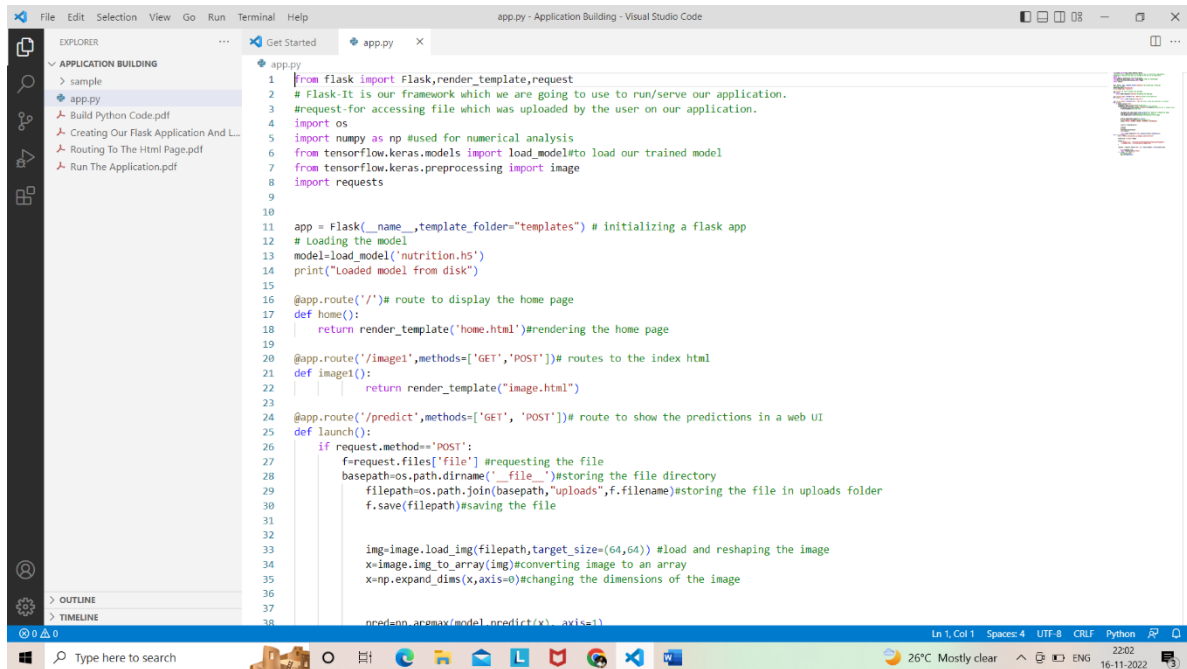


# Creating Our Flask Application And Loading Our Model By Using Load\_model Method

Creating our flask application and loading our model by using the load\_model method



The screenshot displays the Visual Studio Code interface with a Python file named `app.py` open. The code implements a Flask web application that loads a pre-trained Keras model and handles image uploads for prediction.

```
1 from flask import Flask, render_template, request
2 # Flask-It is our framework which we are going to use to run/serve our application.
3 #request-for accessing file which was uploaded by the user on our application.
4 import os
5 import numpy as np #used for numerical analysis
6 from tensorflow.keras.models import load_model #to load our trained model
7 from tensorflow.keras.preprocessing import image
8 import requests
9
10
11 app = Flask(__name__, template_folder="templates") # initializing a flask app
12 # Loading the model
13 model=load_model('nutrition.h5')
14 print("Loaded model from disk")
15
16 @app.route('/')# route to display the home page
17 def home():
18     return render_template('home.html')#rendering the home page
19
20 @app.route('/image1', methods=['GET', 'POST'])# routes to the index html
21 def image1():
22     return render_template("image.html")
23
24 @app.route('/predict', methods=['GET', 'POST'])# route to show the predictions in a web UI
25 def launch():
26     if request.method=='POST':
27         f=request.files['file'] #requesting the file
28         basepath=os.path.dirname('__file__')#storing the file directory
29         filepath=os.path.join(basepath, 'uploads', f.filename)#storing the file in uploads folder
30         f.save(filepath)#saving the file
31
32
33         img=image.load_img(filepath, target_size=(64,64)) #load and reshaping the image
34         x=image.img_to_array(img)#converting image to an array
35         x=np.expand_dims(x, axis=0)#changing the dimensions of the image
36
37
38         pred=np.argmax(model.predict(x), axis=-1)
```