Final Code:

```
import pandas as pd

import cv2

import os

import numpy as np

import random

import pickle

import h5py as h5

train_data='/content/contentdriveMyDriveTRAIN_SET/MyDrive/TRAIN_SET'

test_data='/content/drive/MyDrive/TEST_SET-20221117T140252Z-001'

from keras.preprocessing.image import ImageDataGenerator

x_train =
train_datagen.flow_from_directory('/content/drive/MyDrive/TRAIN_SET',target_size=(64,64),batch_
size=5,color_mode='rgb',class_mode='sparse')

x_test = test_datagen.flow_from_directory('/content/drive/MyDrive/TEST_SET-20221117T140252Z-
001',target_size=(64,64),batch_size=5,color_mode='rgb',class_mode='sparse')

print(x_train.class_indices)

print(x_test.class_indices)

from collections import Counter as c

c(x_train .labels)

import numpy as np

import tensorflow

from tensorflow.keras.models import Sequential

from tensorflow.keras import layers

from tensorflow.keras.layers import Dense,Flatten

from tensorflow.keras.layers import Conv2D,MaxPooling2D,Dropout

from keras.preprocessing.image import ImageDataGenerator


model=Sequential()

classifier = Sequential()

classifier.add(Conv2D(32,(3,3), input_shape=(64,64,3), activation='relu'))
```

```
classifier.add(MaxPooling2D(pool_size=(2,2)))

classifier.add(Conv2D(32,(3,3),activation='relu'))

classifier.add(MaxPooling2D(pool_size=(2,2)))

classifier.add(Flatten())

classifier.summary()

classifier.compile(optimizer='adam', loss='sparse_categorical_crossentropy', metrics=['accuracy'])

classifier.fit_generator(generator=x_train,steps_per_epoch = len(x_train), epochs=20,
validation_data=x_test, validation_steps=len(x_test))

classifier.save('nutrition.h5')

from tensorflow.keras.models import load_model

from keras.preprocessing import image

model = load_model("nutrition.h5")

from tensorflow.keras.preprocessing import imagea

img = image.load_img('/content/drive/MyDrive/TEST_SET-20221117T140252Z-
001/TEST_SET/APPLES/152_100.jpg',grayscale=False,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)

pred

index=['APPLES', 'BANANA', 'ORANGE', 'PINEAPPLE', 'WATERMELON']

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

result
```