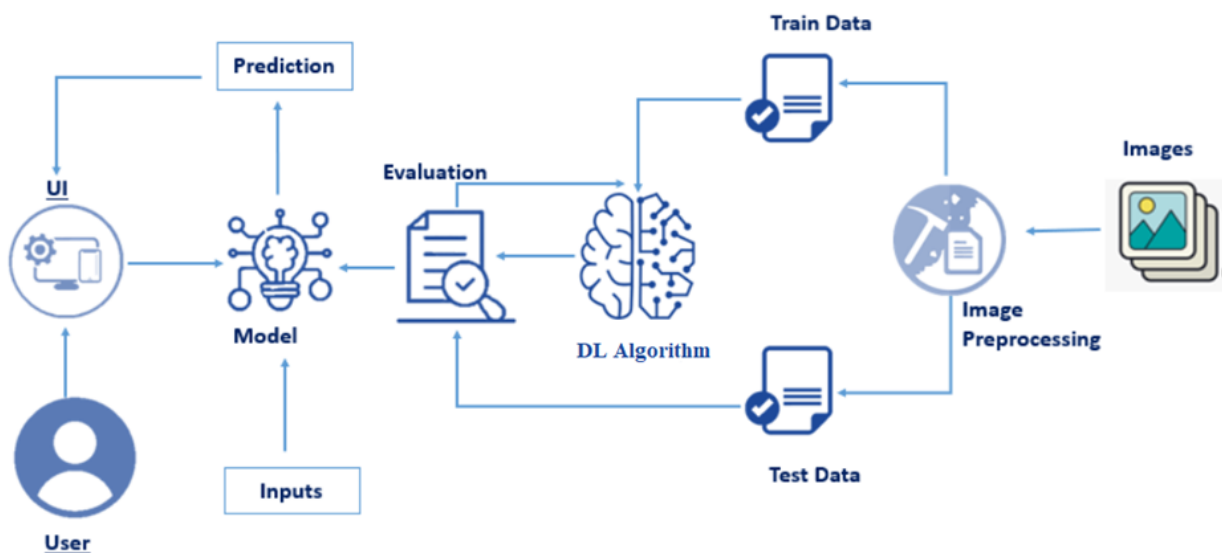


Fertilizers Recommendation System For Disease Prediction

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.



To complete the project following tasks are performed using python in spyder ide and user interface using flask

To accomplish the above task you must complete the below activities and tasks

- Download the dataset.

- Classify the dataset into train and test sets.
- Add the neural network layers.
- Load the trained images and fit the model.
- Test the model.
- Save the model and its dependencies.
- Build a Web application using a flask that integrates with the model built.
- Download the dataset.

Download the dataset

The dataset has been downloaded from the link provided in the workspace. It includes fruit and vegetable dataset

<https://drive.google.com/file/d/1fxs7ptl6zh7NTbCOZARKZ7AmYKjnprY/view?usp=sharing>

Classify the dataset into train and test sets

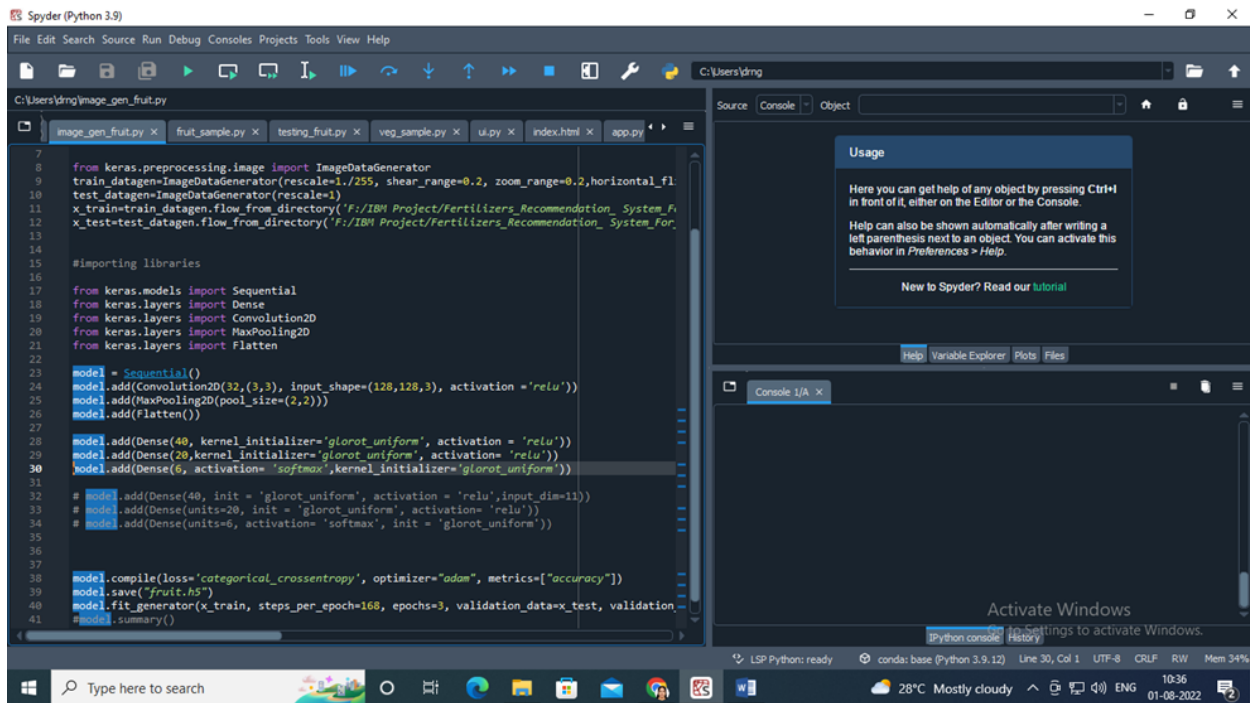
The dataset has been classified as test and train sets.

Image Preprocessing

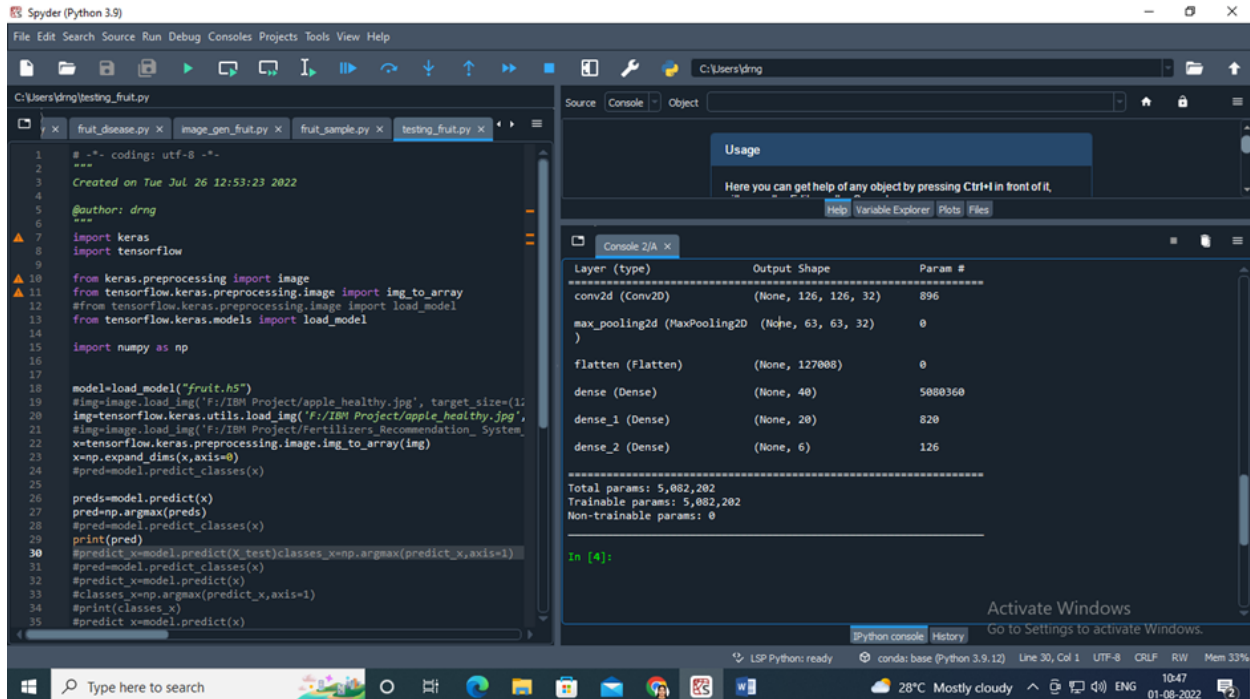
Before training the model images are preprocessed and then fed into the model for training. To perform this task I use of Keras ImageDataGenerator class for image preprocessing.

Model building

Model is built for fruit/vegetable leaf disease detection.

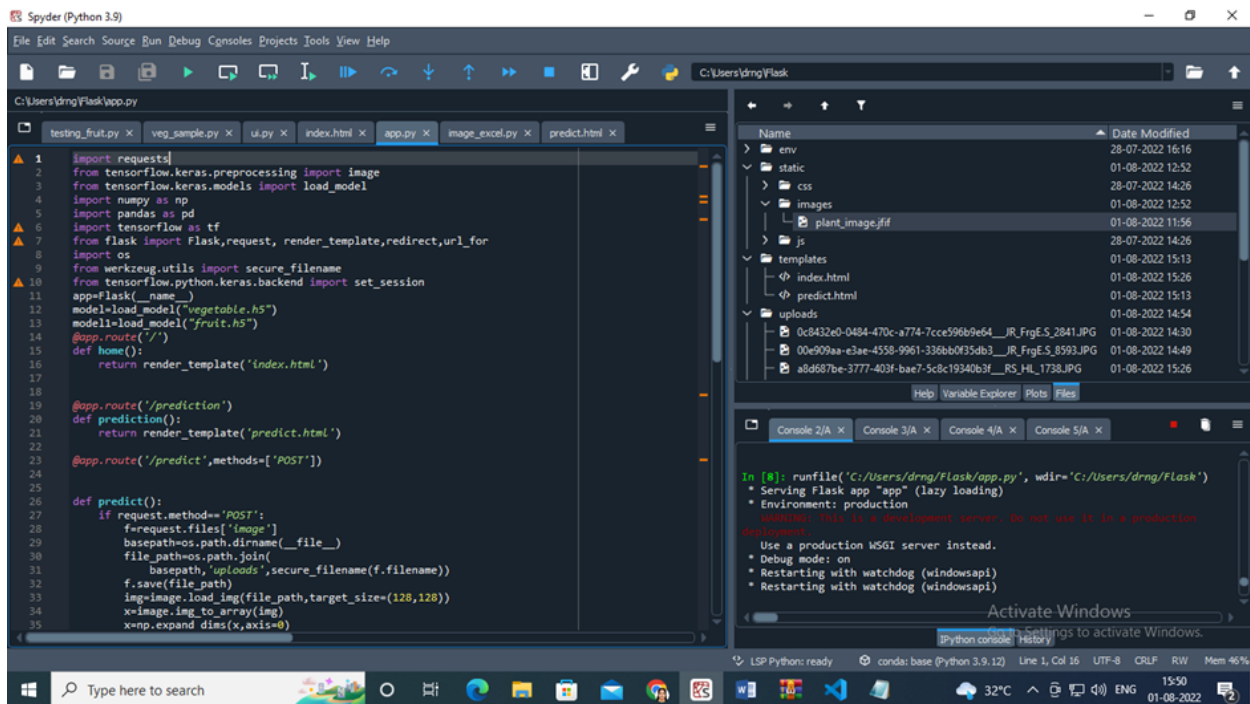


After Model is built, it is tested and the following output is obtained.



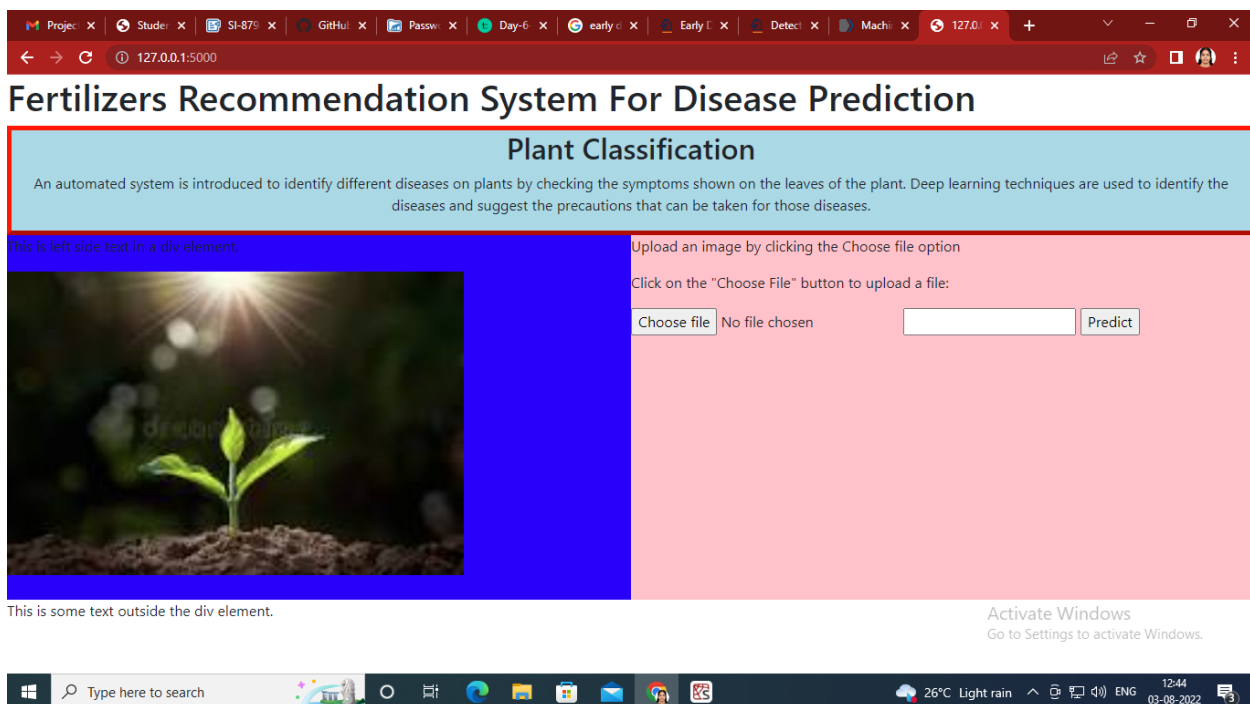
The model is tested with sample input imagewith the help of flask user

interface. The coding for the user interface is given in app.py as shown below.



```
1 import requests
2 from tensorflow.keras.preprocessing import image
3 from tensorflow.keras.models import load_model
4 import numpy as np
5 import pandas as pd
6 import tensorflow as tf
7 from flask import Flask, request, render_template, redirect, url_for
8 import os
9 from werkzeug.utils import secure_filename
10 from tensorflow.python.keras.backend import set_session
11 app = Flask(__name__)
12 model = load_model("vegetable.h5")
13 model1 = load_model("fruit.h5")
14 @app.route("/")
15 def home():
16     return render_template("index.html")
17
18 @app.route("/prediction")
19 def prediction():
20     return render_template("predict.html")
21
22 @app.route("/predict", methods=['POST'])
23
24
25
26 def predict():
27     if request.method == 'POST':
28         f = request.files["image"]
29         basepath = os.path.dirname(__file__)
30         file_path = os.path.join(
31             basepath, 'uploads', secure_filename(f.filename))
32         f.save(file_path)
33         img = image.load_img(file_path, target_size=(128, 128))
34         x = image.img_to_array(img)
35         x = np.expand_dims(x, axis=0)
```

On executing this code, the prediction is carried out and the results are rendered in index.html as below



On executing this code, the prediction is carried out and the results are

rendered in predict.html as below

