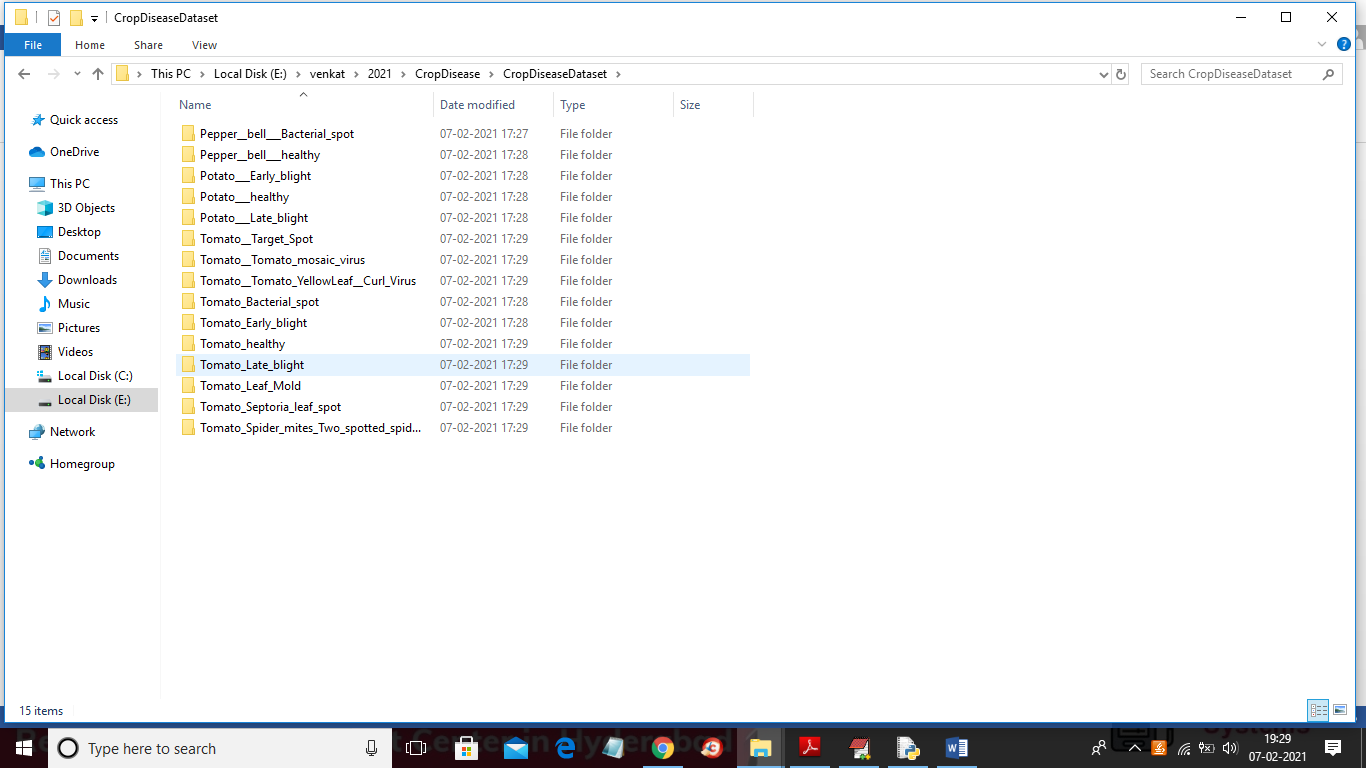
**Agriculture Crop Image Classification**

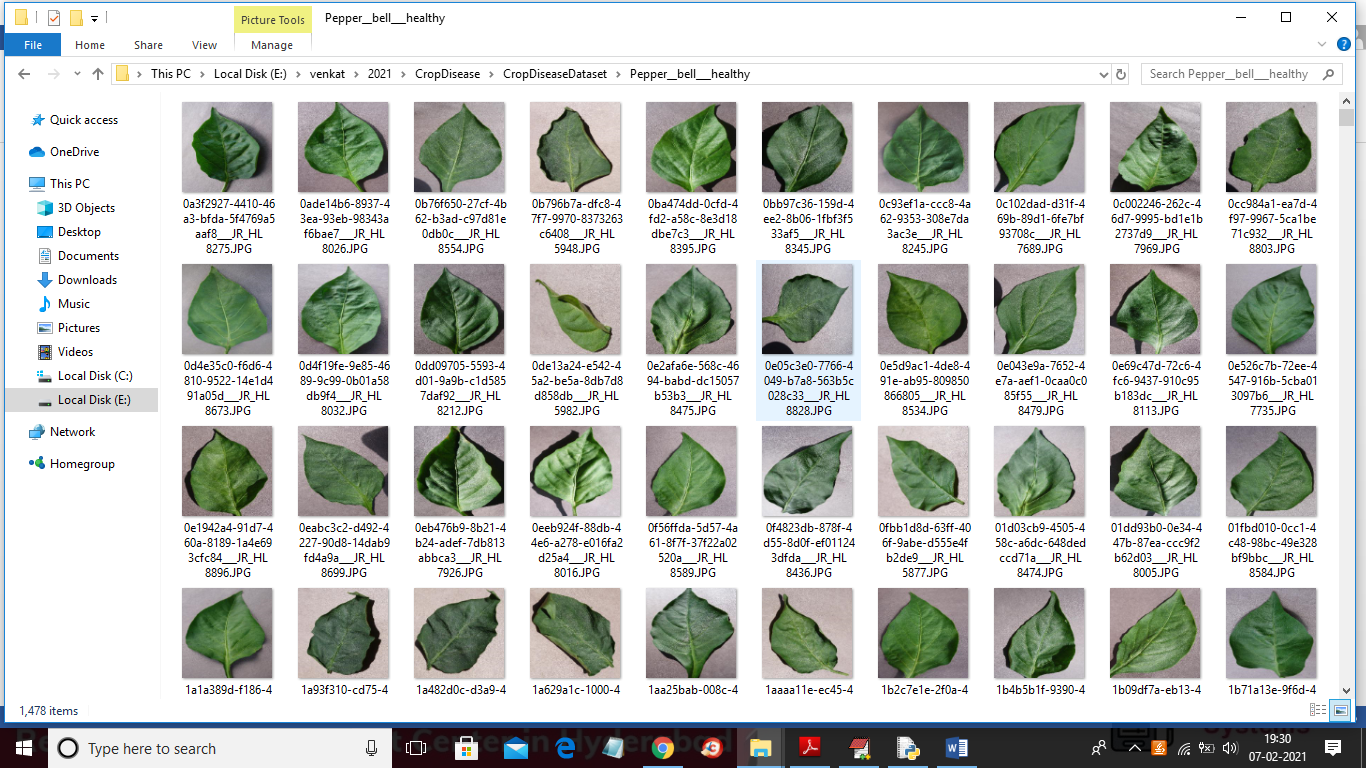
In this paper author is applying deep learning convolution neural network (CNN) to predict crop disease and its pests to reduce economical loss in crop business. To build disease recognition model author is applying RESNET CNN model which consists of 3 parts

1. Feature Extraction: CNN compose of multiple layers and first layer define for feature extraction and this features will be extracted from given input image dataset or any other multidimensional dataset.
2. Feature Selection: Using this layer features will be selected by applying a layer called pooling or max polling.
3. Activation module: using this module RELU will be applied on input features to remove out unimportant features and hold only relevant important features
4. Flatten: This layer will be define to convert multidimensional input features into single dimensional input array
5. Dense: This layer can be used to connect one layer to other layer to receive input features from previous layer to new layer to further filter input features in next layer to get most important features from dataset to have best prediction result.

To implement this project we have used crop disease recognition dataset and this dataset saved inside ‘CropDiseaseDataset’ folder and below screen shots showing various type of crop disease images

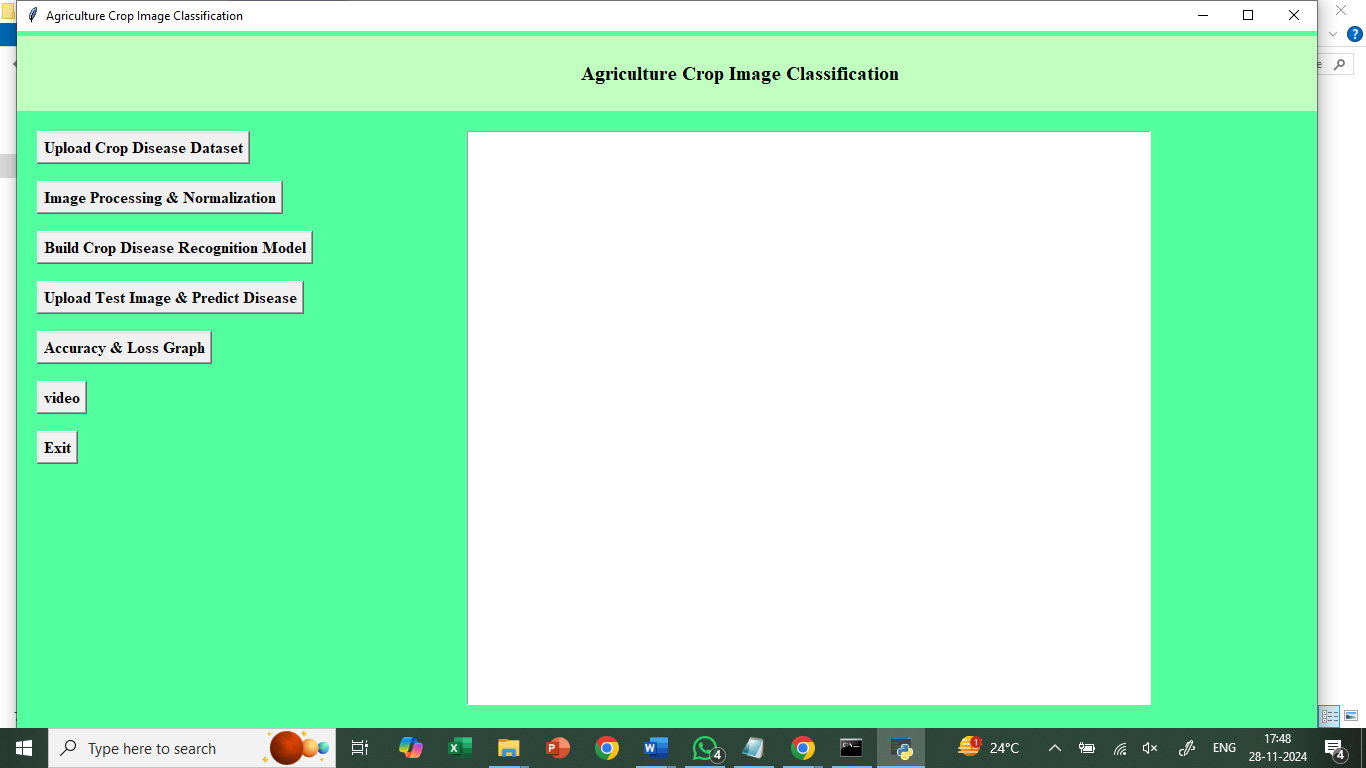


In above screen we are using 15 different type of crop images and each folder contains images of own leaf and in below screen you can see those image. You too just go inside any above folder to see images

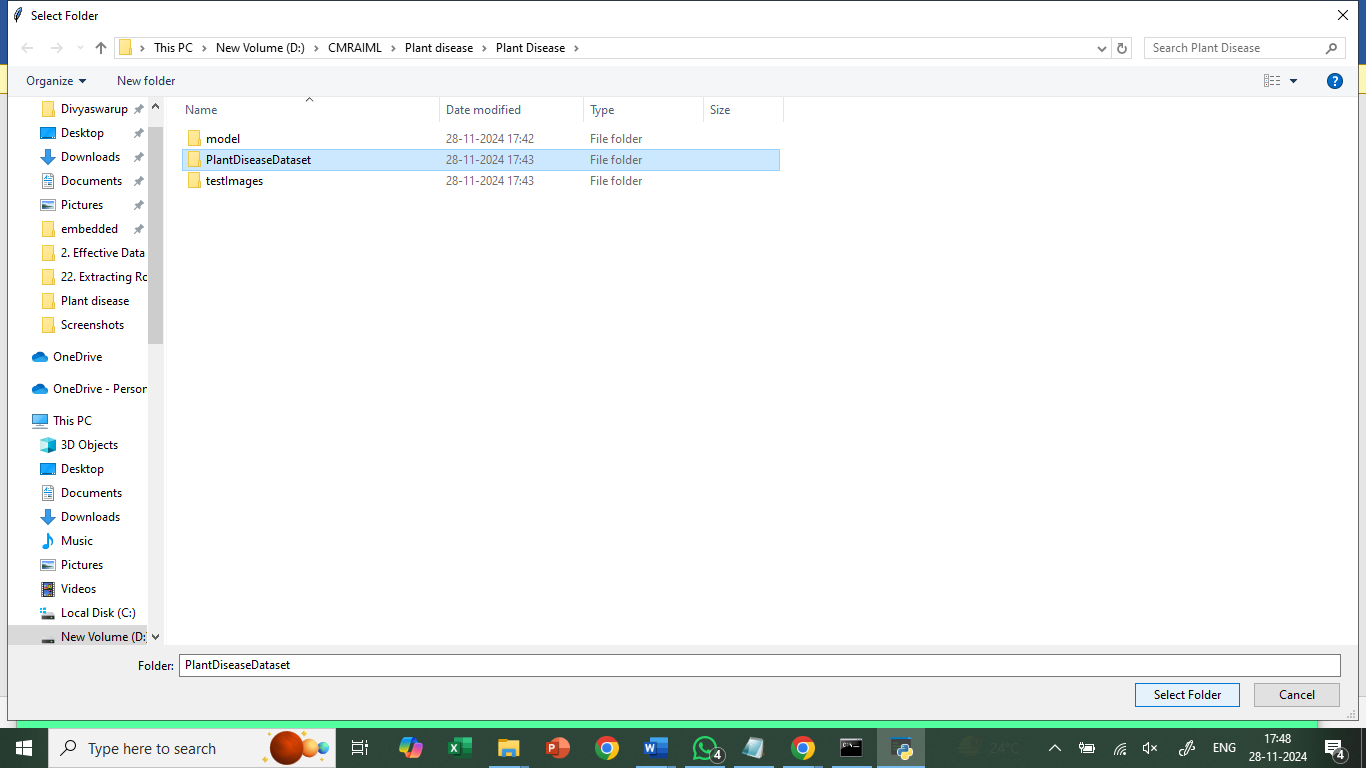


SCREEN SHOTS

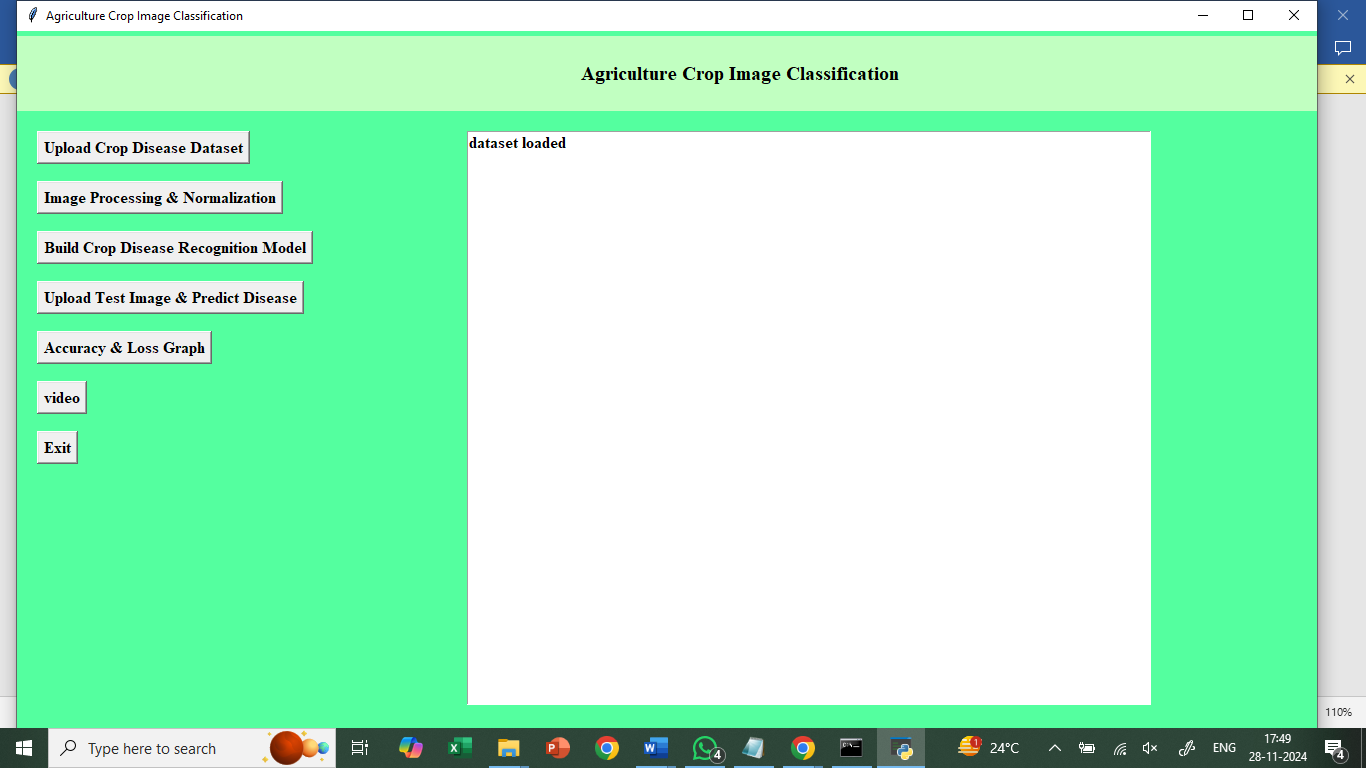
To run project double click on ‘run.bat’ file to get below screen



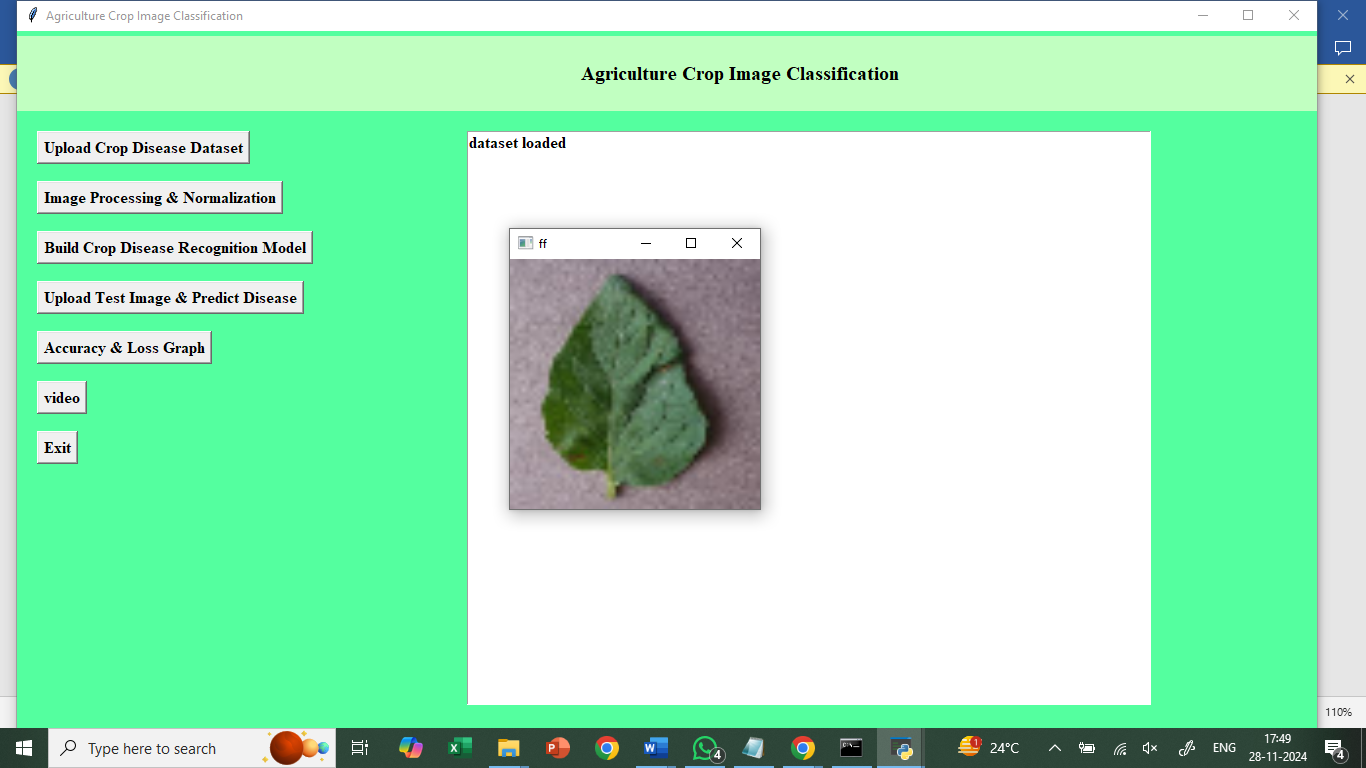
In above screen click on ‘Upload Crop Disease Dataset’ button to upload dataset images



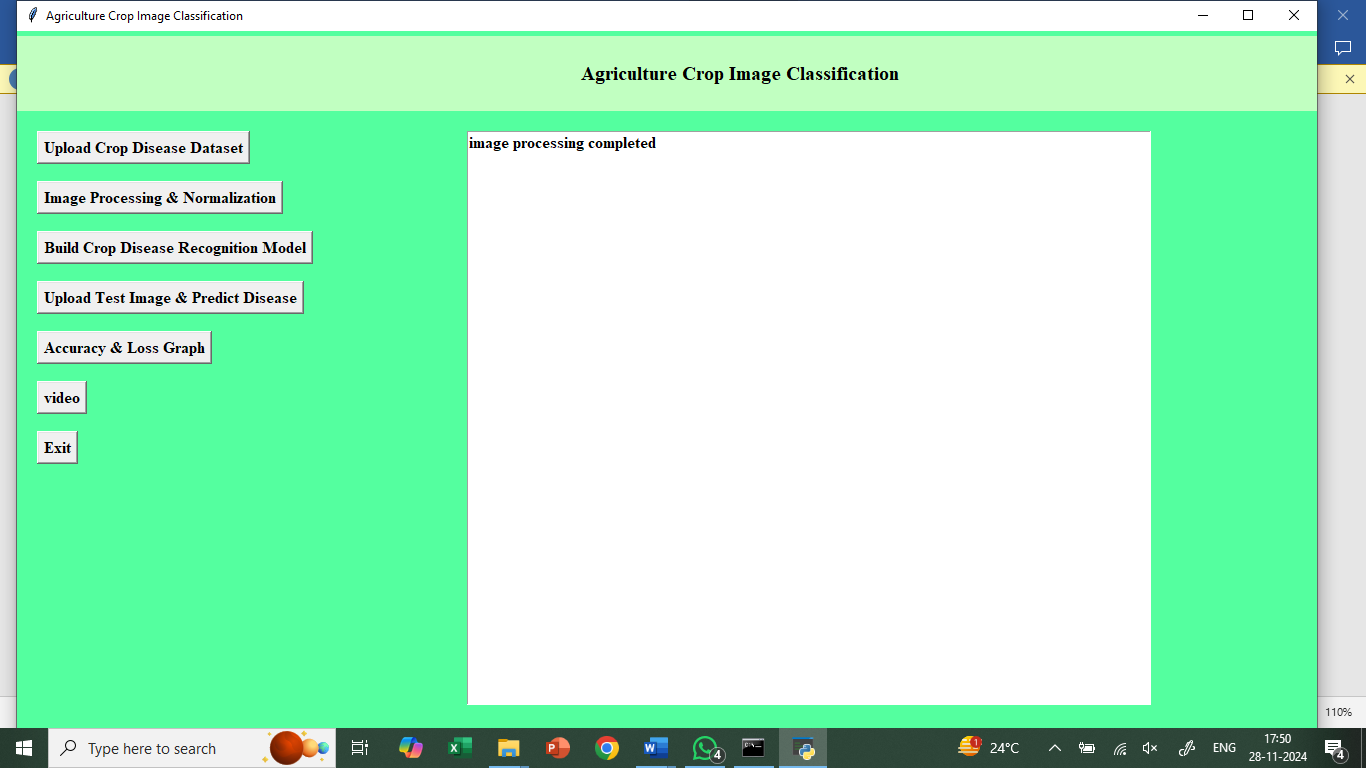
In above screen selecting and uploading ‘CropDiseaseDataset’ folder and then click on ‘SelectFolder’ button to load dataset and to get below screen



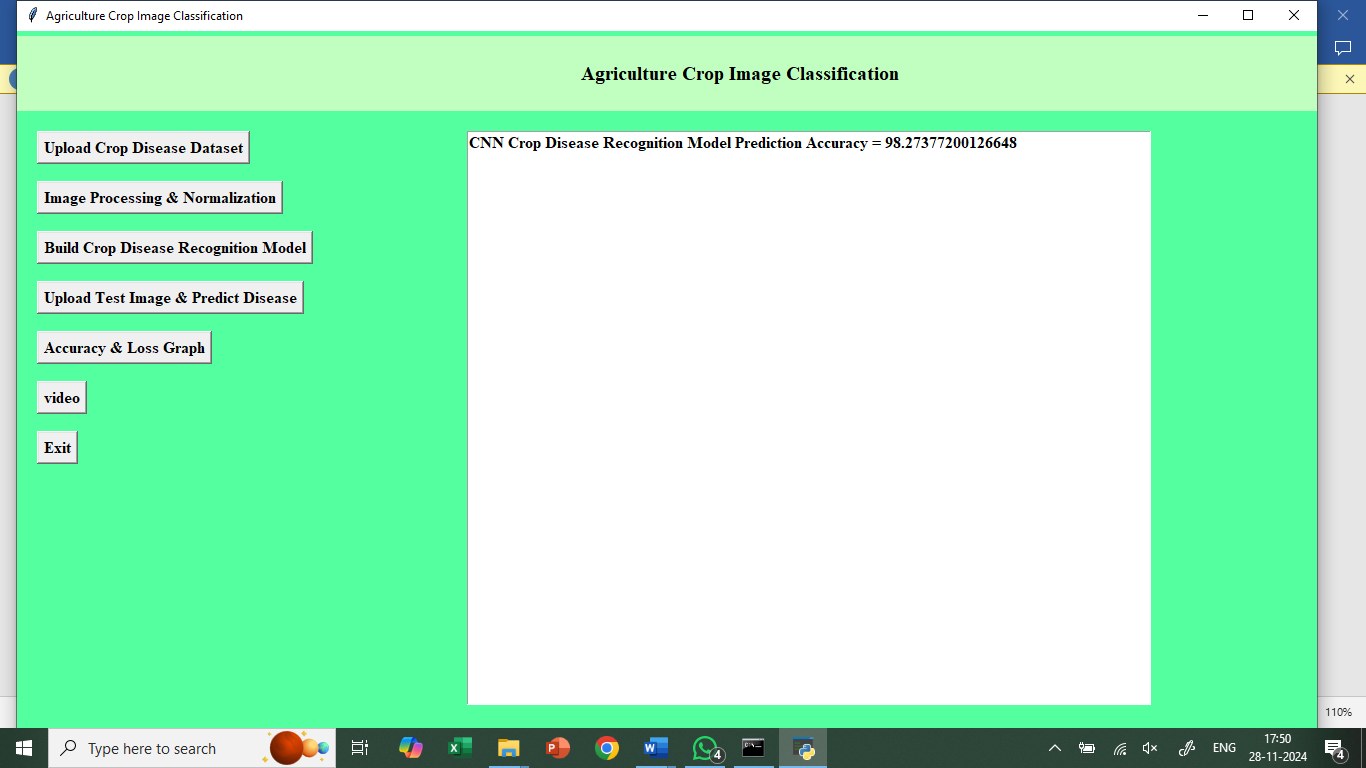
In above screen dataset loaded and now click on ‘Image Processing & Normalization’ button to read all images and then process images to normalize by converting each image pixel value between 0 and 1 and for that normalization we will divide image pixels with 255 and then get value as 0 or 1 as all images pixel value will be between 0 to 255.



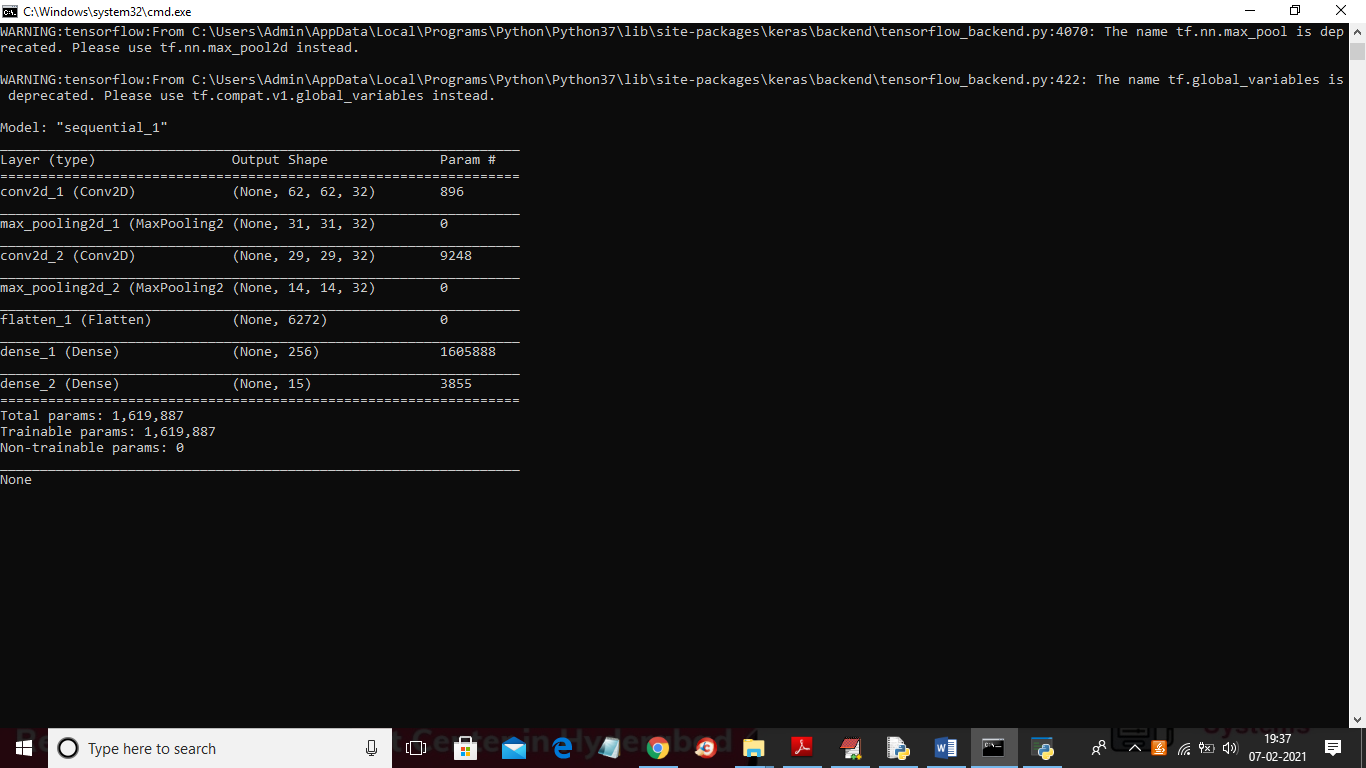
In above screen after applying normalization we are just displaying one random image from dataset to check whether images loaded and process properly or not and now you close above image to get below screen



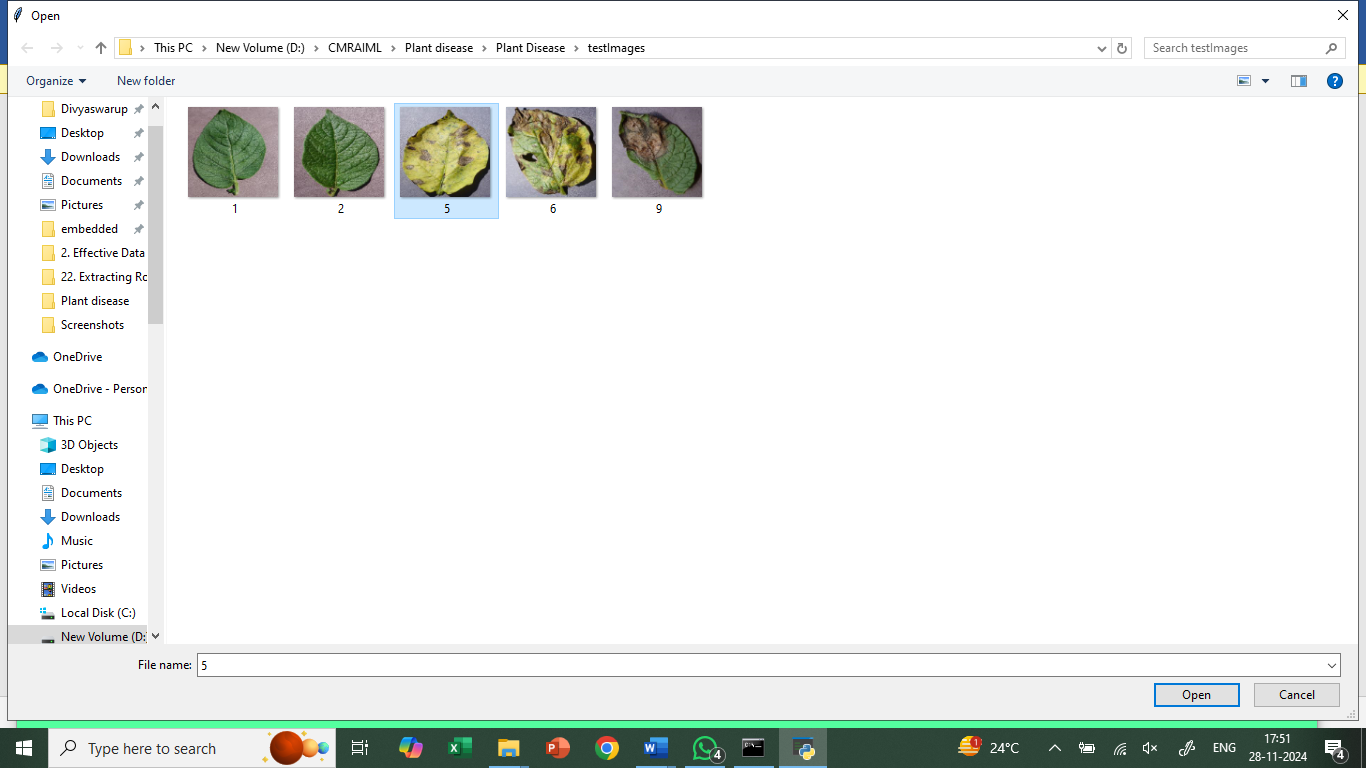
In above screen all images process successfully and now dataset images are ready and now click on ‘Build Crop Disease Recognition Model’ button to build CNN model

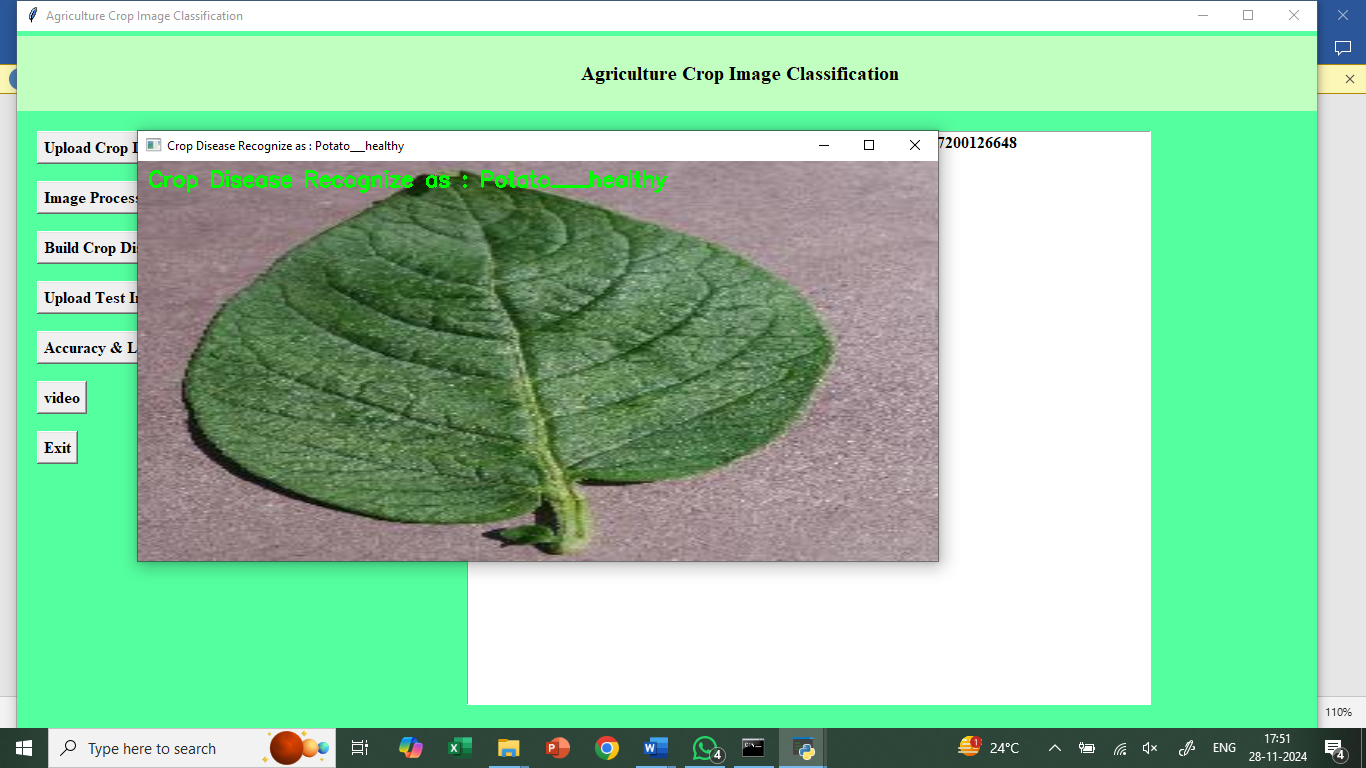


In above screen CNN model generated and its prediction accuracy is 98% and in below console screen we can see all CNN layers details

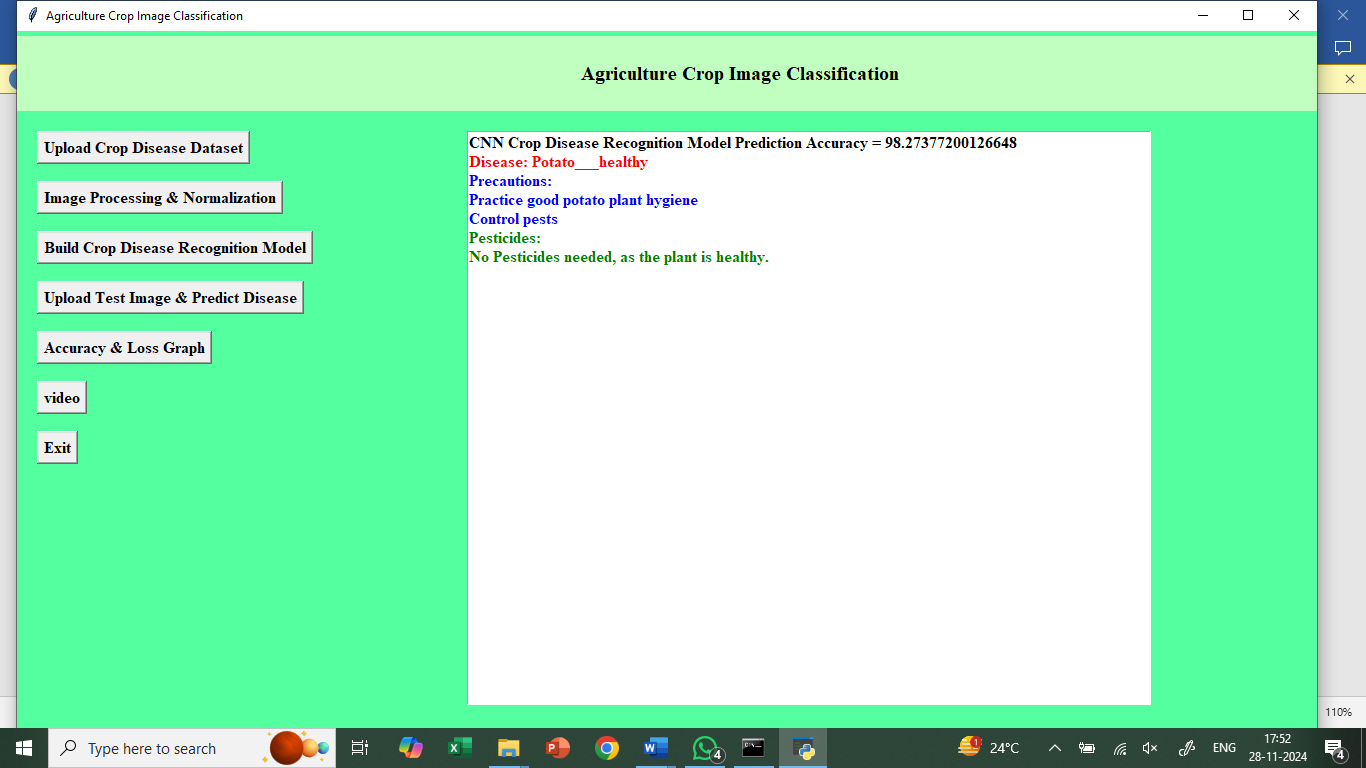


In above screen we can see we have used CONV2D, MAXPOOLING, FLATTEN and DENSE layer to build crop disease recognition model and RELU details you can see in code. Now model is ready and now click on ‘Upload Test Image & Predict Disease’ button to upload any test image and then application will predict disease or healthy from that image

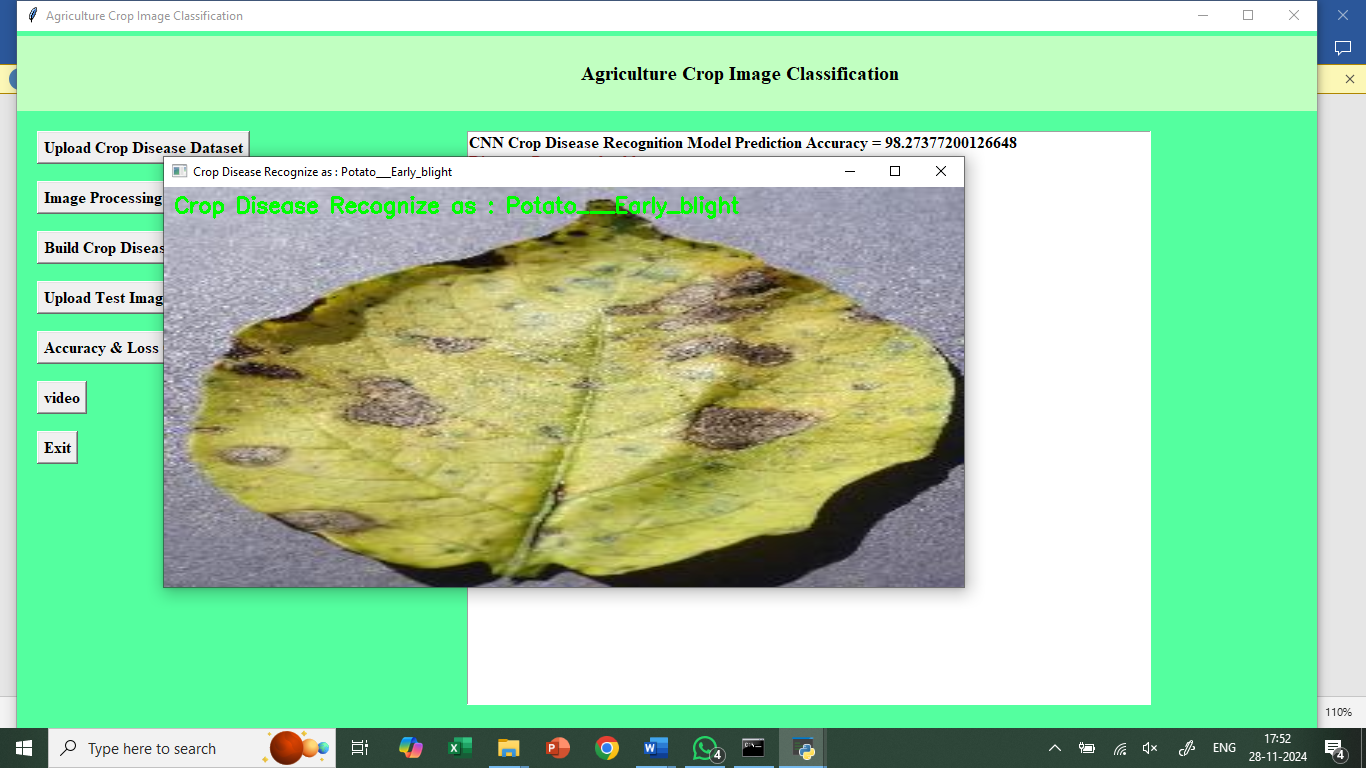
In above screen selecting and uploading ‘1.JPG’ image file and then click on ‘Open’ button to get below prediction result

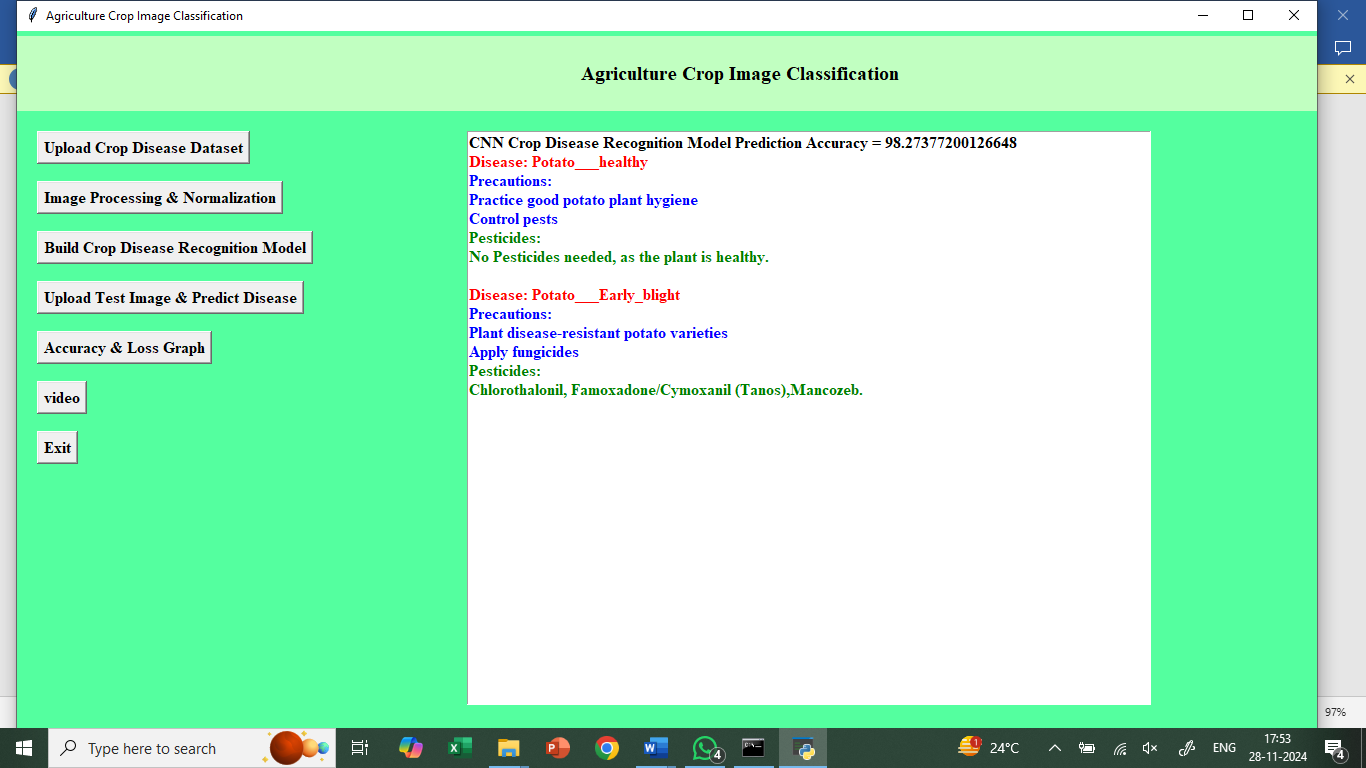


In above screen potato leaf predicted as healthy and now try with other image

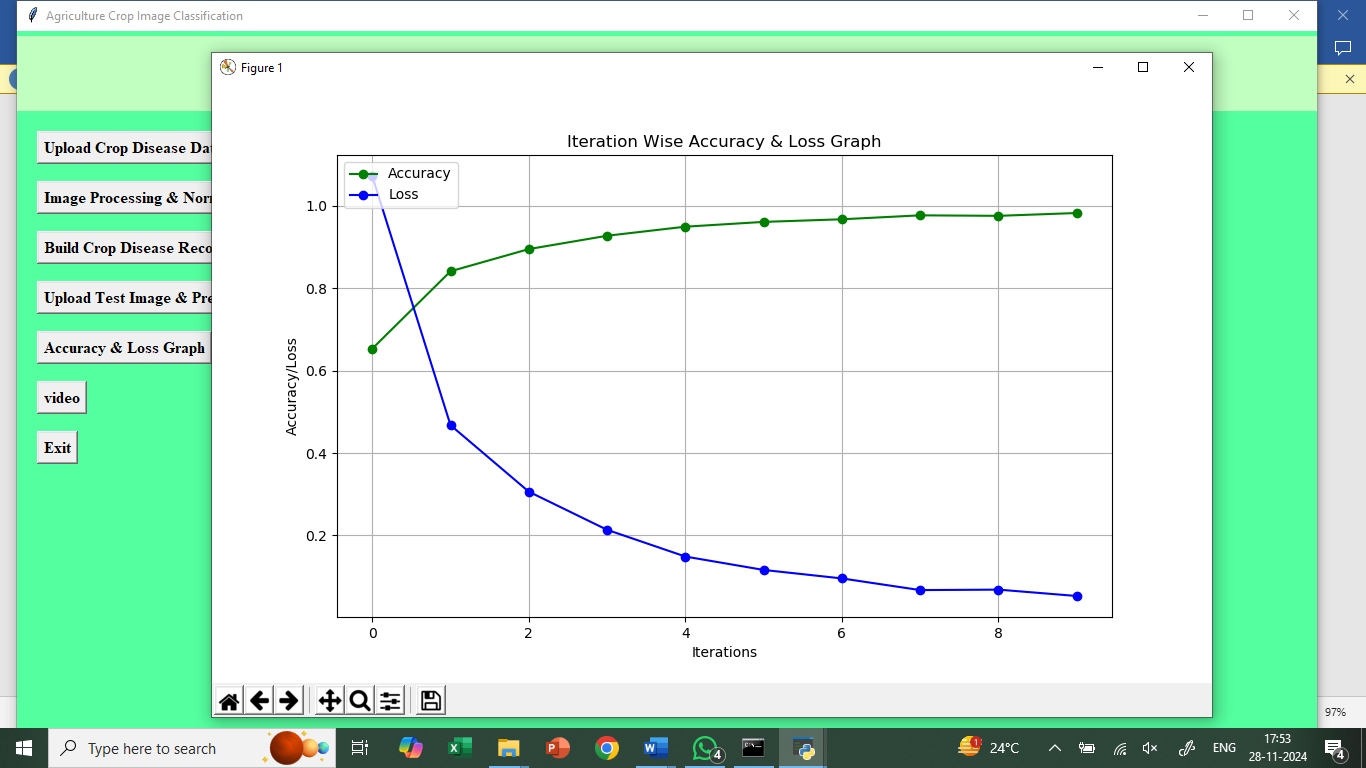


In above screen selecting and uploading ‘5.JPG’ file and click ‘Open’ button to get below result





In above image potato EARLY BLIGHT disease is detected or recognize and similarly you can upload any other image and get result and now click on ‘Accuracy & Loss Graph’ button to get below graph



In above graph x-axis represents epoch/iterations and y-axis represents accuracy/loss and green line represents accuracy and blue line represents loss and from above graph we can see with each increasing iteration accuracy is getting better and better and loss getting decrease