SPRINT-4

TEAM ID	**PNT2022TMID49640**
PROJECT NAME	**NUTRITION ASSISTANT APPLICATION**

1.main.py

```
from flask import Flask, render_template, request, redirect, url_for, session

import ibm_db

from flask_mail import Mail, Message

import re

from werkzeug.utils import secure_filename

from clarifai_grpc.channel.clarifai_channel import ClarifaiChannel

from clarifai_grpc.grpc.api import service_pb2, resources_pb2, service_pb2_grpc

from clarifai_grpc.grpc.api.status import status_code_pb2


app = Flask(_name_)

mail = Mail(app) # instantiate the mail class


# configuration of mail

app.config['MAIL_SERVER']='smtp.gmail.com'

app.config['MAIL_PORT'] = 465

app.config['MAIL_USERNAME'] = 'derinjose886@gmail.com'

app.config['MAIL_PASSWORD'] = 'lhmjtfrjwblfbgeq'
app.config['MAIL_USE_TLS'] = False
app.config['MAIL_USE_SSL'] = True
mail = Mail(app)
app.secret_key = 'a'
conn = ibm_db.connect("DATABASE=bludb;HOSTNAME=ba99a9e6-d59e-4883-
8fc0-
d6a8c9f7a08f.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud;PORT=31321;SE
CURITY=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;UID=ksm24043;PWD=
ZXsdfH0rppztWofo",'','')
@app.route('/')
def home():
        return render_template('register.html')


@app.route('/login', methods =['GET', 'POST'])
def login():
        global userid
        msg = ''
        if request.method == 'POST'and 'username' in request.form and 'password'
in request.form:
                username = request.form['username']
                password = request.form['password']
                stmt = ibm_db.prepare(conn,'SELECT * FROM accounts WHERE
username = ?AND password = ?')
                ibm_db.bind_param(stmt,1,username)
```

```
            ibm_db.bind_param(stmt,2,password)
            ibm_db.execute(stmt)
            account = ibm_db.fetch_assoc(stmt)
            if account:
                session['loggedin'] = True
                session['username'] = account['USERNAME']
                msg = 'Logged in successfully !'
                return render_template('index.html', msg = msg)
            else:
                msg = 'Incorrect username / password !'
        return render_template('login.html', a = msg)

@app.route('/logout')
def logout():
        session.pop('loggedin', None)
        session.pop('id', None)
        session.pop('username', None)
        return redirect(url_for('login'))

@app.route('/register', methods =['GET', 'POST'])
def register():
        msg = ''
```

```
if request.method == 'POST':
    username = request.form['username']
    email = request.form['email']
    password = request.form['password']
    sql = "SELECT * FROM accounts WHERE username = ? "
    stmt = ibm_db.prepare(conn,sql)
    ibm_db.bind_param(stmt,1,username)
    ibm_db.execute(stmt)
    account = ibm_db.fetch_assoc(stmt)
    msgg = Message(
            'Hello',
            sender ='derinjose886@gmail.com',
            recipients = [email]
           )
    msgg.body = ' Welcome to NUTRI LITE!! Thanks for registering.     '
    mail.send(msgg)

    if account:
        msg = 'Account already exists !'
    elif not re.match(r'[^@]+@[^@]+\.[^@]+', email):
        msg = 'Invalid email address !'
    elif not re.match(r'[A-Za-z0-9]+', username):
            msg = 'Username must contain only characters and numbers !'
        elif not username or not password or not email:
            msg = 'Please fill out the form !'
        else:
            insert_sql = "INSERT INTO accounts (username,email,password) VALUES (?, ?, ?)"
            stmt = ibm_db.prepare(conn,insert_sql)
            ibm_db.bind_param(stmt, 1, username)
            ibm_db.bind_param(stmt, 2, email)
            ibm_db.bind_param(stmt, 3, password)
            ibm_db.execute(stmt)
            msg = 'You have successfully registered !'
    elif request.method == 'POST':
        msg = 'Please fill out the form !'
    return render_template('register.html', msg = msg)


@app.route('/bmi', methods =['GET', 'POST'])
def bmi():
    if request.method == 'POST':
        height = request.form['height']
        weight= request.form['weight']
    return render_template('bmi.html')
```

```
@app.route('/img',    methods =['GET', 'POST'])

def img():

        print(request.form)

        return render_template('image.html')


@app.route('/food')

def food():

        return render_template('food.html')


@app.route("/dashboard", methods=["GET", "POST"])

def dashboard():

    global request

    if flask.request.method == "POST" and session['LoggedIn']:

        if 'file' not in flask.request.files:

            flash('No file part')

            return redirect(flask.request.url)

        file = flask.request.files['file']

        if file.filename == '':

            flash('No image selected')

            return redirect(flask.request.url)
        if file and allowed_file(file.filename):
            filename = secure_filename(file.filename)
            file.save(os.path.join(app.config['UPLOAD_FOLDER'], filename))
            flash('Image successfully uploaded')

            with open(os.path.join(app.config['UPLOAD_FOLDER'], filename), "rb") as
f:
                file_bytes = f.read()

            request = service_pb2.PostModelOutputsRequest(
                    model_id="food-item-v1-recognition",

user_app_id=resources_pb2.UserAppIDSet(app_id=YOUR_APPLICATION_ID),
                    inputs=[
                        resources_pb2.Input(
                            data=resources_pb2.Data(image=resources_pb2.Image(
                                    base64=file_bytes
                                )
                            )
                        )
                    ],
            )
```

```
      response = stub.PostModelOutputs(request, metadata=metadata)


      if response.status.code != status_code_pb2.SUCCESS:
          print(response)
          raise Exception(f"Request failed, status code: {response.status}")


      foodname = response.outputs[0].data.concepts[0].name


      ingredients = ''
      for concept in response.outputs[0].data.concepts:
        ingredients += f"{concept.name}: {round(concept.value, 2)}, "


      nutritionValues = ''
    #headers = {'X-RapidAPI-Key':
"e90a2b1101msh8a9c2a55215e6b8p1b6838jsn26de2538dc24",
    #'X-RapidAPI-Host': "spoonacular-recipe-food-nutrition-v1.p.rapidapi.com"}
      nutritions = {
        "recipesUsed": 10,
        "calories": {
          "value": 470,
          "unit": "calories",
          "confidenceRange95Percent": {
```

```
        "min": 408.93,
        "max": 582.22
    },
    "standardDeviation": 139.8
},
"fat": {
    "value": 17,
    "unit": "g",
    "confidenceRange95Percent": {
        "min": 12.81,
        "max": 21.36
    },
    "standardDeviation": 6.9
},
"protein": {
    "value": 15,
    "unit": "g",
    "confidenceRange95Percent": {
        "min": 9.06,
        "max": 29.78
    },
    "standardDeviation": 16.71
```

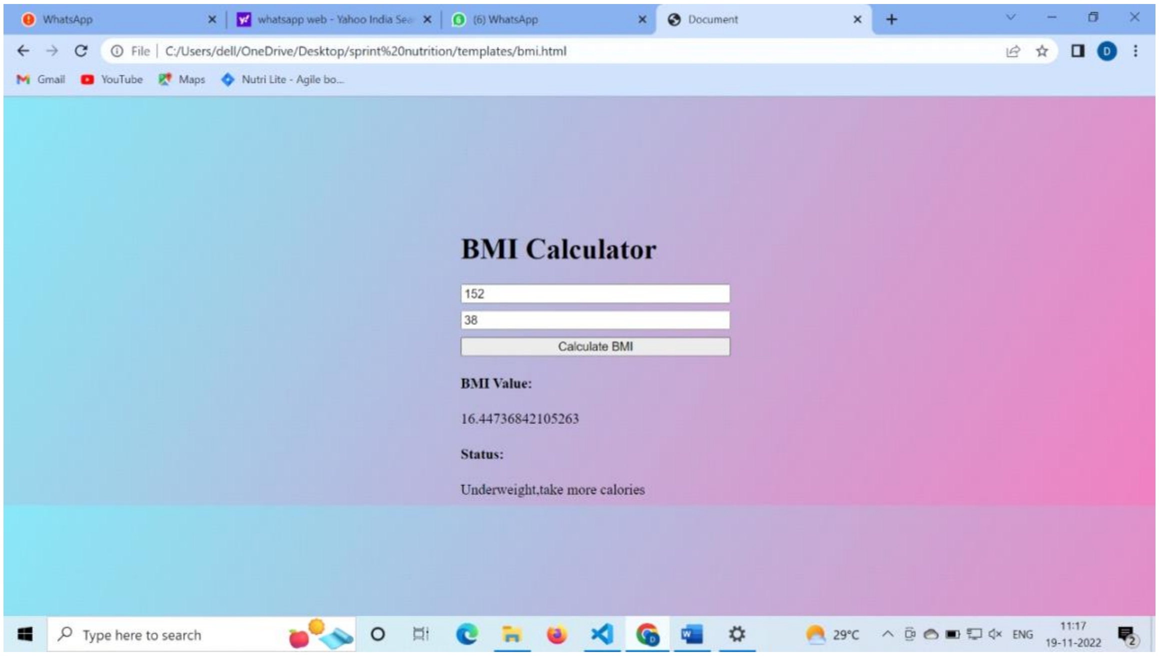
```
            },
            "carbs": {
                "value": 65,
                "unit": "g",
                "confidenceRange95Percent": {
                    "min": 57.05,
                    "max": 77.9
                },
                "standardDeviation": 16.81
            }
        }
        nutritions.pop('recipesUsed')
        for i in nutritions:
            nutritionValues += f"{i}: {nutritions[i]['value']} {nutritions[i]['unit']}, "



        sql = "INSERT INTO foods VALUES(?,?,?,?,?)"
        stmt=ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt, 1, session['userid'])
        ibm_db.bind_param(stmt, 2, datetime.datetime.now().strftime('%Y-%m-%d %H:%M:%S'))
        ibm_db.bind_param(stmt, 3, foodname)
```

```
            ibm_db.bind_param(stmt, 4, ingredients)
            ibm_db.bind_param(stmt, 5, nutritionValues)
            ibm_db.execute(stmt)


            return render_template("dashboard.html",
                filename = filename,
                username = session['username'],
                foodname = foodname,
                ingredients = ingredients,
                nutritionValues = nutritionValues,
            )
        else:
            flash('Allowed image formats - png, jpg, jpeg')
            return redirect(flask.request.url)

if _name_ == '_main_':
  app.run(debug = True)
```

2.BMI.html

```
<!DOCTYPE html>
<html lang="en">
```

```
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <link rel="preconnect" href="https://fonts.gstatic.com">
    <link href="https://fonts.googleapis.com/css2?family=Old+Standard+TT:wght@700&display=swap" rel="stylesheet">
    <div class="container">


    <style>
        .container{
            margin-left: 500px;
            margin-top: 150px;
        }
        .bmi_container {
            display: flex;
            flex-direction: column;
            width: 300px;
            gap: 8px;
        }



    </style>
    </div>
</head>

<body style="background-image: linear-gradient(120deg,#84e1ef,#ea7dbb);">
    <div class="container">
        <h1>BMI Calculator</h1>
        <div class="bmi_container">
            <input id="height" type="number" placeholder="Enter Your Height
in Centimeters:">
            <input id="weight" type="number" placeholder="Enter Your
Weight in Kilograms: ">

            <button type="submit" onclick="Calculate()">Calculate
BMI</button>
        </div>

        <div class="bmi value">
            <h4>BMI Value: </h4>
            <div id="bmioutput"></div>
        </div>
        <div class="status">
            <h4>Status: </h4>
            <div id="bmistatus"></div>
        </div>

    </div>
    <script>
        const bmioutput = document.getElementById('bmioutput')
        const bmistatus = document.getElementById('bmistatus')

        function Calculate() {
            var height = document.getElementById("height").value;
            var weight = document.getElementById("weight").value;

            var result = parseFloat(weight) / (parseFloat(height) / 100) ** 2;

            if (!isNaN(result)) {
                bmioutput.innerHTML = result;
                if (result < 18.5) {
                    bmistatus.innerHTML = "Underweight,take more
calories";
                }
                else if (result < 25) {
                    bmistatus.innerHTML = "Healthy,take nutrition rich
food";
                }
                else if (result < 30) {
                    bmistatus.innerHTML = "Overweight,reduce calorie
intake and do more exercise";
                }
                else {
                    bmistatus.innerHTML = "Obesity,avoid junk food and do
more exercise ";
                }
            }
        }
    </script>

</body>

</html>
```



BMI Calculator

152

38

Calculate BMI

BMI Value:

16.44736842105263

Status:

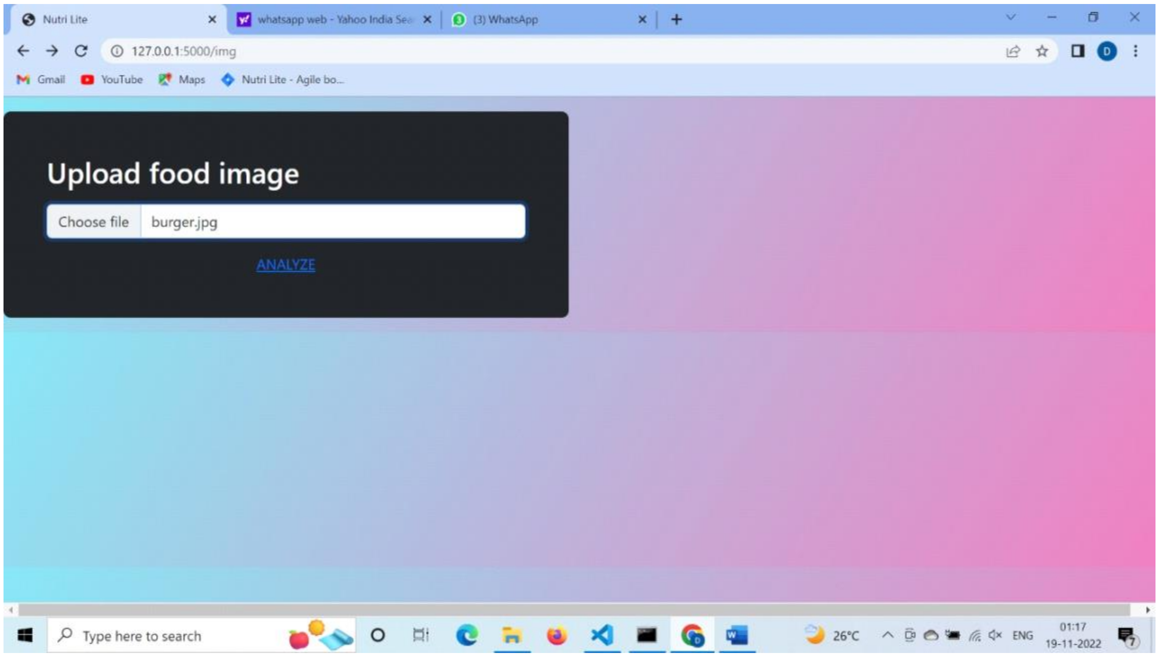
Underweight,take more calories

3.Image.html

```
<!DOCTYPE html>

<html lang="en">

<head>

    <body style="background-image: linear-gradient(120deg,#84e1ef,#ea7dbb);">

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <link rel="stylesheet" href="static/styles.css">

    <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-
Zenh87qX5JnK2Jl0vWa8Ck2rdkQ2Bzep5IDxbcnCeuOxjzrPF/et3URy9Bv1WTRi"
crossorigin="anonymous">
```

```
    <title>Nutri Lite</title>
</head>


        <div class="row align-items-md-stretch">
            <div class="col-md-6 my-3">
                <div class="h-100 p-5 text-bg-dark rounded-3">
                    <h2>Upload food image</h2>
                    <form action = "/dashboard" method = "POST"
enctype="multipart/form-data">
                        <input id="image" class="my-3 form-control" type="file"
name="file" required/>
                        <a href="{{url_for('food')}}"><center>ANALYZE </center></a>



                    </form>
                </div>
            </div>
        </div>


    </div>




    <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/js/bootstrap.bundle.min.
js" integrity="sha384-
OERcA2EqjJCMA+/3y+gxIOqMEjwtxJY7qPCqsdltbNJuaOe923+mo//f6V8Qbsw3"
```

```
crossorigin="anonymous"></script>
</body>
    </body>
</html>
```



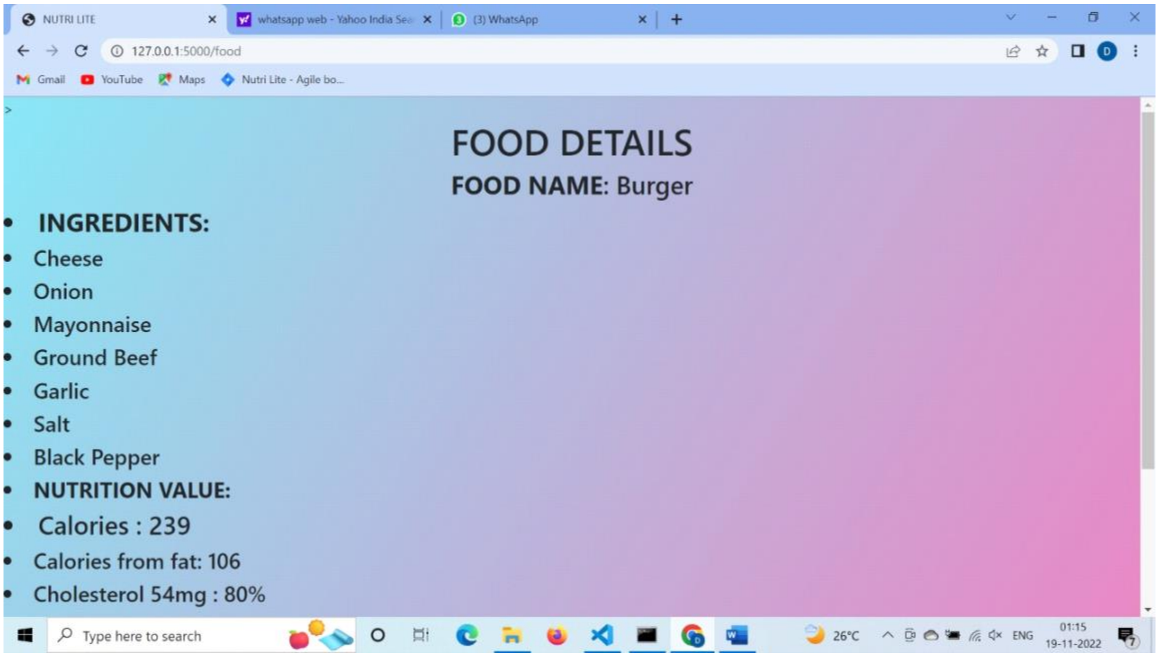
Upload food image

Choose file burger.jpg

4.food.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="static/styles.css">

    <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-
Zenh87qX5JnK2Jl0vWa8Ck2rdkQ2Bzep5IDxbcnCeuOxjzrPF/et3URy9Bv1WTRi"
crossorigin="anonymous">

    <title>NUTRI LITE</title>

</head>

<body style="background-image: linear-
gradient(120deg,#84e1ef,#ea7dbb);",background="bg.png">>

<h1><center>FOOD DETAILS</center></h1>

<h3><center><b>FOOD NAME</b>: </center></h3>

<h3><li><b>INGREDIENTS: </b></li></h3>

<h4><li><ol></ol></li></center></h4>

<h4><li><ol></ol></li></h4>

<h4><li><ol></ol></li></h4>

<h4><li><ol></ol></li></h4>

<h4><li><ol></ol></li></h4>

<h4><li><ol></ol></li></h4>

<h4><li><ol></ol></li></h4>

<h4><li><b>NUTRITION VALUE: </b></li></h4>

<h3><li><ol>CALORIES : </ol></li> </h3>

<h4><li><ol>CALORIES FROM FAT : </ol></li> </h4>

<h4><li><ol>CHOLESTEROL 54mg : </ol></li> </h4>

<h4><li><ol>TOTAL CARBOHYDRATES 18g : </ol></li> </h4>

<h4><li><b>PROTEIN 15g : </b></li> </h4>
```

```
<h4><li><ol>VITAMIN A : </ol></li></h4>
<h4><li><ol>VITAMIN C : </ol></li></h4>
<h4><li><ol>CALCIUM : </ol></li></h4>
<h4><li><ol>IRON : </ol></li></h4>
```



>

FOOD DETAILS

FOOD NAME: Burger

- **INGREDIENTS:**
- Cheese
- Onion
- Mayonnaise
- Ground Beef
- Garlic
- Salt
- Black Pepper
- **NUTRITION VALUE:**
- Calories : 239
- Calories from fat: 106
- Cholesterol 54mg : 80%