project development phase

Sprint-4

Date	Nov 15 2022
Team id	PNT2022TMID32693
Project name	AI powered nutrition analyser for fitness enthusiasts

In sprint-4 we build python code and routed to the html page and analysis image to display nutrition information

Codes

Result.html

```
<html lang="en" dir="ltr">
    <head>
        <meta charset="utf-8">
        <title>Nutrition Image Analysis</title>
    </head>
        <body>
            <h1 id="result-pred" class="bg-warning">{{name}}</h1>
              <table class="table table-dark table-hover">
                <tbody>
                    <tr><td>Calories</td><td>{{calories}} grams</td></tr>
                    <tr><td>Protein</td><td>{{protein}} grams</td></tr>
                    <tr><td>Carbohydrates</td><td>{{carbohydrates}}
grams</td></tr>
                    <tr><td>Sugar</td><td>{{sugar}} grams</td></tr>
                    <tr><td>Fat Saturated</td><td>{{fat_saturated}}
grams</td></tr>
                    <tr><td>Fat Total</td><td>{{fat_total}} grams</td></tr>
                    <tr><td>Cholesterol</td><td>{{cholesterol}}
milligram</td></tr>
                    <tr><td>Fiber</td><td>{{fiber}} grams</td></tr>
                    <tr><td>Sodium</td><td>{{sodium}} milligram</td></tr>
                    <tr><td>Potassium</td><td>{{potassium}}
milligram</td></tr>
                </tbody>
              </table>
        </body>
    </html>
```

Creating flask application and load the model

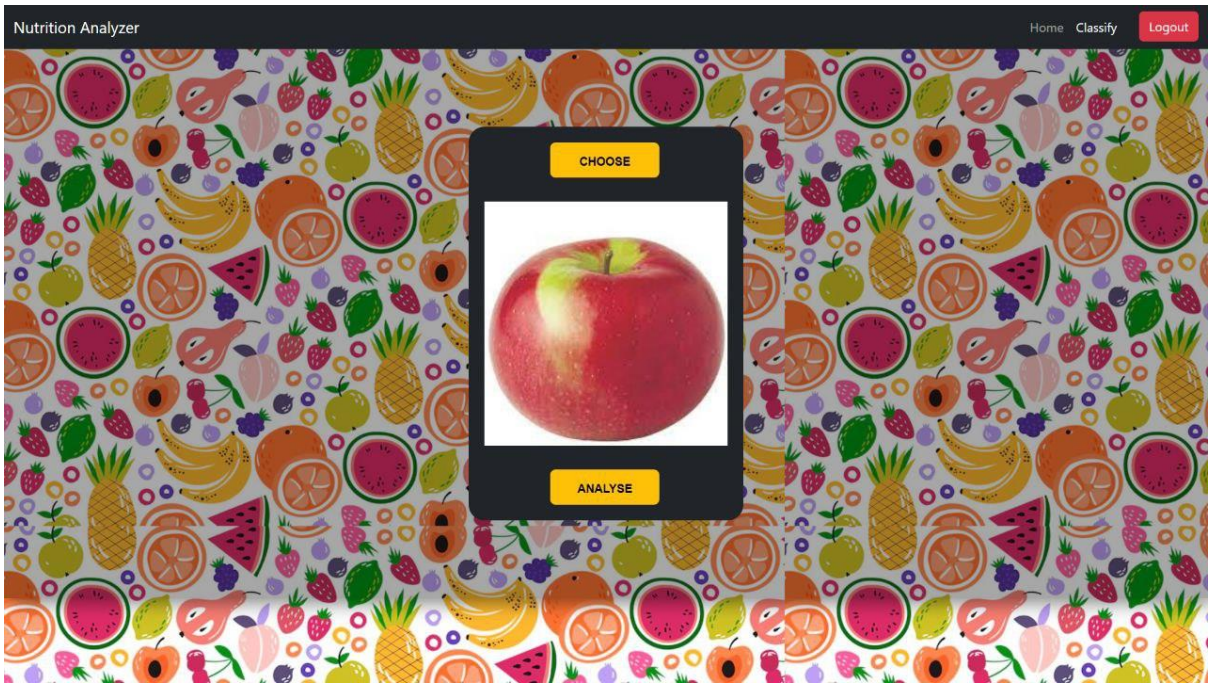
```
from flask import Flask, render_template, request
import os
import numpy as np
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
from flask import session
from flask import Flask, redirect, url_for
import requests
import json
import ibm_db

app = Flask(__name__, template_folder="templates")
app.secret_key = 'NutritionAnalyzer'
model = load_model('NutritionAnalyser.h5')
print("Loaded model from disk")


def connectToDB():
    try:
        connection = ibm_db.connect("DATABASE=bludb;\
        HOSTNAME=125f9f61-9715-46f9-9399-
c8177b21803b.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud;\
        PORT=30426;\
        Security=SSL;\
        SSLServerCertificate=DigiCertGlobalRootCA.crt;\
        UID=qlj81410;\
        PWD=phBPVWNuoifGiYIC;", "", "")
        print("Connected to DB!")
        return connection
    except:
        print("error while connecting ", ibm_db.conn_errormsg())
        return 0


connection = connectToDB()


@app.route('/', methods=['GET', 'POST'])
def login():
    if request.method == 'POST':
        email = request.form['email']
        password = request.form['password']
        print(email)
        print(password)
        query = "SELECT * FROM QLJ81410.USERS WHERE email=? AND password=?"
        stmt = ibm_db.prepare(connection, query)
        ibm_db.bind_param(stmt, 1, email)
```

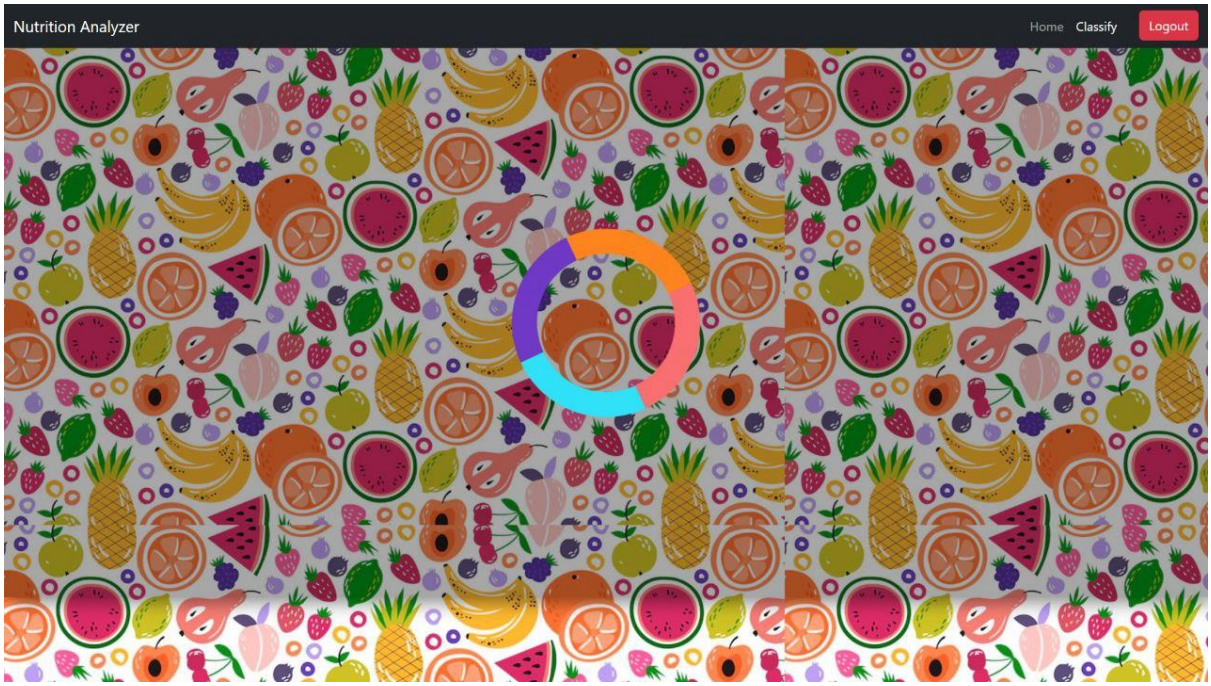
```
        ibm_db.bind_param(stmt, 2, password)
        ibm_db.execute(stmt)
        result = ibm_db.fetch_assoc(stmt)
        print("result - ", result)

        if result:
            print("yes logged in")
            username = result['USERNAME']
            return render_template("home.html", username=(username))

    else:
        return render_template('login.html')


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


@app.route('/Registration', methods=['GET', 'POST'])
def registration():
    if request.method == 'POST':
        username = request.form['username']
        number = request.form['number']
        email = request.form['email']
        password = request.form['password']
        print(username)
        print(number)
        print(email)
        print(password)

        query = "insert into QLJ81410.USERS values('"+username + \
            "','"+number+"','"+email+"','"+password+"')"
        stmt = ibm_db.exec_immediate(connection, query)
        rowcount = ibm_db.num_rows(stmt)

    return render_template('registration.html')


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


@app.route('/predict', methods=['GET', 'POST'])
def launch():
    if request.method == 'POST':

        f = request.files['file']
```

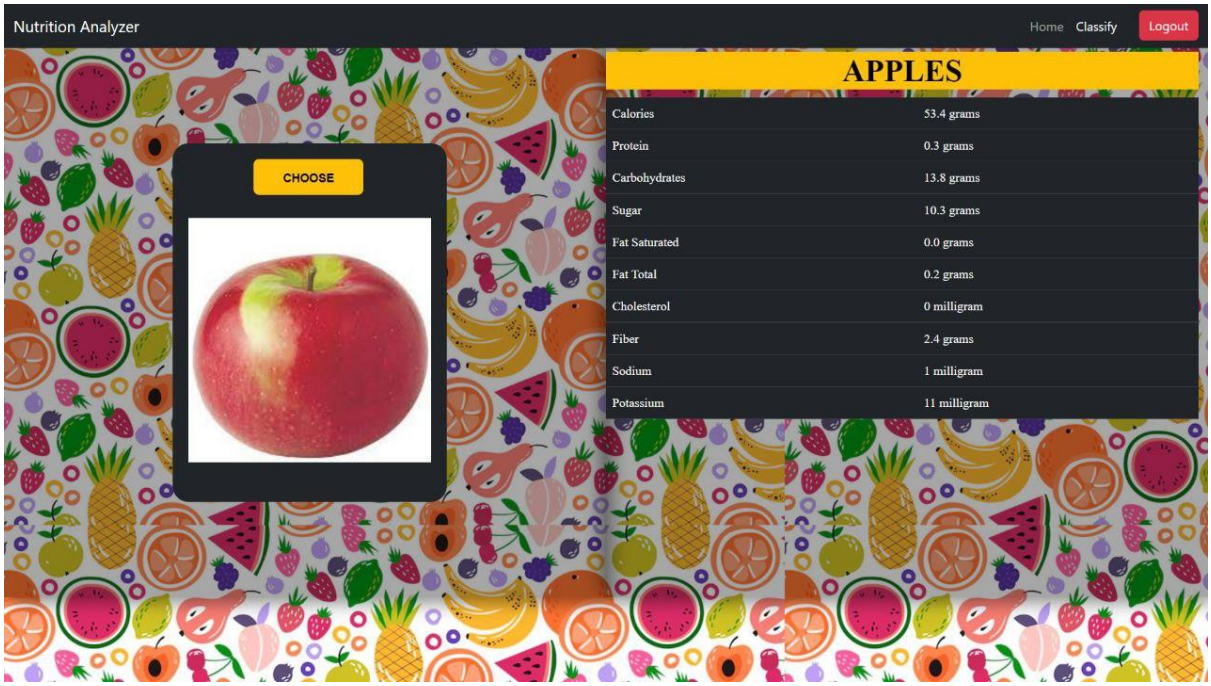
```
        basepath = os.path.dirname('__file__')
        filepath = os.path.join(basepath, "uploads", f.filename)
        f.save(filepath)

        img = image.load_img(filepath, target_size=(64, 64))
        x = image.img_to_array(img)
        x = np.expand_dims(x, axis=0)

        pred = np.argmax(model.predict(x), axis=1)
        print("prediction", pred)
        index = ['APPLES', 'BANANA', 'ORANGE', 'PINEAPPLE', 'WATERMELON']

        predictedValue = str(index[pred[0]])

        print("--------------predictedValue = "+predictedValue)

        result = nutrition(predictedValue)

        print("---------------to server")

        temp = result.json()["items"]
        items = temp[0]
        print(items)
        sugar = items["sugar_g"]
        fiber = items["fiber_g"]
        sodium = items["sodium_mg"]
        potassium = items["potassium_mg"]
        fat_saturated = items["fat_saturated_g"]
        fat_total = items["fat_total_g"]
        calories = items["calories"]
        cholesterol = items["cholesterol_mg"]
        protein = items["protein_g"]
        carbohydrates = items["carbohydrates_total_g"]
        return render_template("result.html", name=(predictedValue),
sugar=(sugar), fiber=(fiber), sodium=(sodium), potassium=(potassium),
fat_saturated=(fat_saturated), fat_total=(fat_total), calories=(calories),
cholesterol=(cholesterol), protein=(protein), carbohydrates=(carbohydrates))


def nutrition(index):

    url = "https://calorieninjas.p.rapidapi.com/v1/nutrition"

    querystring = {"query": index}

    headers = {
        'x-rapidapi-key':
"5d797ab107mshe668f26bd044e64p1ffd34jsnf47bfa9a8ee4",
        'x-rapidapi-host': "calorieninjas.p.rapidapi.com"
```

```
    }

    response = requests.request(
        "GET", url, headers=headers, params=querystring)

    print("from api "+response.text)
    return response


# @app.route('/logout', methods=['GET', 'POST'])
# def logout():
#     if request.method == 'POST':
#         return render_template('login.html')


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

screenshots




Nutrition Analyzer
Home
Classify
Logout


Nutrition Analyzer
Home
Classify
Logout
CHOOSE
APPLES
Calories 53.4 grams
Protein 0.3 grams
Carbohydrates 13.8 grams
Sugar 10.3 grams
Fat Saturated 0.0 grams
Fat Total 0.2 grams
Cholesterol 0 milligram
Fiber 2.4 grams
Sodium 1 milligram
Potassium 11 milligram