Console with the EC2 Dashboard. The left navigation bar has the 'Instances' link highlighted with a red box. The main content area displays a welcome message for the new EC2 console, a summary of resources in the Asia Pacific (Mumbai) Region, and a 'Launch instance' button. The 'Service health' section indicates that the service is operating normally.

Resources

You are using the following Amazon EC2 resources in the Asia Pacific (Mumbai) Region:

Resource	Count
Instances (running)	17
Dedicated Hosts	0
Elastic IPs	5
Instances	17
Key pairs	10
Load balancers	1
Placement groups	0
Security groups	15
Snapshots	1
Volumes	17

Launch instance

To get started, launch an Amazon EC2 Instance, which is a virtual server in the cloud.

[Launch instance](#)

Note: Your instances will launch in the Asia Pacific (Mumbai) Region

Service health

Region: Asia Pacific (Mumbai) Status: ✔ This service is operating normally

Zones

Zone name	Zone ID
ap-south-1a	aps1-az1
ap-south-1b	aps1-az3

Account attributes

Supported platforms

- VPC

Default VPC

Settings

- EBS encryption

Zones

Default credit specification

Console experiments

Explore AWS

Amazon EBS Backup and Restore

Learn how to backup and restore Amazon EBS volumes using AWS Backup in just 10 minutes. [Learn more](#)

10 Things You Can Do Today to Reduce AWS Costs

Explore how to effectively manage your AWS costs without compromising on performance or capacity. [Learn more](#)

Save Up to 45% on ML Inference

EC2 Inf1 instances provide high performance and

1.2.3: Select Launch from top right corner.

The screenshot shows the AWS Management Console with the EC2 Instances page. The 'Launch Instances' button in the top right corner is highlighted with a red box. The main content area displays a list of 17 running EC2 instances.

Instances (17)

[Filter instances](#)

	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...
<input type="checkbox"/>			Running	t2.medium	2/2 checks passed	No alarms	ap-south-1a		
<input type="checkbox"/>			Running	t2.large	2/2 checks passed	No alarms	ap-south-1a		
<input type="checkbox"/>			Running	t2.large	2/2 checks passed	No alarms	ap-south-1a		
<input type="checkbox"/>			Running	t2.large	2/2 checks passed	No alarms	ap-south-1a		
<input type="checkbox"/>			Running	t2.medium	2/2 checks passed	No alarms	ap-south-1a		
<input type="checkbox"/>			Running	t2.medium	2/2 checks passed	No alarms	ap-south-1a		
<input type="checkbox"/>			Running	t2.large	2/2 checks passed	No alarms	ap-south-1b		
<input type="checkbox"/>			Running	t2.medium	2/2 checks passed	No alarms	ap-south-1b		
<input type="checkbox"/>			Running	t2.large	2/2 checks passed	No alarms	ap-south-1b		
<input type="checkbox"/>			Running	t2.medium	2/2 checks passed	No alarms	ap-south-1b		
<input type="checkbox"/>			Running	t2.medium	2/2 checks passed	No alarms	ap-south-1b		
<input type="checkbox"/>			Running	t2.medium	2/2 checks passed	No alarms	ap-south-1b		
<input type="checkbox"/>			Running	t2.medium	2/2 checks passed	No alarms	ap-south-1b		
<input type="checkbox"/>			Running	t2.medium	2/2 checks passed	No alarms	ap-south-1b		
<input type="checkbox"/>			Running	t2.medium	2/2 checks passed	No alarms	ap-south-1b		
<input type="checkbox"/>			Running	t2.medium	2/2 checks passed	No alarms	ap-south-1b		
<input type="checkbox"/>			Running	t2.medium	2/2 checks passed	No alarms	ap-south-1b		

Select an instance above

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.

The screenshot shows the AWS Management Console interface for the 'Choose AMI' step. The top navigation bar includes the AWS logo, 'Services' dropdown, a search bar, and user information. Below the navigation bar, a progress bar shows seven steps: 1. Choose AMI (active), 2. Choose Instance Type, 3. Configure Instance, 4. Add Storage, 5. Add Tags, 6. Configure Security Group, and 7. Review. The main content area is titled 'Step 1: Choose an Amazon Machine Image (AMI)' and includes a search bar and a 'Cancel and Exit' link. A 'Quick Start' sidebar on the left lists 'My AMIs', 'AWS Marketplace', and 'Community AMIs'. The main list displays several AMIs, with 'Ubuntu Server 20.04 LTS (HVM), SSD Volume Type' highlighted by a red rectangular box. Other visible AMIs include Amazon Linux 2, Red Hat Enterprise Linux 8, SUSE Linux Enterprise Server 15 SP2, and Microsoft Windows Server 2019 Base. Each AMI entry shows its name, ID, architecture, and a 'Select' button. The bottom of the console features a footer with 'Feedback', 'English (US)', and copyright information.

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.

- 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.

Services

Search for services, features, marketplace products, and docs

[Alt+S]

AppEmporio

Mumbai

Support

1. Choose AMI

2. Choose Instance Type

3. Configure Instance

4. Add Storage

5. Add Tags

6. Configure Security Group

7. Review

Step 6: 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. For example, if you want to set up a web server and allow Internet traffic to reach your instance, add rules that allow unrestricted access to the HTTP and HTTPS ports. You can create a new security group or select from an existing one below. [Learn more](#) about Amazon EC2 security groups.

Assign a security group: ☒ Create a new security group ☐ Select an existing security group

Security group name:

Description:

Type	Protocol	Port Range	Source	Description
SSH	TCP	22	Custom 0.0.0.0/0	e.g. SSH for Admin Desktop
HTTP	TCP	80	Custom 0.0.0.0/0, ::/0	e.g. SSH for Admin Desktop
HTTPS	TCP	443	Custom 0.0.0.0/0, ::/0	e.g. SSH for Admin Desktop

Add Rule

Warning

Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.

Cancel

Previous

Review and Launch

Feedback

English (US)

© 2008 - 2021, Amazon Internet Services Private Ltd. or its affiliates. All rights reserved.

Privacy Policy

Terms of Use

Cookie preferences

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.

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

AMI Details

Instance Type

Security Groups

Instance Details

Storage

Tags

Ubuntu Server 18.04 LTS (HVM), SSD Volume Type - ami-0b1dee...

Free tier eligible

Root Device Type: ebs

Virtualization type: hvm

ECUs

vCPUs

Memory (GiB)

Instance Type

t2.medium

-

2

4

Security group name

launch-wizard-1

Description

launch-wizard-1 created 2021-07-09T16:01:20.070+05:30

Type

SSH

HTTP

HTTP

HTTPS

HTTPS

TCP

TCP

TCP

TCP

TCP

Protocol

TCP

TCP

TCP

TCP

TCP

Port Range

22

80

80

443

443

Cancel

Previous

Launch

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

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

AMI Details

Instance Type

Security Groups

Instance Details

Storage

Tags

Ubuntu Server 18.04 LTS (HVM), SSD Volume Type - ami-0b1dee...

Free tier eligible

Root Device Type: ebs

Virtualization type: hvm

ECUs

vCPUs

Memory (GiB)

Instance Type

t2.medium

-

2

4

Security group name

launch-wizard-1

Description

launch-wizard-1 created 2021-07-09T16:01:20.070+05:30

Type

SSH

HTTP

HTTP

HTTPS

HTTPS

TCP

TCP

TCP

TCP

TCP

Protocol

TCP

TCP

TCP

TCP

TCP

Port Range

22

80

80

443

443

Cancel

Previous

Launch

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

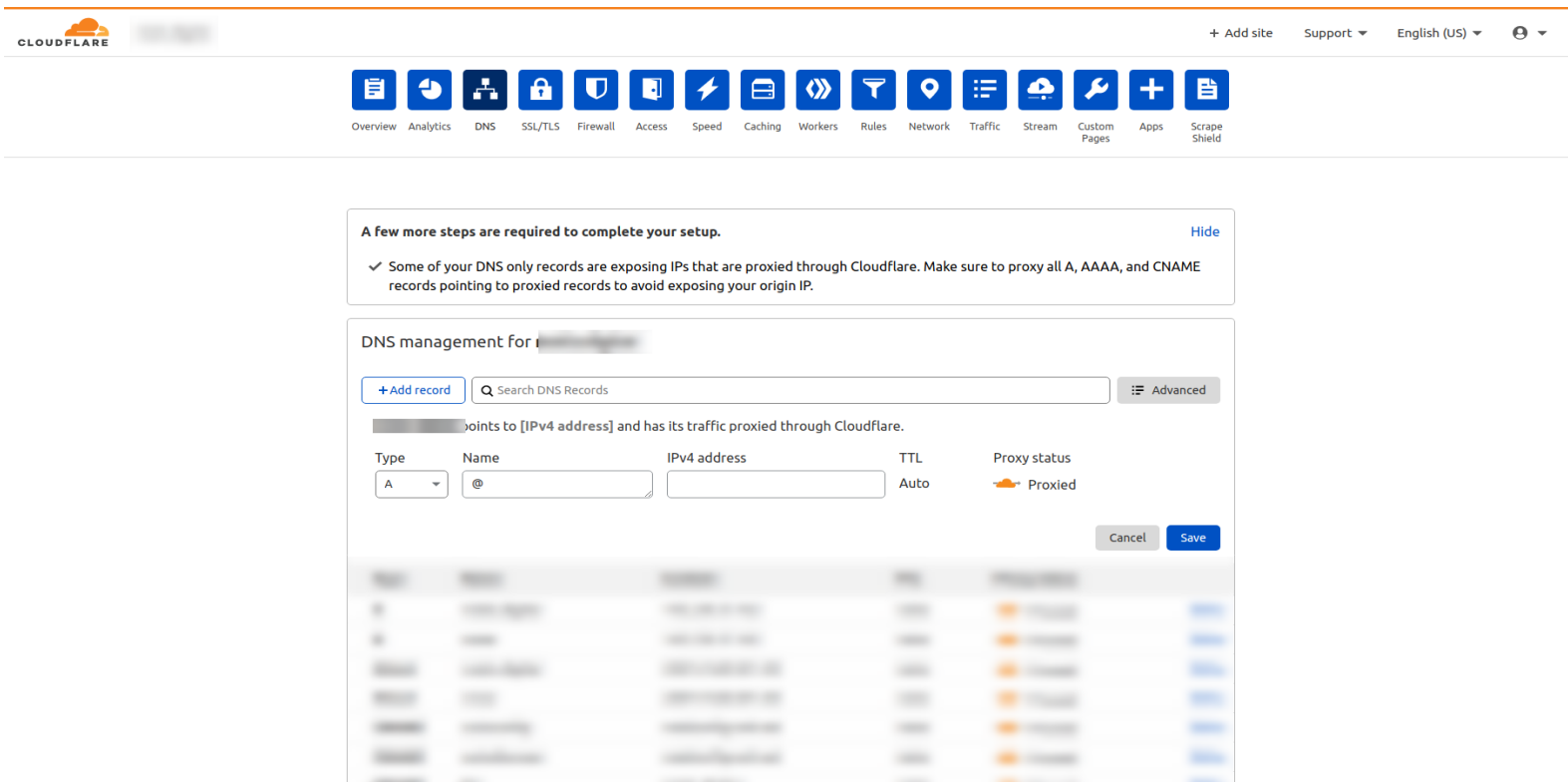
Now as our code is now running on the server, we can't open our application from search engines or web browsers using domain names.

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 records) 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.



A few more steps are required to complete your setup.

Hide

✓ Some of your DNS only records are exposing IPs that are proxied through Cloudflare. Make sure to proxy all A, AAAA, and CNAME records pointing to proxied records to avoid exposing your origin IP.

DNS management for

+ Add record

Search DNS Records

Advanced

is an alias of [target] and has its traffic proxied through Cloudflare.

Type

Name

Target

TTL

Proxy status

CNAME

admin

Auto

Proxied

CancelSave

Type	Name	Target	TTL	Proxy status	
A	www	192.0.2.1	1	Proxied	
A	www	192.0.2.1	1	Proxied	
A	www	192.0.2.1	1	Proxied	
A	www	192.0.2.1	1	Proxied	
A	www	192.0.2.1	1	Proxied	
A	www	192.0.2.1	1	Proxied	
A	www	192.0.2.1	1	Proxied	
A	www	192.0.2.1	1	Proxied	
A	www	192.0.2.1	1	Proxied	
A	www	192.0.2.1	1	Proxied	

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.

✔ 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](#)

APIHelp

SSL/TLS RecommenderBeta

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

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 browser 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 let's move to that file's 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 nginx 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 below 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.

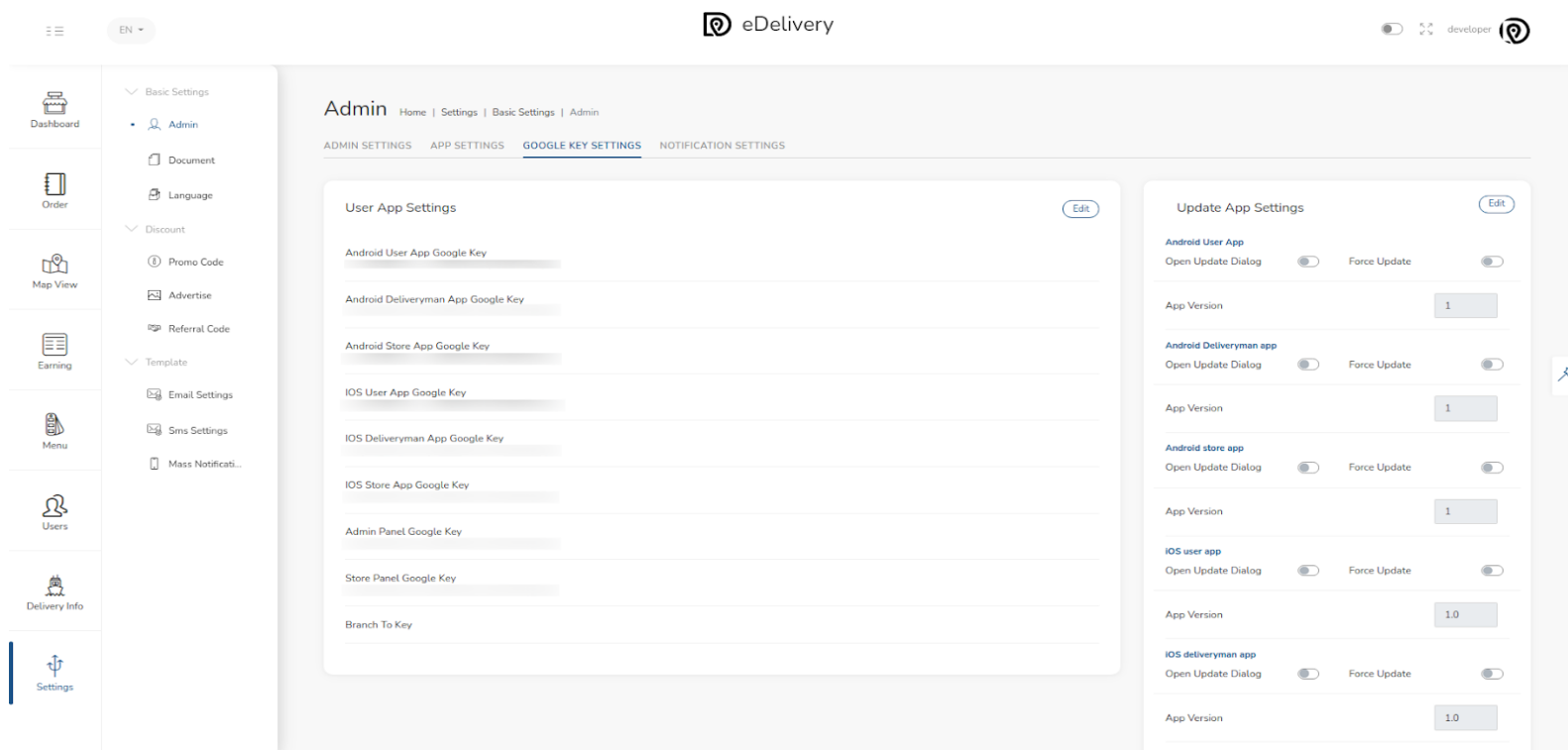
Now our applications are live so now we need to do some configurations to make our application fully functional

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

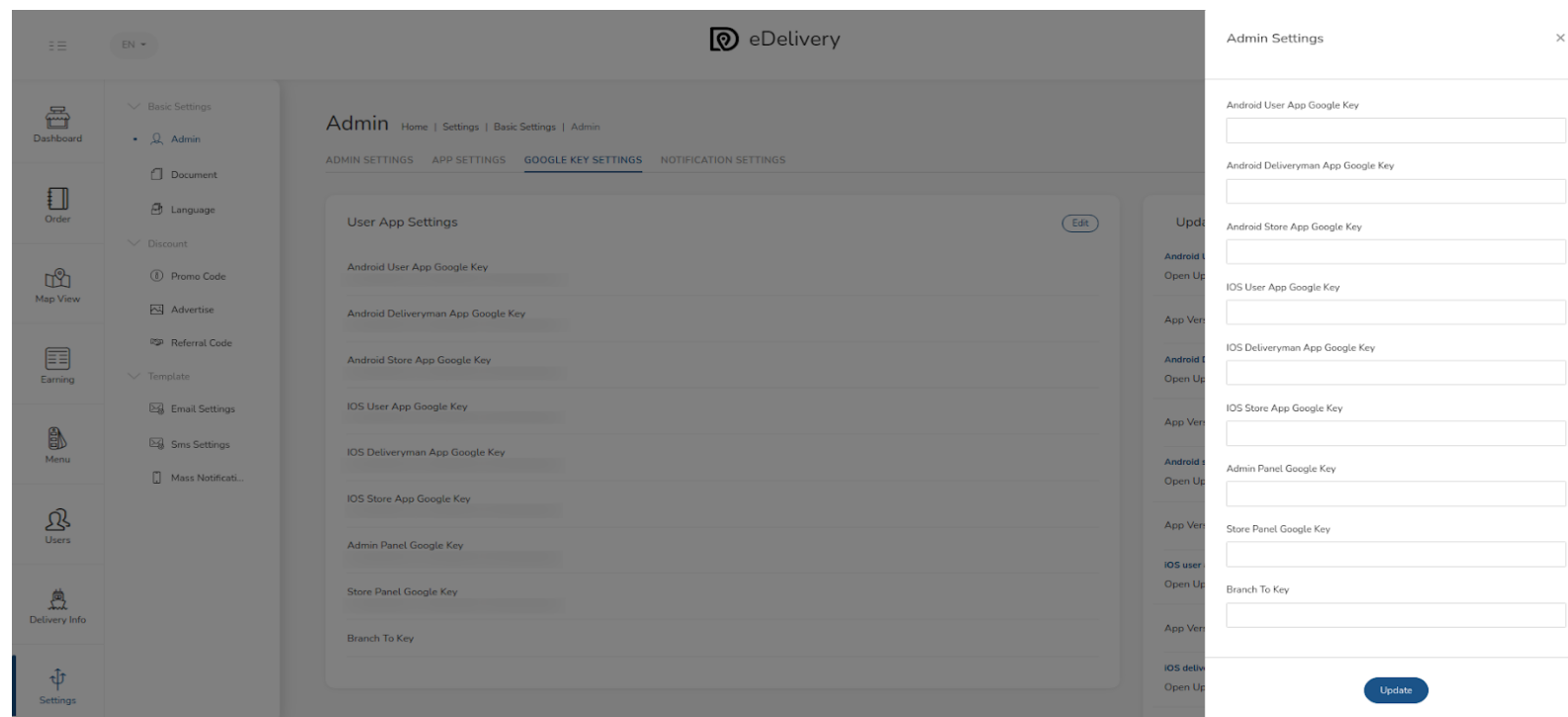
As shown in image head to app settings and press edit button to set you google api keys

Step 1 : Set up Google api keys

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

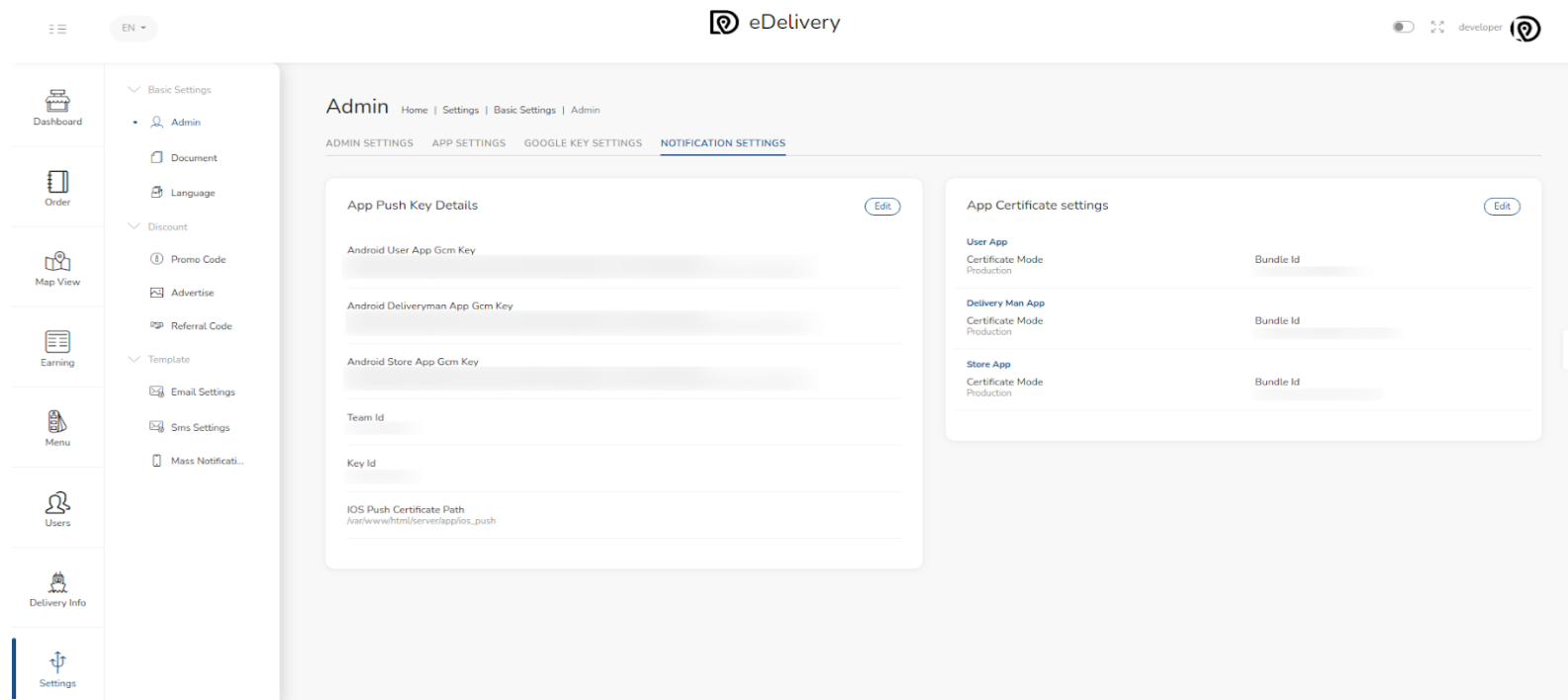


- Press edit button located at top-right corner in User App settings
- Place your google api keys in appropriate fields.



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

EN

eDelivery

Dashboard

Order

Map View

Earning

Menu

Users

Delivery Info

Settings

Basic Settings

Admin

Document

Language

Discount

Promo Code

Advertise

Referral Code

Template

Email Settings

Sms Settings

Mass Notificati...

Admin

Home | Settings | Basic Settings | Admin

ADMIN SETTINGSAPP SETTINGSGOOGLE KEY SETTINGSNOTIFICATION SETTINGS

App Push Key Details

Edit

Android User App Gcm Key

Android Deliveryman App Gcm Key

Android Store App Gcm Key

Team Id

Key Id

IOS Push Certificate Path
/var/www/html/server/app/ios_push

App Certificate settings

User App
Certificate Mode
Production

Delivery Man App
Certificate Mode
Production

Store App
Certificate Mode
Production

App Push Key Details

Android User App Gcm Key

Android Deliveryman App Gcm Key

Android Store App Gcm Key

Team Id

Key Id

IOS Push Certificate Path
/var/www/html/server/app/ios_push

Push P8 File
Choose a fileBrowse

Update