Plant Disease Detection using Deep Learning

Project Overview

This project focuses on building a computer vision system that can automatically detect diseases in plant leaves using machine learning and deep learning models. Students will train a neural network to classify leaf images into multiple disease categories. This project demonstrates practical AI applications in agriculture.

Objective

Develop a **multi-class image classifier** that can detect and identify plant diseases from leaf images.

Dataset

PlantVillage Dataset
Public & Open Source
🔗 https://www.kaggle.com/emmarex/plantdisease

Dataset Details

- ~54,000 leaf images
- 38 disease/healthy classes
- Image resolution: 256x256 (varies slightly)

Suggested Student Scope Option:

Focus only on a single crop category (e.g., **Tomato** – 9 classes) to reduce complexity.

Tools & Technologies

Category	Options
Programming	Python
ML Libraries	TensorFlow/Keras or PyTorch
Image Handling	OpenCV, Pillow
Visualization	Matplotlib, Seaborn
UI (Student choice)	Flask / Tkinter / Streamlit

Project Components

1️⃣ **Data Processing & Augmentation**

- Load dataset and split into Train/Validation/Test
- Apply image augmentations:
    - Rotation, zoom, flip
    - Brightness change
- Normalize pixel values for model compatibility

2️⃣ Model Building

Option A — Train a **Custom CNN**

Option B — Use **Transfer Learning**

- MobileNetV2
- ResNet50
- EfficientNetB0

Outputs:

- Accuracy
- Loss curves
- Confusion matrix

3️⃣ Model Evaluation

Use performance metrics:

- Overall accuracy
- Per-class accuracy (important for agriculture!)
- Precision, recall, F1-score

4️⃣ User Interface

GUI must allow:

- Uploading a leaf image
- Displaying predicted disease + confidence score
- Showing input image preview

Optional bonus:

- **Grad-CAM** visualization: highlight which parts of the leaf were used for detection

5️⃣ Documentation & Reporting

- Write one page document for methodology, result, and conclusion. Use UML diagram, class diagram, confusion matrix to explain the result.

Expected Deliverables

Component	Requirement
Code	Clean, modular Python scripts
UI	Fully functional demo
Report	Methodology, results, business conclusion
Visuals	Confusion matrix, performance charts

Submission Guidelines

1. Upload the source code as zip folder.
2. PDF documentation