**1. INTRODUCTION**

* 1. Project Overview

The project is entitled as FERTILIZER RECOMMEDATION SYSTEM FOR DISEASE PREDICTION. This project is a simple Machine Learning and Deep Learning based website which identify the diseases affected your crops and recommends the best fertilizers to use.

* 1. Purpose

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.

**2. LITERATURE SURVEY**

2.1 Existing problem

Agriculture is the main aspect for the economic development of a country. Agriculture is the heart and life of most Indians. But in recent days, the field was going down due to various natural calamities. To overcome the problem, various issues in this field need to be addressed. The soil type, fertilizer recommendation, diseases in plants and leaves. All these features need to be considered.

2.2 References

Semi-automatic leaf disease detection and classification system for soybean culture IET Image Processing, 2018.

Cloud Based Automated Irrigation and Plant Leaf Disease Detection System Using an Android Application. International Conference on Electronics, Communication and Aerospace Technology, ICECA 2017.

Ms. Kiran R. Gavhale, Ujwalla Gawande, Plant Leaves Disease detection using Image Processing Techniques, January 2014.

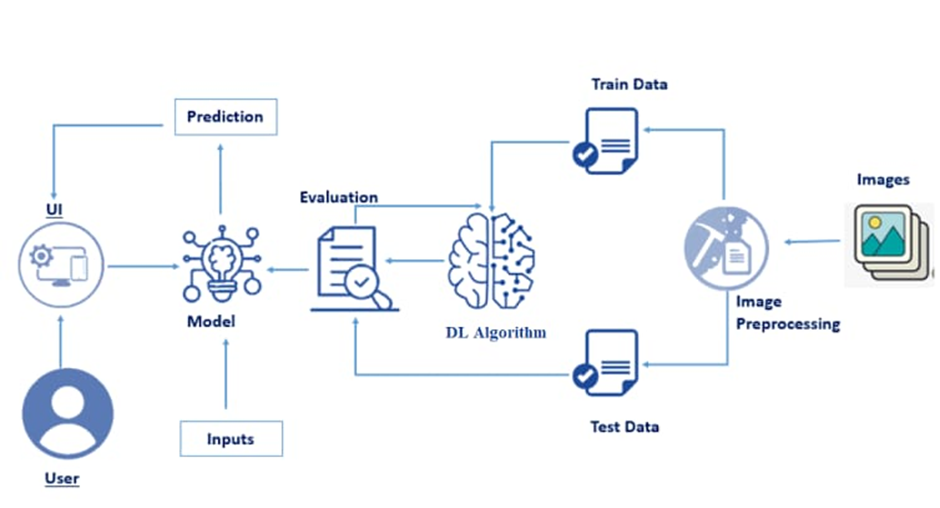
Duan Yan-e, Design of Intelligent Agriculture Management Information System Based on IOTǁ, IEEE,4th, Fourth International reference on Intelligent Computation Technology and Automation, 2011.

2.3 Problem Statement Definition

In India, The Agriculture industry is extremely vital and crucial for economic and social development and jobs. In India, the agricultural sector provides a living for almost 48% of the population. As per the 2019-2020 economic survey, an Indian farmer's median wage in 16 states is Rupees 2500. Most of the Indian population depends on agriculture for their livelihood. Agriculture gives an opportunity of employment to the village people to develop a country like India on large scale and give a push in the economic sector. Most farmers face the problem of planting an inappropriate crop for their land based on a conventional or non-scientific approach. This is a challenging task for a country like India, where agriculture feeds approximately 42% of the population. And the outcomes for the farmer of choosing the wrong crop for land is moving towards metro city for livelihoods, suicide, quitting the agriculture and give land on lease to industrialist or use for the non-agriculture purpose. The outcome of wrong crop selection is less yield and less profit.

**3. IDEATION & PROPOSED SOLUTION**

3.1 Empathy Map Canvas



3.2 Ideation & Brainstorming

There are many contests and challenges being organized by institutions to motivate younger brains to think far and wide to get great ideas with potential impact that will have a chance to be made into a reality.

Bringing new ideas to make agriculture prosper requires an outsider perspective.

When the need is to make an idea become materialized, a prolonged iterative process of brainstorming with experts in the relevant field has been undergone to evolve a template for the selection criteria that can sieve the ideas and be a ready reckoner implementable in the field of agriculture.

3.3 Proposed Solution

Crop disease in plants is predicted and suitable fertilizer is recommended for better yield. The images of the diseased plants are obtained, and it is pre-processed against the dataset of diseased plants.

Deep Learning Algorithm is used to process the images and then it is evaluated.

Then a model is built on the evaluations, it is then trained using no. of. inputs and predictions are given to the users which subsequently helps in recommending the fertilizers.

The Convolutional layers are used to classify and process the images and further helps in recommending the fertilizers. The image classification steps are:

1. Image acquisition
2. Preprocessing
3. Segmentation
4. Disease Prediction
5. Fertilizer Recommendation

3.4 Problem Solution fit

The solution to the problem is Machine learning, which is one of the applications of Artificial Intelligence, is being used to implement the proposed system. Crop recommendation is going to recommend you the best crop you can grow in your land as per the soil nutrition value and along with as per the climate in that region. And recommending the best fertilizer for every crop is also a challenging task. And the other and most important issue is when a plant gets caught by heterogeneous diseases that effect on less amount of agriculture production and compromises with quality as well. To overcome all these issues this recommendation has been proposed. Nowadays a lot of research and work is being implemented in the smart and modern agriculture domain. Crop recommendation is characterized by a soil database comprised of Nitrogen, Phosphorus, potassium. The ensembles technique is used to build a recommendation model that combines the prediction of multiple machine learning.

**4. REQUIREMENT ANALYSIS**

4.1 Functional Requirement

Uploading Dataset (Leaf) - Images of the leaves is to be uploaded

Requesting solution - Uploaded images is compared with the pre-defined Model and solution is generated

4.2 Non - Functional Requirements

Usability - The system allows the user to perform the tasks easily and efficiently and effectively.

Security - Assuring all data inside the system or its part will be protected against malware attacks or unauthorized access.

Reliability - The website does not recover from failure quickly; it takes time as the application is running in single server.

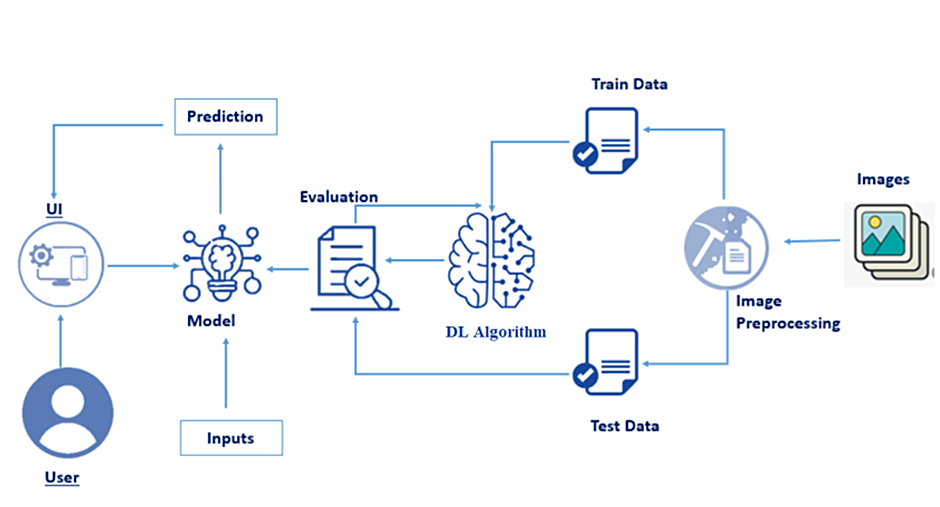
Performance -Response Time and Net Processing Time is Fast.

Availability - The system will be available up to 95% of the time.

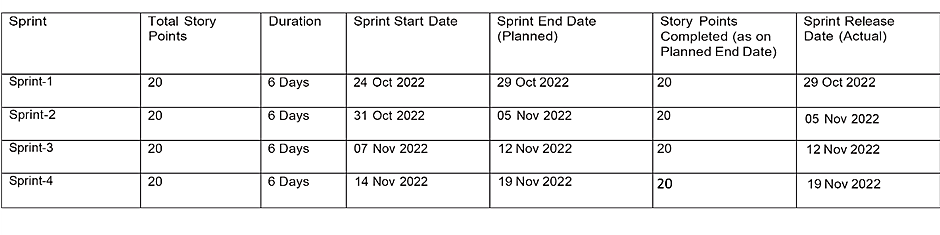
Scalability - The website is scalable.

**5. PROJECT DESIGN**

5.1 Data Flow Diagrams



**6. PROJECT PLANNING & SCHEDULING**

6.1 Sprint Planning & Estimation

6.2 Sprint Delivery Schedule

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Sprint | Functional Requirement | User Story/Task | Priority | Acceptance Criteria | Team Members |
| Sprint 1 | Data collection and pre-processing | As a User, my diseased plant image must be identified in the dataset model | High | Data is required for identifying/ matching with the user input | Deepak Kumar D, Dharankumar AS, Arshat S |
| Sprint 1 Submission | | | | | |
| Sprint 2 | Training the model with preloaded dataset | As a User, my diseased plant image must be predicted by the model | High | The dataset must be trained with the preloaded dataset | Deepak Kumar D, Anisha S, Dharankumar AS, Arshat S |
| Sprint 2 Submission | | | | | |
| Sprint 3 | Testing the model with user’s diseased plant image | As a User, the prediction must be accurate and identify the disease and recommend a fertilizer remedy | High | The test image is pre-processed against the dataset | Arshat S, Deepak Kumar D, Dharankumar AS |
| Sprint 3 Submission | | | | | |
| Sprint 4 | Web Application UI Design | As a User, the website must be attractive and genuine | Low | Better Impression about the website | Arshat S, Deepak Kumar D, Dharankumar AS, Anisha S |
| Sprint 4 Submission | | | | | |

**7. CODING & SOLUTIONING**

7.1 Feature 1

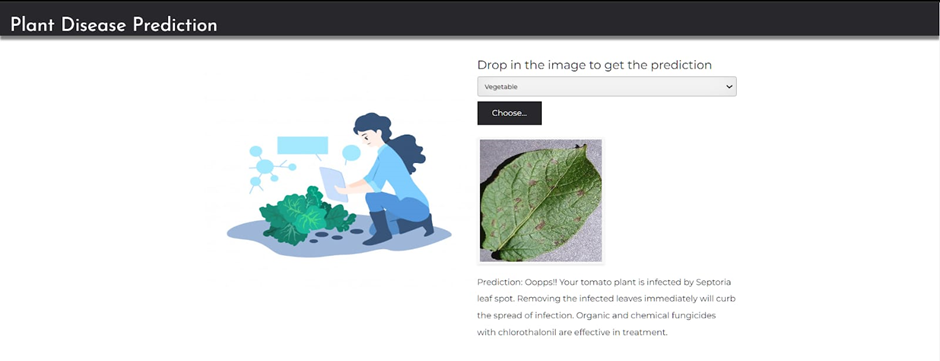
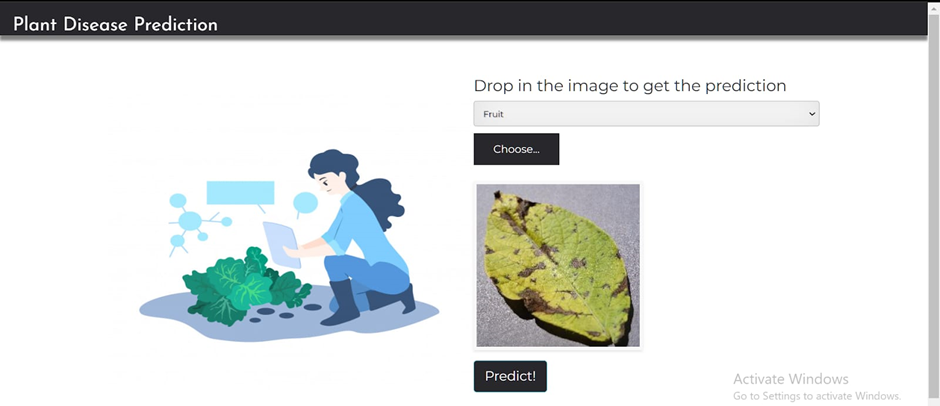
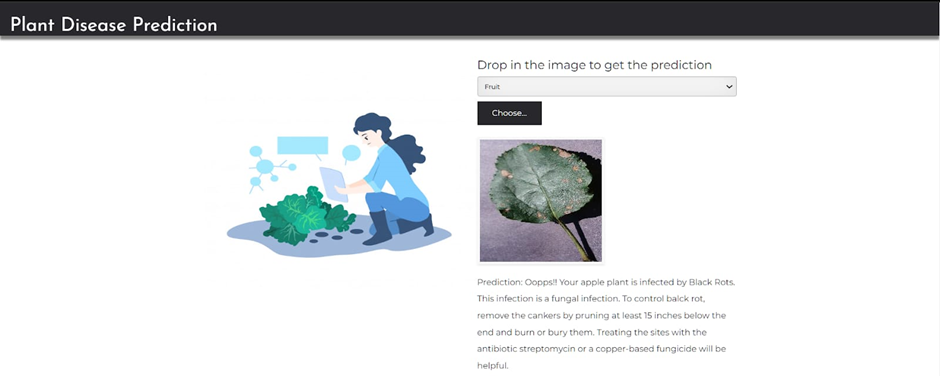
The prediction model is 95% accurate, predicting almost all the diseases in the trained dataset.

Realtime data can be predicted as well.

7.2 Feature 2

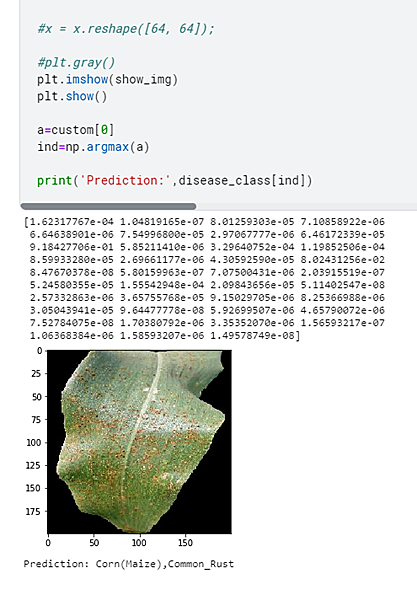
It uses CNN to classify tree leaves, which gives better result compared to SVM which is 0.7.

**8. TESTING**



**9. RESULTS**

9.1 Performance Metrics



**10. ADVANTAGES & DISADVANTAGES**

10.1 Advantages

Fertilizing the diseased crops at the right time might save huge loss to the farmers and prevents spreading.

10.2 Disadvantages

The prediction model is not 100% accurate and might mistake a plant leaf for some other plant leaf and might recommend a wrong fertilizer.

**11. CONCLUSION**

The proposed method uses CNN to classify tree leaves, identify the disease and suggest the fertilizer. The proposed method is compared with the existing SVM based leaf disease prediction. The proposed CNN technique gives a better result when compared to existing SVM. For the same set of images, F-Measure for SVM is 0.7 and 0.8 for CNN, the accuracy of identification of leaf disease of SVM is 0.6 and CNN is 0.8.

**12. FUTURE SCOPE**

This further research is implementing the proposed algorithm with the existing public datasets. Also, various segmentation algorithms can be implemented to improve accuracy. The proposed algorithm can be modified further to identify the disease that affects the various plant organs such as stems and fruits.

**13. APPENDIX**

Source Code

Style.css:

  \*{

    margin: 0;

    padding: 0;

    box-sizing: border-box;

}

.body{

    background-color: rgba(0,0,0,0.5);

}

.main{

    display:flex;

    height: 100vh;

    background-color: grey;

    justify-content: center;

    background-image: url(../static/plantimage.jpg);

    background-position: center;

    background-repeat: no-repeat;

    background-size: cover;

}

.nav\_container{

    display: flex;

    justify-content: center;

    position: absolute;

}

.nav{

    height: 60px;

    background: rgb(31, 28, 28);

    box-shadow: 0 4px 8px 0 rgba(0, 0, 0, 0.2), 0 6px 20px 0 rgba(0, 0, 0, 0.19);

    position: fixed;

    display: flex;

    justify-content: center;

    top: 9px;

    width: 85%;

    z-index: 1;

    border-radius: 25px;

    transition: 0.9s;

}

nav ul{

    margin-right: 25px;

}

nav ul {

    line-height: 50px;

    padding: 5px;

}

nav ul  a{

    color: white;

    font-size: 17px;

    padding: 7px 13px;

    border-radius: 5px;

    text-decoration: none;

    text-transform: uppercase;

}

ul a.active,ul a:hover{

    background: green;

    color: black;

    text-decoration: none;

}

.body{

    margin: auto;

    margin-top: 4.5%;

    height: 89%;

    width: 80%;

    display: flex;

    color: white;

    border-radius: 10px;

}

.left{

    height: 100%;

    width: 40%;

    display: flex;

    justify-content: center;

}

.left img{

    position: absolute;

    top: 200px;

    height: 400px;

}

.right{

    height: 100%;

    width: 60%;

}

#upload-file{

    font-size: 30px;

}

.p{

    font-size: 25px;

    font-weight: 300;

    margin:30px 10px 20px 80px;

    font-family: monospace;

}

.image-section{

    position: relative;

    top: 15px;

    left: 150px;

}

.btn{

    margin: 15px 0px 0px 85px;

}

button{

    width: 150px;

    height: 40px;

    font-size: 30px;

}

#btn-predict{

    position: relative;

    top: -50px;

}

select{

    width: 225px;

    letter-spacing: 2px;

    height: 30px;

    font-size: 20px;

    text-align: center;

    font-family: monospace;

}

input{

    width: 600px;

    letter-spacing: 2px;

    height: 30px;

    font-size: 20px;

    text-align: center;

    font-family: monospace;

}

#result{

    font-family: monospace;

    font-size: 20px;

    font-weight: 400;

    margin: 30px 0px 0px 85px;

}

select{

    background-color: black;

    color: white;

}

\*{

    @import url('https://fonts.googleapis.com/css2?family=Allison&family=IBM+Plex+Sans+Arabic:wght@100;500&display=swap');

    padding: 0;

    margin: 0;

    font-family:'IBM Plex Sans Arabic', sans-serif;

    overflow-y: hidden;

    box-sizing: border-box;

    user-select: none;

}

.homepageintro{

    position: relative;

    top: -20px;

    height: 106vh;

    width: 100%;

    display: flex;

    justify-content: center;

    align-items: center;

    flex-direction: row;

}

.text{

    height: 50%;

    width: 40%;

    top: 300px;

    left: 10%;

    position: absolute;

}

.text h1{

    font-family: Georgia, 'Times New Roman', Times, serif;

    color: black;

    font-size: 60px;

}

.text pre{

    font-family: 'Courier New', Courier, monospace;

    color: white;

    font-size: 20px;

    font-weight: 500;

    margin-left: 10px;

}

.button {

    position: absolute;

    margin-top: 170px;

    left: 165px;

    height: 5vh;

    width: 17vh;

    font-style: italic;

    border-radius: 5px;

    background-color: GREEN;

    color: BLACK;

    border: 2px solid green;

}

.button:hover {

    transition: 0.5s;

    background-color: inherit;

    color: white;

}

.btn{

    color: white;

    background-color: transparent;

    border-radius: 10px;

}

.btn:hover{

    background-color: white;

    color: black;

}

.container{

    display:flex;

    flex-wrap: wrap;

    justify-content: center;

}

.container2{

    position: relative;

    top:80px;

    display:flex;

    flex-wrap: wrap;

    justify-content: center;

}

/\*---------------------header------------------------\*/

.nav\_container{

    display: flex;

    justify-content: center;

}

.nav{

    height: 60px;

    background: rgb(31, 28, 28);

    box-shadow: 0 4px 8px 0 rgba(0, 0, 0, 0.2), 0 6px 20px 0 rgba(0, 0, 0, 0.19);

    position: fixed;

    display: flex;

    justify-content: center;

    top: 9px;

    width: 85%;

    z-index: 1;

    border-radius: 25px;

    transition: 0.9s;

}

nav ul{

    margin-right: 25px;

}

nav ul {

    line-height: 50px;

    padding: 5px;

}

nav ul  a{

    color: white;

    font-size: 17px;

    padding: 7px 13px;

    border-radius: 5px;

    text-decoration: none;

    text-transform: uppercase;

}

ul a.active,ul a:hover{

    background: green;

    color: black;

    text-decoration: none;

}

button{

    padding: 5px;

    font-size: 18px;

    border-radius: 9px;

    color: white;

    transition: .5s ease;

    background: #0082e6;

    margin-top: 5px;

    margin-bottom: 15px;

}

button:hover{

    background: transparent;

    color: black;

}

@media (max-width:950px) {

    .checkbtn{

        display: block;

    }

    nav ul{

        position: fixed;

        width:50%;

        height: 100vh;

        background: lavenderblush;

        top:50px;

        left: -100%;

        transition: .7s ease-out;

        text-align: center;

        margin-top: 20px;

    }

    nav ul li a{

        color: black;

    }

    .nav{

        height: 70px;

    }

    nav ul li{

        display: block;

    }

    nav ul li ul{

       left: 100%;

       width:100%;

       height: 30%;

       background: lavenderblush;

       top: 19%;

    }

    nav ul li ul li a{

        font-size: 13px;

    }

    nav ul li:hover ul{

        visibility: visible;

    }

    a.active,a:hover{

        background: white;

        color: #0082e6;

        padding: 15px;

    }

    #check:checked ~ul{

        left:0;

    }

    .description{

        margin-top: 60px;

    }

    .d2{

        visibility: hidden;

    }

}

Home.html:

<!DOCTYPE html>

<html>

<head>

    <meta charset='utf-8'>

    <meta http-equiv='X-UA-Compatible' content='IE=edge'>

    <title>Plant Disease Prediction</title>

    <meta name='viewport' content='width=device-width, initial-scale=1'>

    <link rel='stylesheet' type='text/css' media='screen' href='../static/style1.css'>

    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">

</head>

<body  style="margin-bottom: 0px;">

    <div class="nav\_container">

        <div class="nav">

            <nav>

                <ul>

                    <a class="active" href="/">Home</a>

                    <a  href="/Predict">Predict</a>

                </ul>

            </nav>

        </div>

    </div>

    <div class="homepageintro">

            <img src="../static/plantimage.jpg">

            <div class="text">

                <h1>Plant Disease Prediction</h1>

                <pre>

If you prefer your plant dead or dying

because then you can diagnose them, you

are probably a plant pathologist</pre>

            </div>

            <div class="buttoncontainer">

                <button class="button" onclick="home\_container">Explore</button>

            </div>

        </div>

        </div>

    </div>

    <script type="text/javascript">

        function googleTranslateElementInit() {

          new google.translate.TranslateElement({pageLanguage: 'en', layout: google.translate.TranslateElement.InlineLayout.SIMPLE}, 'google\_translate\_element');

        }

        </script>

    <script type="text/javascript" src="//translate.google.com/translate\_a/element.js?cb=googleTranslateElementInit"></script>

</body>

</html>

Predict.html:

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <link rel="stylesheet" href="../static/Style.css">

    <script src="../static/javascript.js"></script>

    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js"></script>

    <title>Document</title>

</head>

<body>

    <div class="main">

    <div class="nav\_container">

        <div class="nav">

            <nav>

                <ul>

                    <a href="home.html">Home</a>

                    <a  class="active" href="/">Predict</a>

                </ul>

            </nav>

        </div>

    </div>

    <div class="body">

        <div class="left">

            <img src="../static/68544440-37ffdd80-03e9-11ea-8acd-3f3f9b6fc8b3.png" alt="Unavailable">

        </div>

        <div class="right">

        <div class="p" style="font-weight:600">Drop a Image to start the process of prediction</div>

        <div class="imgtag">

            <form id="upload-file" method="post" enctype="multipart/form-data" >

                <div class="p p1" style="display:inline ;">Choose a Plant Type:</div>

                <select name="Plant" id="Plant">

                <option value="Vegetable">Vegetable</option>

                <option value="Fruit">Fruit</option>

                </select><br>

                 <input type="file" name="file" class="btn btn-success" id="imageUpload" accept=".png, .jpg, .jpeg" onchange="loadFile(event)">

            </form>

            </div>

            <div id="image-section1" class="image-section" style="display:none;">

                 <img id="imagePreview"  class="img-responsive" src="#" style="width:300px;height:300px;"/><br><br><br><br>

                 <div class="choose">

                    <button type="button" class="btn btn-info btn-lg " id="btn-predict" onclick="predict()">Predict</button>

                 </div>

            </div>

                    <div class="loader" style="display:none;"></div>

                    <h3 id="result">

                <span> </span>

            </h3>

        </div>

        </div>

    </div>

</body>

</html>

App.py

import os

import tensorflow as tf

import numpy as np

import pandas as pd

from tensorflow import keras

from skimage import io

from tensorflow.keras.preprocessing import image

from flask import Flask, redirect, url\_for, request, render\_template

from werkzeug.utils import secure\_filename

from gevent.pywsgi import WSGIServer

app = Flask(\_\_name\_\_)

vegmodel = tf.keras.models.load\_model("VegetableModel.h5", compile=False)

fruitmodel = tf.keras.models.load\_model("FruitModel.h5", compile=False)

def veg\_model\_predict(img\_path, model):

    img = image.load\_img(img\_path, grayscale=False, target\_size=(64, 64))

    show\_img = image.load\_img(img\_path, grayscale=False, target\_size=(64, 64))

    x = image.img\_to\_array(img)

    x = np.expand\_dims(x, axis=0)

    x /= 255

    preds = vegmodel.predict(x)

    return preds

def fruit\_model\_predict(img\_path, model):

    img = image.load\_img(img\_path, grayscale=False, target\_size=(64, 64))

    show\_img = image.load\_img(img\_path, grayscale=False, target\_size=(64, 64))

    x = image.img\_to\_array(img)

    x = np.expand\_dims(x, axis=0)

    x /= 255

    preds = fruitmodel.predict(x)

    return preds

@app.route("/", methods=["GET"])

def index():

    return render\_template("home.html")

@app.route("/Predict", methods=["GET", "POST"])

def route():

    return render\_template("predict.html")

@app.route("/predict", methods=["GET", "POST"])

def upload():

    if request.method == "POST":

        print

        f = request.files["file"]

        basepath = os.path.dirname(\_\_file\_\_)

        file\_path = os.path.join(basepath, "uploads", secure\_filename(f.filename))

        f.save(file\_path)

        plant = request.form["Plant"]

        disease\_class = [

            "Apple Black Rot",

            "Apple Cedar Rust",

            "Apple Healthy",

            "Apple Scab",

            "Blueberry Healthy",

            "Cherry(Including\_Sour) Healthy",

            "Cherry(Including\_Sour) Powdery Mildew",

            "Corn(Maize) Cercospora Gray leaf spot",

            "Corn(Maize) Common Rust",

            "Corn(Maize) Northern Leaf Blight",

            "Corn(Maize) healthy",

            "Grape Black Rot",

            "Grape Esca (Black\_Measles)",

            "Grape Healthy",

            "Grape Leaf Blight (Isariopsis\_Leaf\_Spot)",

            "Misc",

            "Orange Haunglongbing (Citrus\_Greening)",

            "Peach Bacterial Spot",

            "Peach Healthy",

            "Pepper Bell Bacterial Spot",

            "Pepper Healthy",

            "Potato Early blight",

            "Potato Healthy",

            "Potato Late Blight",

            "Raspberry Healthy",

            "Soybean Healthy",

            "Squash Powdery Mildew",

            "Strawberry Healthy",

            "Strawberry Leaf Scorch",

            "Tomato Bacterial Spot",

            "Tomato Early Blight",

            "Tomato Healthy",

            "Tomato Late Blight",

            "Tomato Leaf Mold",

            "Tomato Mosaic Virus",

            "Tomato Septoria leaf Spot",

            "Tomato Spider Mite",

            "Tomato Target Spot",

            "Tomato Yellow Leaf Curl Virus",

        ]

        if plant == "Vegetable":

            preds = veg\_model\_predict(file\_path, veg\_model\_predict)

        else:

            preds = fruit\_model\_predict(file\_path, fruit\_model\_predict)

        a = preds[0]

        ind = np.argmax(a)

        result = disease\_class[ind]

        df = pd.read\_excel("Disease Fertilizer.xlsx")

        for row in range(len(df)):

            if df.loc[row, "Disease"] == result:

                fertilizer=df.loc[row, "Description"]

                result+=fertilizer

                break

        return result

    return None

if \_\_name\_\_ == "\_\_main\_\_":

    app.run()

GitHub & Project Demo Link

<https://github.com/IBM-EPBL/IBM-Project-25811-1659973903>

<https://drive.google.com/file/d/1xjh9pPhDztncxueDOLm7S9ZmWvQZSDi9/view?usp=share_link>