# Plant Disease Detection System for Sustainable Agriculture

## Problem Statement

Develop a CNN-based model capable of detecting and classifying plant diseases from images of leaves of various crops such as apple, cherry, grape, and corn. The model should accurately identify both healthy and diseased leaves while predicting the specific type of disease. This system will aid in precision agriculture by enabling early detection and effective disease management.

# CNN Image Classification Pipeline

## Step 1: Data Collection & Loading

The dataset is divided into three subsets:

- Train: Used to train the model.

- Validation: Used to tune and validate the model during training.

- Test: Used to evaluate the model after training.

Each folder contains subfolders representing different image categories. This structure ensures a clean and organized workflow for model development.

## Step 2: Image Preprocessing & Augmentation

To work with datasets in Google Colab, follow these steps:

1. Zip the dataset folder.

2. Upload the ZIP file to your Google Drive.

3. Mount Google Drive in Colab.

4. Extract the ZIP file within Colab.

After this, the dataset becomes accessible and ready for preprocessing and model training.

## Step 3: Preprocessing Techniques

Preprocessing ensures all images are consistent in size and quality. Common steps include:

- Resizing all images to the same dimension (e.g., 128x128).

- Normalizing pixel values.

## Step 4: Data Augmentation Techniques

Augmentation increases data diversity and improves model generalization. Techniques include:

- Rotation, flipping, zooming, shifting, and brightness changes.

## Step 5: CNN Model

A Convolutional Neural Network (CNN) is designed to process image data. Key components:

- Convolutional Layers: Extract features using filters.

- ReLU Activation: Adds non-linearity.

- Pooling Layers: Reduce spatial dimensions.

- Flatten & Dense Layers: Used for classification.

The CNN learns patterns from the training images and makes predictions.

## Step 6: Test and Evaluate

After training, the model is tested using the test dataset. Evaluation metrics include:

- Accuracy: Overall correctness of predictions.

- Loss: Indicates prediction error.

- Precision, Recall, F1-Score: For detailed performance analysis.

- Confusion Matrix: Visual summary of prediction results.

This step confirms how well the model performs on unseen data.