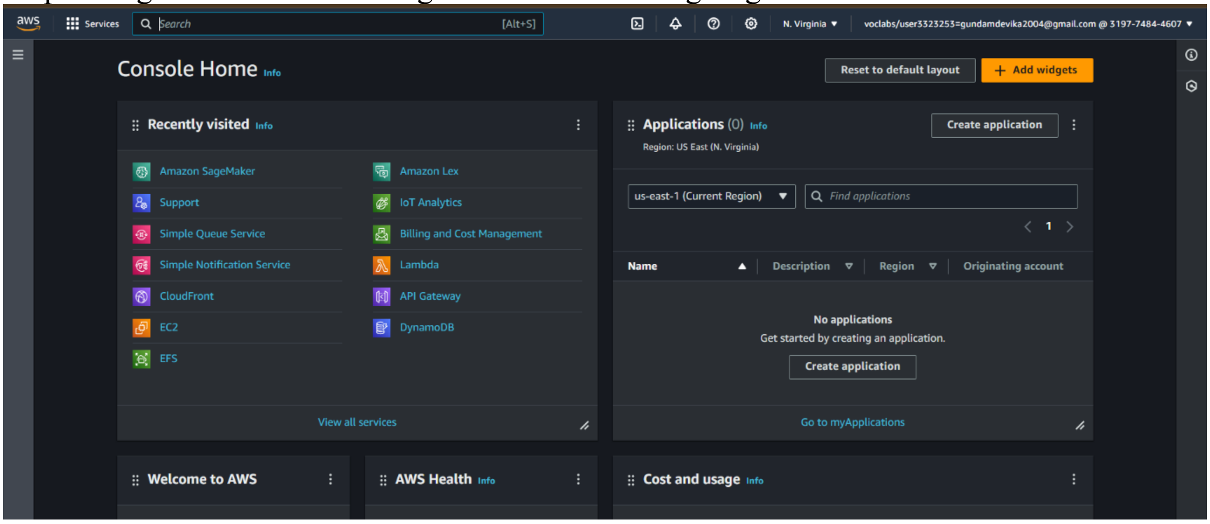
Aim: Migrate a website from local server to Cloud using Docker.

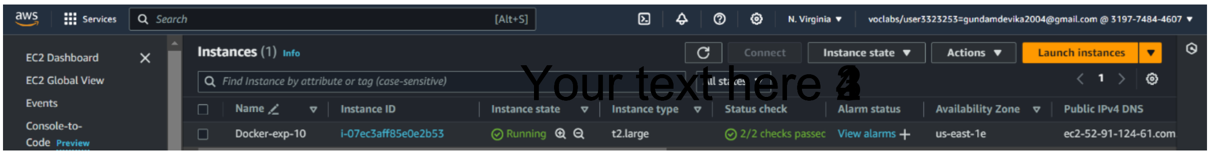
Step 1: Login to the AWS Management Console using Login Credentials



No applications

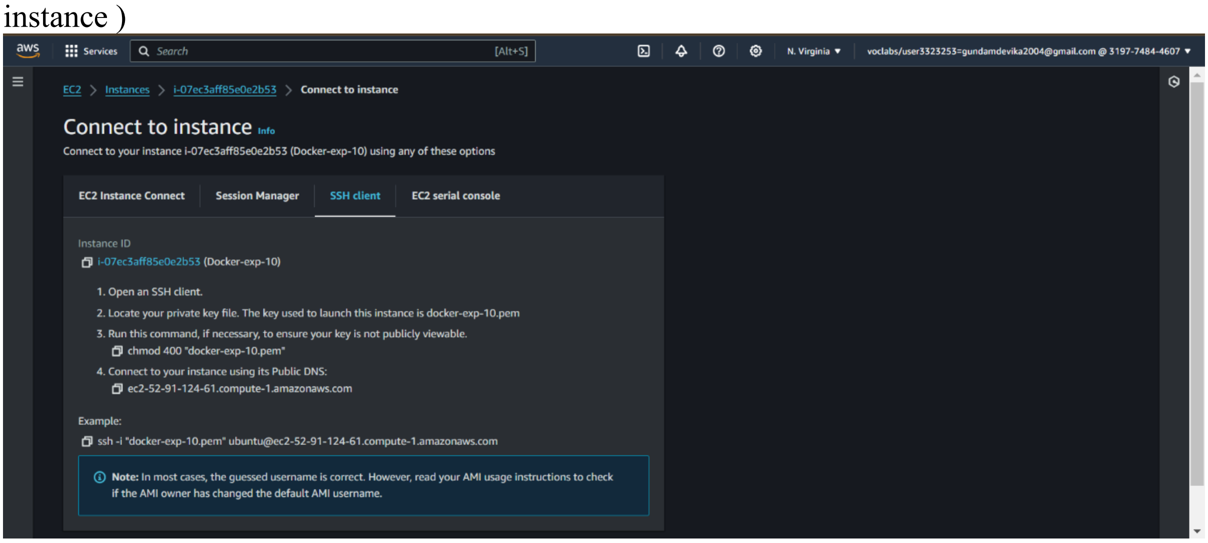
Get started by creating an application.

Step 2: Navigate to the EC2 Service by using Search bar and Launch an EC2 instance with required storage (30GB) and type (t2.large) and Launch the instance



Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
Docker-exp-10	i-07ec3aff85e0e2b53	Running	t2.large	2/2 checks passec	View alarms +	us-east-1e	ec2-52-91-124-61.com

Step 3: Connect to the instance through SSH Client (or through any other way to connect to instance)



aws
Services
Search
[Alt+S]
N. Virginia
voclabs/user3323253=gundamdevika2004@gmail.com @ 3197-7484-4607
EC2 > Instances > i-07ec3aff85e0e2b53 > Connect to instance
EC2 Instance Connect
Session Manager
SSH client
EC2 serial console

Connect to instance Info

Connect to your instance i-07ec3aff85e0e2b53 (Docker-exp-10) using any of these options

Instance ID

i-07ec3aff85e0e2b53 (Docker-exp-10)

1. Open an SSH client.
2. Locate your private key file. The key used to launch this instance is docker-exp-10.pem
3. Run this command, if necessary, to ensure your key is not publicly viewable.

```
chmod 400 "docker-exp-10.pem"
```

4. Connect to your instance using its Public DNS:

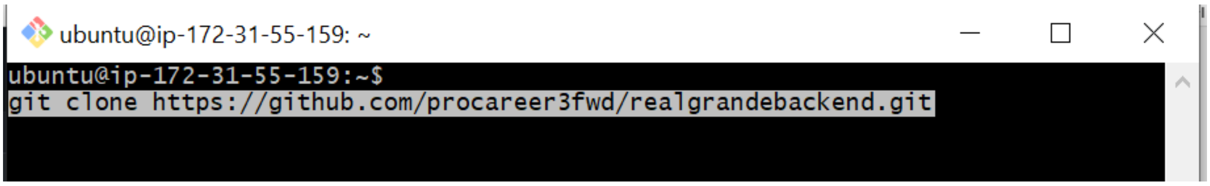
```
ec2-52-91-124-61.compute-1.amazonaws.com
```

Example:

```
ssh -i "docker-exp-10.pem" ubuntu@ec2-52-91-124-61.compute-1.amazonaws.com
```

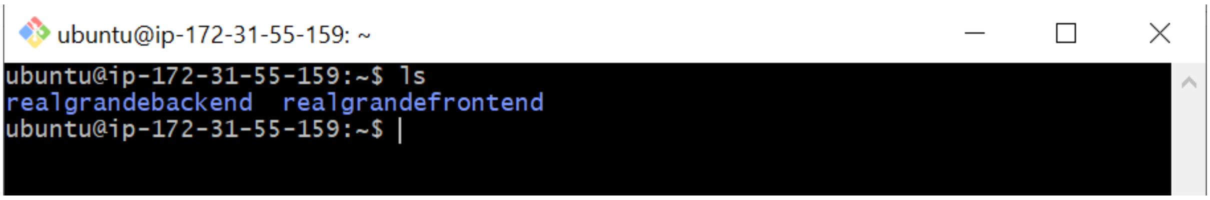
Note: In most cases, the guessed username is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI username.

Step 4: Login to the GitHub and get the URL’s for the FrontEnd and BackEnd repositories and clone them into the instance connected, such as $git clone https://url-to-frontend.git



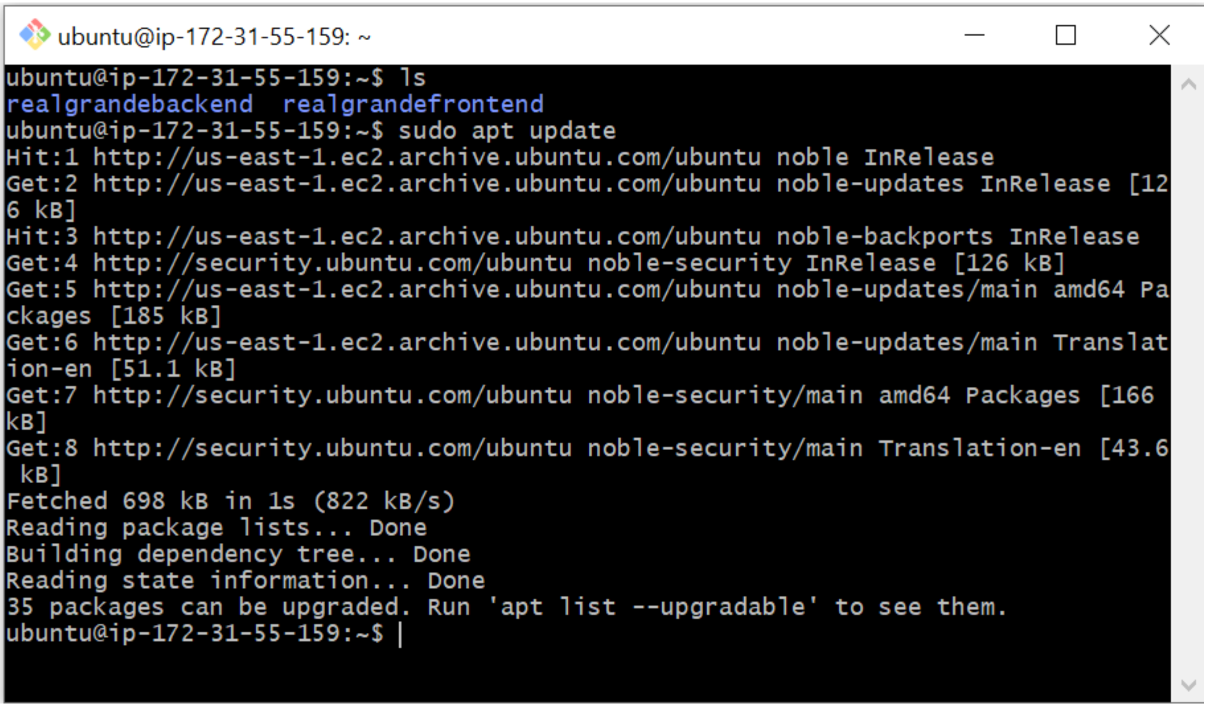
```
ubuntu@ip-172-31-55-159:~$
git clone https://github.com/procareer3fwd/realgrandebackend.git
```

Step 5: Perform same step for the backend repository also. Now we can check the two repositories have cloned to our local server. $ls



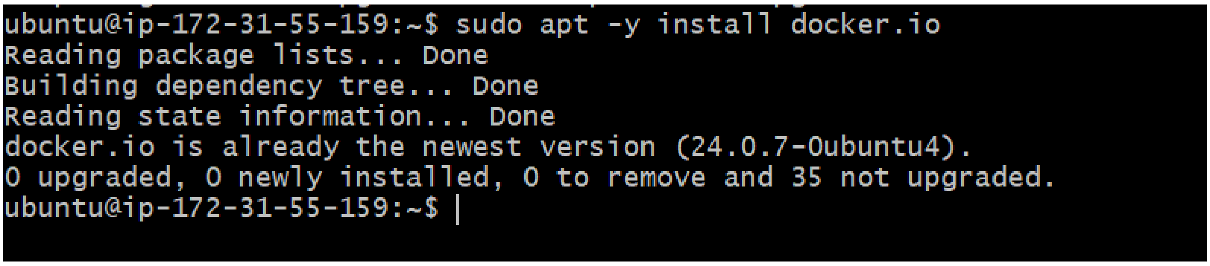
```
ubuntu@ip-172-31-55-159:~$ ls
realgrandebackend  realgrandefrontend
ubuntu@ip-172-31-55-159:~$ |
```

Step 6: Perform the update operation on Instance. $sudo apt update, all the required packages and updates will be done to the instance.



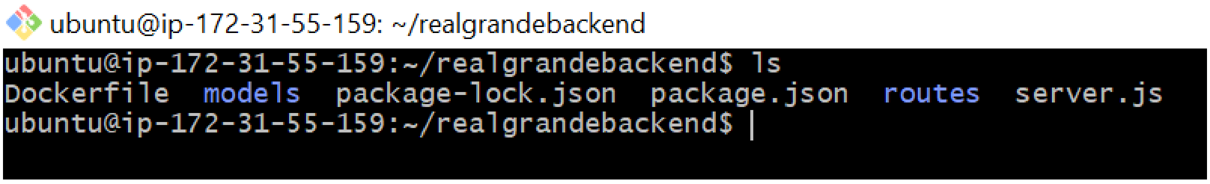
```
ubuntu@ip-172-31-55-159:~$ ls
realgrandebackend  realgrandefrontend
ubuntu@ip-172-31-55-159:~$ sudo apt update
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease [12
6 kB]
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Get:4 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 Pa
ckages [185 kB]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main Translat
ion-en [51.1 kB]
Get:7 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [166
kB]
Get:8 http://security.ubuntu.com/ubuntu noble-security/main Translation-en [43.6
 kB]
Fetched 698 kB in 1s (822 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
35 packages can be upgraded. Run 'apt list --upgradable' to see them.
ubuntu@ip-172-31-55-159:~$ |
```

Step 7: After updating the instance, now to create the images of our front end and back end we need to install the docker on the local machine. $sudo apt -y install docker.io



```
ubuntu@ip-172-31-55-159:~$ sudo apt -y install docker.io
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
docker.io is already the newest version (24.0.7-0ubuntu4).
0 upgraded, 0 newly installed, 0 to remove and 35 not upgraded.
ubuntu@ip-172-31-55-159:~$ |
```

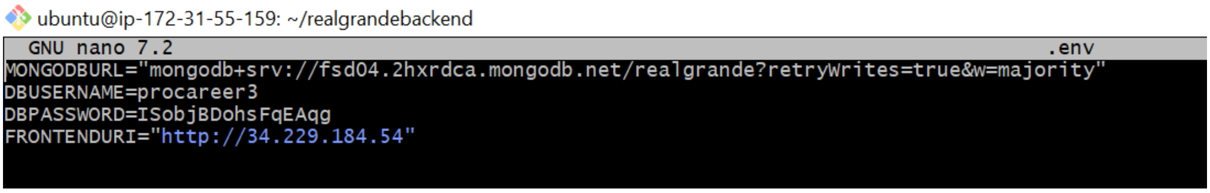
Step 8: Now Navigate to the realgrandebackend folder and search for .env file wether it is exists in the folder or not, if not we need to create the .env file



ubuntu@ip-172-31-55-159: ~/realgrandebackend

```
ubuntu@ip-172-31-55-159:~/realgrandebackend$ ls
Dockerfile  models  package-lock.json  package.json  routes  server.js
ubuntu@ip-172-31-55-159:~/realgrandebackend$ |
```

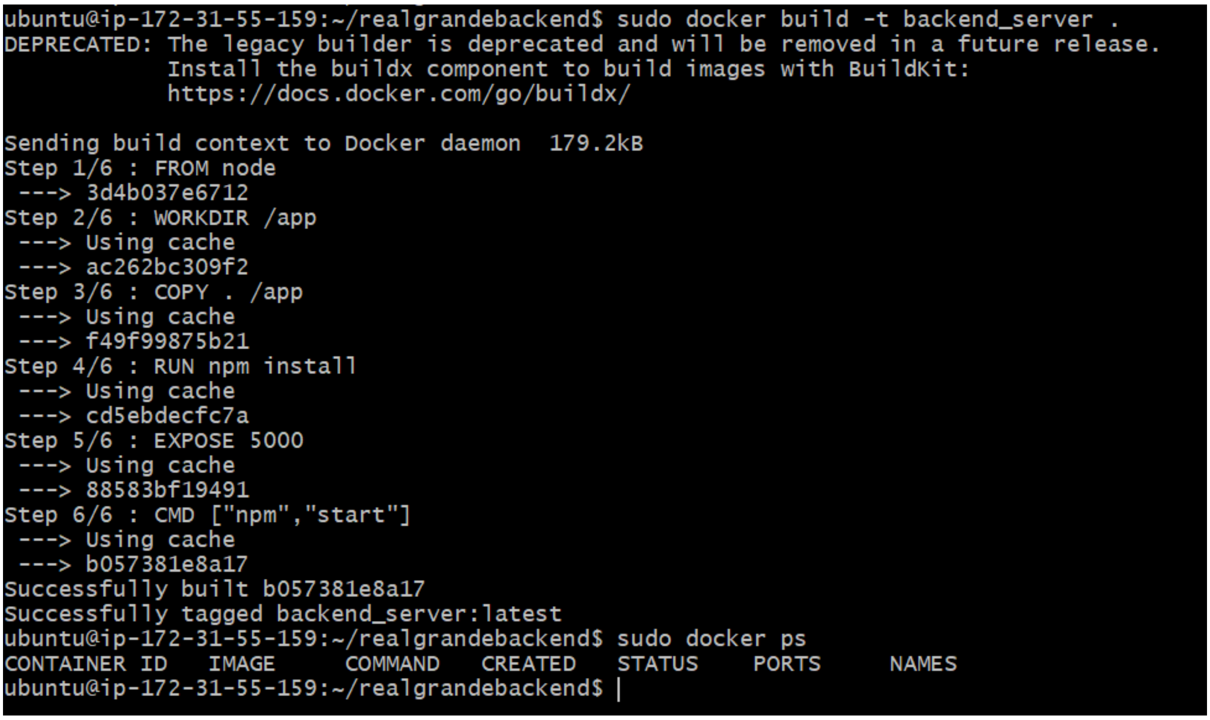
Step 9: To create .env file, run the command $nano .env and specify the database credentials in this file such as mongodburl, dbusername, dbpassword, in frontenduri specify the public ip of our instance



ubuntu@ip-172-31-55-159: ~/realgrandebackend

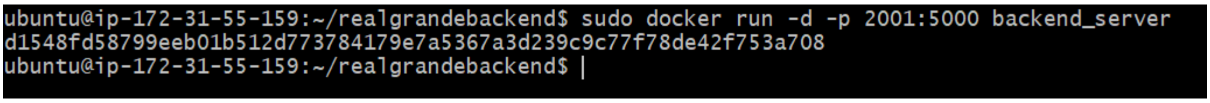
```
  GNU nano 7.2                                                                   .env
MONGODBURL="mongodb+srv://fsd04.2hxrdca.mongodb.net/realgrande?retryWrites=true&w=majority"
DBUSERNAME=procareer3
DBPASSWORD=ISobjBDohsFqEAqg
FRONTENDURI="http://34.229.184.54"
```

Step 10: Now we need to build the docker image for the backend, run the command to build the docker image. $sudo docker build -t backend_server . and check the docker image $sudo docker ps



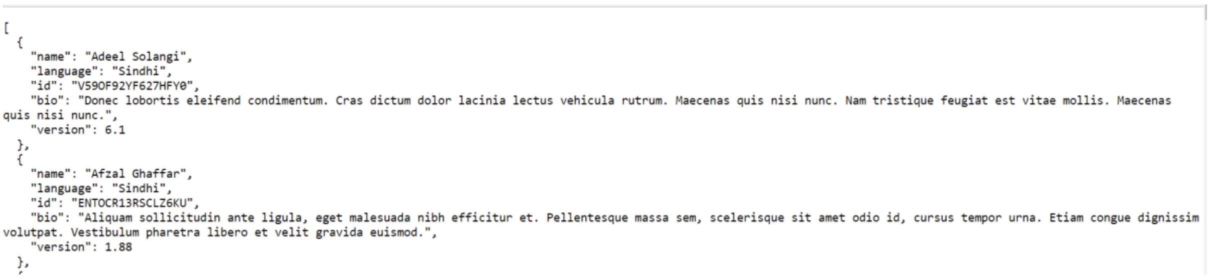
```
ubuntu@ip-172-31-55-159:~/realgrandebackend$ sudo docker build -t backend_server .
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
            Install the buildx component to build images with BuildKit:
            https://docs.docker.com/go/buildx/

Sending build context to Docker daemon  179.2kB
Step 1/6 : FROM node
 ---> 3d4b037e6712
Step 2/6 : WORKDIR /app
 ---> Using cache
 ---> ac262bc309f2
Step 3/6 : COPY . /app
 ---> Using cache
 ---> f49f99875b21
Step 4/6 : RUN npm install
 ---> Using cache
 ---> cd5ebdecfc7a
Step 5/6 : EXPOSE 5000
 ---> Using cache
 ---> 88583bf19491
Step 6/6 : CMD ["npm","start"]
 ---> Using cache
 ---> b057381e8a17
Successfully built b057381e8a17
Successfully tagged backend_server:latest
ubuntu@ip-172-31-55-159:~/realgrandebackend$ sudo docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
ubuntu@ip-172-31-55-159:~/realgrandebackend$ |
```

Step 11: Now run the image created for the backend $sudo docker run -d -p 2001:5000 backend_server



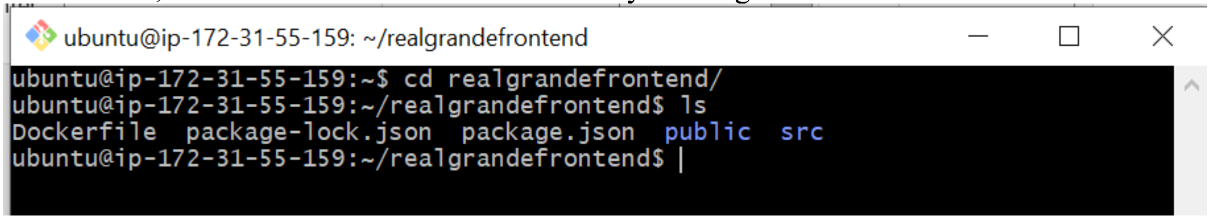
```
ubuntu@ip-172-31-55-159:~/realgrandebackend$ sudo docker run -d -p 2001:5000 backend_server
d1548fd58799eeb01b512d773784179e7a5367a3d239c9c77f78de42f753a708
ubuntu@ip-172-31-55-159:~/realgrandebackend$ |
```

Step 12: Now try to access the data in the browser by <Ec2_ip_address>:2001/api, we need to get the json data



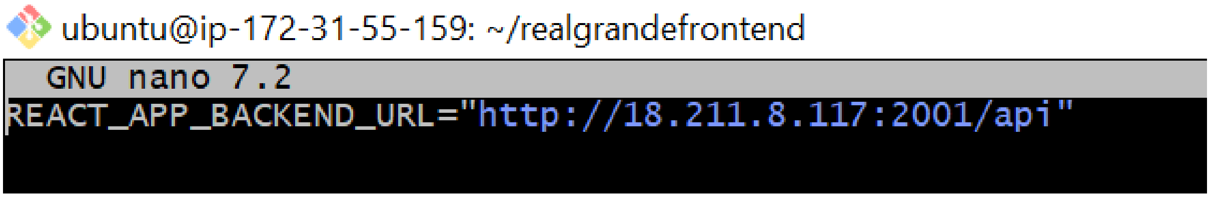
```
[
  {
    "name": "Adeel Solangi",
    "language": "Sindhi",
    "id": "V59OF92YF627HFY0",
    "bio": "Donec lobortis eleifend condimentum. Cras dictum dolor lacinia lectus vehicula rutrum. Maecenas quis nisi nunc. Nam tristique feugiat est vitae mollis. Maecenas
quis nisi nunc.",
    "version": 6.1
  },
  {
    "name": "Afzal Ghaffar",
    "language": "Sindhi",
    "id": "ENTOCR13RSCLZ6KU",
    "bio": "Aliquam sollicitudin ante ligula, eget malesuada nibh efficitur et. Pellentesque massa sem, scelerisque sit amet odio id, cursus tempor urna. Etiam congue dignissim
volutpat. Vestibulum pharetra libero et velit gravida euismod.",
    "version": 1.88
  },
```

Step 13: Perform the above steps for frontend, navigate to the frontend folder and search for the .env file, if not found create a new .env file by running command $nano .env



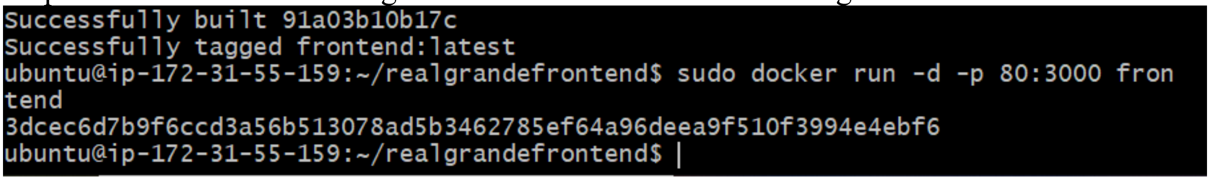
```
ubuntu@ip-172-31-55-159:~$ cd realgrandefrontend/
ubuntu@ip-172-31-55-159:~/realgrandefrontend$ ls
Dockerfile  package-lock.json  package.json  public  src
ubuntu@ip-172-31-55-159:~/realgrandefrontend$ |
```

Step 14: Replace our public ip address in the .env file



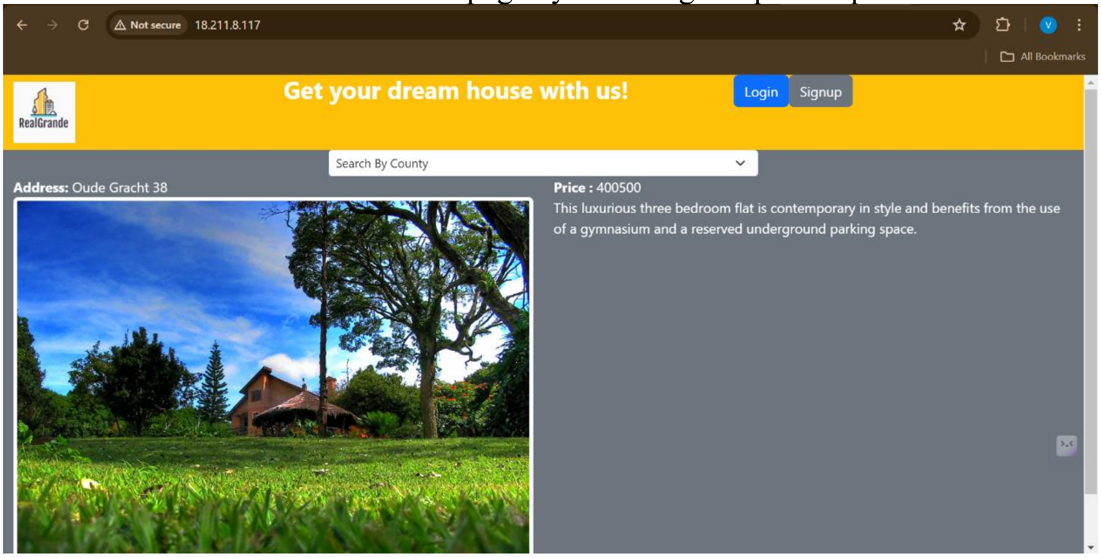
```
REACT_APP_BACKEND_URL="http://18.211.8.117:2001/api"
```

Step 15: build the docker image for the front end and run the image



```
Successfully built 91a03b10b17c
Successfully tagged frontend:latest
ubuntu@ip-172-31-55-159:~/realgrandefrontend$ sudo docker run -d -p 80:3000 fron
tend
3dcec6d7b9f6ccd3a56b513078ad5b3462785ef64a96deea9f510f3994e4ebf6
ubuntu@ip-172-31-55-159:~/realgrandefrontend$ |
```

Step 16: Now successfully we created the frontend and backend images by using docker containers and Access the front end page by browsing our public ip address of our instance.



Get your dream house with us!

Address: Oude Gracht 38

Price : 400500

This luxurious three bedroom flat is contemporary in style and benefits from the use of a gymnasium and a reserved underground parking space.