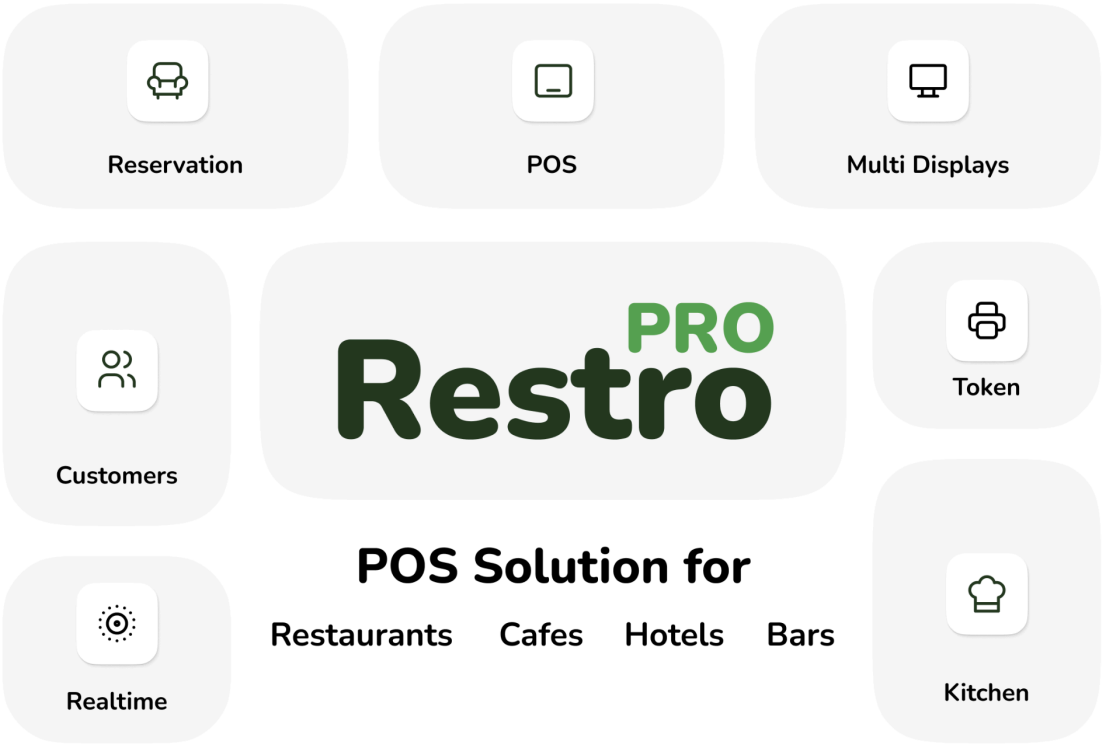
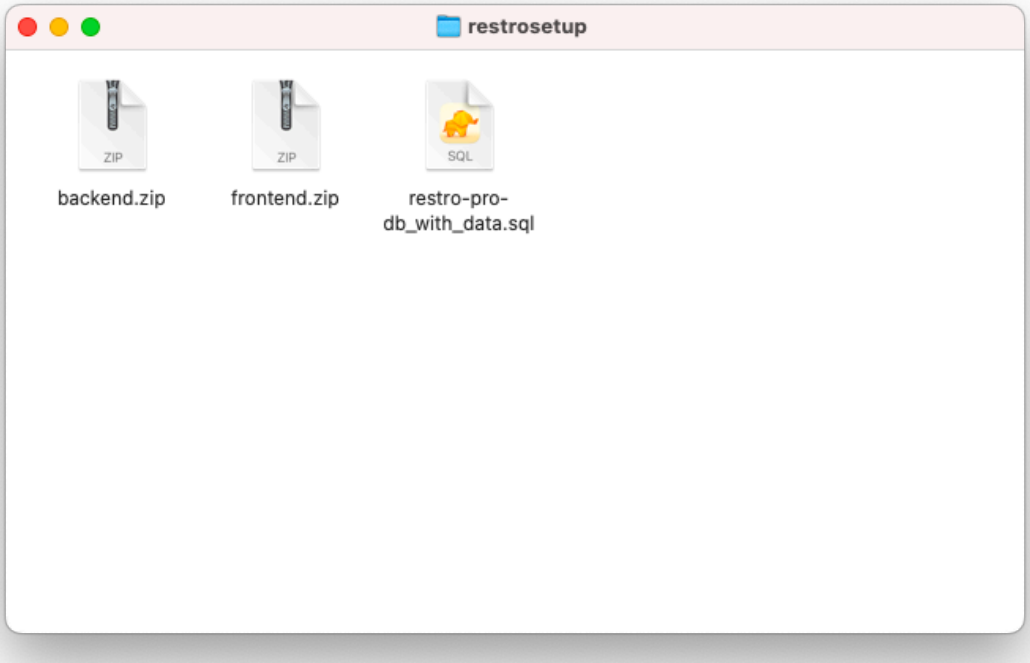


Documentation

7. Installation & Customisation Service

1. Prerequisites

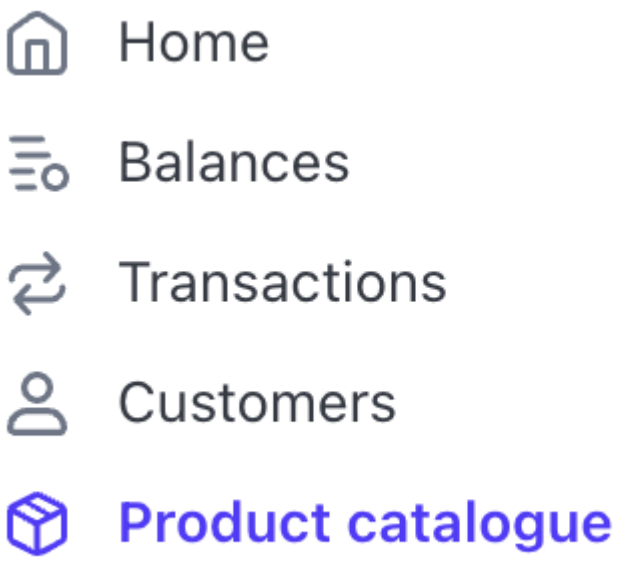
RestroPRO, the product is built using following technologies NodeJS, MySQL, ReactJS.
So to Setup the project, you'll need technical knowledge and, you'll need to purchase a Ubuntu Server and domain. Before doing that, unzip the provided files, and keep the source code zip file ready.



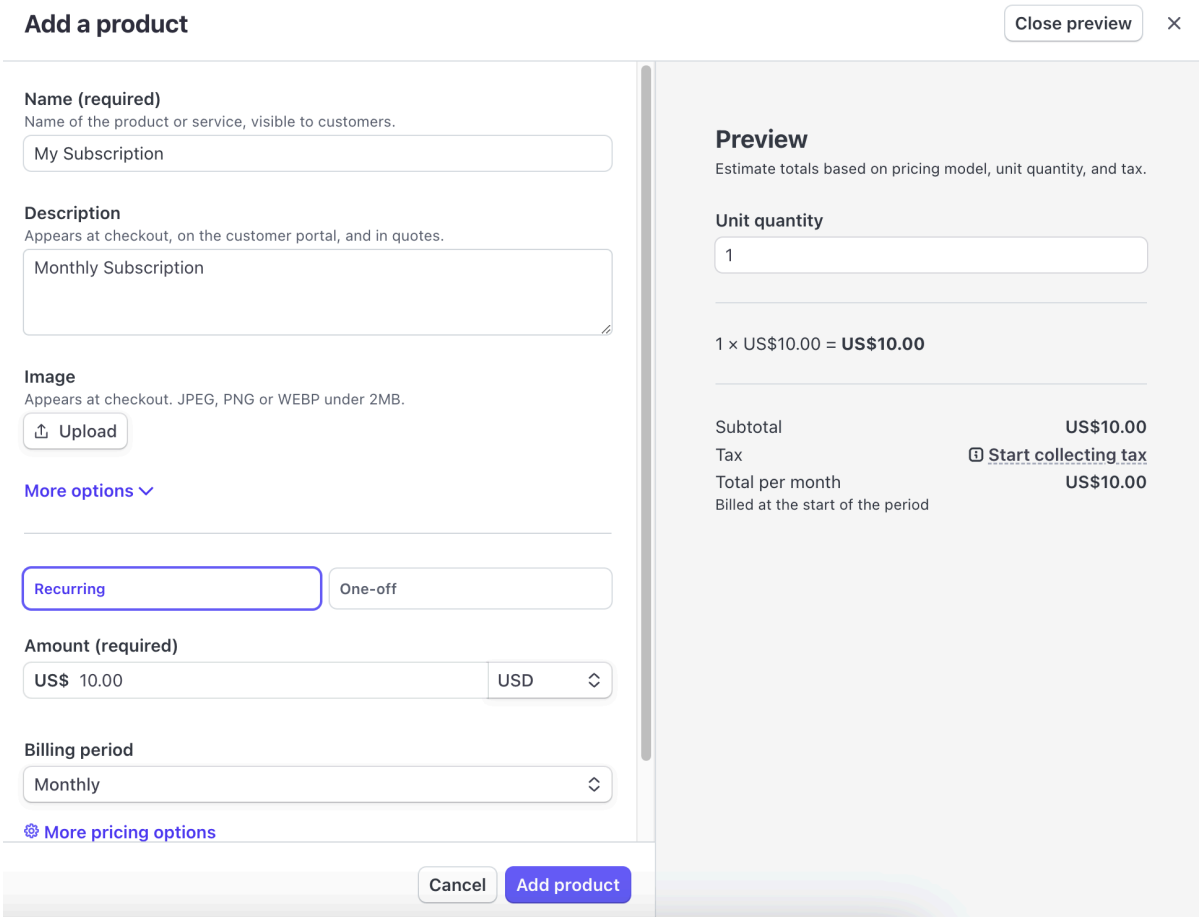
2. Stripe Subscription Setup

Goto stripe.com and set up your account according to your business registration type. Once you activate your account, follow the given procedure.

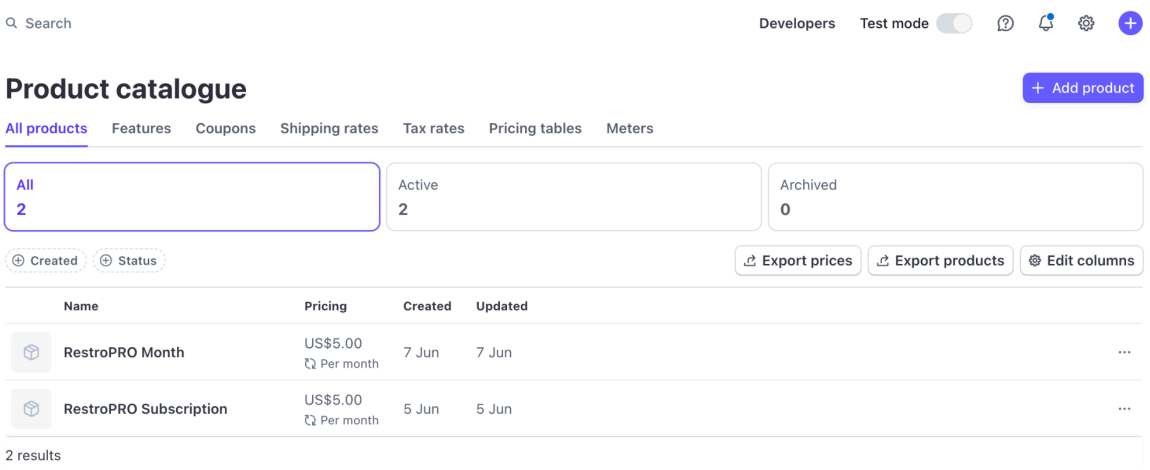
Goto Stripe Dashboard and Select "Product catalogue", then Add new product.



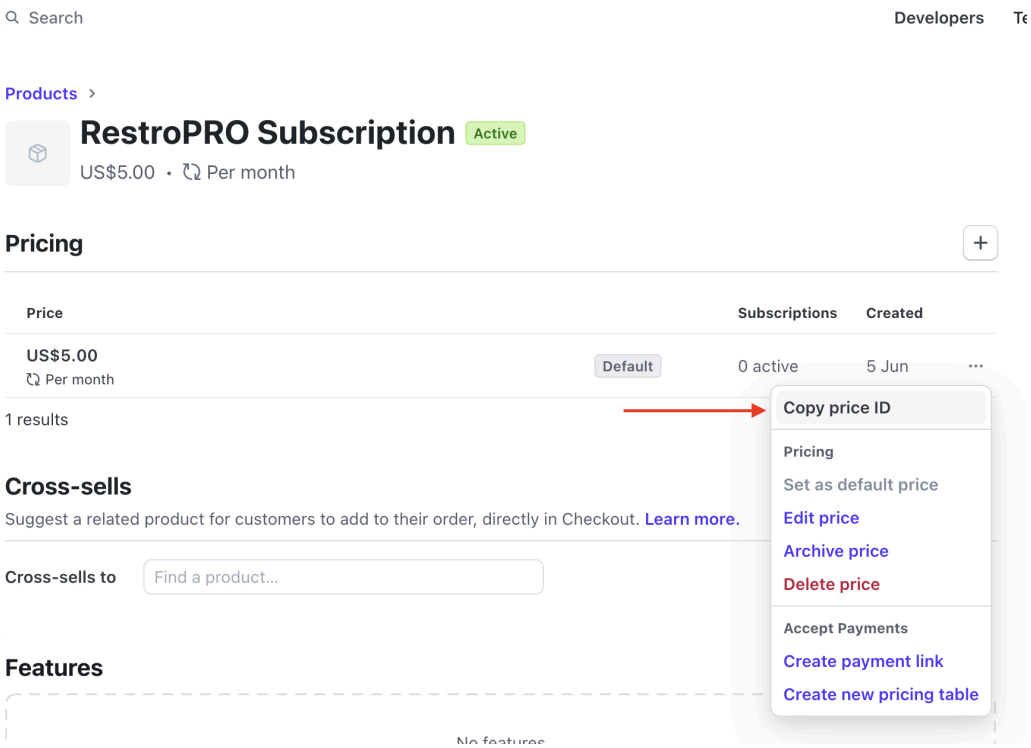
Feel the details as per your business requirements.



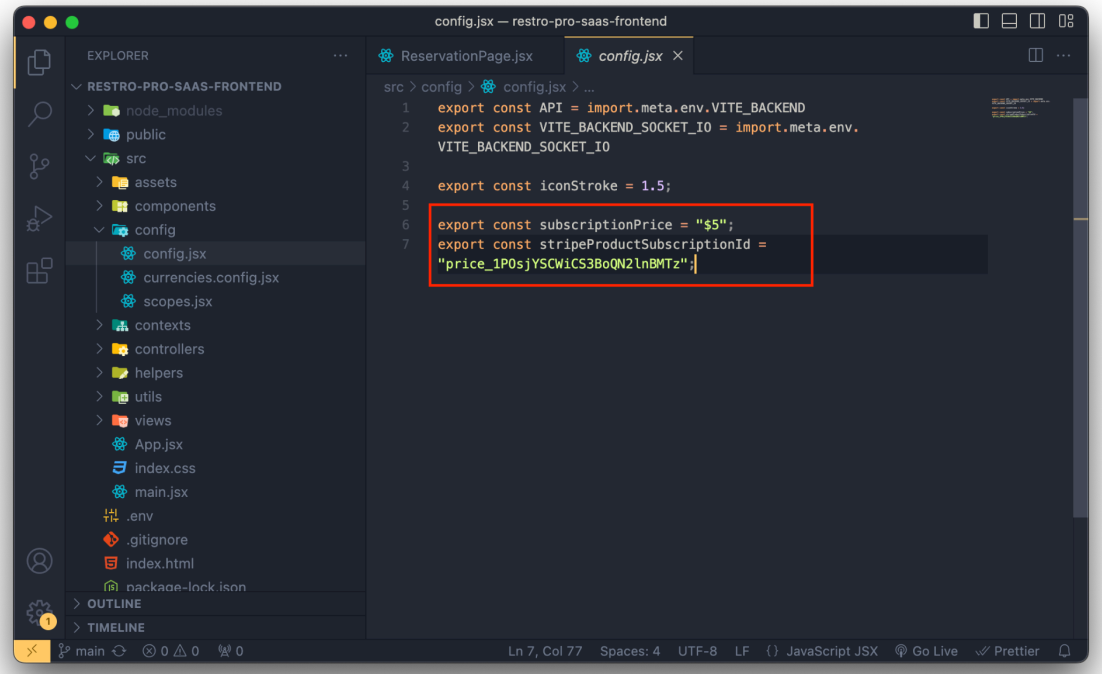
Once you create the Subscription Product, you will see it like this.



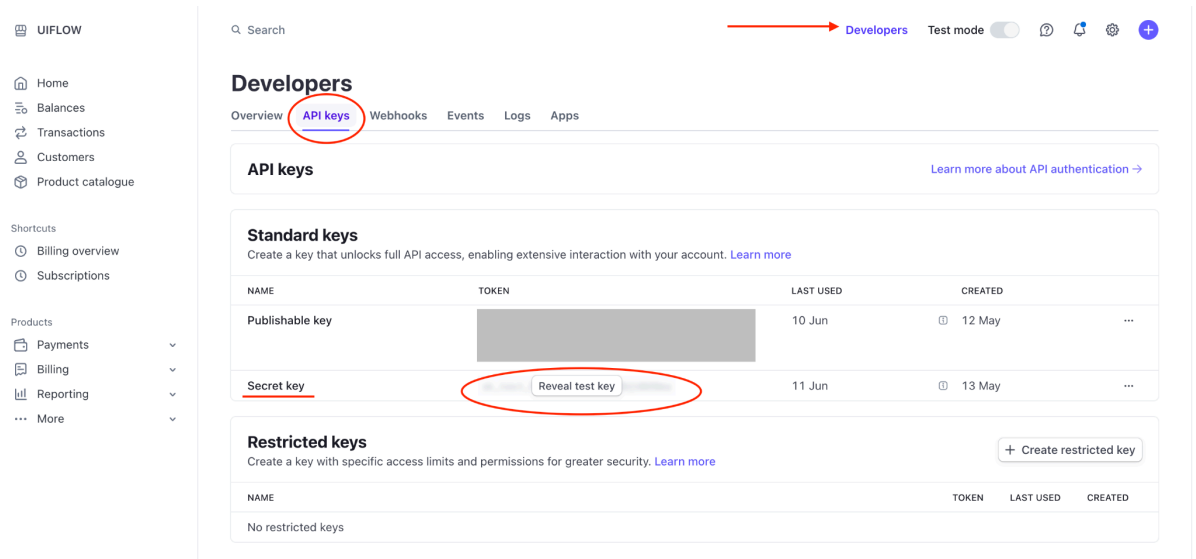
Click on your created Subscription Product. Look for Pricing, and click on options three dots, look for option to copy price id, copy it and save temporary in notes.



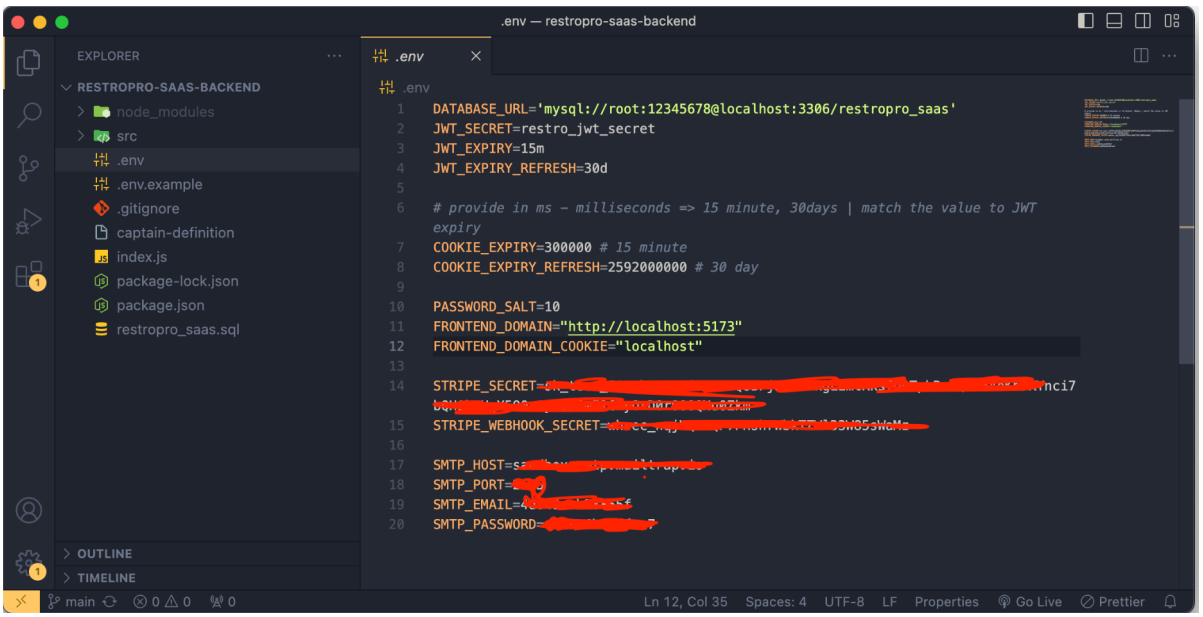
Now open the front-end code. And navigate to src/config/config.jsx, open the file and look at the highlighted part, we need to edit the default values there. First change the subscriptionPrice to the price that you decided for the subscription. Then replace the price id.



Now go back to Stripe Dashboard and open the developers page, you will get API keys from there. Look for a secret key, and copy it.

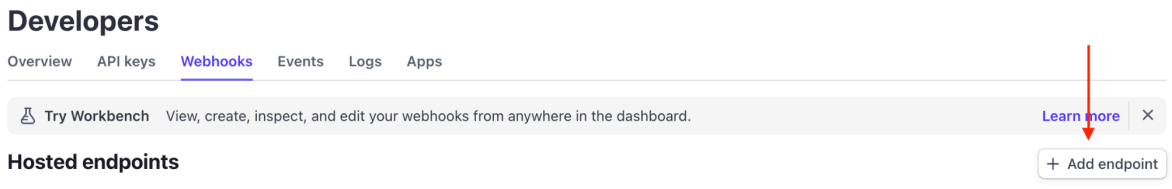


Now open the backend code.



Paste the secret code in STRIPE_SECRET

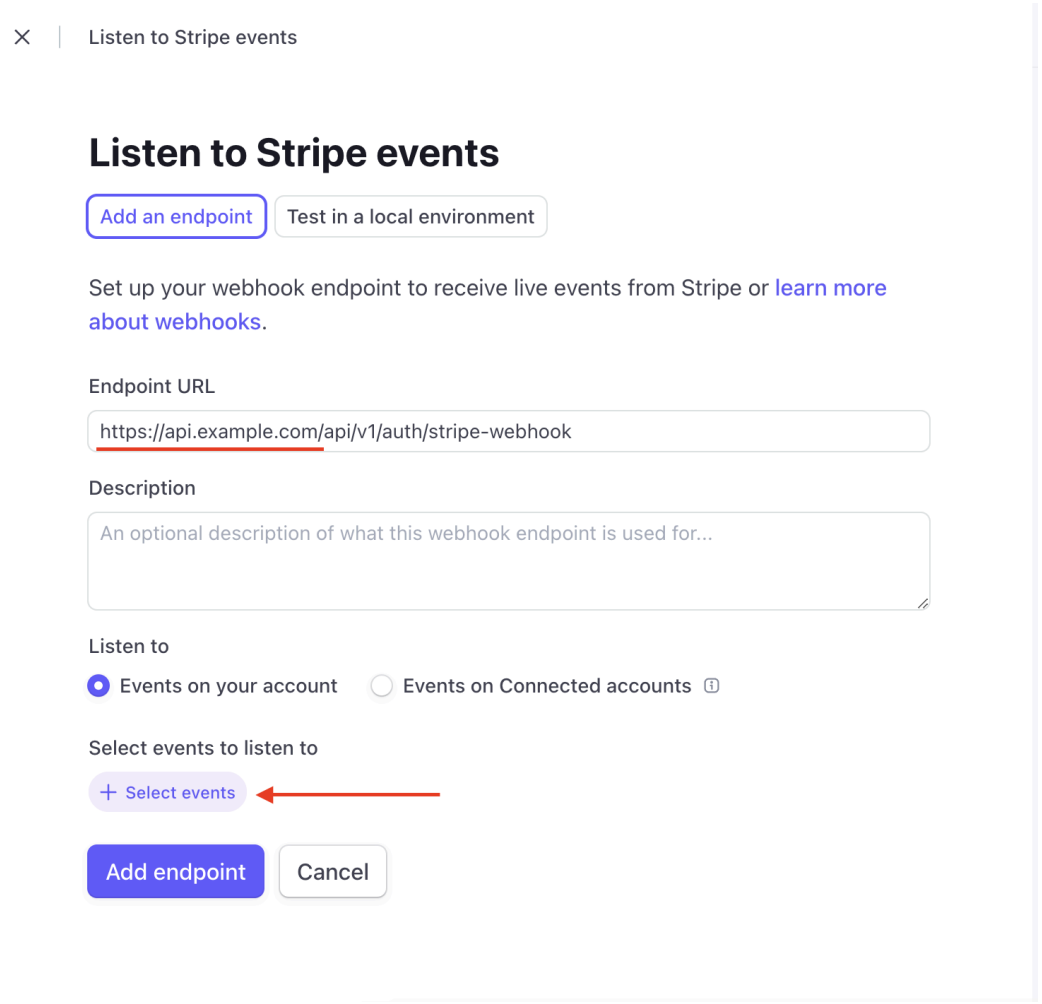
Now we will set up the webhook. For that open Stripe Dashboard and goto developers page again. Goto Webhooks Tab, then click on Add endpoint.



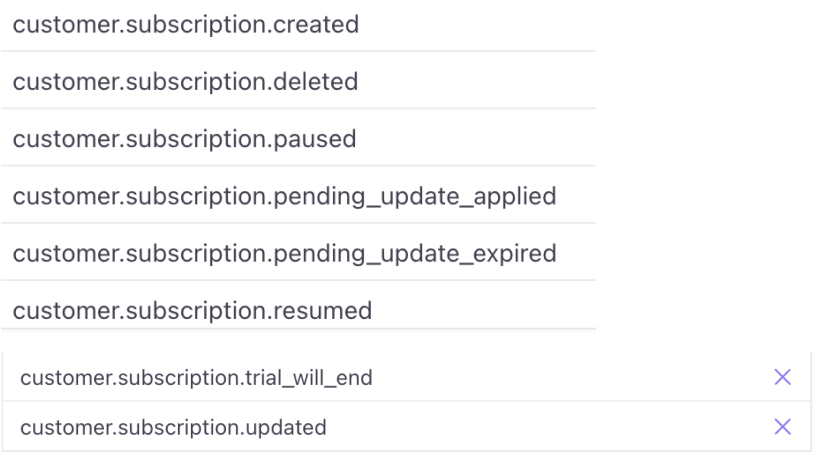
Then provide the webhook URL, this will be our backend route webhook handler, so we're yet to setup the backend, so you just need to provide the url. For example if i'm setting Backend on domain "api.example.com" then the webhook URL will look like this.

Webhook URL: **https://api.example.com/api/v1/auth/stripe-webhook**

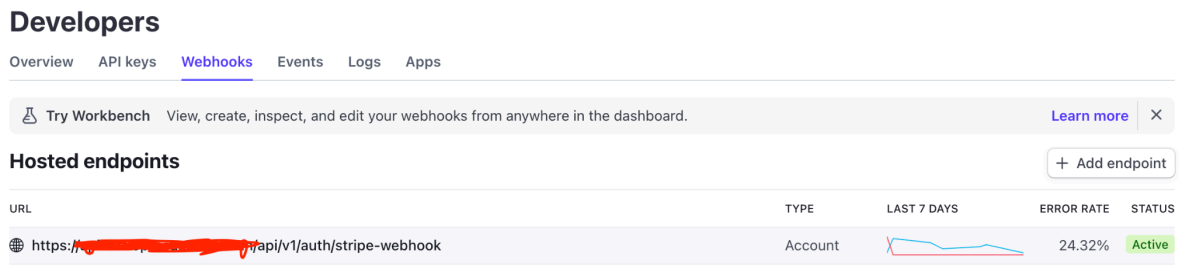
Replace the domain, to your backend domain. After placing endpoint URL, click on the 'select events to listen to'



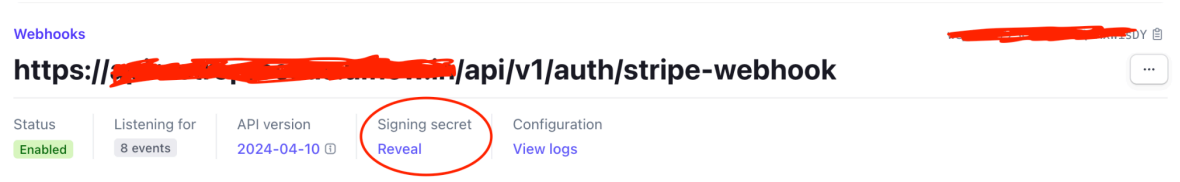
Look for the following events and select them.



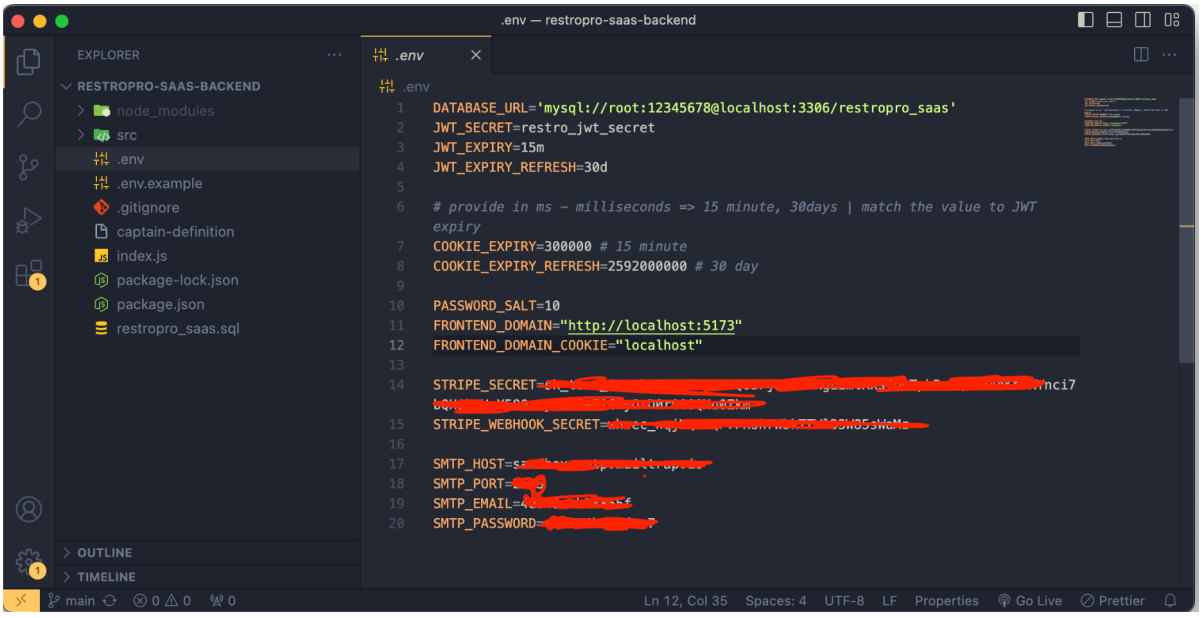
Once you add it open the created webhook endpoint in Stripe Dashboard.



Click on the 'Reveal' to copy the Webhook Signing Secret.



Now go back to the backend .env file. And paste the copied secret in STRIPE_WEBHOOK_SECRET

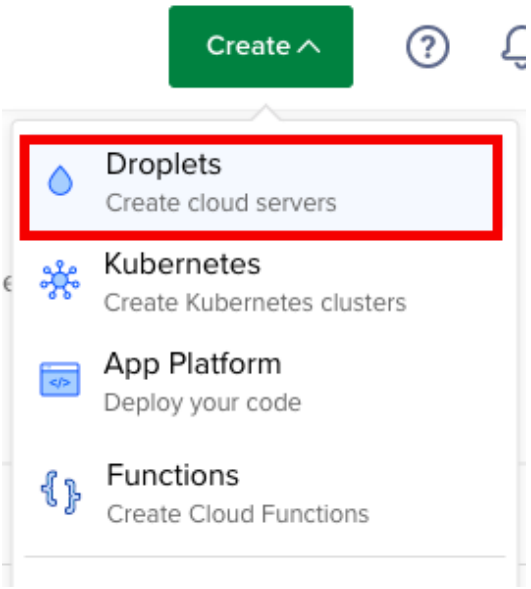


Extra setup for sending forgot password email:
Paste your SMTP configuration in the .env file

3. Server Setup & Database config

We'll be using DigitalOcean to setup the server, however you can use any Provider which provides Ubuntu server.

Login to digitalocean or signup. After login you will need to create a droplet (ubuntu server). Click on the Create button and then select "Droplets".



Select the Ubuntu server and complete the process. Make sure you select the SSH method for signing into the droplet, it will ask you to put your device's SSH key, so you can easily and securely SSH into your cloud server.

After the completion of the creation process, open the Terminal on your computer. And type the following command.

```
› ssh root@your_server_ip
```

Then You'll be prompted for your SSH key passphrase, just provide it and you'll be connected to your server.

Now update the server by running the following command.

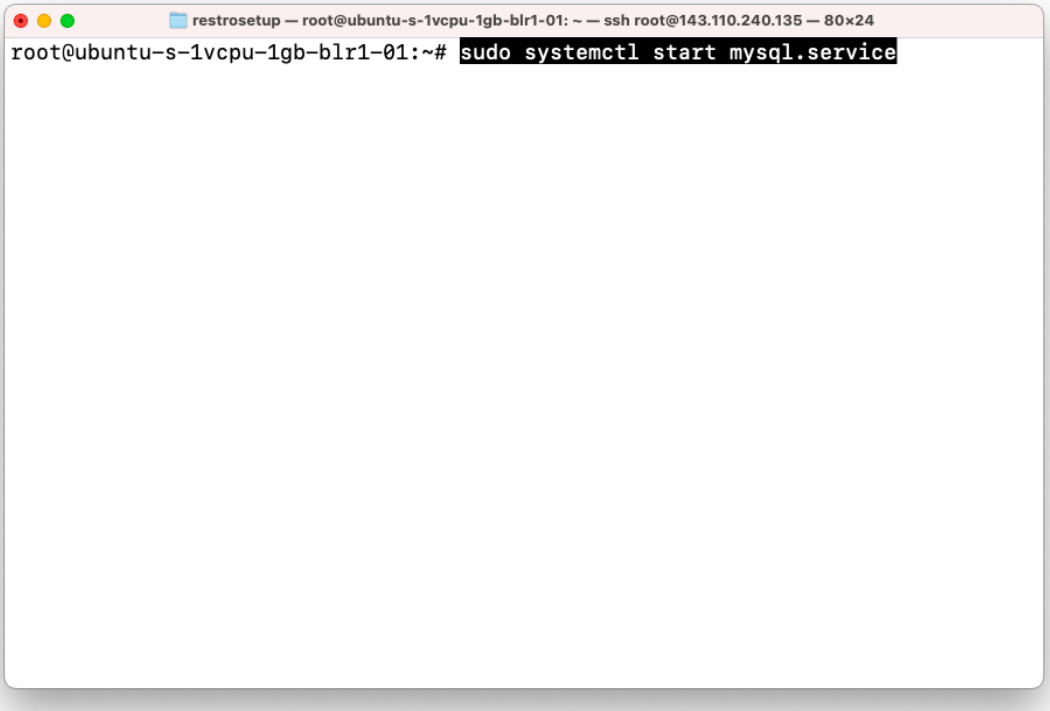
```
› sudo apt update
```

Once the update process is completed we'll begin installing the MySQL server.

```
› sudo apt install mysql-server
```

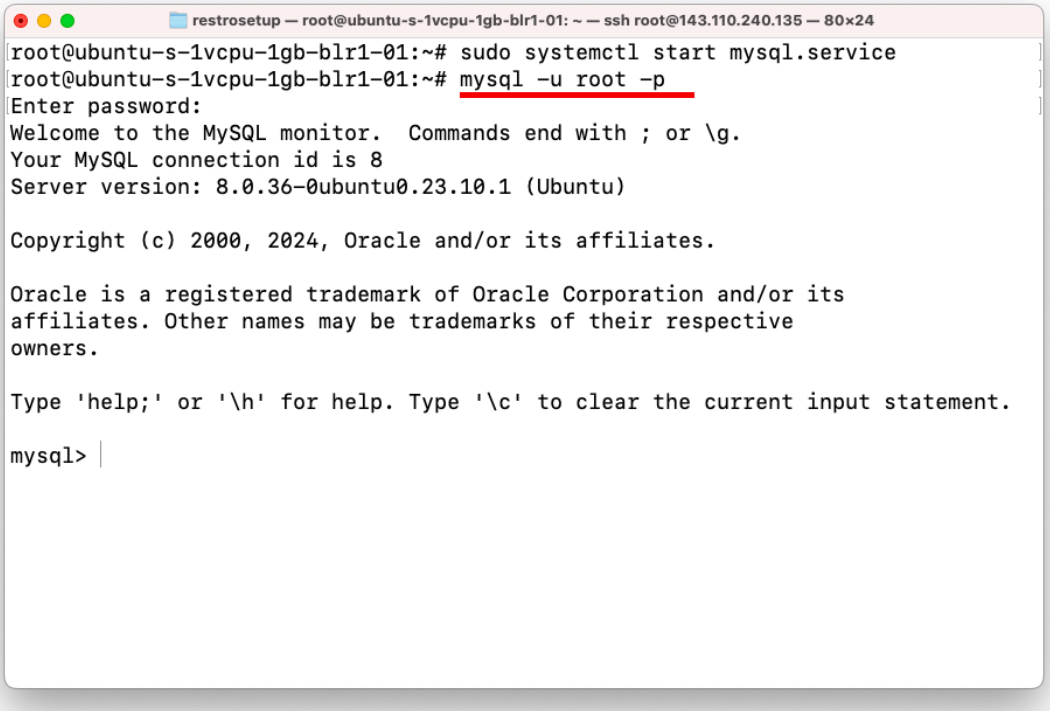
Once the MySQL server is installed run this command to start it.

```
› sudo systemctl start mysql.service
```



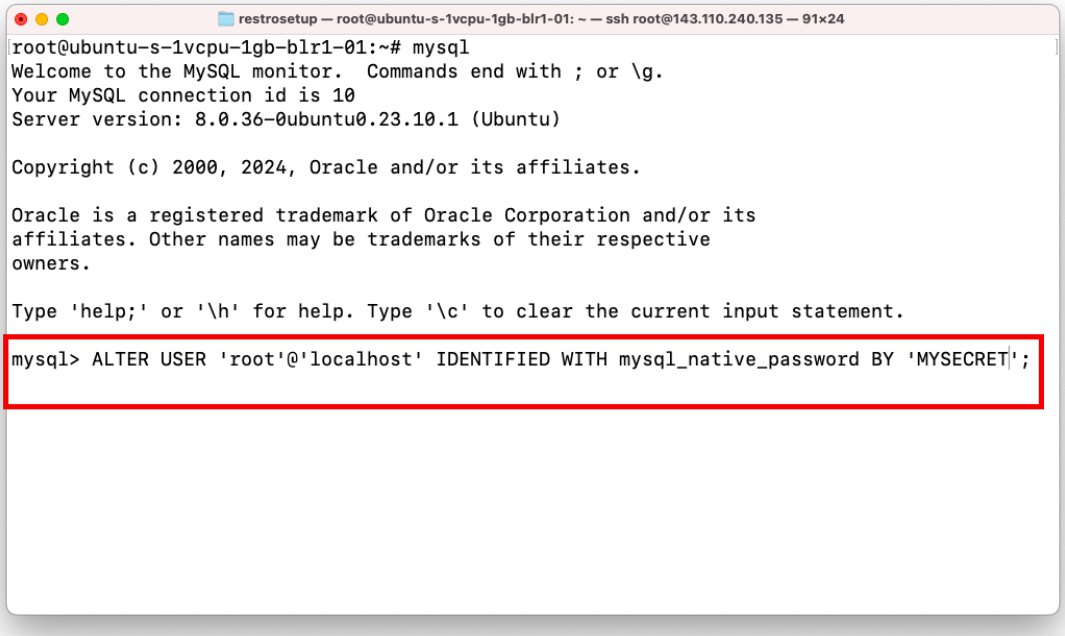
You can now login to mysql server by writing the following command.
Make sure when password is prompted the first time, just hit enter, we'll reset the root password in next steps.

```
› mysql -u root -p
```



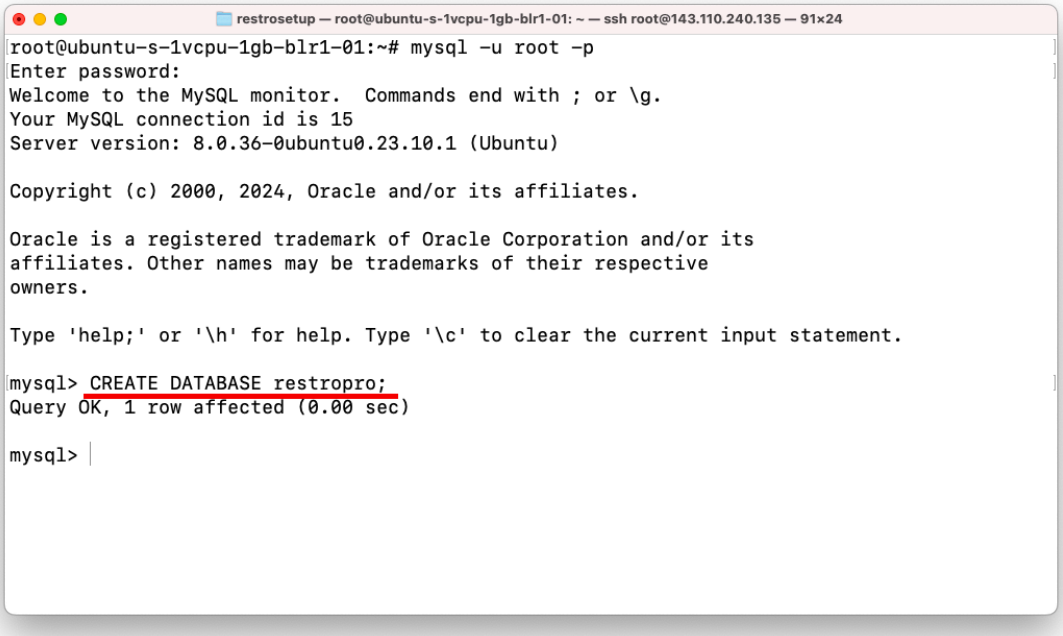
To reset the password paste following SQL command. And make sure to put your own password.

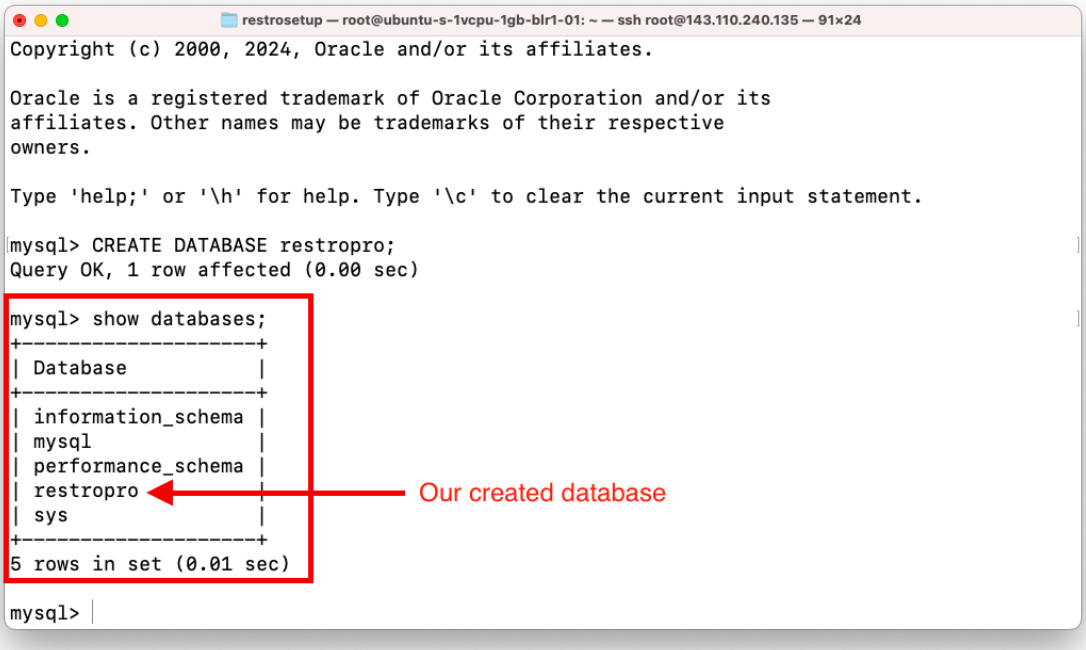
```
> ALTER USER 'root'@'localhost' IDENTIFIED WITH mysql_native_password BY 'MYSECRET';
```



Now we'll create a database. Use the following command to create it.

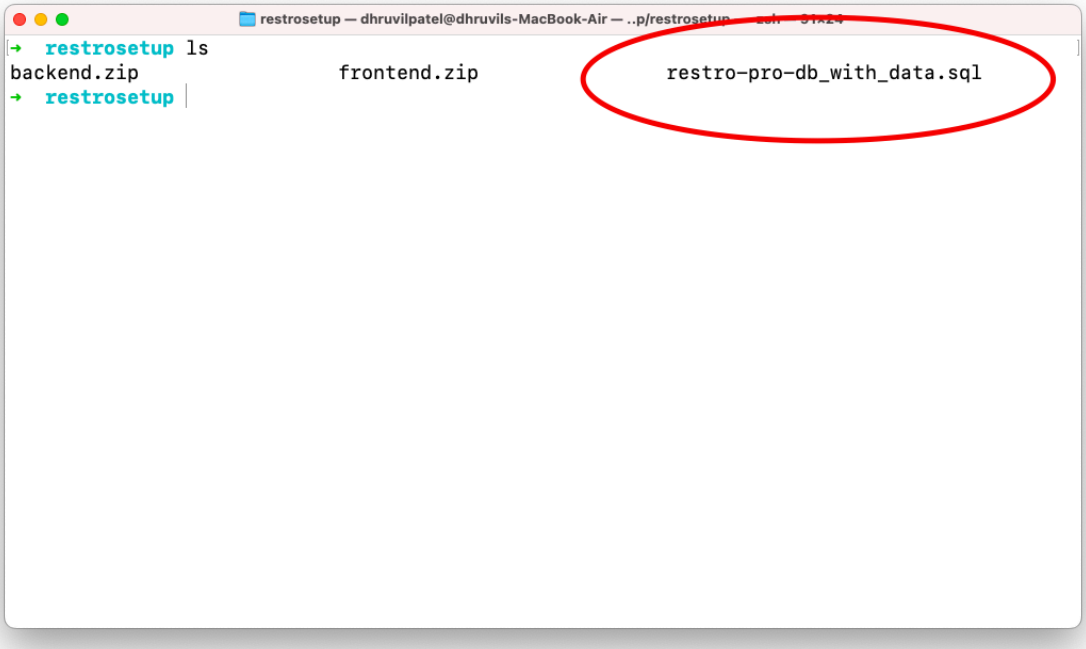
```
> CREATE DATABASE restropro;
```





Now close the connection to the server.

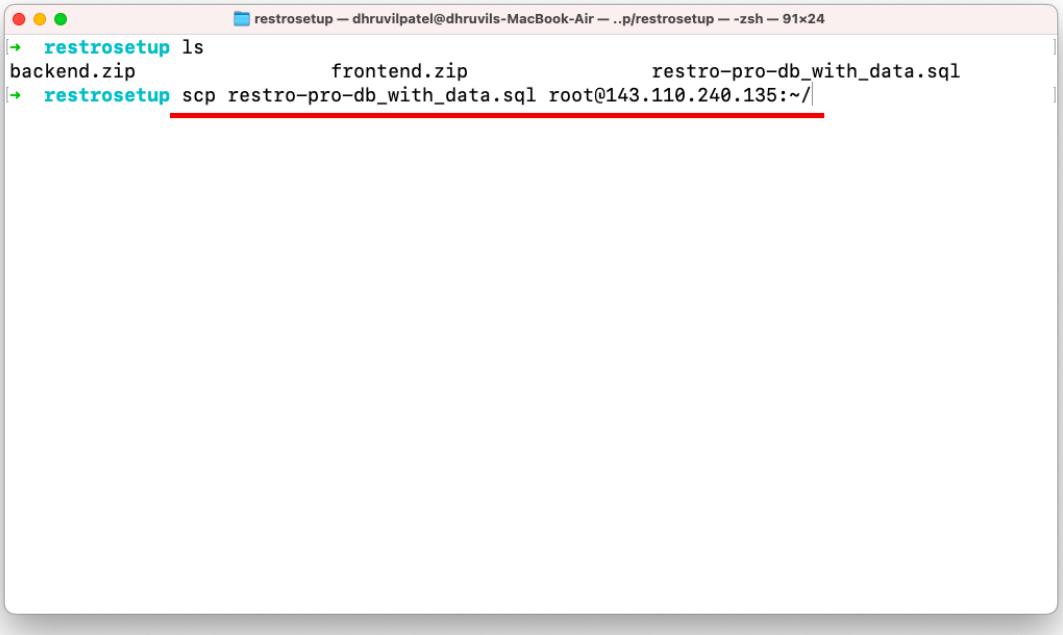
Now we'll import the database backup file, the database backup file includes structure and sample data pre configured.



Copy the highlighted file to the server from your computer. To do that run the following command.

```
› scp restro-pro-do_with_data.sql root@YOUR_SERVER_IP:~/
```

Replace the YOUR_SERVER_IP with actual IP ADDRESS



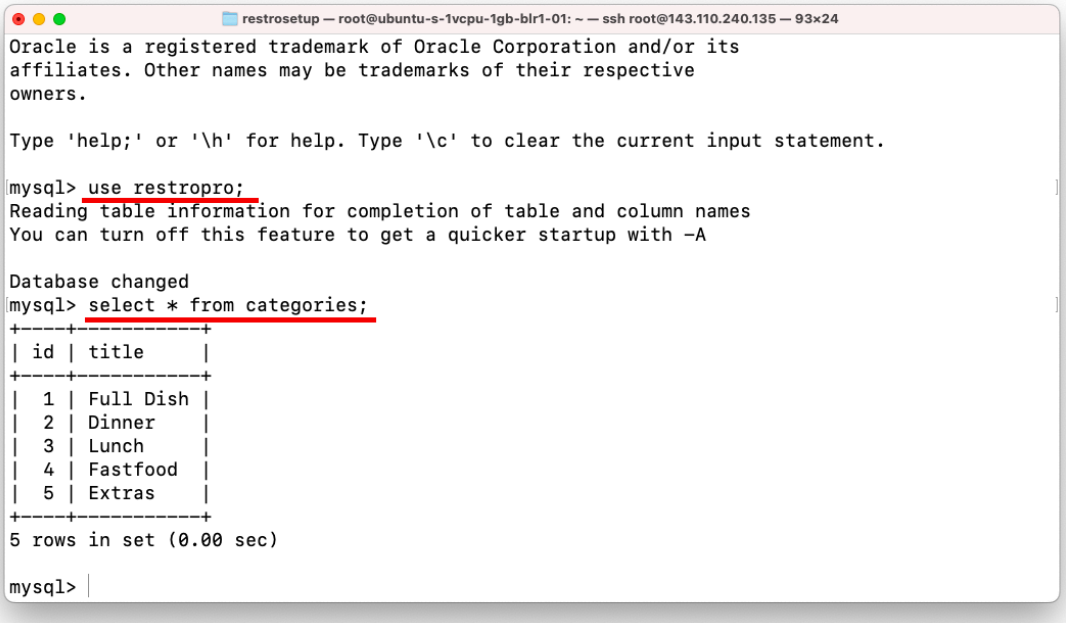
Now login to your server using the SSH command. And just hit the “ls” command to see if the file is copied or not.

On server, paste the following command to import the database backup

```
› mysql -u root -p restropro < restro-pro-db_with_data.sql
```

Once you hit enter it will ask for your MySQL server password, enter it, and your database import will be completed.

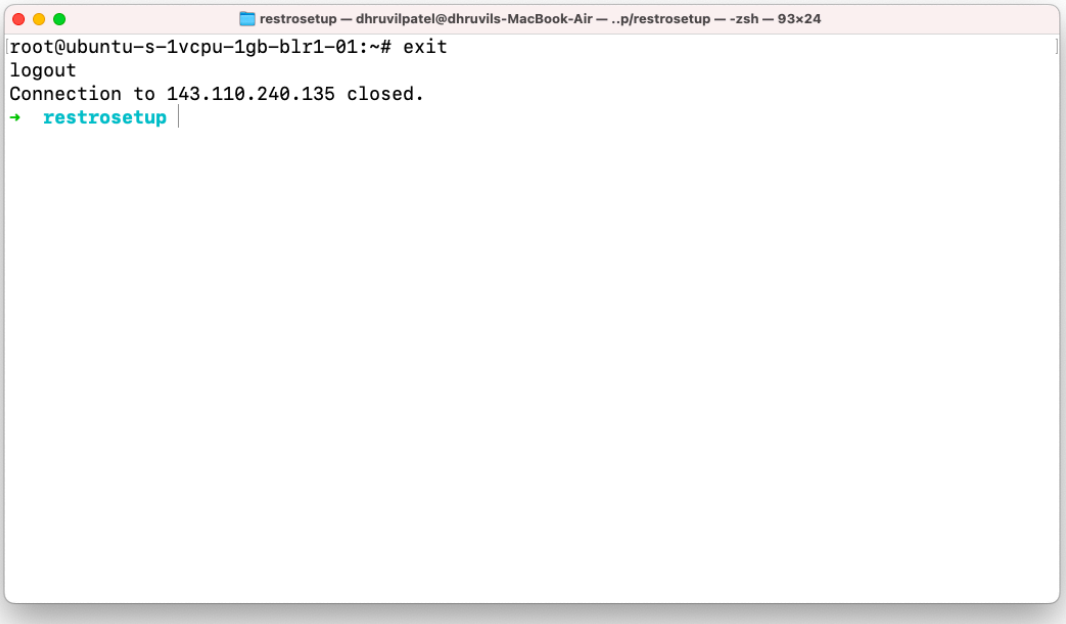
You can verify the data by login into MySQL server.



So now our Database Server setup is completed, and we'll proceed for Backend code deployment.

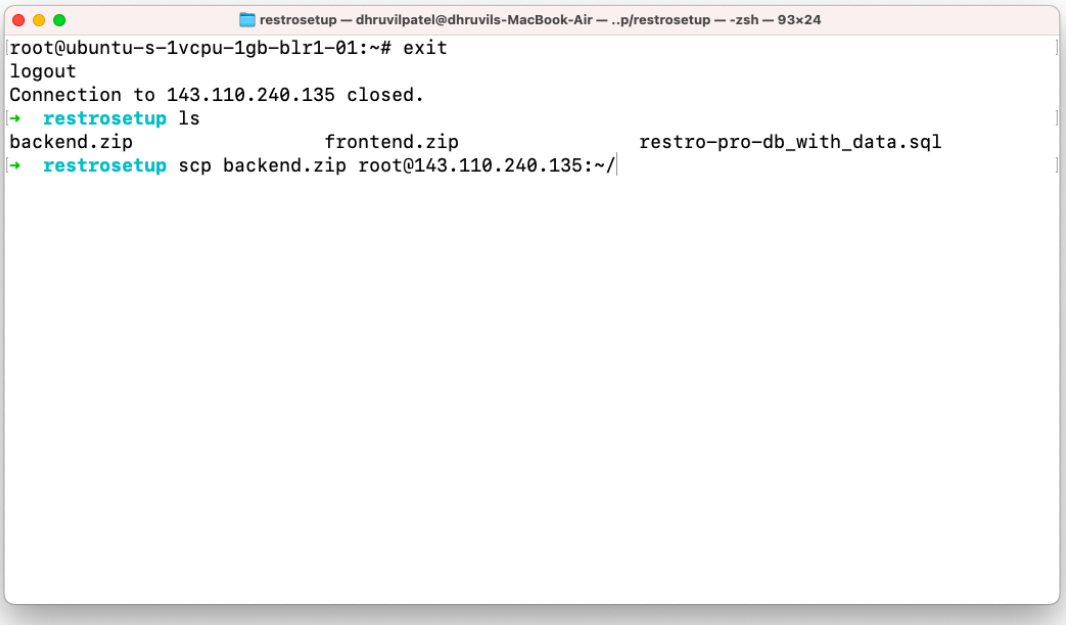
4. Backend code Deployment

First exit the server by sending 'exit' command in the terminal. As we have to copy the backend zip code from the computer to the Server.



Use the following command to copy backend zip file to server.

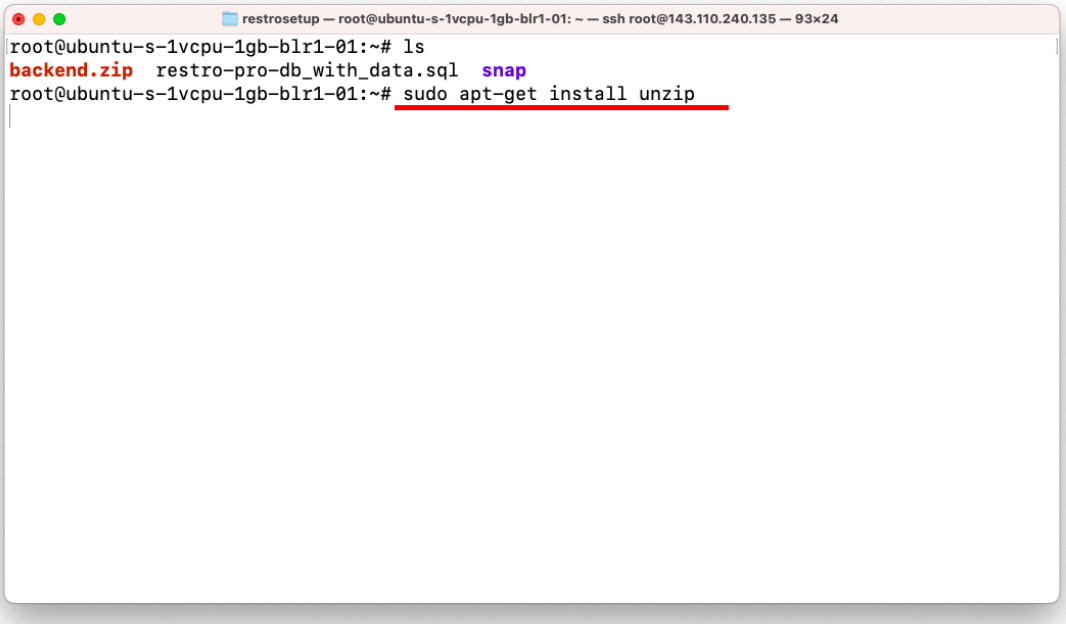
```
› scp backend.zip root@YOUR_SERVER_IP:~/
```



Once the file is successfully sent to the server, login to the server by using ssh command.

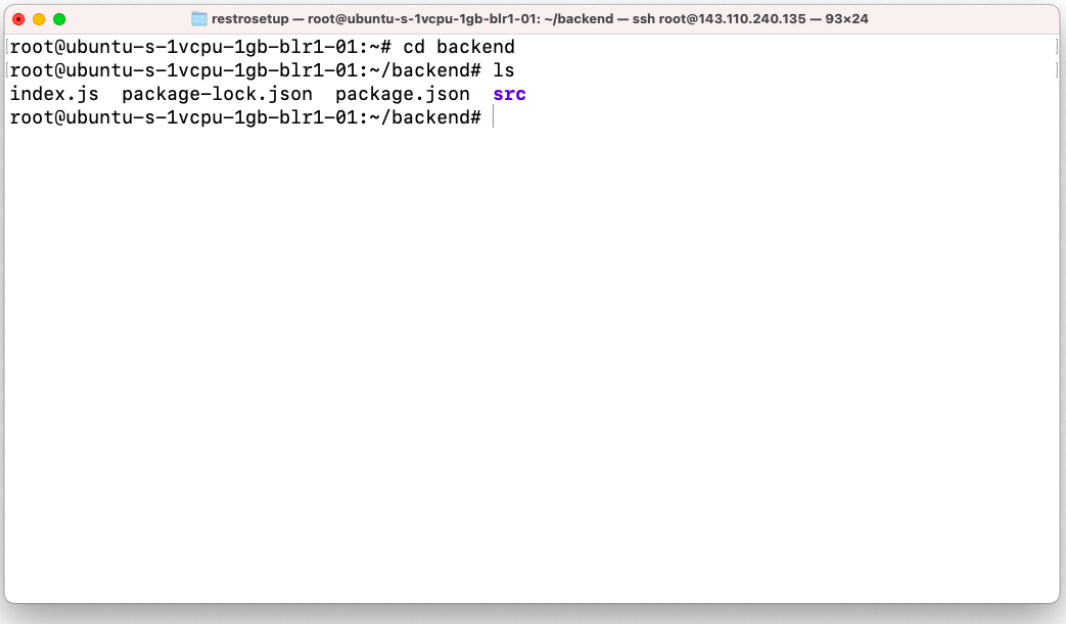
First thing we will do is unzipping the backend.zip, for that use the following commands.

```
› sudo apt-get install unzip
```



› unzip backend.zip


```
› cd backend
```



Now install nodejs using the following command.

```
› sudo apt install nodejs
```

You can check the installation by running the “node -v” command.

We will also need to install the “npm”. So to install that use the following command.

```
› sudo apt install npm
```

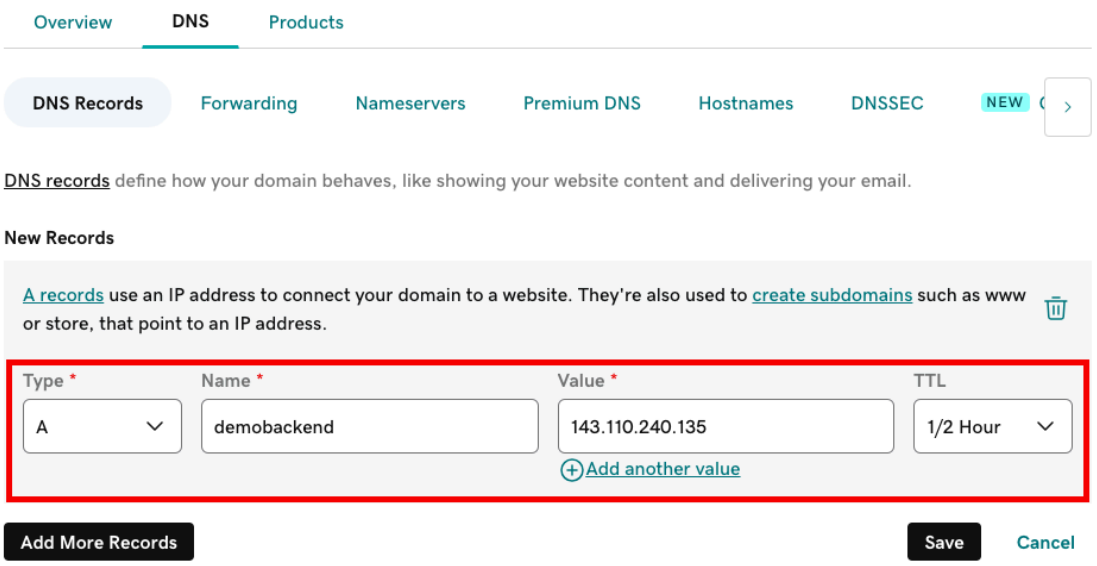
We've successfully installed NodeJS and NPM. it's the perfect time that we map the domain to the Server's IP.

For mapping the domain, open the DNS settings available in your domain provider like GoDaddy, Name.com, Hostgator, Bigrock, etc.

And add **'A'** Record pointing to **your server's IP**. in the Name value if you want to host the backend on a root URL like myapi.com then write "@".

Else you can use subdomain, for that in the name field write the subdomain, and your server will point to subdomain, in this case i'm linking the Backend server first so in the name i provided "demobackend", this will result in following URL "demobackend.mydomain.com".

Similarly you can also add Another "A" Record for pointing to Frontend. Use the Same IP in the Value Field of your Server. And change the name field to something like "myrestroapp". Anyways for frontend configuration we'll cover this again.



Now back to the server, we will install the pm2, for running nodejs apps in the production environment.

```
› sudo npm install pm2@latest -g
```

Now we will install the backend code's packages, run the following command.

```
› npm install
```

After installing the packages, open the .env file, we will edit a few details there.

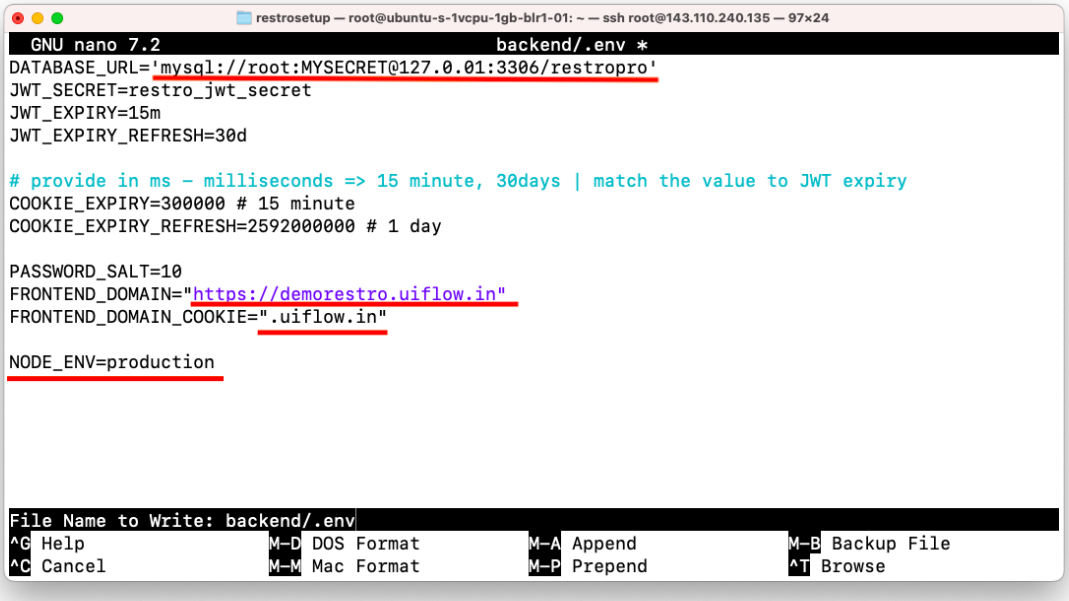
```
› nano .env
```

We will modify 4 Environment variables, keep rest as it is. Those four environment variables are highlighted in Screenshot.

```
DATABASE_URL="mysql://root:MYPASSWORD@127.0.0.1:3306/restropro"
FRONTEND_DOMAIN="https://app.example.com"
FRONTEND_DOMAIN_COOKIE=".example.com"
```

See here in "frontend domain" value there is the domain of your Frontend where you will host it. Ex.: app.johnsrestro.com
And in the front-end domain the value starts with "." (dot) and then the domain

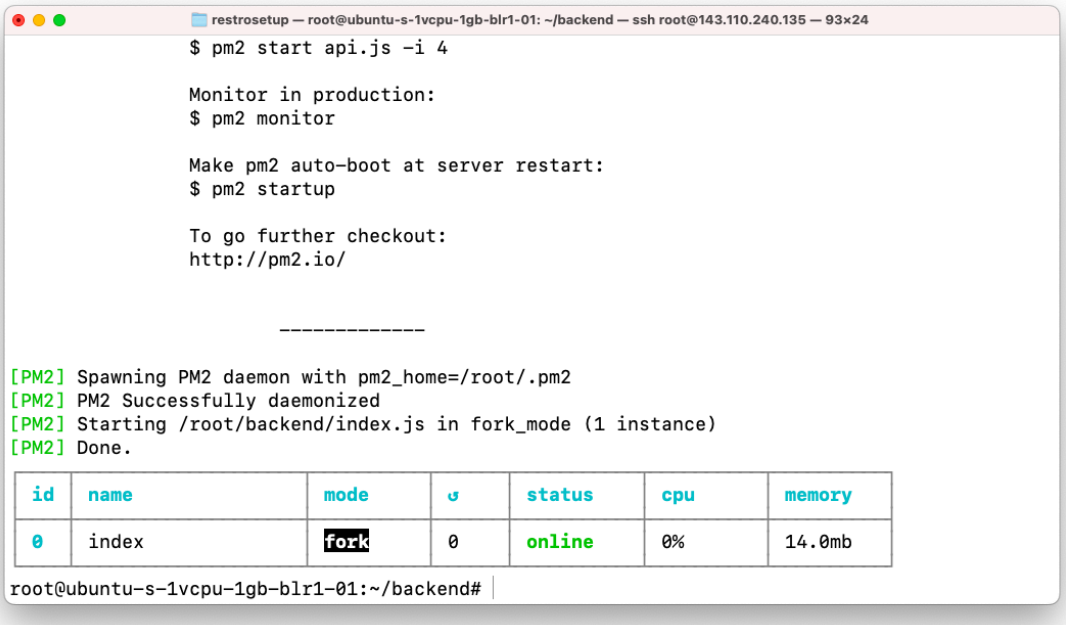
Make sure you configure these details carefully, failing to do so, the app will not work.



Once you edit the details, to exit the editor press Ctrl + X, then press Y, and save.

Now to start the backend server run the following command.

```
› pm2 start index.js
```

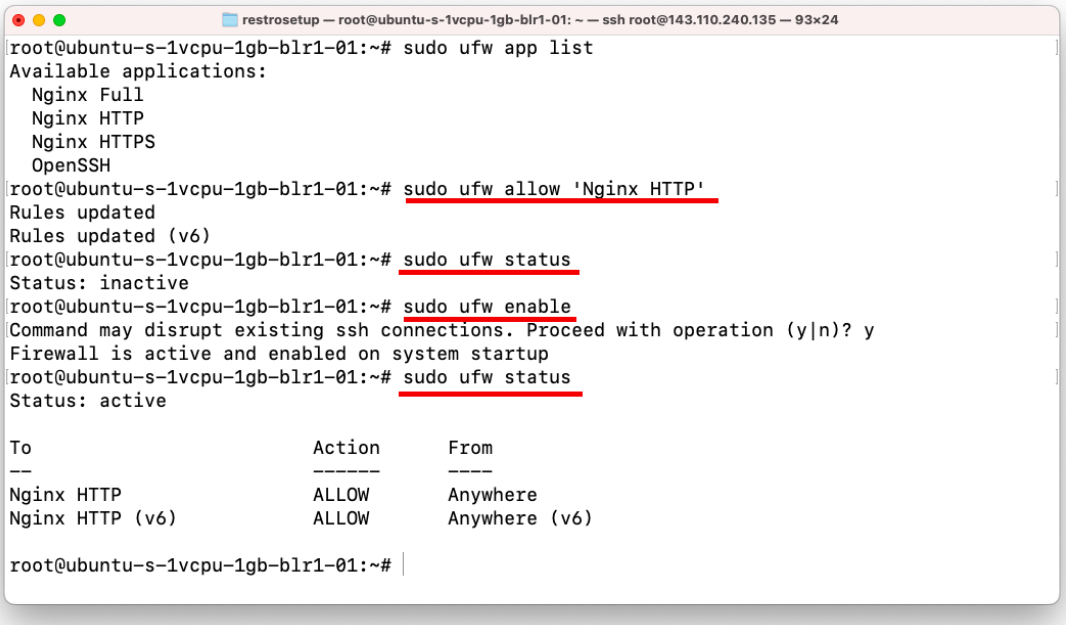


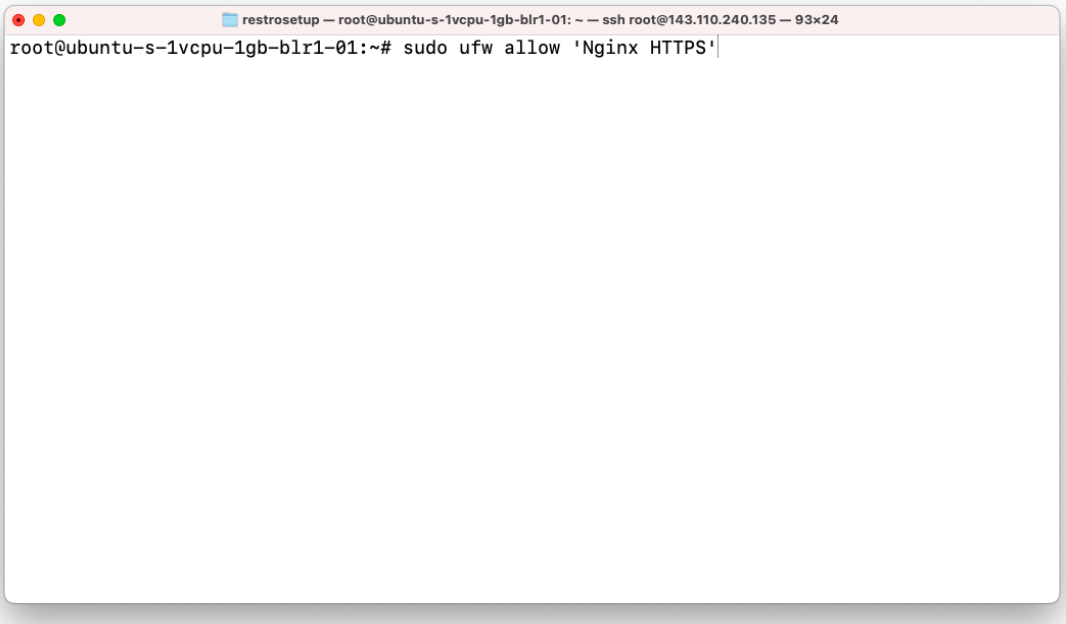
That's it, your backend server is started on your Server's IP. But we've to link the domain too. For that we've to install and configure the nginx. So let's do it.

```
› sudo apt update
› sudo apt install nginx
```

Now we will enable Firewall, run the following commands.

```
› sudo ufw app list
› sudo ufw allow 'Nginx HTTP'
› sudo ufw allow 'Nginx HTTPS'
› sudo ufw allow ssh
› sudo ufw enable
› sudo ufw status
```





After you allow rules in Firewall, we will need to add domain config and server config in nginx config, just follow the commands.

```
› cd /etc/nginx/sites-available/
› sudo nano demobackend.mydomain.com
```

Here look after the nano in the second command, you should put the domain of your backend, Ex: api.johnsrestro.com, the domain should be the same that we added in DNS config for Backend.

Once the editor is opened, put following text there, make sure to replace the highlighted values with your backend domain.

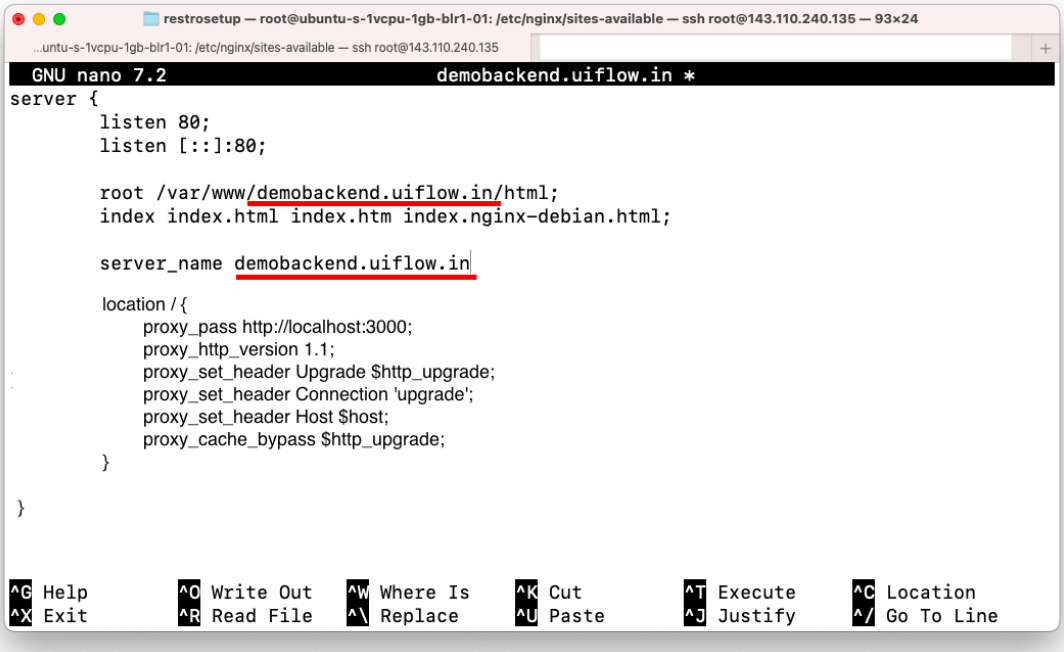
```
server {
  listen 80;
  listen [::]:80;

   root /var/www/demobackend.uiflow.in/html;

  index index.html index.htm index.nginx-debian.html;

  server_name demobackend.uiflow.in;

  location / {
    proxy_pass http://localhost:3000;
    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;
  }
}
```



Now save the file and exit the editor.

We'll copy this newly created file from sites-available to sites-enabled. Use following command.

```
› cp demobackend.uiflow.in ../sites-enabled/demobackend.uiflow.in
```

Make sure you replace the provided domain with your backend domain.

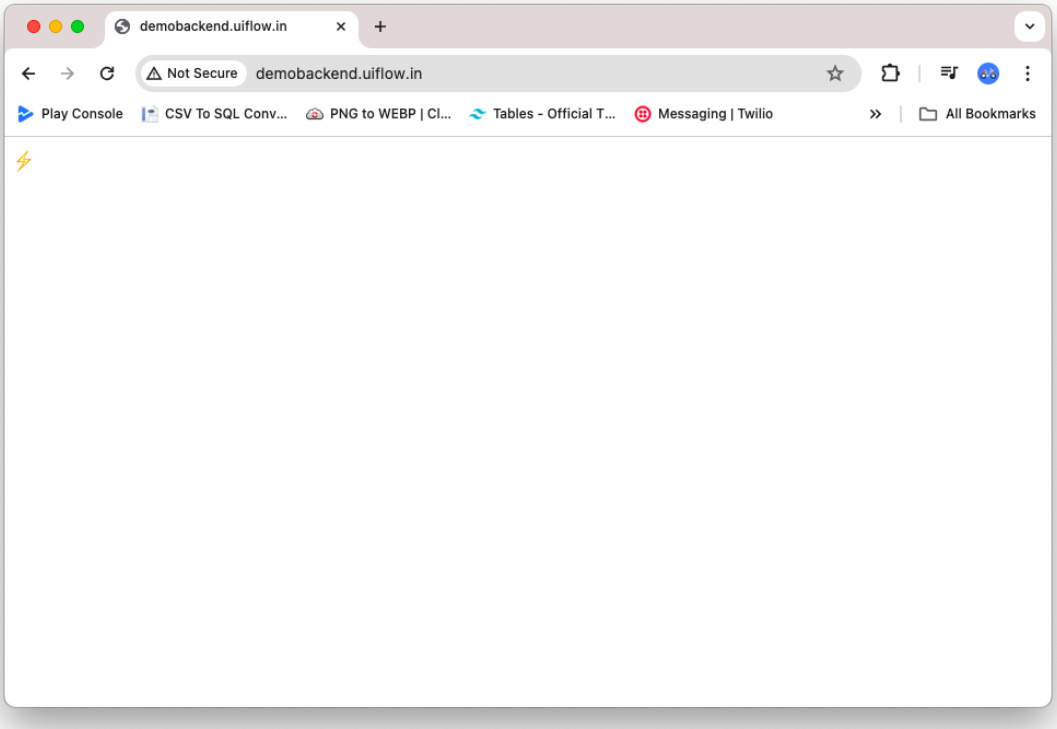
Once you do it, run the following command to check that your applied changes are error free.

```
› sudo nginx -t
```

If it says “test is successful” then it’s time to restart the nginx server.

```
› sudo systemctl restart nginx
```

After restarting the server, you can visit the backend domain on your browser.

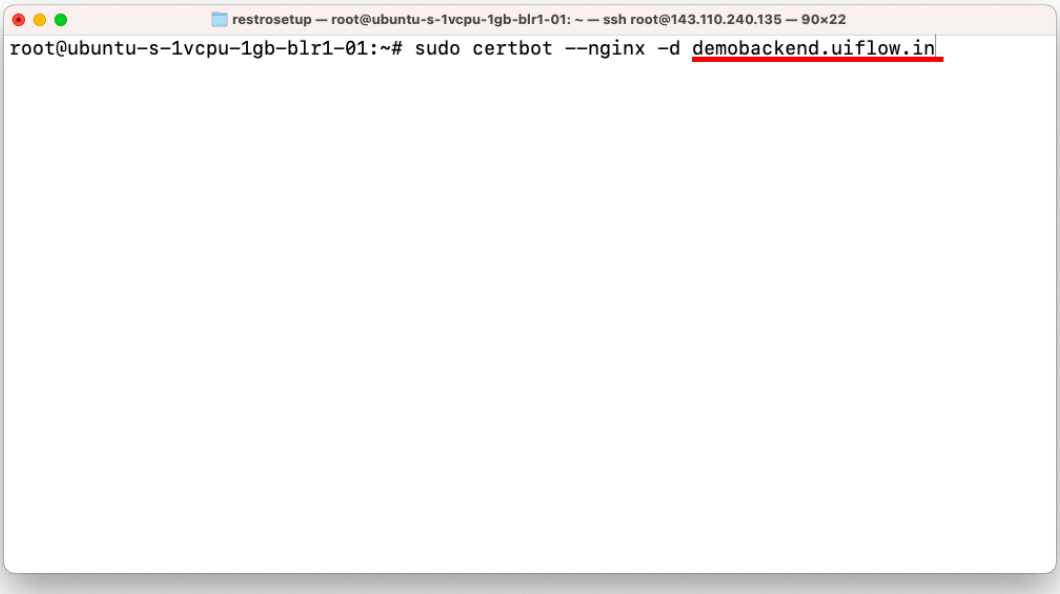


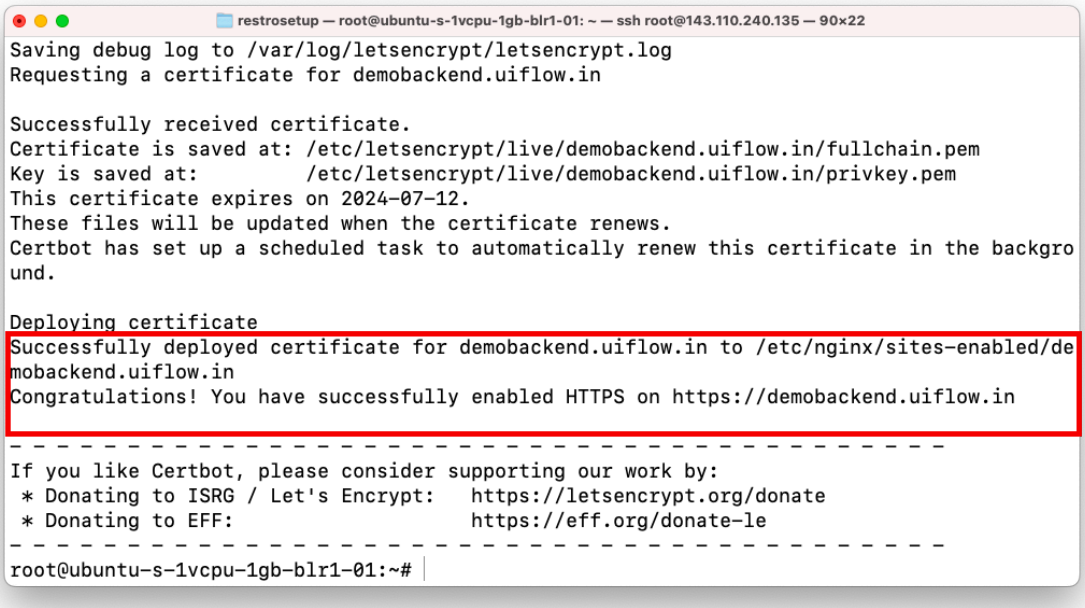
Hooray 🙌 we've configured the domain for backend successfully! But still it's not Secured, since SSL installation is pending. Don't worry, it's very quick.

```
› sudo apt install certbot python3-certbot-nginx
```

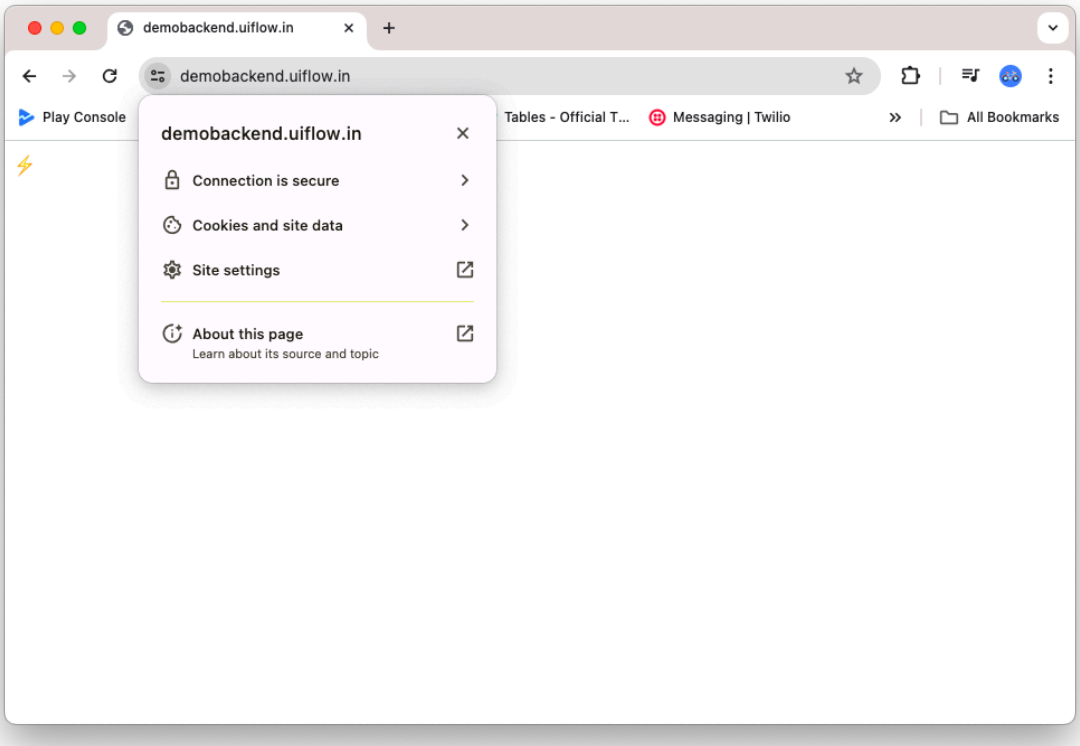

```
› sudo certbot –nginx -d demobackend.uiflow.in
```

Make sure you replace the domain with yours.





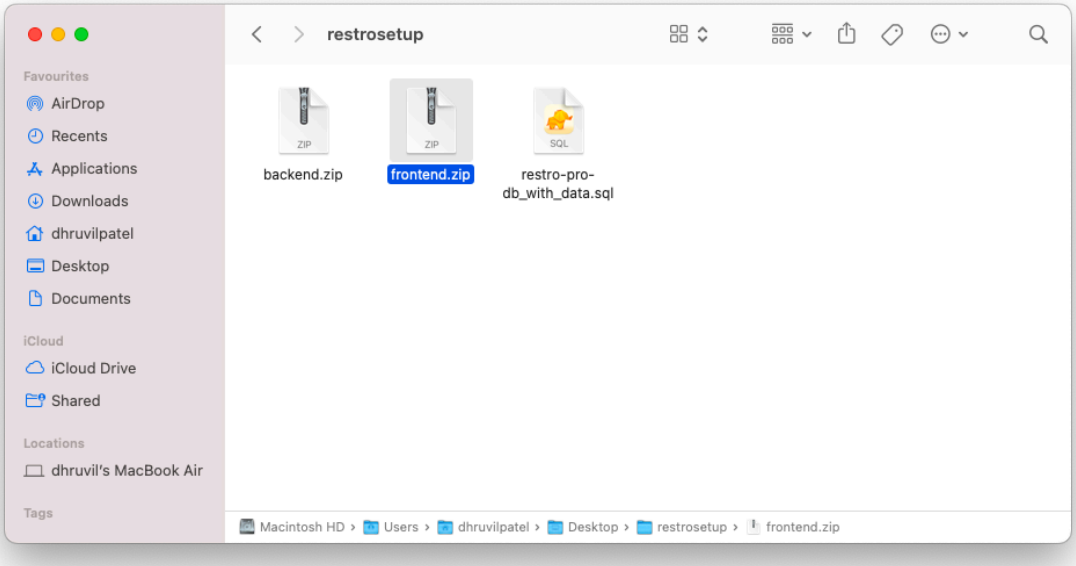
Now visit the backend domain URL again and refresh.

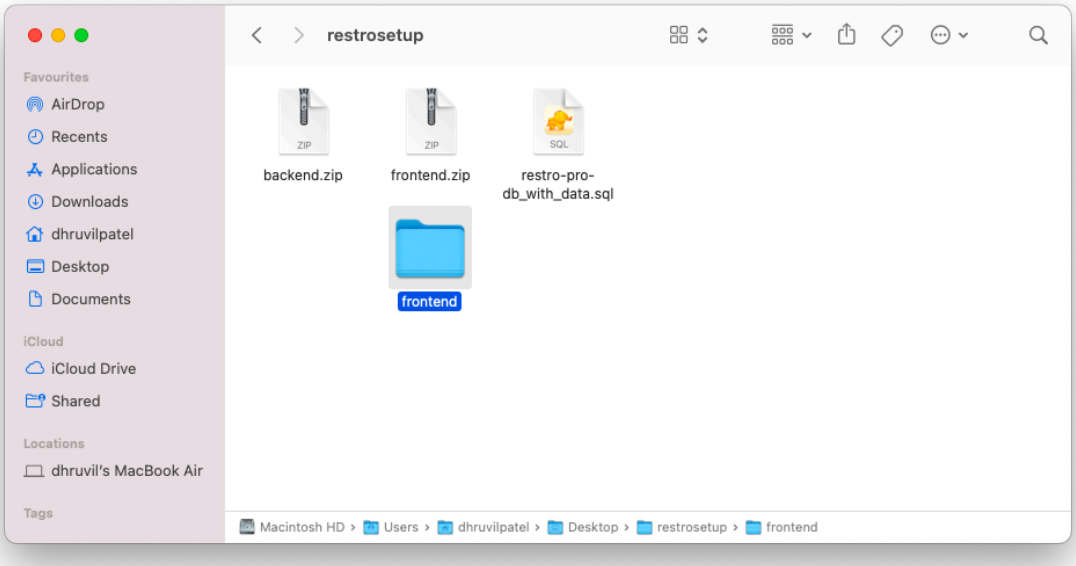


Yay! Our site is now HTTPS! 😎

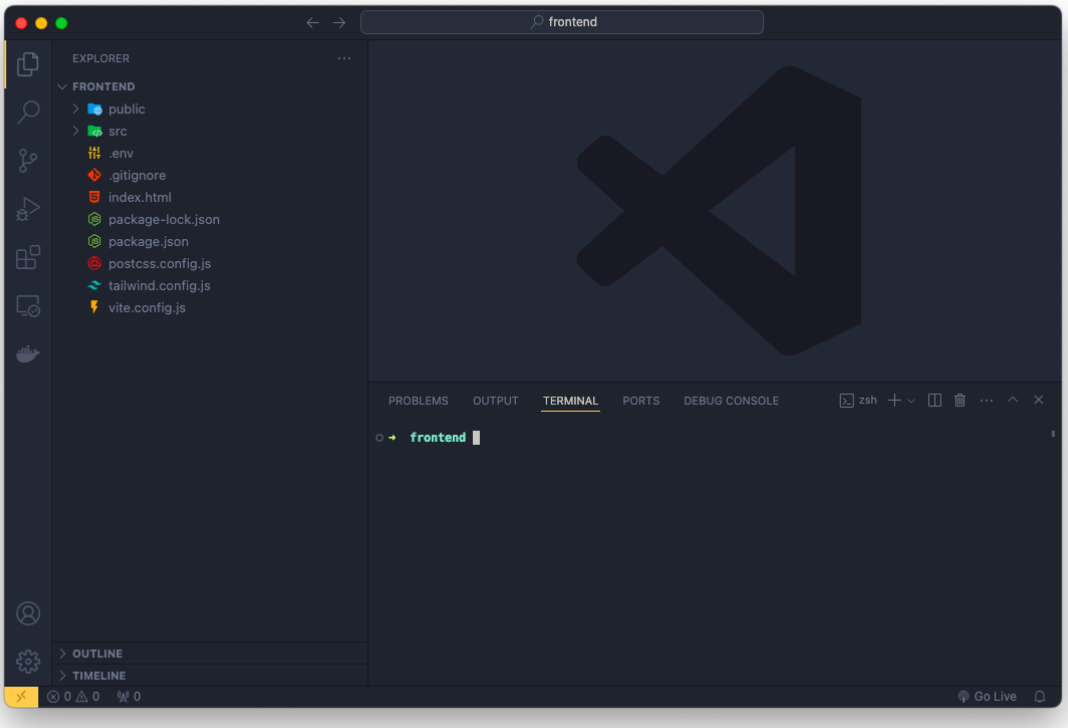
5. Frontend code Deployment

Unzip the frontend code.



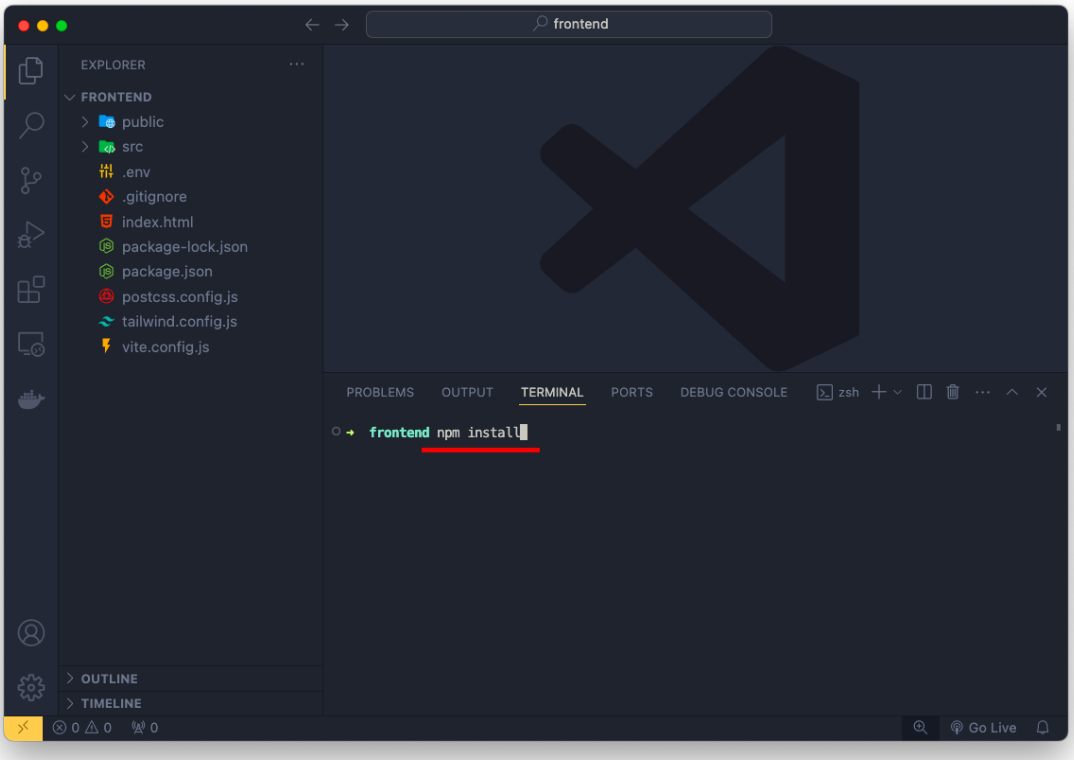


And open in the Code editor like VS Code.



Install frontend dependencies by running the following command in the frontend directory.

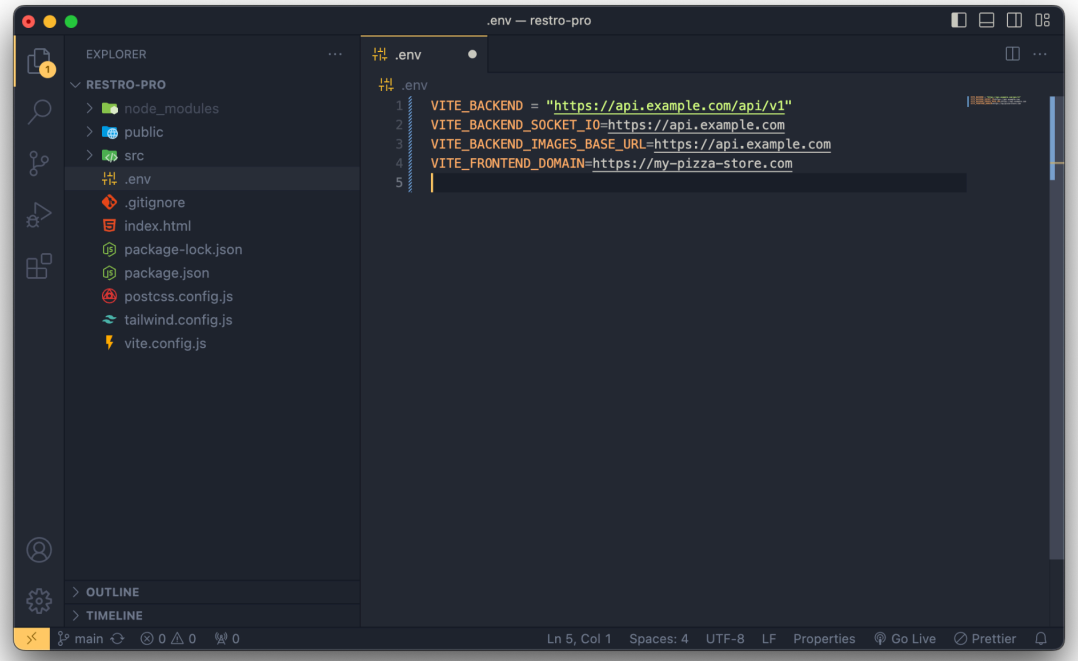
```
› npm install
```



Now open .env file, and replace the domain with your backend server domain.

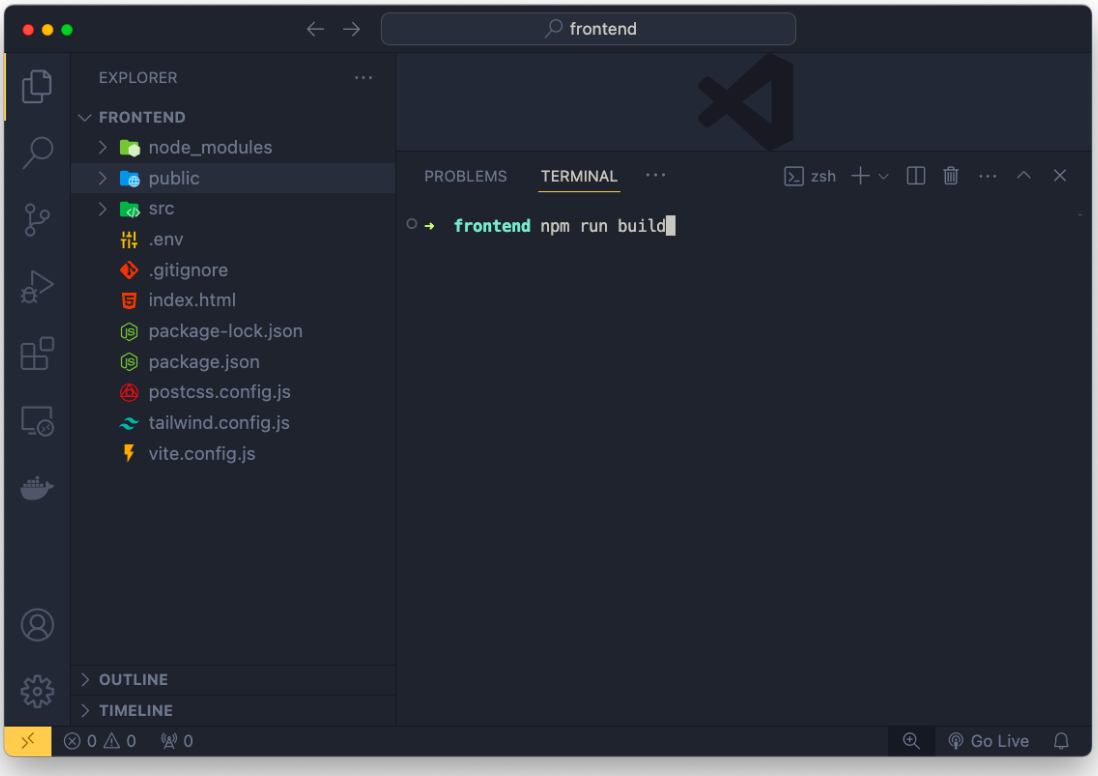
```
VITE_BACKEND=”https://api.example.com/api/v1”
VITE_BACKEND_SOCKET_IO=”https://api.example.com”
VITE_BACKEND_IMAGES_BASE_URL=https://api.example.com
VITE_FRONTEND_DOMAIN=https://my-pizza-store.com
```

Note: There should not be any trailing slashes.

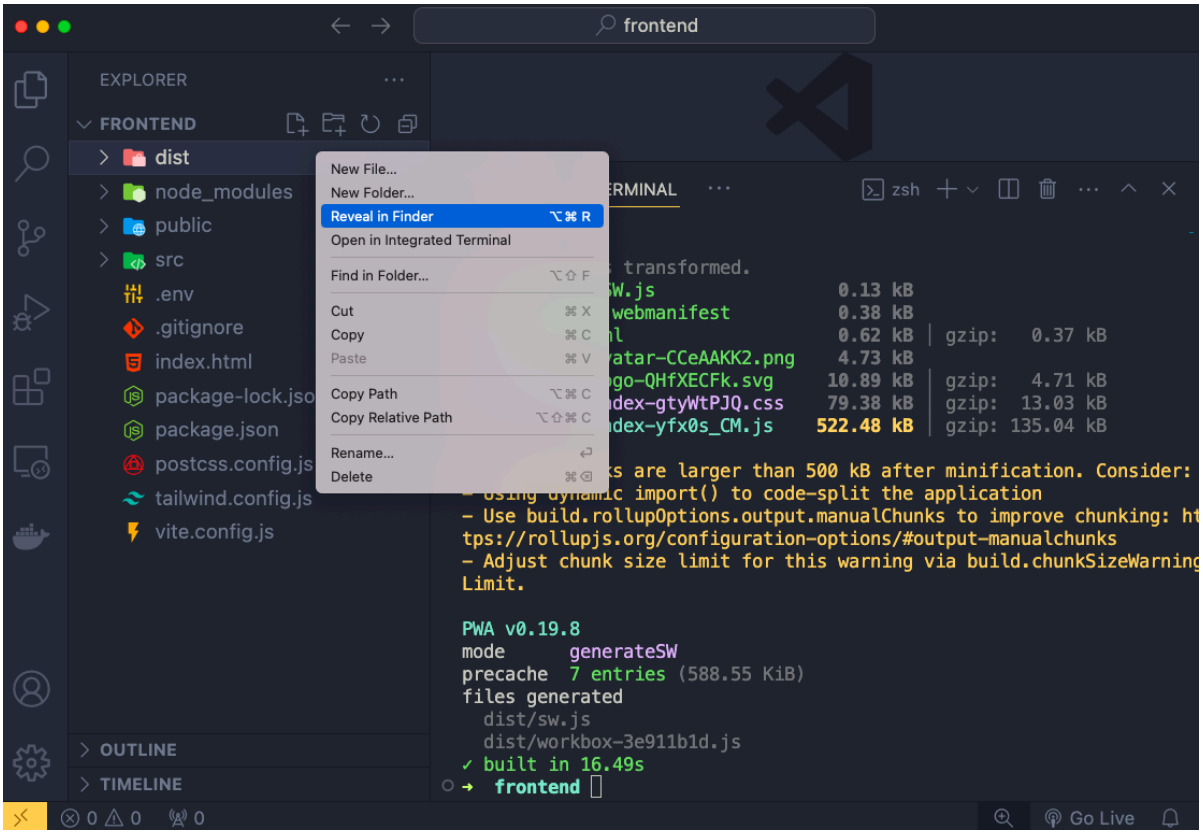


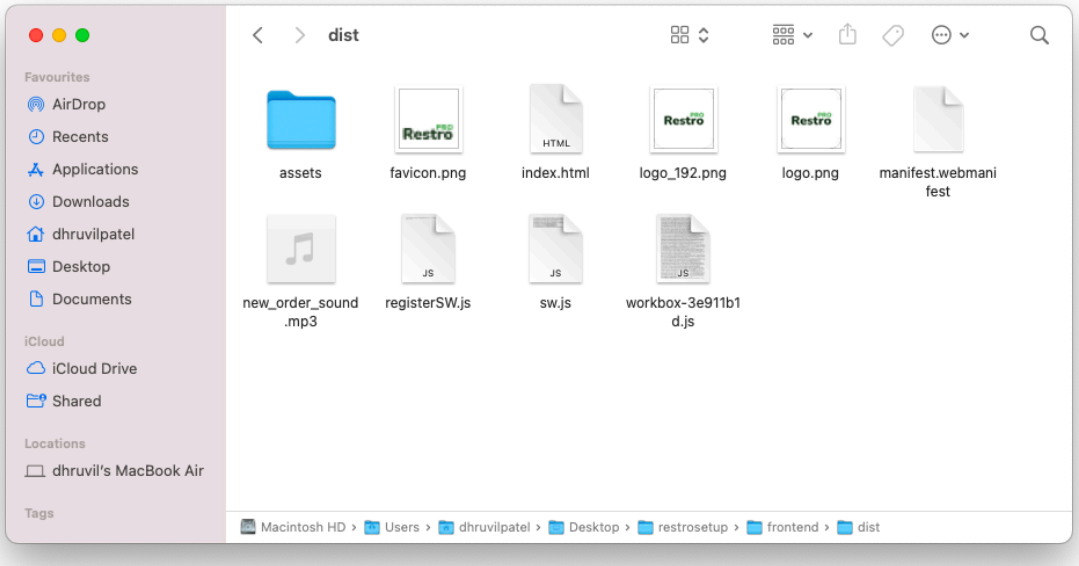
Now we will build the frontend code. Use following command.

```
› npm run build
```

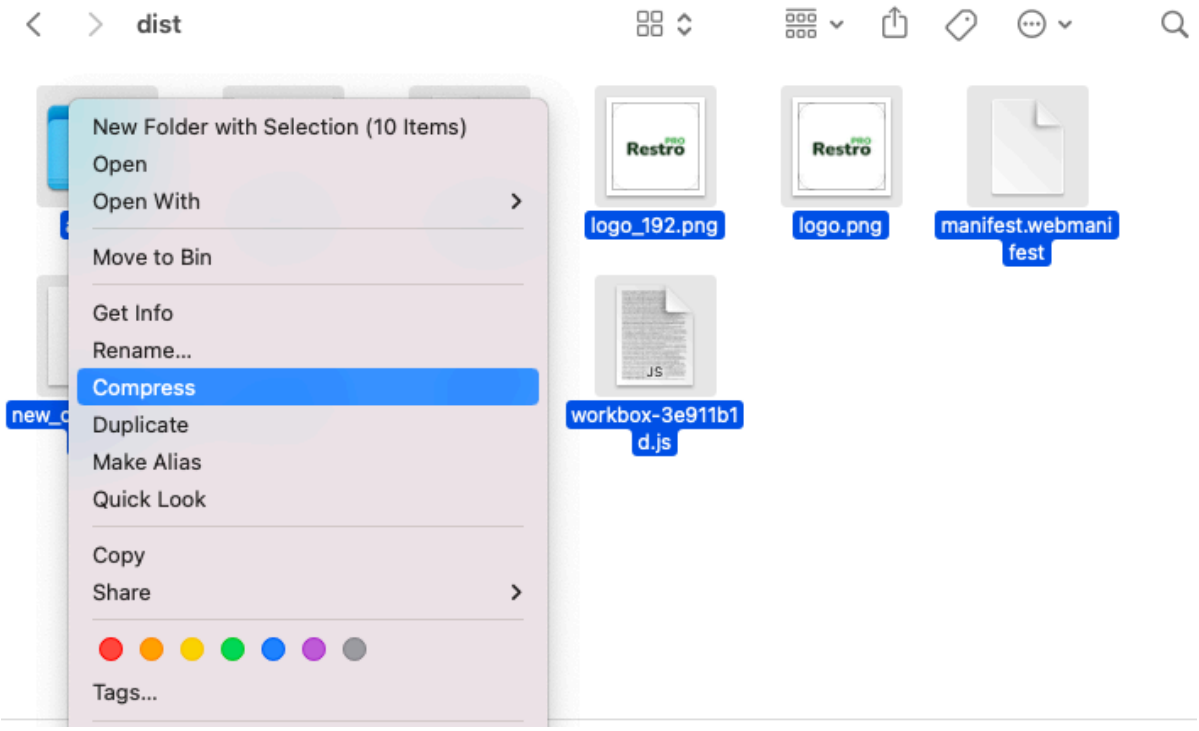


You'll see new folder appeared called "dist/" now open in Finder or File Explorer.





Select all files and make a zip file.

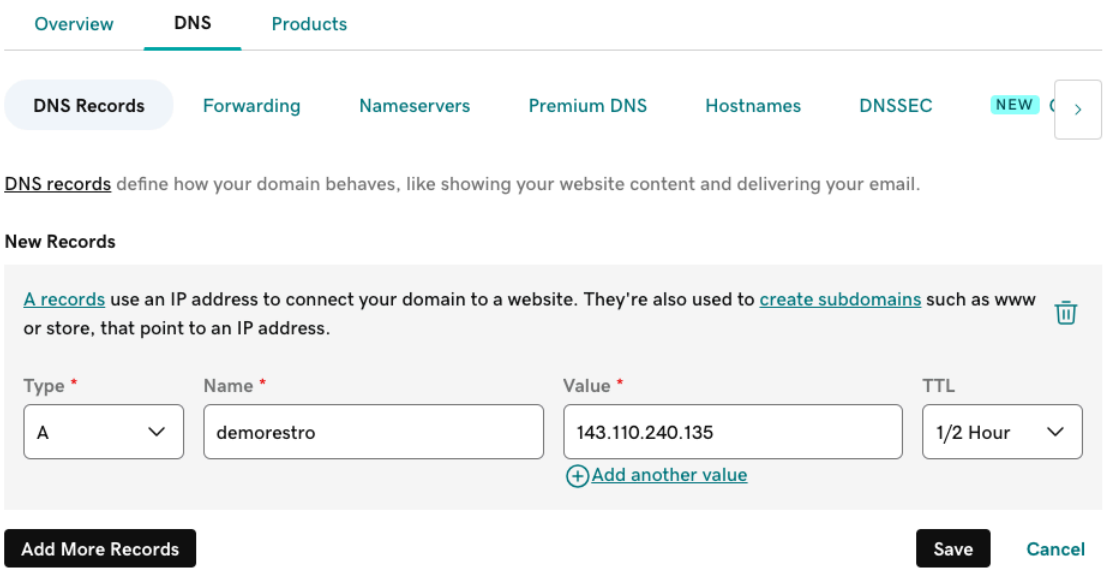


Now we will send this zip file to the Digital Ocean Droplet (Server).

Once send you can SSH into your server, and do quick “ls” to check.

Now remember when editing DNS records in Backend Configuration section, if you haven’t configured frontend domain that time then here is the time, let’s do it.

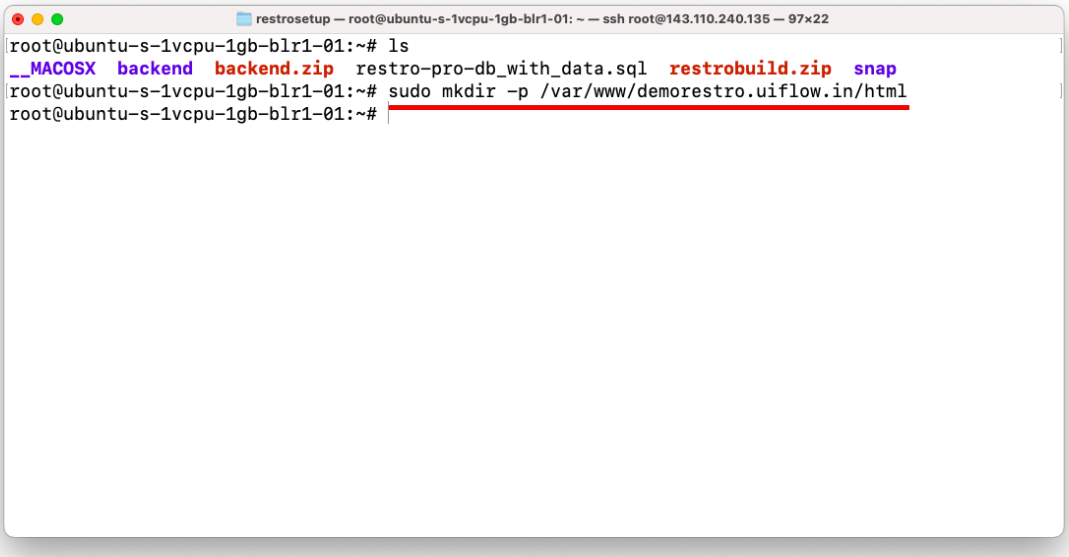
Goto DNS Settings of your domain provider and add “A” Record.



Now we'll configure the NGINX server to serve frontend code.

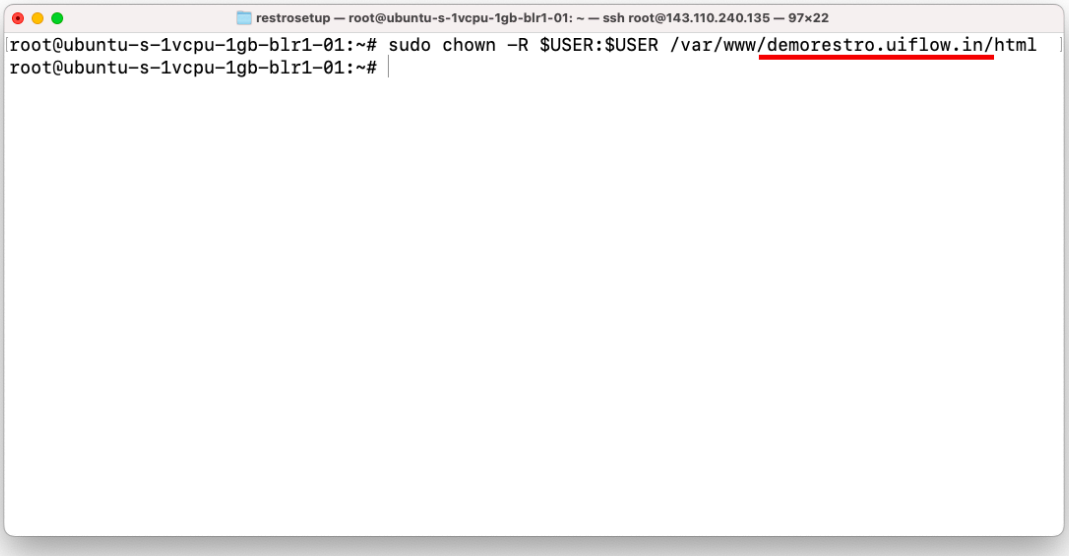
```
› sudo mkdir -p /var/www/demorestro.uiflow.in/html
```

Note: Make sure you replace the frontend domain with your frontend domain.

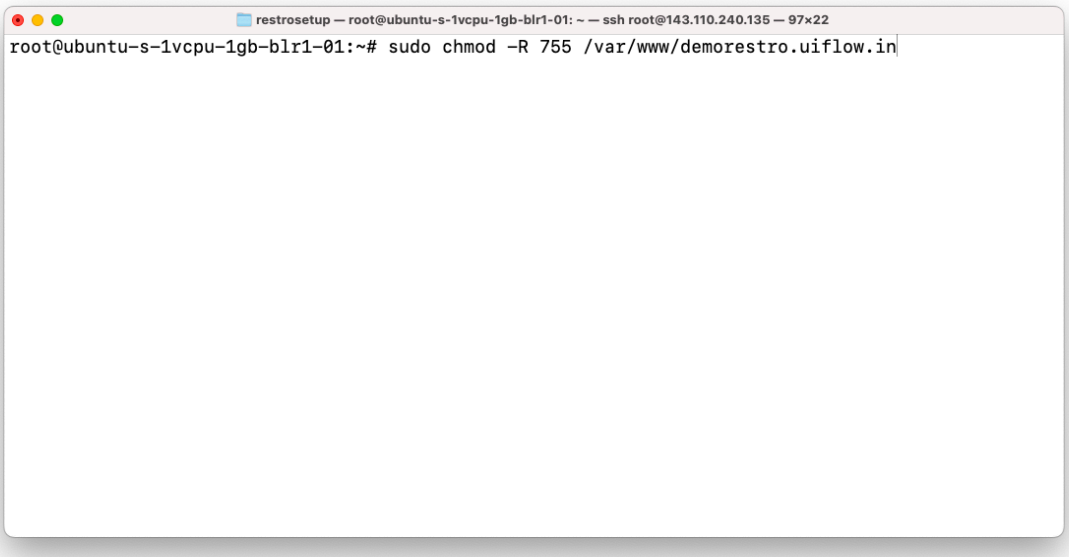


Grant the access to this newly created directory.

```
› sudo chown -R $USER: $USER /var/www/demorestro.uiflow.in/html
```



› sudo chmod -R 755 /var/www/demorestro.uiflow.in



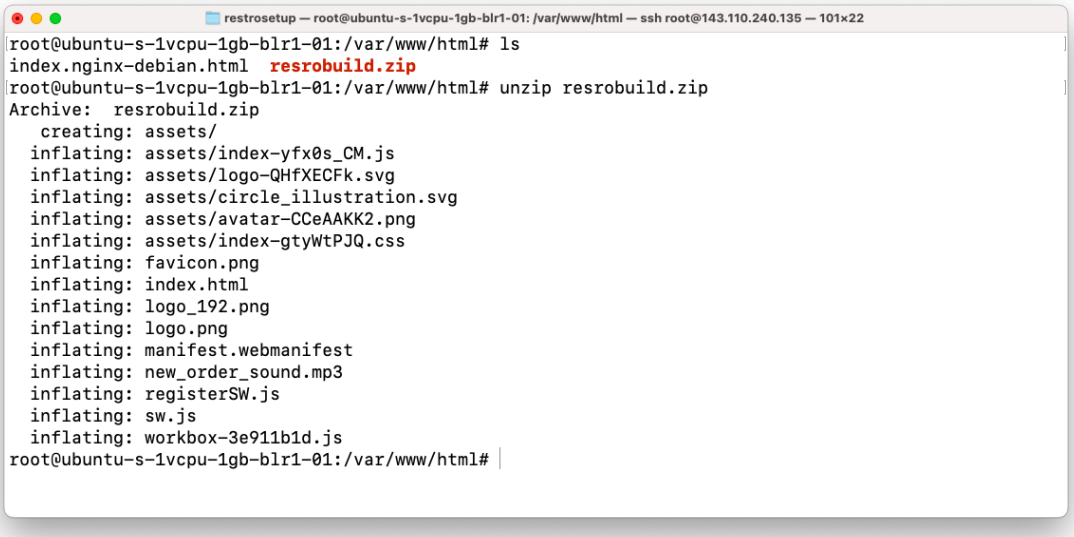
› sudo chmod -R 755 /var/www/demorestro.uiflow.in

Now move the frontend zip file to this folder.

```
› mv restrobuild.zip /var/www/html/resrobuild.zip
```

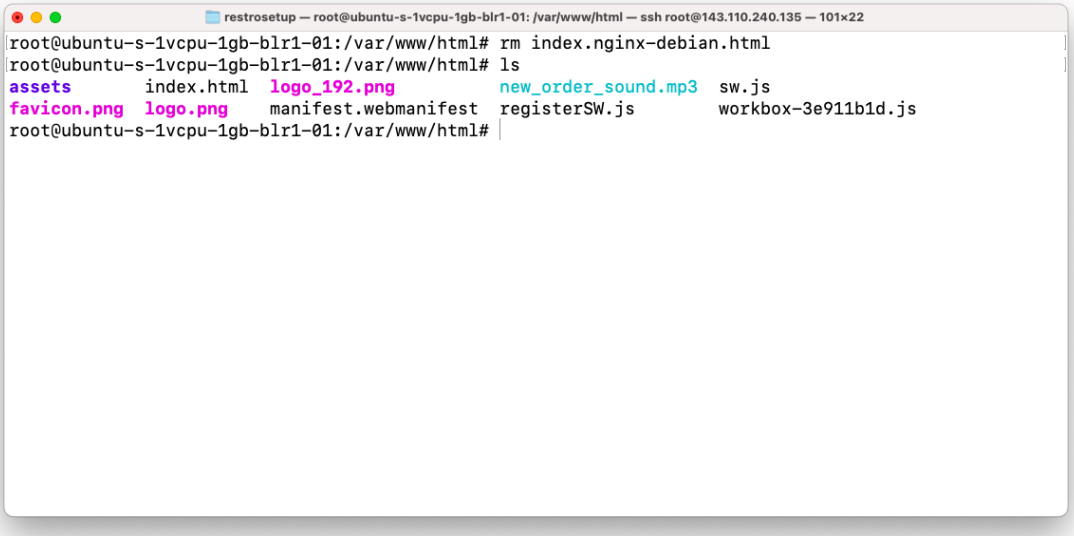
Unzip the code.

```
› unzip restrobuild.zip
```



Now you can remove zip file.

```
› rm restrobuild.zip
› rm index.nginx-debian.html
```



It's time to configure the NGINX settings to serve the frontend code to the frontend domain.

```
› sudo nano /etc/nginx/sites-available/demorestro.uiflow.in
```

Note: make sure you replace the domain with your frontend domain

And paste the following text.

```
server {
  listen 80;
  listen [::]:80;

  root /var/www/demorestro.uiflow.in/html;

  index index.html index.htm index.nginx-debian.html;
  server_name demorestro.uiflow.in;

  location / {
    try_files $uri $uri/ =404;
  }
}
```

Save the file and exit editor.

Now copy the file from sites-available to sites-enabled

```
› sudo ln -s /etc/nginx/sites-available/demorestro.uiflow.in /etc/nginx/sites-enabled/
```

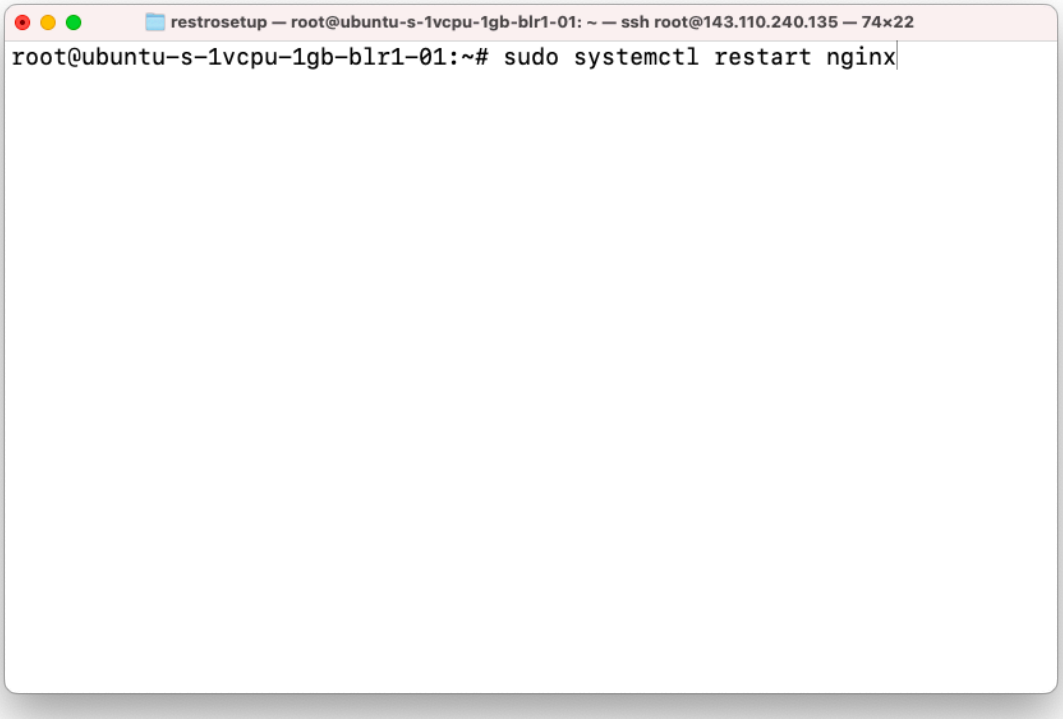
Note: Make sure you replace the domain.

Check that your applied changes are correct or not, by running following command.

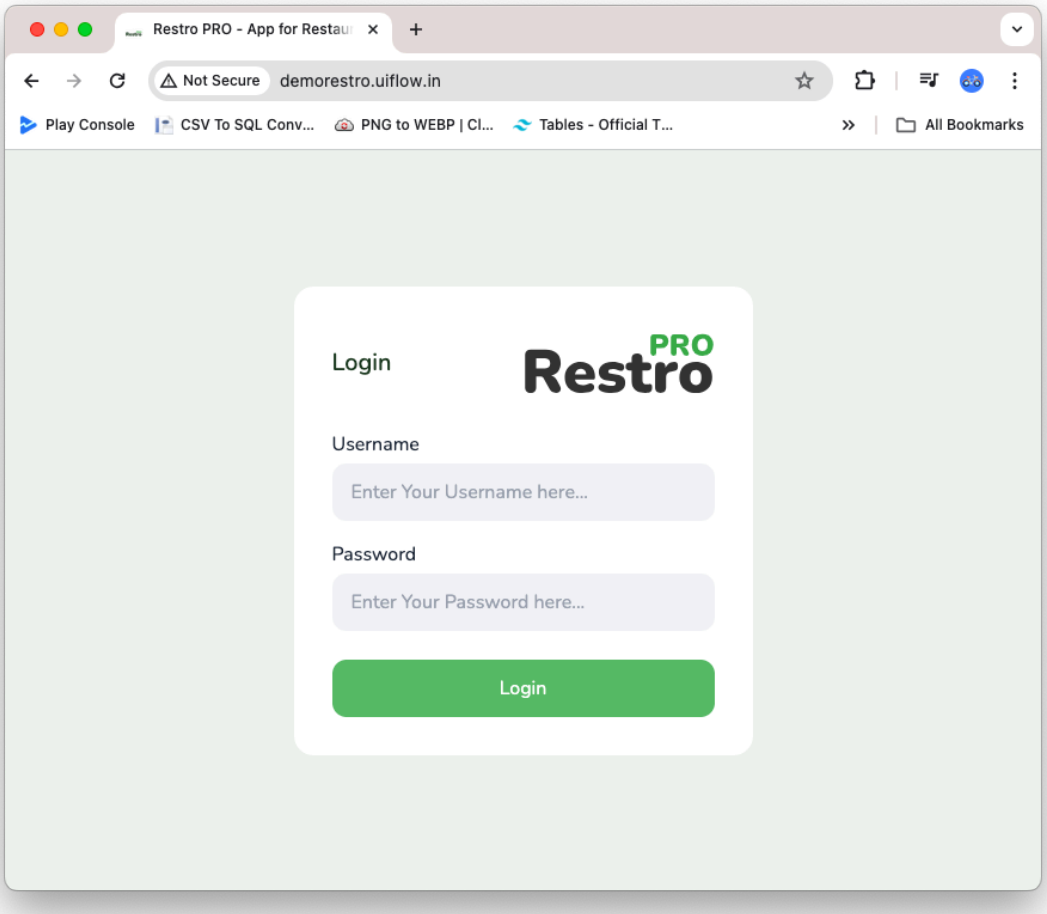
```
› sudo nginx -t
```

Now restart the nginx server.

```
› sudo systemctl restart nginx
```



Now you can visit the frontend domain, and you should see the app running.

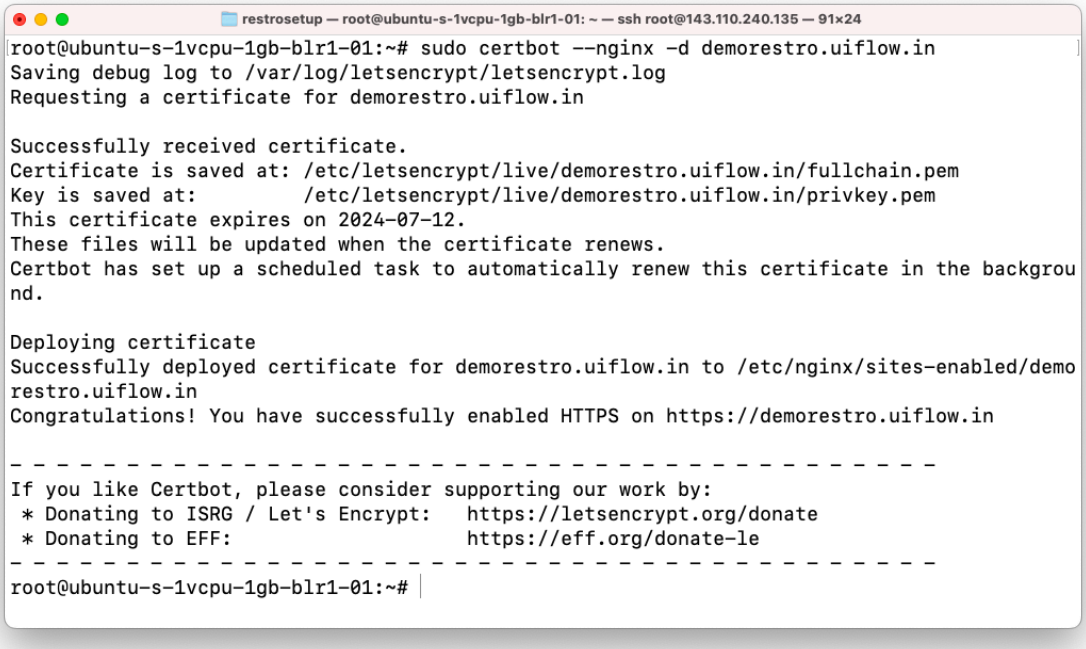


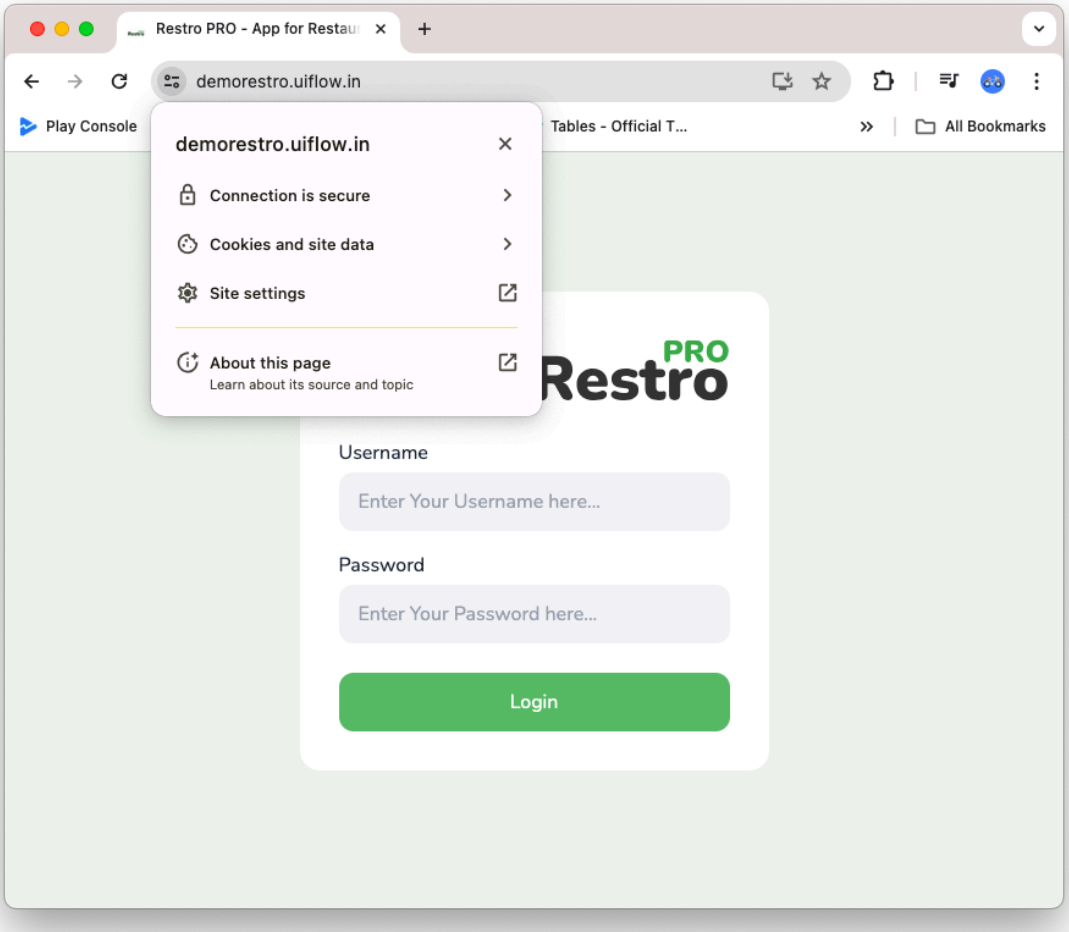
Hooray 🎉 only thing remaining is to install SSL for frontend domain.

```
› sudo certbot –nginx -d demorestro.uiflow.in
```

Note: make sure you put your frontend domain.


restrosetup — root@ubuntu-s-1vcpu-1gb-blr1-01: ~ — ssh root@143.110.240.135 — 91×22
root@ubuntu-s-1vcpu-1gb-blr1-01:~# sudo certbot --nginx -d demorestro.uiflow.in


restrosetup — root@ubuntu-s-1vcpu-1gb-blr1-01: ~ — ssh root@143.110.240.135 — 91×24
root@ubuntu-s-1vcpu-1gb-blr1-01:~# sudo certbot --nginx -d demorestro.uiflow.in
Saving debug log to /var/log/letsencrypt/letsencrypt.log
Requesting a certificate for demorestro.uiflow.in
Successfully received certificate.
Certificate is saved at: /etc/letsencrypt/live/demorestro.uiflow.in/fullchain.pem
Key is saved at: /etc/letsencrypt/live/demorestro.uiflow.in/privkey.pem
This certificate expires on 2024-07-12.
These files will be updated when the certificate renews.
Certbot has set up a scheduled task to automatically renew this certificate in the backgrou
nd.
Deploying certificate
Successfully deployed certificate for demorestro.uiflow.in to /etc/nginx/sites-enabled/demo
restro.uiflow.in
Congratulations! You have successfully enabled HTTPS on https://demorestro.uiflow.in
If you like Certbot, please consider supporting our work by:
* Donating to ISRG / Let's Encrypt: https://letsencrypt.org/donate
* Donating to EFF: https://eff.org/donate-le
root@ubuntu-s-1vcpu-1gb-blr1-01:~#



Yay! App is now HTTPS SSL.

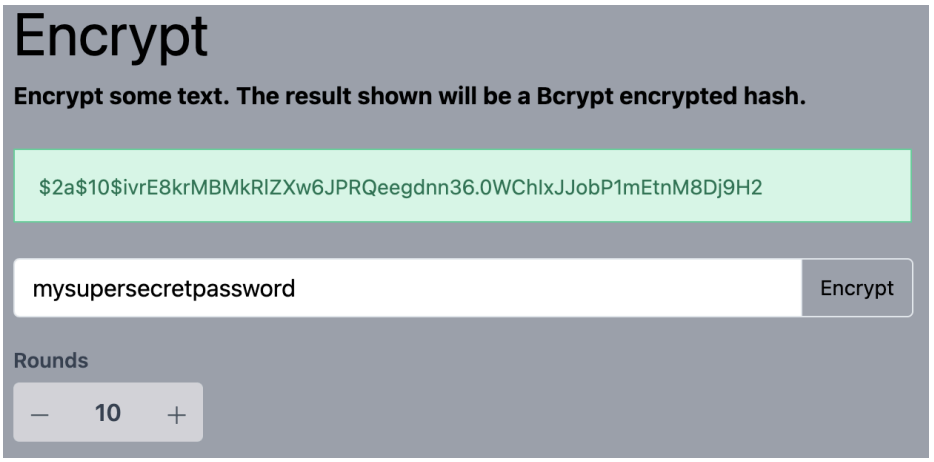
6. Configuring Settings of RestroPRO

6.1: Demo and App Settings

https://www.youtube.com/watch?v=WuSNCqsSnk0

6.2: Creating Super Admin User

Open https://bcrypt-generator.com/ and paste your plain text password in the field and use “10” rounds. And click on the Encrypt button. Now copy the encrypted text and paste it somewhere temporary.



Login to your server and open up mysql cli. And run the following query. Make sure to replace the placeholder values;

```
INSERT INTO `superadmins` VALUES
('example@gmail.com','paste_here_your_encrypted_password','Super
Admin user name');
```

7. Our Installation & Customisation Service

We understand that setting up a Node.js and React project on a server can be a technical task. That's why we offer a deployment service to take care of everything for you. This service is perfect for those who are not comfortable with the technical aspects or simply want to save time.

Contact us on: hi@uiflow.in

Looking for customisation? You're welcome Contact us on: hi@uiflow.in

Looking for a Development Outsourcing Service?
We offer custom app development services. Our team of experienced developers & designers will work closely with you to understand your specific needs and build a custom application that perfectly fits your workflow.
Contact us on: hi@uiflow.in

Crafting the Perfect Software. We Build it for You.

Our software empowers businesses. This POS system is a prime example, but we don't stop there. ***Need a custom solution?*** Contact us today - our team will craft a proposal within 24 hours.

hi@uiflow.in