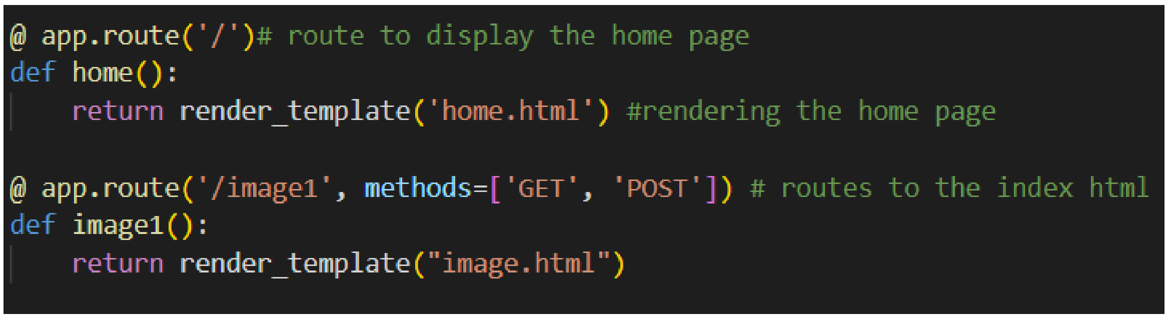
APPLICATION BUILDING

PROJECT TITLE: AI-powered Nutrition Analyzer for Fitness Enthusiasts

Team id: PNT2022TMID21516

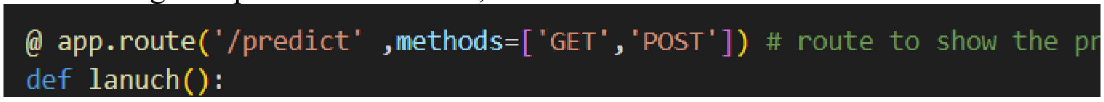
ROUTING TO THE HTML PAGE:

Here, the declared constructor is used to route to the HTML page created earlier. In the above example, the '/' URL is bound with the home.html function. Hence, when the home page of the webserver is opened in the browser, the HTML page is rendered. Whenever you enter the values from the HTML page the values can be retrieved using the POST Method. Here, "home.html" is rendered when the home button is clicked on the UI



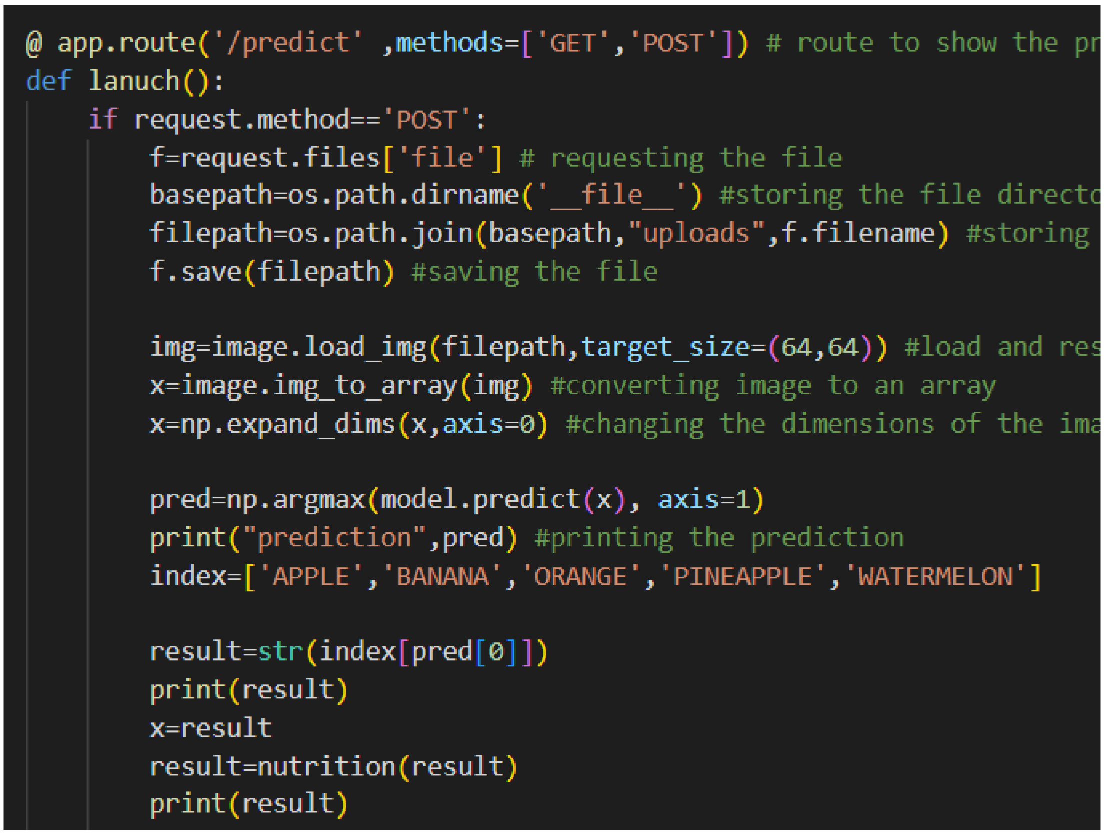
```
@ app.route('/')# route to display the home page
def home():
    return render_template('home.html') #rendering the home page

@ app.route('/image1', methods=['GET', 'POST']) # routes to the index html
def image1():
    return render_template("image.html")
```

When "image is uploaded "on the UI, the launch function is executed



```
 @ app.route('/predict' ,methods=['GET','POST']) # route to show the pr
 def lanuch():
```

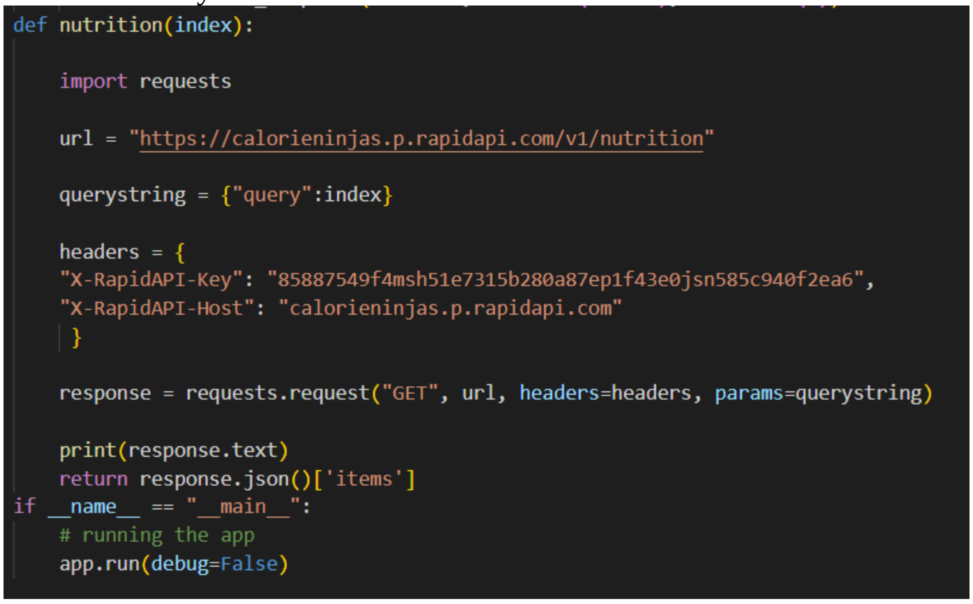
It will take the image request and we will be storing that image in our local system then we will convert the image into our required size and finally, we will be predicting the results with the help of our model which we trained and depending upon the class identified we will showcase the class name and its properties by rendering the respective html pages.



```
@ app.route('/predict' ,methods=['GET','POST']) # route to show the pr
def lanuch():
    if request.method=='POST':
        f=request.files['file'] # requesting the file
        basepath=os.path.dirname('__file__') #storing the file directo
        filepath=os.path.join(basepath,"uploads",f.filename) #storing
        f.save(filepath) #saving the file

        img=image.load_img(filepath,target_size=(64,64)) #load and res
        x=image.img_to_array(img) #converting image to an array
        x=np.expand_dims(x,axis=0) #changing the dimensions of the ima

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

        result=str(index[pred[0]])
        print(result)
        x=result
        result=nutrition(result)
        print(result)
```

API Integration:
Here we will be using Rapid API. Using RapidAPI, developers can search and test the APIs, subscribe, and connect to the APIs — all with a single account, single API key and single SDK. Engineering teams also use RapidAPI to share internal APIs and microservice documentation.

API used:
The link above will allow us to test the food item and will result the nutrition content present in the food item.
NOTE: When we keep hitting the API the limit of it might expire. So, making a smart use of it will be an efficient way.



```
def nutrition(index):

    import requests

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

    querystring = {"query":index}

    headers = {
    "X-RapidAPI-Key": "85887549f4msh51e7315b280a87ep1f43e0jsn585c940f2ea6",
    "X-RapidAPI-Host": "calorieninjas.p.rapidapi.com"
    }

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

    print(response.text)
    return response.json()['items']
if __name__ == "__main__":
    # running the app
    app.run(debug=False)
```

Finally, Run the application. This is used to run the application in a localhost. The local host runs on port number 5000.

CODE:

```
from flask import Flask,render_template,request
# Flask-It is our framework which we are going to use to run/serve our application.
#request-for accessing file which was uploaded by the user on our application.
import os
import numpy as np #used for numerical analysis
from tensorflow.keras.models import load_model#to load our trained model
from tensorflow.keras.preprocessing import image
import requests

app = Flask(__name__,template_folder="templates") #initializing a flask app
# Loading the model
model=load_model('nutrition.h5')
print("Loaded model from disk")

@ app.route('/')# route to display the home page
def home():
   return render_template('home.html') #rendering the home page

@ app.route('/image1', methods=['GET', 'POST']) # routes to the index html
def image1():
   return render_template("image.html")

@ app.route('/predict' ,methods=['GET','POST']) # route to show the predictions in a Web UI
def lanuch():
   if request.method=='POST':
      f=request.files['file'] # requesting the file
      basepath=os.path.dirname('__file__') #storing the file directory
      filepath=os.path.join(basepath,"uploads",f.filename) #storing the file in uploads folder
      f.save(filepath) #saving the file

      img=image.load_img(filepath,target_size=(64,64)) #load and reshaping the image
      x=image.img_to_array(img) #converting image to an array
      x=np.expand_dims(x,axis=0) #changing the dimensions of the image

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

      result=str(index[pred[0]])
      print(result)
      x=result
      result=nutrition(result)
      print(result)

      return render_template("0.html",showcase=(result),showcase1=(x))
def nutrition(index):
   import requests

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

   querystring = {"query":index}

   headers = {
   "X-RapidAPI-Key": "85887549f4msh51e7315b280a87ep1f43e0jsn585c940f2ea6",
   "X-RapidAPI-Host": "calorieninjas.p.rapidapi.com"
    }

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

   print(response.text)
   return response.json()['items']
if __name__ == "__main__":
   # running the app
   app.run(debug=False)
```