Crop Disease Diagnosis for Maize and Coffee

1. Use Case Description

- **Problem**: Farmers in Rwanda and across Africa face significant challenges in identifying and managing diseases in maize and coffee crops. These diseases lead to reduced yields, economic losses, and food insecurity.
- **Solution**: A digital tool for diagnosing crop diseases and recommending treatments. **Relevance**:
    - Maize and coffee are critical crops for food security and economic stability in Rwanda.
    - Early diagnosis and treatment can prevent yield losses and improve farmers' livelihoods.
    - The tool will empower farmers with limited access to agricultural experts.

2. Stakeholders

Stakeholders are individuals or groups who have an interest in or are affected by the crop disease diagnosis system. Here's a detailed breakdown:

Primary Stakeholders:

a) **Farmers**:
    - **Role**: End-users of the diagnostic tool.
    - **Needs**: Accurate and timely disease diagnosis, affordable treatment options, and easy-to-use tools.

b) **Agricultural Experts**:
    - **Role**: Provide knowledge on disease symptoms, treatments, and prevention methods.
    - **Needs**: A reliable system to disseminate their expertise to farmers.

c) **Government Agencies**:
    - **Role**: Support agricultural development and food security initiatives.
    - **Needs**: Data on disease prevalence to inform policy decisions and allocate resources.

d) **NGOs/Development Organizations**:
    - **Role**: Promote sustainable farming practices and provide support to farmers.
    - **Needs**: Tools to enhance their outreach and impact.

e) **Technology Providers**:

- **Role**: Develop and maintain the diagnostic tool. ○ **Needs**: Clear requirements and feedback to improve the system.

Secondary Stakeholders:

a) **Local Communities**:
   - **Role**: Benefit from improved food security and economic stability.

b) **Researchers**: ○ **Role**: Study disease patterns and develop improved solutions.

c) **Input Suppliers**:
   - **Role**: Provide seeds, fertilizers, and pesticides recommended by the system

3. Identified Rules (IF-THEN Rules)

- **Maize Diseases**:

a) **IF** leaves have grayish lesions with yellow halos, **THEN** it is **Northern Leaf Blight**.
   - **Treatment**: Apply fungicides containing chlorothalonil or mancozeb.

b) **IF** leaves have small, oval, or elongated spots with a gray center, **THEN** it is **Gray Leaf Spot**.
   - **Treatment**: Use resistant maize varieties and apply fungicides.

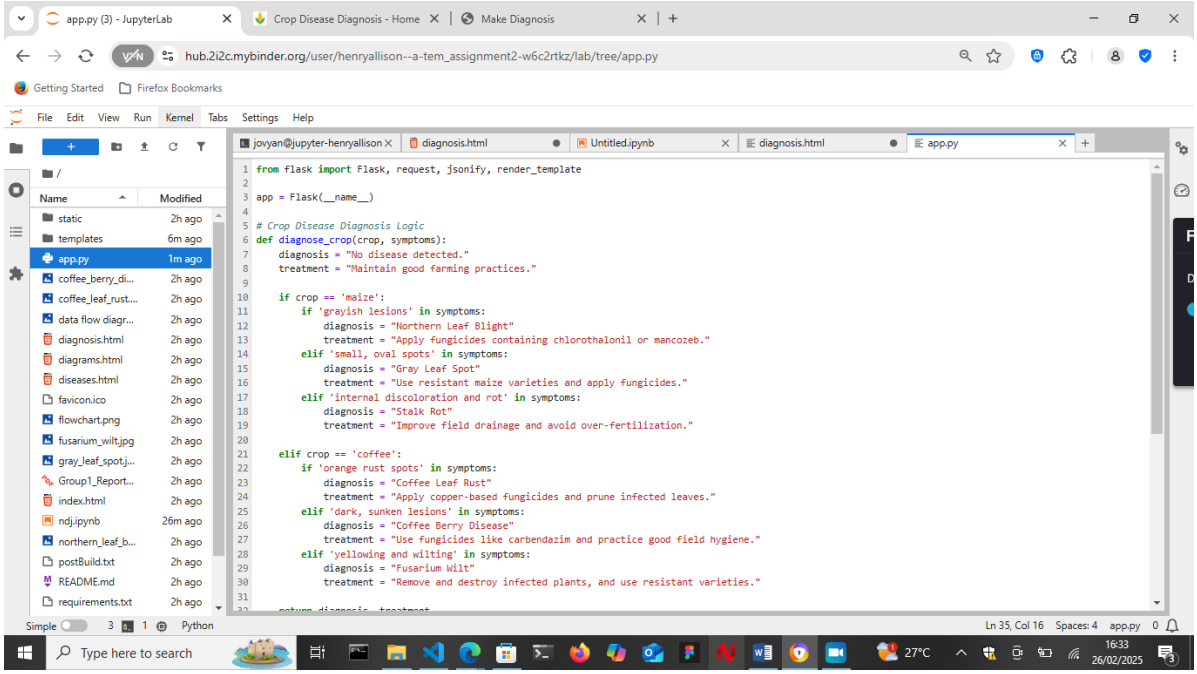
c) **IF** the stalk has internal discoloration and rot, **THEN** it is **Stalk Rot**.
   - **Treatment**: Improve field drainage and avoid over-fertilization.
   - **Coffee Diseases**:

a) **IF** leaves have orange rust spots, **THEN** it is **Coffee Leaf Rust**.
   - **Treatment**: Apply copper-based fungicides and prune infected leaves.

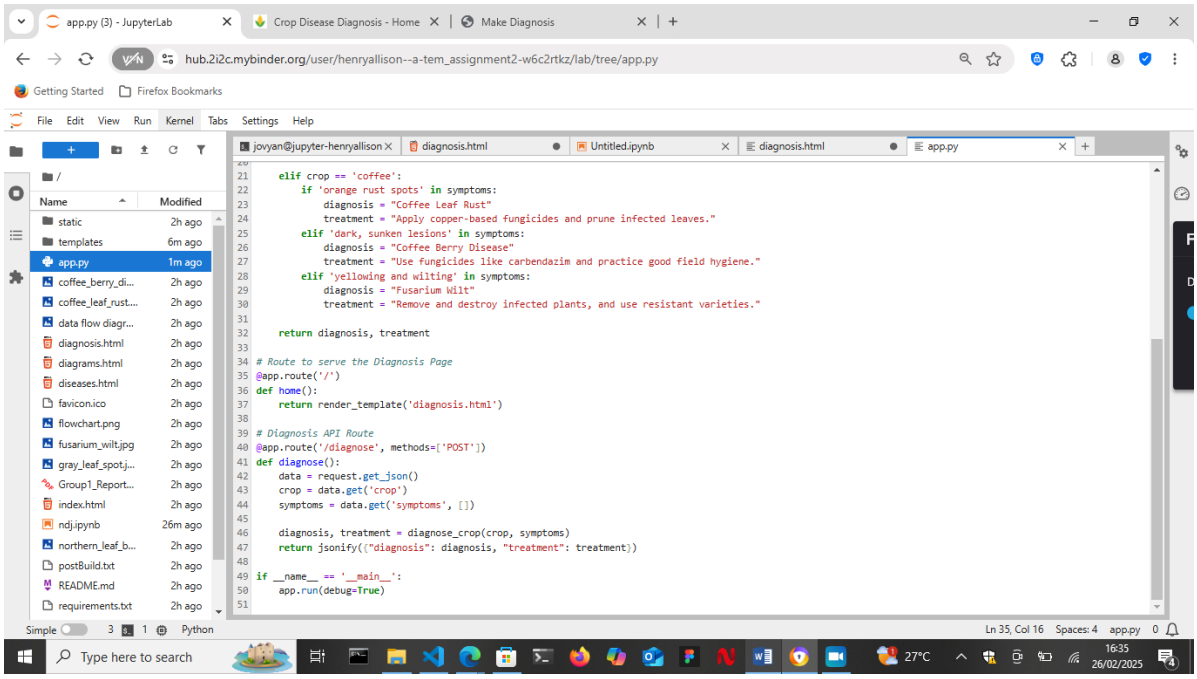
b) **IF** berries have dark, sunken lesions, **THEN** it is **Coffee Berry Disease**.
   - **Treatment**: Use fungicides like carbendazim and practice good field hygiene.

c) **IF** leaves show yellowing and wilting, **THEN** it is **Fusarium Wilt**.
   - **Treatment**: Remove and destroy infected plants, and use resistant varieties.

Python Flask backend codes



```python
from flask import Flask, request, jsonify, render_template

app = Flask(__name__)

# Crop Disease Diagnosis Logic
def diagnose_crop(crop, symptoms):
    diagnosis = "No disease detected."
    treatment = "Maintain good farming practices."

    if crop == 'maize':
        if 'grayish lesions' in symptoms:
            diagnosis = "Northern Leaf Blight"
            treatment = "Apply fungicides containing chlorothalonil or mancozeb."
        elif 'small, oval spots' in symptoms:
            diagnosis = "Gray Leaf Spot"
            treatment = "Use resistant maize varieties and apply fungicides."
        elif 'internal discoloration and rot' in symptoms:
            diagnosis = "Stalk Rot"
            treatment = "Improve field drainage and avoid over-fertilization."

    elif crop == 'coffee':
        if 'orange rust spots' in symptoms:
            diagnosis = "Coffee Leaf Rust"
            treatment = "Apply copper-based fungicides and prune infected leaves."
        elif 'dark, sunken lesions' in symptoms:
            diagnosis = "Coffee Berry Disease"
            treatment = "Use fungicides like carbendazim and practice good field hygiene."
        elif 'yellowing and wilting' in symptoms:
            diagnosis = "Fusarium Wilt"
            treatment = "Remove and destroy infected plants, and use resistant varieties."

    return diagnosis, treatment
```



```python
    elif crop == 'coffee':
        if 'orange rust spots' in symptoms:
            diagnosis = "Coffee Leaf Rust"
            treatment = "Apply copper-based fungicides and prune infected leaves."
        elif 'dark, sunken lesions' in symptoms:
            diagnosis = "Coffee Berry Disease"
            treatment = "Use fungicides like carbendazim and practice good field hygiene."
        elif 'yellowing and wilting' in symptoms:
            diagnosis = "Fusarium Wilt"
            treatment = "Remove and destroy infected plants, and use resistant varieties."

    return diagnosis, treatment

# Route to serve the Diagnosis Page
@app.route('/')
def home():
    return render_template('diagnosis.html')

# Diagnosis API Route
@app.route('/diagnose', methods=['POST'])
def diagnose():
    data = request.get_json()
    crop = data.get('crop')
    symptoms = data.get('symptoms', [])

    diagnosis, treatment = diagnose_crop(crop, symptoms)
    return jsonify({"diagnosis": diagnosis, "treatment": treatment})

if __name__ == '__main__':
    app.run(debug=True)
```