

Smart Green System

Final Review

Group number:14

Guided by

Mr. Sherikh K K

Assistant Professor

ADS Department

Presented by

Fathimma Sana

Farah Fahmi

Ameen Ali

CONTENTS

- Introduction
- Literature Survey
- Problem Statement
- Methodology
- Design and Implementation
- Working Principle
- Experimental Setup
- Result & Analysis
- Conclusion
- Future Scope
- References

INTRODUCTION

- Agriculture faces critical challenges in yield prediction, disease diagnosis, weed management and medicinal plant identification
- These issues directly affect productivity, profitability, and environmental sustainability.
- Inaccurate predictions and delayed detection lead to significant crop losses.
- Advanced machine learning and deep learning models offer transformative solutions.
- Techniques like Naïve Bayes, Random Forest, and CNNs enable precise analysis of field data.
- Early detection and targeted interventions reduce resource waste and environmental impact.

Objective

- Optimize crop management through data-driven, efficient decision-making.

LITERATURE SURVEY

Title	Authors	Methods	Results
Crop Prediction Model Using Machine Learning Algorithms	Ersin Elbasi, Chamseddine Zaki, Ahmet E. Topcu, Wiem Abdelbaki, Aymen I. Zreikat, Elda Cina, Ahmed Shdefat, and Louai Saker	Used Naïve Bayes, Decision Trees, and Random Forest for crop yield prediction based on soil, weather, and crop data.	Improved yield prediction accuracy but needs real-world validation.
Machine Learning Models for Predicting Crop Yield and Disease Detection	Andrew J, Jennifer Eunice , Daniela Elena Popescu , M. Kalpana Chowdary and Jude Hemanth	Applied CNN and LSTM for disease detection and yield prediction with image augmentation and environmental analysis.	Enhanced disease detection accuracy but requires adaptation for different regions.
Weed Species Identification in Different Crops Using Precision Weed Management: A Review	Anand Muni Mishra and Vinay Gautam	Trained deep learning models on labeled datasets using CNNs for weed identification.	Reduced herbicide use but demands high computational power and large datasets.
Deep Learning-Based Medicinal plant identification Using Images for Agricultural Applications	S. Kavitha, T. Satish Kumar,E.Naresh, Vijay H. Kalmani,Kalyan Devappa Bamane,Piyush Kumar Pareek	Used CNN and BPNN models with image preprocessing and augmentation for medicinal plant identification.	Achieved high accuracy and effective model for identification but requires extensive datasets and resources.

PROBLEM STATEMENT

- Crop diseases, inaccurate yield prediction, and weed infestations cause major agricultural losses.
- Traditional detection methods are slow, inefficient, and require expert intervention.
- Machine learning and deep learning improve accuracy but require large datasets and high computational power.
- Existing models struggle to generalize across different crops and environmental conditions.
- A real-time, automated disease detection and yield prediction and medicinal plant identification is essential.
- Integrating CNN and LSTM enhances accuracy and efficiency in crop monitoring.
- This solution helps farmers take timely action, improving productivity and sustainability.

METHODOLOGY

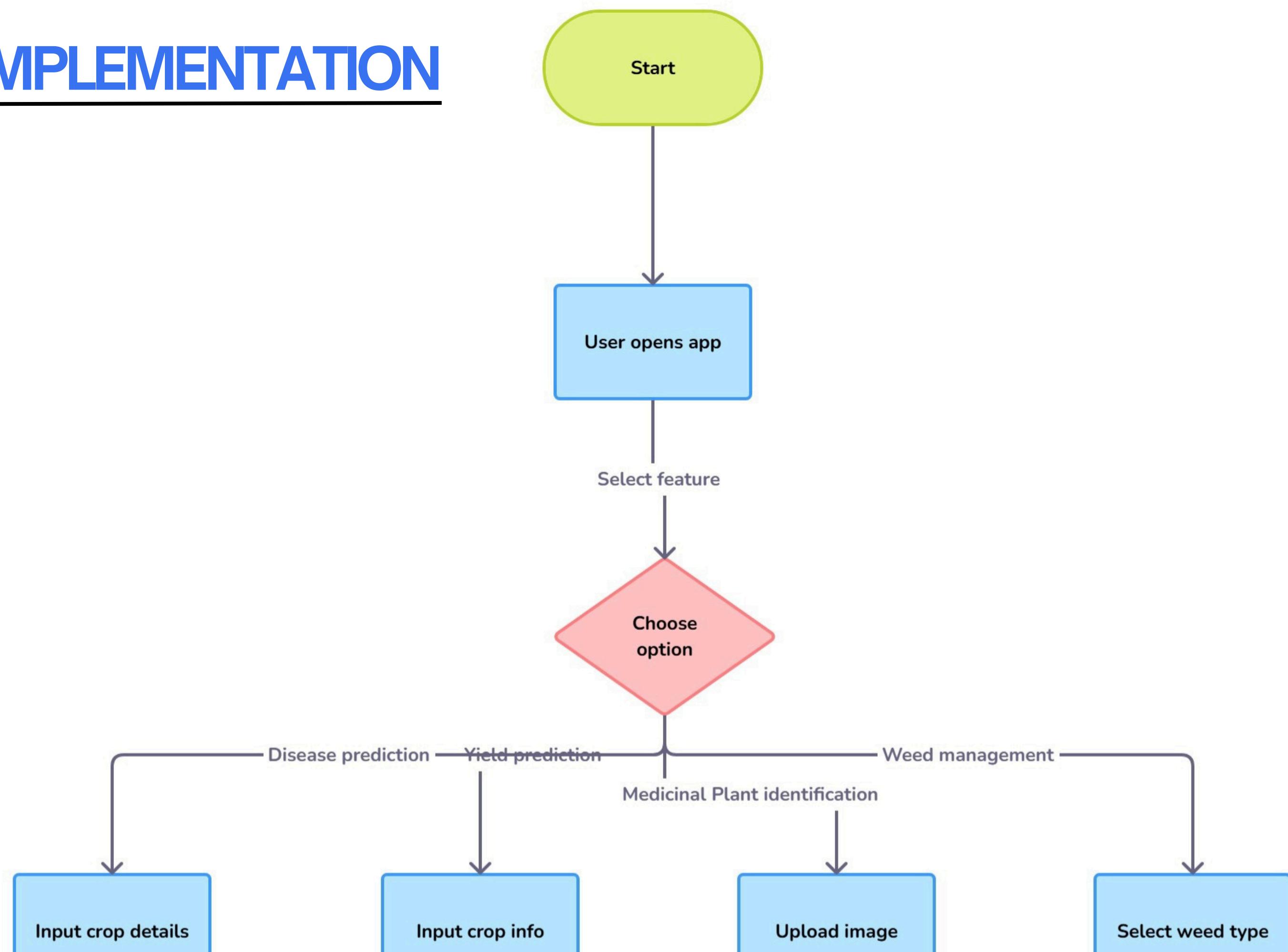
Approaches

- User uploads image and environmental data.
- OpenCV processes image, CNN detects disease.
- LSTM predicts yield, results are displayed.

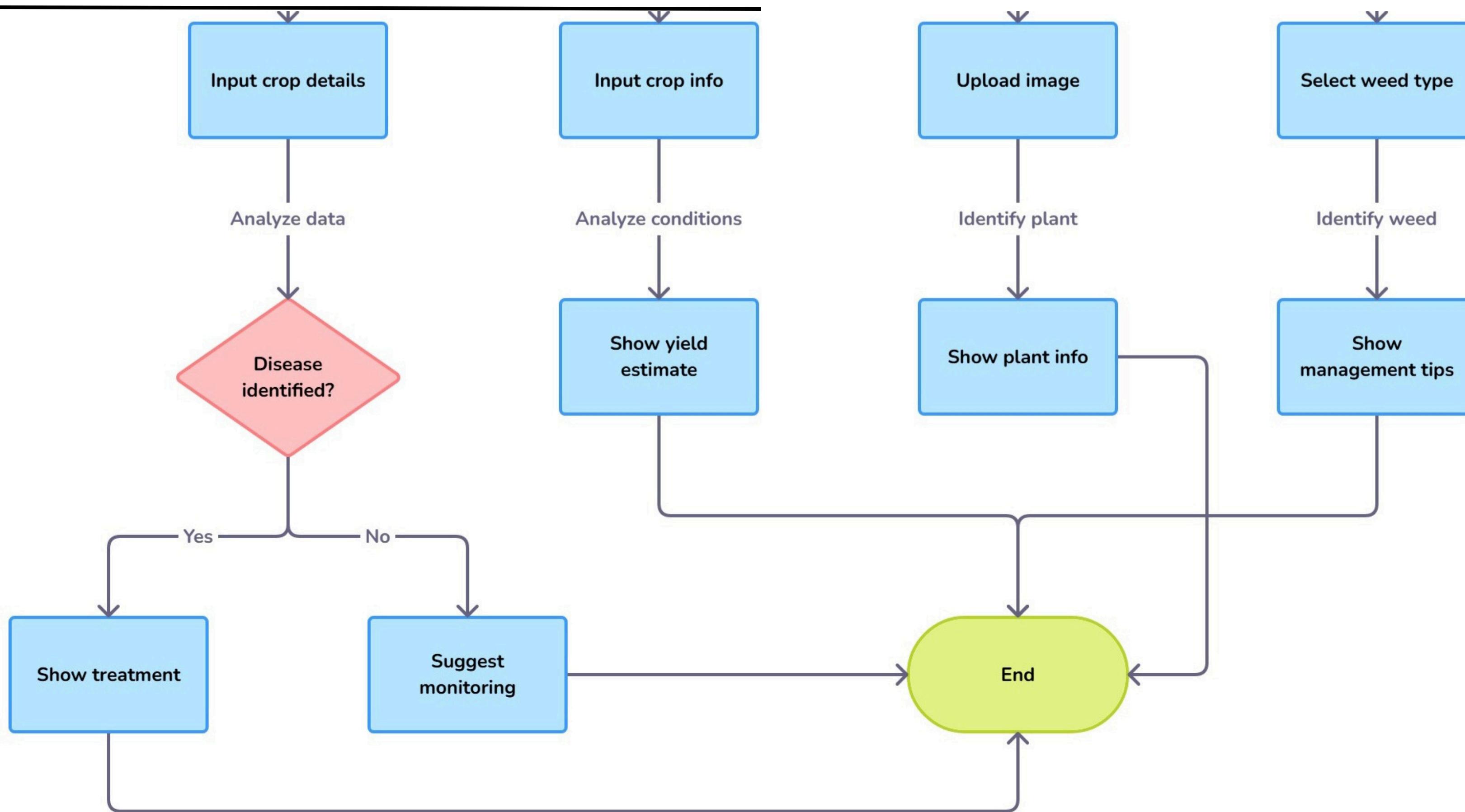
Tools and Software:

- Languages: Python, HTML/CSS, JavaScript.
- Frameworks: Flask, TensorFlow/Keras.
- Libraries: OpenCV, SQLite.

DESIGN AND IMPLEMENTATION



DESIGN AND IMPLEMENTATION



WORKING PRINCIPLE

Identification & Prediction Systems

1.Yield Prediction

- The model uses a neural network to learn patterns from environmental data to classify crops.
- The softmax activation function in the output layer predicts the most suitable crop by assigning probabilities to each class.

2.Crop Disease Detection

- The CNN model analyzes plant leaf images using convolutional layers to detect patterns associated with healthy and infected conditions.
- The final layer uses a sigmoid activation function to classify the image as either "Healthy" or "Infected" based on probability scores.

WORKING PRINCIPLE

Identification & Prediction Systems

3. Weed Identification

- The CNN model extracts features from images using convolutional layers to differentiate between "Weed" and "No Weed" based on learned patterns.
- The final layer uses a sigmoid activation function to classify the image, assigning a probability to determine whether it contains weeds.

4. Medicinal Plant Identification

- The CNN model extracts features from medicinal plant images using convolutional layers, learning patterns specific to each plant type.
- The model processes the input image and predicts the plant type by selecting the class with the highest probability.

EXPERIMENTAL SETUP

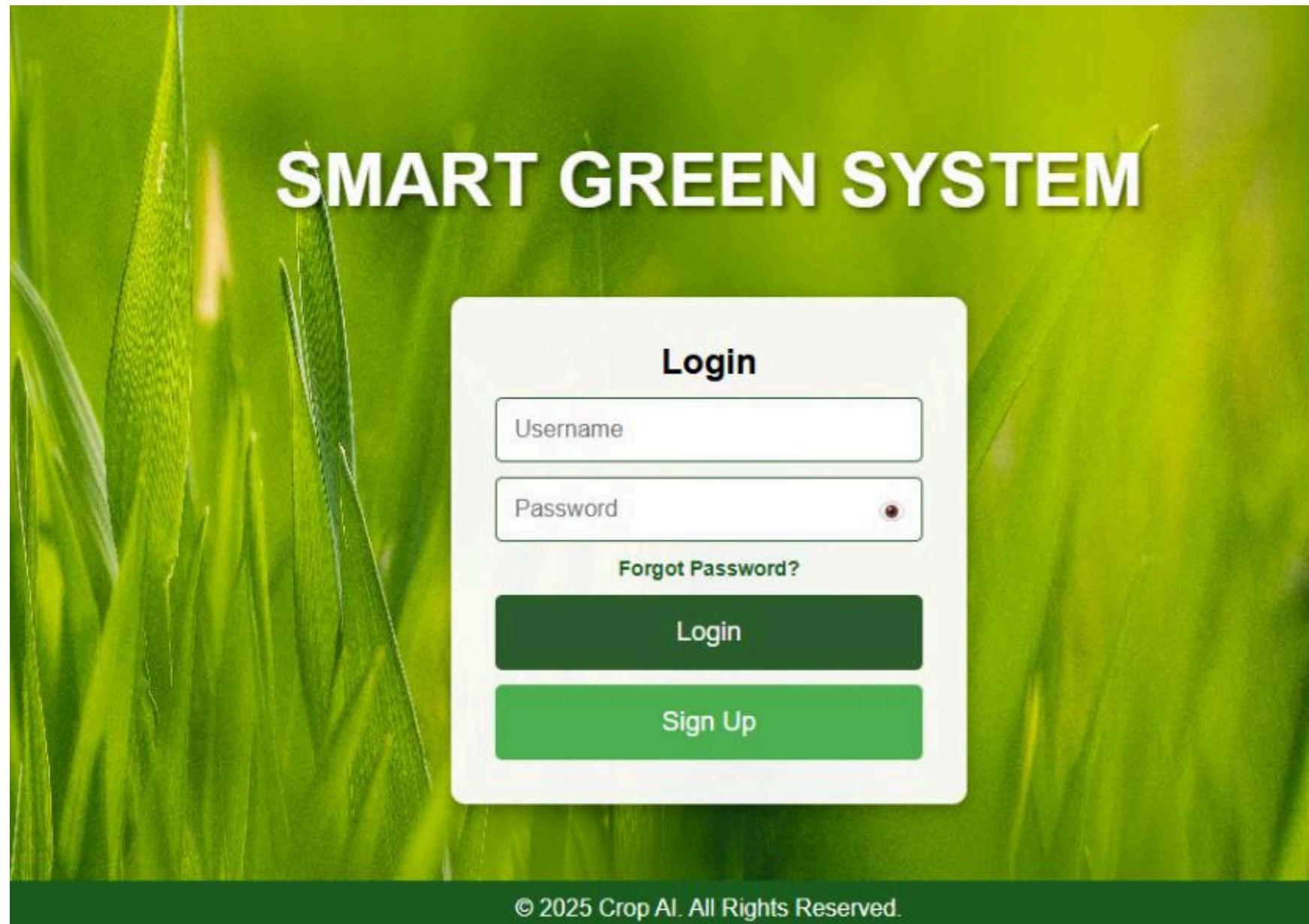
Developing Environment

- Operating System: Windows 10 / Linux (Ubuntu 20.04 recommended for TensorFlow performance)
- Libraries: TensorFlow/Keras, NumPy, Pandas, Matplotlib
- Database: SQLite (to store results and plant data)
- Browser: Google Chrome / Mozilla Firefox (for the web-based interface)
- Development Tools: Python (3.9+), Jupyter Notebook, Flask for the front end

Hardware Setup

- Processor: Intel Core i5/i7 or AMD Ryzen 5/7 (Quad-core or higher)
- RAM: Minimum 8GB (16GB recommended for smoother model training)
- GPU: NVIDIA GTX 1650 or higher (for TensorFlow GPU acceleration)
- Storage: SSD with at least 256GB (faster data processing)
- Camera: 8MP or higher (for capturing plant images)

EXPERIMENTAL SETUP



A screenshot of the Smart Green System home page. The background features a dense field of green corn plants. Overlaid on the image is the text "REVOLUTIONIZING AGRICULTURE WITH AI" in large, white, bold letters. Below the main image is a light green header section containing the "ABOUT US" heading and a brief project description. The main content area is divided into four white rectangular boxes, each representing a different AI service: "PROFITABLE CROP PREDICTION" (with a "Enter Data" button), "MEDICINAL PLANT IDENTIFICATION" (with a "Choose File" button), "CROP DISEASE DETECTION" (with a "Choose File" button), and "WEED MANAGEMENT" (with a "Choose File" button). At the bottom of the page, a dark green footer bar contains the copyright text "© 2025 Crop AI. All Rights Reserved.".

EXPERIMENTAL SETUP

PROFITABLE CROP PREDICTION

ENTER ENVIRONMENTAL DATA

State

City

Rainfall (mm)

Nitrogen (N) Level

Phosphorous (P) Level

pH Level

Potassium (K) Level

Predict Crop

© 2025 Smart Green System. All Rights Reserved.

CROP DISEASE DETECTION

UPLOAD CROP IMAGE

Select or drag & drop an image to detect diseases in crops.

Click Here or Drag & Drop Image
(Supported formats: JPG, PNG, JPEG)

Detect Disease

© 2025 Smart Green System. All Rights Reserved.

EXPERIMENTAL SETUP

WEED MANAGEMENT

UPLOAD WEED IMAGE

Select or drag & drop an image to detect and manage weeds.

Click Here or Drag & Drop Image
(Supported formats: JPG, PNG, JPEG)

Identify Weed

© 2025 Smart Green System. All Rights Reserved.

MEDICINAL PLANT DETECTION

UPLOAD PLANT IMAGE

Select or drag & drop an image to identify medicinal plants.

Click Here or Drag & Drop Image
(Supported formats: JPG, PNG, JPEG)

Identify Plant

© 2025 Smart Green System. All Rights Reserved.

The screenshot shows a Microsoft Visual Studio Code (VS Code) interface. The main area displays a Python script named `crop_prediction.py`. The code performs data preprocessing, takes user inputs for soil properties and yield, defines a neural network model, compiles it, and trains the model on a dataset. The script concludes with a success message and ends with a warning about SSL support.

```
crop_prediction.py 2 X
crop_prediction.py > ...
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
# Split Dataset (80% Train, 20% Test)
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y_encoded, test_size=0.2, random_state=42)
print("Data Preprocessing Complete.")
# User Inputs Data FIRST
soil_pH = float(input("Enter Soil pH: "))
organic_matter = float(input("Enter Organic Matter (%): "))
nitrogen = float(input("Enter Nitrogen Content (mg/kg): "))
phosphorus = float(input("Enter Phosphorus Content (mg/kg): "))
potassium = float(input("Enter Potassium Content (mg/kg): "))
rainfall = float(input("Enter Rainfall (mm): "))
temperature = float(input("Enter Temperature (°C): "))
yield_tons = float(input("Enter Yield (Tons per Hectare): "))
# Define Crop Prediction Model
model = Sequential([
    Dense(64, activation='relu', input_shape=(X_train.shape[1],)),
    Dropout(0.3),
    Dense(32, activation='relu'),
    Dense(len(np.unique(y_encoded)), activation='softmax') # Classification output
])
# Compile and Train Model
model.compile(optimizer='adam', loss='sparse_categorical_crossentropy', metrics=['accuracy'])
history = model.fit(X_train, y_train, validation_data=(X_test, y_test), epochs=50, batch_size=16, verbose=0)
print("Crop Prediction Model Training Complete.")

PROBLEMS 2 OUTPUT DEBUG CONSOLE TERMINAL PORTS + - X
● (venv) ameenali@Ameens-MacBook-Air Smart % /usr/bin/python3 /Users/ameenali/.vscode/extensions/ms-python.python-2025.2.0-darwin-arm64/python_files/printEnvVariablesToFile.py /Users/ameenali/.vscode/extensions/ms-python.python-2025.2.0-darwin-arm64/python_files/deactivate/zsh/envVars.txt
● (venv) ameenali@Ameens-MacBook-Air Smart % /usr/bin/python3 /Users/ameenali/.vscode/extensions/ms-python.python-2025.2.0-darwin-arm64/python_files/printEnvVariablesToFile.py /Users/ameenali/.vscode/extensions/ms-python.python-2025.2.0-darwin-arm64/python_files/deactivate/zsh/envVars.txt
● (venv) ameenali@Ameens-MacBook-Air Smart % python3 crop_prediction.py
/Users/ameenali/Desktop/Smart /venv/lib/python3.9/site-packages/urllib3/_init_.py:35: NotOpenSSLError: urllib3 v2 only supports OpenSSL 1.1.1+, currently the 'ssl' module is compiled with 'LibreSSL 2.8.3'. See: https://github.com/urllib3/urllib3/issues/3020
  warnings.warn(
Data Preprocessing Complete.
Enter Soil pH: ^[2
Traceback (most recent call last):
  File "/Users/ameenali/Desktop/Smart /crop_prediction.py", line 35, in <module>
    soil_pH = float(input("Enter Soil pH: "))
ValueError: could not convert string to float: '\x1b2'
(venv) ameenali@Ameens-MacBook-Air Smart %
```

The screenshot shows a development environment in VS Code with the following details:

- EXPLORER**: Shows the project structure with files like `crop_prediction.py`, `app.py`, `crop_final_5.csv`, and `1.html`.
- OPEN EDITORS**: Displays multiple editor tabs: `crop_prediction.py`, `app.py`, `crop_final_5.csv`, and `1.html`.
- SMART**: Shows the virtual environment setup with `venv` and its dependencies.
- PROBLEMS**: Shows 2 errors, related to `crop_prediction.py`.
- OUTPUT**: Shows the terminal output of a Flask development server running on port 5000.
- DEBUG CONSOLE**: Shows the Python debugger interface.
- TERMINAL**: Active tab, showing the command to run the application and logs of the server's activity.
- PORTS**: Shows open ports 5000 and 5001.
- RIGHT SIDE BAR**: Includes a preview pane showing the content of `1.html` and a sidebar with icons for file operations.

```
crop_prediction.py 2 app.py X crop_final_5.csv 1.html
app.py > predict
1   from flask import Flask, render_template, request, jsonify
2
3   app = Flask(__name__)
4
5   @app.route("/")
6   def home():
7       return render_template("1.html") # Ensure this is your correct template
8
9   @app.route("/predict", methods=["POST"])
10  def predict():
11      try:
12          # Get form data
13          state = request.form.get("state")
14          city = request.form.get("city")
15          rainfall = request.form.get("rainfall")
16          nitrogen = request.form.get("nitrogen")
17          phosphorous = request.form.get("phosphorous")
18          ph = request.form.get("ph")
19          potassium = request.form.get("potassium")
20
21          # Validate form fields
22          if not all([state, city, rainfall, nitrogen, phosphorous, ph, potassium]):
23              return jsonify({"error": "Missing required fields"}), 400
24
25          # Placeholder prediction logic (replace this with your ML model)
26          prediction = "Wheat" # Example output
27
28          return render_template("1.html", prediction=f"Recommended Crop: {prediction}")
29
30      except Exception as e:
31          print(e)
```

PROBLEMS 2 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
(venv) ameenali@Ameens-MacBook-Air Smart % "/Users/ameenali/Desktop/Smart /venv/bin/python" "/Users/ameenali/Desktop/Smart /app.py"
* Serving Flask app 'app'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
* Restarting with stat
* Debugger is active!
* Debugger PIN: 113-988-390
127.0.0.1 - - [21/Mar/2025 12:14:44] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [21/Mar/2025 12:14:44] "GET /static/plant.jpg HTTP/1.1" 404 -
127.0.0.1 - - [21/Mar/2025 12:14:54] "POST /predict HTTP/1.1" 200 -
127.0.0.1 - - [21/Mar/2025 12:18:06] "POST /predict HTTP/1.1" 200 -
```

Ln 15, Col 48 Spaces: 4 UTF-8 LF () Python 3.9.6 ('venv')

```
crop_prediction.py 2 Medicinal.py 3
Medicinal.py > ...
9  from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
10 import matplotlib.pyplot as plt
11
12 # Define Dataset Path
13 dataset_path = "/Users/ameenali/Desktop/Smart /medicinal_plant1" # Ensure this contains subfolders 'AloeVera', 'Tulsi', 'Neem'
14
15 # Image Parameters
16 img_size = (150, 150)
17 batch_size = 32
18
19 # Data Preprocessing
20 datagen = ImageDataGenerator(rescale=1./255, validation_split=0.2)
21
22 train_data = datagen.flow_from_directory(dataset_path, target_size=img_size, batch_size=batch_size, class_mode='categorical', subset='training')
23 val_data = datagen.flow_from_directory(dataset_path, target_size=img_size, batch_size=batch_size, class_mode='categorical', subset='validation')
24
25 # CNN Model Architecture
26 model = Sequential([
27     Conv2D(32, (3, 3), activation='relu', input_shape=(150, 150, 3)),
28     MaxPooling2D(2, 2),
29     Conv2D(64, (3, 3), activation='relu'),
30     MaxPooling2D(2, 2),
31     Conv2D(128, (3, 3), activation='relu'),
32     MaxPooling2D(2, 2),
33     Flatten(),
34     Dense(128, activation='relu'),
35     Dropout(0.5),
36     Dense(3, activation='softmax') # 3 classes: Aloe Vera, Tulsi, Neem
37 ])
38
```

PROBLEMS 5 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
7/7 1s 194ms/step - accuracy: 0.8328 - loss: 0.3762 - val_accuracy: 0.8077 - val_loss: 0.4907
2/2 0s 36ms/step - accuracy: 0.7989 - loss: 0.5161

Validation Accuracy: 80.77%
WARNING:absl:You are saving your model as an HDF5 file via `model.save()` or `keras.saving.save_model(model)`. This file format is considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.keras')` or `keras.saving.save_model(model, 'my_model.keras')`.

Model Saved Successfully.
```

Enter the path of the image: /Users/ameenali/Desktop/Smart /medicinal_plant1/Aloe Vera/859.jpg

```
1/1 0s 50ms/step
2025-03-19 12:19:49.768 Python[13209:991376] +[IMKClient subclass]: chose IMKClient_Modern
2025-03-19 12:19:49.768 Python[13209:991376] +[IMKInputSession subclass]: chose IMKInputSession_Modern
```

Predicted Plant: Aloe Vera

(venv) ameenali@Ameens-MacBook-Air Smart %

The screenshot shows a code editor window for a Python file named `crop_prediction.py`. The code implements a machine learning model to predict crop yield based on environmental inputs. It includes data preprocessing, user input for features, model definition (using Sequential API), compilation, and training.

```
crop_prediction.py 2 X
crop_prediction.py > ...
scaler = StandardScaler()
28 X_scaled = scaler.fit_transform(X)
29
# • Split Dataset (80% Train, 20% Test)
30 X_train, X_test, y_train, y_test = train_test_split(X_scaled, y_encoded, test_size=0.2, random_state=42)
31 print("✅ Data Preprocessing Complete.")
32
# • User Inputs Data FIRST
33 soil_pH = float(input("Enter Soil pH: "))
34 organic_matter = float(input("Enter Organic Matter (%): "))
35 nitrogen = float(input("Enter Nitrogen Content (mg/kg): "))
36 phosphorus = float(input("Enter Phosphorus Content (mg/kg): "))
37 potassium = float(input("Enter Potassium Content (mg/kg): "))
38 rainfall = float(input("Enter Rainfall (mm): "))
39 temperature = float(input("Enter Temperature (°C): "))
40 yield_tons = float(input("Enter Yield (Tons per Hectare): "))
41
# • Define Crop Prediction Model
42 model = Sequential([
43     Dense(64, activation='relu', input_shape=(X_train.shape[1],)),
44     Dropout(0.3),
45     Dense(32, activation='relu'),
46     Dense(len(np.unique(y_encoded)), activation='softmax') # Classification output
47 ])
48
# • Compile and Train Model
49 model.compile(optimizer='adam', loss='sparse_categorical_crossentropy', metrics=['accuracy'])
50 history = model.fit(X_train, y_train, validation_data=(X_test, y_test), epochs=50, batch_size=16, verbose=0)
51 print("✅ Crop Prediction Model Training Complete.")
52
53
54
55
56
```

The bottom panel displays the terminal output, showing the execution of the script and its results. A warning about SSL support is visible, followed by the success message and user input for soil pH.

```
PROBLEMS 2 OUTPUT DEBUG CONSOLE TERMINAL PORTS + ✎ ... ⌂ X
● (venv) ameenali@Ameens-MacBook-Air Smart % /usr/bin/python3 /Users/ameenali/.vscode/extensions/ms-python.python-2025.2.0-darwin-arm64/python_files/printEnvVariablesToFile.py /Users/ameenali/.vscode/extensions/ms-python.python-2025.2.0-darwin-arm64/python_files/deactivate/zsh/envVars.txt
● (venv) ameenali@Ameens-MacBook-Air Smart % /usr/bin/python3 /Users/ameenali/.vscode/extensions/ms-python.python-2025.2.0-darwin-arm64/python_files/printEnvVariablesToFile.py /Users/ameenali/.vscode/extensions/ms-python.python-2025.2.0-darwin-arm64/python_files/deactivate/zsh/envVars.txt
● (venv) ameenali@Ameens-MacBook-Air Smart % python3 crop_prediction.py
/Users/ameenali/Desktop/Smart /venv/lib/python3.9/site-packages/urllib3/__init__.py:35: NotOpenSSLWarning: urllib3 v2 only supports OpenSSL 1.1.1+, currently the 'ssl' module is compiled with 'LibreSSL 2.8.3'. See: https://github.com/urllib3/urllib3/issues/3020
  warnings.warn(
    ✅ Data Preprocessing Complete.
Enter Soil pH: ^[2
Traceback (most recent call last):
  File "/Users/ameenali/Desktop/Smart /crop_prediction.py", line 35, in <module>
    soil_pH = float(input("Enter Soil pH: "))
ValueError: could not convert string to float: '\x1b[2
(venv) ameenali@Ameens-MacBook-Air Smart %
```

On the right side, there are several open terminal tabs labeled "Python", "zsh", and "zsh".

```
plantdisease.py 5 ×

Users > ameenali > Desktop > Smart > plantdisease.py > ...
19     datagen = ImageDataGenerator(rescale=1./255, validation_split=0.2)
20
21     train_data = datagen.flow_from_directory(
22         dataset_path, target_size=img_size, batch_size=batch_size, class_mode='binary', subset='training')
23
24     val_data = datagen.flow_from_directory(
25         dataset_path, target_size=img_size, batch_size=batch_size, class_mode='binary', subset='validation')
26
27 # CNN Model Architecture
28 model = Sequential([
29     Conv2D(32, (3, 3), activation='relu', input_shape=(150, 150, 3)),
30     MaxPooling2D(2, 2),
31     Conv2D(64, (3, 3), activation='relu'),
32     MaxPooling2D(2, 2),
33     Conv2D(128, (3, 3), activation='relu'),
34     MaxPooling2D(2, 2),
35     Flatten(),
36     Dense(128, activation='relu'),
37     Dropout(0.5),
38     Dense(1, activation='sigmoid') # Binary classification (Healthy vs. Infected)
39 ])
40
41 # Compile Model
42 model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
43
44 # Train Model
45 history = model.fit(train_data, validation_data=val_data, epochs=10)
46
47 # Evaluate Model
48 val_loss, val_acc = model.evaluate(val_data)
```

PROBLEMS 5 OUTPUT DEBUG CONSOLE TERMINAL PORTS

Python + ×

```
2025-03-19 13:53:13.669 Python[15512:1073521] +[IMKClient subclass]: chose IMKClient_Modern
2025-03-19 13:53:13.669 Python[15512:1073521] +[IMKInputSession subclass]: chose IMKInputSession_Modern
```

Enter the image filename (e.g., healthy1.jpg or infected2.jpg): jhf

```
✖ Error: File not found. Please enter a correct filename.
● ameenali@Ameens-MacBook-Air ~ % /usr/bin/python3 "/Users/ameenali/Desktop/Smart /plantdisease.py"
/Users/ameenali/Library/Python/3.9/lib/python/site-packages/urllib3/_init_.py:35: NotOpenSSLWarning: urllib3 v2 only supports OpenSSL 1.1.1+, currently the 'ssl' module is compiled with 'LibreSSL
2.8.3'. See: https://github.com/urllib3/urllib3/issues/3020
    warnings.warn(
Traceback (most recent call last):
  File "/Users/ameenali/Desktop/Smart /plantdisease.py", line 4, in <module>
    import matplotlib.pyplot as plt
ModuleNotFoundError: No module named 'matplotlib'
ameenali@Ameens-MacBook-Air ~ %
```

```
crop_prediction.py 2 Medicinal.py 4 Weed.py 3 x
crop_prediction.py
12
13     # * Image Parameters
14     img_size = (150, 150)
15     batch_size = 32
16
17     # * Data Preprocessing
18     datagen = ImageDataGenerator(rescale=1./255, validation_split=0.2)
19
20     train_data = datagen.flow_from_directory(dataset_path, target_size=img_size, batch_size=batch_size,
21                                              class_mode='binary', subset='training')
22     val_data = datagen.flow_from_directory(dataset_path, target_size=img_size, batch_size=batch_size,
23                                              class_mode='binary', subset='validation')
24
25     # * CNN Model Architecture
26     model = Sequential([
27         Conv2D(32, (3, 3), activation='relu', input_shape=(150, 150, 3)),
28         MaxPooling2D(2, 2),
29         Conv2D(64, (3, 3), activation='relu'),
30         MaxPooling2D(2, 2),
31         Flatten(),
32         Dense(128, activation='relu'),
33         Dropout(0.5),
34         Dense(1, activation='sigmoid') # Binary classification: Weed (1) / No Weed (0)
35     ])
36
37     # * Compile Model
38     model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
39
40     # * Train Model
41     history = model.fit(train_data, validation_data=val_data, epochs=10)
```

PROBLEMS 9 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
/Users/ameenali/Desktop/Smart /venv/bin/python" "/Users/ameenali/Desktop/Smart /Weed.py"
(venv) ameenali@Ameens-MacBook-Air Smart % "/Users/ameenali/Desktop/Smart /venv/bin/python" "/Users/ameenali/Desktop/Smart /Weed.py"
/Users/ameenali/Desktop/Smart /venv/lib/python3.9/site-packages/urllib3/_init_.py:35: NotOpenSSLWarning: urllib3 v2 only supports OpenSSL 1.1.1+, currently the 'ssl' module is compiled with 'LibreSSL 2.8.3'. See: https://github.com/urllib3/urllib3/issues/3020
  warnings.warn(
Found 209 images belonging to 2 classes.
Found 51 images belonging to 2 classes.
/Users/ameenali/Desktop/Smart /venv/lib/python3.9/site-packages/keras/src/layers/convolutional/base_conv.py:107: UserWarning: Do not pass an `input_shape`/`input_dim` argument to a layer. When using Sequential models, prefer using an `Input(shape)` object as the first layer in the model instead.
  super().__init__(activity_regularizer=activity_regularizer, **kwargs)
/Users/ameenali/Desktop/Smart /venv/lib/python3.9/site-packages/keras/src/trainers/data_adapters/py_dataset_adapter.py:121: UserWarning: Your `PyDataset` class should call `super().__init__(**kwargs)` in its constructor. `**kwargs` can include `workers`, `use_multiprocessing`, `max_queue_size`. Do not pass these arguments to `fit()`, as they will be ignored.
  self._warn_if_super_not_called()
Epoch 1/10
```

Python
zsh
Python

The screenshot shows a dark-themed interface of the Visual Studio Code (VS Code) code editor. The main area displays a Python script named `crop_prediction.py`. The script performs data preprocessing, user input for soil properties, defines a neural network model, and trains it on a dataset. The code uses standard libraries like `StandardScaler` from `sklearn.preprocessing`, `Sequential` and `Dense` from `keras.models`, and `Adam` from `keras.optimizers`.

```
crop_prediction.py
1  # Import Libraries
2  import numpy as np
3  import pandas as pd
4  from sklearn.preprocessing import StandardScaler
5  from sklearn.model_selection import train_test_split
6  from tensorflow.keras.models import Sequential
7  from tensorflow.keras.layers import Dense
8
9  # Read CSV File
10 df = pd.read_csv("Crop_recommendation.csv")
11
12 # Preprocess Data
13 df.fillna(0, inplace=True)
14 X = df.drop("label", axis=1)
15 y = df["label"]
16
17 # Scale Data
18 scaler = StandardScaler()
19 X_scaled = scaler.fit_transform(X)
20
21 # Split Dataset (80% Train, 20% Test)
22 X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2, random_state=42)
23 print("Data Preprocessing Complete.")
24
25 # User Inputs Data FIRST
26 soil_pH = float(input("Enter Soil pH: "))
27 organic_matter = float(input("Enter Organic Matter (%): "))
28 nitrogen = float(input("Enter Nitrogen Content (mg/kg): "))
29 phosphorus = float(input("Enter Phosphorus Content (mg/kg): "))
30 potassium = float(input("Enter Potassium Content (mg/kg): "))
31 rainfall = float(input("Enter Rainfall (mm): "))
32 temperature = float(input("Enter Temperature (°C): "))
33 yield_tons = float(input("Enter Yield (Tons per Hectare): "))
34
35 # Define Crop Prediction Model
36 model = Sequential([
37     Dense(64, activation='relu', input_shape=(X_train.shape[1],)),
38     Dropout(0.3),
39     Dense(32, activation='relu'),
40     Dense(len(np.unique(y)), activation='softmax') # Classification output
41 ])
42
43 # Compile and Train Model
44 model.compile(optimizer='adam', loss='sparse_categorical_crossentropy', metrics=['accuracy'])
45 history = model.fit(X_train, y_train, validation_data=(X_test, y_test), epochs=50, batch_size=16, verbose=0)
46 print("Crop Prediction Model Training Complete.")
47
```

Below the code editor, the terminal window shows the execution of the script. It prints environment variables, the command run, and the output of the script. The script asks for user input for soil pH, which is then converted to a float and printed.

Terminal Output:

```
(venv) ameenali@Ameens-MacBook-Air Smart % /usr/bin/python3 /Users/ameenali/.vscode/extensions/ms-python.python-2025.2.0-darwin-arm64/python_files/printEnvVariablesToFile.py /Users/ameenali/.vscode/extensions/ms-python.python-2025.2.0-darwin-arm64/python_files/deactivate/zsh/envVars.txt
(venv) ameenali@Ameens-MacBook-Air Smart % /usr/bin/python3 /Users/ameenali/.vscode/extensions/ms-python.python-2025.2.0-darwin-arm64/python_files/printEnvVariablesToFile.py /Users/ameenali/.vscode/extensions/ms-python.python-2025.2.0-darwin-arm64/python_files/deactivate/zsh/envVars.txt
(venv) ameenali@Ameens-MacBook-Air Smart % python3 crop_prediction.py

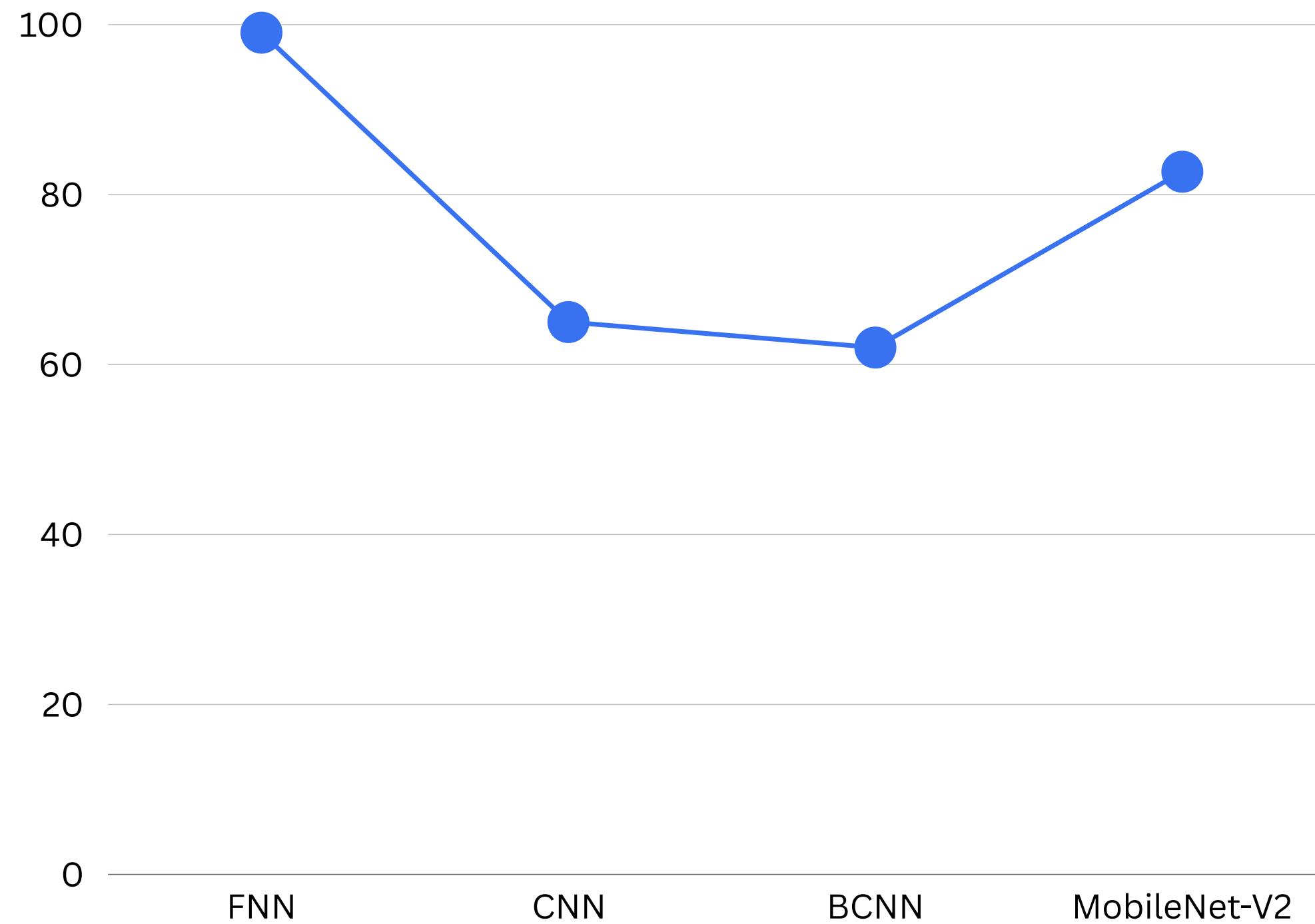
/Users/ameenali/Desktop/Smart /venv/lib/python3.9/site-packages/urllib3/__init__.py:35: NotOpenSSLWarning: urllib3 v2 only supports OpenSSL 1.1.1+, currently the 'ssl' module is compiled with 'LibreSSL 2.8.3'. See: https://github.com/urllib3/urllib3/issues/3020
  warnings.warn(
    ✓ Data Preprocessing Complete.
Enter Soil pH: ^[2
Traceback (most recent call last):
  File "/Users/ameenali/Desktop/Smart /crop_prediction.py", line 35, in <module>
    soil_pH = float(input("Enter Soil pH: "))
ValueError: could not convert string to float: '\x1b[2'
(venv) ameenali@Ameens-MacBook-Air Smart %
```

RESULTS & ANALYSIS

TOPIC	MODEL	ACCURACY(%)	PRECISION(%)	RECALL(%)	F1 SCORE (%)
Crop Prediction	fully connected feedforward neural network (FNN)	99.05	99.02	99.05	99.03
Medicinal Plant Identification	Convolutional Neural Network (CNN)	82.69	82	81	81
Disease Detection	Binary Classification CNN (Convolutional Neural Network)	62	60	63	60
Weed Detection	Transfer Learning CNN using MobileNetV2	65	64	61	63

RESULTS & ANALYSIS

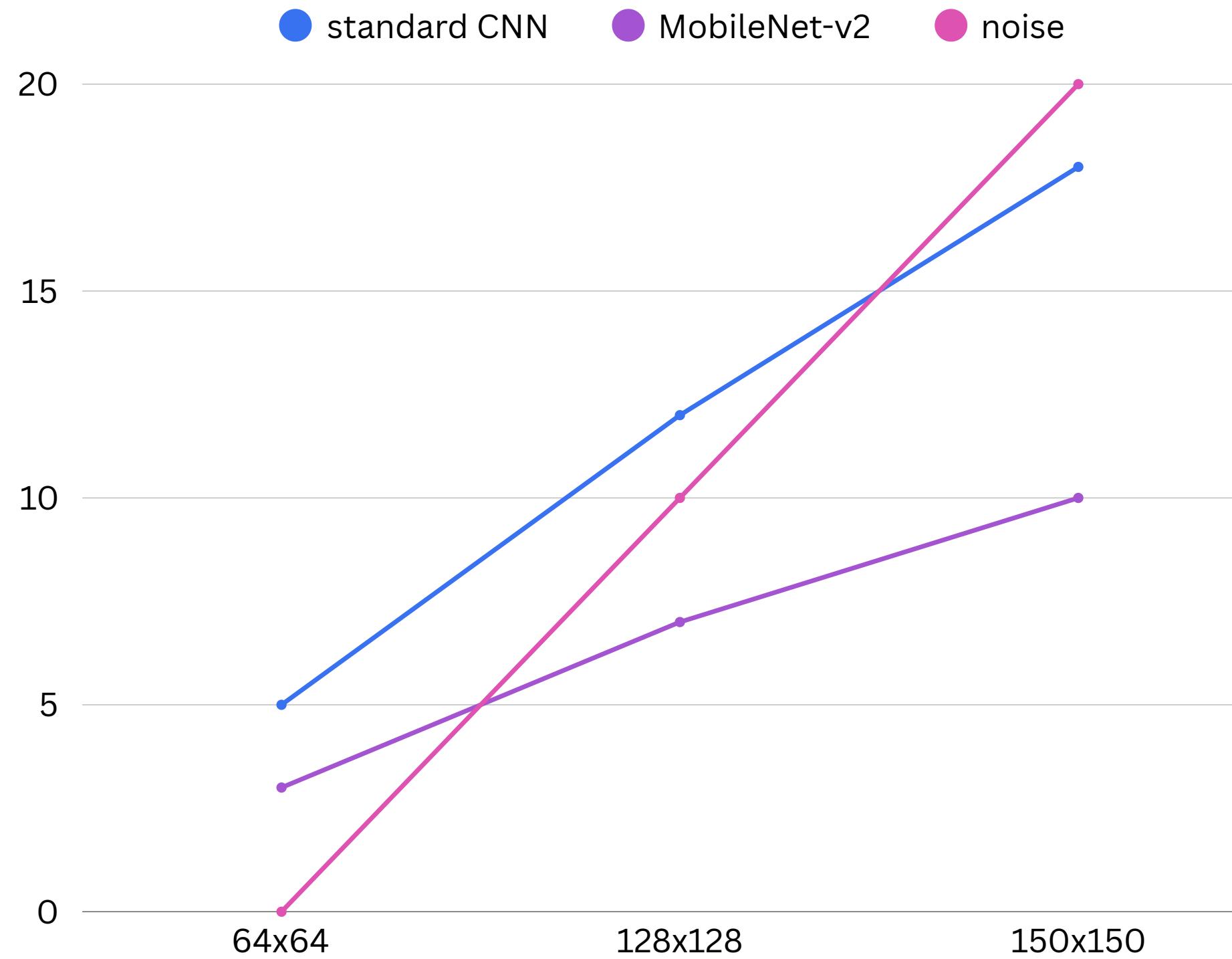
Accuracy comparision



- This graph shows the comparison of accuracy performance across different models. The FNN model achieves the highest accuracy, while CNN and BCNN have lower accuracy levels. MobileNetV2 performs better than CNN and BCNN, highlighting the advantage of transfer learning.

RESULTS & ANALYSIS

Processing Speed by Image Resolution & Noise (Image Resolution vs Processing Time Graph)



The graph compares Processing Speed by Image Resolution for Standard CNN (blue) vs. MobileNet-v2 (purple):

- X-axis: Image resolutions (64x64, 128x128, 150x150)
- Y-axis: Processing time (seconds)
- Noise increases processing time

RESULTS & ANALYSIS

Observation and Trends on the Result

1. Model Performance:

- The models achieve moderate to high accuracy (62%–99%) based on the dataset and architecture used.
- Transfer learning models (e.g., MobileNetV2) improve performance over traditional CNNs by extracting better features.

2. Potential Challenges:

- High-resolution images increase processing time, affecting real-time predictions.
- Class imbalance in datasets may lead to biased predictions, requiring data augmentation or rebalancing techniques.

RESULTS & ANALYSIS

Observation and Trends on the Result

3. User Interaction & Experience:

- The models provide real-time image classification with visual feedback through Matplotlib.
- Simple command-line input for image selection makes the system accessible but could be improved with a GUI for better usability.

4. Accuracy Trends:

- Accuracy generally improves with higher image resolution, but excessive resolution slows down processing.
- Transfer learning models consistently outperform basic CNNs, especially in low-data scenarios.

CONCLUSION

Key Findings

- CNN-based models effectively classify crops, plant diseases, and weeds with accuracy ranging from 62% to 99%.
- Higher image resolution enhances confidence but increases processing time.

Project Contributions

- Enhances user interaction through real-time image predictions and confidence scores.
- Implements automated weed detection and disease identification, aiding precision farming.

Limitations

- Processing time increases for high-resolution images, impacting real-time applications.
- Model accuracy depends on dataset quality, and imbalanced data can lead to biased predictions.

FUTURE SCOPE

1. Enhancing Dataset Diversity

- Expanding the training dataset with images from different regions, seasons, and plant growth stages can improve model generalization.

2. Integration with Augmented Reality (AR)

- AR-based plant identification can allow users to scan plants in real-time with detailed overlays of medicinal properties.

3. Mobile and Edge AI Optimization

- Implementing lightweight models (e.g., MobileNet, EfficientNet) for real-time identification on smartphones and low-power devices.

FUTURE SCOPE

4. Multilingual Support & Voice Assistance

- Adding multilingual support for plant information to make the system accessible to a broader audience.

5. IoT-Based Smart Farming Integration

- Using environmental sensors to enhance plant classification based on soil quality, humidity, and temperature.

6. Collaboration with Botanical Research Centers

- Partnering with research institutions to continuously improve the model and validate new medicinal plant species.

REFERENCES

1.Crop Prediction Model Using Machine Learning Algorithms

- [1] Elbasi E., [2] Zaki C., [3] Topcu A. E., [4] Abdelbaki W., [5] Zreikat A. I., [6] Cina E., [7] Shdefat A., [8] Saker L. (2023). Applied Sciences, 13, 9288. <https://www.mdpi.com/2076-3417/13/16/9288>

2.Deep Learning-Based Leaf Disease Detection in Crops Using Images for Agricultural Applications

- [1] Andrew J., [2] Eunice J., [3] Popescu D. E., [4] Chowdary M. K., [5] Hemanth J. (2022). Agronomy, 12, 2395. <https://www.mdpi.com/2073-4395/12/10/2395>

3.Weed Species Identification in Different Crops Using Precision Weed Management: A Review

- [1] Anand Muni Mishra, [2] Vinay Gautam. Weed Species Identification in Different Crops using Precision Weed Management: A Review.https://www.researchgate.net/publication/349463320_Weed_Species_Identification_in_Different_Crops_using_Precision_Weed_Management_A_Review

4.Medicinal Plant Identification in Real-Time Using Deep Learning Model

- [1] Kavitha S., [2] Kumar T. S., [3] Naresh E., [4] Kalmani V. H., [5] Bamane K. D., [6] Pareek P. K. (2024). SN Computer Science, 5, 73. <https://doi.org/10.1007/s42979-023-02398-5>

THANK YOU!

Any Queries?