Build python **Code**

Date	18 November 2022
Team ID	PNT2022TMID13764
Project Name	AI-powered Nutrition Analyzer for Fitness Enthusiasts

app.py

```
import
requests
from flask import Flask, render_template, request, url_for, redirect
from werkzeug.utils import secure_filename
from werkzeug.exceptions import HTTPException
import os
import json

UPLOAD_FOLDER = 'static/uploads/'
app = Flask(   name   , static_url_path='/')
app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER
my_secret = os.environ['apikey']

def demo_cal(num):
   if int(num)==1:
      data_load = "testdata2burger.json"
   else:
      data_load= "testdata.json"
   with open(data_load, "r") as f:
      data = json.load(f)
   return data

def get_cal(fname):
   try:
      img = f"static/uploads/{fname}"
      api_user_token = my_secret
```

```
headers = {'Authorization': 'Bearer ' + api_user_token}


# Single/Several Dishes Detection
```

```
url = 'https://api.logmeal.es/v2/recognition/complete'
                                    resp = requests.post(url,files={'image': open(img, 'rb')},headers=headers)
                                            print(resp.json())
                                            #print("response21:\n")
                                             # Nutritional information
                                                    url = 'https://api.logmeal.es/v2/recipe/nutritionalInfo'
                                    resp = requests.post(url,json={'imageId': resp.json()['imageId']}, headers=headers)
                                                    print(resp.json()) # display nutritional info
                                            return resp.json()
                                     except:
                                            return "Error"

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

                                     @app.route("/api")
                                     def testdata():
                                          data = demo_cal(1)
                                         return data

                                     @app.route("/demo/<num>")
                                    def demo(num):
                                          data = demo_cal(num)
                                          fname = "damplefood.jpg"
                                          if int(num)==1:
                                                  fname = "istockphoto-1125149183-612x612.jpg"
                                       else:
                                                  fname = "depositphotos_50523105-stock-photo-pizza-with-tomatoes.jpg"
                                        #print(num)
                                                return render_template("demo.html",fname=fname, data=data)

                                            @app.route('/result', methods = ['GET', 'POST'])
                                     def upload_file():
                    if request.method == 'POST':                    f =
                request.files['file']
                                                 fname = secure_filename(f.filename)
```

```
                f.save(os.path.join(app.config['UPLOAD_FOLDER'], fname))
            data = get_cal(fname)
            if data=="Error":
                return "Service has been exhausted please try after 24hrs!"
             an_object = data["foodName"]
                check_list = isinstance(an_object, list)
             if check_list==True:
                data["foodName"] = data["foodName"][0]
                 return render_template("result.html",fname=fname, data=data)
                    #return redirect(url_for('static', filename='uploads/' + fname), code=301)


         @app.errorhandler(HTTPException)
        def handle_exception(e):
            """Return JSON instead of HTML for HTTP errors."""
                # start with the correct headers and status code from the error
            response = e.get_response()
# replace the body with JSON          response.data = json.dumps({
    "code": e.code,              "name": e.name,
                "description": e.description,
        })
            response.content_type = "application/json"
        return response


    if__name__=="__main__":
            app.run(host="0.0.0.0", port=8000, debug=True)
```

main.yml

```
name: Build and deploy Python app to Azure Web App - foood

on:
  push:
    branches:
- main
  workflow_dispatch:

jobs:
  build:
```

```
    runs-on: ubuntu-latest

    steps:
- uses: actions/checkout@v2

- name: Set up Python version
        uses: actions/setup-python@v1
        with:
          python-version: '3.8'

- name: Create and start virtual environment
        run: |
          python -m venv venv
          source venv/bin/activate

- name: Install dependencies
        run: pip install -r requirements.txt

      # Optional: Add step to run tests here (PyTest, Django test suites, etc.)

- name: Upload artifact for deployment jobs
        uses:  actions/upload-artifact@v2
        with:
          name: python-app
          path: |
            .
            !venv/
  deploy:
    runs-on: ubuntu-latest
    needs: build
    environment:
      name: 'Production'
      url: ${{ steps.deploy-to-webapp.outputs.webapp-url }}

    steps:
- name: Download artifact from build job
        uses: actions/download-artifact@v2
        with:
          name: python-app
```

```
          path:
.
- name: 'Deploy to Azure Web App'
        uses: azure/webapps-deploy@v2
        id: deploy-to-webapp
        with:
          app-name: 'foood'
          slot-name: 'Production'
          publish-profile: ${{
secrets.AZUREAPPSERVICE_PUBLISHPROFILE_F6FCF510CE004208B6D1C454B08695A7 }}
```

Test


```
{
  "foodName":"pizza",
  "hasNutritionalInfo":true,
  "ids":168,
  "imageId":1330495,
  "nutritional_info":{
"calories":701.9,
    "dailyIntakeReference":{
      "CHOCDF":{
        "label":"Carbs",
        "level":"HIGH",
```

```
      "percent":44.990981165671165
    },
    "ENERC_KCAL":{
      "label":"Energy",
      "level":"NONE",
      "percent":34.10113830889581
    },
    "FASAT":{
      "label":"Saturated",
      "level":"HIGH",
      "percent":31.164453872938235
    },
    "FAT":{
      "label":"Fat",
      "level":"HIGH",
      "percent":38.023813771298215
    },
    "NA":{
      "label":"Sodium",
      "level":"HIGH",
      "percent":89.64
    },
    "PROCNT":{
      "label":"Protein",
      "level":"NONE",
      "percent":14.445654828102326
    },
    "SUGAR":{
      "label":"Sugars",
      "level":"MEDIUM",
      "percent":15.968000000000002
    }
  },
  "totalNutrients":{
    "CA":{
      "label":"Calcium",
      "quantity":181.65,
      "unit":"mg"
    },
    "CHOCDF":{
      "label":"Carbs",
      "quantity":104.18,
      "unit":"g"
    },
    "CHOLE":{
      "label":"Cholesterol",
      "quantity":22.4,
      "unit":"mg"
```

```
        },
        "ENERC_KCAL":{
          "label":"Energy",
          "quantity":701.9,
          "unit":"kcal"
        },
        "FAMS":{
          "label":"Monounsaturated fats",
          "quantity":12.05,
          "unit":"g"
        },
        "FAPU":{
          "label":"Polyunsaturated",
          "quantity":2.3,
          "unit":"g"
        },
        "FASAT":{
          "label":"Saturated",
          "quantity":5.88,
          "unit":"g"
        },
        "FAT":{
          "label":"Fat",
          "quantity":21.74,
          "unit":"g"
        },
        "FATRN":{
          "label":"Trans fat",
          "quantity":0.0,
          "unit":"g"
        },
        "FE":{
          "label":"Iron",
          "quantity":7.28,
          "unit":"mg"
        },
        "FIBTG":{
          "label":"Fiber",
"quantity":6.3,
          "unit":"g"
        },
        "FOLAC":{
          "label":"Folic acid",
          "quantity":192.5,
          "unit":"µg"
        },
        "FOLDFE":{
          "label":"Folate equivalent (total)",
```

```
    "quantity":470.7,
    "unit":"µg"
  },
  "FOLFD":{
    "label":"Folate (food)",
    "quantity":143.2,
    "unit":"µg"
  },
  "K":{
    "label":"Potassium",          "quantity":559.05,
    "unit":"mg"
  },
  "MG":{
    "label":"Magnesium",
    "quantity":54.04,
    "unit":"mg"
  },
  "NA":{
    "label":"Sodium",
    "quantity":1344.6,
    "unit":"mg"
  },
  "NIA":{
    "label":"Niacin (B3)",
    "quantity":10.24,
    "unit":"mg"
  },
  "P":{
    "label":"Phosphorus",
    "quantity":294.19,
    "unit":"mg"
  },
  "PROCNT":{
    "label":"Protein",
    "quantity":22.3,
    "unit":"g"
  },
  "RIBF":{
    "label":"Riboflavin (B2)",
    "quantity":0.94,
    "unit":"mg"
  },
  "SUGAR":{
    "label":"Sugars",
    "quantity":4.99,
    "unit":"g"
  },
  "SUGAR.added":{
```

```
            "label":"Sugars, added",
            "quantity":0.0,
            "unit":"g"
          },
          "THIA":{
            "label":"Thiamin (B1)",
            "quantity":1.46,
            "unit":"mg"
},
"TOCPHA":{
            "label":"Vitamin E",
            "quantity":3.83,
            "unit":"mg"
          },
          "VITA_RAE":{
            "label":"Vitamin A",
            "quantity":79.02,
            "unit":"µg"
          },
          "VITB12":{
            "label":"Vitamin B12",
            "quantity":0.65,
            "unit":"µg"
          },
          "VITB6A":{
            "label":"Vitamin B6",
            "quantity":0.25,
            "unit":"mg"
          },
          "VITC":{
            "label":"Vitamin C",
            "quantity":8.68,
            "unit":"mg"
          },
          "VITD":{
            "label":"Vitamin D",
            "quantity":4.65,
            "unit":"µg"
          },
          "VITK1":{
            "label":"Vitamin K",
            "quantity":14.67,
            "unit":"µg"
          },
          "ZN":{
            "label":"Zinc",
            "quantity":2.3,
            "unit":"mg"
```

```
        }
      }
    },
    "serving_size":295.35
}
```