CONTENTS

1.INTRODUCTION

Overview:

Agriculture is the most important sector in today’s life. Most plants are affected by a wide variety of bacterial and fungal diseases. Diseases on plants placed a major constraint on the production and a major threat to food security. Hence, early and accurate identification of plant diseases is essential to ensure high quantity and best quality. In recent years, the number of diseases on plants and the degree of harm caused has increased due to the variation in pathogen varieties, changes in cultivation methods, and inadequate plant protection techniques.

An automated system is introduced to identify different diseases on plants by checking the symptoms shown on the leaves of the plant. Deep learning techniques are used to identify the diseases and suggest the precautions that can be taken for those diseases.

Purpose:

The main purpose of the project is to enhance the utilization of the technology by using the advancement in the Deep Learning domain to enable the formers to identify the disease that is affecting their farm from the comfort of their home with the help of a single click through the website. In this, the farmer or the user has to submit the image of disease affected leaf on our website, and our website in turn produces the result about the disease and the necessary actions to be taken. With the help of this project, we achieve an interactive system that helps the farmers to identify the disease affecting the crop at early stages and reduce crop damage.

2.LITERATURE SURVEY

Existing Problem

Previously people have to visit help centers or depend on fertilizer retailers in order to identify and understand the appropriate solution to the disease affecting the crop. People from rural areas have to travel long distances to visit help centers, in some cases, they have to depend on local fertilizer retailers which in turn either increase their expenses or result in crop damage.

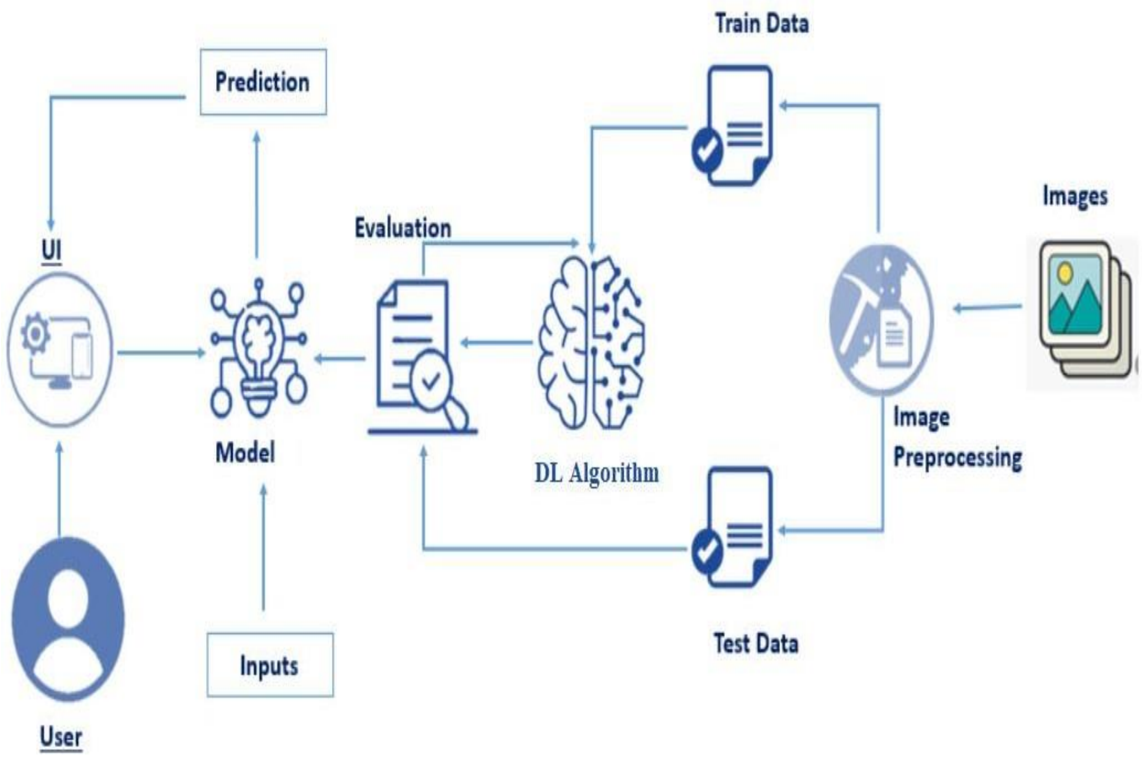
And there are even a few solutions using machine learning techniques like Naive Bayes, Random Forest Classifiers, etc. But they fail to produce good accuracy because they are not appropriate for image processing. In some cases, they ask for chemical values of the crop which requires heavy testing on the leaf resulting in a burden to the farmers again.

Proposed Solution

In this proposed project, we will use an advanced Deep Learning model that is a Convolution neural network to perform processing on the image of the disease-affected leaf submitted by a user. The usage of Convolution neural networks brought a large leap in accuracy in predicting the leaf disease of a crop. With this, we will help the former to monitor the crop growth and identify diseases of the crop as earlier as possible. In this, we also implemented the web page to help the former to understand the process with the help of the visual aids of the website.The project provides disease detections for a wide group of vegetable and fruit crops which helps the vast group of farmers.

3. THEORITICAL ANALYSIS

Block diagram



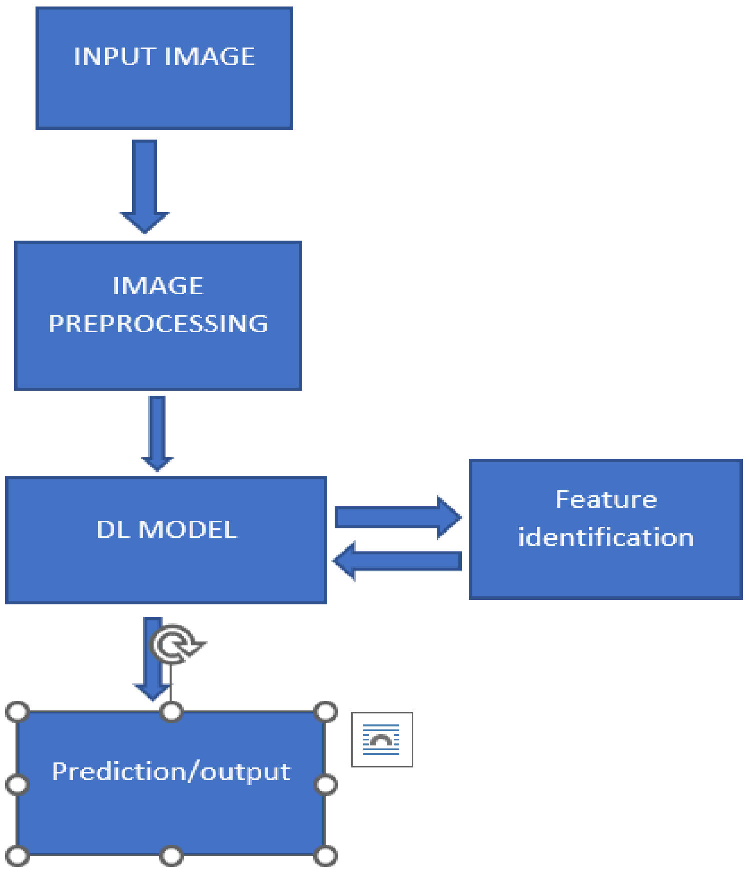
Hardware / Software designing

- Hardware Requirements
    - Processor-i5
    - RAM-8GB
- Software requirements
    - Anaconda3
    - Jupyter Notebook
    - Google calloboratory
    - Spyder

4.EXPERIMENTAL INVESTIGATION

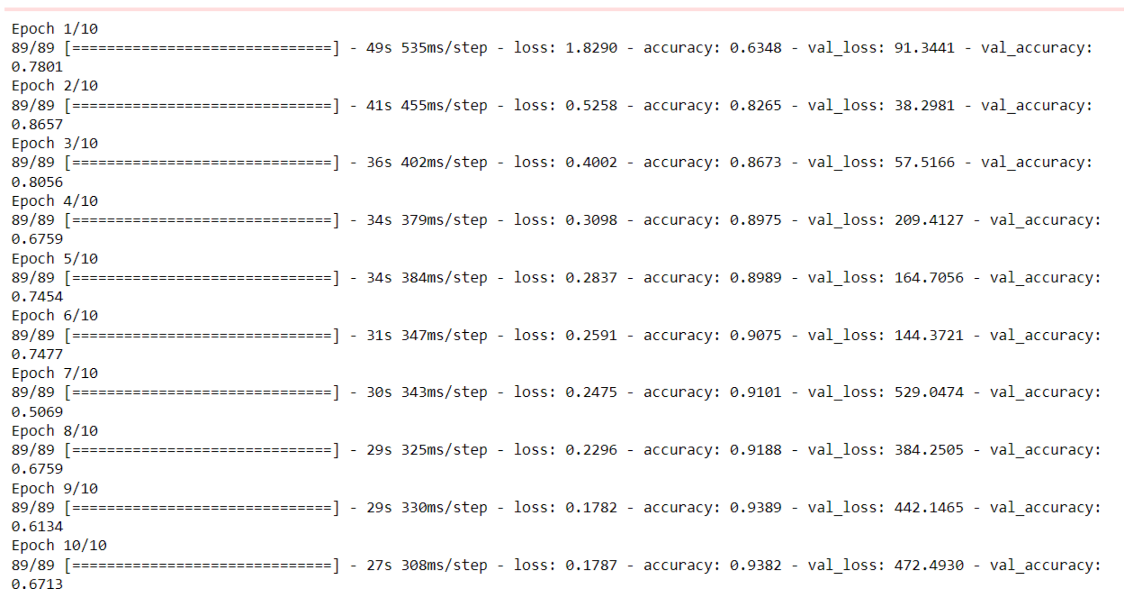
For the implementation of the project, we have gone through several research papers from the "researchgate" website. We have also gone through several Youtube channels including the Krish Naik channel. We have gone through several websites including towardsdatascience.com, tutorialspoint, geeks for geeks, etc.

5.FLOW CHART



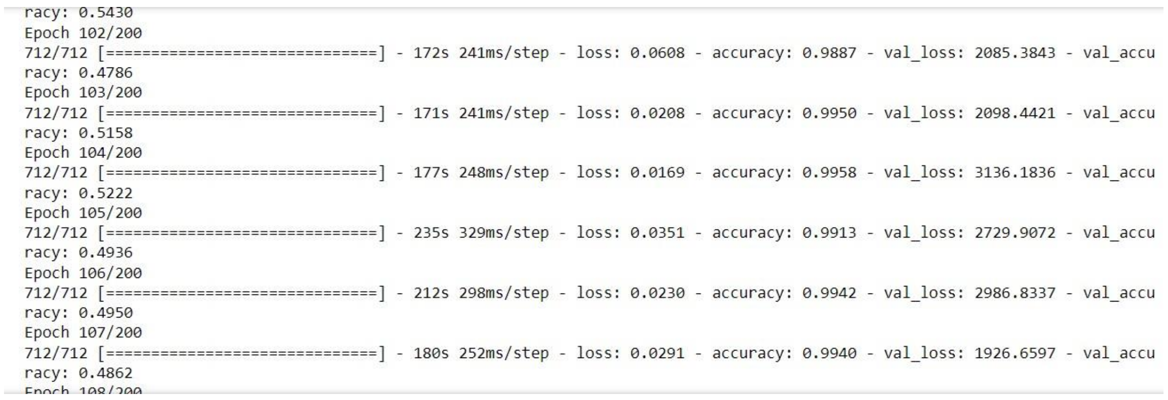
6.RESULTS

1.Vegetable training:



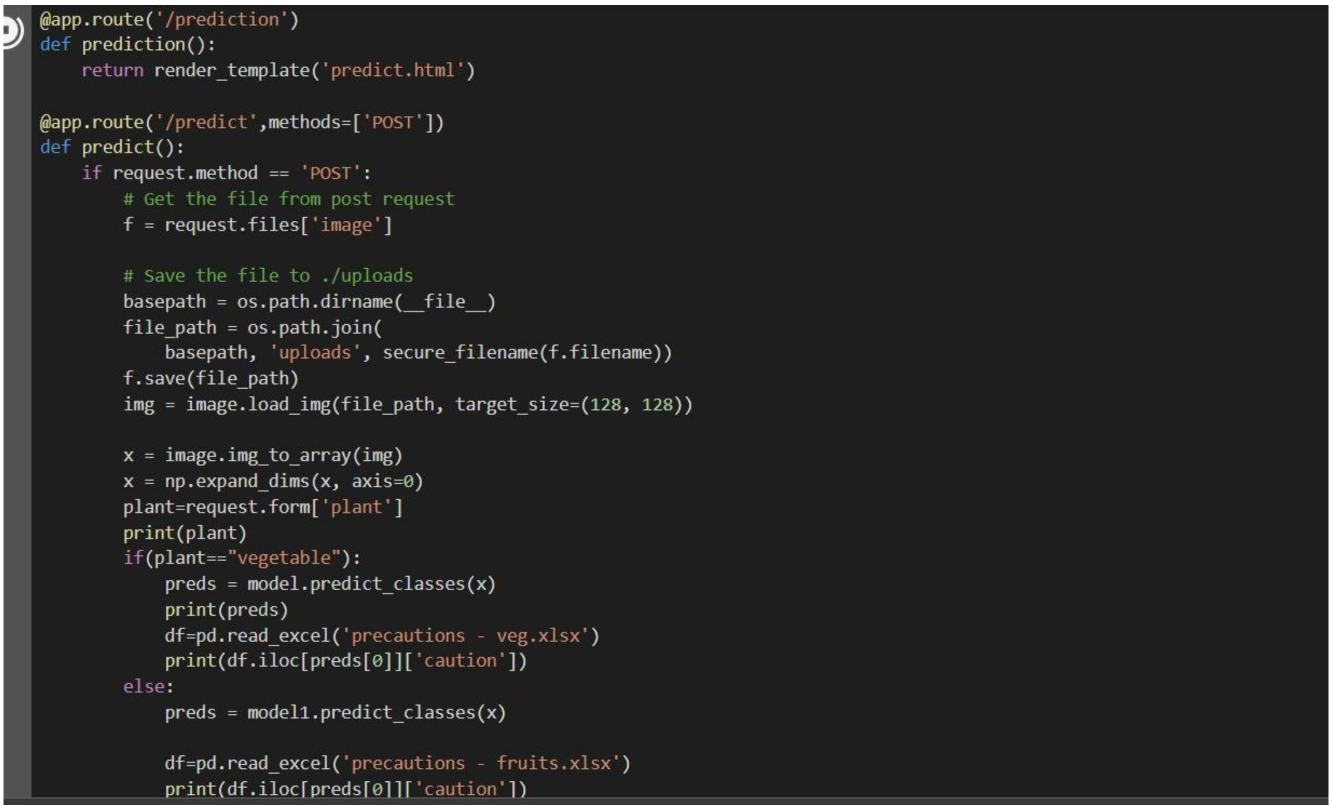
```
Epoch 1/10
89/89 [==============================] - 49s 535ms/step - loss: 1.8290 - accuracy: 0.6348 - val_loss: 91.3441 - val_accuracy:
0.7801
Epoch 2/10
89/89 [==============================] - 41s 455ms/step - loss: 0.5258 - accuracy: 0.8265 - val_loss: 38.2981 - val_accuracy:
0.8657
Epoch 3/10
89/89 [==============================] - 36s 402ms/step - loss: 0.4002 - accuracy: 0.8673 - val_loss: 57.5166 - val_accuracy:
0.8056
Epoch 4/10
89/89 [==============================] - 34s 379ms/step - loss: 0.3098 - accuracy: 0.8975 - val_loss: 209.4127 - val_accuracy:
0.6759
Epoch 5/10
89/89 [==============================] - 34s 384ms/step - loss: 0.2837 - accuracy: 0.8989 - val_loss: 164.7056 - val_accuracy:
0.7454
Epoch 6/10
89/89 [==============================] - 31s 347ms/step - loss: 0.2591 - accuracy: 0.9075 - val_loss: 144.3721 - val_accuracy:
0.7477
Epoch 7/10
89/89 [==============================] - 30s 343ms/step - loss: 0.2475 - accuracy: 0.9101 - val_loss: 529.0474 - val_accuracy:
0.5069
Epoch 8/10
89/89 [==============================] - 29s 325ms/step - loss: 0.2296 - accuracy: 0.9188 - val_loss: 384.2505 - val_accuracy:
0.6759
Epoch 9/10
89/89 [==============================] - 29s 330ms/step - loss: 0.1782 - accuracy: 0.9389 - val_loss: 442.1465 - val_accuracy:
0.6134
Epoch 10/10
89/89 [==============================] - 27s 308ms/step - loss: 0.1787 - accuracy: 0.9382 - val_loss: 472.4930 - val_accuracy:
0.6713
```

2.Fruit training:



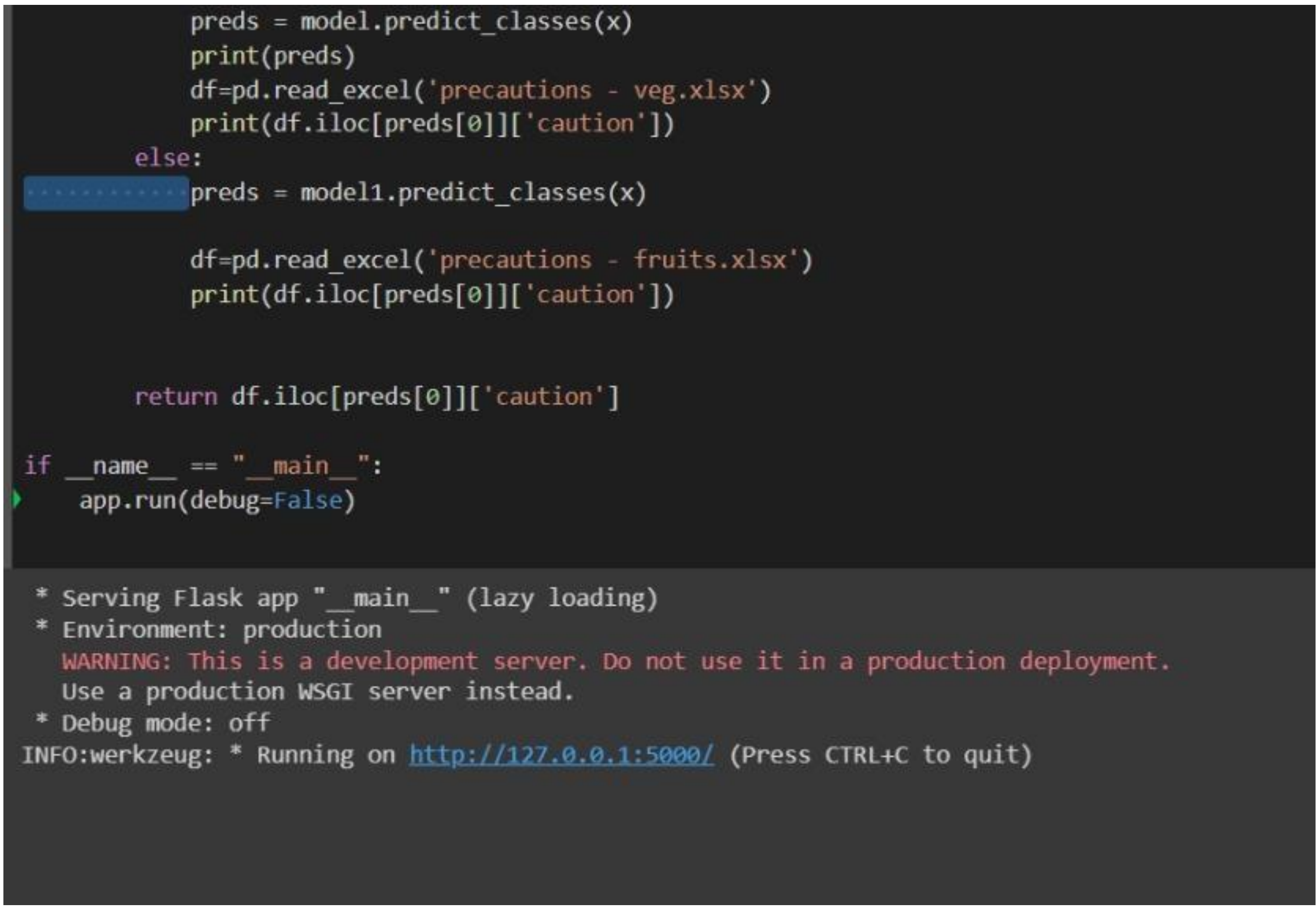
```
racy: 0.5430
Epoch 102/200
712/712 [==============================] - 172s 241ms/step - loss: 0.0608 - accuracy: 0.9887 - val_loss: 2085.3843 - val_accu
racy: 0.4786
Epoch 103/200
712/712 [==============================] - 171s 241ms/step - loss: 0.0208 - accuracy: 0.9950 - val_loss: 2098.4421 - val_accu
racy: 0.5158
Epoch 104/200
712/712 [==============================] - 177s 248ms/step - loss: 0.0169 - accuracy: 0.9958 - val_loss: 3136.1836 - val_accu
racy: 0.5222
Epoch 105/200
712/712 [==============================] - 235s 329ms/step - loss: 0.0351 - accuracy: 0.9913 - val_loss: 2729.9072 - val_accu
racy: 0.4936
Epoch 106/200
712/712 [==============================] - 212s 298ms/step - loss: 0.0230 - accuracy: 0.9942 - val_loss: 2986.8337 - val_accu
racy: 0.4950
Epoch 107/200
712/712 [==============================] - 180s 252ms/step - loss: 0.0291 - accuracy: 0.9940 - val_loss: 1926.6597 - val_accu
racy: 0.4862
Epoch 108/200
```

3.Flask app



```
@app.route('/prediction')
def prediction():
    return render_template('predict.html')

@app.route('/predict',methods=['POST'])
def predict():
    if request.method == 'POST':
        # Get the file from post request
        f = request.files['image']

        # Save the file to ./uploads
        basepath = os.path.dirname(__file__)
        file_path = os.path.join(
            basepath, 'uploads', secure_filename(f.filename))
        f.save(file_path)
        img = image.load_img(file_path, target_size=(128, 128))

        x = image.img_to_array(img)
        x = np.expand_dims(x, axis=0)
        plant=request.form['plant']
        print(plant)
        if(plant=="vegetable"):
            preds = model.predict_classes(x)
            print(preds)
            df=pd.read_excel('precautions - veg.xlsx')
            print(df.iloc[preds[0]]['caution'])
        else:
            preds = model1.predict_classes(x)

            df=pd.read_excel('precautions - fruits.xlsx')
            print(df.iloc[preds[0]]['caution'])
```

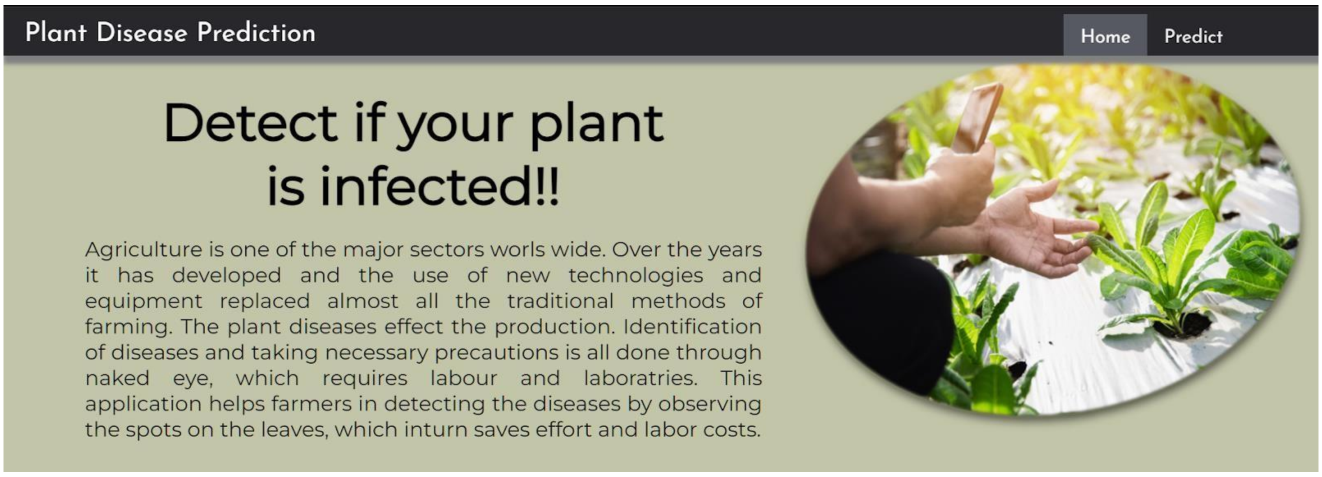
```
@app.route('/prediction')
def prediction():
    return render_template('predict.html')

@app.route('/predict',methods=['POST'])
def predict():
    if request.method == 'POST':
        # Get the file from post request
        f = request.files['image']

        # Save the file to ./uploads
        basepath = os.path.dirname(__file__)
        file_path = os.path.join(
            basepath, 'uploads', secure_filename(f.filename))
        f.save(file_path)
        img = image.load_img(file_path, target_size=(128, 128))

        x = image.img_to_array(img)
        x = np.expand_dims(x, axis=0)
        plant=request.form['plant']
        print(plant)
        if(plant=="vegetable"):
            preds = model.predict_classes(x)
            print(preds)
            df=pd.read_excel('precautions - veg.xlsx')
            print(df.iloc[preds[0]]['caution'])
        else:
            preds = model1.predict_classes(x)

            df=pd.read_excel('precautions - fruits.xlsx')
            print(df.iloc[preds[0]]['caution'])
```



```
            preds = model.predict_classes(x)
            print(preds)
            df=pd.read_excel('precautions - veg.xlsx')
            print(df.iloc[preds[0]]['caution'])
        else:
            preds = model1.predict_classes(x)

            df=pd.read_excel('precautions - fruits.xlsx')
            print(df.iloc[preds[0]]['caution'])


        return df.iloc[preds[0]]['caution']

if __name__ == "__main__":
    app.run(debug=False)
```

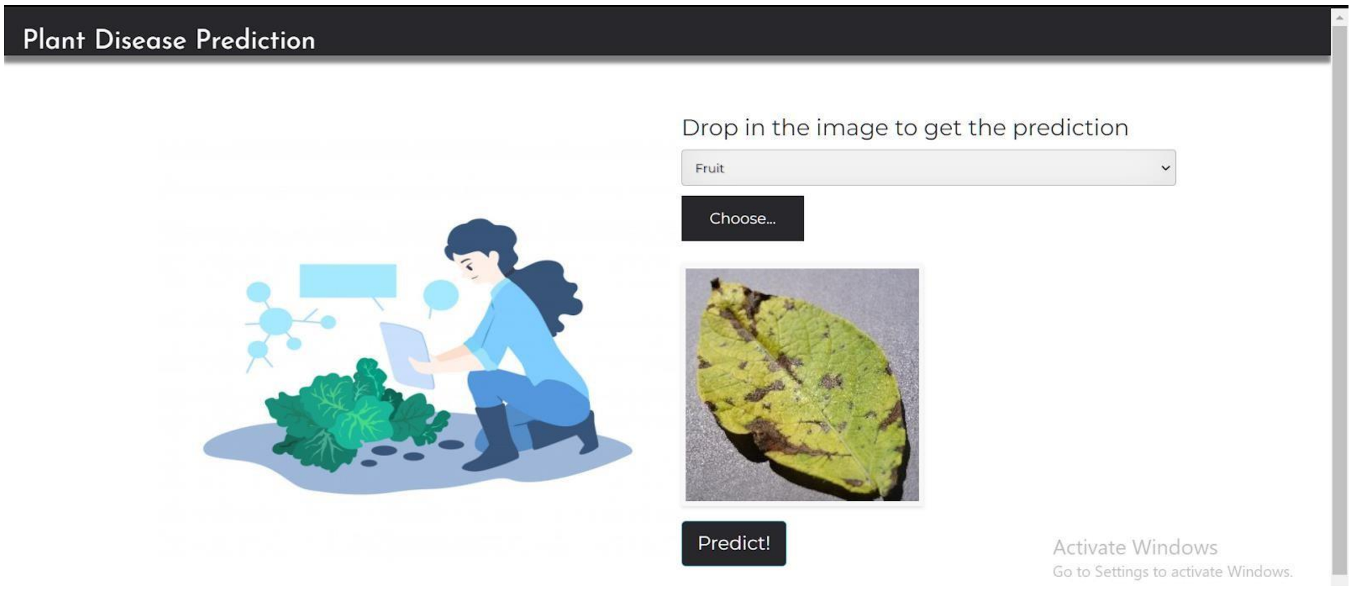
```
 * Serving Flask app "__main__" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: off
INFO:werkzeug: * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

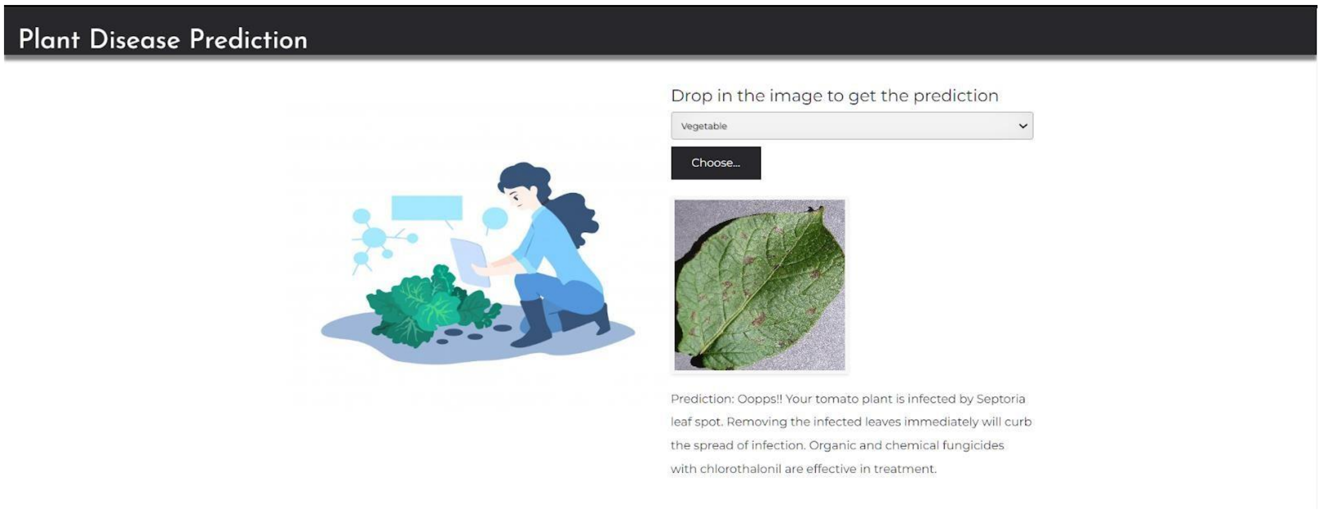
3.Website:



Detect if your plant is infected!!

Agriculture is one of the major sectors worls wide. Over the years it has developed and the use of new technologies and equipment replaced almost all the traditional methods of farming. The plant diseases effect the production. Identification of diseases and taking necessary precautions is all done through naked eye, which requires labour and laboratries. This application helps farmers in detecting the diseases by observing the spots on the leaves, which inturn saves effort and labor costs.


Plant Disease Prediction
Drop in the image to get the prediction
Fruit
Choose...
Predict!
Activate Windows
Go to Settings to activate Windows.



Prediction: Oopps!! Your tomato plant is infected by Septoria leaf spot. Removing the infected leaves immediately will curb the spread of infection. Organic and chemical fungicides with chlorothalonil are effective in treatment.

7.ADVANTAGES AND DISADVANTAGES

Advantages:

1. The first advantage of the model is accuracy. The accuracy of the model is around 80-90% which is a pretty decent accuracy.

2. With the advancement of the technology the task was made simple for the farmers. This enables the farmers to resolve their crop issues at home.

3. The third advantage is that with a simple click on camera is just enough to get the disease of crop.

Disadvantages:

1. The accuracy of the model is good. But really not so appropriate to the current situation. Since according to current situation, we need the accuracy around 97%. So that we can get good result.

2. For training model we need high quality images. Which is quite expensive and time taking.

8.APPLICATIONS

1. This can be used to help the farmers to identify the crop disease with a single photo click which helps the user to identify the disease. And provide appropriate solution.

2. This also brings the modernization of the farming domain.

9.CONCLUSION

In this we have used the Convolution Neural Network for both the vegetables and fruits leaf disease detection which in turn yielded a good accuracy around 93% and 92% while execution. This algorithm performed morebetter than all previous ML techniques including Decision Tress, K-Nearest Neighbours etc. This project also includes the interactive interphase which in turn adds necessary put ups to it. With this the ML domain has been doing a fabulous job in every sector letting us to make the society a better place to live.

10.FUTURE SCOPE

In this project we have used the convolution Neural Network which yielded a good accuracy around 92% but there are far more advanced techniques including Resnet networks, VGG networks and DenseNet networks which can work much better when compared to traditional CNN

11. BIBILOGRAPHY

1. P.Pandi Selvi,P. Poornima, "Soil Based Fertilizer Recommendation System for Crop Disease Prediction System", International Journal of Engineering Trends and Applications(IJETA)–Volume 8 Issue 2,Mar-Apr 2021

2. R Indumathi.N .Saagari.; V Thejuswini.;R Swarnareka.,"Leaf Disease Detection and Fertilizer Suggestion", IEEE International Conference on System, Computation, Automation and Networking (ICSCAN), 29-30 March 2019, DOI:10.1109/ICSCAN.2019.8878781

APPENDIX

SOURCE CODE OF FLASK:

```
import requests
from tensorflow.keras.preprocessing import image
from tensorflow.keras.models import load_model
import numpy as np
import pandas as pd
import tensorflow as tf
from flask import Flask, request, render_template, redirect, url_for
import os
from werkzeug.utils import secure_filename
from tensorflow.python.keras.backend import set_session


app = Flask(_name_)
global sess
#sess = tf.Session()
from tensorflow.keras.preprocessing import image

global graph
graph=tf.compat.v1.get_default_graph()
#set_session(sess)

#load both the vegetable and fruit models
model = load_model("vegetable.h5")
model1=load_model("fruit.h5")

#home page
@app.route('/')
def home():
    return render_template('home.html')

#prediction page
@app.route('/prediction')
```

```
def prediction():
    return render_template('predict.html')

@app.route('/predict',methods=['POST'])
def predict():
    if request.method == 'POST':
        # Get the file from post request
        f = request.files['image']

        # Save the file to ./uploads
        basepath = os.path.dirname(_file_)
        file_path = os.path.join(
            basepath, 'uploads', secure_filename(f.filename))
save(file_path)
        img = image.load_img(file_path, target_size=(128, 128))

        x = image.img_to_array(img)
        x = np.expand_dims(x, axis=0)
        plant=request.form['plant']
        print(plant)
        if(plant=="vegetable"):
            preds = model.predict_classes(x)
            print(preds)
            df=pd.read_excel('precautions - veg.xlsx')
            print(df.iloc[preds[0]]['caution'])
        else:
            preds = model1.predict_classes(x)

            df=pd.read_excel('precautions - fruits.xlsx')
            print(df.iloc[preds[0]]['caution'])


        return df.iloc[preds[0]]['caution']

if _name_ == "_main_":
    app.run(debug=False)
```