Plasma Donor App with Serverless Computing

INTRODUCTION

- Overview

The main objective of this project is to provide the recipient with a donor who is in good form with no health ailments to donate blood of the corresponding blood group. This project provides quick access to donors for an immediate requirement of blood. In case of an emergency/surgery, blood procurement is always a major problem which consumes a lot of time. This helps serve the major time-lapse in which a life can be saved!

- Purpose

Serverless computing is the current trend in software application development. Microservices are a popular new approach for building maintainable, scalable, cloud-based applications. AWS is the perfect platform for hosting micro-services. In this project, we will be building a plasma donor app with AWS services like lambda functions, API gateway, and DynamoDB.

LITERATURE SURVEY

- Existing problem

During the COVID 19 crisis, the requirement of plasma became high and the donor count being low. Saving the donor information and helping the need by notifying the current donors would be a helping hand. In regard to the problem faced, an application is to be built which would take the donor details store it and inform them upon a request.

- Proposed solution

To build a web application that is capable of acting as a medium for recipients and donors of blood. The application must be deployed on Elastic Beanstalk. Create an API Endpoint for the model with the help of API Gateway and AWS Lambda Service. An alert is to be sent using the Simple Notification Service to all the registered users whenever a request for blood is posted.
Recipient: The one who has a requirement of blood can register for the first time and then log in from the next for any requirement of blood. He or She can provide the recipient's details such as the blood group, sex and age, and the minimum time for donation so that the admin as well as the donors can view and act accordingly.
Donor: The one who wants to donate blood can register and login to the site and check for any updates on requirements. If they wish to donate, they can get into contact with the recipient and proceed.

THEORITICAL ANALYSIS

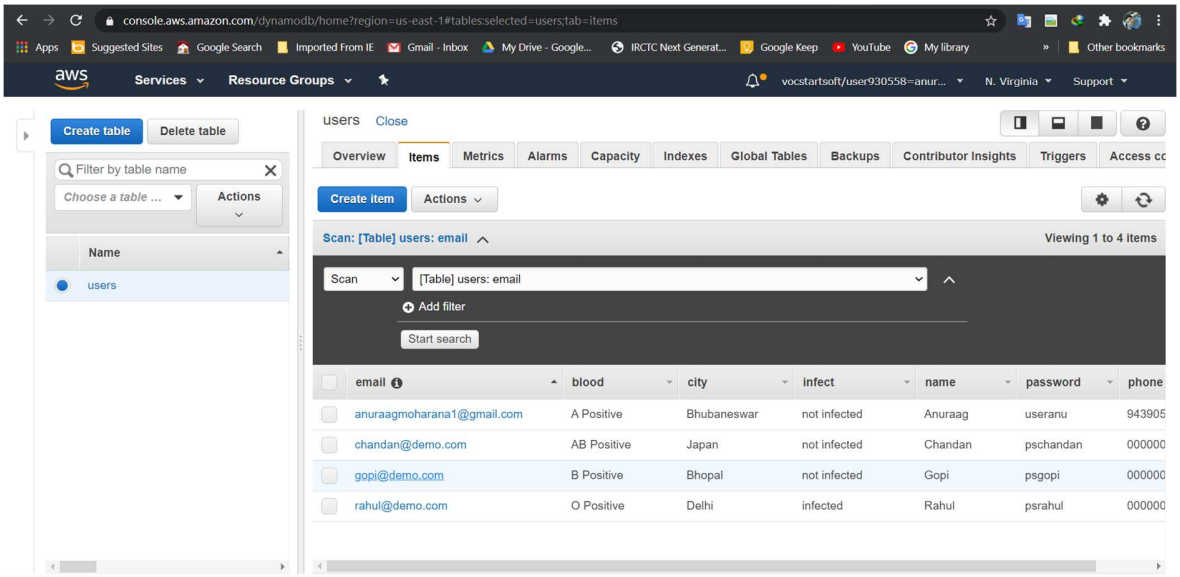
Hardware / Software designing
Block Diagram:

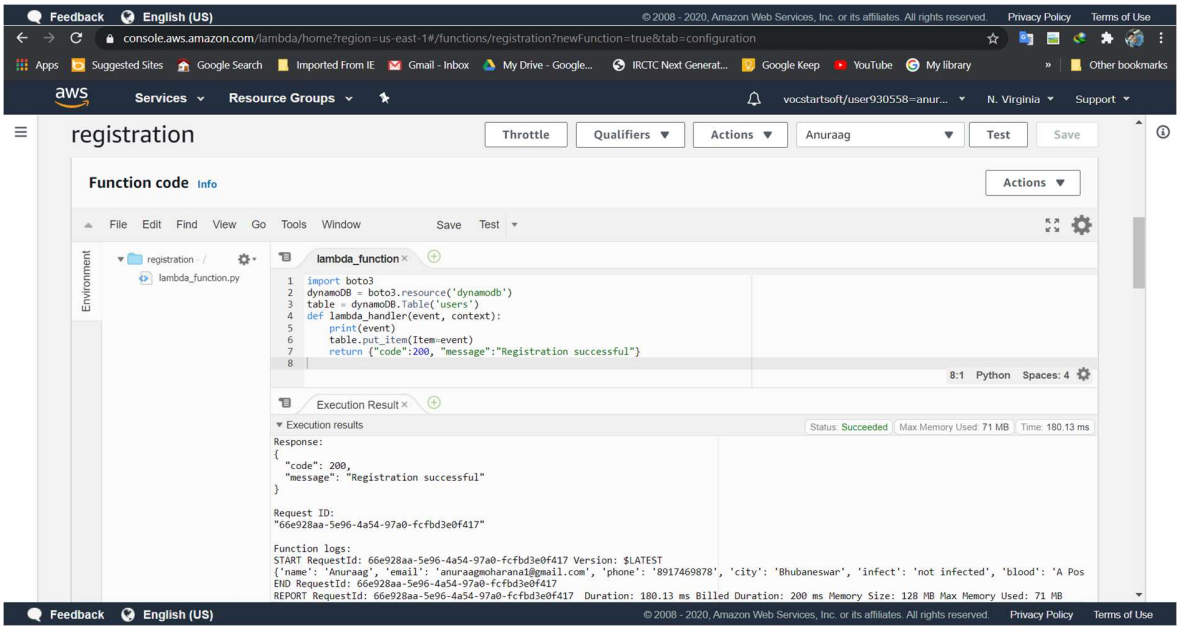
Project Work Flow:

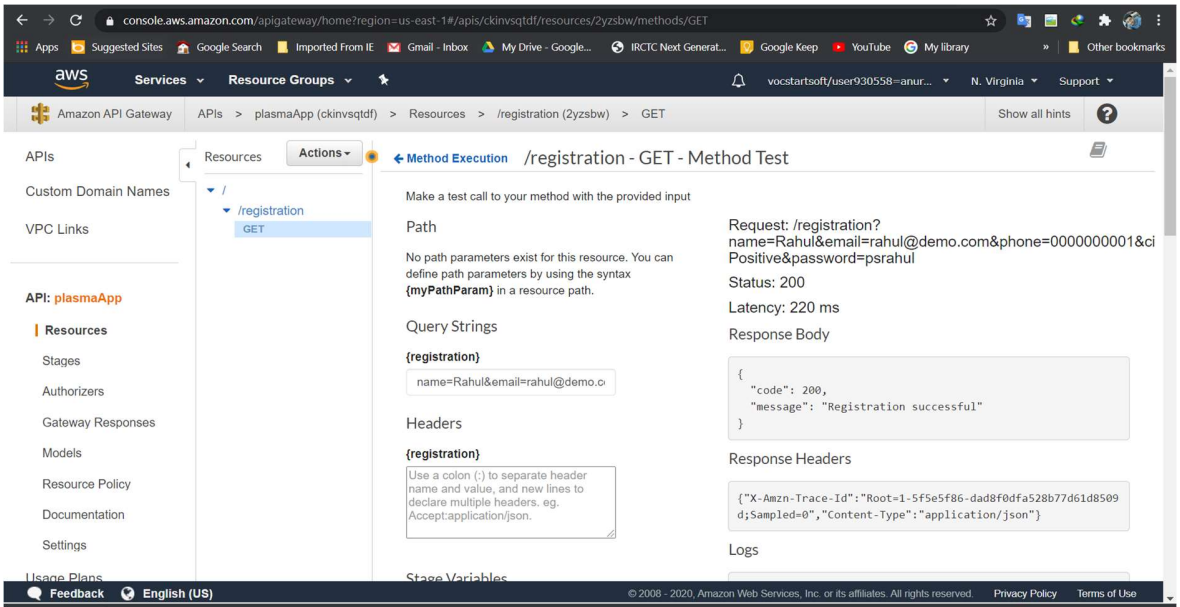
- The user interacts with the application.
- Register by giving the details as a donor.
- The database will have all the details and if a user posts a request then the concerned blood group donors will get notified about it.

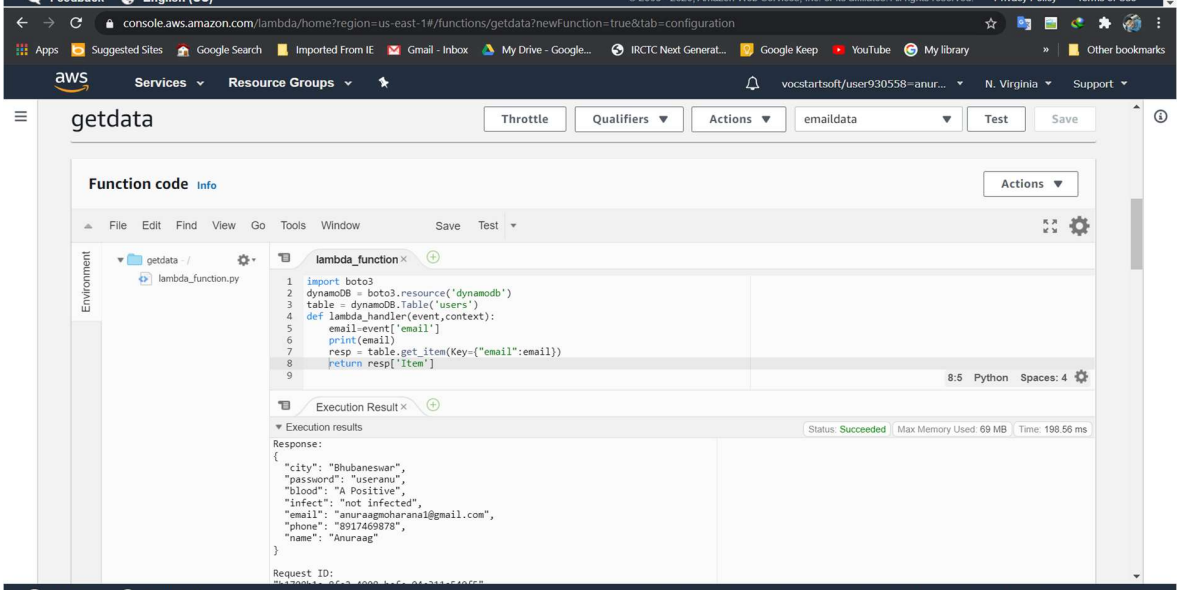
RESULT

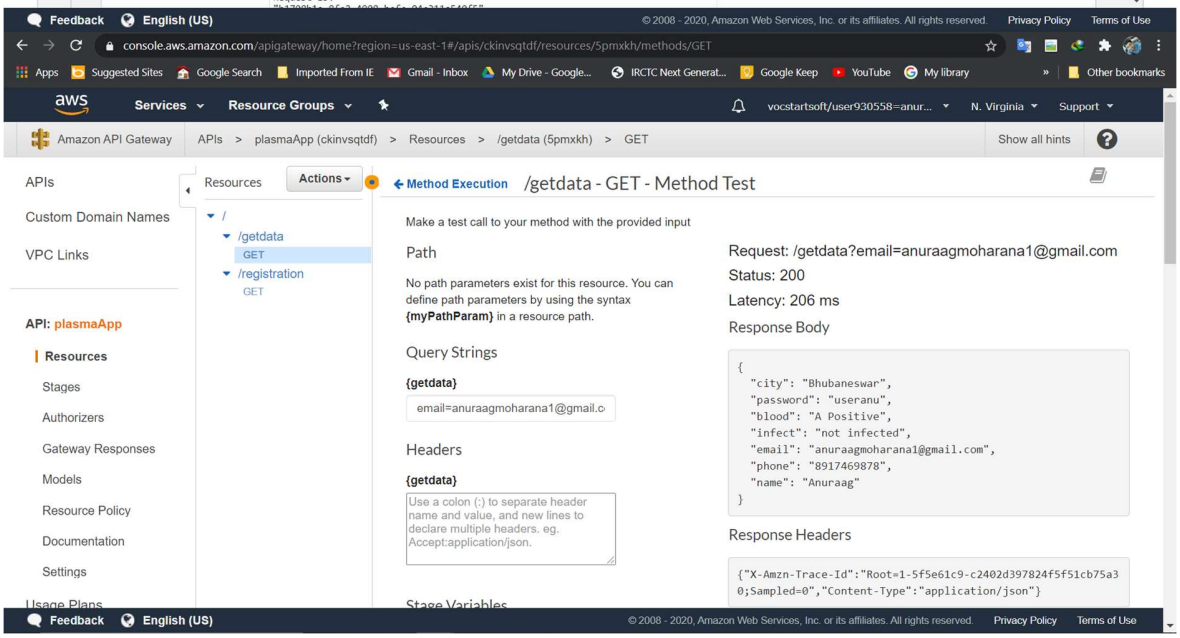
- Screenshots of output

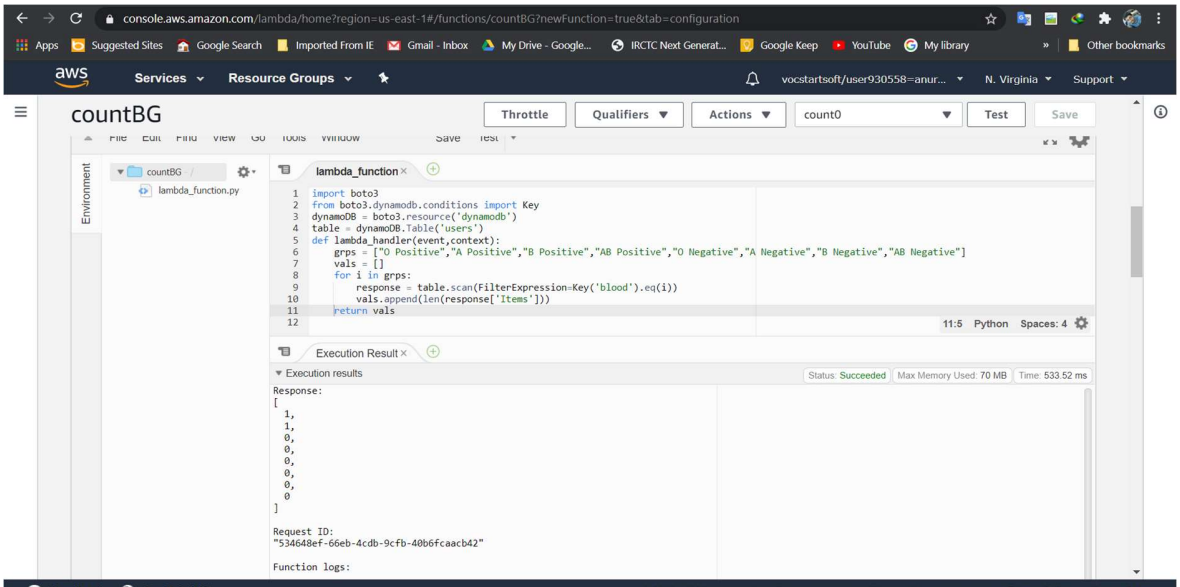


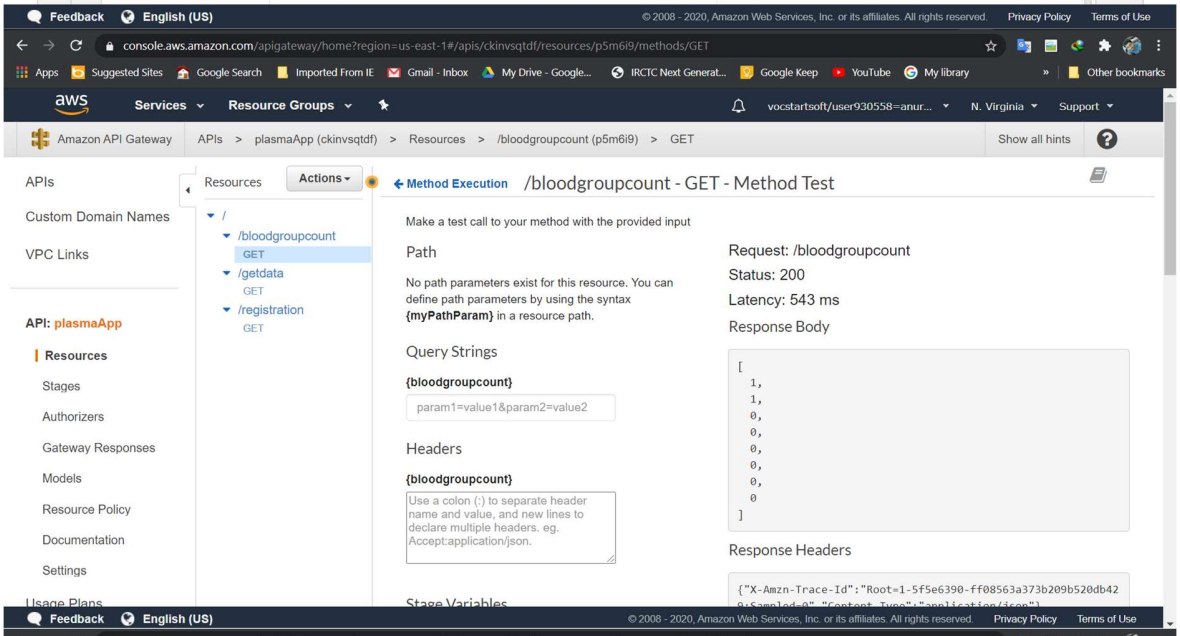


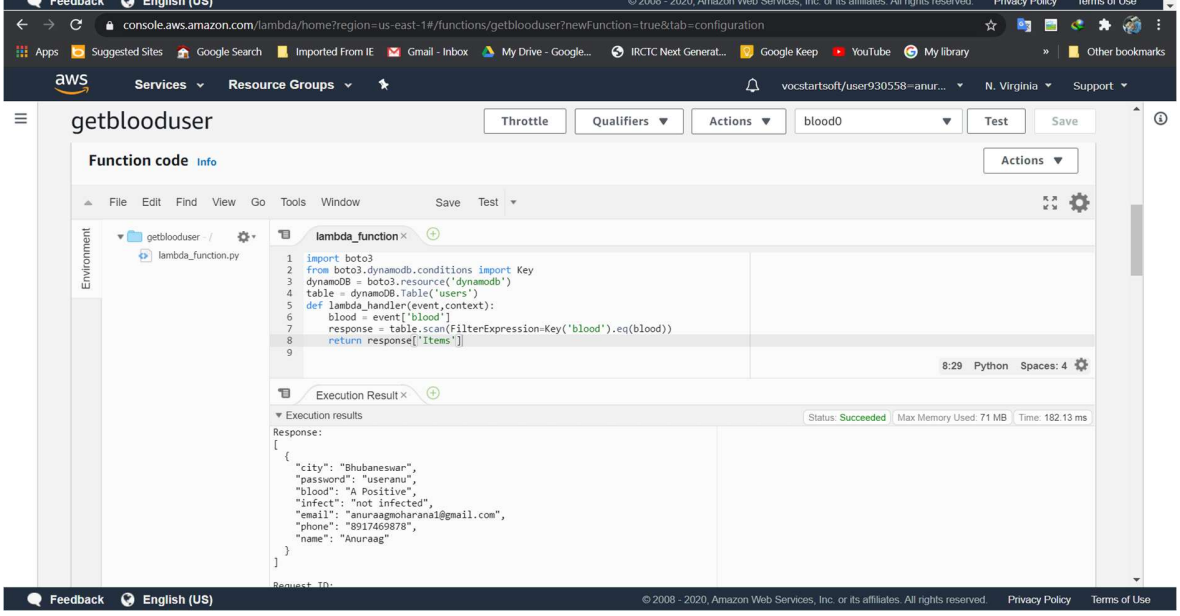

console.aws.amazon.com/apigateway/home?region=us-east-1#/apis/ckinvsqtdf/resources/2yzsbw/methods/GET
Services
Resource Groups
Amazon API Gateway
APIs > plasmaApp (ckinvsqtdf) > Resources > /registration (2yzsbw) > GET
Show all hints
APIs
Custom Domain Names
VPC Links
API: plasmaApp
Resources
Stages
Authorizers
Gateway Responses
Models
Resource Policy
Documentation
Settings
Usage Plans
Resources
Actions
/
/registration
GET
Method Execution
/registration - GET - Method Test
Make a test call to your method with the provided input
Path
No path parameters exist for this resource. You can define path parameters by using the syntax {myPathParam} in a resource path.
Query Strings
{registration}
name=Rahul&email=rahul@demo.c
Headers
{registration}
Use a colon (:) to separate header name and value, and new lines to declare multiple headers. eg. Accept:application/json.
Request: /registration?name=Rahul&email=rahul@demo.com&phone=0000000001&ci
Positive&password=psrahul
Status: 200
Latency: 220 ms
Response Body
{
"code": 200,
"message": "Registration successful"
}
Response Headers
{"X-Amzn-Trace-Id":"Root=1-5f5e5f86-dad8f0dfa528b77d61d8509
d;Sampled=0","Content-Type":"application/json"}
Logs
Feedback
English (US)
© 2008 - 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy Policy
Terms of Use

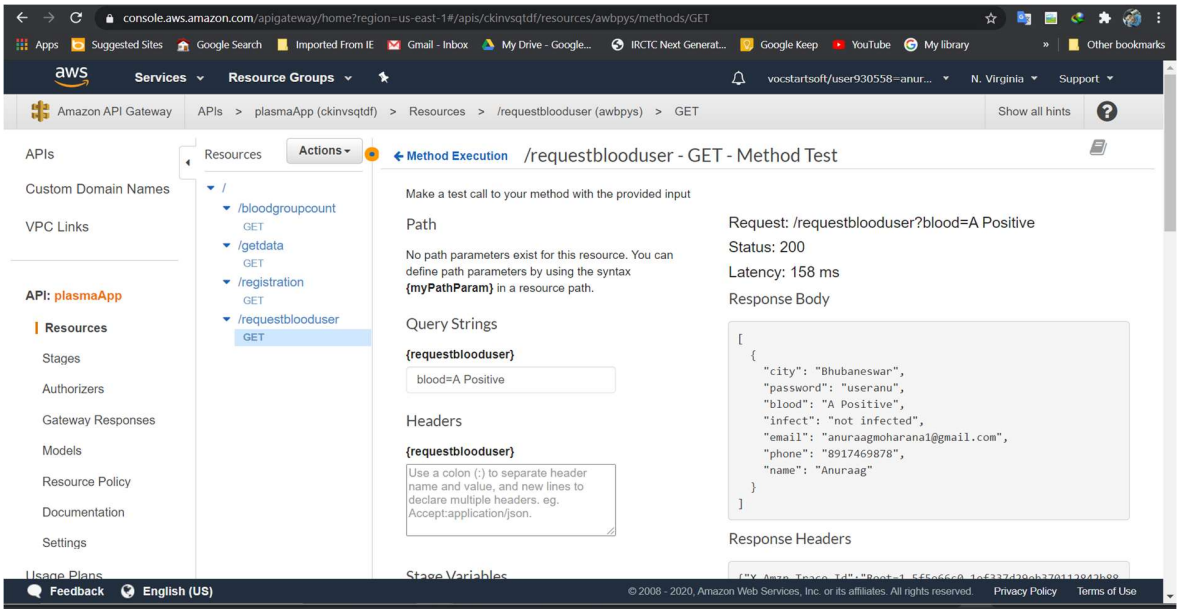

console.aws.amazon.com/lambda/home?region=us-east-1#/functions/getdata?newFunction=true&tab=configuration
getdata
Throttle
Qualifiers
Actions
emaildata
Test
Save
Function code Info
Actions
File Edit Find View Go Tools Window
Save Test
Environment
getdata
lambda_function.py
lambda_function
import boto3
dynamoDB = boto3.resource('dynamodb')
table = dynamoDB.Table('users')
def lambda_handler(event,context):
email=event['email']
print(email)
resp = table.get_item(Key={"email":email})
return resp['Item']
8:5 Python Spaces: 4
Execution Result
Execution results
Status: Succeeded
Max Memory Used: 69 MB
Time: 198.56 ms
Response:
{
"city": "Bhubaneswar",
"password": "useranu",
"blood": "A Positive",
"infect": "not infected",
"email": "anuraagmoharana1@gmail.com",
"phone": "8917469878",
"name": "Anuraag"
}
Request ID:
Feedback
English (US)
© 2008 - 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy Policy
Terms of Use

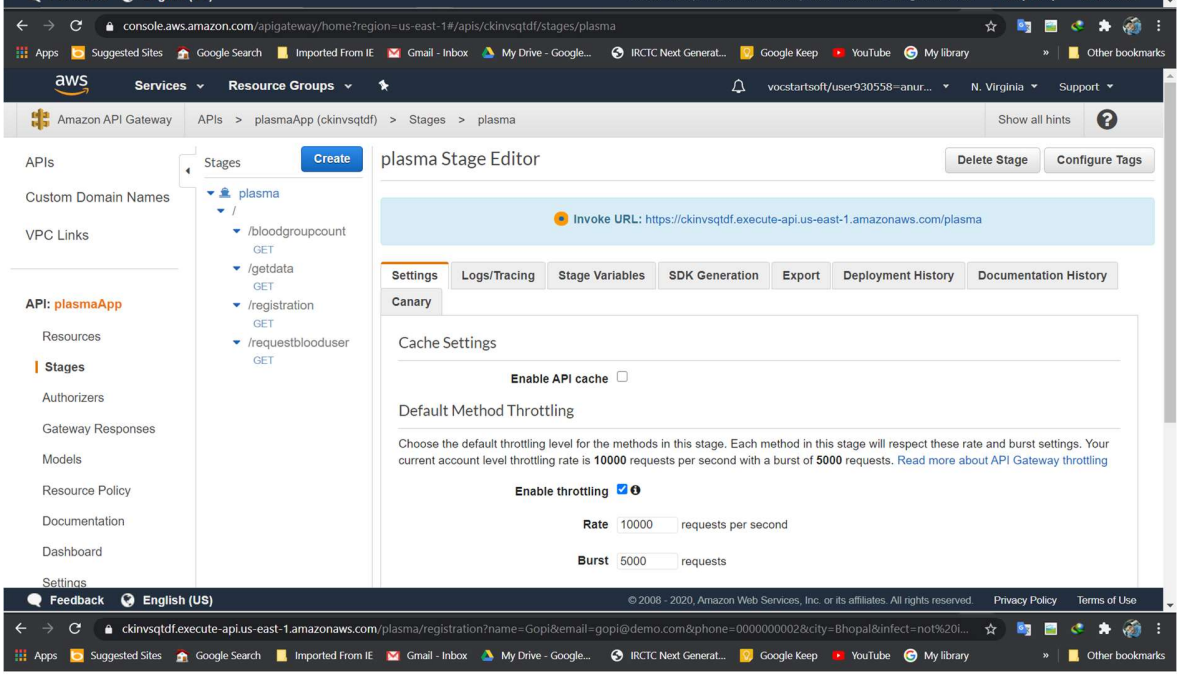

console.aws.amazon.com/apigateway/home?region=us-east-1#/apis/ckinvsqtdf/resources/5pmxkh/methods/GET
Services
Resource Groups
Amazon API Gateway
APIs > plasmaApp (ckinvsqtdf) > Resources > /getdata (5pmxkh) > GET
Show all hints
APIs
Custom Domain Names
VPC Links
API: plasmaApp
Resources
Stages
Authorizers
Gateway Responses
Models
Resource Policy
Documentation
Settings
Usage Plans
Resources
Actions
/
/getdata
GET
/registration
GET
Method Execution
/getdata - GET - Method Test
Make a test call to your method with the provided input
Path
No path parameters exist for this resource. You can define path parameters by using the syntax {myPathParam} in a resource path.
Query Strings
{getdata}
email=anuraagmoharana1@gmail.c
Headers
{getdata}
Use a colon (:) to separate header name and value, and new lines to declare multiple headers. eg. Accept:application/json.
Request: /getdata?email=anuraagmoharana1@gmail.com
Status: 200
Latency: 206 ms
Response Body
{
"city": "Bhubaneswar",
"password": "useranu",
"blood": "A Positive",
"infect": "not infected",
"email": "anuraagmoharana1@gmail.com",
"phone": "8917469878",
"name": "Anuraag"
}
Response Headers
{"X-Amzn-Trace-Id":"Root=1-5f5e61c9-c2402d397824f5f51cb75a3
0;Sampled=0","Content-Type":"application/json"}
Feedback
English (US)
© 2008 - 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy Policy
Terms of Use


console.aws.amazon.com/lambda/home?region=us-east-1#/functions/countBG?newFunction=true&tab=configuration
countBG
Throttle
Qualifiers
Actions
count0
Test
Save
lambda_function
import boto3
from boto3.dynamodb.conditions import Key
dynamoDB = boto3.resource('dynamodb')
table = dynamoDB.Table('users')
def lambda_handler(event,context):
grps = ["O Positive","A Positive","B Positive","AB Positive","O Negative","A Negative","B Negative","AB Negative"]
vals = []
for i in grps:
response = table.scan(FilterExpression=Key('blood').eq(i))
vals.append(len(response['Items']))
return vals
Execution Result
Execution results
Status: Succeeded
Response:
[
1,
1,
0,
0,
0,
0,
0,
0
]
Request ID:
"534648ef-66eb-4cdb-9cfb-40b6fcaacb42"
Function logs:

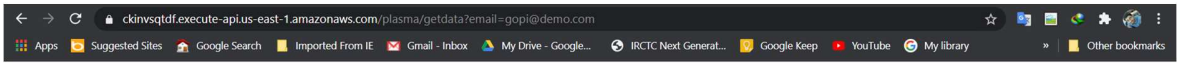

console.aws.amazon.com/apigateway/home?region=us-east-1#/apis/ckinvsqtdf/resources/p5m6i9/methods/GET
Amazon API Gateway
APIs > plasmaApp (ckinvsqtdf) > Resources > /bloodgroupcount (p5m6i9) > GET
Show all hints
APIs
Custom Domain Names
VPC Links
API: plasmaApp
Resources
Stages
Authorizers
Gateway Responses
Models
Resource Policy
Documentation
Settings
Usage Plans
Resources
Actions
/
/bloodgroupcount
GET
/getdata
GET
/registration
GET
Method Execution
/bloodgroupcount - GET - Method Test
Make a test call to your method with the provided input
Path
No path parameters exist for this resource. You can define path parameters by using the syntax {myPathParam} in a resource path.
Query Strings
{bloodgroupcount}
param1=value1&param2=value2
Headers
{bloodgroupcount}
Use a colon (:) to separate header name and value, and new lines to declare multiple headers. eg. Accept:application/json.
Request: /bloodgroupcount
Status: 200
Latency: 543 ms
Response Body
[
1,
1,
0,
0,
0,
0,
0,
0
]
Response Headers
{"X-Amzn-Trace-Id":"Root=1-5f5e6390-ff08563a373b209b520db42


console.aws.amazon.com/lambda/home?region=us-east-1#/functions/getblooduser?newFunction=true&tab=configuration
getblooduser
Throttle
Qualifiers
Actions
blood0
Test
Save
Function code Info
Actions
lambda_function
import boto3
from boto3.dynamodb.conditions import Key
dynamoDB = boto3.resource('dynamodb')
table = dynamoDB.Table('users')
def lambda_handler(event,context):
blood = event['blood']
response = table.scan(FilterExpression=Key('blood').eq(blood))
return response['Items']
Execution Result
Execution results
Status: Succeeded
Response:
[
{
"city": "Bhubaneswar",
"password": "useranu",
"blood": "A Positive",
"infect": "not infected",
"email": "anuraagmoharana1@gmail.com",
"phone": "8917469878",
"name": "Anuraag"
}
]

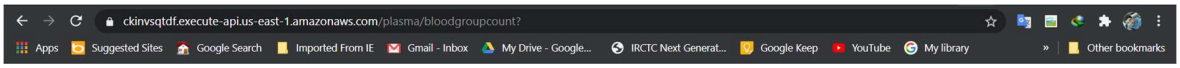





```
{"code": 200, "message": "Registration successful"}
```

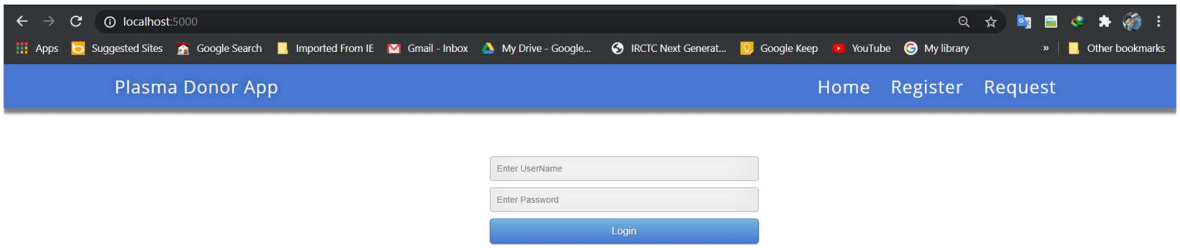


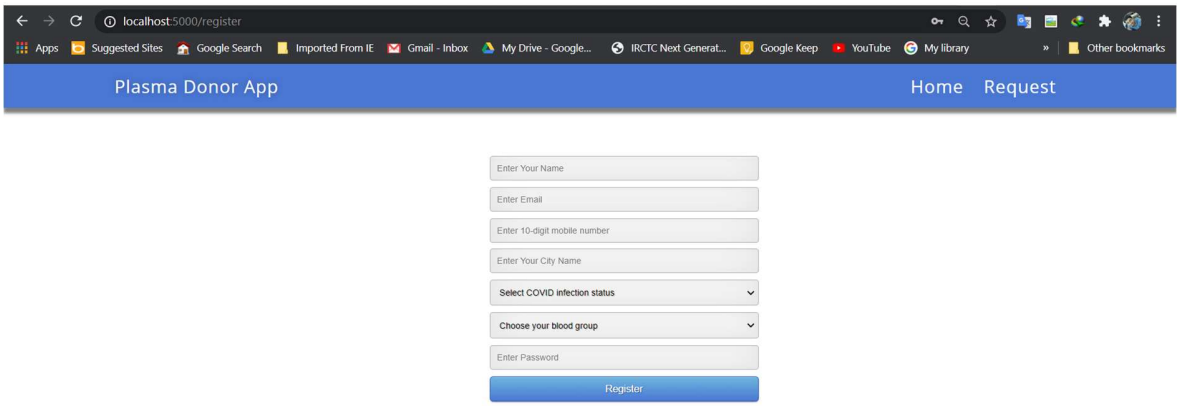

```
{"city": "Bhopal", "password": "psgopi", "blood": "B Positive", "infect": "not infected", "email": "gopi@demo.com", "phone": "0000000002", "name": "Gopi"}
```

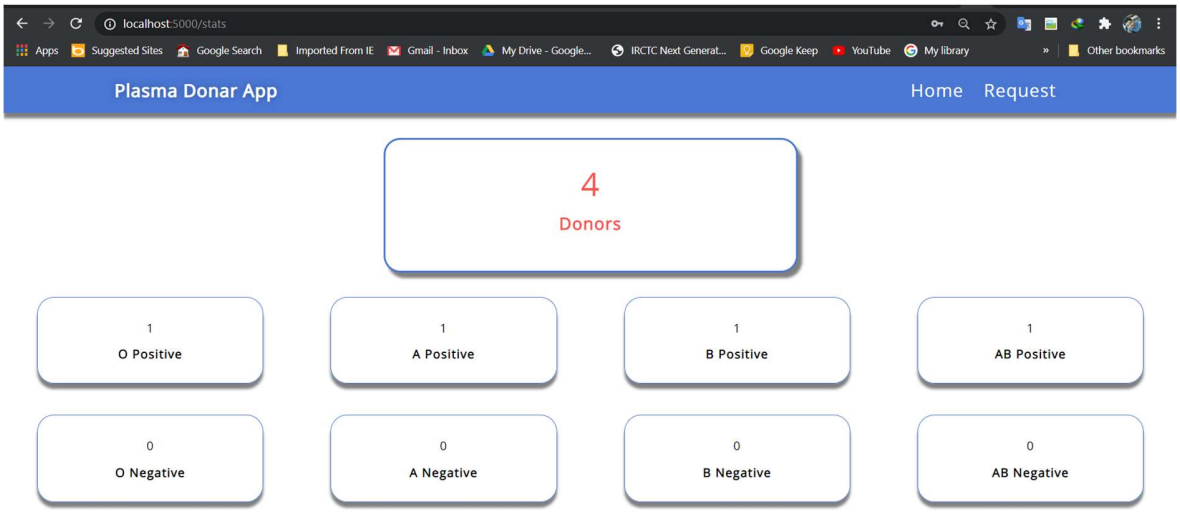


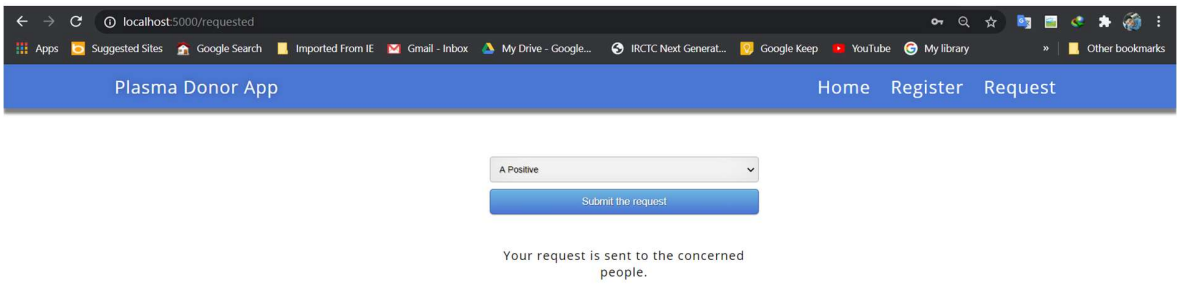

```
[1, 1, 1, 0, 0, 0, 0, 0]
```


```
[{"city": "Bhopal", "password": "psgopi", "blood": "B Positive", "infect": "not infected", "email": "gopi@demo.com", "phone": "0000000002", "name": "Gopi"}]
```


localhost:5000
Apps
Suggested Sites
Google Search
Imported From IE
Gmail - Inbox
My Drive - Google...
IRCTC Next Generat...
Google Keep
YouTube
My library
Other bookmarks
Plasma Donor App
Home
Register
Request
Enter UserName
Enter Password
Login


localhost:5000/register
Apps
Suggested Sites
Google Search
Imported From IE
Gmail - Inbox
My Drive - Google...
IRCTC Next Generat...
Google Keep
YouTube
My library
Other bookmarks
Plasma Donor App
Home
Request
Enter Your Name
Enter Email
Enter 10-digit mobile number
Enter Your City Name
Select COVID infection status
Choose your blood group
Enter Password
Register
Registration Successful, please login using your details


localhost:5000/stats
Apps
Suggested Sites
Google Search
Imported From IE
Gmail - Inbox
My Drive - Google...
IRCTC Next Generat...
Google Keep
YouTube
My library
Other bookmarks
Plasma Donar App
Home
Request
4
Donors
1
O Positive
1
A Positive
1
B Positive
1
AB Positive
0
O Negative
0
A Negative
0
B Negative
0
AB Negative


localhost:5000/requested
Apps
Suggested Sites
Google Search
Imported From IE
Gmail - Inbox
My Drive - Google...
IRCTC Next Generat...
Google Keep
YouTube
My library
Other bookmarks
Plasma Donor App
Home
Register
Request
A Positive
Submit the request
Your request is sent to the concerned people.

ADVANTAGES & DISADVANTAGES

It has many advantages; it has a very simple and clean user interface. A large number of Active Donors around can help a large number of needy people free of cost.
There are some disadvantages also, it can be misused for back marketing, and false blood request and registration vulnerabilities are also there.

APPLICATIONS

- Thousands of Active Donors around can help in any public workplaces and societies.
- Thousands of Donors immediately Get a push notification/email/SMS of your Request.
- It can be used in hospitals, big public institutions and workplaces.

CONCLUSION

Plasma & Blood Donation App ,which puts the power to save a live in the palm of your hand. The main purpose of this App is to create & manage a platform for all donors of the world & remove the recent crisis

FUTURE SCOPE

Many more improvements can be made in future, with larger community with integrity and variety , and its area of application shall be highly innovated and implemented.

BIBILOGRAPHY

- smartinternz.com
- AWS Educate resources
- https://www.ncbi.nlm.nih.gov/books/NBK138212/
- Eder A, et al. Selection criteria to protect the blood donor in North America and Europe: past (dogma), present (evidence), and future (hemovigilance). Transfusion Medicine Reviews. 2009;23(3):205–220. [PubMed]
- Moreno J. “Creeping precautionism” and the blood supply. Transfusion. 2003;43:840–842. [PubMed]
- Farrugia A. The mantra of blood safety: time for a new tune? Vox Sanguinis. 2004;86:1–7. [PubMed]

APPENDIX

Source code

```
# -*- coding: utf-8 -*-


from flask import Flask, render_template, request, redirect, url_for
import requests

app = Flask(__name__)

def check(email):
```

```
    url = "https://z94gc1lts1.execute-api.us-east-1.amazonaws.com/plasma/getdata?email="+email
    status = requests.request("GET",url)
    print(status.json())
    return status.json()


@app.route('/registration')
def home():
    return render_template('register.html')

@app.route('/register',methods=['POST'])
def register():
    x = [x for x in request.form.values()]
    print(x)
    params =
"name="+x[0]+"&email="+x[1]+"&phone="+x[2]+"&city="+x[3]+"&infect="+x[4]+"&blood="+x[5]+"&pas
sword="+x[6]

    if('errorType' in check(x[1])):
        url = "https://z94gc1lts1.execute-api.us-east-1.amazonaws.com/plasma/registration?"+params
        response = requests.get(url)
        return render_template('register.html', pred="Registration Successful, please login using your
details")
    else:
        return render_template('register.html', pred="You are already a member, please login using your
details")

@app.route('/')
@app.route('/login')
def login():
    return render_template('login.html')

@app.route('/loginpage',methods=['POST'])
def loginpage():
    user = request.form['user']
    passw = request.form['passw']
    print(user,passw)
    data = check(user)
    if('errorType' in data):
        return render_template('login.html', pred="The username is not found, recheck the spelling or
please register.")
    else:
        if(passw==data['password']):
            return redirect(url_for('stats'))
        else:
            return render_template('login.html', pred="Login unsuccessful. You have entered the wrong
password.")
```

```
@app.route('/stats')
def stats():
   url = "https://z94gc1lts1.execute-api.us-east-1.amazonaws.com/plasma/bloodgroupcount"
   response = requests.get(url)
   r = response.json()
   print(r)
   return
render_template('stats.html',b=sum(r),b1=str(r[0]),b2=str(r[1]),b3=str(r[2]),b4=str(r[3]),b5=str(r[4]),b6=s
tr(r[5]),b7=str(r[6]),b8=str(r[7]))

@app.route('/requester')
def requester():
   return render_template('request.html')


@app.route('/requested',methods=['POST'])
def requested():
   bloodgrp = request.form['bloodgrp']
   #print(bloodgrp)
   url = "https://z94gc1lts1.execute-api.us-east-
1.amazonaws.com/plasma/requestblooduser?blood="+bloodgrp
   status = requests.request("GET",url)
   a=status.json()
   emailids=[]
   for i in a:
      emailids.append(i['email'])
   print(emailids)
   return render_template('request.html', pred="Your request is sent to the concerned people.")


if __name__ == "__main__":
   app.run(debug=True)
```