Project Report

on

"CROP DISEASE DETECTION USING MACHINE LEARNING"

Submitted for partial fulfillment of requirement for the degree of

BACHELOR OF ENGINEERING

(Computer Science & Engineering)

Submitted by

NIKITA GOPAL DEOCHE **NIKITA SHYAMSUNDAR NASARE**

RUNJAL PRAMOD GULAHE **RITESH HARIDAS DESHMUKH**

Under the Guidance of

PROF. (DR.) N. S. DANDGE

Department of Computer Science & Engineering,

Prof. Ram Meghe Institute of Technology & Research, Badnera

2023-2024

Project Report

on

"CROP DISEASE DETECTION USING MACHINE LEARNING"

Submitted for partial fulfillment of requirement for the degree of

BACHELOR OF ENGINEERING

(Computer Science & Engineering)

Submitted by

Nikita Gopal Deoche

Nikita Shyamsundar Nasare

Runjal Pramod Gulahe

Ritesh Haridas Deshmukh

Under the Guidance of

Prof. (Dr.) N. S. Dandge

Department of Computer Science & Engineering,

Prof. Ram Meghe Institute of Technology & Research, Badnera

2023-2024

Department of Computer Science & Engineering
Prof. Ram Meghe Institute of Technology & Research, Badnera
2023 - 2024

CERTIFICATE

This is to certify that the Project (8KS07) entitled

"CROP DISEASE DETECTION USING MACHINE LEARNING"

is a bonafide work and it is submitted to the

Sant Gadge Baba Amravati University, Amravati

By

Nikita Gopal Deoche
Nikita Shyamsundar Nasare
Runjal Pramod Gulahe
Ritesh Haridas Deshmukh

in the partial fulfillment of the requirement for the degree of ***Bachelor of Engineering*** *in* ***Computer Science & Engineering****, during the academic year* ***2023-2024*** *under my guidance.*

Prof.(Dr.) N. S. Dandge	_____	***Dr. M. A. Pund***
Guide Department of Computer Sci.& Engg. Prof. Ram Meghe Institute of Technology & Research, Badnera	External Examiner	Head, Department of Computer Sci.& Engg. Prof. Ram Meghe Institute of Technology & Research, Badnera

ACKNOWLDEGMENT

With great pleasure, we hereby acknowledge the help given to us by various individuals throughout the project. This Project itself is an acknowledgment to the inspiration, drive, and technical assistance contributed by many individuals.

This project would have never seen the light of day without the help and guidance we have received. We would like to express our profound thanks to our Project guide Prof. (Dr.) N. S, Dandge, for their guidance and constant supervision as well as for providing necessary informationregarding the project & also for their support in completing the project.

We would like to express my profound thanks to the HOD of the CSE department Dr. M. A. Pund, for providing us with the opportunity to undertake this project and for your continuous encouragement and support throughout the process. We would like to express my gratitude to Principal Dr. G. R. Bamnote, for fostering an academic environment that encourages research and intellectual growth.

Your vision and dedication to the institution have been a constant source of inspiration for all of us. We extend our heartfelt thanks to the faculties of the Department of Computer Science & Engineering, for their kind cooperation and encouragement which helped us in the completion of this project. We owe an incalculable debt to all staff of the Department of Computer Science & Engineering for their direct and indirect help.

Name of Students:

Nikita Gopal Deoche _____

Nikita Shyamsundar Nasare _____

Runjal Pramod Gulahe _____

Ritesh Haridas Deshmukh _____

TABLE OF CONTENTS

ABSTRACT

Crop diseases pose significant threats to global food security, affecting yield and quality. In this project, we propose a novel approach utilizing machine learning techniques for the early detection of crop diseases to mitigate these threats. Leveraging a dataset of images depicting healthy and diseased crops, we employed various machine learning algorithms for classification and detection. Our methodology involved preprocessing techniques for image enhancement, feature extraction using convolutional neural networks (CNNs), and model training using deep learning architectures. Through extensive experimentation and validation, we achieved promising results in accurately identifying crop diseases based on image inputs. The developed system demonstrates potential for real-time monitoring and decision support in agricultural settings, enabling timely intervention to prevent yield losses and optimize crop management practices. This project contributes to the ongoing efforts in precision agriculture, offering a scalable and effective solution for early disease detection, ultimately enhancing agricultural productivity and sustainability. This work has reviewed the most relevant CNN articles on detecting various plant leaf diseases over the last five years. This project proposes a dynamic crop disease detection system in which agriculture experts will register crops with their diseases and images.

LIST OF FIGURE

LIST OF SCREENSHOT

1. Introduction

Crop disease detection is a crucial aspect of modern agriculture, aiming to identify and diagnose diseases affecting crops in their early stages. Timely detection is essential for implementing effective management strategies, preventing yield losses, and ensuring food security. With the advent of advanced technologies such as artificial intelligence (AI) and machine learning (ML), the process of detecting crop diseases has undergone significant advancements, offering more accurate and efficient solutions. Traditionally, crop disease detection relied heavily on manual inspection by agricultural experts, which is time-consuming, labor-intensive, and often prone to human error. However, technological innovations have revolutionized this field by introducing automated and data-driven approaches.

Modern crop disease detection systems leverage various techniques, including computer vision, image processing, and pattern recognition, to analyze visual symptoms exhibited by diseased plants. These systems utilize large datasets of images depicting both healthy and diseased crops to train ML models, such as convolutional neural networks (CNNs), to recognize characteristic patterns associated with different diseases. Crop disease detection technologies offer several benefits, including early detection of diseases, proactive management strategies, reduced reliance on chemical inputs, and increased productivity. By empowering farmers with timely and accurate information, these systems play a vital role in promoting sustainable agriculture and ensuring global food security in the face of evolving environmental challenges and emerging pathogens.

Crop disease detection technologies have the potential to revolutionize agricultural practices by providing farmers with valuable insights and actionable information to mitigate the impact of diseases on crop yields. These technologies offer a non-invasive and cost-effective approach to monitoring crop health, allowing for early intervention and targeted treatment strategies. By promptly identifying disease outbreaks, farmers can implement timely measures such as precision spraying, crop rotation, and genetic resistance breeding, thereby minimizing yield losses and reducing the need for chemical pesticides. Furthermore, crop disease detection systems contribute to sustainable farming practices by promoting integrated pest management (IPM) approaches and reducing environmental risks associated with excessive pesticide use. By precisely targeting areas affected by diseases, farmers can optimize resource utilization, minimize input costs, and mitigate the negative environmental impacts associated with conventional farming practices. Additionally, by providing valuable insights into disease dynamics and trends, these technologies facilitate proactive decision-making and long-term planning, helping farmers adapt to changing environmental conditions and emerging threats.

In essence, crop disease detection represents a transformative tool in modern agriculture, offering farmers the ability to safeguard their crops against diseases, optimize production practices, and promote sustainable food systems. As these technologies continue to evolve and become more accessible, their widespread adoption has the potential to revolutionize the way we manage crop health and ensure global food security in the face of mounting environmental challenges and growing population pressures.

Machine Learning based disease classification

A number of Machine Learning (ML) methods have been used for disease detection in crops. Some of the methods used by researchers are described below:

a) k-Nearest Neighbors: When the class of an input feature vector is decided based on the weighted distance of its distance from k nearest neighbors, it is called k-Nearest Neighbor algorithm [25–27]. Bauer et al. [28] have used multi-spectral images of sugar beet root to classify between healthy, Cercospora beticola and Uromyces betae diseases. Authors used k-NN and Bayes classification with Gaussian Mixture Model (GMM) and found GMM gave better classification rate of 94% for healthy leaves, 91% for the leaf spot disease Cercospora beticola and 86% for the rust fungus Uromyces betae disease.

b) Naive Bayes: The technique [29, 30] is based on estimating the probability of a class given a set of features (feature vector x), based on probability of feature taking the value given that it belongs to that class and probability of that class. The probability is computed by the formula:

$$P(y|x_1, x_2, ...x_n) = \frac{P(x_1|y)P(x_2|y).....P(x_n|y) \times P(y)}{P(x_1)P(x_2).....P(x_n)} \qquad (1.1)$$

where y is the output class; and x1, x2,….,xn are components of the feature vector. Johannes et al. [24] used the Naive Bayes technique for the identification of three diseases of wheat leaves.

c) Support Vector Machine: Main idea governing SVM [17, 31–33] is to construct a separating hyperplane given a set of training data such that separation between positive and negative samples is maximized. Support Vector Machines (SVM) have been widely used by researchers for classifying diseases in various crops.

Rumpf et al. [5] have demonstrated the use of SVMs for detecting 3 types of diseases in sugar beet root using leaves images. SVM has also been used by Mokhtar et al. [23] to classify two types of tomato plant diseases and obtained an accuracy of 92%. Leaf image classification is also found in [34] where Pantazi et al. have tried to classify vine images in 4 classes. Initially Local Binary Pattern (LBP) is used to divide

the colored image into 32 bin histograms. One Class Support Vector Machines (SVM) were trained for 4 types of diseases. When tested on 100 images of each class recognition rate was 97%, 95% and 93% for infected leaves and 100% for healthy leaves. Prince et al. [35] have reported that SVM classifier with linear kernel gives an accuracy of more than 90% for tomato plant powdery mildew disease identification. Shrivastava and Pradhan [36] have used 7 classifiers namely SVM, Discriminant Classifier, k-NN, Naive Bayes, Decision Trees, Random Forest and Logistic Regression for classification of 3 classes of rice plant disease and a healthy class. Authors mention that maximum accuracy of 94.65% was achieved using SVM classifier.

d) Other ML methods: Other Machine Learning techniques are evaluated in [37] by Yang and Guo who provide a review of machine learning techniques such as Naive Bayes classifier, Support Vector Machine (SVM), K-Means clustering, Artificial Neural Networks (ANN), Decision Trees and Random Forests to identify plant diseases from leaf images. The authors use machine learning for the identification of genes involved in plant-pathogen (disease-causing bacteria or virus) interactions. Authors mentioned that Sankaran et al. [38] have shown best accuracy of 95% using Quadratic Discriminant Analysis (QDA) for detection of Huanglongbing in citrus orchards. ML methods use a feature set which is in form of an input vector. The common features extracted using various Image Processing techniques are Haralick features, Hu-moments, HSV histogram, LBP histogram, etc. for image classification problems. There are some problems in ML methods like accuracy in predicting disease of a plant, saturation of performance after a certain point as classes and dataset volume is increased. This causes low performance in case of large datasets. To solve this problem Neural Networks based solutions have been proposed by several researchers.

1.4 Deep Learning based disease classification

Recently several researchers have reported that Deep Learning is a better approach to achieve high accuracy in plant disease identification [6, 39, 40]. Generally, researchers have used transfer learning on pre-trained models of other domains and reported good results in disease classification.

1.4.1 2D Convolution Neural Network

2D Convolution Neural Networks (CNN) [41–43] are a special type of Neural Networks. These work on two-dimensional input (example images) whereas Artificial Neural Network (ANN) [44, 45] works on one-dimensional vectors. Each layer in CNN has a certain number of filters which convolves over input 2-D signal. In initial layers, these filters detect edges and corners and in deeper layers, they get trained to recognize more complex features pertaining to the objects in the image. In CNN forward

propagation is guided by a convolution operation on input images using a kernel and explained with equation (1.2):

$$(I * K)[i, j] = \sum_{p=0}^{m-1} \sum_{q=0}^{n-1} I[i - p, j - q]K[p, q] \qquad (1.2)$$

Here I is the input 2-D matrix and K is the filter of size m × n. At the start of training of CNN, filters are initialized with random values and as the model gets trained by back-propagation algorithm using the errors in estimating the actual output, the weights of the filters are modified to identify specific patterns in inputs. The convolution layers are followed by max-pooling layers which reduce the size of the 2-D signal to retain only important features and to speed up computations. After couple of convolution and max pool layers, a flattened layer is included to transform the 2-D signal to 1-D signals similar to those used in ANN. The last layer of the model is a softmax layer which has nodes equal to the number of output classes. This layer indicates the probability for all the classes under consideration. The maximum probability node is assigned the value 1 and the rest are assigned 0 values. Thus the output is in form of a 1-D vector of 0's and 1's. The node with value 1 represents the class to which the input image belongs to.

1.4.1.1 Forward Propagation in CNN: The images which are given as input to CNN are resized to some fixed size and convolved over image processing filters. The role of the convolution filter is to extract the important features from the training dataset.

1.4.1.2 Back Propagation in CNN: The Deep CNN model starts with random weights and during back propagation, the loss is calculated with the help of equation (1.3) as given below. As per the calculated loss, the weights and biases are changed with respect to the gradient of the loss. The whole process (forward and backward propagation) is iterated for a certain number of epochs to minimize the average loss

$$L(a, y) = -(y \log(a) + (1 - y) \log(1 - a)) \qquad (1.3)$$

Here L is the loss, y is expected output and a is predicted output for a single test case

1.4.1.3 Tuning of hyper parameters: Deep CNN involves tuning of several hyper parameters like number of epochs, number of hidden layers, number of nodes in these layers, choice of activation function, drop out rate, learning rate, batch size, etc. which affect the performance of the model. In the hyper parameters tuning, experiments are repeated with a different number of hidden layers and epochs and

different activation functions or learning rate. The accuracy of CNN models increases with fine-tuning of these parameters.

1.4.1.4 Performance evaluation metrics: The performance of the a new CNN model is evaluated with other traditional machine learning methods and pre-trained CNN models based on following evaluation metrics suggested by Sokolova and Lapalme [46] and Tharwat [47]: • Accuracy: It is the main criterion for evaluating the efficiency of a model. A better model will have higher accuracy than the other model. It is calculated using equation (1.4).

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \qquad (1.4)$$

where TP is number of True Positive values, TN is number of True Negative values, FP is number of cases which were falsely reported as positive, and FN is number of cases which were falsely reported as Negative. • **F1-score:** It is also a popular performance measure and is computed as the harmonic mean of precision and recall. F1-score is calculated using equation (1.5).

$$F_1 = \frac{2 * Precision * Recall}{Precision + Recall} \qquad (1.5)$$

where precision and recall are calculated using equations (1.6) and (1.7)

$$Precision = \frac{TP}{TP + FP} \qquad (1.6)$$

$$Recall = \frac{TP}{TP + FN} \qquad (1.7)$$

Here TP - True Positive, TN - True Negative, FP - False Positive and FN - False Negative

Area Under the Receiver Operating Characteristic (AUC - ROC) curve: It serves as a measure to show how good a given machine learning method can distinguish between positive and negative classes under different threshold settings. It is a plot between TPR (y-axis) and FPR (x-axis) (refer equation (1.8) and (1.9)). A good model would have the AUC (Area Under the Curve) close to 1, while a value near about 0.5 indicates that the model is not suitable for classification. A good model will have high y-axis values on the ROC curve to show higher true positives.

$$TPR = \frac{TP}{TP + FN} \quad (1.8)$$

$$FPR = \frac{FP}{FP + TN} \quad (1.9)$$

Here TPR refers to True Positive Rate and FPR refers to False Positive Rate. Next section describes a special type of CNN designed for a specific imagenet dataset having 1000 classes.

1.4.2 CNN based disease classification The Convolution layers of CNN architecture help to extract features of images such as edges, points texture etc. The kernels in convolution layers can be of size 3 × 3, 5 × 5, 7 × 7, etc. which start with random real values and with help of back-propagation method these kernels attain values suited for a particular training dataset. Max-pooling layers help to reduce the image size by retaining only the important features in the model. This helps to reduce the floating-point operations and helps in increasing efficiency of classification. A Flattened layer follows the convolution layers which converts the 2-D image to 1-D vector. It is followed by few Dense layers having some number of hidden nodes such as 128, 256, 512, 1024, 4096, etc. The Activation function used in current CNN models is mostly ReLU which provides linear outputs for positive values and suppresses the negative values. Dropout layers can also be introduced to increase model performance as it probabilistically drops some nodes in each iteration. The last layer is a softmax layer which indicates the probability of each of the possible classes.

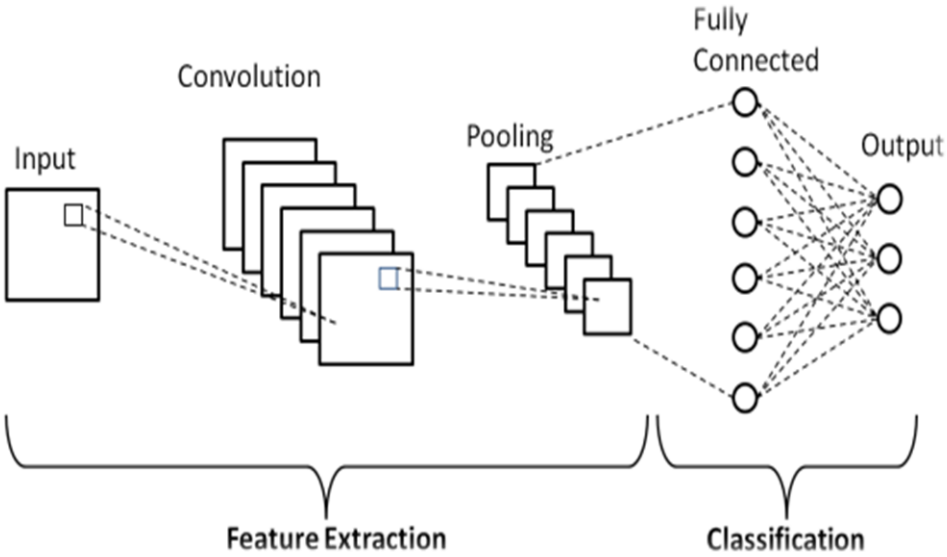
1.4.3 Pre-trained CNN models Instead of designing new CNN architectures for various classification problems in different domains, it is a common practice to utilize the parameters of a model which has been pre-trained for a large sized problem like ILSVRC (ImageNet Large Scale Visual Recognition Competition). This is a challenging 1000 class classification problem. The parameters learned from such pre-trained architectures can be used in other image classification problems by using transfer learning. The popular pre-trained models ResNet50, VGG16, Alex Net, InceptionV3, Mobile Net and Squeeze Net are briefly described below: 1.3.3.1 VGG16 VGG 16 [48] is the CNN architecture designed by Visual Geometry Group (VGG), University of Oxford. Input image size in this model is generally 224 × 224 but can be varied with a minimum size of 48 × 48. Filters used are of size 3 × 3. It has 13 convolution layers with 64, 64, 128, 128, 256, 256, 256, 512, 512, 512, 512, 512 and 512 filters in various convolution layers starting from the input layer. It also has 3 Dense layers with nodes equal to 4096, 4096 and 1000 nodes.

Activation function used is ReLu in all layers. It has pre-defined ImageNet weights which have been trained on ILSVRC. Its pictorial representation is given in Figure 1.1

Defining the CNN (Convolutional Neural Network)

CNN uses filters on the pixels of any image to learn detailed patterns compared to global patterns with a traditional neural network. To create CNN, we have to define:

1. **A convolutional Layer:** Apply the number of filters to the feature map. After convolution, we need to use a relay activation function to add non-linearity to the network.
2. **Pooling Layer:** The next step after the Convention is to downsampling the maximum facility. The objective is to reduce the mobility of the feature map to prevent overfitting and improve the computation speed. Max pooling is a traditional technique, which splits feature maps into subfields and only holds maximum values.
3. **Fully connected Layers:** All neurons from the past layers are associated with the other next layers. The CNN has classified the label according to the features from convolutional layers and reduced with any pooling layer.



CNN Architecture

- **Convolutional Layer:** It applies 14 5x5 filters (extracting 5x5-pixel sub-regions),
- **Pooling Layer:** This will perform max pooling with a 2x2 filter and stride of 2 (which specifies that pooled regions do not overlap).
- **1,764 neurons,** with the dropout regularization rate of 0.4 (where the probability of 0.4 that any given element will be dropped in training)
- **Dense Layer (Logits Layer):** There are ten neurons, one for each digit target class (0-9).

Important modules to use in creating a CNN:

1. Conv2d (). Construct a two-dimensional convolutional layer with the number of filters, filter kernel size, padding, and activation function like arguments.
2. max_pooling2d (). Construct a two-dimensional pooling layer using the max-pooling algorithm.
3. Dense (). Construct a dense layer with the hidden layers and units

We can define a function to build CNN.

Let's see in detail how to construct every building block before wrapping everything in the function.

Step 2: Input layer

```
#Input layer
def cnn_model_fn(mode, features, labels):
input_layer = tf.reshape(tensor= features["x"],shape=[-1, 26, 26, 1])
```

We need to define a tensor with the shape of the data. For that, we can use the **module tf.reshape**. In this module, we need to declare the tensor to reshape and to shape the tensor. The first argument is the feature of the data, that is defined in the argument of a function.

A picture has a width, a height, and a channel. The **MNIST** dataset is a monochromic picture with the **28x28** size. We set the batch size into -1 in the shape argument so that it takes the shape of the features ["x"]. The advantage is to tune the batch size to hyperparameters. If the batch size is 7, the tensor feeds **5,488** values (**28 * 28 * 7**).

Step 3: Convolutional Layer

```
# first CNN Layer
conv1 = tf.layers.conv2d(
 inputs= input_layer,
 filters= 18,
 kernel_size= [7, 7],
 padding="same",
 activation=tf.nn.relu)
```

The first convolutional layer has 18 filters with the kernel size of 7x7 with equal padding. The same padding has both the output tensor and input tensor have the same width and height. TensorFlow will add zeros in the rows and columns to ensure the same size. We use the Relu activation function. The output size will be [28, 28, and 14].

Step 4: Pooling layer

The next step after the convolutional is pooling computation. The pooling computation will reduce the extension of the data. We can use the module max_pooling2d with a size of 3x3 and stride of 2. We use the previous layer as input. The output size can be [batch_size, 14, 14, and 15].

```
##first Pooling Layer
pool1 = tf.layers.max_pooling2d (inputs=conv1,
pool_size=[3, 3], strides=2)
```

Step 5: Pooling Layer and Second Convolutional Layer

The second CNN has exactly 32 filters, with the output size of [batch size, 14, 14, 32]. The size of the pooling layer has the same as ahead, and output shape is [batch size, 14, 14, and18].

```
conv2 = tf.layers.conv2d(
    inputs=pool1,
    filters=36,
    kernel_size=[5, 5],
    padding="same",
    activation=tf.nn.relu)
pool2 = tf.layers.max_pooling2d (inputs=conv2, pool_size=[2, 2],strides=2).
```

Step6: Fully connected (Dense) Layer

We have to define the fully-connected layer. The feature map has to be compressed before to be combined with the dense layer. We can use the module reshape with a size of **7*7*36**.

The dense layer will connect **1764** neurons. We add a ReLu activation function and can add a Relu activation function. We add a dropout regularization term with a rate of 0.3, meaning 30 percent of the weights will be 0. The dropout takes place only along the training phase. The **cnn_model_fn()** has an argument mode to declare if the model needs to trained or to be evaluate.

```
pool2_flat = tf.reshape(pool2, [-1, 7 * 7 * 36])
dense = tf.layers.dense(inputs=pool2_flat, units=7 * 7 * 36, activation=tf.nn.relu)
dropout = tf.layers.dropout(
inputs=dense, rate=0.3, training=mode == tf.estimator.ModeKeys.TRAIN)
```

Step 7: Logits Layer

Finally, we define the last layer with the prediction of model. The output shape is equal to the batch size 12, equal to the total number of images in the layer.

```
#Logit Layer
logits = tf.layers.dense(inputs=dropout, units=12)
```

We can create a dictionary that contains classes and the possibility of each class. The module returns the highest value with tf.argmax () if the logit layers. The softmax function returns the probability of every class.

```
predictions= {
 # Generate predictions
"classes":tf.argmax(input=logits, axis=1),
"probabilities":tf.nn.softmax (logits, name="softmax_tensor")}
```

We only want to return the dictionary prediction when the mode is set to prediction. We add these codes to display the predictions.

1.1 Motivation/History

The motivation for developing crop disease detection projects stems from several pressing challenges facing global agriculture:

1. **Food Security**: With the world's population projected to reach 9 billion by 2050, ensuring food security is a paramount concern. Crop diseases pose a significant threat to agricultural productivity and food supply, making early detection and management crucial for meeting the growing demand for food.
2. **Yield Losses**: Crop diseases can cause substantial yield losses, impacting farmer livelihoods and exacerbating food shortages. Timely detection allows for prompt intervention, reducing the extent of yield losses and ensuring sustainable production.
3. **Environmental Sustainability**: Conventional methods of disease management often rely on the indiscriminate use of chemical pesticides, which can have adverse effects on the environment, including soil degradation, water pollution, and harm to non-target organisms. By enabling targeted and precise intervention, crop disease detection projects promote environmentally friendly farming practices.
4. **Economic Viability**: Crop diseases can result in significant economic losses for farmers, affecting their income and livelihoods. By reducing yield losses and optimizing resource use, disease detection technologies contribute to the economic viability of agricultural enterprises, particularly for smallholder farmers in developing countries.
5. **Human Health**: Some crop diseases can have indirect effects on human health through the contamination of food crops with toxins or pathogens. Early detection and management of such diseases contribute to safer food production systems and public health.

In summary, the motivation for crop disease detection projects lies in addressing these interconnected challenges to ensure sustainable agriculture, enhance food security, protect the environment, and safeguard farmer livelihoods. By harnessing technology and innovation, these projects aim to empower farmers with the tools and knowledge needed to manage crop diseases effectively and sustainably in a rapidly changing world.

1.2 Problem Definition

Despite advancements in agriculture, crop diseases continue to pose a significant threat to global food security, leading to substantial yield losses and economic impacts for farmers. Traditional methods of disease detection rely on manual observation and diagnosis, which are often time-consuming, labor-intensive, and prone to inaccuracies. There is a pressing need for automated and accurate crop disease detection systems that can identify diseases early, enable targeted intervention, and minimize the negative impacts on crop yields and farmer livelihoods. Additionally, with the increasing challenges posed by climate change and the emergence of new disease strains, there is a demand for robust and adaptable detection technologies capable of addressing evolving disease dynamics. The goal of this project is to develop an efficient, reliable, and scalable crop disease detection solution leveraging cutting-edge technologies such as artificial intelligence, machine learning, and computer vision. By addressing these challenges, the project aims to enhance agricultural sustainability, promote food security, and empower farmers with the tools and knowledge needed to manage crop diseases effectively in a rapidly changing agricultural landscape.

1.3 Objectives

1. To gather information about various crop diseases.
2. To Study about neural network algorithms.
3. To collect the images of various parts of crops which are infected.
4. To implement CNN algorithm for disease prediction
5. To develop an user interface.
6. To real time detect and monitor the system.

1.4 Social aspects of project

The social aspect of a crop disease detection project encompasses several dimensions that directly impact farmers, communities, and broader society. Here's how the project can address social aspects:

1. **Empowering Farmers**: By providing farmers with access to advanced technologies for crop disease detection, the project empowers them to make informed decisions about crop management, leading to improved yields, reduced losses, and enhanced livelihoods.
2. **Enhancing Food Security**: Effective disease detection helps safeguard crop productivity, contributing to food security at local, regional, and global levels. By reducing yield losses due to

diseases, the project helps ensure a stable food supply, which is crucial for addressing hunger and malnutrition.

3. **Promoting Sustainable Agriculture**: Crop disease detection encourages the adoption of sustainable farming practices by minimizing reliance on chemical pesticides and optimizing resource use. This promotes environmental sustainability by reducing pollution, preserving biodiversity, and protecting ecosystems
4. **Improving Rural Livelihoods**: By enhancing crop productivity and reducing losses, the project contributes to the economic well-being of rural communities. Increased income from agriculture can stimulate local economies, create employment opportunities, and improve living standards in rural areas.

the social aspect of a crop disease detection project extends beyond technological innovation to encompass broader impacts on farmers' livelihoods, community resilience, food security, environmental sustainability, and social equity. By addressing these dimensions, the project contributes to building more resilient, inclusive, and sustainable agricultural systems that benefit society as a whol

1.5 Organization of report

In this report, Chapter 1 consists of the needs to develop this project and the things are needed to be made efficient as a result and all the introductory part to this project. Chapter 2 contains the past work done on these technologies to be used in the project and the surveys done on the related topics. In Chapter 3, the roles of the technologies used to develop this project. Chapter 4 consist of the working design and flow of the project. Chapter 5 consists of the different user interfaces and how they are implemented to interact with authorized user. Chapter 6 consists of conclusion and future scope of this project.

2. LITERATURE SURVEY

Many studies have been conducted to find an ideal solution to the problem of crop disease detection by creating techniques that can assist in identifying crops in an agricultural environment. This section will provide the most recently reviewed studies on CNN's applicability in the broad field of agriculture; this section includes papers from peer reviewed articles that use CNN methods and plant datasets. Abade et al. [9] reviewed CNN algorithms for the detection of plant diseases. The authors studied 121 papers that were published between 2010 and 2019. Plant Village was selected as the most widely used dataset, while TensorFlow was identified as the most frequently used framework in this review.

In the study author [1] proposed show utilizes these progressed highlights to progress the precision and productivity of plant recognizable proof. By leveraging GLCM for surface investigation, lacunarity for spatial design acknowledgment, and Shen highlights for shape representation, the show offers a comprehensive approach to plant species classification. In general, the paper contributes to the progression of plant recognizable proof innovation by presenting a novel demonstrate that coordinating different include extraction strategies. This inquire about has suggestions for different areas, counting farming, natural science, and botany, where precise and proficient plant recognizable proof is significant.

In the [2] author discusses the shortcomings of current techniques as well as the significance of early disease detection in plants. It suggests an approach that uses Support Vector Machine (SVM) classification for precise detection and HSV (Hue, Saturation, Value) attributes to describe leaf properties and illnesses. Preprocessing leaf photos, extracting HSV features to provide color information, and training an SVM classifier to differentiate between healthy and unhealthy leaves are probably the steps in the suggested methodology.

In [3] paper the author explains about the process probably entails training a GAN to produce realistic crowd images, and then training a multiple target regression-based crowd counting model with the synthetic images and the restricted amount of real data.
The efficiency of the suggested strategy in comparison to conventional crowd counting techniques is probably illustrated in the study through experimental results, especially in situations where there is a dearth of available data.

The paper [4] likely introduces a novel approach to image-based search engines that utilizes machine learning techniques. It may propose a system where images are indexed and retrieved based on their visual content, rather than relying solely on textual metadata. The methodology likely involves

training machine learning models, such as convolutional neural networks (CNNs), to extract meaningful features from images. These features are then used to index images and perform similarity searches.

Moreover, Nagaraju et al. [11] also provided a review to find the best datasets, pre- processing approaches, and DL techniques for various plants. They reviewed and analyzed 84 papers on DL's applicability in plant disease diagnosis. They observed that so many DL methods are limited in their ability to analyze original images and that effective model performance necessitates using a suitable pre-processing technique.

Kamilaris et al[12] found that DL approaches were used to solve various agricultural challenges. According to the study, DL methods performed better than standard image processing techniques. Fernandez-Quintanilla et al. [13] evaluated weed-monitoring technologies in crops. They focused on weed monitoring devices in agricultural fields that were both remote sensed and ground-based. Weed monitoring is critical for weed control, according to them. They predicted that data acquired by various sensors would be saved in a public cloud and used in appropriate contexts at the optimal time.

Lu et al. [14] introduced a review for plant disease classification using a CNN. They evaluated the significant problems and solutions of CNN used for plant disease classification and DL criteria in plant disease classification. They discovered that additional research with more complex datasets was required to obtain a more satisfactory result.

Golhani et al. [15] presented a review paper on hyperspectral data for plant leaf disease identification, highlighting existing problems and potential prospects. They also presented NN approaches for SDI development in a short time. They discovered that, as long as SDIs remain relevant for proper crop protection, they must be tested on various hyperspectral sensors at the plant leaf scale.

Bangari et al. [16] presented a review on disease detection using CNN, focusing on potato leaf disease. They reviewed several papers and concluded that convolutional neural networks work better at detecting the disease. They also identified that CNN contributed significantly to the maximum possible accuracy for disease identification.

3. SYSTEM ANALYSIS

Technologies Used (Java EE)

- Deployment platform: Microsoft Azure Cloud
- Application Server: Apache Tomcat
- Technology: Java EE
- Development Tools (Server side): Servlets, JSP, Java Beans & REST API
- MVC framework: Spring boot
- Database technologies: MySQL CloudDB, JDBC
- Web Development: HTML5, JavaScript, AJAX, Bootstrap CSS, XML, JSON
- IDE: Eclipse 2023.12

Technologies used (python)

- Application Server: Apache HTTP Server (XAMPP)
- ML tools: Python, Scikit Learn, CSV datasets
- Development Tool: Visual Studio Code / JupyterLab

Brief description of the technologies used in the project

Enterprise Java (JakartaEE)

Java Platform, Enterprise Edition (Java EE) is the standard in community-driven enterprise software. Java EE is developed using the Java Community Process, with contributions from industry experts, commercial and open-source organizations, Java User Groups, and countless individuals. Each release integrates new features that align with industry needs, improves application portability, and increases developer productivity.

The Java EE stands for Java Enterprise Edition, which was earlier known as J2EE and is currently known as Jakarta EE. It is a set of specifications wrapping around Java SE (Standard Edition). The

Java EE provides a platform for developers with enterprise features such as distributed computing and web services. Java EE applications are usually run on reference run times such as microservers or application servers. Examples of some contexts where Java EE isused are e-commerce, accounting, banking information systems.

JSP

Java Server Pages (JSP) is a technology that allows developers to create dynamic web pages using a combination of HTML, XML, and Java code. JSP pages are executed on a web server, and the resulting output is sent to the client's web browser. JSP provides a way to easily access Java code and objects from within a web page, simplifying the creation of dynamic web pages. JSP pages are typically used in conjunction with Java servlets, which handle data processing andclient requests. JSP is part of the Java EE platform and is supported by most web servers and servlet containers.

JSP simplifies the process of developing web applications by providing a way to separate the presentation layer (HTML) from the business logic (Java code). This separation of concerns makes it easier to maintain and update web applications. Developers can use JSP tags to include Java code, perform flow control, access JavaBeans components, and more. Additionally, JSP supports standard tag libraries (JSTL) and custom tag libraries, which further enhance its capabilities for building complex web applications.

JDBC

JDBC or Java Database Connectivity is a Java API to connect and execute the query with the database. It is a specification from Sun microsystems that provides a standard abstraction(API orProtocol) for java applications to communicate with various databases. It provides the language with java database connectivity standards. It is used to write programs required to access databases. JDBC, along with the database driver, can access databases and spreadsheets. The enterprise data stored in a relational database (RDB) can be accessed with the help of JDBC APIs.

Spring Boot

Spring Boot is the top tool in the industry today for Java applications and services development.It is also the leading tool in job market. Spring Boot is a Spring module that provides the RAD (Rapid Application Development) featureto the Spring Framework1. It is an open-source Java-based framework used to create a microservice. It makes it easy to create stand-alone, production-grade Spring based applications that you can “just run” without much configuration.

Key Features:

1. **Auto-configuration**: Spring Boot automatically configures various components of the application based on dependencies and conventions. It eliminates the need for boilerplate configuration, allowing developers to focus on writing business logic.
2. **Standalone:** Spring Boot applications can be deployed as standalone JAR files, which include an embedded servlet container (such as Tomcat, Jetty, or Undertow). This eliminates the need for external application servers, simplifying deployment and distribution.
3. **Dependency Management:** Spring Boot provides a dependency management mechanism that simplifies the management of library dependencies. It uses a "starter" concept, where dependencies for common tasks (such as web applications, security, data access, etc.) are bundled into pre-configured starter modules.
4. **Spring Boot Actuator**: Actuator is a feature of Spring Boot that provides monitoring and management capabilities for applications. It includes built-in endpoints for monitoring application health, metrics, environment information, and more.
5. **Spring Boot CLI**: The Spring Boot Command Line Interface (CLI) allows developers to quickly create and run Spring Boot applications from the command line. It provides features like auto-restart, Groovy script support, and more.
6. **Integration with Spring ecosystem**: Spring Boot seamlessly integrates with other Spring projects like Spring Framework, Spring Data, Spring Security, and more, allowing developers to leverage the full power of the Spring ecosystem.

Benefits:

1. **Rapid Development**: Spring Boot simplifies the setup and configuration of Spring applications, enabling developers to get started quickly and focus on writing application logic.
2. **Reduced Boilerplate Code**: By providing sensible defaults and conventions, Spring Boot eliminates much of the boilerplate configuration typically required in Spring applications.
3. **Increased Productivity**: With features like auto-configuration, embedded servlet containers, and dependency management, Spring Boot boosts developer productivity and reduces time-to-market for applications.
4. **Easy Integration**: Spring Boot seamlessly integrates with other Spring projects and third-party libraries, making it easy to extend and customize applications as needed.

Apache Tomcat

It is used to host and distribute Java applications across the network. Apache Tomcat (called "Tomcat" for short) is a free and open-source implementation of the Jakarta Servlet, Jakarta Expression Language, and WebSocket technologies. It provides a "pure Java" HTTP web serverenvironment in which Java code can also run. Thus it is a Java web application server, althoughnot a full JEE application server.

Tomcat is developed and maintained by an open community of developers under the auspices ofthe Apache Software Foundation, released under the Apache License 2.0 license.

Spring Data JPA

JPA is a Java specification (Jakarta Persistence API) and it manages relational data in Java applications. To access and persist data between Java object(Plain Old Java object)/ class and relational database, we can use JPA. Upon Object-Relation Mapping (ORM), it follows the mechanisms. It has the runtime Entity Manager API and it is responsible for processing queriesand transactions on the Java objects against the database. The main highlight is it uses JPQL (Java Persistent Query Language) which is platform-independent.

Advantages of Using JPA

- No need to write DDL/DML queries, instead we can map by using XML/annotations.
- JPQL is used and since it is platform-independent, we no need to depend on any native SQL table. Complex expressions and filtering expressions are all handled via JPQL only.
- Entity can be partially stored in one database like MySQL and the rest can be in Graph database Management System.
- Dynamic generation of queries is possible.
- Integration with Spring framework is easier with a custom namespace.

REST API

A REST API (Representational State Transfer Application Programming Interface) is a type of web service that allows communication between different software systems over the internet. It follows the principles of REST, which include statelessness, uniform interface, client-server architecture, and the use of standard HTTP methods for performing operations on resources. Here's an overview of key components and characteristics of a REST API:

1. Resources:

- Resources are the entities that the API exposes, such as users, products, or documents.
- Each resource is identified by a unique URI (Uniform Resource Identifier), which serves as its address on the web.

2. HTTP Methods:

- RESTful APIs use standard HTTP methods (GET, POST, PUT, DELETE, PATCH, etc.) to perform operations on resources.
  - GET: Retrieve resource data.
  - POST: Create a new resource.
  - PUT: Update an existing resource or create if not exists.
  - DELETE: Delete a resource.
  - PATCH: Partially update a resource.

2. Representation:

- Resources are represented in a specific format such as JSON (JavaScript Object Notation) or XML (extensible Markup Language).
- JSON is widely used due to its simplicity, readability, and ease of parsing by most programming languages.

4. Statelessness:

- REST APIs are stateless, meaning that each request from a client to the server must contain all the necessary information to process the request.

- The server does not store any client state between requests, which improves scalability and reliability.

5. Uniform Interface:

- A uniform interface simplifies communication between clients and servers by defining a standard way to interact with resources.
- It includes the use of standard HTTP methods, resource identifiers (URIs), and representations.

6. Client-Server Architecture:

- REST APIs follow a client-server architecture, where clients make requests to servers to perform operations on resources.
- This separation of concerns allows for scalability and flexibility in designing and deploying systems.

7. Response Codes:

- HTTP status codes are used to indicate the outcome of a request (e.g., 200 for successful, 404 for not found, 500 for server error).
- They provide valuable information to clients about the status of their requests.

8. Authentication and Authorization:

- REST APIs often use authentication mechanisms such as API keys, OAuth, or JWT (JSON Web Tokens) for secure access to resources.
- Authorization mechanisms control which users or clients have permission to access specific resources.

Overall, REST APIs provide a flexible and scalable approach to building distributed systems by leveraging standard protocols and architectural principles. They are widely used in web development, mobile app development, and integration of different software systems.

JSON

JSON, short for JavaScript Object Notation, serves as a lightweight and easily readable data interchange format, drawing inspiration from JavaScript's object literal syntax. Its structure revolves around key-value pairs, enabling straightforward representation of complex data structures. With support for various data types like strings, numbers, booleans, arrays, objects, and null, JSON remains highly versatile. Data within JSON is organized into key-value pairs, where keys are strings enclosed in double quotes, separated by commas, and enclosed in curly braces for objects. Arrays, on the other hand, are ordered collections enclosed in square brackets. JSON finds extensive use cases, including data exchange between web servers and clients, configuration file storage, and structured data transmission in databases and web services. Its straightforward syntax and broad language support make it a favored choice for developers across diverse domains. Additionally, most programming languages offer built-in functionalities or libraries for seamless parsing and generation of JSON data, ensuring its widespread adoption and integration into various applications.

JSON (JavaScript Object Notation) is an open standard file format and data interchange formatthat uses human-readable text to store and transmit data objects consisting of attribute–value pairs and arrays (or other serializable values). It is a common data format with diverse uses in electronic data interchange, including that of web applications with servers. JSON is a language-independent data format. It was derived from JavaScript, but many modernprogramming languages include code to generate and parse JSON-format data. JSON filenamesuse the extension. json.

JSP simplifies the process of developing web applications by providing a way to separate the presentation layer (HTML) from the business logic (Java code). This separation of concerns makes it easier to maintain and update web applications. Developers can use JSP tags to include Java code, perform flow control, access JavaBeans components, and more. Additionally, JSP supports standard tag libraries (JSTL) and custom tag libraries, which further enhance its capabilities for building complex web applications.

MySQL

MySQL is an open-source relational database management system (RDBMS). Its name is a combination of "My", the name of co-founder Michael Widenius's daughter My, and "SQL", the acronym for Structured Query Language. A relational database organizes data into one or more data tables in which data may be related to each other; these relations help structure the data. SQL is a language programmers use to create, modify and extract data from the relational database, as well as

control user access to the database. In addition to relational databases and SQL, an RDBMS like MySQL works with an operating system to implement a relational database in a computer's storage system, manages users, allows for network access and facilitatestesting database integrity and creation of backups. MySQL is free and open-source software under the terms of the GNU General Public License,and is also available under a variety of proprietary licenses. MySQL was owned and sponsored by the Swedish company MySQL AB, which was bought by Sun Microsystems (now Oracle Corporation). In 2010, when Oracle acquired Sun, Widenius forked the open-source MySQL project to create MariaDB.

Features:

1. **Relational Database:** MySQL follows the relational model and stores data in tables consisting of rows and columns, with relationships established between tables using foreign keys.
2. **SQL Support**: MySQL supports the Structured Query Language (SQL), providing a comprehensive set of SQL commands for querying, updating, and managing databases.
3. **ACID Compliance**: MySQL ensures data integrity and consistency by adhering to the ACID (Atomicity, Consistency, Isolation, Durability) properties, which guarantee reliable transaction processing.
4. **Scalability:** MySQL is designed to handle large volumes of data and can scale horizontally (across multiple servers) or vertically (by upgrading hardware resources) to accommodate growing workloads.
5. **High Performance**: MySQL is known for its high performance and low latency, achieved through features like indexing, query optimization, and efficient storage engines.
6. **Replication and Clustering**: MySQL supports replication, allowing data to be copied asynchronously to multiple servers for redundancy and fault tolerance. It also supports clustering for load balancing and high availability.
7. **Security:** MySQL provides robust security features, including user authentication, access control, encryption, and auditing, to protect sensitive data from unauthorized access and malicious attacks.

Eclipse

Eclipse is a popular integrated development environment (IDE) for Java and other programming languages. It provides tools for editing, debugging, testing, and deploying software applications.

Features and Capabilities:

1. **Modular Architecture**: Eclipse is built on a modular architecture called the Eclipse Rich Client Platform (RCP), which allows developers to extend and customize the IDE with plugins.
2. **Language Support**: While Eclipse is primarily known for Java development, it supports various other programming languages such as C/C++, Python, PHP, and more through plugins.
3. **Plugin Ecosystem:** Eclipse has a vast ecosystem of plugins, both official and third-party, which extend its functionality for different purposes. These plugins cover areas like database integration, web development, modeling, testing, and more.
4. **Integrated Development Environment (IDE):** Eclipse provides a comprehensive IDE for software development, including features like code editing, debugging, testing, version control, and project management.
5. **User Interface Customization**: Eclipse allows users to customize the user interface according to their preferences, including layouts, themes, and keyboard shortcuts.
6. **Collaboration Tools**: Eclipse supports collaboration features such as code sharing, code review, and integration with collaboration platforms like GitHub and GitLab.
7. **Extensibility**: Eclipse is highly extensible, allowing developers to create their own plugins to add new features or enhance existing ones.

Machine Learning

Machine learning is a field of study that gives computers the ability to learn without being explicitly programmed. It is a subfield of artificial intelligence (AI) and computer science which focuses on the use of data and algorithms to imitate the way that humans learn, gradually improving its accuracy.

In other words, machine learning is a method of teaching computers to learn from data, without being explicitly programmed. It is used in many applications such as image recognition, speech recognition, natural language processing and more.

There are primarily three types of machine learning: supervised, unsupervised, and reinforcement learning.

Supervised learning: In this type of machine learning, the algorithm is trained on labeled data. The algorithm learns to predict the output from the input data2.

Unsupervised learning: In this type of machine learning, the algorithm is trained on unlabeled data. The algorithm learns to find patterns in the input data without any guidance2.

Reinforcement learning: In this type of machine learning, the algorithm learns by interacting with its environment. The algorithm learns to take actions that maximize a reward signal

There are many benefits of machine learning. Here are some of them:

- Easily identifies trends and patterns: Machine learning can review large volumes of data and discover specific trends and patterns that would not be apparent to humans.
- No human intervention needed (automation): With machine learning, you don't need to babysit your project every step of the way.
- Continuous Improvement: Machine learning models can continuously learn and improve over time.
- Handling multi-dimensional and multi-variety data: Machine learning algorithms can handle a variety of data types, including structured and unstructured data.
- Wide Applications: Machine learning is used in many applications such as image recognition, speech recognition, natural language processing and more.

There are many popular machine learning algorithms. Here are some of them:

- Linear Regression: Linear regression is a statistical method that allows us to study the relationship between two continuous variables.
- Logistic Regression: Logistic regression is a statistical method used to analyze a dataset in which there are one or more independent variables that determine an outcome.
- Decision Tree: A decision tree is a tree-like model of decisions and their possible consequences.
- Random Forest: Random forest is an ensemble learning method for classification, regression and other tasks that operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees.

- K-Nearest Neighbors: K-nearest neighbors (KNN) is a type of instance-based learning or lazy learning where the function is only approximated locally and all computation is deferred until classification.

Python for Machine Learning

Python is one of the most popular programming languages for machine learning. It has many libraries and frameworks that make it easy to work with machine learning algorithms. Some of the popular libraries for machine learning in Python are:

Scikit-learn: Scikit-learn is a free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, k-means and DBSCAN.

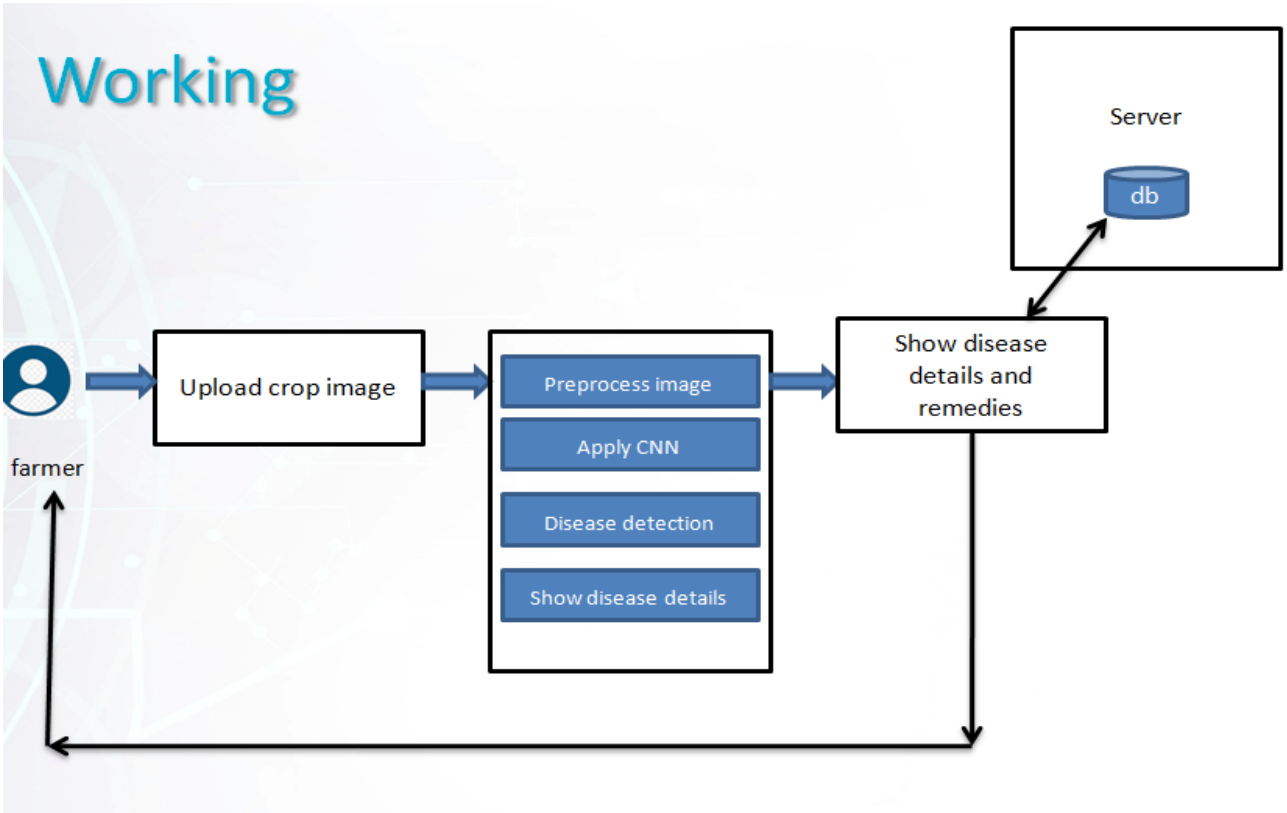
TensorFlow: TensorFlow is an open-source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library and is also used for machine learning applications such as neural networks.

Keras: Keras is an open-source software library for neural networks written in Python. It is capable of running on top of TensorFlow, Microsoft Cognitive Toolkit, Theano, or PlaidML.

PyTorch: PyTorch is an open-source machine learning library based on the Torch library. It is primarily developed by Facebook's AI Research lab.

4. SYSTEM DESIGN

In this project, we proposed crop disease detection system using CNN algorithm. We have developed an online web application in which we have developed admin panel. From admin panel admin will register different crops, crop diseases and upload image for crop diseases. The uploaded images will be preprocessed and saved on python server. Preprocessed images will be sent to CNN algorithm for features extraction. Using Extracted features, model will be trained and saved on python server. At the time of disease prediction, we will load model and predict disease. This technique will reduce execution time required for prediction.



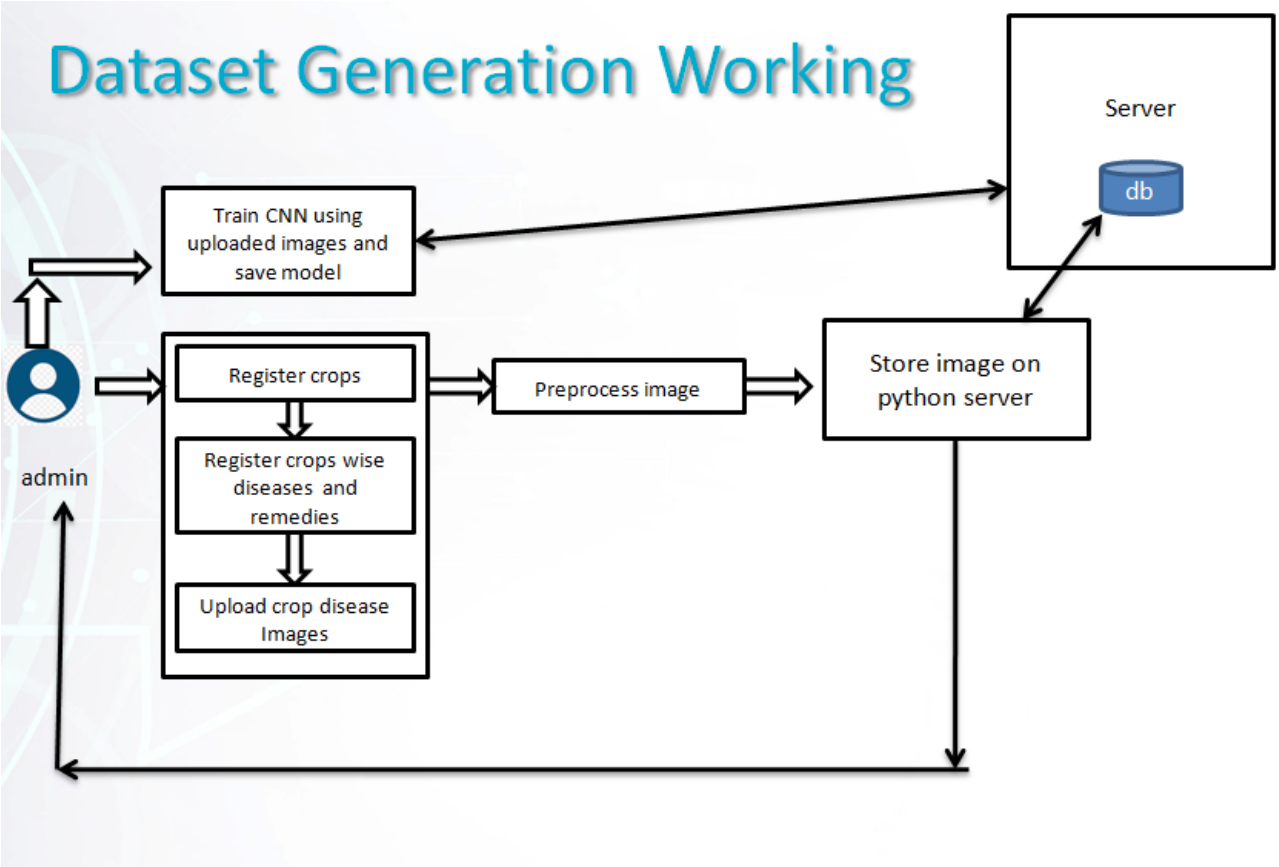


Image augmentation

Image augmentation is a powerful technique that artificially expands the available dataset by creating variations of existing images. It helps improve the performance of deep learning models, especially when the amount of labeled data is limited. Image augmentation is a technique commonly used in computer vision and deep learning applications, particularly in tasks like image classification, object detection, and segmentation. It involves generating new training samples by applying a variety of transformations to existing images. This process helps in diversifying the training dataset, improving the robustness and generalization of the trained models. Here's an overview of image augmentation techniques:

1. Geometric Transformations:

- Rotation: Rotating the image by a certain angle, which helps in making the model invariant to rotation variations.
- Translation: Shifting the image horizontally and vertically, simulating changes in viewpoint.
- Scaling: Resizing the image to a different scale, capturing variations in object size.
- Shearing: Skewing the image along the x or y-axis, mimicking perspective changes.

2. Color and Contrast Adjustments:

- Brightness Adjustment: Changing the overall brightness of the image.
- Contrast Adjustment: Altering the difference in intensity between the bright and dark areas.
- Saturation Adjustment: Modifying the intensity of colors in the image.
- Hue Adjustment: Shifting the hue of the image, changing the colors.

3. Spatial Transformations:

- Flip: Flipping the image horizontally or vertically, capturing mirror variations.
- Crop: Cropping a portion of the image, focusing on specific regions of interest.
- Padding: Adding zero or constant value pixels around the image borders to adjust its size.

4. Noise Addition:

- Gaussian Noise: Adding random Gaussian noise to the image.
- Salt and Pepper Noise: Introducing random black and white pixels to simulate sensor noise.
- Speckle Noise: Multiplying the image with random noise values to mimic speckle interference.

5. Image Blurring:

- Gaussian Blur: Applying a Gaussian filter to the image to reduce high-frequency noise.
- Median Blur: Replacing each pixel's value with the median value in its neighborhood.
- Average Blur: Replacing each pixel's value with the average value in its neighborhood.

6. Advanced Techniques:

- Elastic Deformation: Simulating elastic deformations to the image to capture local distortions.
- Cutout: Randomly removing square patches from the image to encourage the model to focus on different parts.

- Style Transfer: Transferring the style of one image onto another to create new training samples.

These techniques can be applied individually or in combination to generate augmented images. It's important to balance augmentation intensity to prevent overfitting and maintain the semantic integrity of the images. Additionally, augmentation parameters such as rotation angles, brightness levels, and noise magnitudes should be carefully chosen based on the characteristics of the dataset and the requirements of the task at hand.

Here are some key points related to image augmentation

- **Why Image Augmentation?**
  - Deep learning models thrive on large amounts of diverse training data to avoid overfitting.
  - However, collecting a massive dataset manually can be challenging.
  - Image augmentation addresses this by creating additional training samples from existing data.

- **What Is Image Augmentation?**
  - Image augmentation involves altering existing images to generate new ones.
  - These variations maintain the same label as the original image.
  - By introducing diversity, we help the model generalize better.

- **Common Image Augmentation Techniques**:
  - **Rotation**: Rotate images at different angles. A cat remains a cat even when viewed from various perspectives.
  - **Shift**: Change the position of objects within an image. This adds variety and helps the model learn robust features.
  - **Flip**: Horizontally flip images. Cats look the same whether facing left or right.
  - **Zoom**: Crop and resize images to simulate different scales.
  - **Brightness and Contrast**: Adjust pixel values to create variations in lighting conditions.
  - **Noise**: Add random noise to images.
  - **Color Jitter**: Modify color channels (hue, saturation, brightness) to diversify appearance.

  - **Example**:

    Suppose you have an original image of a cat. By applying rotation, shifting, and other transformations, you can generate multiple augmented images of the same cat.

These augmented images become part of your training dataset, enhancing model performance.

- **Benefits**:

  Increased Training Data: Image augmentation expands the dataset without manual collection.

  Generalization: Models trained on augmented data tend to generalize better to unseen examples.

  Remember, image augmentation is a valuable tool for enhancing the quality and diversity of your training data. It allows deep learning models to learn more robust features and perform better on classification tasks.

Image Preprocessing

Image preprocessing refers to a series of operations performed on images before they are fed into a machine learning or computer vision algorithm. Preprocessing is crucial for improving the quality of images, removing noise, enhancing features, and making them suitable for analysis by the model. Here are some common preprocessing techniques used in image processing pipelines:

Resizing:

- Resizing images to a consistent resolution is often necessary to ensure uniformity in the dataset.
- It helps in reducing computational complexity and memory requirements.
- Resizing can be done while preserving aspect ratio or by cropping or padding the images to fit a specific size.

Normalization:

- Normalizing pixel values to a common scale (e.g., [0, 1] or [-1, 1]) helps in stabilizing the training process and accelerating convergence.
- Mean subtraction and standard deviation normalization are common techniques used to center pixel values around zero and scale them to a certain range.

Grayscale Conversion:

- Converting color images to grayscale can simplify the processing pipeline and reduce computational overhead, especially when color information is not crucial for the task.
- Grayscale images also have a single channel, which can reduce the input dimensionality for some models.

Histogram Equalization:

- Histogram equalization adjusts the distribution of pixel intensities in an image, enhancing the contrast and making details more visible.
- It is particularly useful for images with low contrast or uneven lighting conditions.

Noise Reduction:

- Applying filters such as Gaussian blur, median blur, or bilateral filter can help in reducing noise and smoothing out irregularities in the image.
- Noise reduction is especially important for improving the performance of algorithms sensitive to noise, such as edge detection or segmentation.

Edge Detection:

- Edge detection algorithms like Sobel, Prewitt, or Canny can be used to detect edges and boundaries in images.
- Extracting edges can help in identifying important features and reducing the dimensionality of the data.

Feature Extraction:

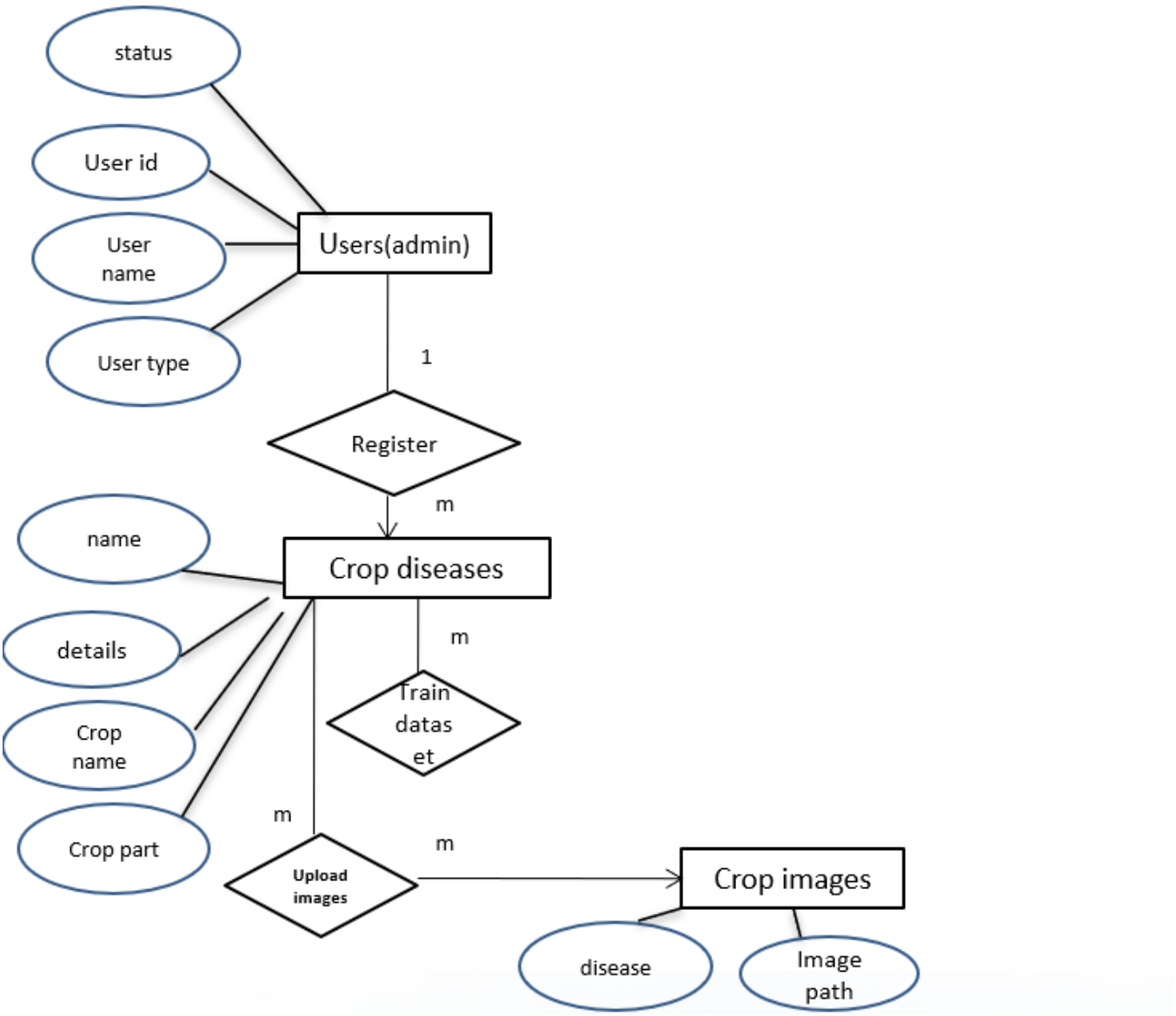
- Techniques like SIFT (Scale-Invariant Feature Transform), SURF (Speeded Up Robust Features), or deep learning-based feature extractors can be used to extract informative features from images.
- Feature extraction is essential for tasks like object recognition, image classification, and image retrieval.

Data Augmentation:

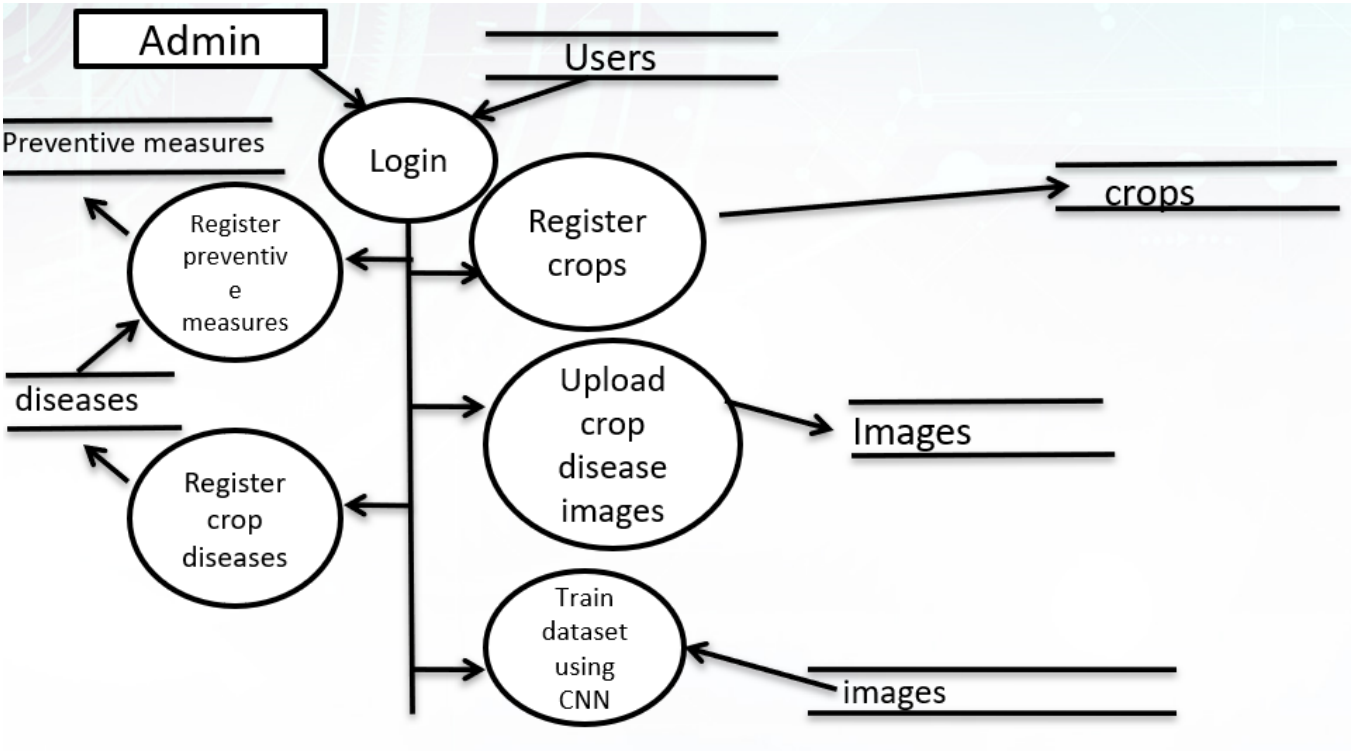
- As mentioned earlier, data augmentation techniques such as rotation, translation, flipping, and cropping can be applied to generate additional training samples and improve the robustness of the model.

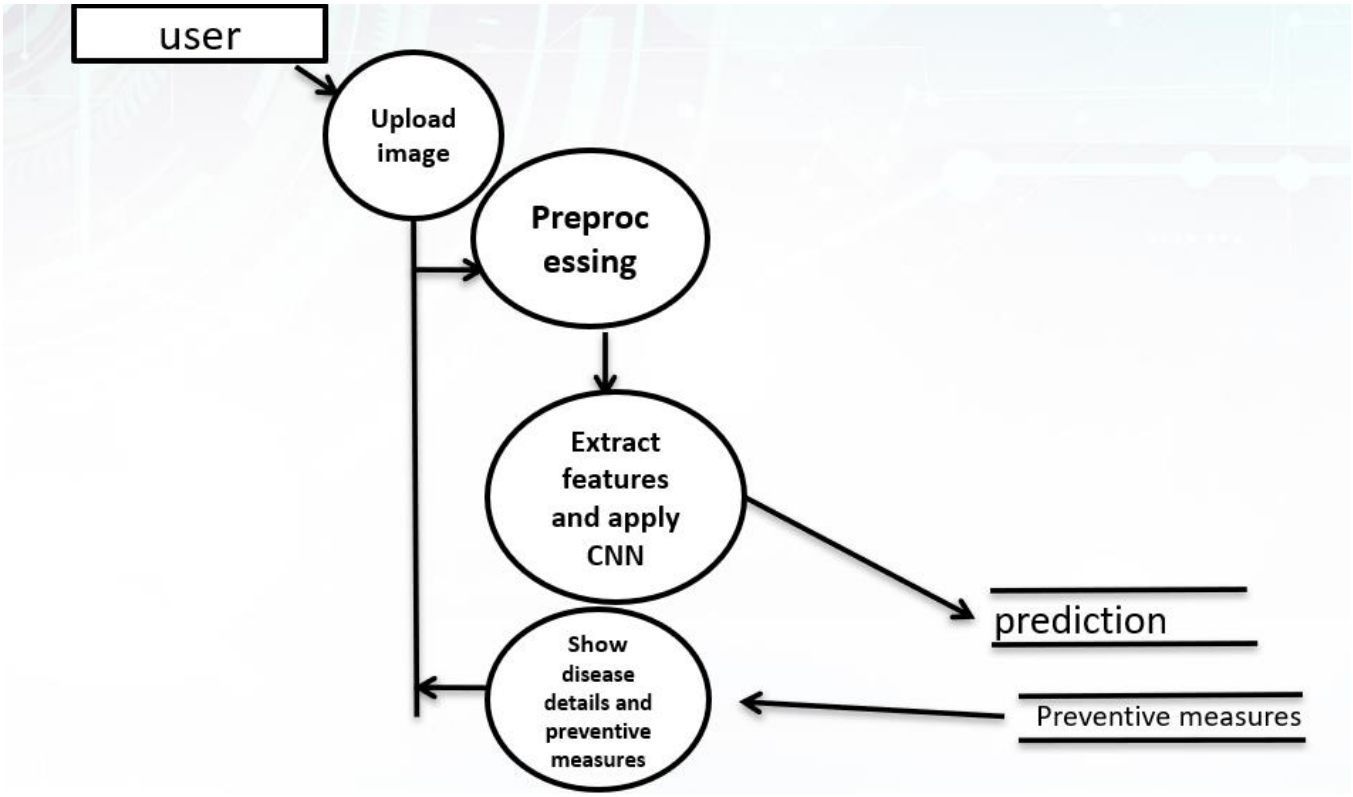
These preprocessing steps can be combined and customized based on the requirements of the specific task and characteristics of the dataset. Proper preprocessing can significantly enhance the performance and efficiency of machine learning and computer vision algorithms by providing clean and informative input data.

ER Diagram


status
User id
User name
User type
Users(admin)
1
Register
m
name
Crop diseases
details
Crop name
Crop part
m
Train datas et
m
m
Upload images
Crop images
disease
Image path

Data Flow Diagram





5. IMPLEMENTATION AND RESULTS

- Home Page:

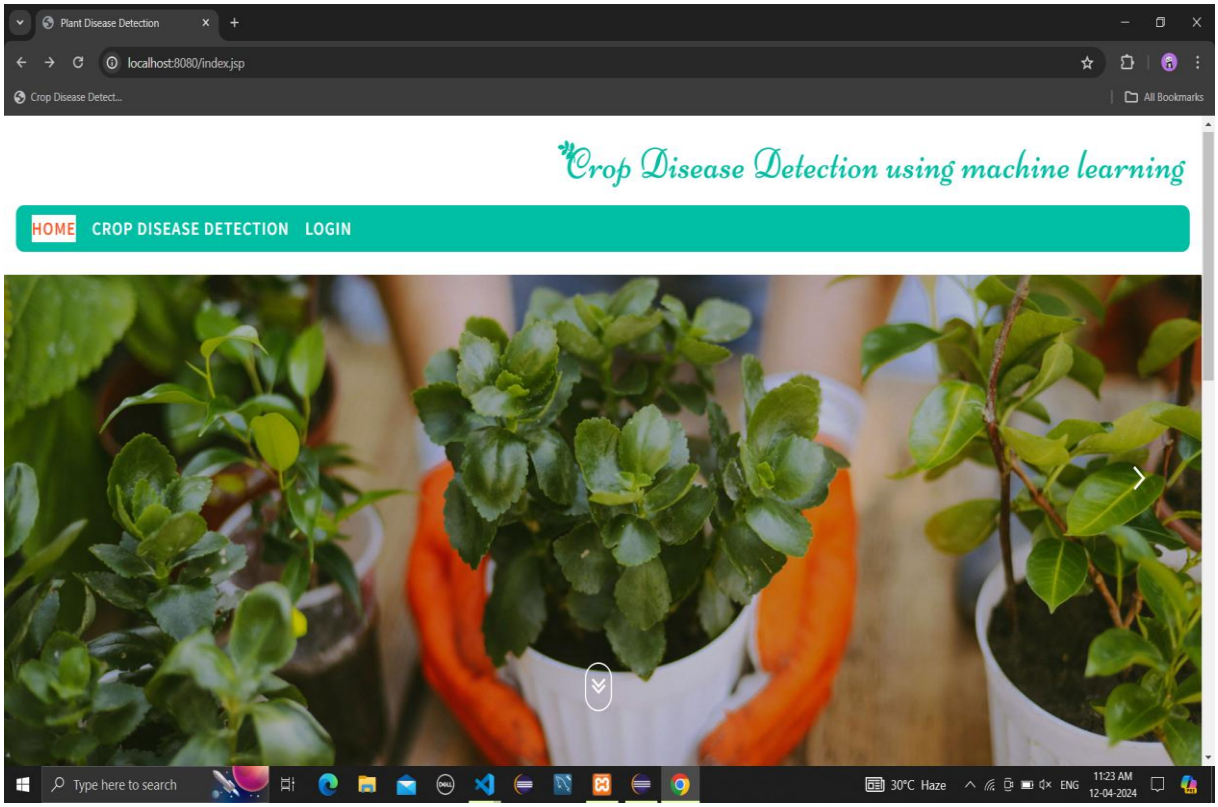


Fig. Home Page

This is the home page of the website. Initially there are three main tabs which are home, crop disease detection and log-in.

In Crop disease detection there will be prediction part of crop diseases in that tab farmer will upload the disease infected image and then prediction of the disease will be takes place and the preventative measures will be suggested. In Log-in tab there will be a username and password field for administrator login

- Log-in Page:

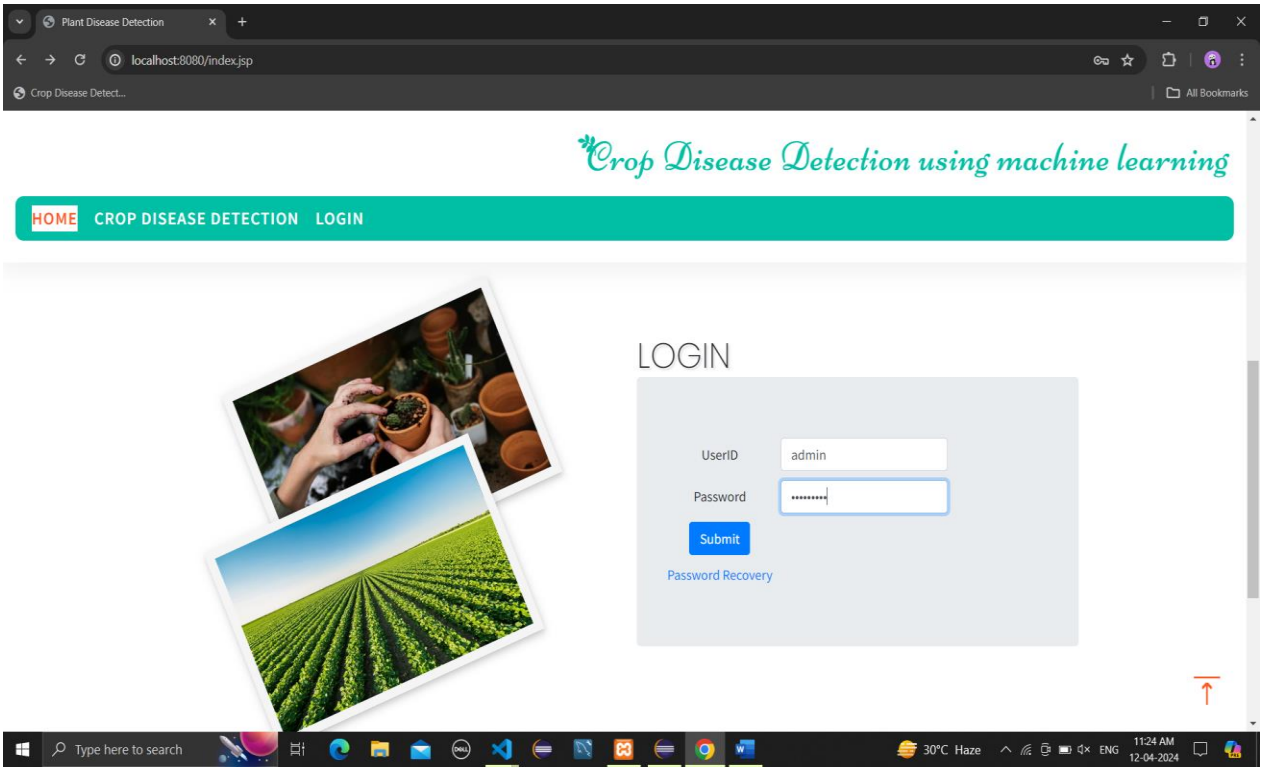


Fig. Log-in Page

This page is mainly for the administrator (admin), admin can log in to the system with the help of username and password,

There is forgot password tab is also present in this page which can be used by the admin if he forgets the password or the username of the system.

- Crop Registration Page:

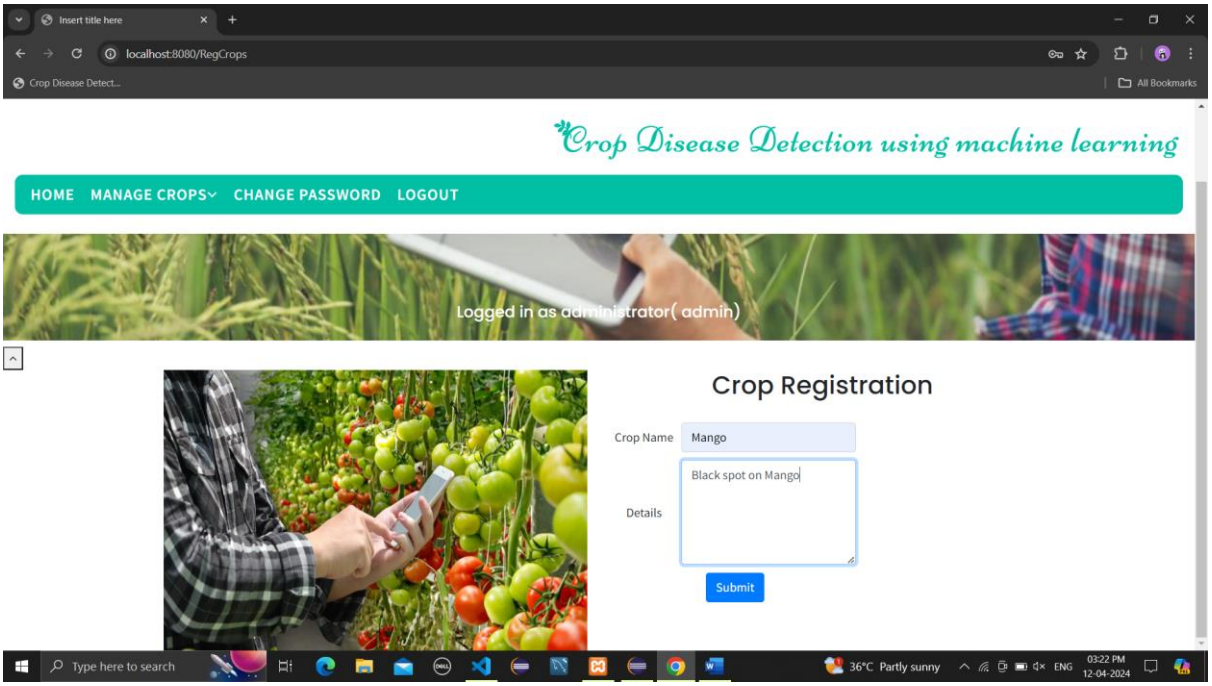


Fig. Crop Registration Page

After the admin will be logged in to the system, he will get various tabs to work with the first one in to register crops.

In this tab admin will able to register the different crops to the database with their details, after that he will get the different options like crop disease registration, upload crop disease images, register preventive measures and train dataset under the manage crops tab.

- Disease Registration Page:

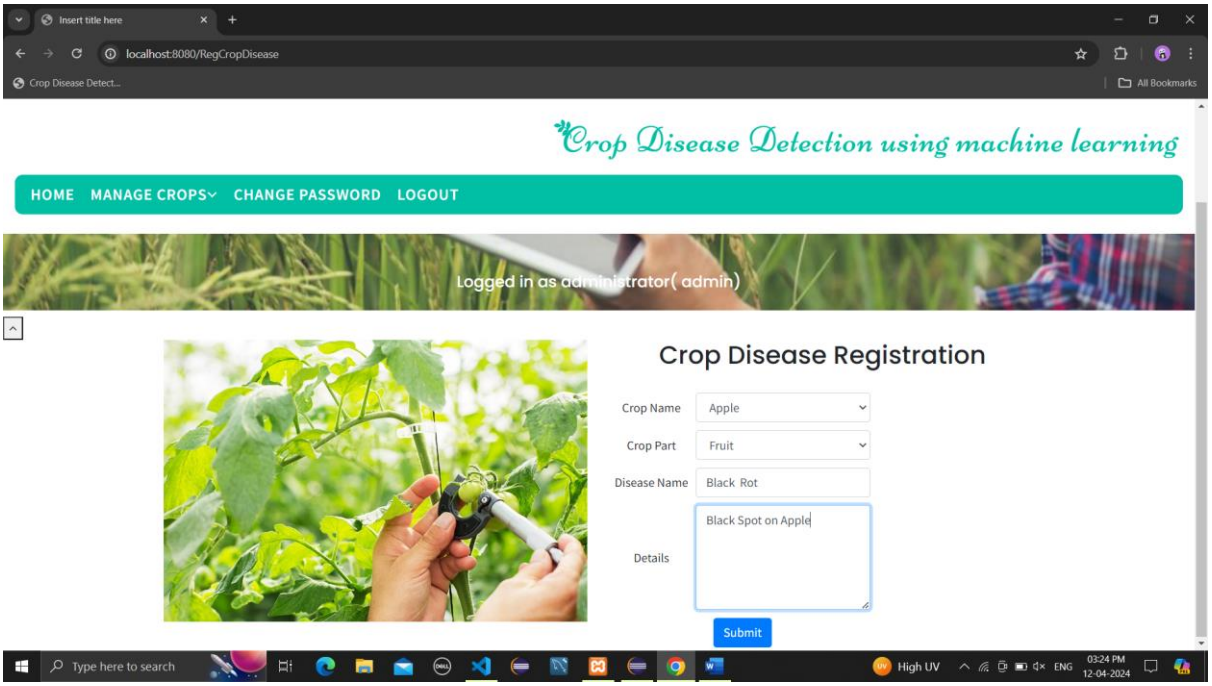


Fig. Crop Disease Registration

After successfully registration of crops, the next step for admin is that to register different crop diseases under that specific disease.

For example, if in first case the user added the crop i.e. tomato then the next step for the admin will be that he has to register various crop diseases related to the tomato.

- Upload Infected and fresh crop images:

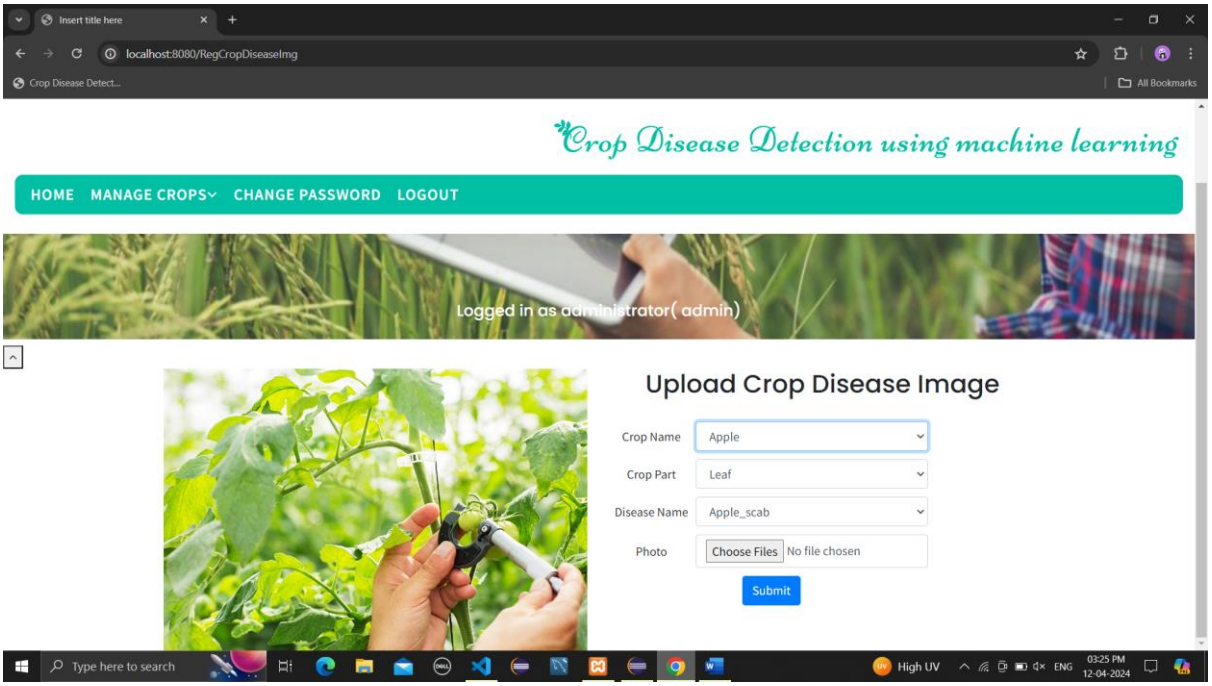


Fig. Upload Crop Disease image

After the registration of crops and crop diseases, the next work for admin will be he have to upload the various crop disease images related to that crop.

In this tab which comes under manage tab category the admin will have to specifically upload the images according to the category which is leaf and fruit and also the admin will have to upload the fresh fruits and leaves images, if in case the user (farmer) uploaded the fresh fruits the system has to recognize it as fresh.

- Register Preventive Measures:

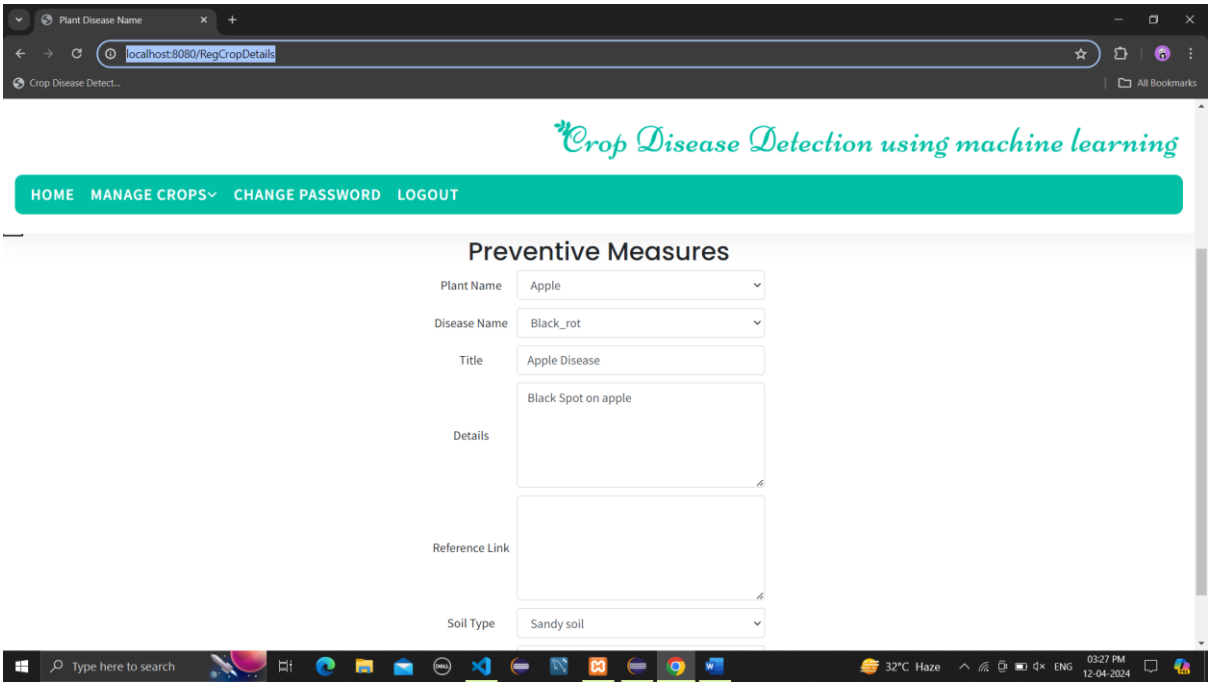


Fig. Preventive Measures

In this tab the admin will have to register preventive measures, this will display after the disease prediction. Preventive measures will be helpful for the farmers to secure their crops and increase their lifespan. In this admin will also have to tell the system about the preventive measures, soil type and weather conditions. This all process will only do at the admin side and after all this wee have to train the dataset according to the particular crop

- Train Dataset:

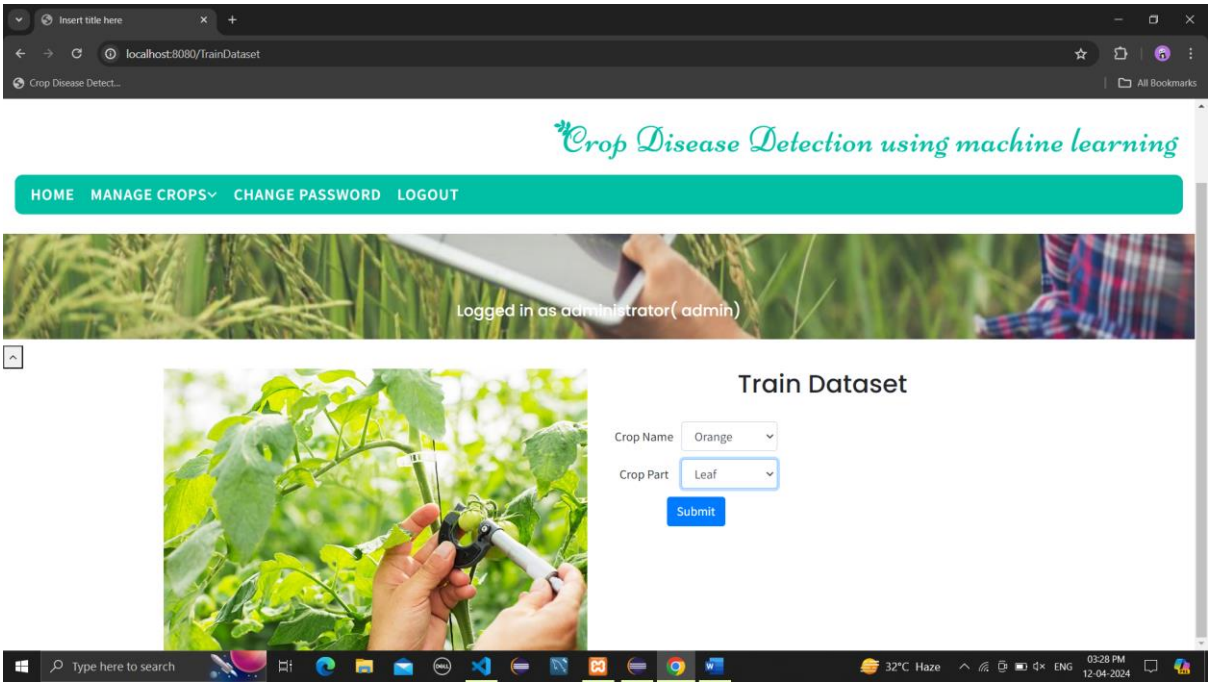


Fig. Train dataset

After the completion of all the steps which comes under the tab of manage crops, the last step will be the train dataset this will take the time according to how many images you have uploaded of that crop category, larger the number of images higher the training time. This step is mainly focused on the images part in which image augmentation, segmentation and image preprocessing done and the images are stored on the python server.

- Change Password:

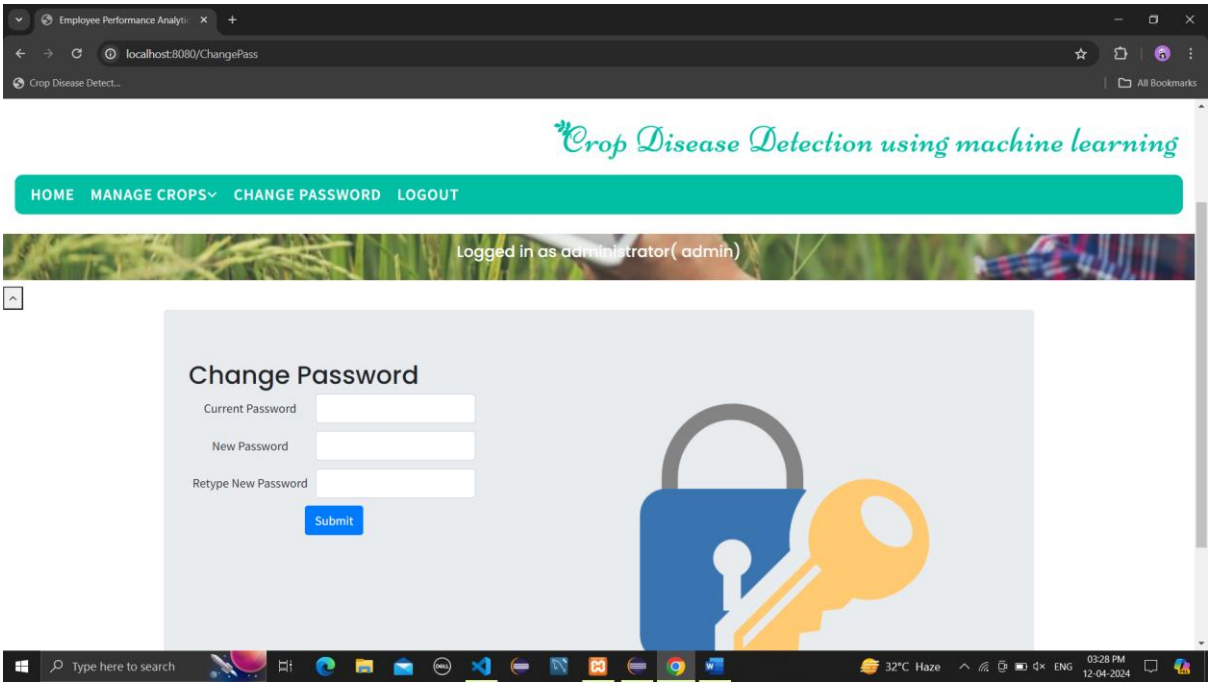


Fig. Change Password

This time is only used by admin to change the password only if he knows the existing password, because in this tab he has to type the current password to change it and if he doesn't know the existing one the he has to use the forgot password tab which is basically present at the home page.

- Disease Prediction:

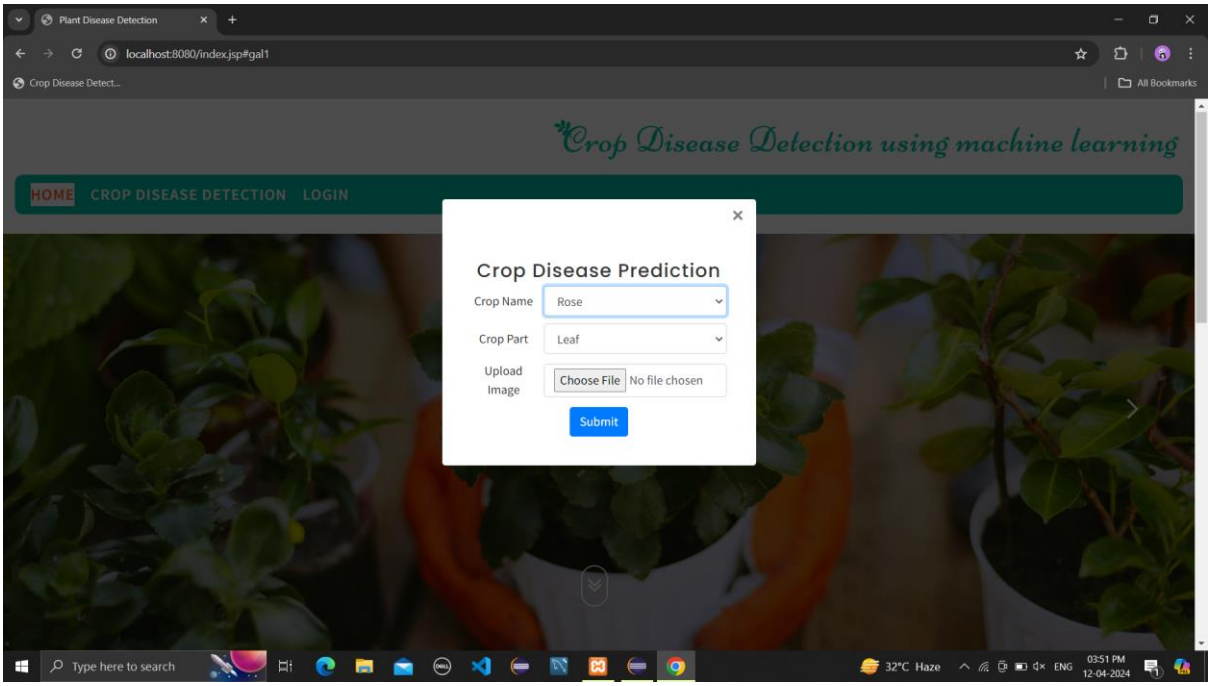


Fig. Disease Prediction

This is the main tab for the farmer in which, farmer have to select the crop name and crop part and then he has to upload the disease infected part image for the further processing, after he clicks the submit button the preprocessing steps are applied to the image and then the important pixels are going to be extracted after the extraction this matrix results are compared with the result stored on the python server.

- Prediction:

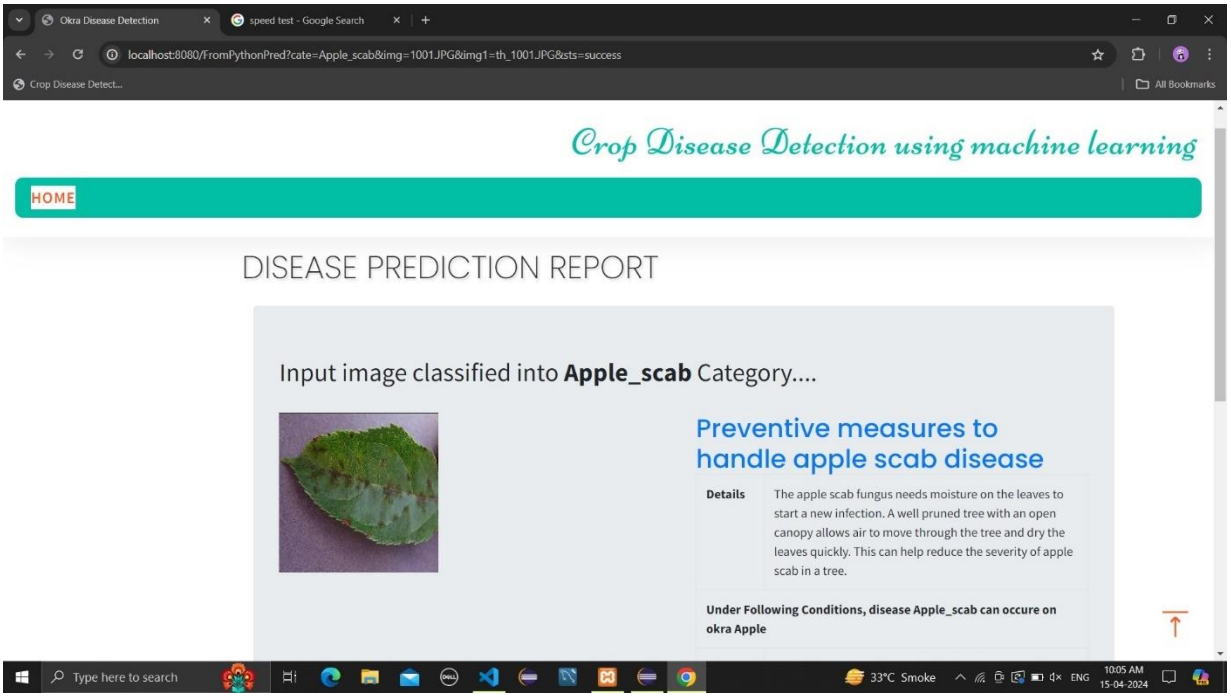


Fig. Prediction

In this tab the disease is predicted and the preventive measures are suggested to the farmer this is the main result tab of the whole system and with the preventive measures soil type and weather conditions are also suggested to the farmer.

6. CONCLUSION AND FUTURE SCOPE

6.1 Conclusion

The field of crop disease detection holds immense potential for revolutionizing agriculture. Through the integration of advanced technologies such as machine learning, remote sensing, and image processing, significant progress has been made in early and accurate identification of crop diseases. This has profound implications for farmers, researchers, and global food security. By enabling early detection, these technologies empower farmers to take proactive measures to manage crop diseases, thereby minimizing yield losses and reducing reliance on chemical inputs. Additionally, the data-driven insights generated by disease detection systems facilitate more informed decision-making in crop management practices, leading to improved productivity and sustainability. Furthermore, the development of precise and automated disease detection methods contributes to the advancement of precision agriculture, allowing for targeted interventions and resource optimization. This not only enhances economic efficiency but also mitigates environmental impact by reducing the indiscriminate use of pesticides and fertilizers.

6.2 Future Scope

- In this project we proposed crop disease prediction and remedies recommender system, in future we can add expert communication module and AI based question answers chatbots for farmers.
- Along with this we can add the module for possible crop diseases recommendation with the help of environmental changes
- Also, we can also use IOT device to predict the diseases.

REFERANCES

1. Kadir, A., "A Model of Plant Identification System Using GLCM, Lacunarity and Shen Features," Research Journal of Pharmaceutical, Biological, and Chemical Sciences Vol.5(2) 2014.
2. Naik, M.R., Sivappagari, C., "Plant Leaf and Disease Detection by Using HSV Features and SVM," IJESC, Volume 6 Issue No.12, 2016.
3. Greg Olmschenk, Hao Tang, Zhigang Zhu, "Crowd Counting with Minimal Data Using Generative Adversarial Networks for Multiple Target Regression, Applications of Computer Vision (WACV), 2018 IEEE Winter Conference on Computer Vision (WACV), 2018.
4. Kadir, A., "A Model of Plant Identification System Using GLCM, Lacunarity and Shen Features," Research Journal of Pharmaceutical, Biological, and Chemical Sciences Vol.5(2) 2014.
5. Abdel-Hamid, O.; Mohamed, A.R.; Jiang, H.; Deng, L.; Penn, G.; Yu, D. Convolutional neural networks for speech recognition. IEEE/ACM Trans. Audio Speech Lang. Process. 2014.
6. Le Cun, Y.; Jackel, L.D.; Boser, B.; Denker, J.S.; Graf, H.P.; Guyon, I.; Henderson, D.; Howard, R.E.; Hubbard, W. Handwritten digit recognition: Applications of neural network chips and automatic learning. IEEE Commun. Mag. 1989.
7. LeCun, Y.; Bottou, L.; Bengio, Y.; Haffner, P. Gradient-based learning applied to document recognition. Proc. IEEE 1998.
8. Fernández-Quintanilla, C.; Peña, J.; Andújar, D.; Dorado, J.; Ribeiro, A.; López-Granados, F. Is the current state of the art of weed monitoring suitable for site-specific weed management in arable crops? Weed Res. 2018.
9. Abade, A.; Ferreira, P.A.; de Barros Vidal, F. Plant diseases recognition on images using convolutional neural networks: A systematic review. Comput. Electron. Agric. 2021.
10. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going deeper with convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015
11. Nagaraju, M.; Chawla, P. Systematic review of deep learning techniques in plant disease detection. Int. J. Syst. Assur. Eng. Manag. 2020.

12. Kamilaris, A.; Prenafeta-Boldú, F.X. Deep learning in agriculture: A survey. Comput. Electron. Agric. 2018.
14. Lu, J.; Tan, L.; Jiang, H. Review on convolutional neural network (CNN) applied to plant leaf disease classification. Agriculture 2021.
15. Golhani, K.; Balasundram, S.K.; Vadamalai, G.; Pradhan, B. A review of neural networks in plant disease detection using hyperspectral data. Inf. Process. Agric. 2018.
16. Bangari, S.; Rachana, P.; Gupta, N.; Sudi, P.S.; Baniya, K.K. A Survey on Disease Detection of a potato Leaf Using CNN. In Proceedings of the 2nd IEEE International Conference on Artificial Intelligence and Smart Energy (ICAIS), Coimbatore, India, 23–25 February 2022.
17. Carvalho, F.P. Agriculture, pesticides, food security and food safety. Environ. Sci. Policy 2006, 9.
18. Miller, S.A.; Beed, F.D.; Harmon, C.L. Plant disease diagnostic capabilities and networks. Annu. Rev. Phytopathol. 2009.
19. Najafabadi, M.M.; Villanustre, F.; Khoshgoftaar, T.M.; Seliya, N.; Wald, R.; Muharemagic, E. Deep learning applications and challenges in big data analytics. J. Big Data 2015.
20. Gebbers, R.; Adamchuk, V.I. Precision agriculture and food security. Science 2010