****Plant Disease Detection System** for **sustainable agriculture****

**Problem Statement:**

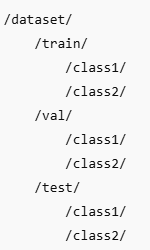
Early detection and diagnosis of plant diseases remain a significant challenge in sustainable agriculture, particularly in regions with limited access to expert agronomists or diagnostic labs. This leads to delayed treatment, reduced crop yield, and excessive use of chemical pesticides, harming both the environment and food security. There is a critical need for an intelligent, automated plant disease detection system that can identify diseases from leaf images and provide actionable insights to farmers, enabling timely and precise interventions for healthier crops and sustainable farming practices.

**Pipeline: Plant Disease Detection System:**

1. **Data Collection:**

* Collect images of healthy and diseased plant leaves.
* Data sources:
* Field photography (smartphones, drones).
* Public datasets (e.g., PlantVillage).
* Organize the dataset into separate folders for each disease class (e.g.,Tomato\_\_\_Bacterial\_spot, Tomato\_\_\_Healthy).

1. **Data Uploadation / Loading:**

* Compress the dataset into a .zip file for efficient handling and uploading.
* Extract the zip file into a structured directory:
* 

#### ****3. Dataset Preparation****

* Load the dataset using appropriate frameworks:
* TensorFlow/Keras: ImageDataGenerator
* PyTorch: Dataset and DataLoader
* Split the data into:
* **Training Set (70%)**
* **Validation Set (15%)**
* **Test Set (15%)**
* Ensure class balance using stratified sampling.

#### ****4. Image Processing and Augmentation****

* Resize images to a standard size (e.g., 224x224 pixels).
* Normalize pixel values between 0 and 1.
* Apply data augmentation techniques (only on training set):
* Rotation
* Horizontal/vertical flip
* Zoom and crop
* Brightness/contrast adjustment
* Noise injection

#### ****5. CNN Model Building****

#### **Build a Convolutional Neural Network (CNN) using:**

#### **Custom layers (Conv2D, MaxPooling, Flatten, Dense).**

#### **Or Transfer Learning with pre-trained models (e.g., ResNet, MobileNet, EfficientNet).**

#### **Use activation functions like ReLU and Softmax.**

#### **Compile the model using:**

#### **Loss function: Categorical Crossentropy**

#### **Optimizer: Adam**

#### **Metrics: Accuracy**

#### ****6. Model Training and Validation****

* Train the model using the training data.
* Validate performance using the validation set.
* Monitor metrics such as:
* Accuracy
* Precision
* Recall
* F1-score
* Use EarlyStopping and ModelCheckpoint callbacks to save the best model.

#### ****7. Model Testing and Evaluation****

* Evaluate final performance on the unseen test set.
* Generate:
* Confusion matrix
* Classification report
* ROC curves (if applicable)