|  |
| --- |
| Team ID : LTVIP2025TMID32402 |
| Project Name : Enchanted Wings: Marvels of Butterfly Species |

**🔹 index.html — Home Page with Upload Form**

**html**

**CopyEdit**

**<!DOCTYPE html>**

**<html>**

**<head>**

**<title>Fruit/Veggie Classifier</title>**

**</head>**

**<body>**

**<h1>Upload an Image for Classification</h1>**

**<form action="/inspect" method="POST" enctype="multipart/form-data">**

**<input type="file" name="image" accept="image/\*" required><br><br>**

**<button type="submit">Inspect</button>**

**</form>**

**</body>**

**</html>**

**🔹 inspect.html — Preview Uploaded Image & Go to Prediction**

**html**

**CopyEdit**

**<!DOCTYPE html>**

**<html>**

**<head>**

**<title>Image Preview</title>**

**</head>**

**<body>**

**<h1>Image Preview</h1>**

**<img src="{{ image\_path }}" alt="Uploaded Image" width="300"><br><br>**

**<form action="/predict" method="POST">**

**<input type="hidden" name="image\_path" value="{{ image\_path }}">**

**<button type="submit">Predict</button>**

**</form>**

**</body>**

**</html>**

**🔹 inner.html — Display Prediction Output**

**html**

**CopyEdit**

**<!DOCTYPE html>**

**<html>**

**<head>**

**<title>Prediction Result</title>**

**</head>**

**<body>**

**<h1>Prediction Result</h1>**

**<img src="{{ image\_path }}" alt="Input Image" width="300"><br><br>**

**<h2>Prediction: {{ label }}</h2>**

**<h3>Class ID: {{ class\_id }}</h3>**

**</body>**

**</html>**

**🖥️ Flask Backend Example (app.py)**

**You can use this simple Flask backend:**

**python**

**CopyEdit**

**from flask import Flask, render\_template, request**

**from werkzeug.utils import secure\_filename**

**import os**

**import random**

**app = Flask(\_\_name\_\_)**

**UPLOAD\_FOLDER = 'static/uploads'**

**os.makedirs(UPLOAD\_FOLDER, exist\_ok=True)**

**# Dummy labels**

**classes = {**

**0: 'Apple\_healthy',**

**23: 'Potato\_rotten',**

**24: 'Strawberry\_healthy',**

**26: 'Tomato\_healthy',**

**27: 'Tomato\_rotten'**

**}**

**@app.route('/')**

**def index():**

**return render\_template('index.html')**

**@app.route('/inspect', methods=['POST'])**

**def inspect():**

**image = request.files['image']**

**filename = secure\_filename(image.filename)**

**filepath = os.path.join(UPLOAD\_FOLDER, filename)**

**image.save(filepath)**

**return render\_template('inspect.html', image\_path='/' + filepath)**

**@app.route('/predict', methods=['POST'])**

**def predict():**

**image\_path = request.form['image\_path']**

**# Dummy model prediction**

**class\_id = random.choice(list(classes.keys()))**

**label = classes[class\_id]**

**return render\_template('inner.html', image\_path=image\_path, label=label, class\_id=class\_id)**

**if \_\_name\_\_ == '\_\_main\_\_':**

**app.run(debug=True)**

**Let me know if you want:**

* **Real model integration using TensorFlow or PyTorch**
* **Bootstrap-based UI design**
* **Code packaged into a zip file for download**

**Ask ChatGPT**

**Building HTML Pages:**

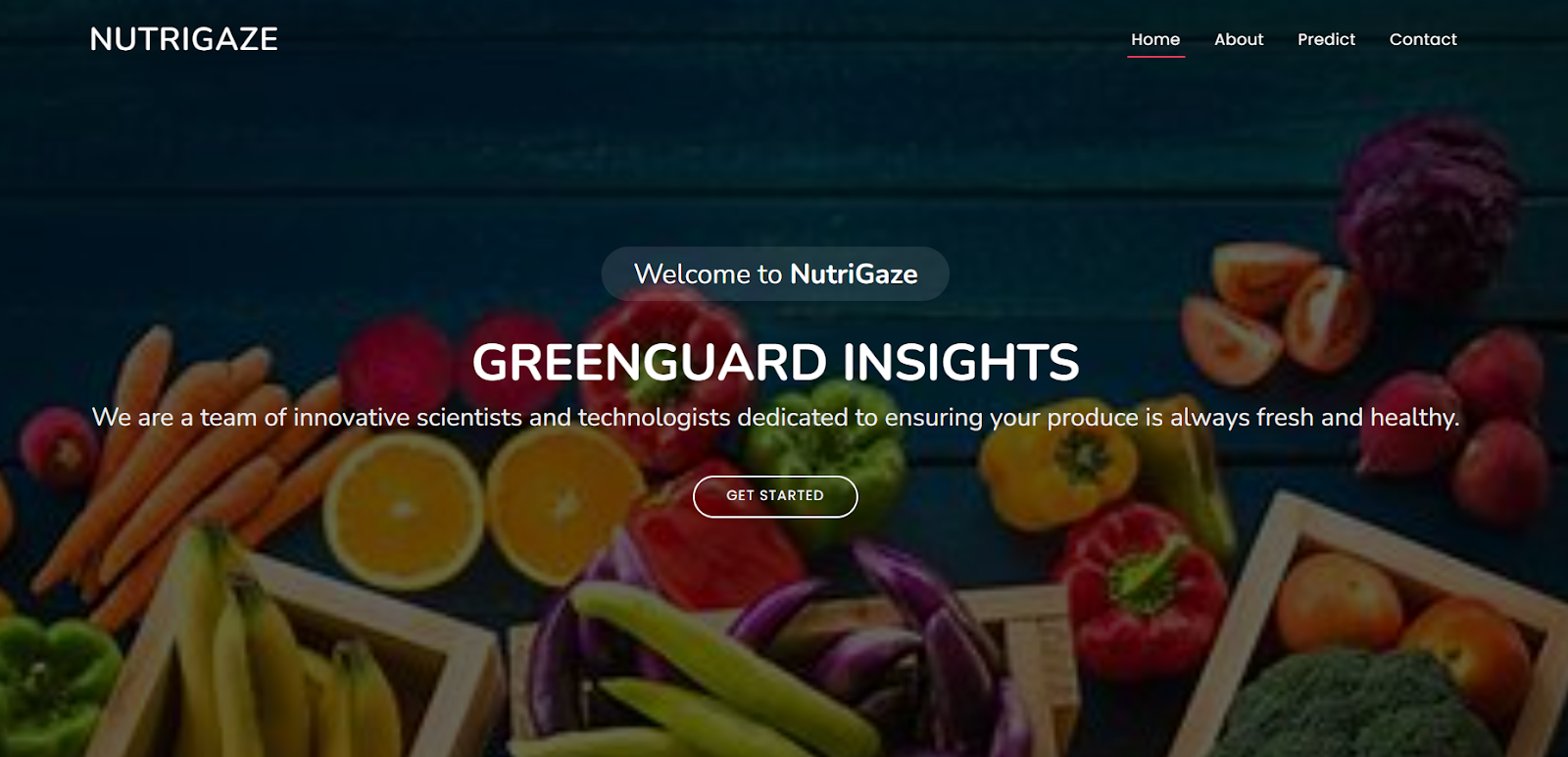
For this project create three HTML files namely

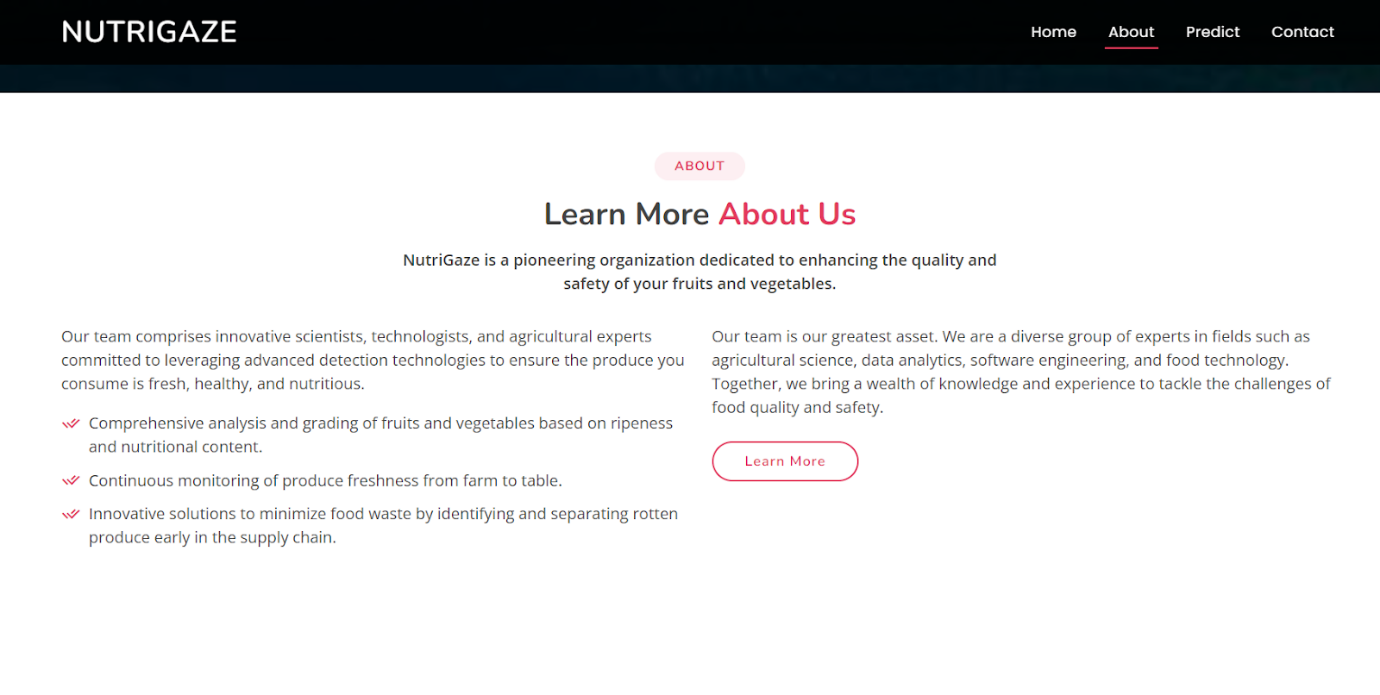
* index.html

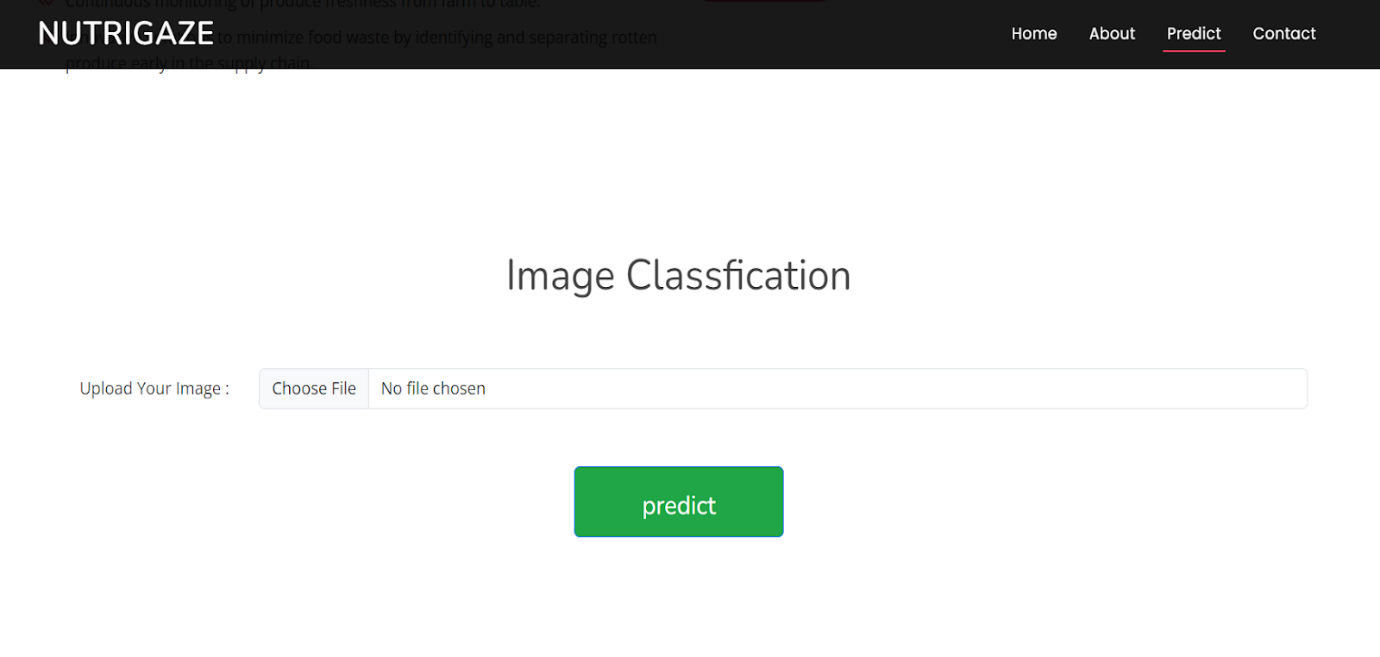
And save them in the templates folder.

UI Image preview:

Let’s see what our index.html page looks like:



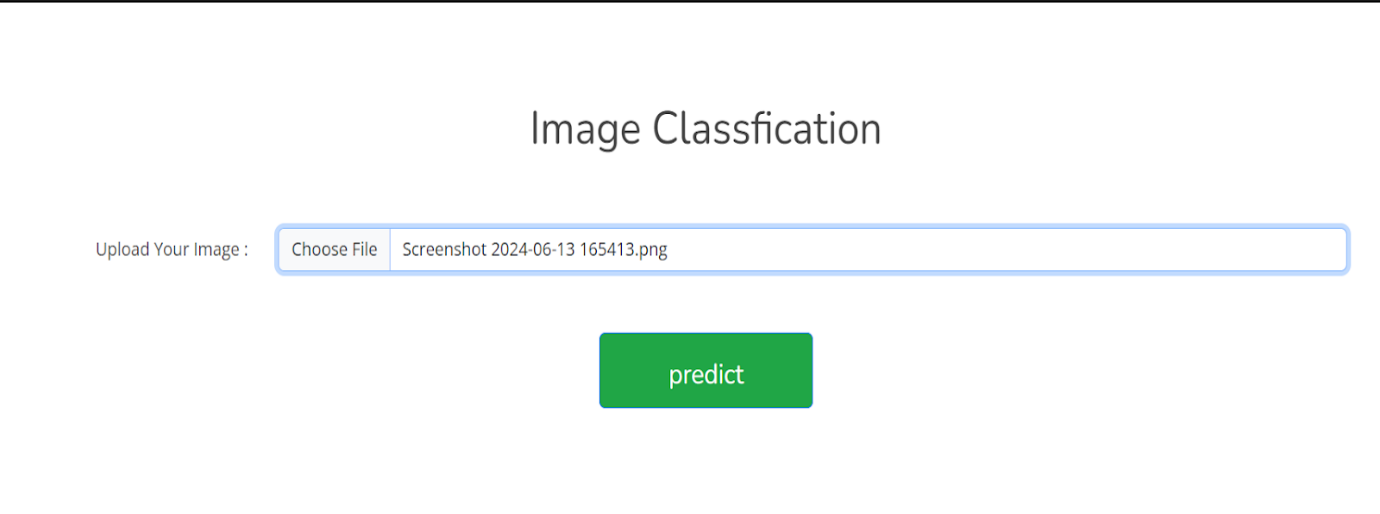




Now when you click on the inspect button further in the top right corner you will get redirected to Inspect.html

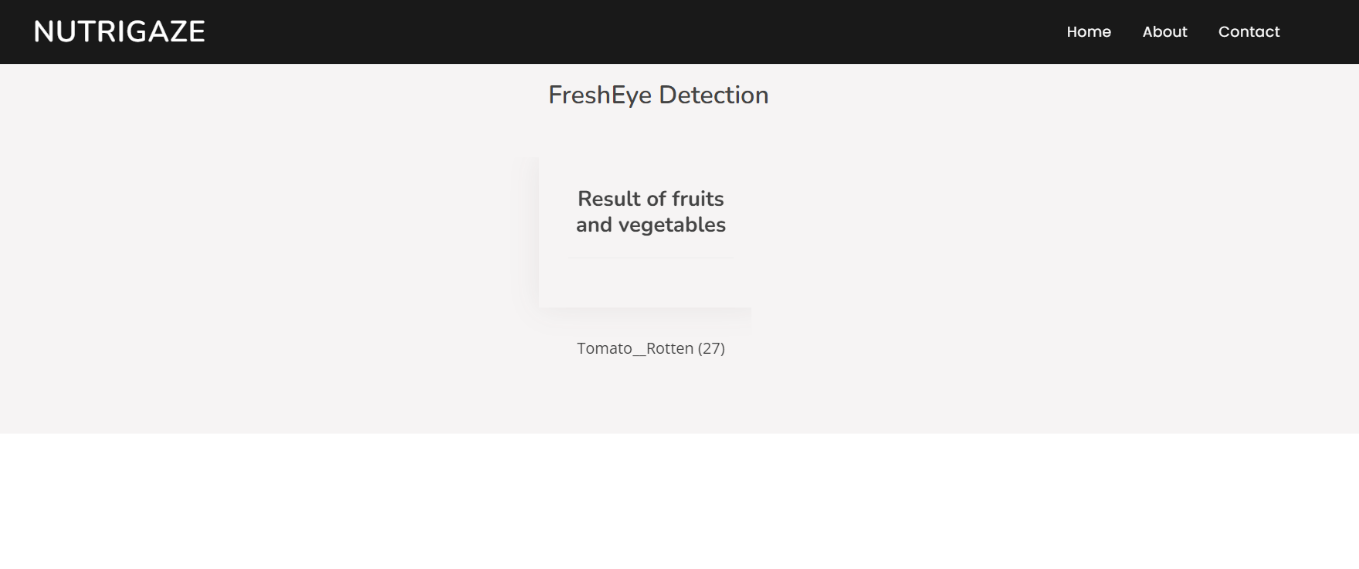
Let’s look at what our inner.html file looks like and test the model:

Test Class tomato rotten (27):

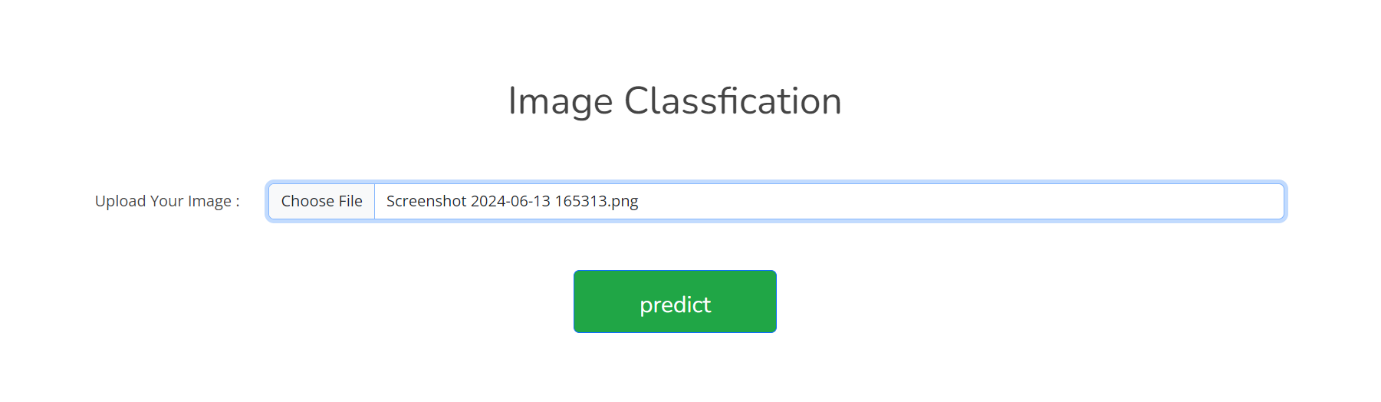


Now when you click on the predict button you will get output down to the image itself

Let’s look at what our output looks like

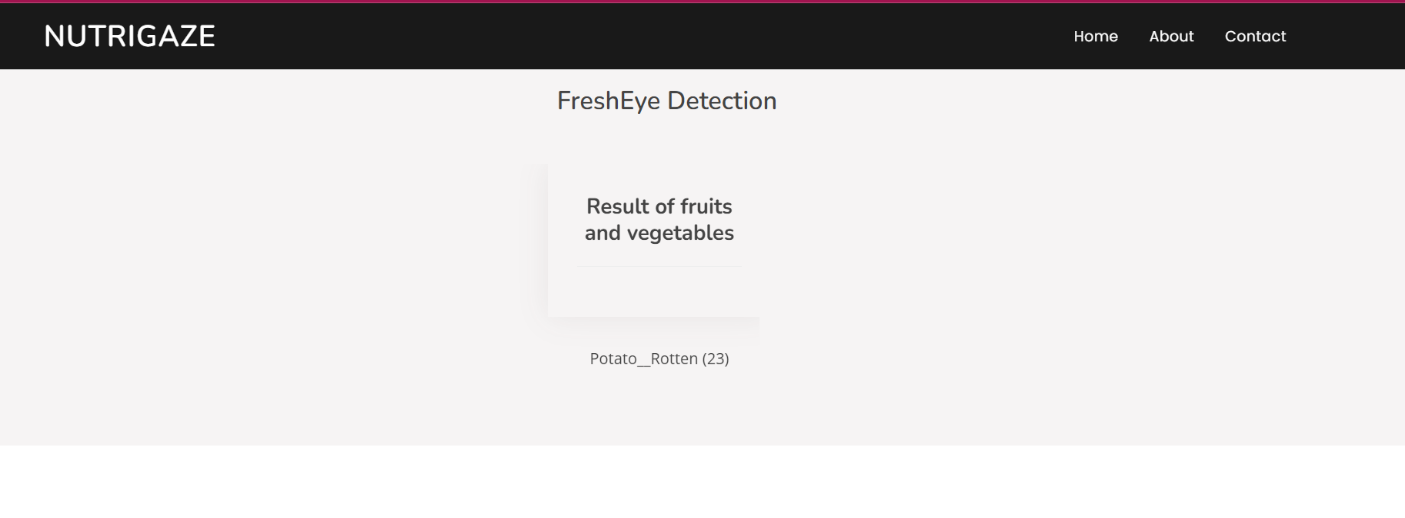


Test Class potato rotten (23):

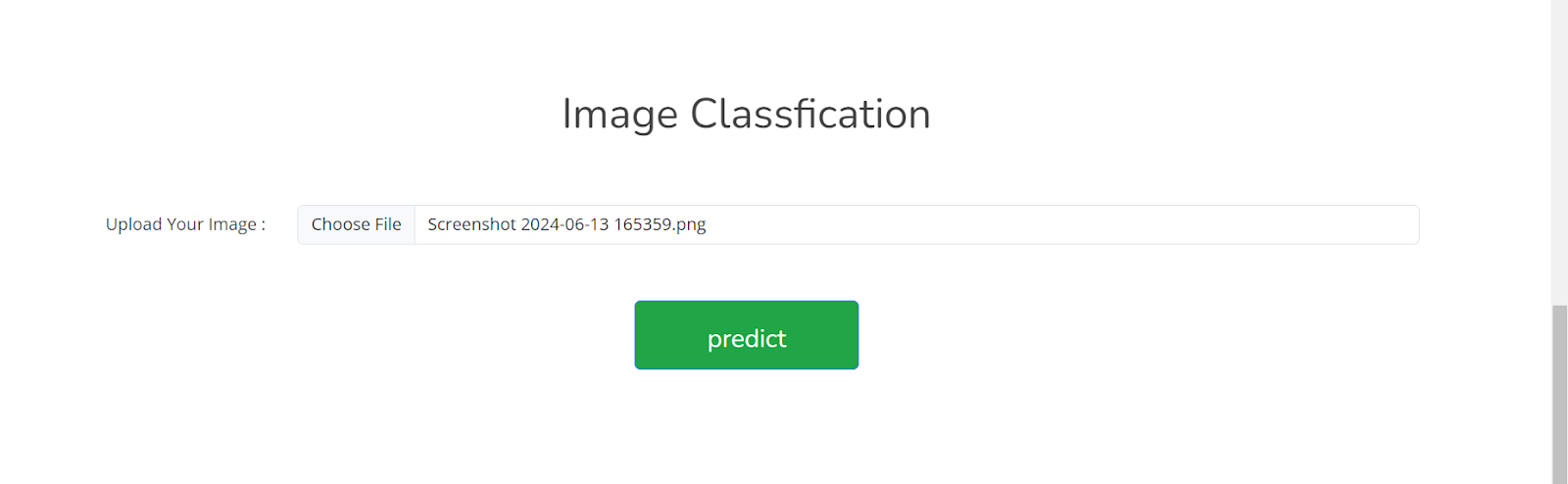


Now when you click on the predict button you will get output down to the image itself

Let’s look how our output looks like:

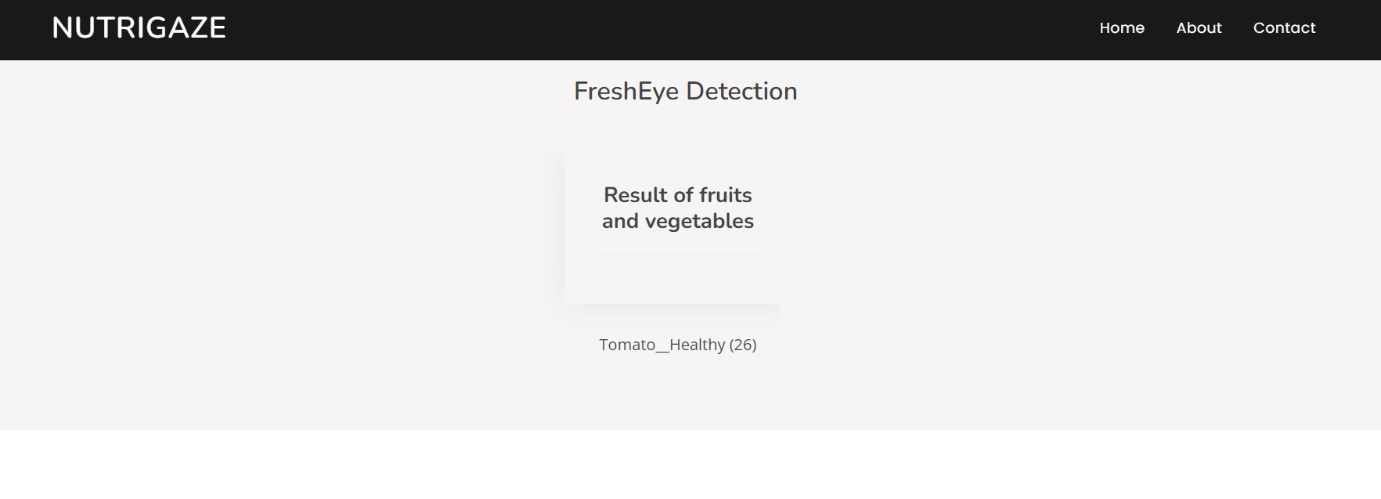


Test Class tomato healthy (26):

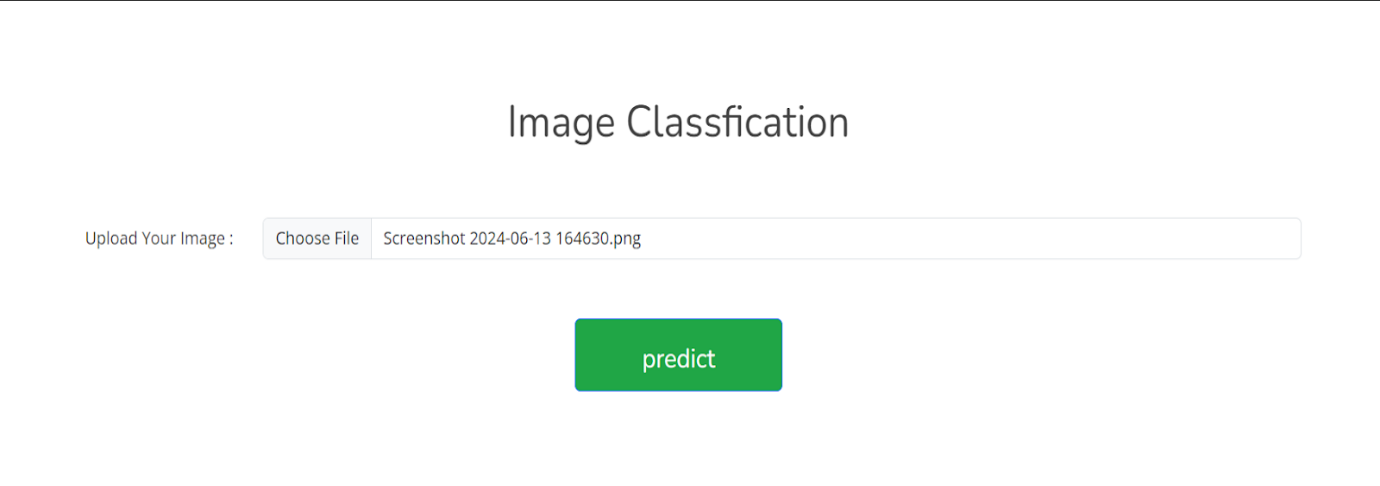


Now when you click on predict button you will get output down to the image itself

Let’s look how our output looks like:

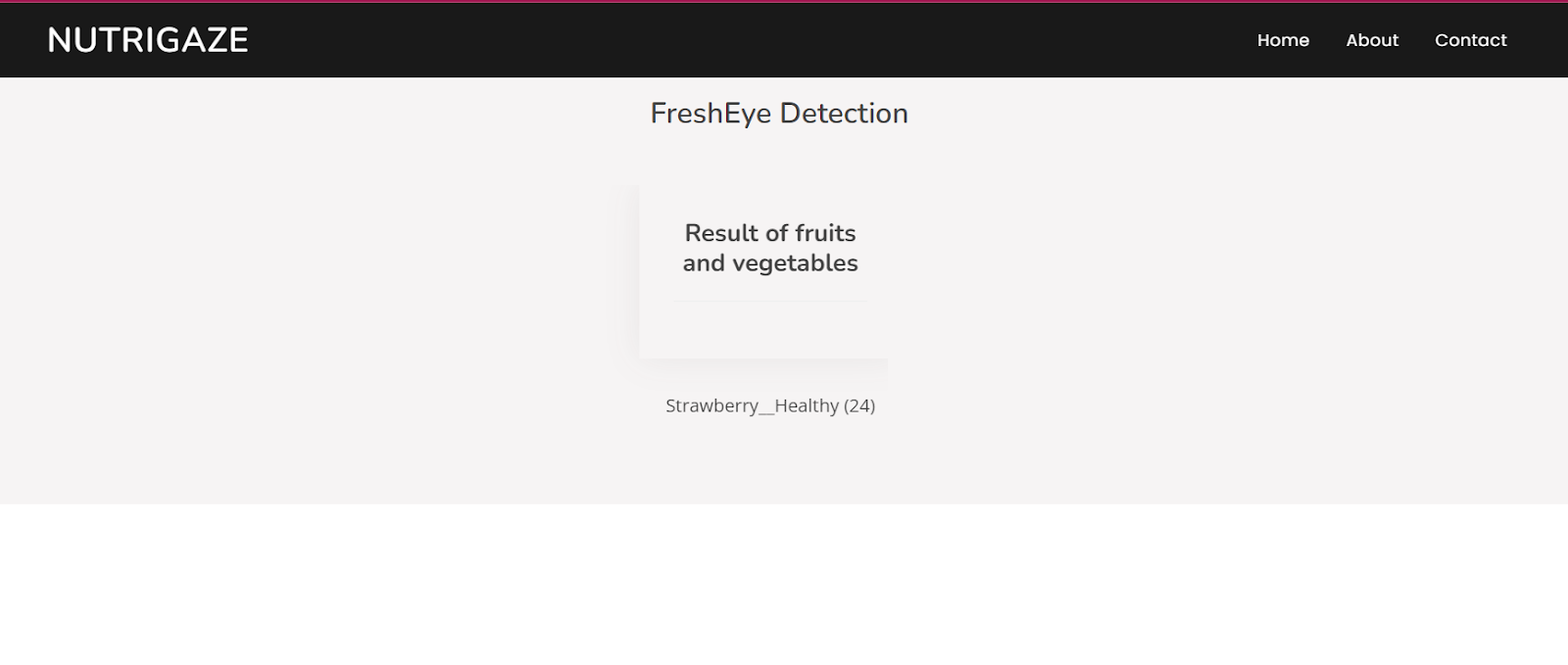


Test Class strawberry\_healthy (24):

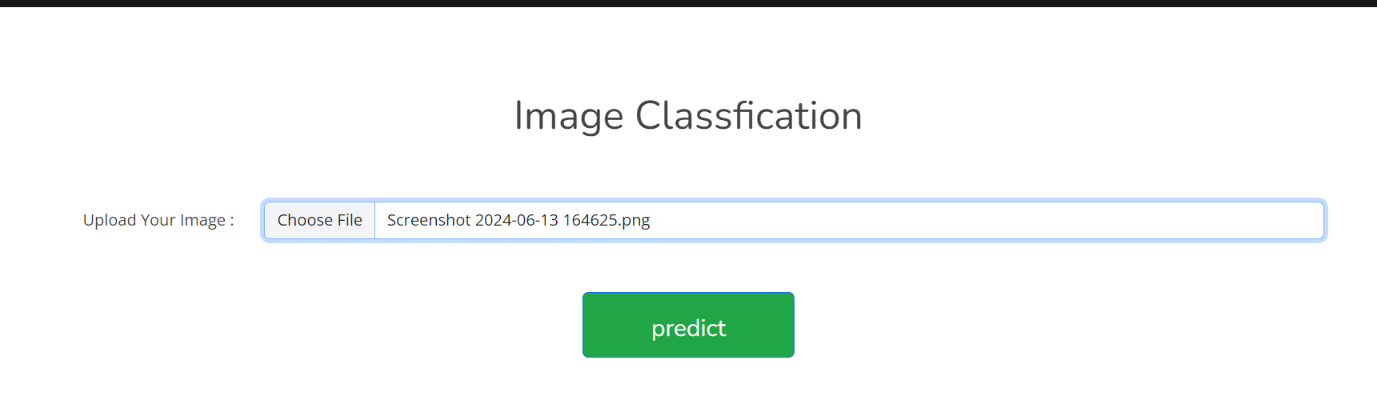


Now when you click on predict button you will get output down to the image itself

Let’s look how our output looks like:



Test Class Apple\_healthy (0):



Now when you click on predict button you will get output down to the image itself

Let’s look how our output looks like:

