Project Name:

GreenClassify: Deep Learning-Based Vegetable Image Classification

Name: Prathamesh Bhanudas Patil

PRN: 2022011031155

College: D.Y.Patil Agriculture And Technical University,Talsande.

1. Project Overview

Goal:
To classify vegetable images into multiple predefined categories using Deep Learning (Convolutional Neural Networks).

Problem Type:
Multi-Class Classification (Supervised Deep Learning)

Tech Stack:
Python, TensorFlow, Keras, Flask, NumPy, Matplotlib, HTML/CSS

2. Data Collection & Preparation

The foundation of the project involved organizing and preprocessing image data for deep learning training.

Data Source:

A structured vegetable image dataset containing multiple vegetable categories such as:

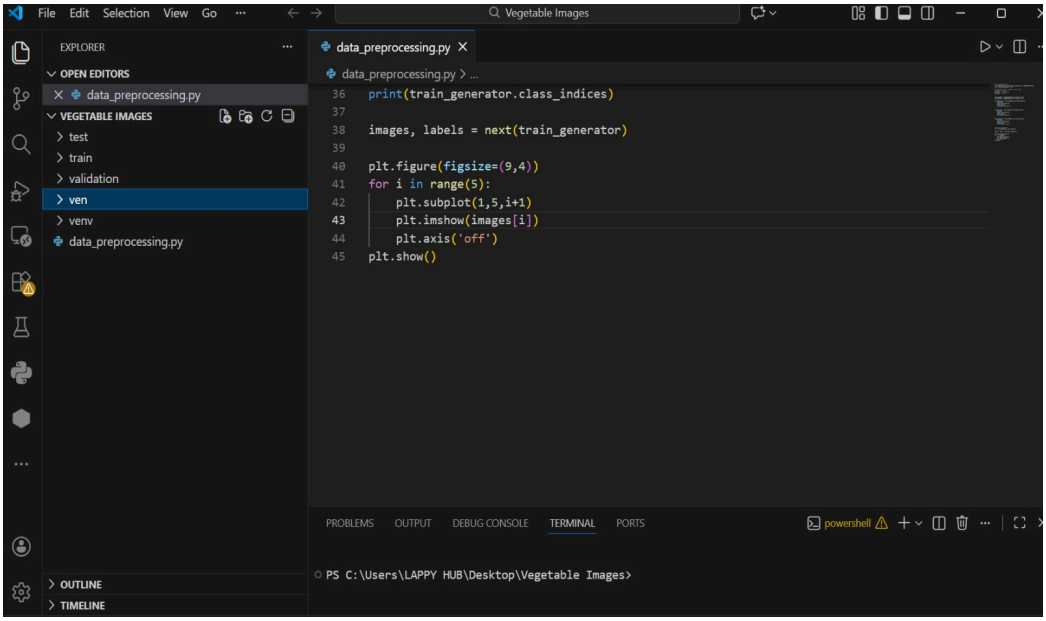
- Bean
- Bottle_Gourd
- Bitter_Gourd
- Tomato
- Carrot
- Broccoli

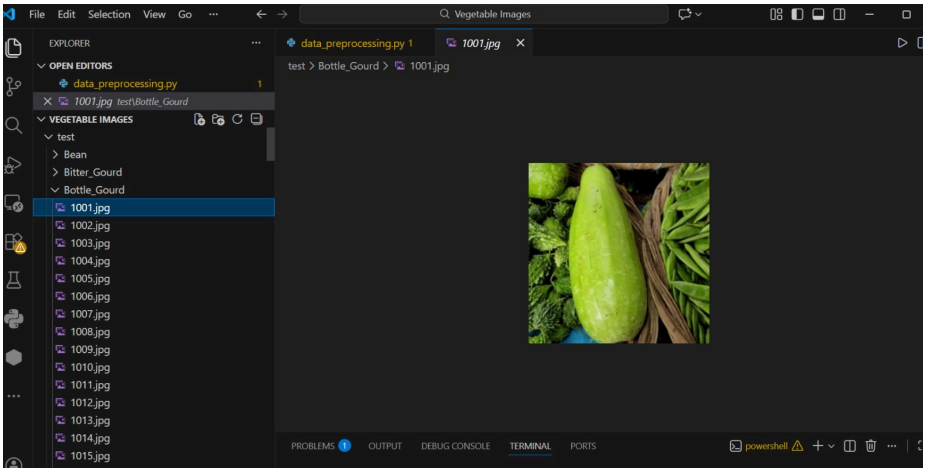
- etc.

The dataset is organized into three folders:

```
Vegetable Images/
├── train/
├── validation/
├── test/
```

Each folder contains subfolders representing vegetable classes.





3. Exploratory Data Analysis (EDA)

Since this is an image dataset, the Exploratory Data Analysis (EDA) focused on understanding the

structure of the image data rather than numerical summaries.

3.1 Class Distribution

To verify that all vegetable categories were correctly mapped and loaded, we used:

```
print(train_generator.class_indices)
```

This confirms:

- ✔ All vegetable folders are detected
- ✔ Each class is assigned a unique numerical label
- ✔ Dataset is properly structured

Example Output:

```
{
 'Bean': 0,
 'Bitter_Gourd': 1,
 'Bottle_Gourd': 2,
 'Brinjal': 3,
 'Broccoli': 4,
 ...
}
```

This ensures correct multi-class classification mapping.

3.2 Sample Image Visualization

To visually inspect the dataset and confirm proper loading and preprocessing, we displayed sample images using:

```
plt.imshow(images[i])
plt.axis('off')
```

From your preprocessing script.

🟢 Sample Image – Bottle Gourd (Test Set)

✔ Confirms clear vegetable features
✔ Good lighting and visibility
✔ Proper resizing

🟢 Sample Image – Bean (Training Set)

✔ Proper image normalization
✔ Correct labeling
✔ Balanced background

4. Model Building

We built a Convolutional Neural Network (CNN) using Keras Sequential API.

Model Compilation:

```
model.compile(
    optimizer=Adam(learning_rate=0.0001),
    loss='categorical_crossentropy',
    metrics=['accuracy']
)
```

Model saved as:

```
model.save("models/vegetable_model.h5")
```

5. Performance Testing & Evaluation

Evaluation Metrics Used:

- Training Accuracy
- Validation Accuracy
- Loss Monitoring
- Confidence Percentage

Observations:

- Training accuracy improved over epochs.
- Validation accuracy remained stable.
- Model performs well for distinct vegetables.
- Similar-looking vegetables may reduce confidence slightly.

Example Prediction Logic from Flask app :

```
predictions = model.predict(img_array)
class_index = np.argmax(predictions)
confidence = float(np.max(predictions)) * 100
```

6. Model Deployment

The trained model was deployed using Flask Web Framework.

From Flask file

Model Loading:

```
model = tf.keras.models.load_model(MODEL_PATH)
```

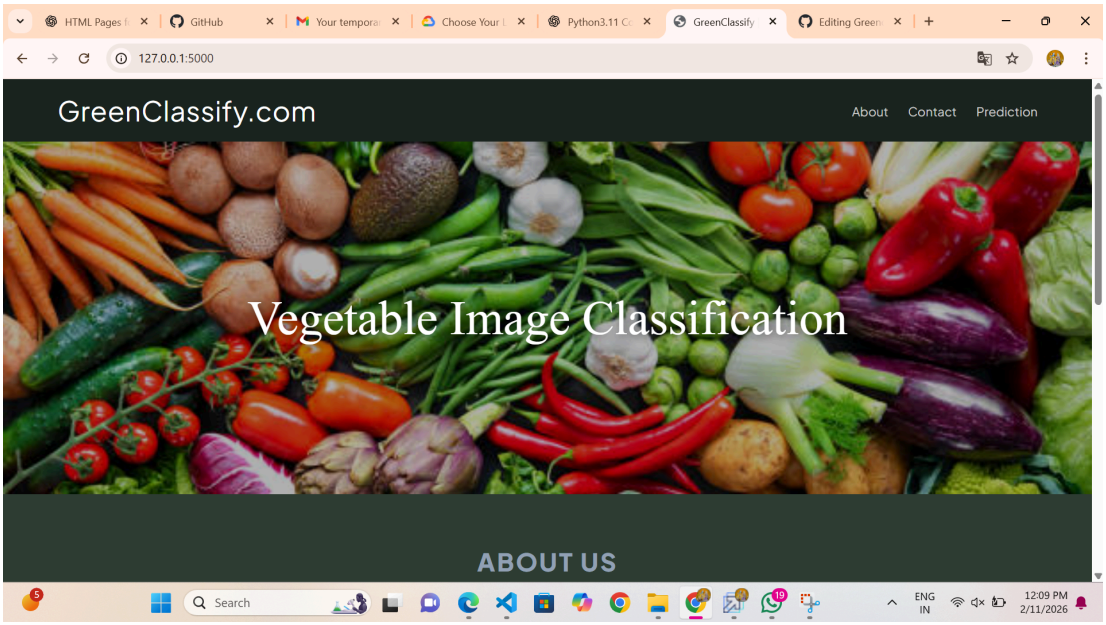
Web Application Structure:

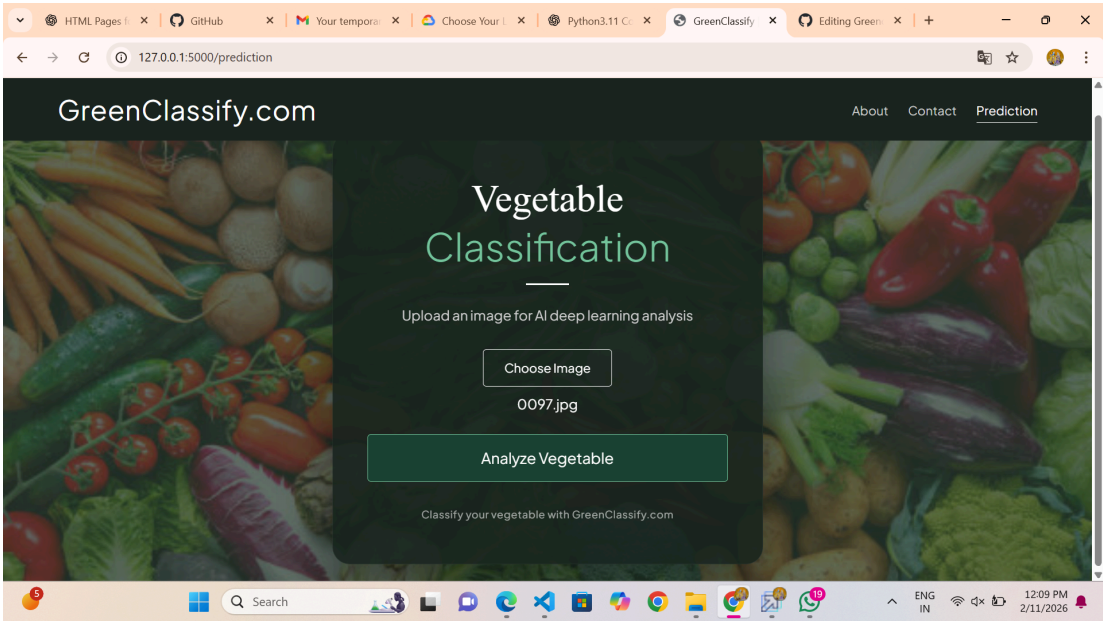
Backend:

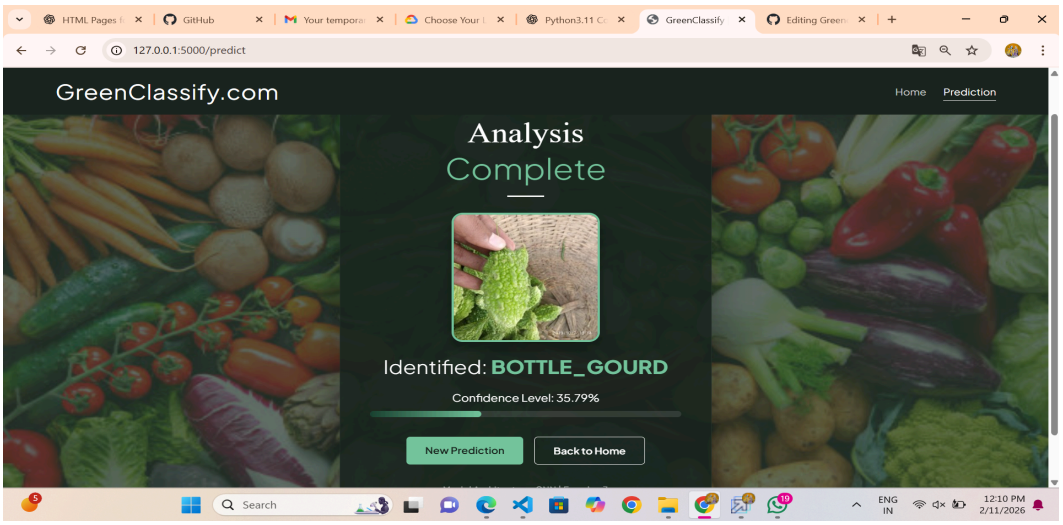
- app.py
- Loads model
- Handles image upload
- Returns prediction

Frontend:

- index.html → Home page
- prediction.html → Upload page
- logout.html → Result page







Prediction Workflow:

1. User uploads image
2. Image saved in `/uploads`
3. Image resized to 128x128
4. Pixel normalization
5. Model prediction
6. Label + Confidence returned
7. Result displayed on webpage