**Project Phases Template**

**Project Title:**

**Smart Sorting: Transfer Learning for Identifying Rotten Fruits and Vegetables**

Team Lead:

* Chalamalasetti Dharani

Team members:

* Bhanu Prakash
* Hari nayak

Phase -1

**Brainstorming and Ideation:**

**Objective:**

**Problem Statement**

Many people struggle to identify whether fruits and vegetables are fresh (healthy) or spoiled (rotten) just by looking. This leads to:

* Wastage of good produce
* Eating spoiled items by mistake
* Difficulty in sorting large amounts of produce in markets, stores, or homes

There is a need for a smart and easy system that can automatically check and tell if a fruit or vegetable is healthy or rotten just from an image**.**

**Purpose of the Project**

* To help users quickly identify whether fruits or vegetables are healthy or rotten using AI and images.
* To reduce food waste by detecting spoiled items early.
* To make sorting easier for farmers, shopkeepers, households, and food businesses.

**Impact of the Project**

* Minimizes food wastage by preventing spoiled produce from mixing with good ones.
* Helps in better quality control in markets and stores.
* Supports farmers in grading their produce accurately.
* Increases consumer trust by improving food safety and freshness.
* Encourages digital agriculture and smart food practices.

**Empathy map**

| **DOES** | **FEELS** |
| --- | --- |
| - Picks and checks items by hand. | - Frustrated with manual sorting. |
| - Uses eyesight or smell to judge. | - Anxious about customer complaints. |
| - Sells at local markets. | - Hopeful for a simple tech solution. |

| **SAYS** | **THINKS** |
| --- | --- |
| “I want to sell fresh produce.” | “Will people trust my product’s quality?” |
| “It’s hard to check everything manually.” | “I wish there was an easier way to sort good and bad items.” |
| “I lose money when produce rots.” | “Rotten fruits damage my reputation.” |

**Brainstorming Ideas**

* Use AI to classify fruits/vegetables as *healthy* or *rotten*.
* Create a mobile/web interface for uploading images.
* Support real-time predictions using a camera or file upload.
* Use VGG16 or other CNN models with transfer learning.
* Build a clean, simple UI for farmers and vendors.
* Integrate multilingual support for rural users (future).
* Add a report or history feature (future enhancement).
* Make it lightweight and fast for low-end devices.

**Phase-2**

**Requirement Analysis:**

Customer Journey Map (Farmer/Vendor Perspective)

| Step | Experience | User Goal |
| --- | --- | --- |
| Visit Website | Loads Smart Sorting system | Wants to easily check quality |
| Upload Image | Uploads fruit/vegetable photo | Wants instant result |
| Prediction Display | Sees if it’s Healthy or Rotten | Make a quick decision to keep or remove |
| Take Action | Sort items accordingly | Reduce waste and maintain good quality |

**Solution Requirements:**

**Functional Requirements:**

* Upload an image of a fruit/vegetable.
* Predict if the item is *Healthy* or *Rotten*.
* Display the result clearly with an image preview.
* User-friendly web interface.
* Allow multiple users to use it simultaneously.

**Non-Functional Requirements:**

* Should be responsive (works on phone/laptop).
* Prediction should be fast (within seconds).
* Works with minimal internet connection.
* Scalable for adding more categories in the future.

**Data Flow Diagram**

[User] --> (Upload Image) --> [Flask App] --> [CNN Model]

⬇ ⬇

Save Image in /static Predict Class

⬇ ⬇

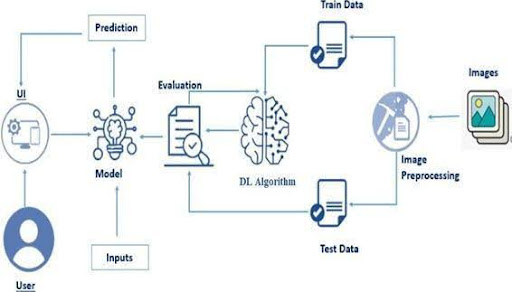
Show Output (HTML) <----- Return Result

**Technology Stack**

| **Layer** | **Tools/Technologies Used** |
| --- | --- |
| UI / Frontend | HTML, CSS, Bootstrap, JavaScript |
| Backend Server | Python + Flask |
| ML Model | Keras with TensorFlow backend, VGG16 (Transfer Learning) |
| Deployment | Localhost or cloud (optional: Streamlit/Flask deploy) |
| Image Processing | Pillow, NumPy |

**Phase-3**

**Project Design:**



**Project flow**

Data Collection: Collect or download the dataset that you want to train.

Data pre-processing

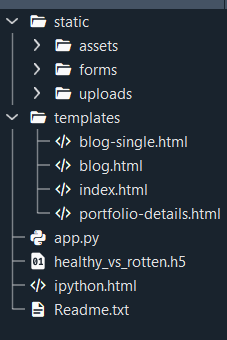
* Data Augmentation
* Splitting data into train and test

Model building

* Import the model-building libraries
* Initializing the model
* Training and testing the model
* Evaluating the performance of the model
* Save the model

Application Building

* Create an HTML file
* Build python code



**Phase-4**

**Project planning:**

[User]

⬇ Upload Image

[Frontend (HTML/CSS/JS)]

⬇

[Flask Web Server (app.py)]

⬇

[Model Prediction using VGG16]

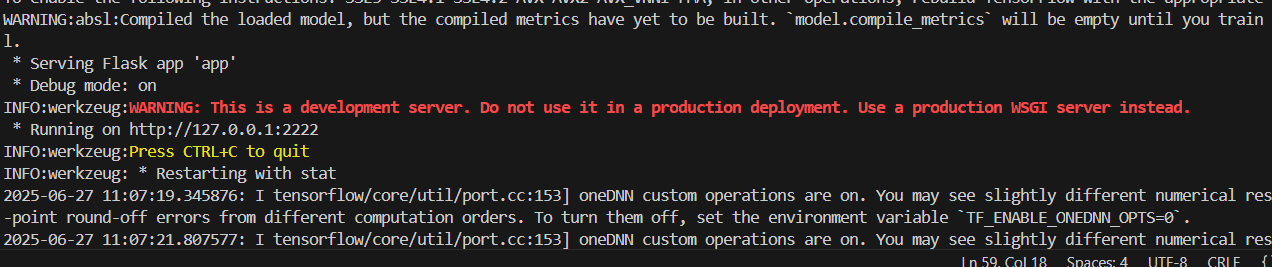
⬇

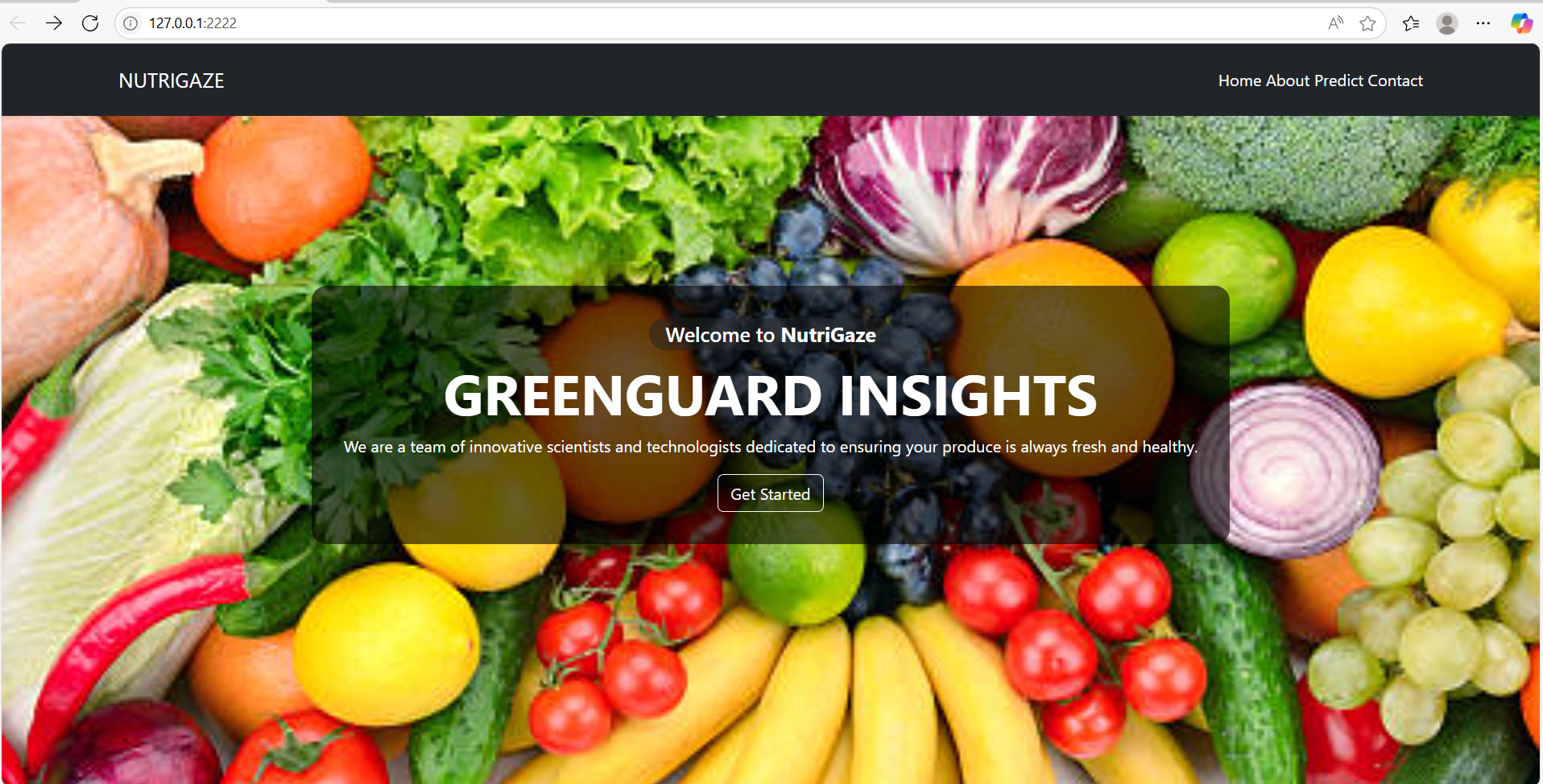
[Output Display (Prediction + Image)]

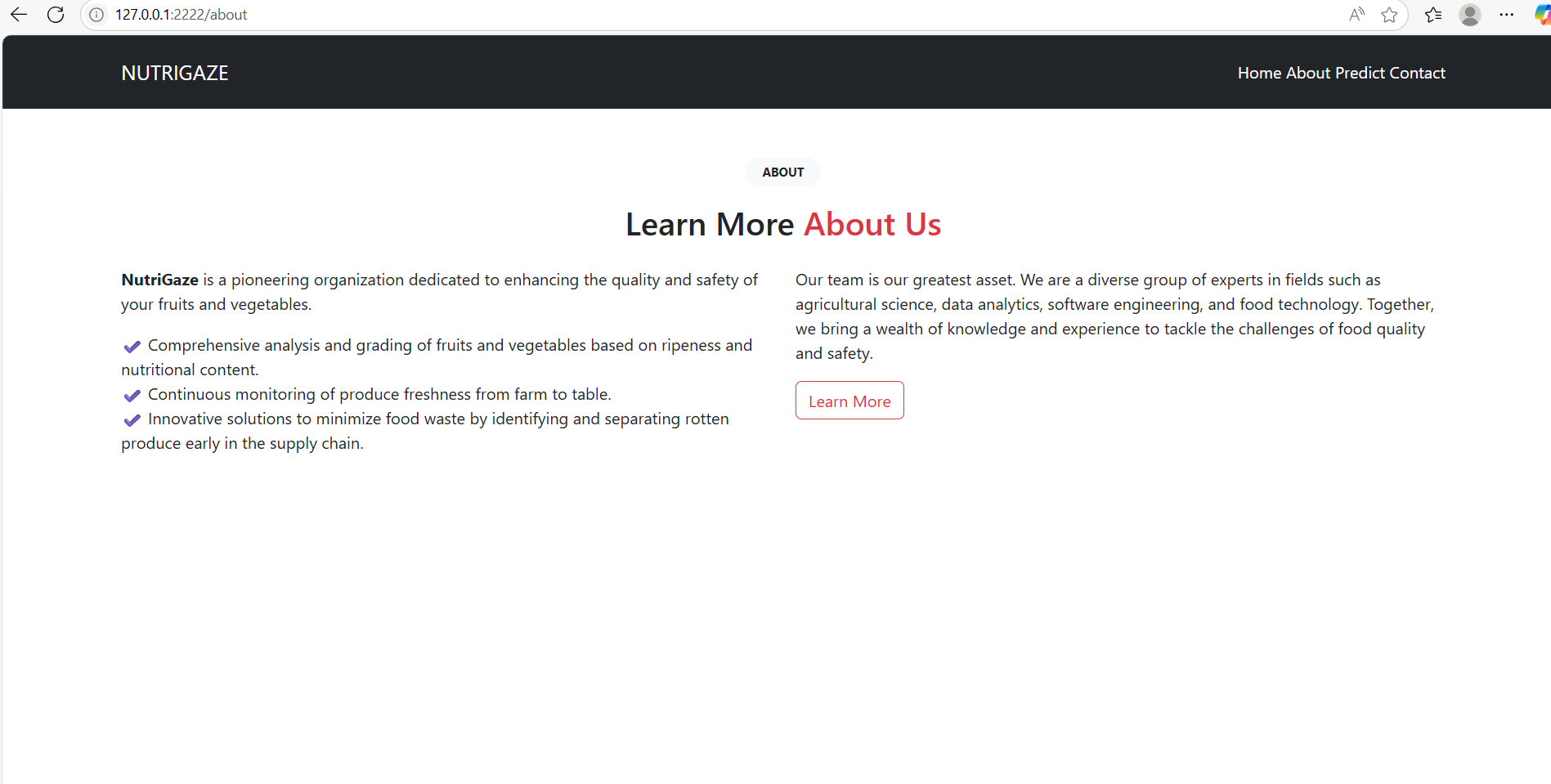
⬇

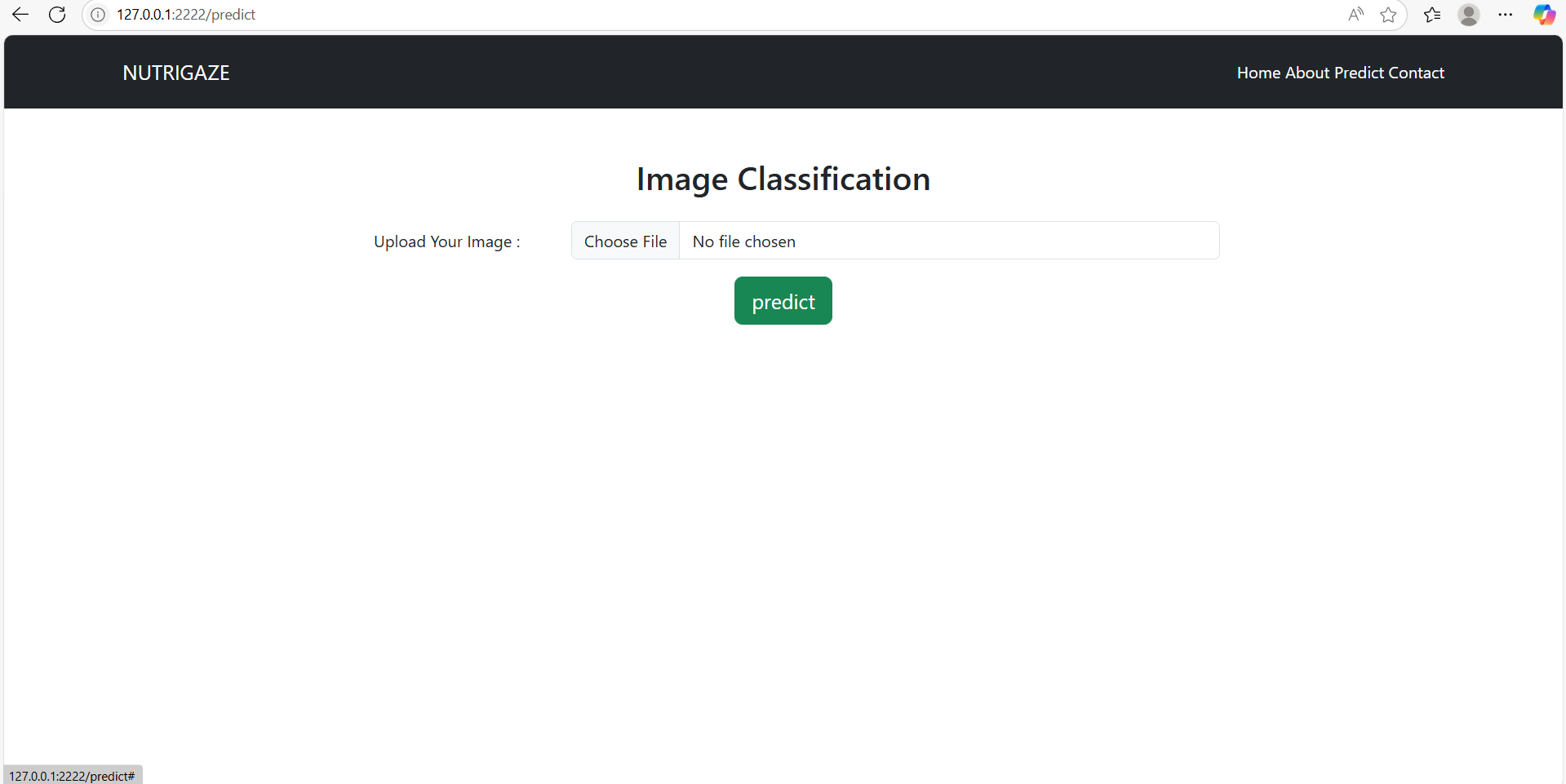
[Result shown in output.html]

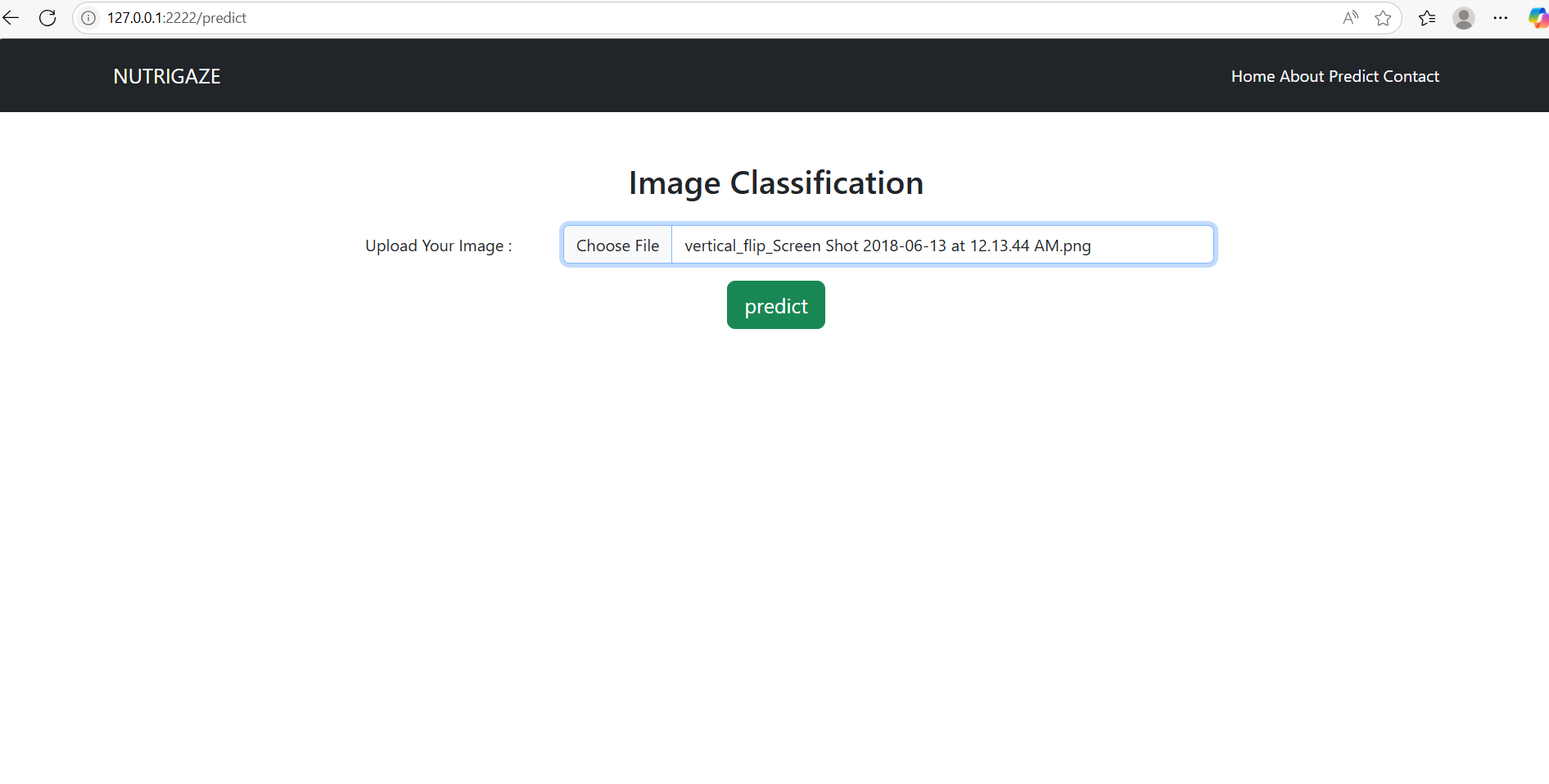
**Results:**

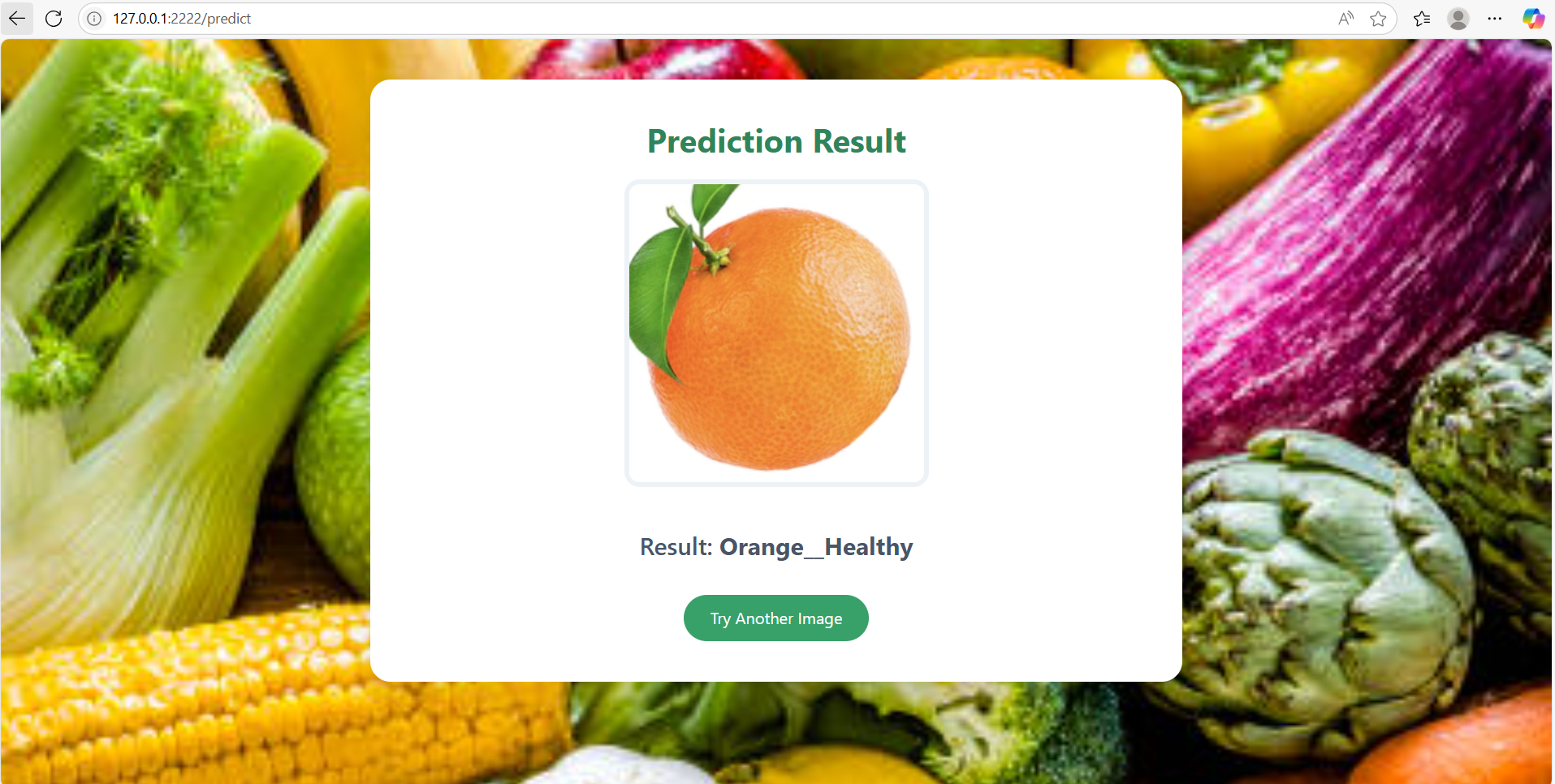
****

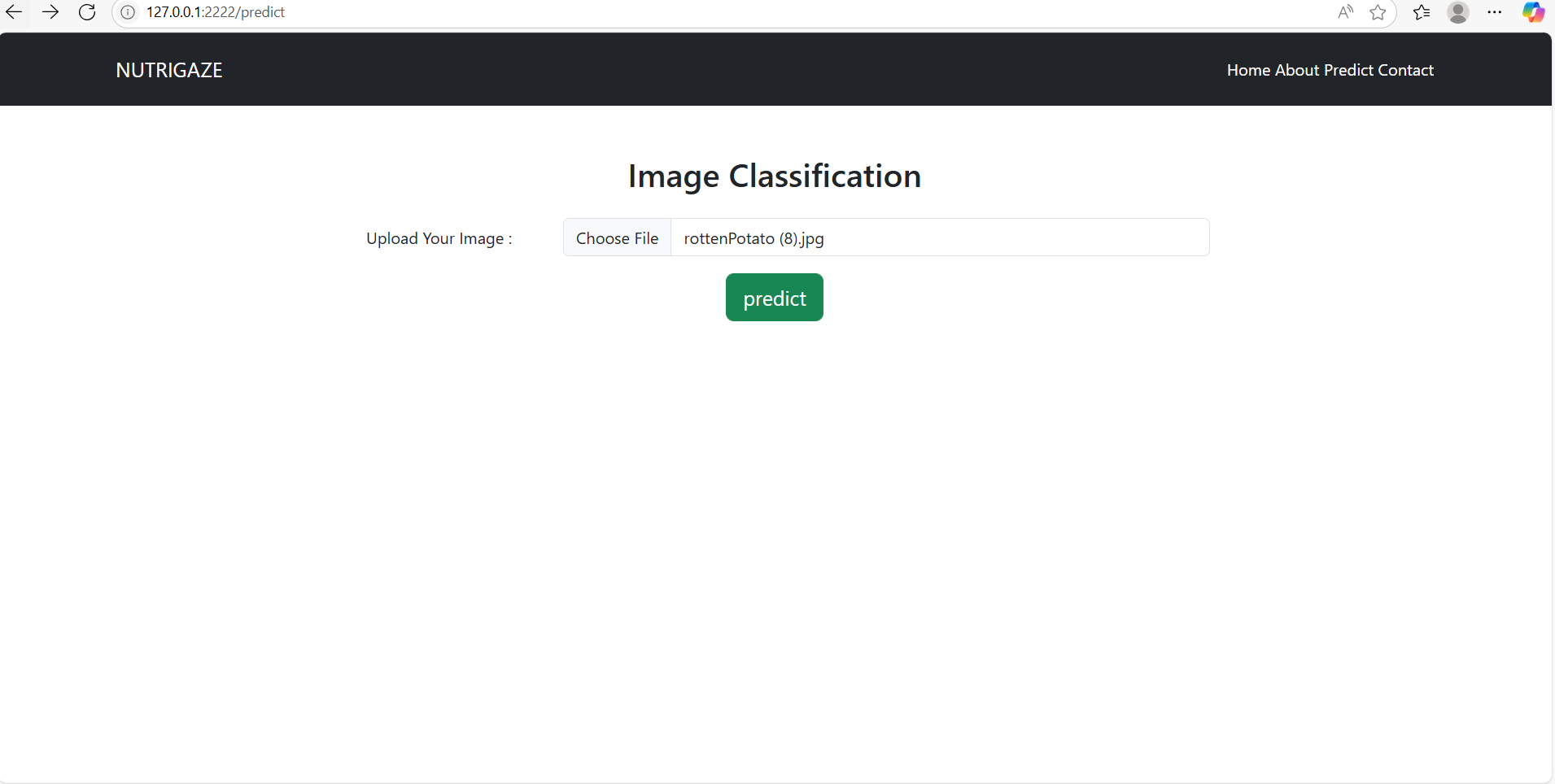












**Appendix:**

**Dataset:** [**https://www.kaggle.com/datasets/muhammad0subhan/fruit-and-vegetable-disease-healthy-vs-rotten**](https://www.kaggle.com/datasets/muhammad0subhan/fruit-and-vegetable-disease-healthy-vs-rotten)

Code:



Git hub: