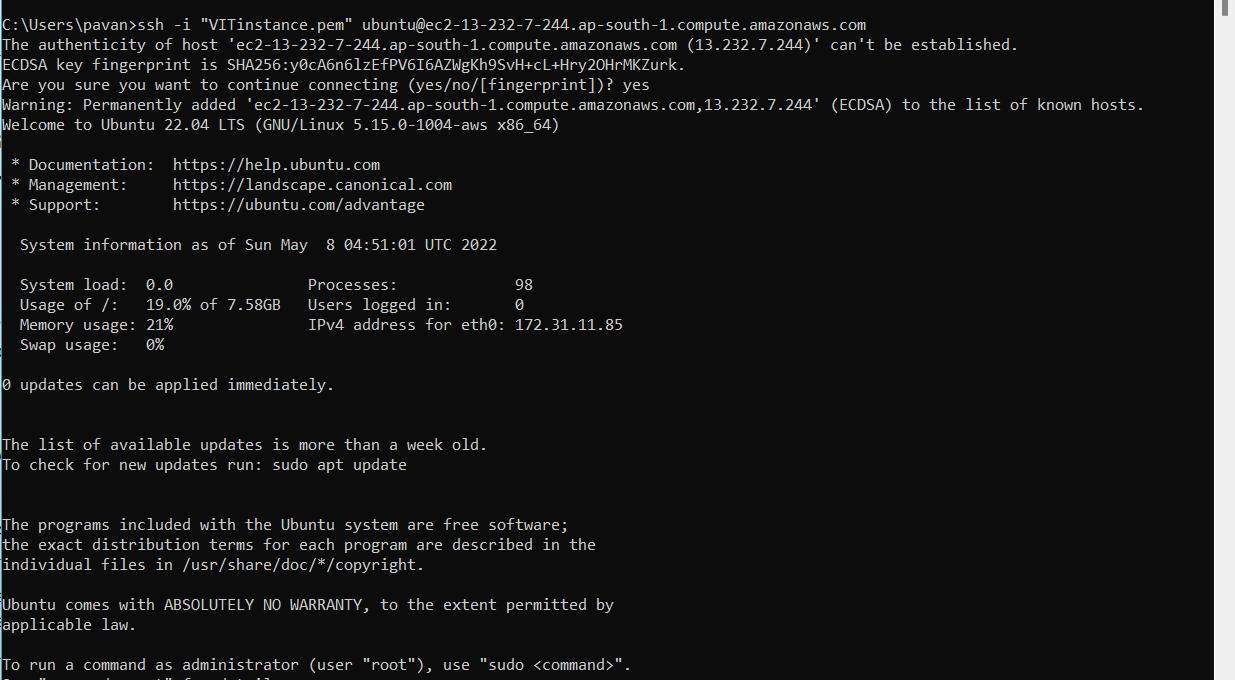
Deploying the application on the server

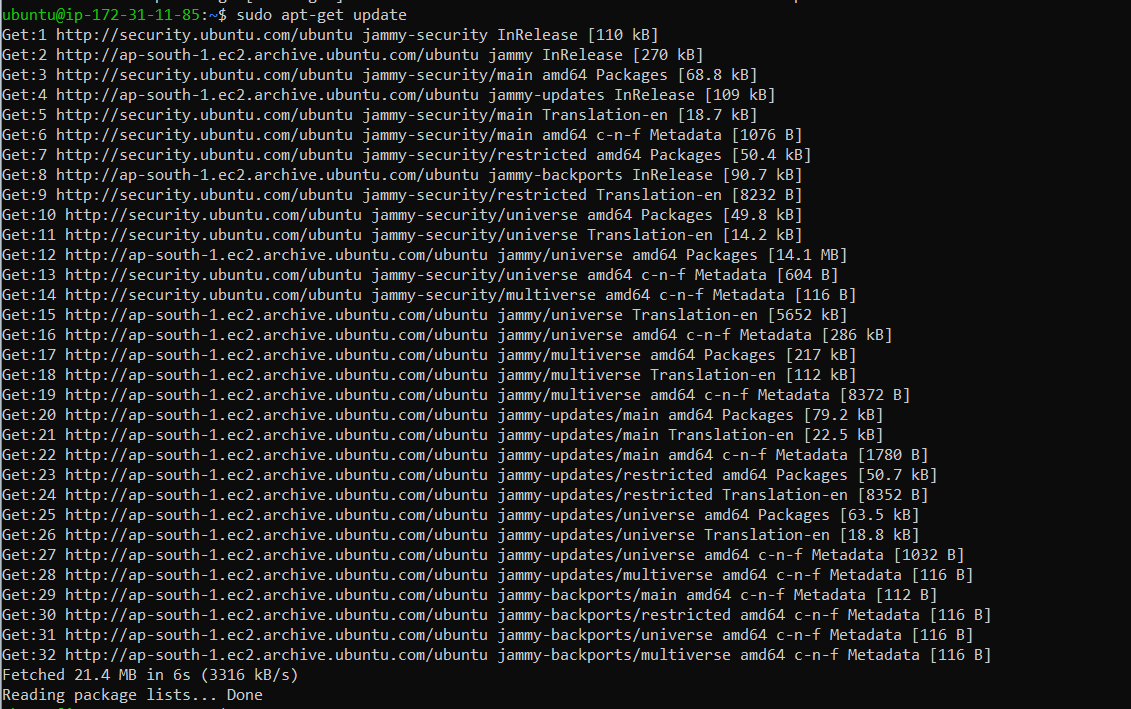
The plasma application can be run on EC2 instance.

we need to add key pair in order to maintain a secure connection between Local computer and the cloud server

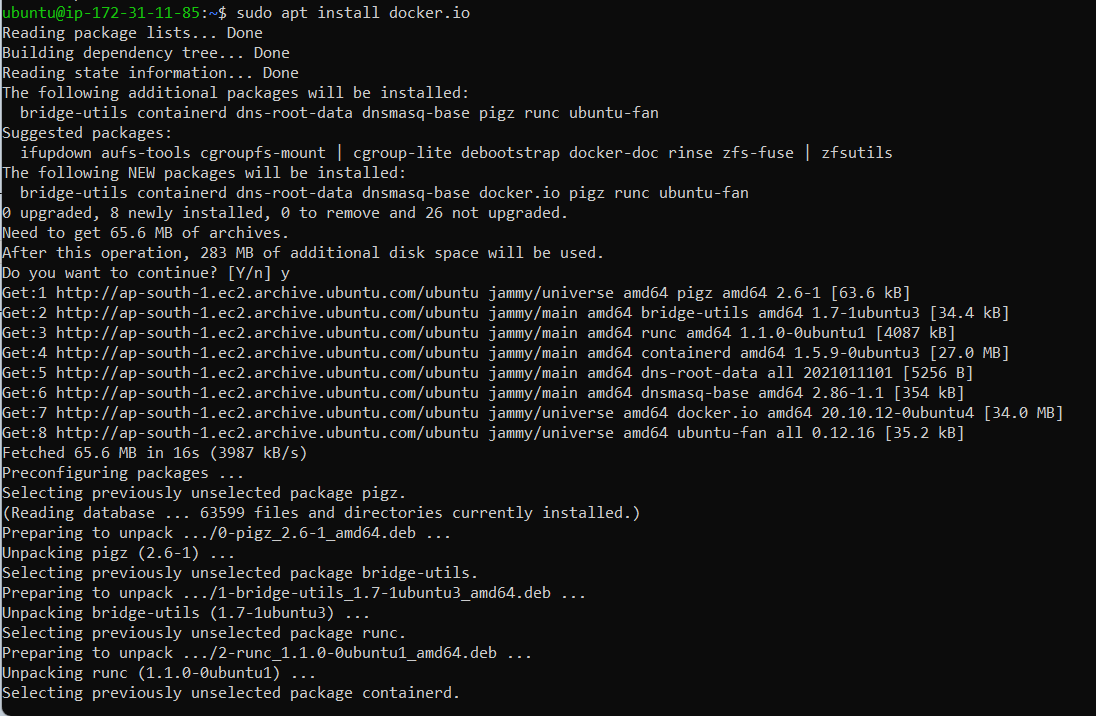
Once after creating key pair, create EC2 instance by selecting ubuntu 18.04 version. After launching EC2 instance click on connect.

Use the following command:-

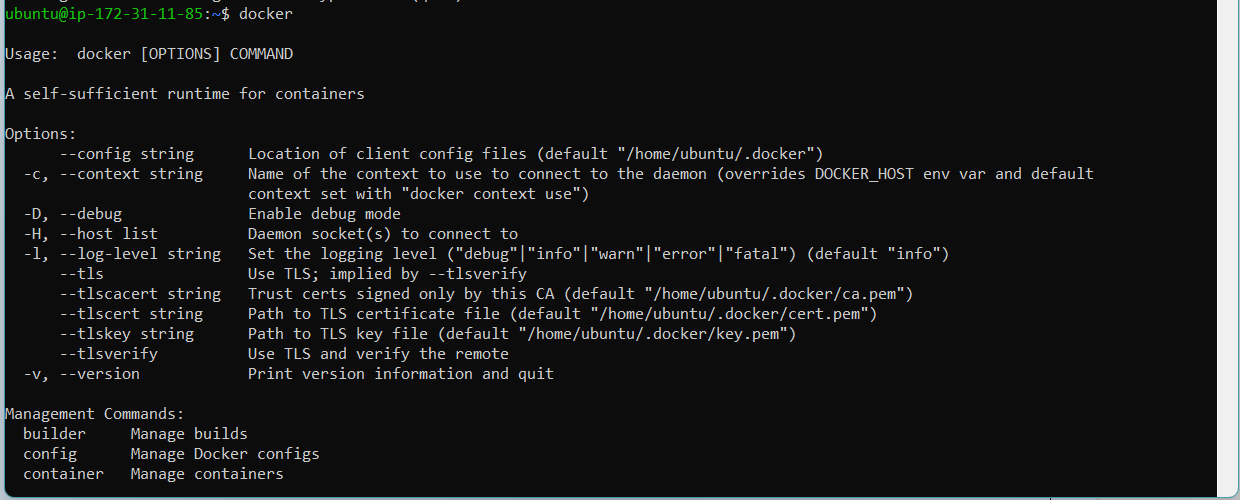
$ sudo apt-get update



$ sudo apt install docker.io //installing the docker in Ubuntu

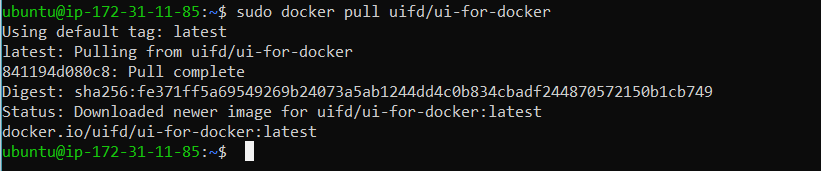


$ docker

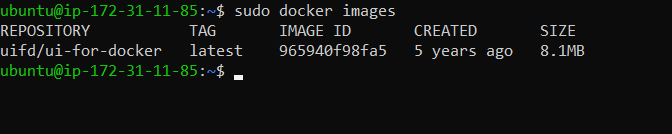


Now we are pulling the image from the docker hub to the Ubuntu machine

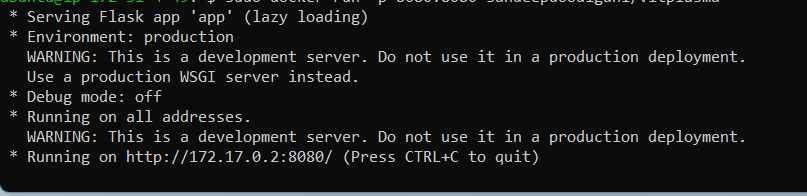
$ sudo docker pull uifd/ui-for-docker //Pulling image from the docker hub



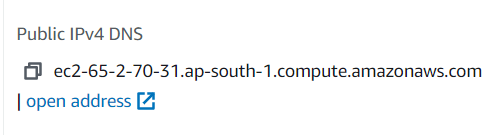
$ sudo docker images



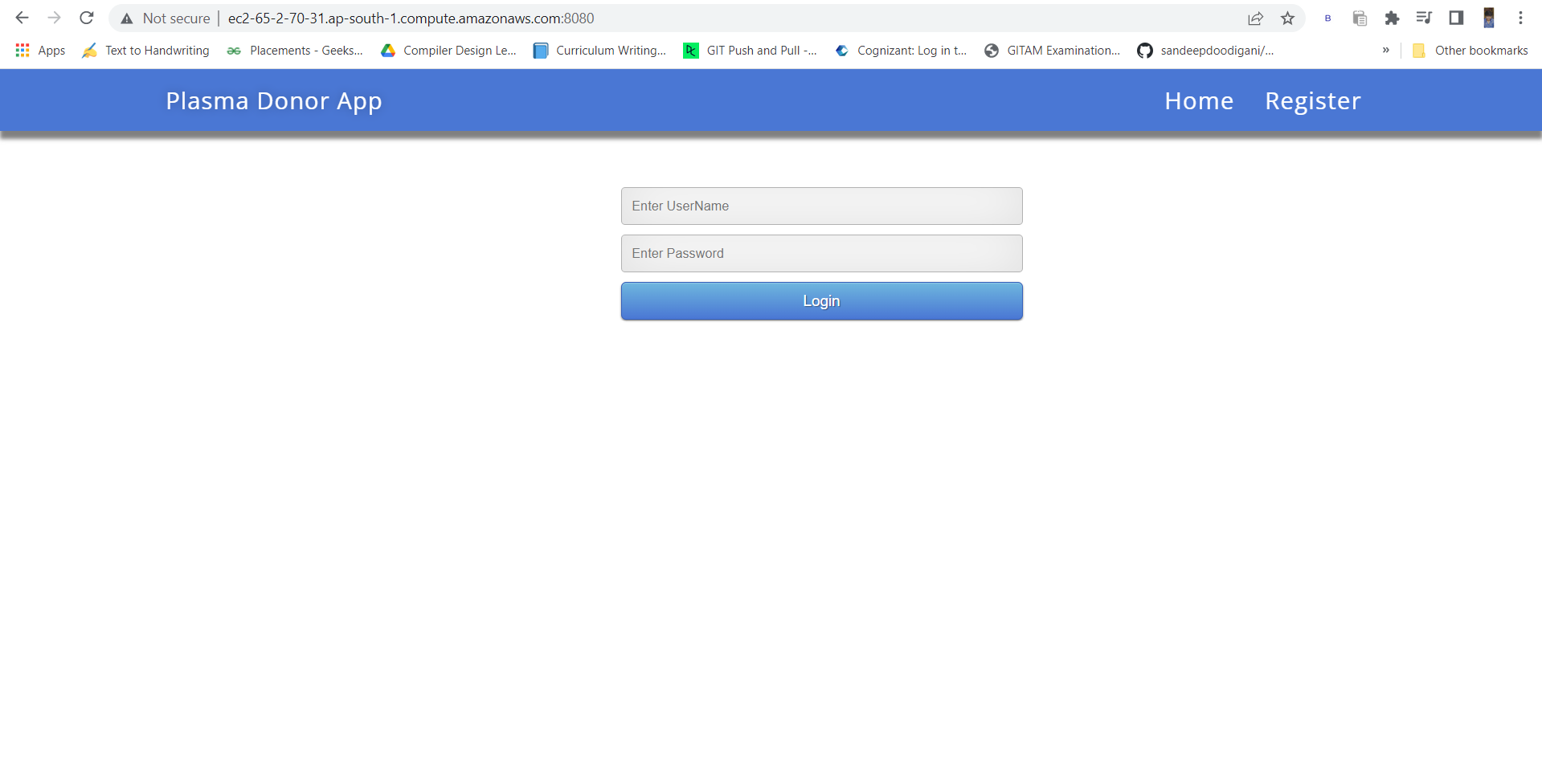
$ sudo docker run -p 8080:8080 sandeepdoodigani/vitplasma

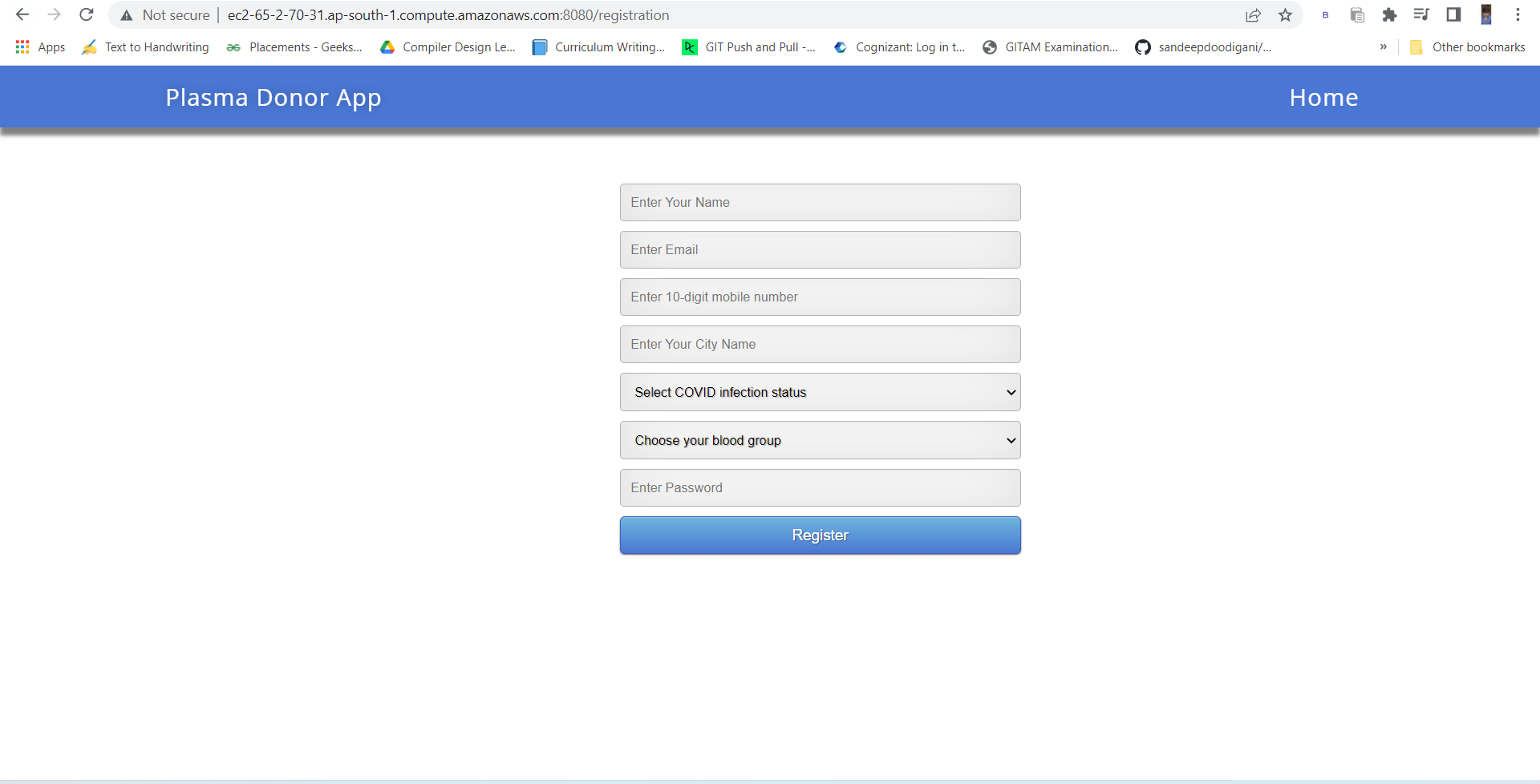


Copy the public Ipv4 address and add port 8080 at the last

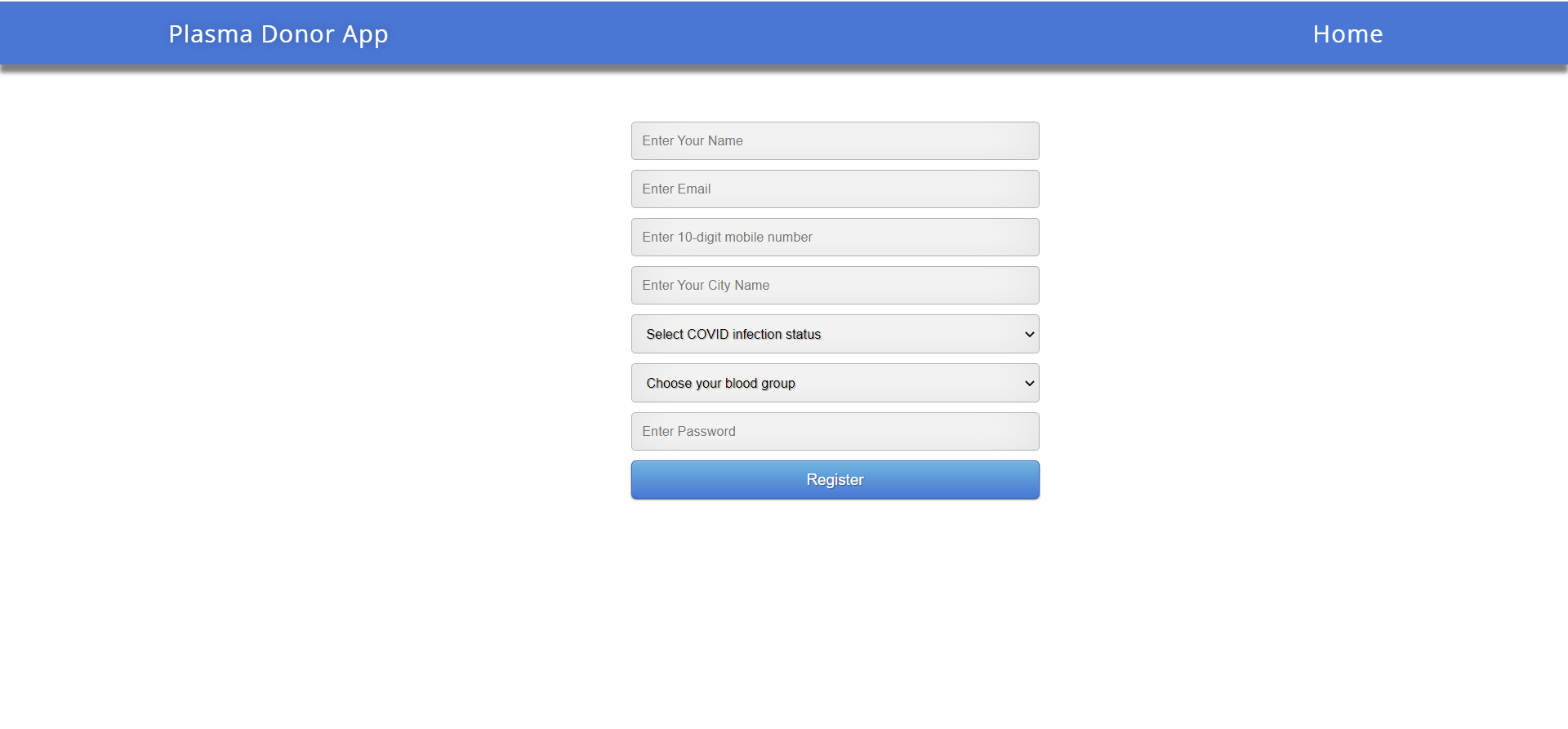


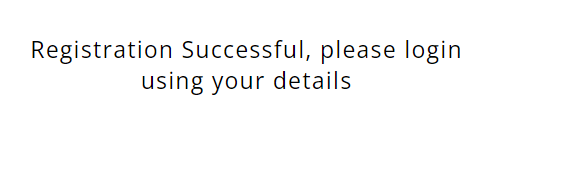
http://ec2-65-2-70-31.ap-south-1.compute.amazonaws.com:8080/





We are going to register the account



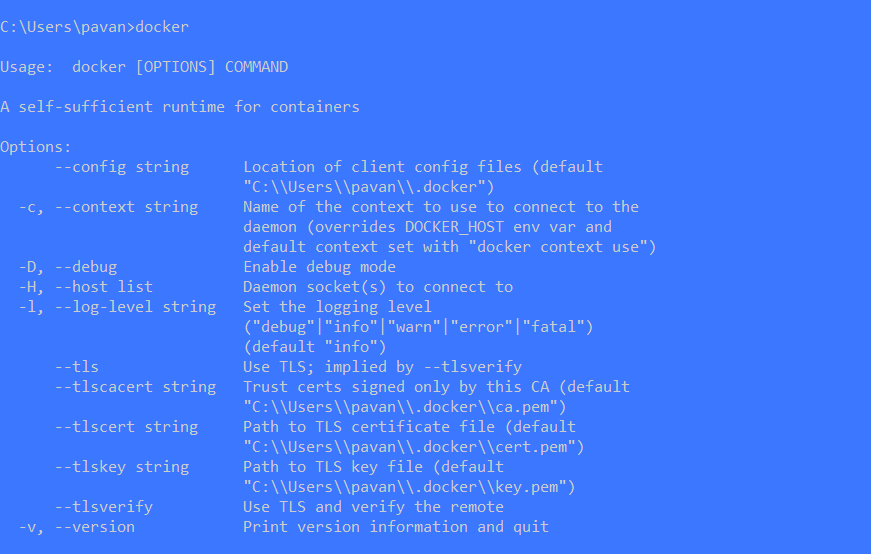


Now we are going to login to the app

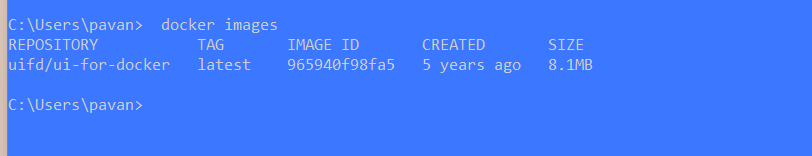




Docker

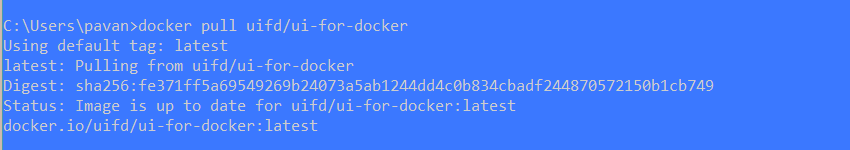


docker images // shows all the images present in the docker

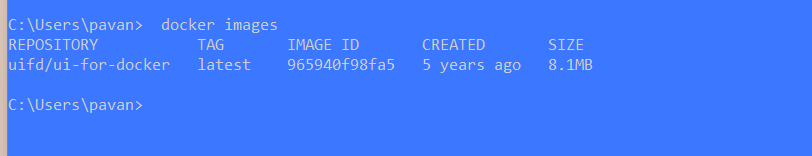


docker pull uifd/ui-for-docker //Command to pull images from the Docker hub

docker pull (image\_name)



docker images //list of images in the docker engine

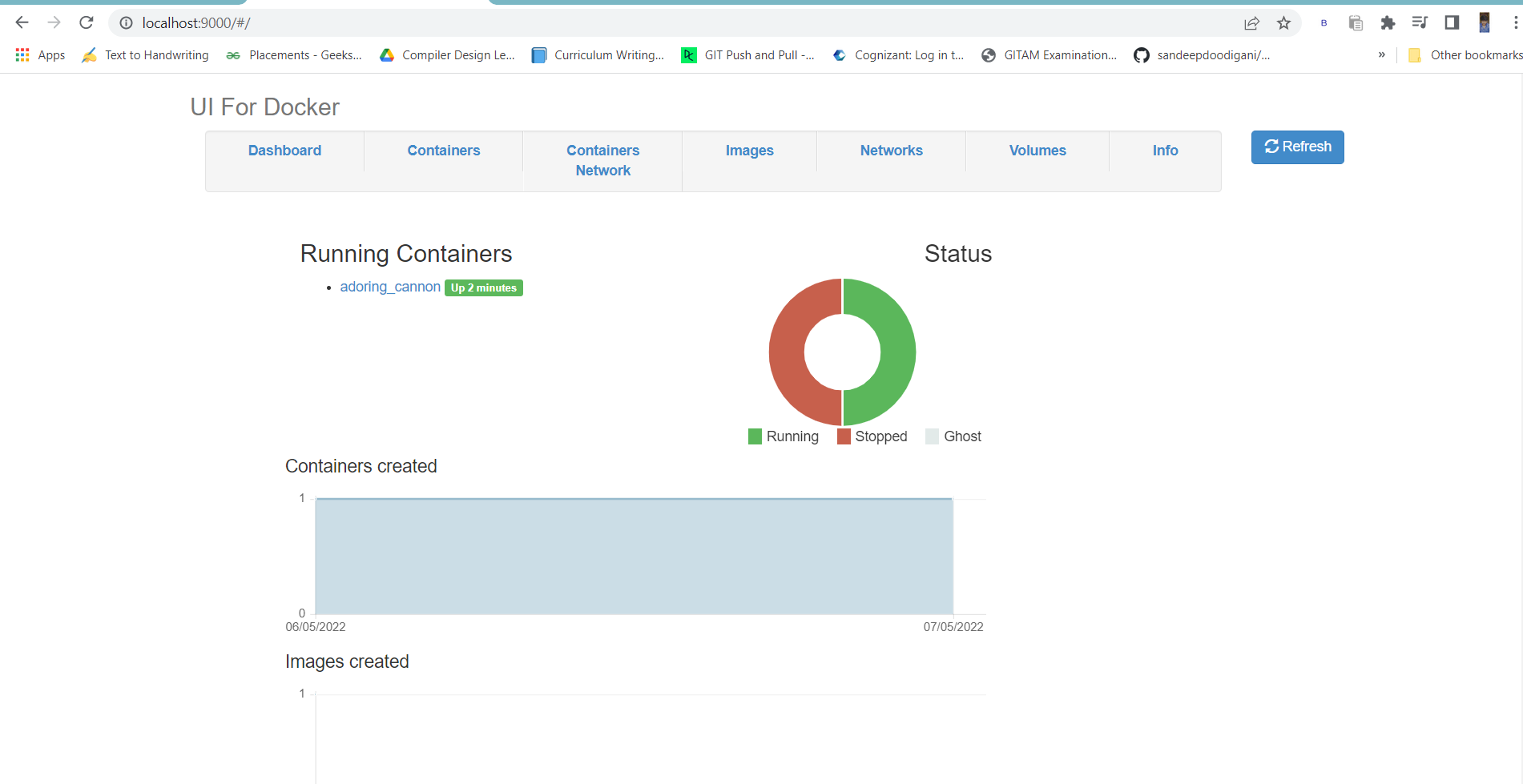


docker run -d -p 9000:9000 --privileged -v /var/run/docker.sock:/var/run/docker.sock uifd/ui-for-docker

docker ps //docker process state, can see what applications

are running in the docker engine

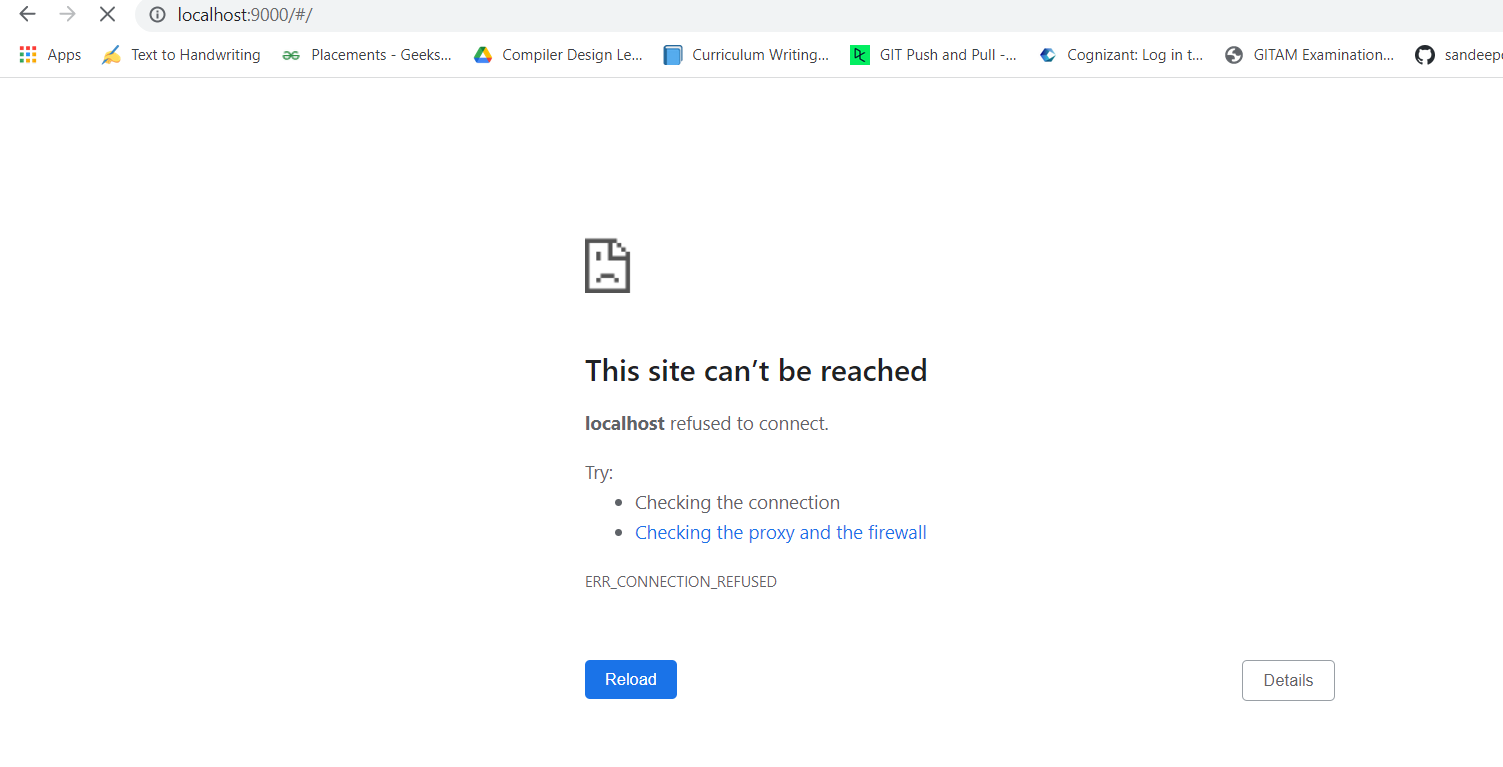
docker run -p 9000:9000 imagename //running a docker image in local host 9000

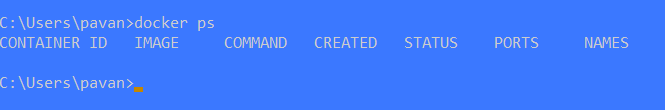


Once the application is running, it is called as a container

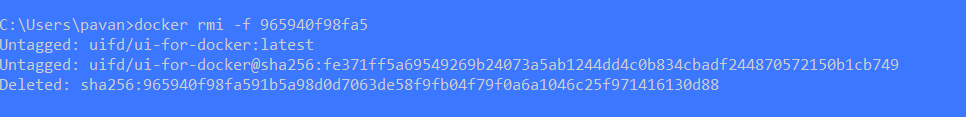
before that it is called as an image

docker stop 660488b34d69 // Stopping the container(conatiner id is used to stop)





docker rmi -f 965940f98fa5 //Deleting the image (rmi is remove image,-f is force)





Now process to convert a application into Docker Image and uplod to Docker HUB

1. We need to navigate the folder location

cd ..

cd ..

c:

cd DevOps

cd Plasma-Docker-main



FROM python:3.6.5-alpine //installing python in the OS Alpine

WORKDIR /app //Creates an file in Alphine and copy all the files

ADD . /app

COPY requirements.txt /app

RUN pip install -r requirements.txt //Libraries required for installing flask

CMD ["python","app.py"]

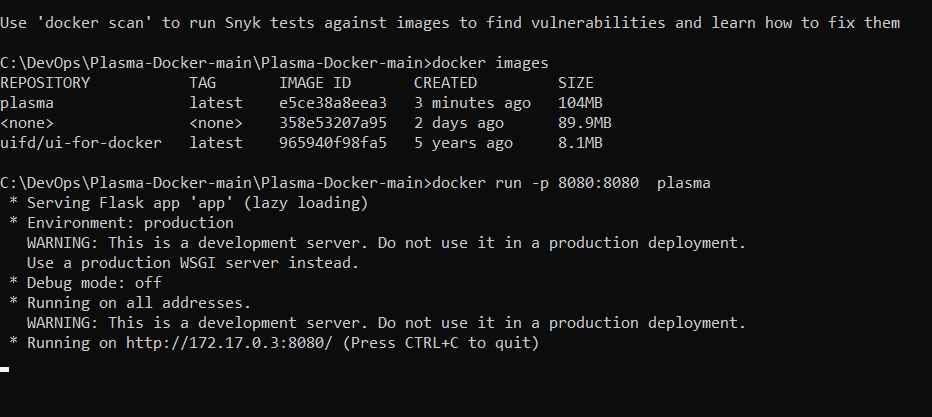
\*\*\*Building the image\*\*\*

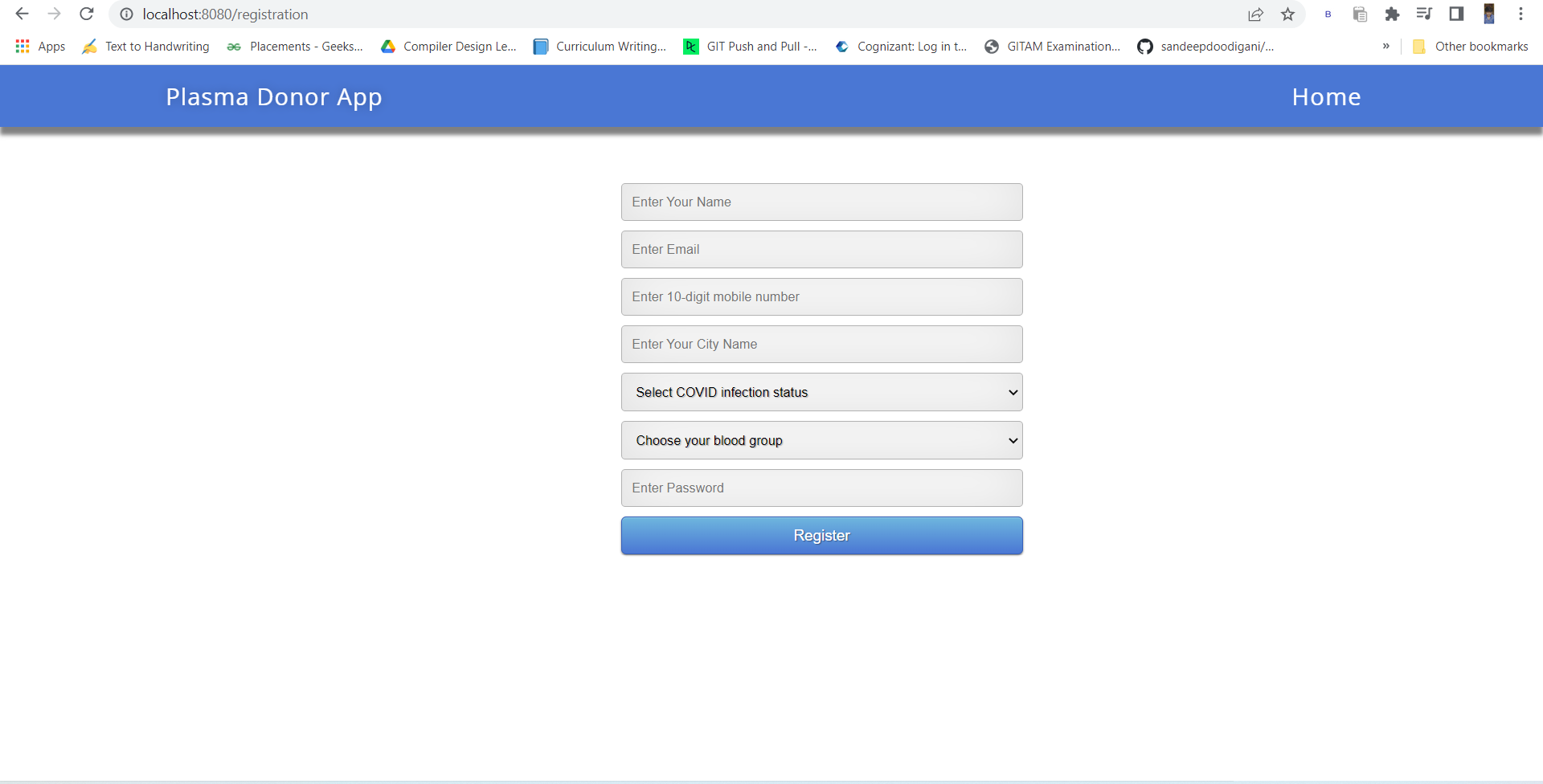
docker build . -f Dockerfile.txt //Navigating to the file

docker build -t plasma .

docker images

docker run -p 8080:8080 plasma //running the file





\*\*\*Making the Application live\*\*\*

\*\*\*Now we are pushing the plasma App to Docker hub\*\*\*

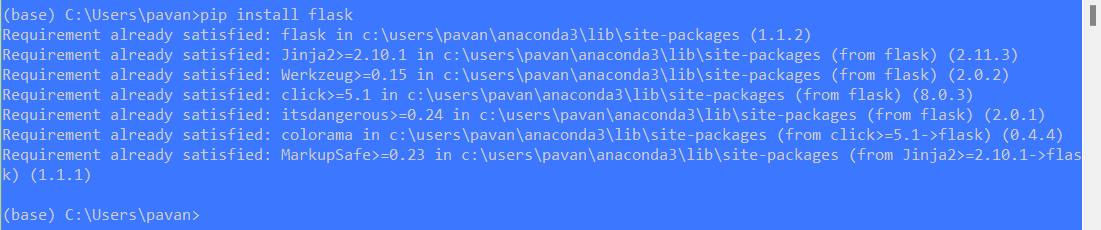
1. Create a new repository in docker hub
2. Open command Prompt
3. docker login

Now the images get uploded in the docker Hub

We are going to design a web application using Flask Framework and convert the image into Docker image

1. **Install Flask in Anaconda Prompt**

pip install flask



1. **Run the following code in Syper prompt**

from flask import Flask

app=Flask(\_\_name\_\_)

@app.route("/")

def helloworld():

return "Helloworld"

@app.route("/home")

def home():

return "Home page"

@app.route("/about")

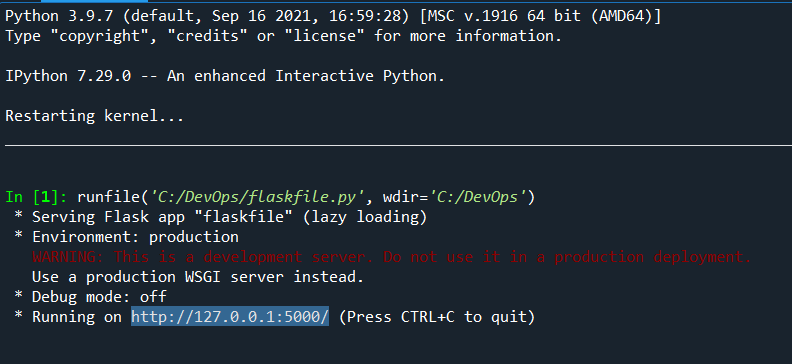
def aboutus():

return "About Us"

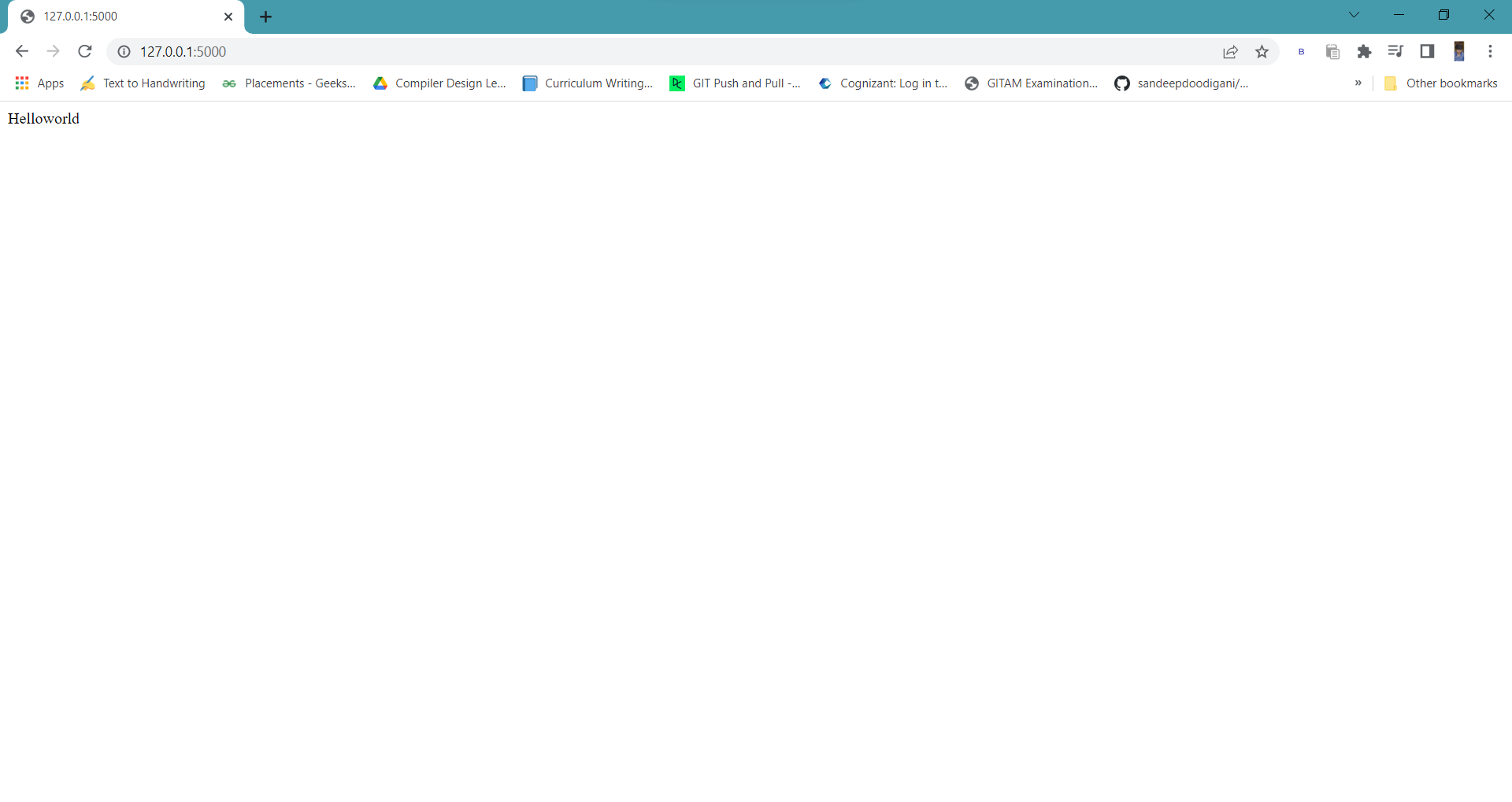
if \_\_name\_\_=="\_\_main\_\_":

app.run()

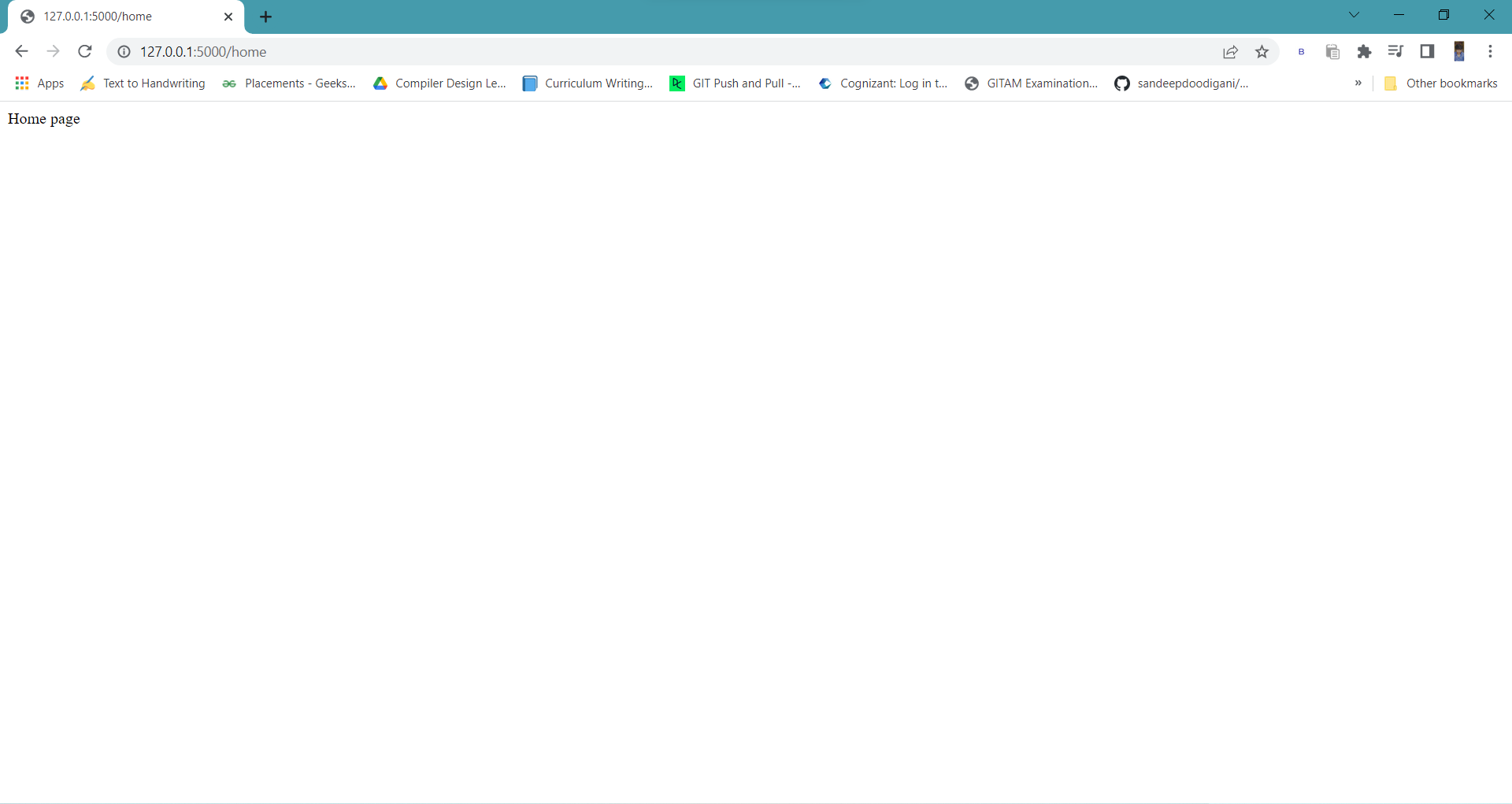
**Here we are creating the Routes**



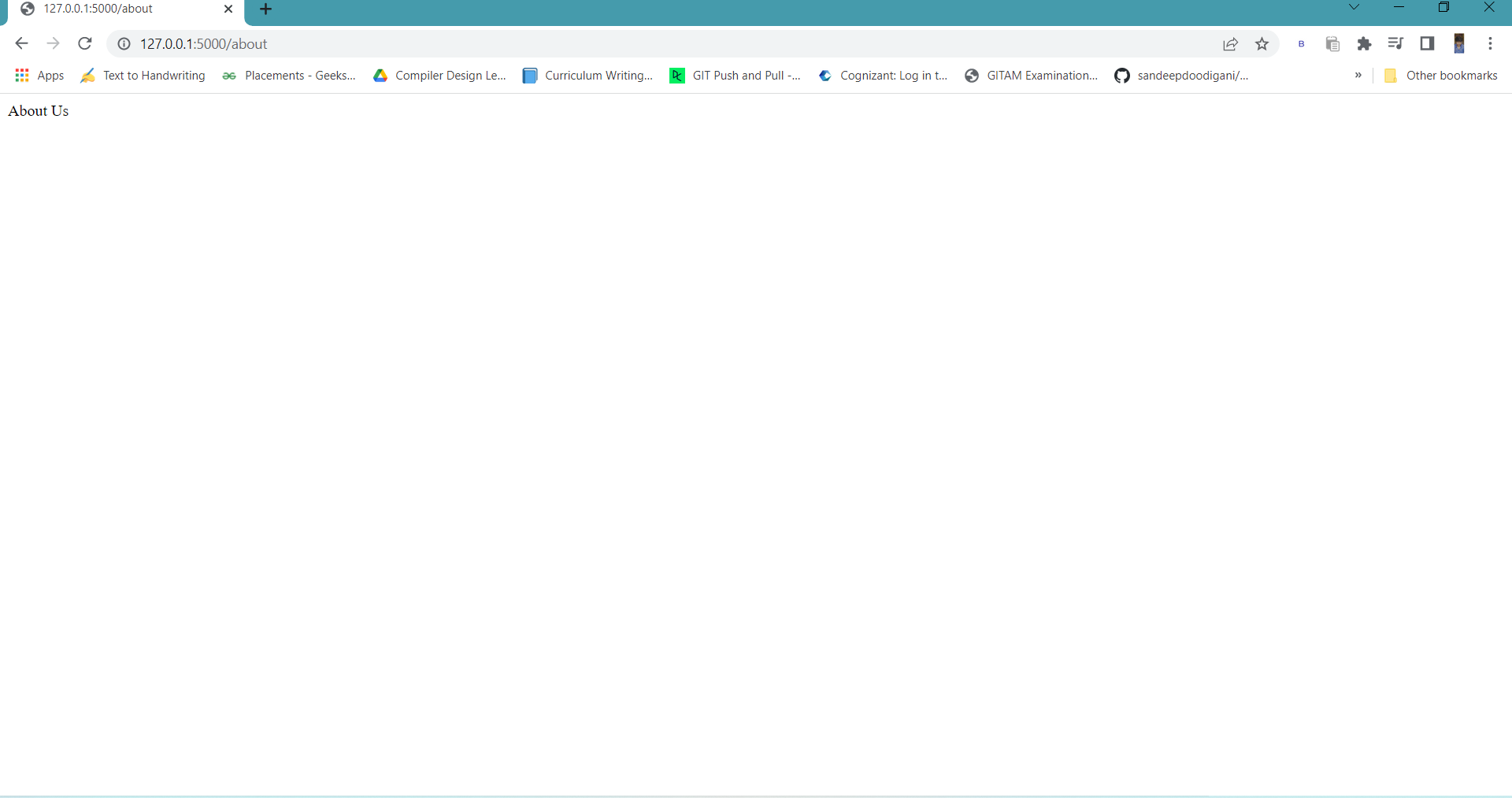
<http://127.0.0.1:5000/> or http://localhost:5000/



<http://127.0.0.1:5000/home>



<http://127.0.0.1:5000/about>



1. Now create a new lambda function

import boto3

dynamoDB = boto3.resource('dynamodb')

table = dynamoDB.Table('plasma')

def lambda\_handler(event,context):

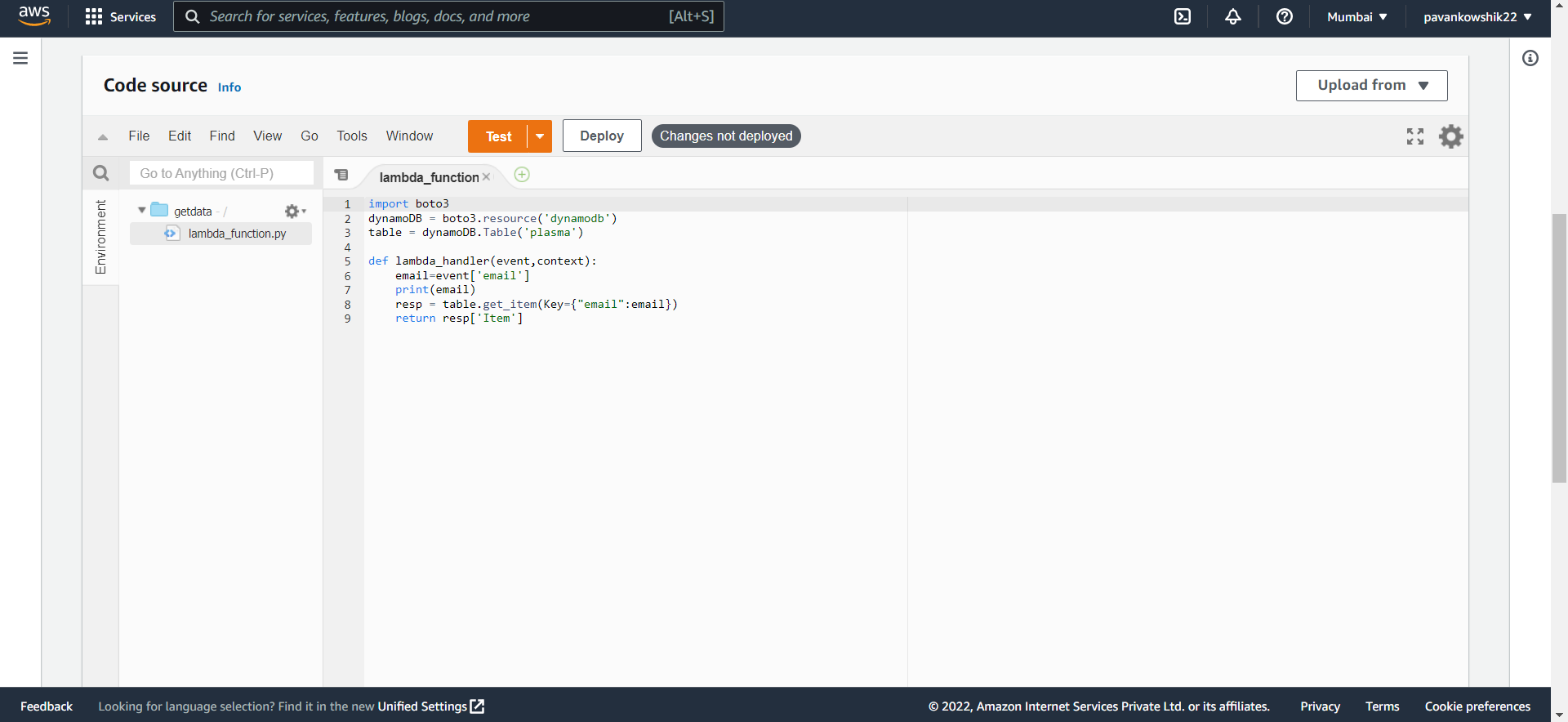
email=event['email']

print(email)

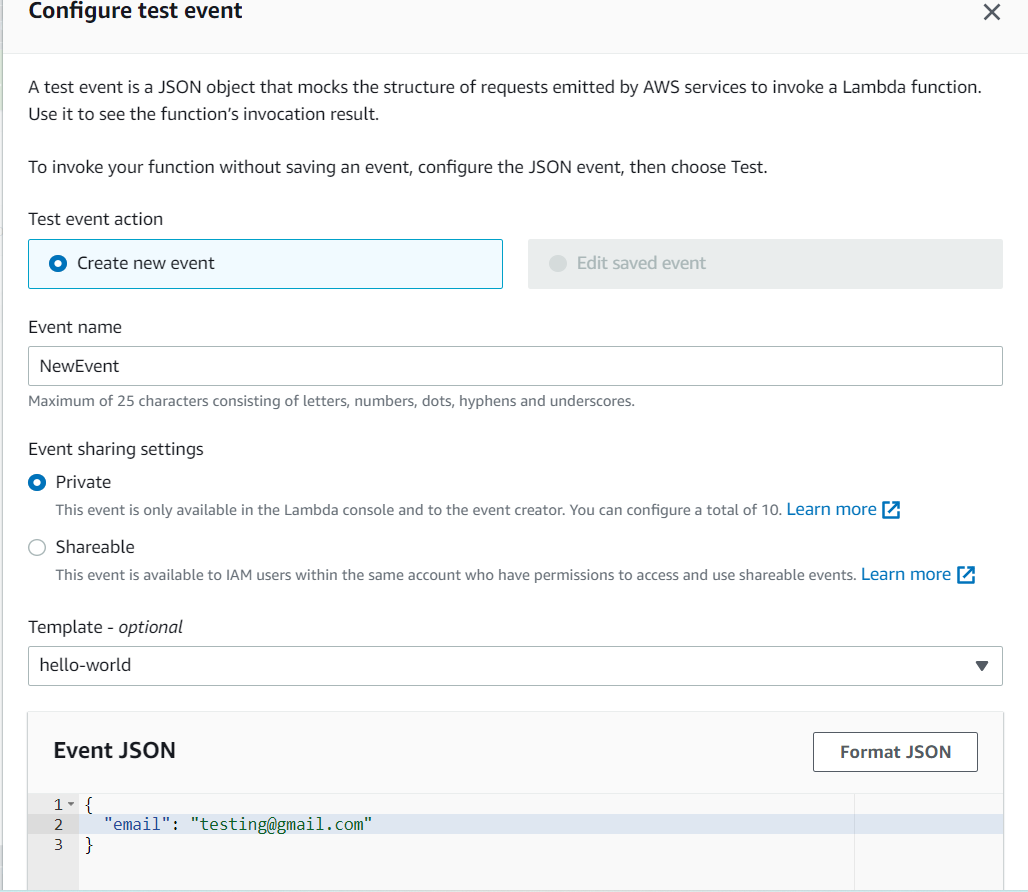
resp = table.get\_item(Key={"email":email})

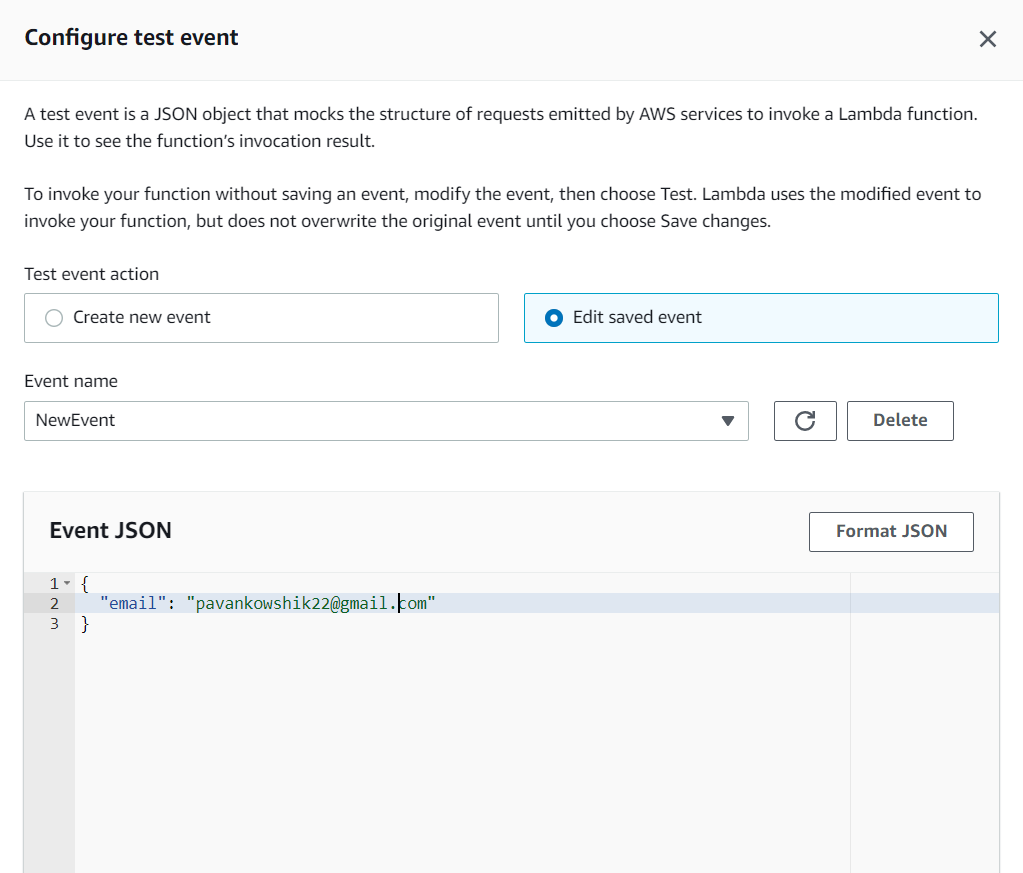
return resp['Item']





1. Now click on Configure and do the following changes





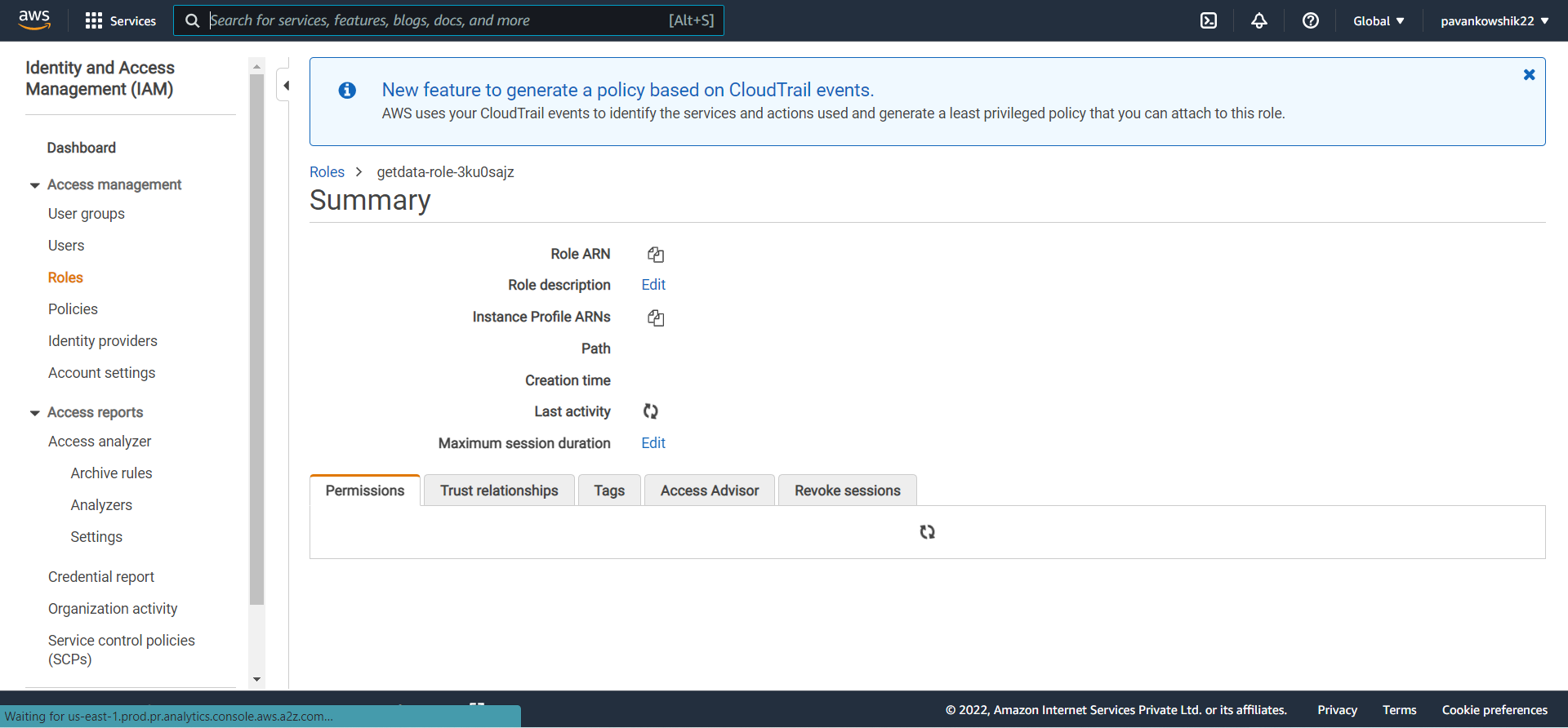
1. Now we need to give permission to connect to DynamoDb

**Execution role**

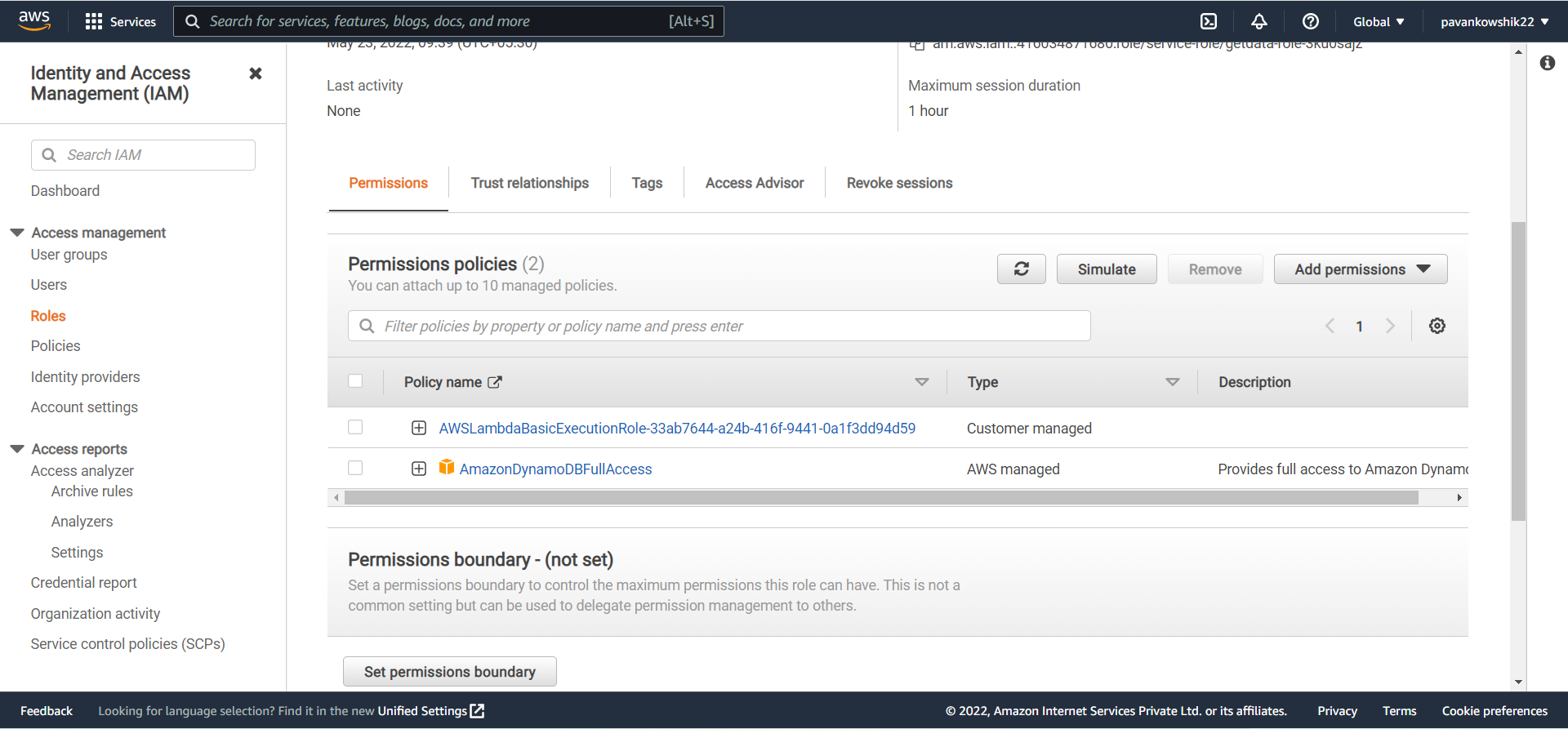
**Edit**

Role name

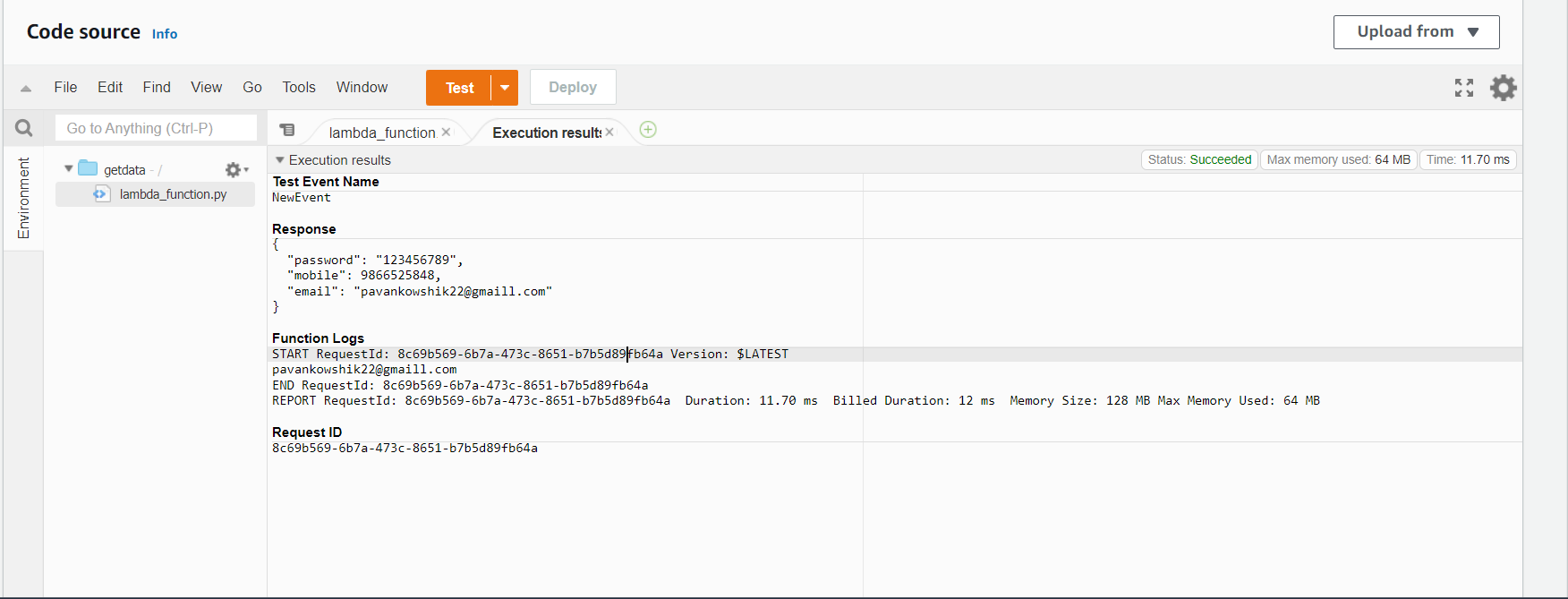
[getdata-role-3ku0sajz](https://ap-south-1.console.aws.amazon.com/iam/home#/roles/getdata-role-3ku0sajz?section=permissions)



1. Give access to DynamoDb full



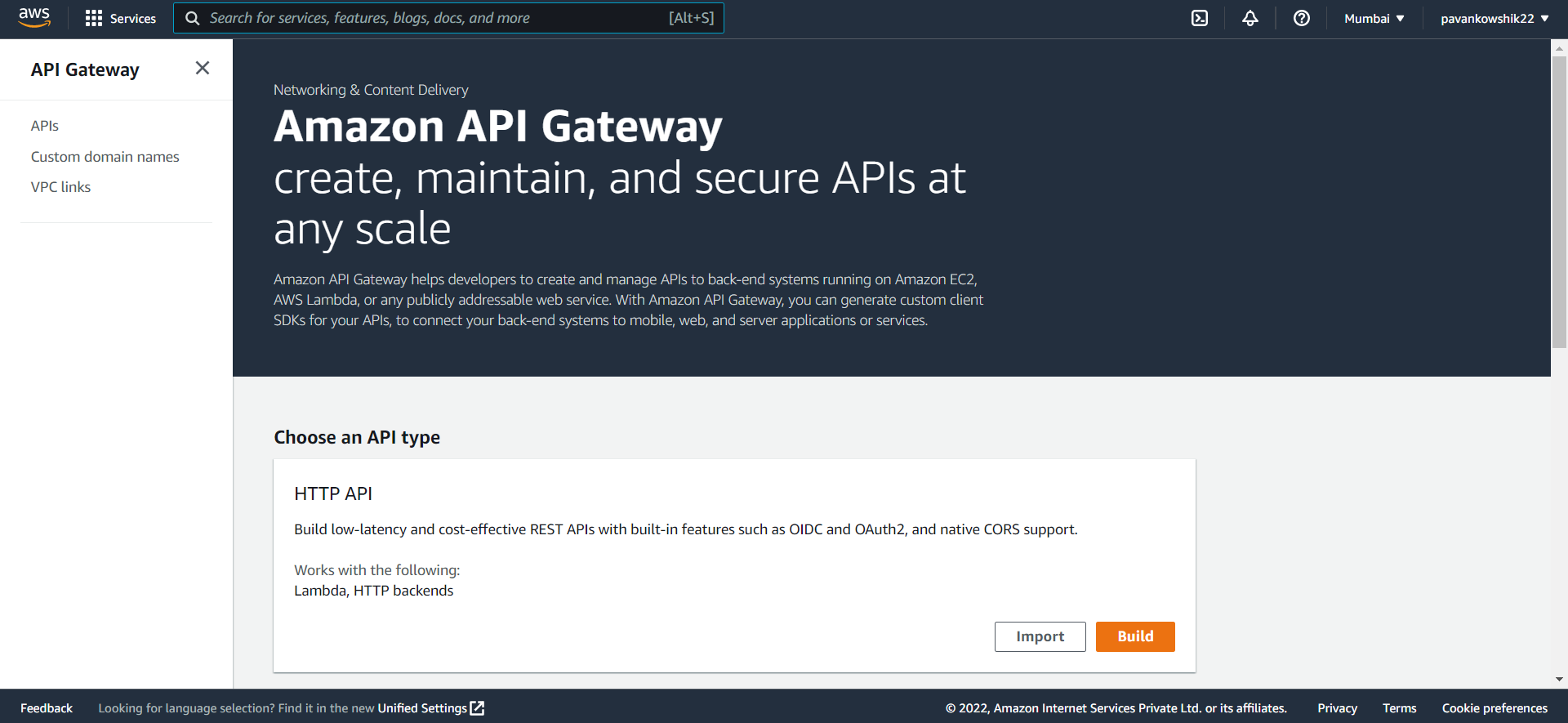
1. We get the details of the person when we type the email which is present in the table



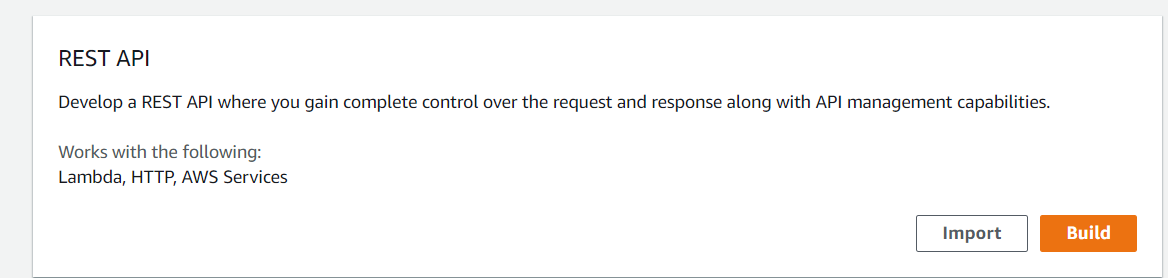
1. In the above process, we are fetching the details of the person from the data base and through primary key i.e . email.

Now we need to intergrate this service to a application

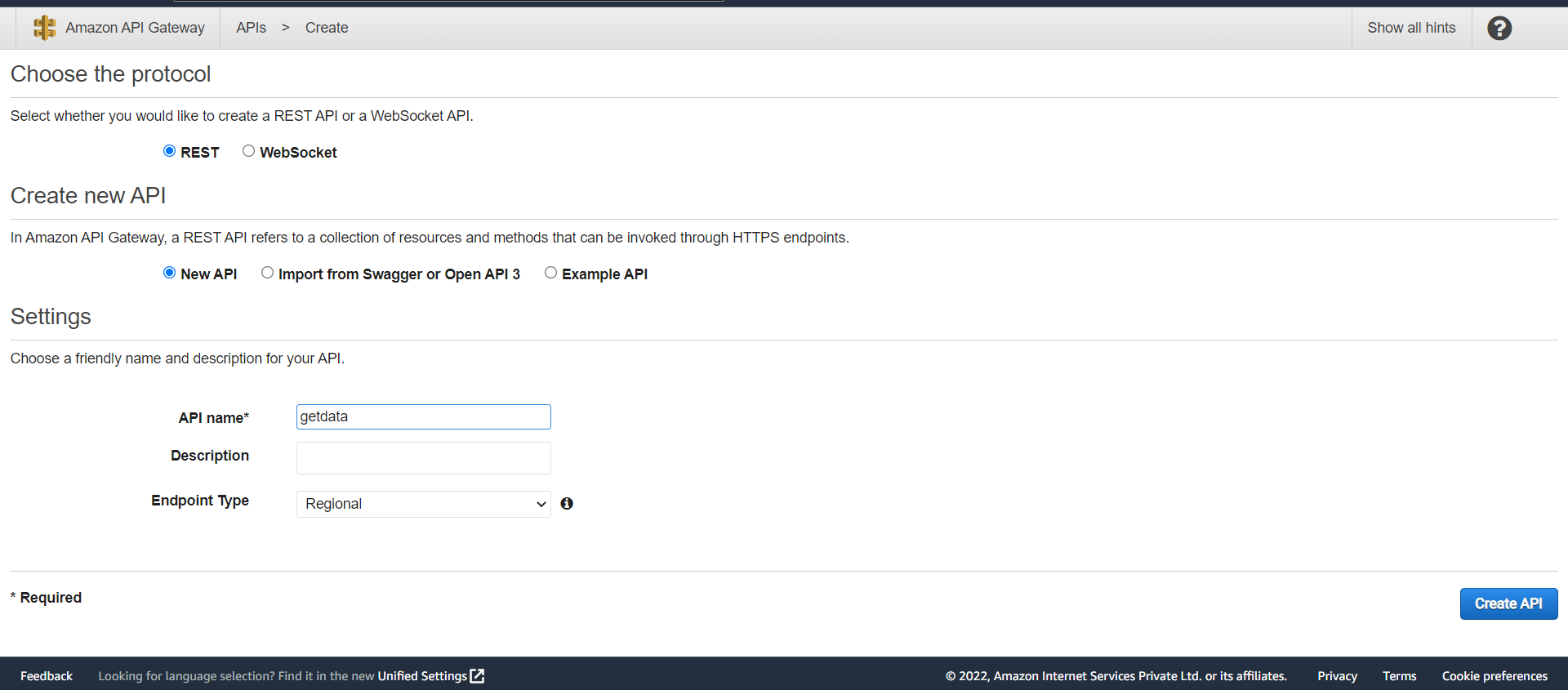
1. Open API GATEWAY



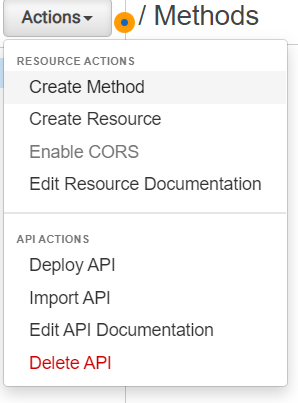
1. Click on rest API



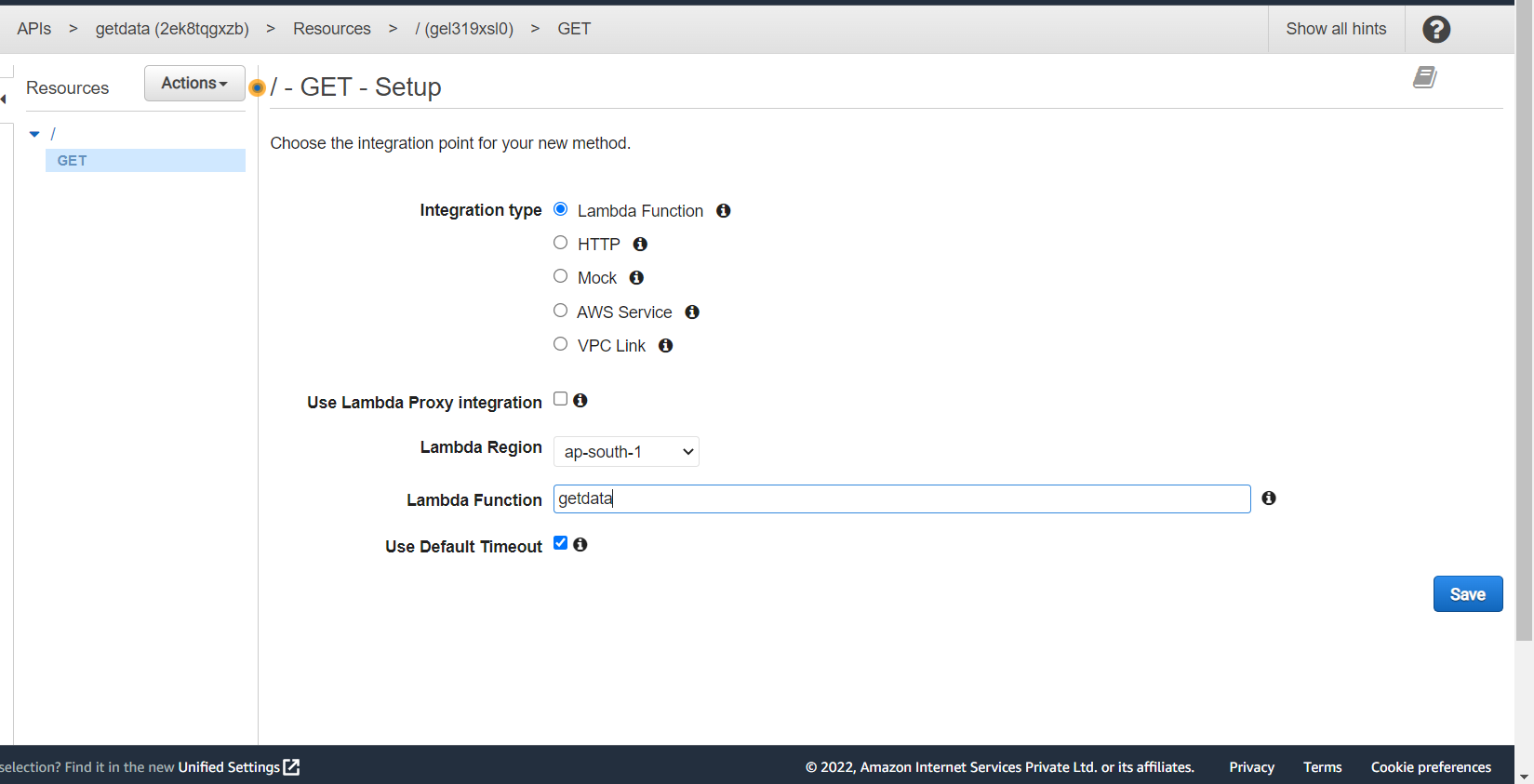
1. Complete the details and click on create



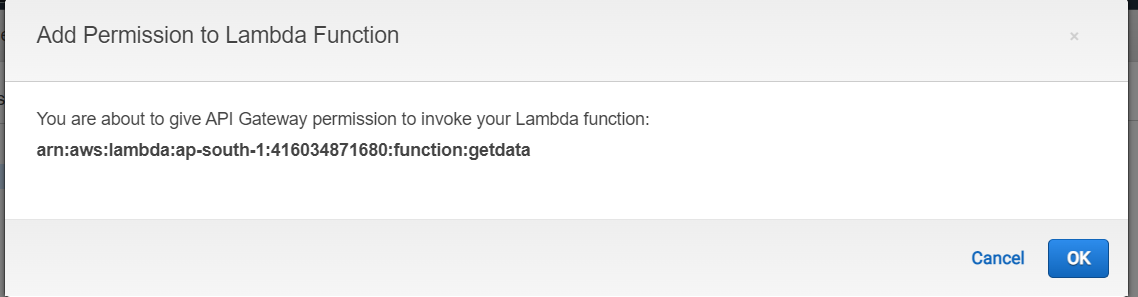
1. Now click on actions and click on create method and select the GET METHOD

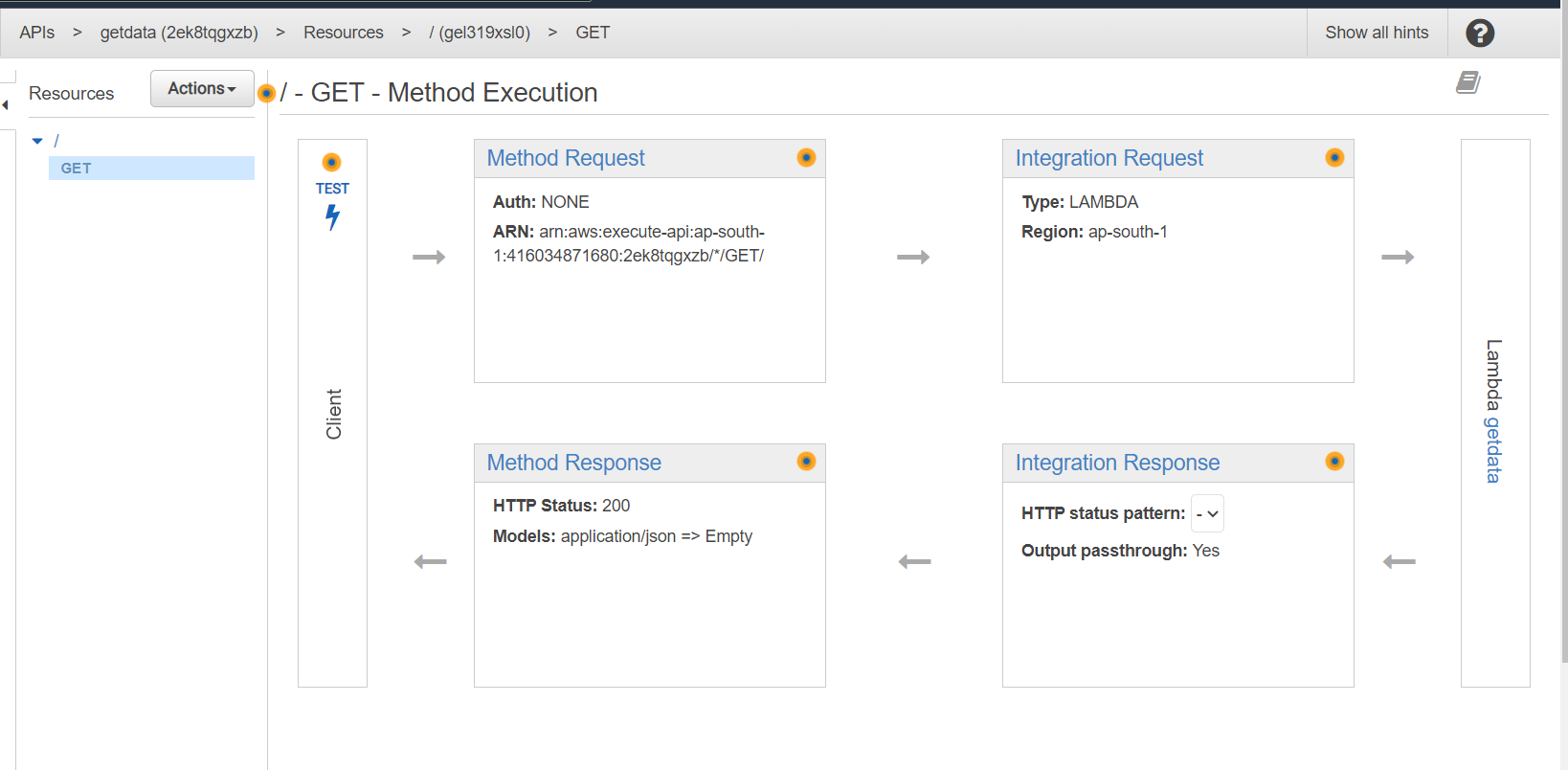


1. Now Follow the following fields

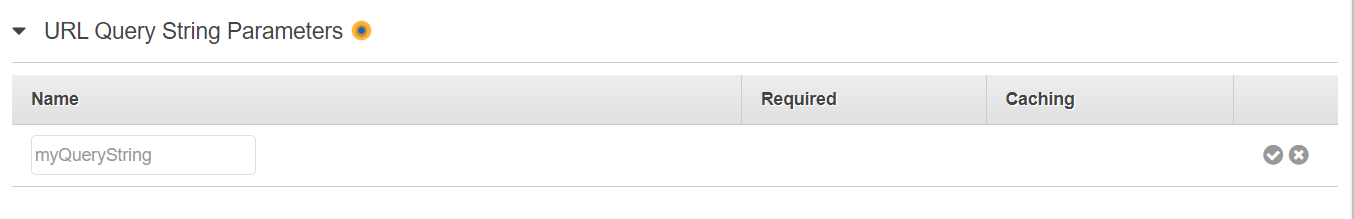
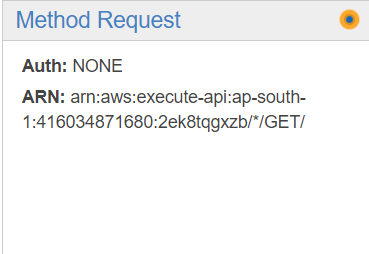


1. The output is as follows



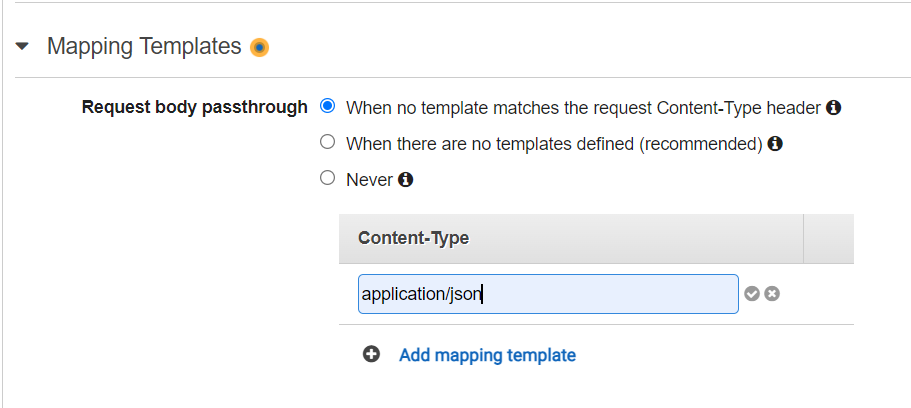
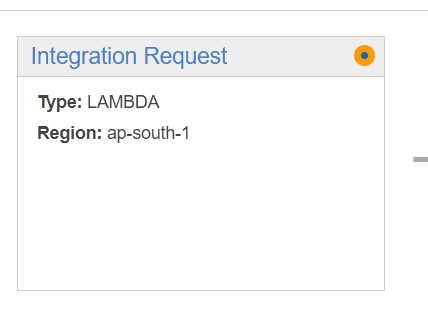


1. Click on Method Request





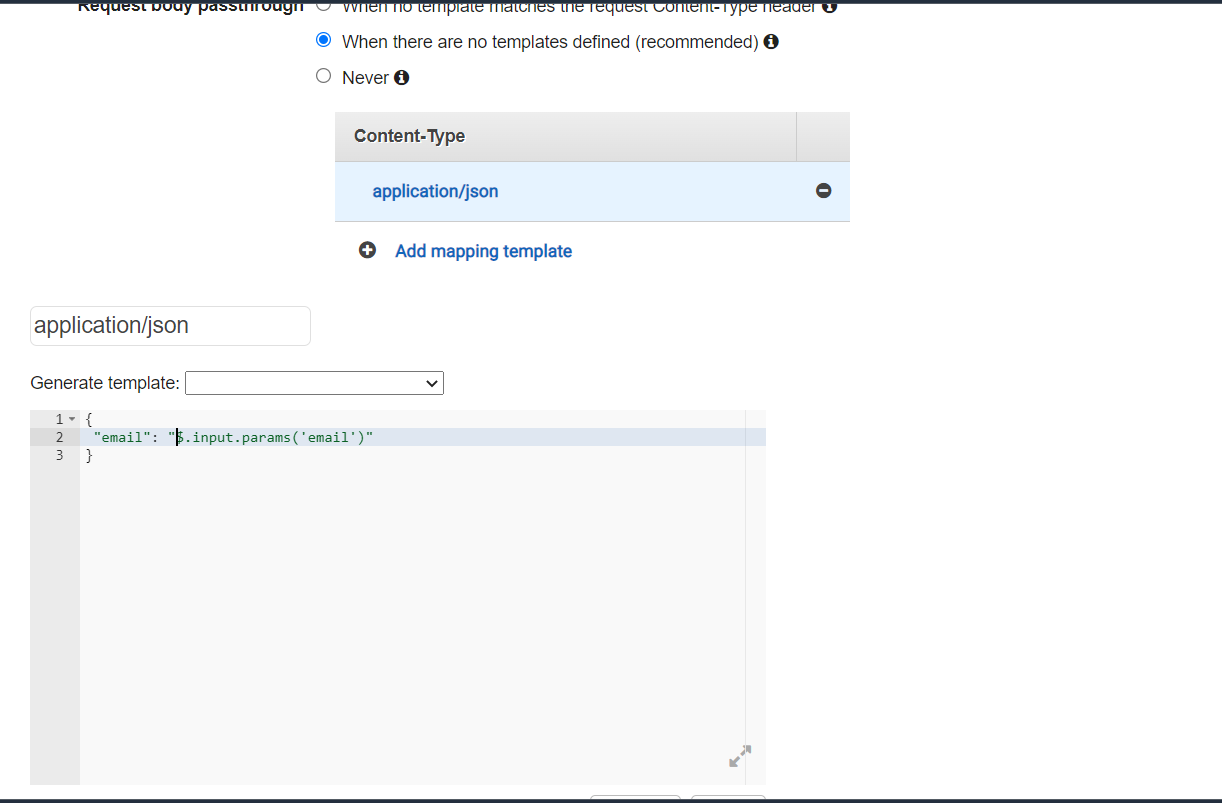
1. Now click on Integration Request



{

"email":"$input.params('email')"

}

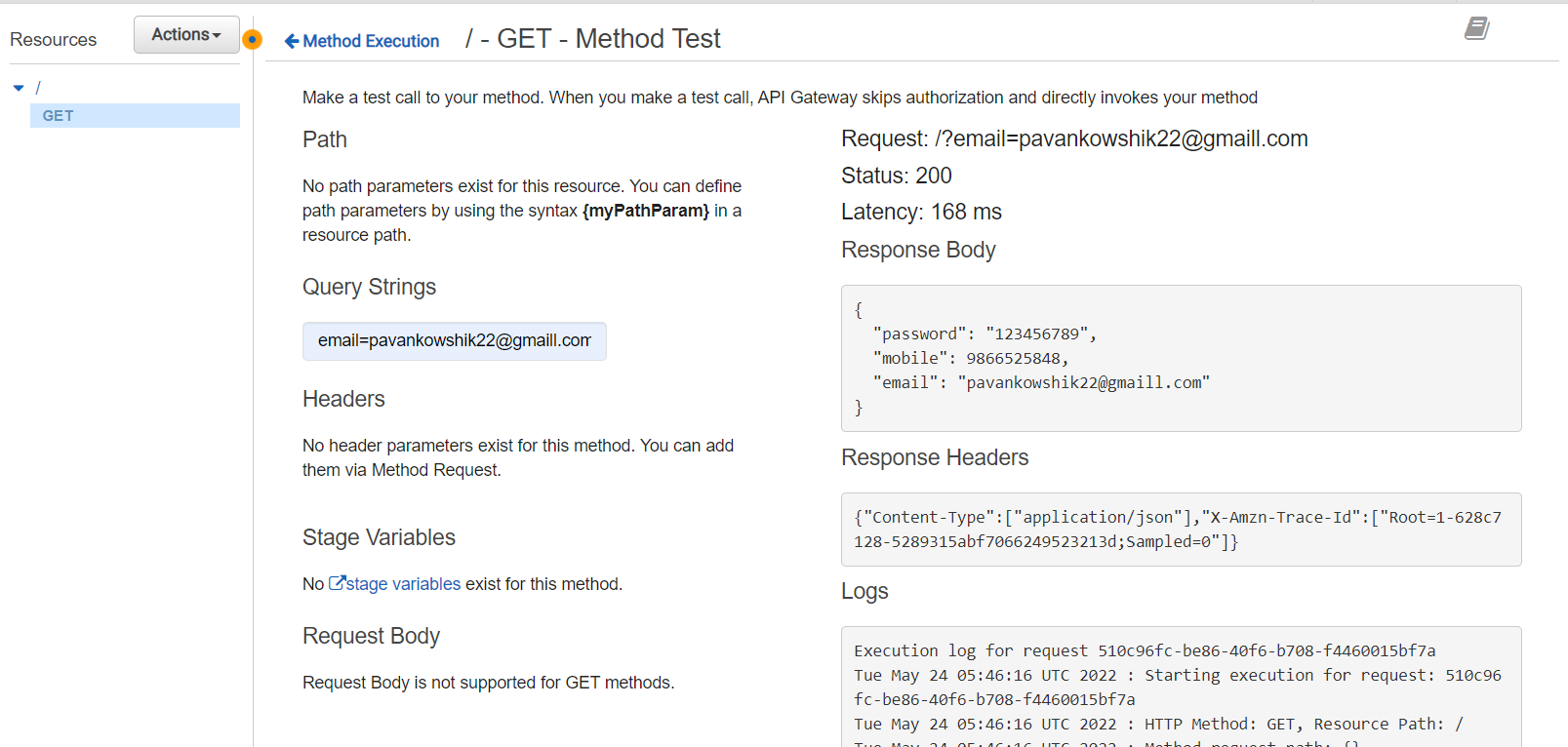


1. Now click on Test

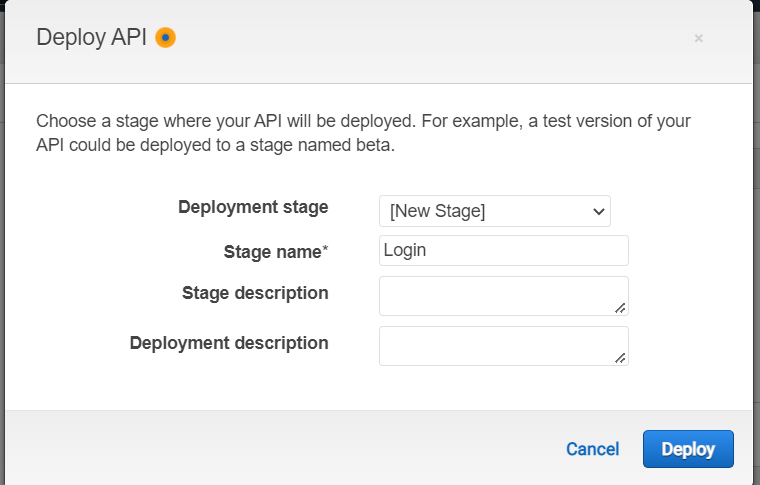
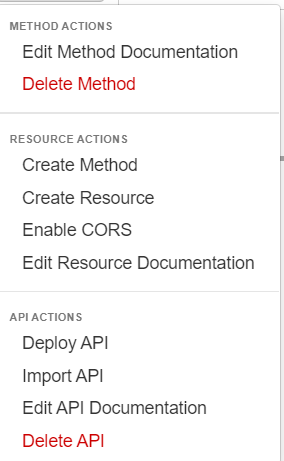


[email=pavankowshik22@gmaill.com](mailto:email=pavankowshik22@gmaill.com)

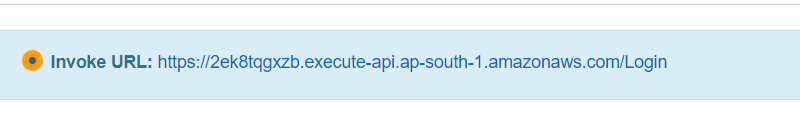
1. We get the details of the candidate



1. Now click on actions and deploy api

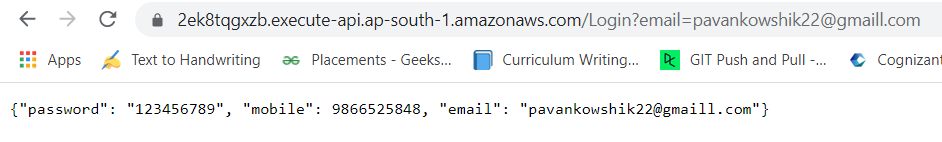


1. Click on the link



<https://2ek8tqgxzb.execute-api.ap-south-1.amazonaws.com/Login?email=pavankowshik22@gmaill.com>

Throgh Api we are going to fetch the data



Now We are goin to integrate the service to Flask app

def check(email):

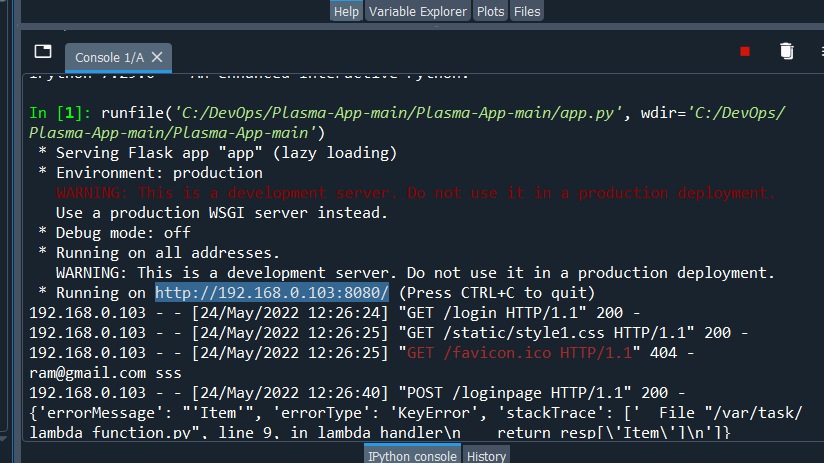
url = "https://2ek8tqgxzb.execute-api.ap-south-1.amazonaws.com/Login?email="+email

status = requests.request("GET",url)

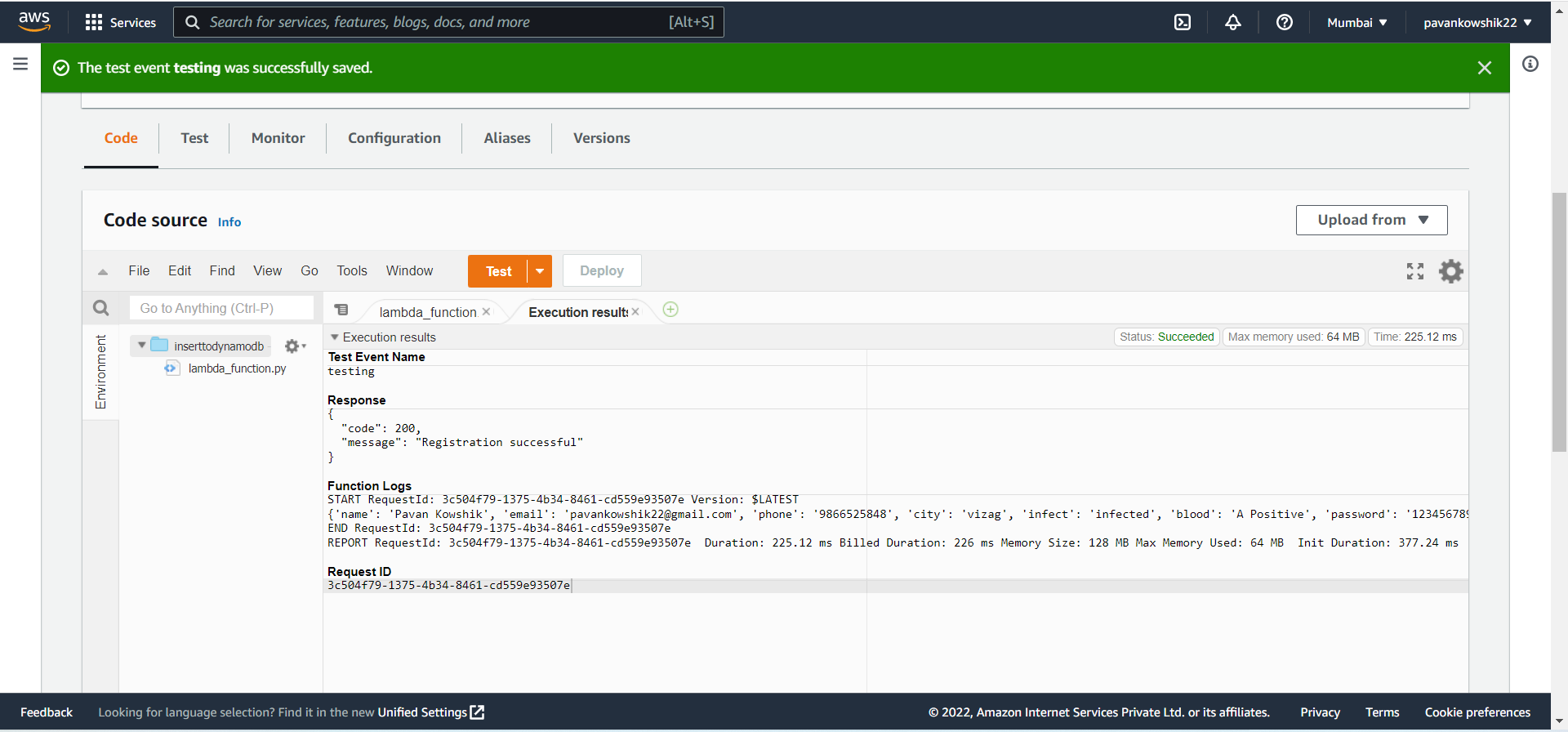
print(status.json())

return status.json()

Run the code with user names provided in the databaseof DynamoDb



1. Now add the data in Dynamodb



The following data:

{

"name":"Ram",

"email": "ram123 @gmail.com",

"phone":"987654321",

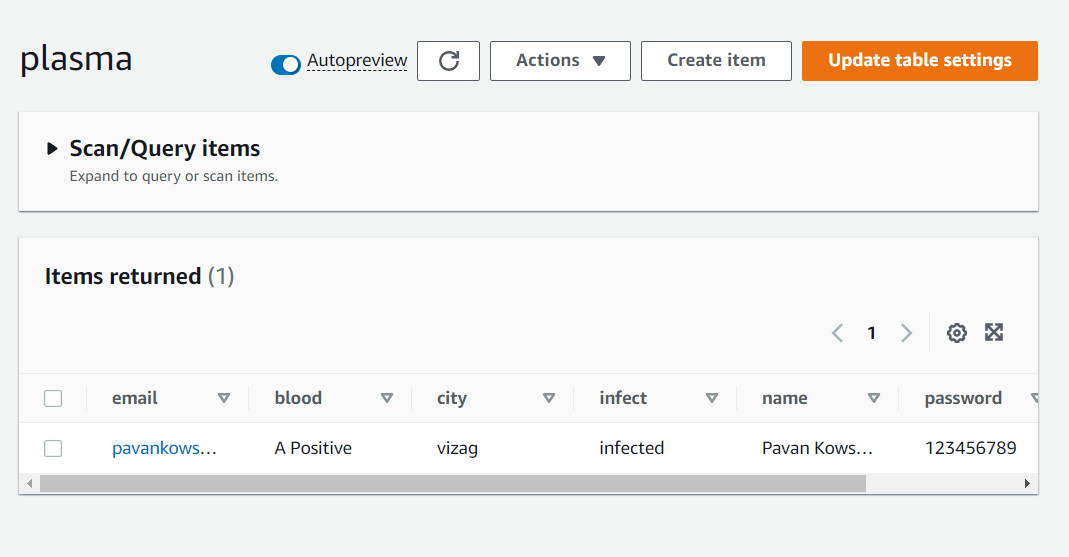
"city":"hyd",

"infect":"infected",

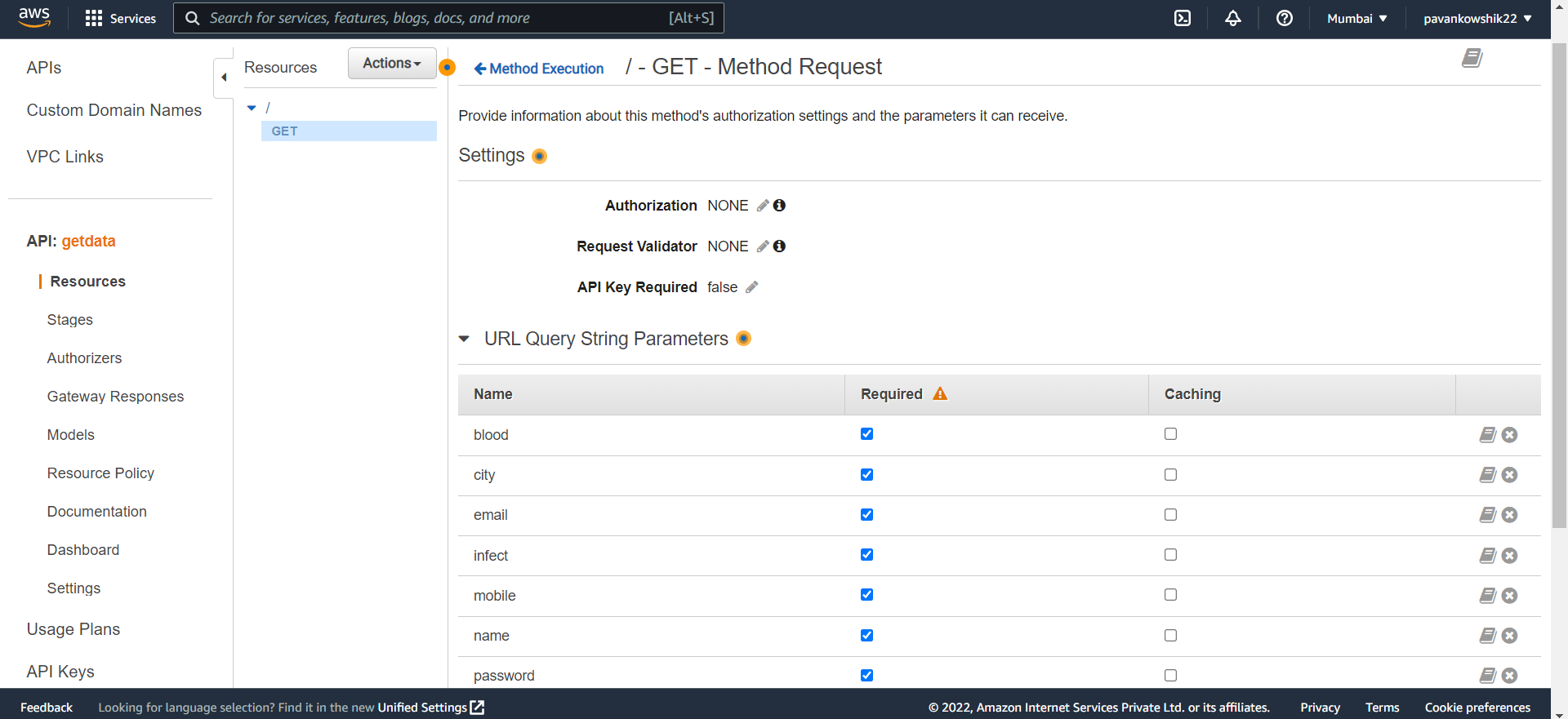
"blood":"A Positive",

"password":"123456789"

}



Now we need to modify the following:-



Apply the following changes:-

{

"email":"$input.params('email')",

"phone":"$input.params('phone')",

"city":"$input.params('city')",

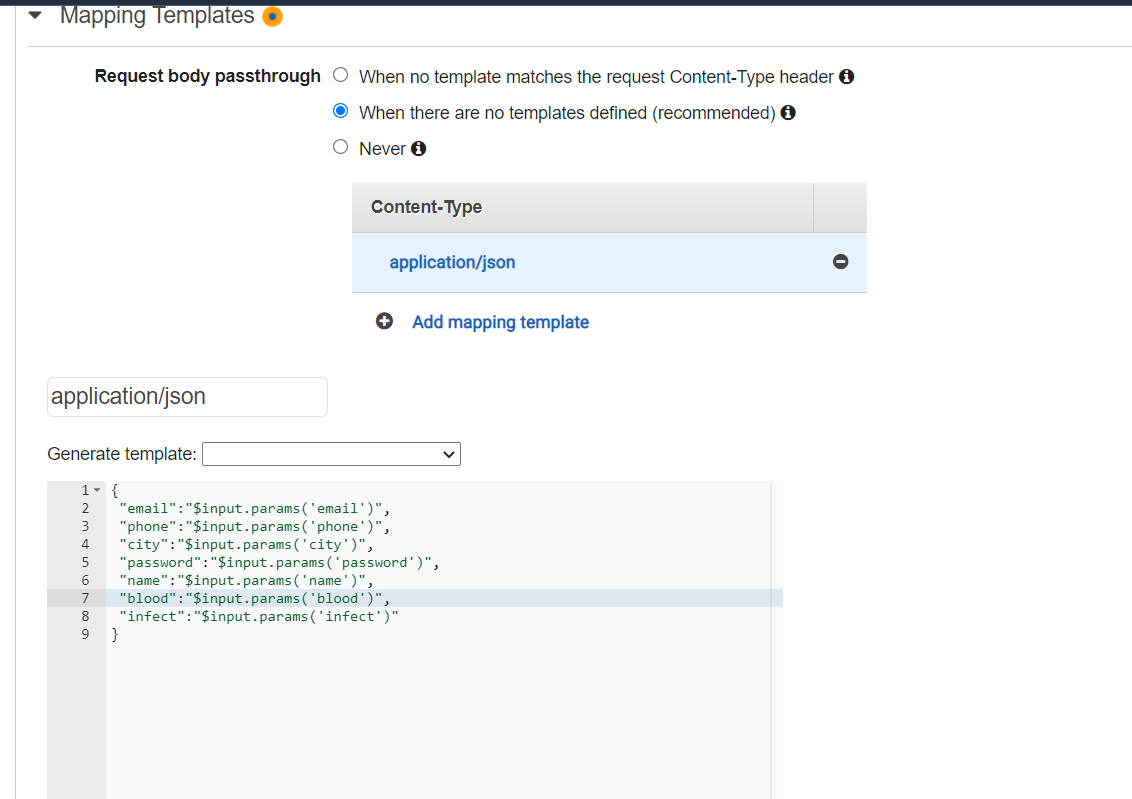
"password":"$input.params('password')",

"name":"$input.params('name')",

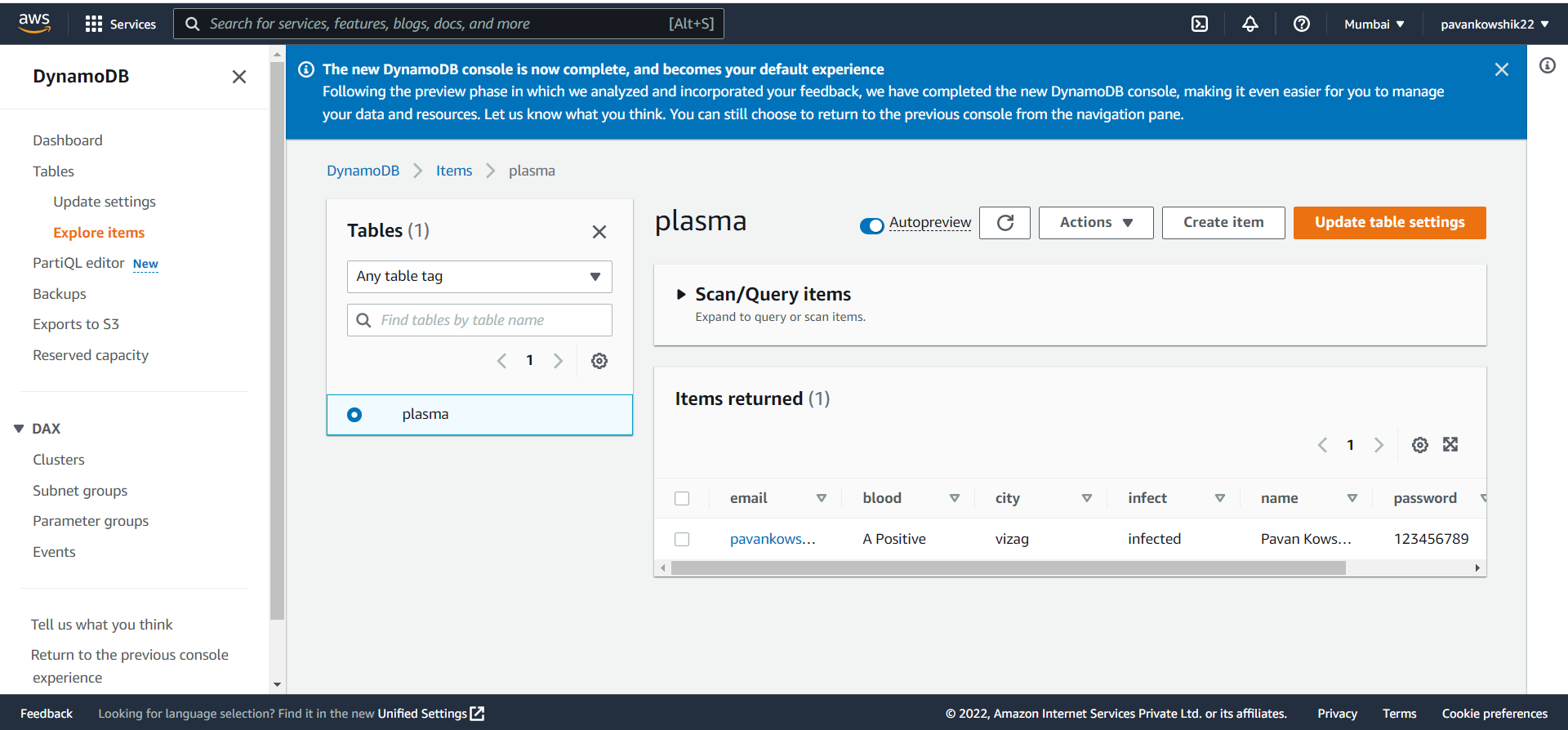
"blood":"$input.params('blood')",

"infect":"$input.params('infect')"

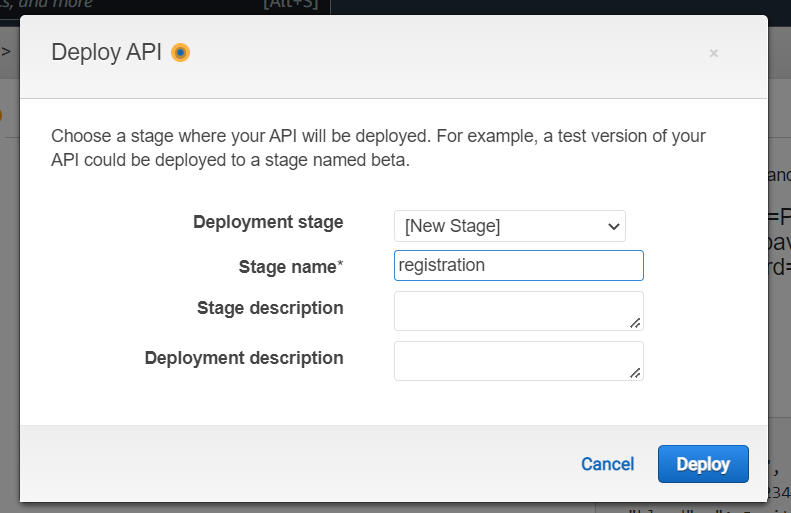
}



name=PavanKowshik&email=ram123 @gmail.com&phone=987654321&city=hyd&infect=infected&blood=A Positive&password=123456789



1. Now deploy the application



**Invoke URL:** <https://2ek8tqgxzb.execute-api.ap-south-1.amazonaws.com/registration>