In [ ]:

```
#Import necessary libraries
from flask import Flask, render_template, request

import numpy as np
import os

from tensorflow.keras.preprocessing.image import load_img
from tensorflow.keras.preprocessing.image import img_to_array
from tensorflow.keras.models import load_model

filepath = r"E:\5th sem\NLP\project\Plant-Leaf-Disease-Prediction-main\model.h5"
model = load_model(filepath)
print(model)

print("Model Loaded Successfully")

def pred_tomato_dieas(tomato_plant):
  test_image = load_img(tomato_plant, target_size = (128, 128)) # load image
  print("@@ Got Image for prediction")

  test_image = img_to_array(test_image)/255 # convert image to np array and normalize
  test_image = np.expand_dims(test_image, axis = 0) # change dimention 3D to 4D

  result = model.predict(test_image) # predict diseased palnt or not
  print('@@ Raw result = ', result)

  pred = np.argmax(result, axis=1)
  print(pred)
  if pred==0:
      return "Tomato - Bacteria Spot Disease", 'Tomato-Bacteria Spot.html'

  elif pred==1:
      return "Tomato - Early Blight Disease", 'Tomato-Early_Blight.html'

  elif pred==2:
      return "Tomato - Healthy and Fresh", 'Tomato-Healthy.html'

  elif pred==3:
      return "Tomato - Late Blight Disease", 'Tomato - Late_blight.html'

  elif pred==4:
      return "Tomato - Leaf Mold Disease", 'Tomato - Leaf_Mold.html'
```

```
  elif pred==5:
      return "Tomato - Septoria Leaf Spot Disease", 'Tomato - Septoria_leaf_spot.html'

  elif pred==6:
      return "Tomato - Target Spot Disease", 'Tomato - Target_Spot.html'

  elif pred==7:
      return "Tomato - Tomoato Yellow Leaf Curl Virus Disease", 'Tomato - Tomato_Yellow_Leaf_Curl_Virus.html'
  elif pred==8:
      return "Tomato - Tomato Mosaic Virus Disease", 'Tomato - Tomato_mosaic_virus.html'

  elif pred==9:
      return "Tomato - Two Spotted Spider Mite Disease", 'Tomato - Two-spotted_spider_mite.html'



# Create flask instance
app = Flask(__name__)

# render index.html page
@app.route("/", methods=['GET', 'POST'])
def home():
        return render_template('index.html')


# get input image from client then predict class and render respective .html page for solution
@app.route("/predict", methods = ['GET','POST'])
def predict():
     if request.method == 'POST':
        file = request.files['image'] # fet input
        filename = file.filename
        print("@@ Input posted = ", filename)

        file_path = os.path.join(r"E:\5th sem\NLP\project\Plant-Leaf-Disease-Prediction-main\static\upload", filename)
        file.save(file_path)

        print("@@ Predicting class......")
        pred, output_page = pred_tomato_dieas(tomato_plant=file_path)

        return render_template(output_page, pred_output = pred, user_image = file_path)

# For local system & cloud
if __name__ == "__main__":
    app.run(threaded=False,port=8080)
```

```
C:\Users\T VENKAT SAI PRATHAP\anaconda3\lib\site-packages\scipy\__init__.py:146: UserWarning: A NumPy version >=1.16.5 an
d <1.23.0 is required for this version of SciPy (detected version 1.23.0
  warnings.warn(f"A NumPy version >={np_minversion} and <{np_maxversion}"
<keras.engine.sequential.Sequential object at 0x000001AF39E17550>
Model Loaded Successfully
 * 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
 * Running on http://127.0.0.1:8080/ (Press CTRL+C to quit)
127.0.0.1 - - [25/Sep/2022 14:00:57] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [25/Sep/2022 14:00:57] "GET /favicon.ico HTTP/1.1" 404 -
@@ Input posted =  Tomato___Early_blight (2).JPG
@@ Predicting class......
@@ Got Image for prediction
1/1 [==============================] - 2s 2s/step
127.0.0.1 - - [25/Sep/2022 14:02:54] "POST /predict HTTP/1.1" 200 -
@@ Raw result =  [[4.2397756e-02 9.9968201e-01 6.8095960e-06 9.9659353e-01 9.9983364e-01
  9.9635094e-01 5.5650221e-03 3.5719045e-02 1.1371197e-05 9.1179150e-01]]
[4]
127.0.0.1 - - [25/Sep/2022 14:02:54] "GET /static/images/Tomato___Leaf_Mold.JPG HTTP/1.1" 200 -
@@ Input posted =  Tomato___Early_blight (1).JPG
@@ Predicting class......
@@ Got Image for prediction
1/1 [==============================] - 0s 54ms/step
127.0.0.1 - - [25/Sep/2022 14:03:11] "POST /predict HTTP/1.1" 200 -
@@ Raw result =  [[7.72157252e-01 9.97671247e-01 5.55676641e-04 9.31212246e-01
  9.68938649e-01 7.95125961e-01 1.10570574e-04 7.01098919e-01
  3.74734693e-04 1.03741489e-01]]
[1]
```

In [ ]: