Team ID	**PNT2022TMID31776**
Project Name	**AI-powered Nutrition Analyzer forFitness Enthusiasts**

TESTING

```
import numpy as np
from tensorflow.keras.models
 import load_model
from tensorflow.keras.preprocessing import image
model=load_model('train.h5')
model=load_model('dataset.h5')
model=load_model('nutrition.h5')
img=image.load_img(r"/content/drive/MyDrive
Training/CNN/Dataset/TEST_SET/WATERMELON/3_100.jpg")
img
```

```
img=image.load_img(r"/content/drive/MyDrive
Training/CNN/Dataset/TEST_SET/WATERMELON/3_100.jpg",
target_size=(64,64))
img
```

```
x=image.img_to_array(img)
x
array([[[[255., 255., 255.],
        [255., 255., 255.],
```

[255., 255., 255.],
...,
[255., 255., 255.],
[255., 255., 255.],
[255., 255., 255.]],

[[255., 255., 255.],
[255., 255., 255.],
[255., 255., 255.],
...,
[255., 255., 255.],
[255., 255., 255.],
[255., 255., 255.]],

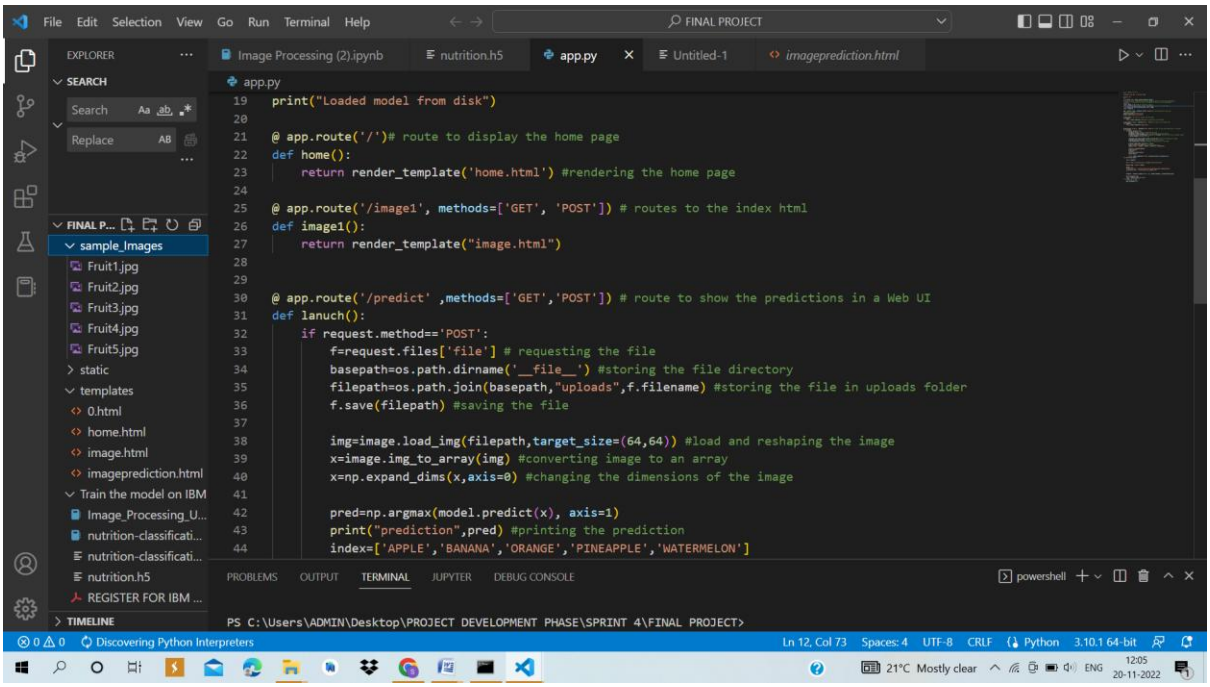
[[255., 255., 255.],
[255., 255., 255.],
[255., 255., 255.],
...,
[255., 255., 255.],
[255., 255., 255.],
[255., 255., 255.]],
...,

[[255., 255., 255.],
[255., 255., 255.],
[255., 255., 255.],
...,
[255., 255., 255.],
[255., 255., 255.],
[255., 255., 255.]],

```
  [[255., 255., 255.],
   [255., 255., 255.],
   [255., 255., 255.],
   ...,
   [255., 255., 255.],
   [255., 255., 255.],
   [255., 255., 255.]],

   [[255., 255., 255.],
   [255., 255., 255.],
   [255., 255., 255.],
   ...,
   [255., 255., 255.],
   [255., 255., 255.],
   [255., 255., 255.]]], dtype=float32)
x=np.expand_dims(x,axis=0)
  [[255., 255., 255.],
  [255., 255., 255.],
  [255., 255., 255.],
  ...,
  [255., 255., 255.],
  [255., 255., 255.],
  [255., 255., 255.]],

 [[255., 255., 255.],
  [255., 255., 255.],
  [255., 255., 255.],
  ...,
  [255., 255., 255.],
  [255., 255., 255.],
  [255., 255., 255.]],

 [[255., 255., 255.],
  [255., 255., 255.],
  [255., 255., 255.],
  ...,
```

```
       [255., 255., 255.],
       [255., 255., 255.],
       [255., 255., 255.]]], dtype=float32)
pred = model.predict
pred
array
([[0.25227112, 0.17414774, 0.15219809, 0.20493415, 0.21644896],
 [0.26760292, 0.1759095 , 0.15206912, 0.19424875, 0.21016978],
 [0.26474723, 0.165203  , 0.14452063, 0.20434381, 0.2211853 ],
       ...,
 [0.24550524, 0.1721549 , 0.16282505, 0.21065485, 0.20885986],
 [0.25395462, 0.1735253 , 0.16055605, 0.20655352, 0.20541045],
 [0.24495909, 0.15889102, 0.16927534, 0.20705006, 0.21982446]],
  dtype=float32
<bound method Model.predict of <keras.engine.
sequential.Sequential object at 0x7f94abfd7c10>>
predict_x=model.predict(x_test)
classes_x=np.argmax(predict_x,axis=1)
classes_x
array([0, 0, 0, ..., 0, 0, 0])
x_test.class_indices
index=['APPLE','BANANA','ORANGE','WATERMELON','PINEAPPLE']
result=str(index[classes_x[0]])
result
'Watermelon''
```

8.1 TEST CASES



8.2 USER ACCEPTANCE TESTING

User Acceptance Testing (UAT) is a type of testing performed by the end user or the client to verify/accept the software system before moving the software application to the production environment. UAT is done in the final phase of testing after functional, integration and system testing is done. The main Purpose of UAT is to validate end to end business flow. It does not focus on cosmetic errors, spelling mistakes or system testing. User Acceptance Testing is carried out in a separate testing environment with production-like data setup. It is a kind of black box testing where two or more end-users will be involved. Need of User Acceptance Testing arises once software has undergone Unit, Integration and System testing because developers might have built software based on requirements document by their own understanding and further required changes during development may not be effectively communicated to them, so for testing whether the final product is accepted by client/end-user, user acceptance testing is needed.

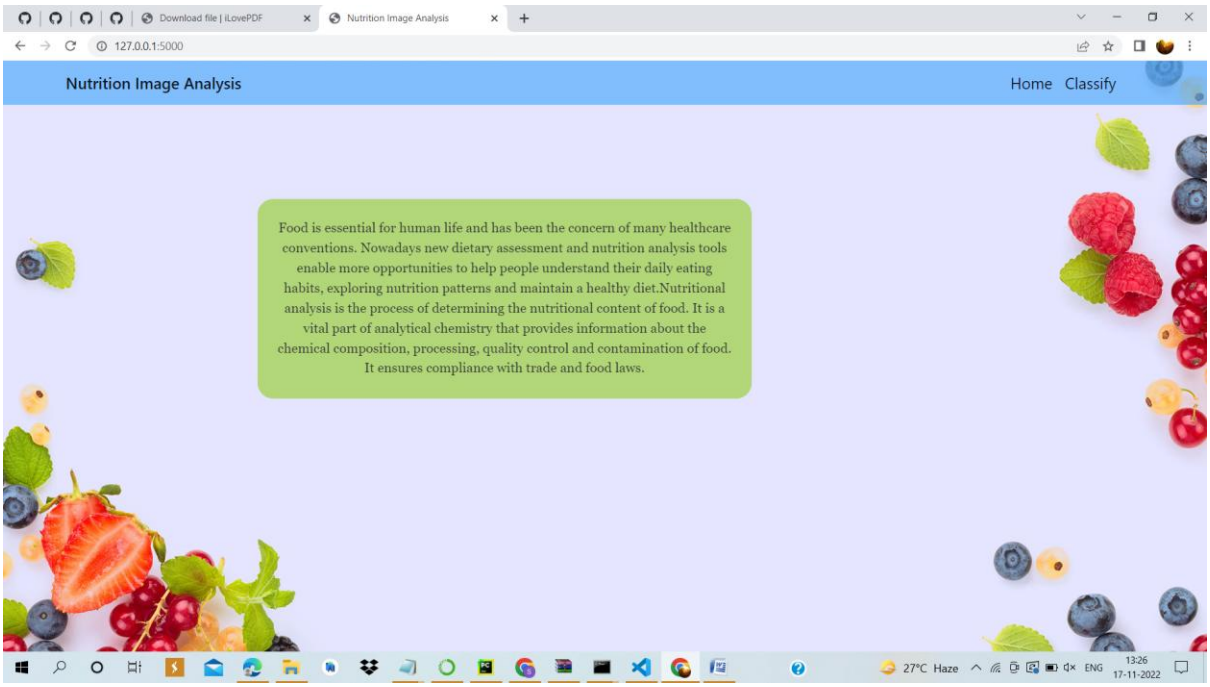
APPLE BANANA ORANGE PINEAPPLE WATERMELON

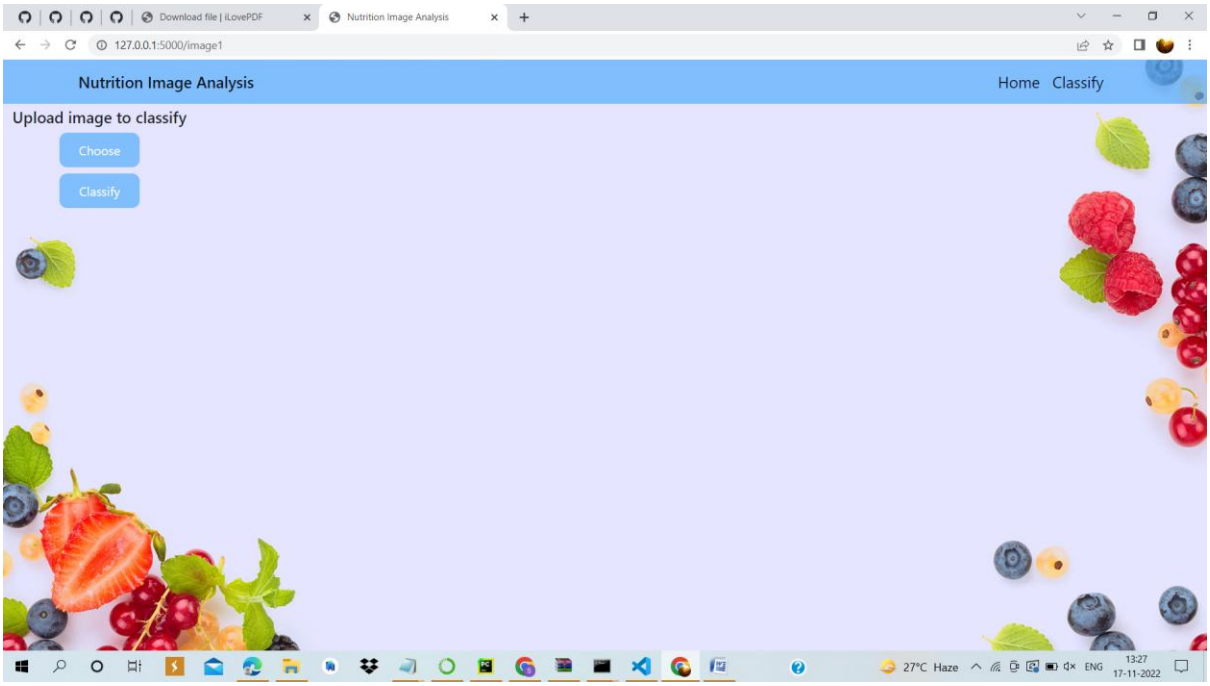
TEST_IMAGE1 TEST_IMAGE2 TEST_IMAGE3 TEST_IMAGE4 TEST_IMAGE5

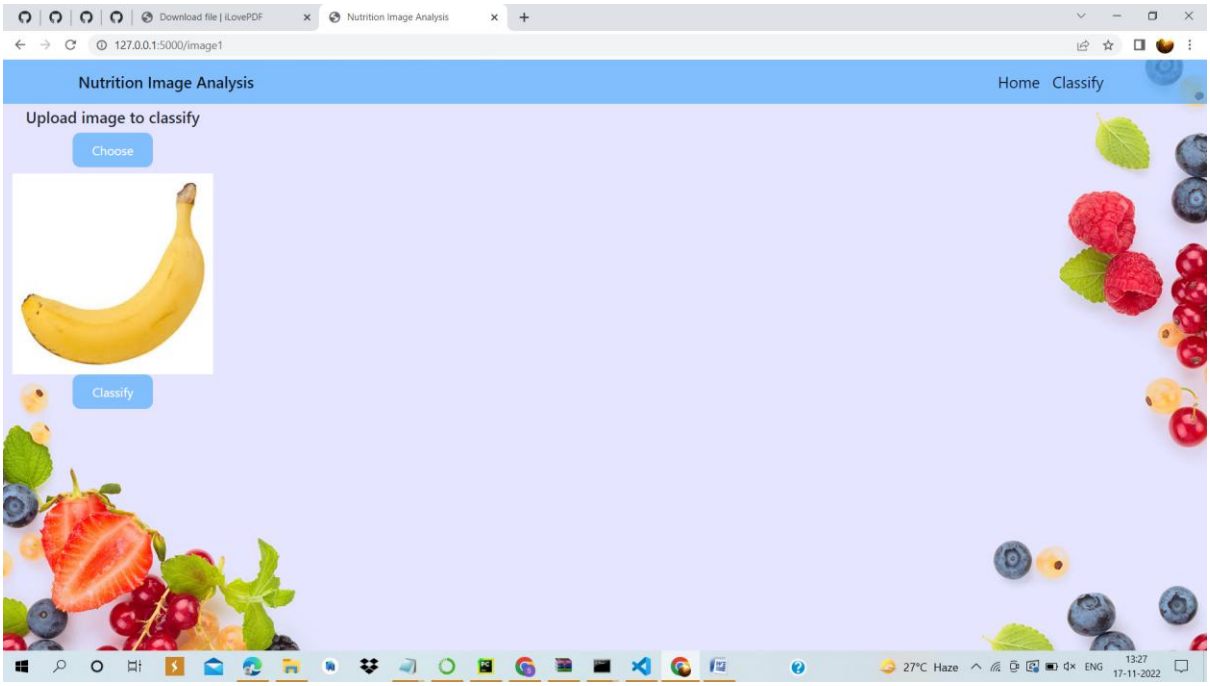
PERFORMANCE TESTING

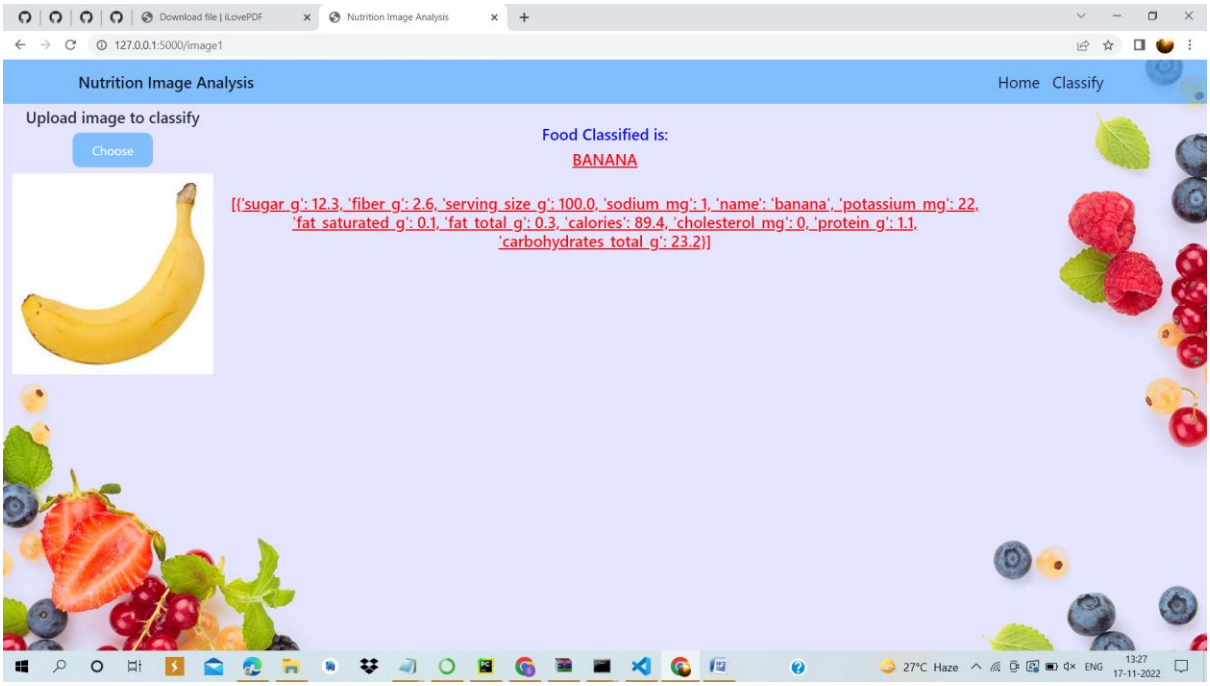
/tmp/wsuser/ipykernel_165/2706448856.py:2: UserWarning: `Model.fit_generator` is deprecated and will be removed in a future version. Please use `Model.fit`, which supports generators.
  classifier.fit_generator(
Epoch 1/20
828/828 [==============================] - 56s 67ms/step - loss: 0.6839 - accuracy: 0.7255 - val_loss: 0.4220 - val_accuracy: 0.8308
Epoch 2/20
828/828 [==============================] - 54s 66ms/step - loss: 0.4299 - accuracy: 0.8366 - val_loss: 0.3379 - val_accuracy: 0.8712
Epoch 3/20
828/828 [==============================] - 55s 66ms/step - loss: 0.3900 - accuracy: 0.8523 - val_loss: 0.3233 - val_accuracy: 0.8697
Epoch 4/20
828/828 [==============================] - 55s 66ms/step - loss: 0.3700 - accuracy: 0.8596 - val_loss: 0.3347 - val_accuracy: 0.8775
Epoch 5/20
828/828 [==============================] - 55s 66ms/step - loss: 0.3381 - accuracy: 0.8748 - val_loss: 0.3260 - val_accuracy: 0.8719
Epoch 6/20
828/828 [==============================] - 54s 65ms/step - loss: 0.3316 - accuracy: 0.8731 - val_loss: 0.2455 - val_accuracy: 0.9108
Epoch 7/20
828/828 [==============================] - 55s 66ms/step - loss: 0.3080 - accuracy: 0.8821 - val_loss: 0.2553 - val_accuracy: 0.9072
Epoch 8/20
828/828 [==============================] - 54s 65ms/step - loss: 0.2942 - accuracy: 0.8891 - val_loss: 0.2722 - val_accuracy: 0.8990
Epoch 9/20
828/828 [==============================] - 54s 66ms/step - loss: 0.2709 - accuracy: 0.8983 - val_loss: 0.2202 - val_accuracy: 0.9176
Epoch 10/20

828/828 [==============================] - 54s 65ms/step - loss: 0.2648 - accuracy: 0.8949 - val_loss: 0.2655 - val_accuracy: 0.8946
Epoch 11/20
828/828 [==============================] - 54s 66ms/step - loss: 0.2417 - accuracy: 0.9101 - val_loss: 0.2126 - val_accuracy: 0.9174
Epoch 12/20
828/828 [==============================] - 53s 65ms/step - loss: 0.2282 - accuracy: 0.9130 - val_loss: 0.2247 - val_accuracy: 0.9108
Epoch 13/20
828/828 [==============================] - 52s 63ms/step - loss: 0.2246 - accuracy: 0.9132 - val_loss: 0.2408 - val_accuracy: 0.9070
Epoch 14/20
828/828 [==============================] - 54s 65ms/step - loss: 0.2162 - accuracy: 0.9176 - val_loss: 0.1503 - val_accuracy: 0.9454
Epoch 15/20
828/828 [==============================] - 54s 65ms/step - loss: 0.1903 - accuracy: 0.9261 - val_loss: 0.1458 - val_accuracy: 0.9420
Epoch 16/20
828/828 [==============================] - 54s 65ms/step - loss: 0.1776 - accuracy: 0.9316 - val_loss: 0.1147 - val_accuracy: 0.9526
Epoch 17/20
828/828 [==============================] - 54s 65ms/step - loss: 0.1696 - accuracy: 0.9357 - val_loss: 0.1248 - val_accuracy: 0.9553
Epoch 18/20
828/828 [==============================] - 54s 65ms/step - loss: 0.1477 - accuracy: 0.9442 - val_loss: 0.1842 - val_accuracy: 0.9321
Epoch 19/20
349/828 [===========>.....................] - ETA: 24s - loss: 0.1653 - accuracy: 0.9323




Download file | iLovePDF
Nutrition Image Analysis
127.0.0.1:5000/image1
Nutrition Image Analysis
Home Classify
Upload image to classify
Choose
Classify
27°C Haze
ENG
13:27
17-11-2022


Download file | iLovePDF
Nutrition Image Analysis
127.0.0.1:5000/image1
Nutrition Image Analysis
Home Classify
Upload image to classify
Choose
Classify
27°C Haze
ENG
13:27
17-11-2022


Download file | iLovePDF
Nutrition Image Analysis
127.0.0.1:5000/image1
Nutrition Image Analysis
Home Classify
Upload image to classify
Choose
Food Classified is:
BANANA
[{'sugar_g': 12.3, 'fiber_g': 2.6, 'serving_size_g': 100.0, 'sodium_mg': 1, 'name': 'banana', 'potassium_mg': 22, 'fat_saturated_g': 0.1, 'fat_total_g': 0.3, 'calories': 89.4, 'cholesterol_mg': 0, 'protein_g': 1.1, 'carbohydrates_total_g': 23.2}]
27°C Haze
ENG
13:27
17-11-2022