**Plant Disease Detection System for Sustainable Agriculture**

**Week 1 Milestone Document**

**Problem Statement**

In modern agriculture, plant diseases pose a significant threat to crop yields, food security, and the economy. Early detection of plant diseases is crucial to mitigate the damage caused by pests, pathogens, and environmental stressors. Traditional methods of disease detection rely on visual inspections, which can be time-consuming and prone to human error. Moreover, diseases often spread rapidly, making it difficult for farmers to take timely action. This leads to the use of chemical treatments, which can have harmful effects on the environment and human health.

To address this issue, there is a need for an automated and accurate system that can detect plant diseases early, allowing farmers to take proactive measures to protect their crops and promote sustainable agriculture practices.

**Proposed Problem Solution**

The proposed solution is to develop a **Plant Disease Detection System** that automates the detection of plant diseases using machine learning and image processing techniques. The solution consists of the following key components:

1. **Dataset Collection**:  
   A collection of plant leaf images is gathered from New Plant Diseases Dataset from Kaggle. These images are categorized into healthy and diseased plant leaves, representing various types of diseases.
2. **Image Preprocessing**:  
   The collected images are preprocessed to enhance the quality of the data and remove any noise. Preprocessing techniques like resizing, normalization, and augmentation are applied to ensure that the model can learn from a diverse set of images and generalize well.
3. **Disease Classification using CNN**:  
   A Convolutional Neural Network (CNN**)** model is developed to classify the images of plant leaves into different categories like healthy or diseased. CNNs are particularly effective for image classification tasks as they can automatically learn relevant features from the raw pixel data of images. The CNN model is trained using the preprocessed dataset, and the output is a prediction of whether a given plant leaf is diseased and, if so, the specific type of disease.
4. **Model Evaluation**:  
   After training the CNN model, its performance is evaluated using various metrics such as accuracy, precision, recall, and F1 score. This ensures that the model is reliable and can accurately classify plant diseases based on the images.
5. **Website Development with Streamlit**:  
   Once the model is trained and evaluated, a user-friendly web application is developed using Streamlit. The website allows farmers or agricultural experts to upload images of plant leaves and get real-time predictions on whether the plant is diseased or healthy. The Streamlit app provides a simple interface where users can:
   * Upload images of plant leaves.
   * View the predicted disease classification (healthy or diseased).
   * Receive information about the specific disease and recommended actions or treatments.

**Tools Used**

1. **Google Colab**:  
   Google Colab is used for coding and model training in the cloud. It provides an interactive environment with access to powerful GPUs, which significantly speeds up the training process of deep learning models. It is especially useful for tasks like image classification, where large datasets are involved.
2. **Libraries and Frameworks**:
   * **TensorFlow and Keras**: These libraries are used for developing and training the CNN model. TensorFlow is an open-source machine learning framework, and Keras is a high-level neural networks API built on top of TensorFlow, which simplifies the process of building, training, and evaluating deep learning models.
   * **OpenCV**: OpenCV is used for image preprocessing tasks such as resizing, color correction, and feature extraction from the plant leaf images.
   * **Matplotlib and Seaborn**: These libraries are used for visualizing the results of the model, such as accuracy plots, confusion matrices, and other performance metrics.
3. **Development Environment**:
   * **Visual Studio Code**: Visual Studio Code (VS Code) is used for writing and debugging Python code. It provides an efficient development environment with extensions for Python, TensorFlow, and other necessary tools.
   * **Terminal**: The terminal is used for running scripts, managing dependencies, and interacting with the command-line interface for various tasks, including managing libraries and packages.
4. **Dataset**:  
   This dataset has been recreated through offline augmentation of the original dataset. It contains approximately 87K RGB images of healthy and diseased crop leaves, categorized into 38 distinct classes. The dataset is split into training and validation sets in an 80/20 ratio, while maintaining the original directory structure. Additionally, a separate directory with 33 test images has been created for prediction purposes.
5. **Hardware**:
   * **Camera or Smartphone**: A camera or smartphone is used to capture high-resolution images of plant leaves, which are then uploaded to the system for prediction.
   * **Computer with GPU**: A computer with GPU support is used for faster training of the deep learning model, especially when working with large datasets.

**Conclusion**

The Plant Disease Detection System aims to leverage machine learning and image processing techniques to automate the identification of plant diseases. By using a CNN model trained on a dataset of plant leaf images, the system can accurately classify plants as healthy or diseased. The development of a user-friendly web application using Streamlit further enhances the accessibility of the system, allowing farmers to quickly diagnose plant health. This solution not only aids in early disease detection but also promotes sustainable agricultural practices by reducing reliance on harmful pesticides and supporting more informed decision-making in farming.