SPRINT-4

Final report and screenshot for the nutrition assistant application

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

import requests, json, os
import ibm_db
import re

import cv2

app = Flask(__name__)

app.secret_key = 'a'


conn = ibm_db.connect("DATABASE=bludb;HOSTNAME=6667d8e9-9d4d-4ccb-
ba32-
21da3bb5aafc.c1ogj3sd0tgtu0lqde00.databases.appdomain.cloud;PORT=30376;SEC
URITY=SSL;SSLServerCertificate=DigiCertGlobalRootCA.crt;UID=jqr24896;PWD
=XEJTmHY5JprWZngW",",")



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

@app.route('/pythonlogin/', methods=['GET', 'POST'])
def login():
    global userid
    msg=''
```

```
    if request.method =='POST':
        username = request.form['username']
        password =request.form['password']
        sql ="SELECT * FROM users WHERE username =? AND password=?"
        stmt = ibm_db.prepare(conn, sql)
        ibm_db.bind_param(stmt,1,username)
        ibm_db.bind_param(stmt,2,password)
        ibm_db.execute(stmt)
        account = ibm_db.fetch_assoc(stmt)
        print (account)
        if account:
            session['loggedin']=True
            session['id'] = account ['USERNAME']
            userid = account['USERNAME']
            session['username'] = account['USERNAME']
            msg = 'logged in successfully !'

            return render_template('submission.html',msg = msg)

        else:
            msg ='Incorrect username / password !'
    return render_template('login.html',msg=msg)

@app.route('/pythonlogin/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 users 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)
        print(account)
        if account:
            msg ='Account already exists !'
        elif not re.match(r'[^@]+@[^@]+\.[^@]+',email):
            msg ='Invaild email address !'
        elif not re.match(r'[A-Za-z0-9]+',username):
            msg = 'Name must contain only characters and numbers!'
        else:
            insert_sql = "INSERT INTO users VALUES (?,?,?)"
            prep_stmt= ibm_db.prepare(conn, insert_sql)
            ibm_db.bind_param(prep_stmt, 1 , username)
            ibm_db.bind_param(prep_stmt, 2, email)
            ibm_db.bind_param(prep_stmt, 3, password)
            ibm_db.execute(prep_stmt)

            sendmsg(email,' you have successfully registered !')
            msg = ' you have successfully registered !'
    elif request.method == 'POST':
        # Form is empty... (no POST data)
        msg = 'Please fill out the form!'
    # Show registration form with message (if any)
    return render_template('register.html', msg=msg)
@app.route('/submission')
def submission():

    return render_template('submission.html')

@app.route('/pythonlogin/submission/display', methods = ["POST", "GET"])
def display():
    if request.method == "POST":
```

```
image = request.files["food"]
image.save('static/Out/Test.jpg')

import tensorflow as tf
classifierLoad = tf.keras.models.load_model('model.h5')

import numpy as np
from keras.preprocessing import image

test_image = image.load_img('static/Out/Test.jpg', target_size=(200, 200))
img1 = cv2.imread('static/Out/Test.jpg')
# test_image = image.img_to_array(test_image)
test_image = np.expand_dims(test_image, axis=0)
result = classifierLoad.predict(test_image)

print(result)

out = ''
fer = ''
if result[0][0] == 1:
    out = "APPLES"
elif result[0][1] == 1:
    out = "Badam"
elif result[0][2] == 1:
    out = "Badam Drink"
elif result[0][3] == 1:
    out = "Banana"
elif result[0][4] == 1:
    out = "Beef Steak"
elif result[0][5] == 1:
    out = "BeetrootFry"

elif result[0][6] == 1:
    out = "Biriyani"
```

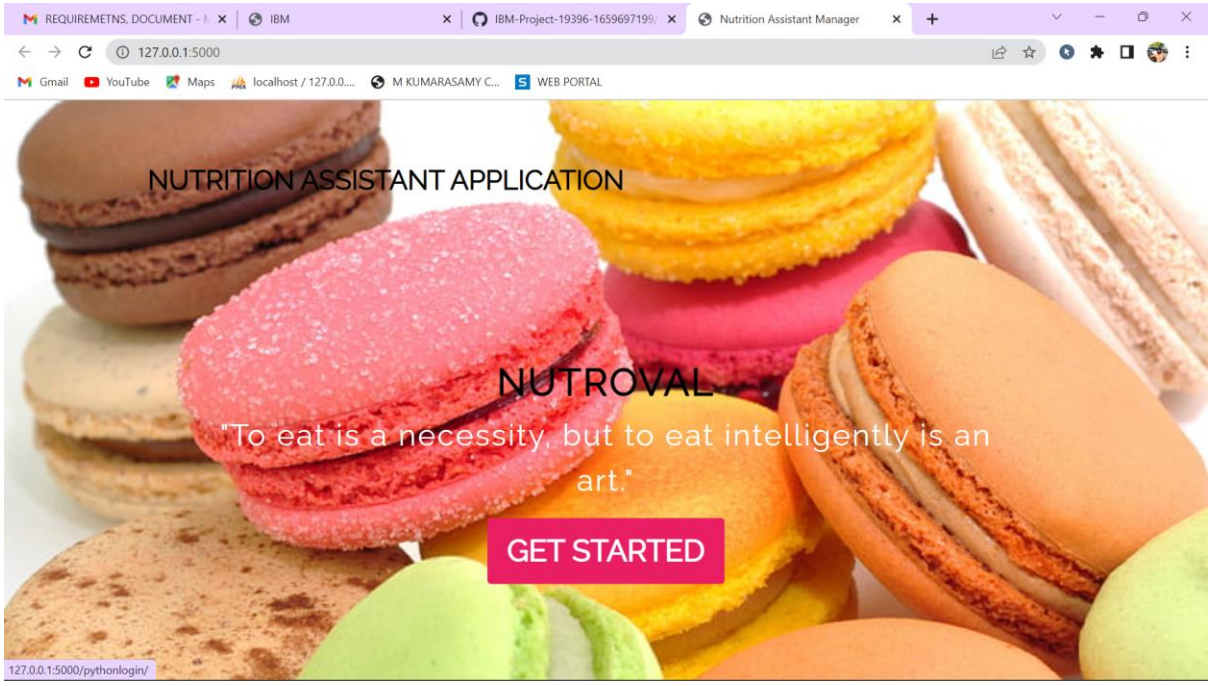
```
elif result[0][7] == 1:
    out = "Biscuits"
elif result[0][8] == 1:
    out = "BitterGuardFry"
elif result[0][9] == 1:
    out = "Boiledegg"
elif result[0][10] == 1:
    out = "Bread with Peanutbutter"

elif result[0][11] == 1:
    out = "BreadandJam"
elif result[0][12] == 1:
    out = "Badam"
elif result[0][13] == 1:
    out = "Burger"
elif result[0][14] == 1:
    out = "CapsicumCurry"
elif result[0][15] == 1:
    out = "Cashew"
elif result[0][16] == 1:
    out = "Chappathi"

elif result[0][17] == 1:
    out = "Cheeseballs"
elif result[0][18] == 1:
    out = "ChilliBeef"
elif result[0][19] == 1:
    out = "Chocolate"
elif result[0][20] == 1:
    out = "ChocolateIcecream"
elif result[0][21] == 1:
    out = "ChoolapooriwithChanna"
elif result[0][22] == 1:
    out = "CoffeeorLatte"
```

```
        elif result[0][23] == 1:
            out = "CrabMasala"
        elif result[0][24] == 1:
            out = "Cucumber"
        elif result[0][25] == 1:
            out = "Curdrice"
        elif result[0][26] == 1:
            out = "Dosa"



        foodName= out

        session["out"]=foodName

        return render_template('Result.html', data=session["out"])

    else:
        return render_template('submission.html')

@app.route("/pythonlogin/submission/out", methods=['GET', 'POST'])
def out():
    if request.method == "POST":
        nutrients = {}
        USDAapiKey = 'flYmBbhNL4RXqkJ7M80adL5dNKvn3THtM0eQ48Er'

        foodName = session["out"]
        response = requests.get(

'https://api.nal.usda.gov/fdc/v1/foods/search?api_key={}&query={}&requireAllWord
s={}'.format(USDAapiKey,
                                                                foodName,
                                                                True))
```

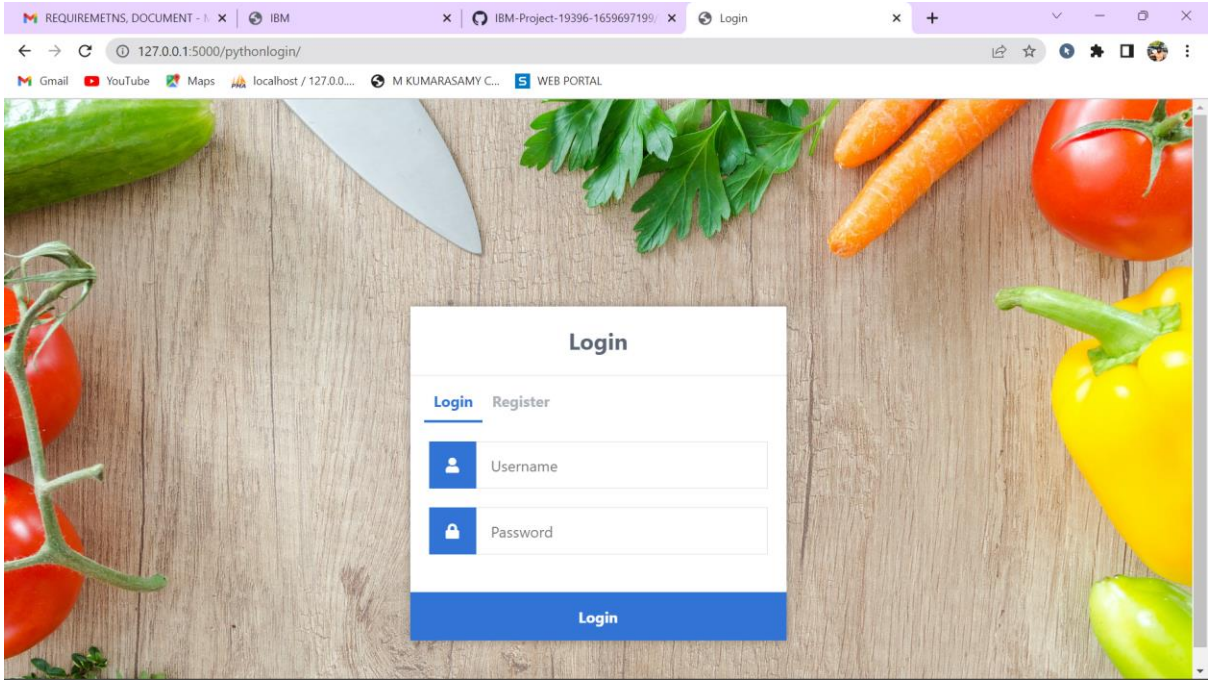
```
        data = json.loads(response.text)
        concepts = data['foods'][0]['foodNutrients']
        arr = ["Sugars", "Energy", "Vitamin A", "Vitamin D", "Vitamin B", "Vitamin
C", "Protein", "Fiber", "Iron",
               "Magnesium",
               "Phosphorus", "Cholestrol", "Carbohydrate", "Total lipid (fat)", "Sodium",
"Calcium", ]
        for x in concepts:
            if x['nutrientName'].split(',')[0] in arr:
                if (x['nutrientName'].split(',')[0] == "Total lipid (fat)"):
                    nutrients['Fat'] = str(x['value']) + " " + x['unitName']
                else:
                    nutrients[x['nutrientName'].split(',')[0]] = str(x['value']) + " " +
x['unitName']

        return render_template('display.html', x=foodName, data=nutrients,
account=session['username'])



def sendmsg(Mailid,message):
    import smtplib
    from email.mime.multipart import MIMEMultipart
    from email.mime.text import MIMEText
    from email.mime.base import MIMEBase
    from email import encoders

    fromaddr = "sampletest685@gmail.com"
    toaddr = Mailid

    # instance of MIMEMultipart
    msg = MIMEMultipart()

    # storing the senders email address
    msg['From'] = fromaddr
```

```
        # storing the receivers email address
        msg['To'] = toaddr

        # storing the subject
        msg['Subject'] = "Alert"

        # string to store the body of the mail
        body = message

        # attach the body with the msg instance
        msg.attach(MIMEText(body, 'plain'))

        # creates SMTP session
        s = smtplib.SMTP('smtp.gmail.com', 587)

        # start TLS for security
        s.starttls()

        # Authentication
        s.login(fromaddr, "hneucvnontsuwgpj")

        # Converts the Multipart msg into a string
        text = msg.as_string()

        # sending the mail
        s.sendmail(fromaddr, toaddr, text)

        # terminating the session
    if __name__=='__main__':
app.run(host='0.0.0.0',debug = True, port = 5000)
```

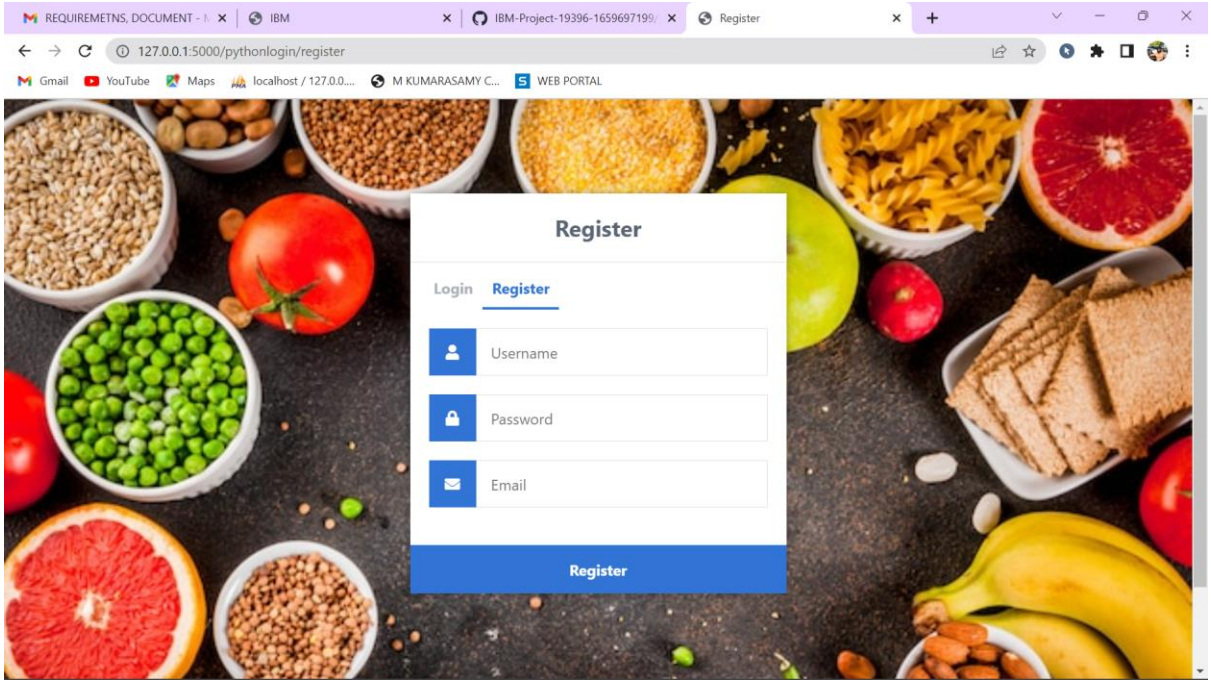
Home page:



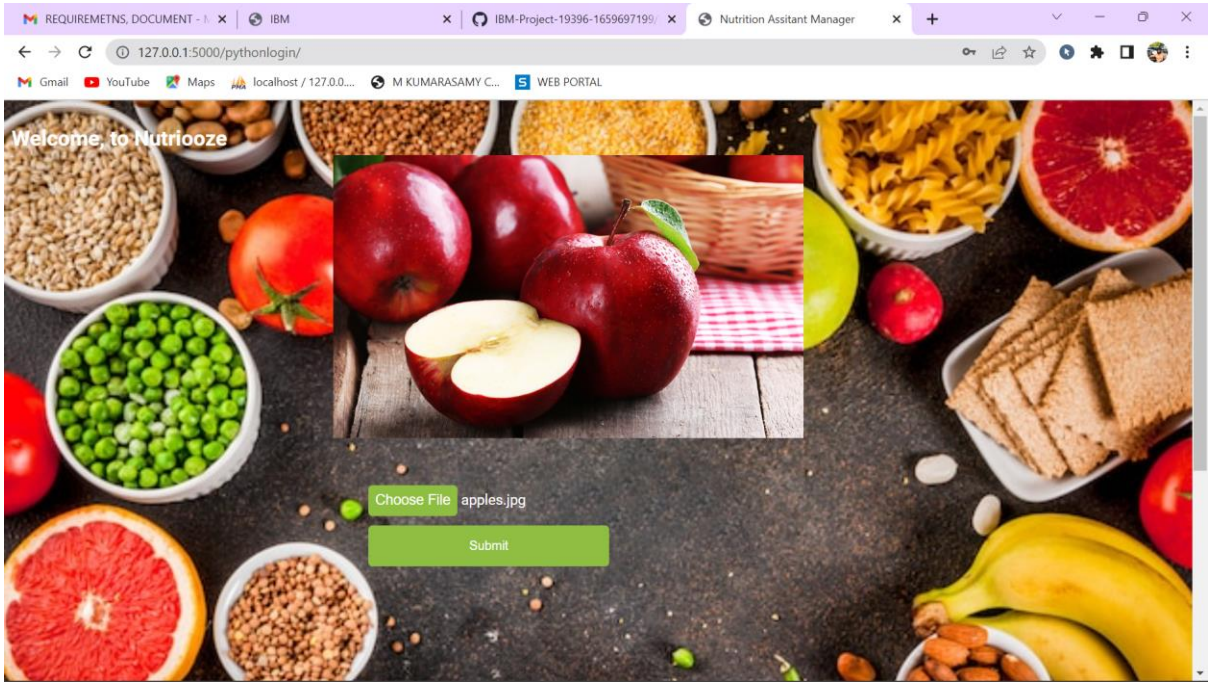
Login page:



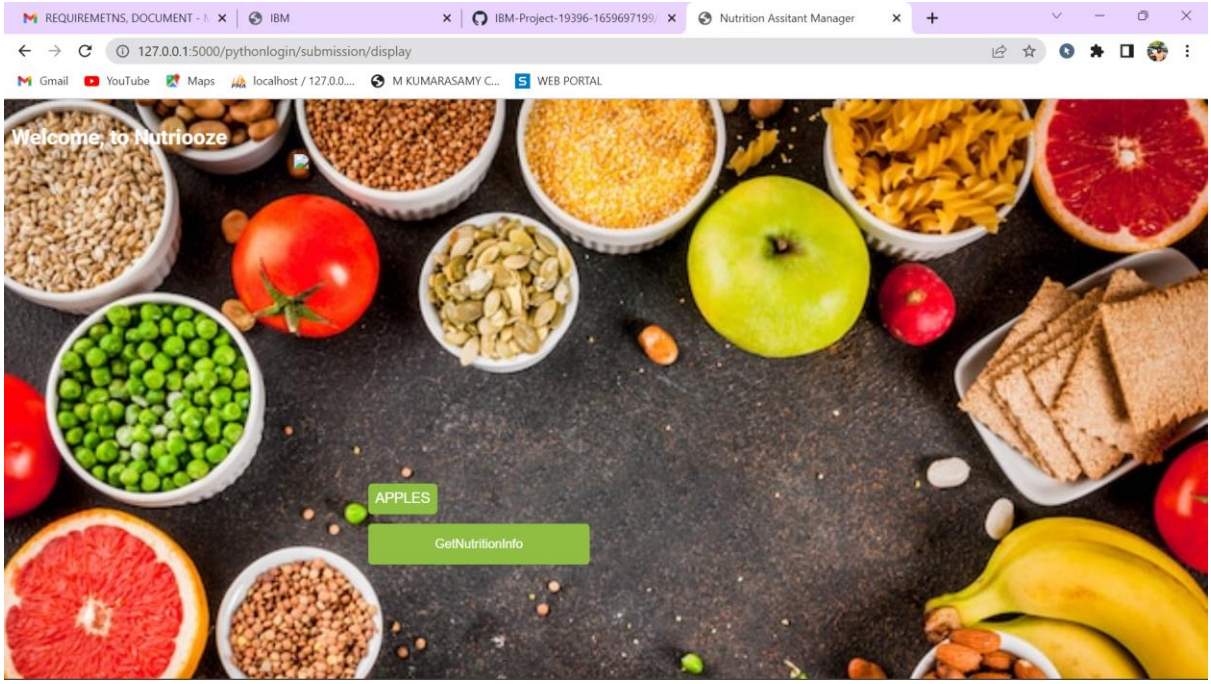
Register page:



Selecting the Items:



Getting Nutrition Info:



Resultant page: