

AGRI DOC

A Multifunctional
Mobile Application for
Enhancing Paddy
Farming Efficiency

GROUP NO R25-057





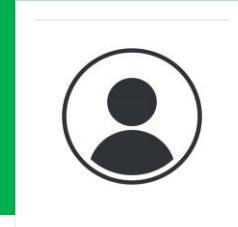
Members



Supervisor
Ms. Sanjeevi Chandrasiri



Co-Supervisor
Ms. Karthiga Rajendran



External Supervisor
T. Tharshanathan
Agrarian Development Officer,
Agraron Service Center,
Uppuveli,Trincomale.



U. Sutharson



A.Shivaphiriyan



Lavanya .M



V. Abilaxshan

Introduction to The Overall Project

How can an integrated digital platform address the challenges of irrigation management, weed identification, pest identification and control , and market analytics to improve the sustainability and productivity of paddy farming



Research Questions

1. How can AI-driven weed identification enhance precision agriculture and reduce herbicide dependency in paddy farming?
2. What measurable impact can cause pest identification & Effete
3. How does an IoT-based automated irrigation system impact paddy crop yield and water resource efficiency?
4. How can a user-friendly mobile platform improve accessibility and adoption of digital farming solutions among small-scale farmers?



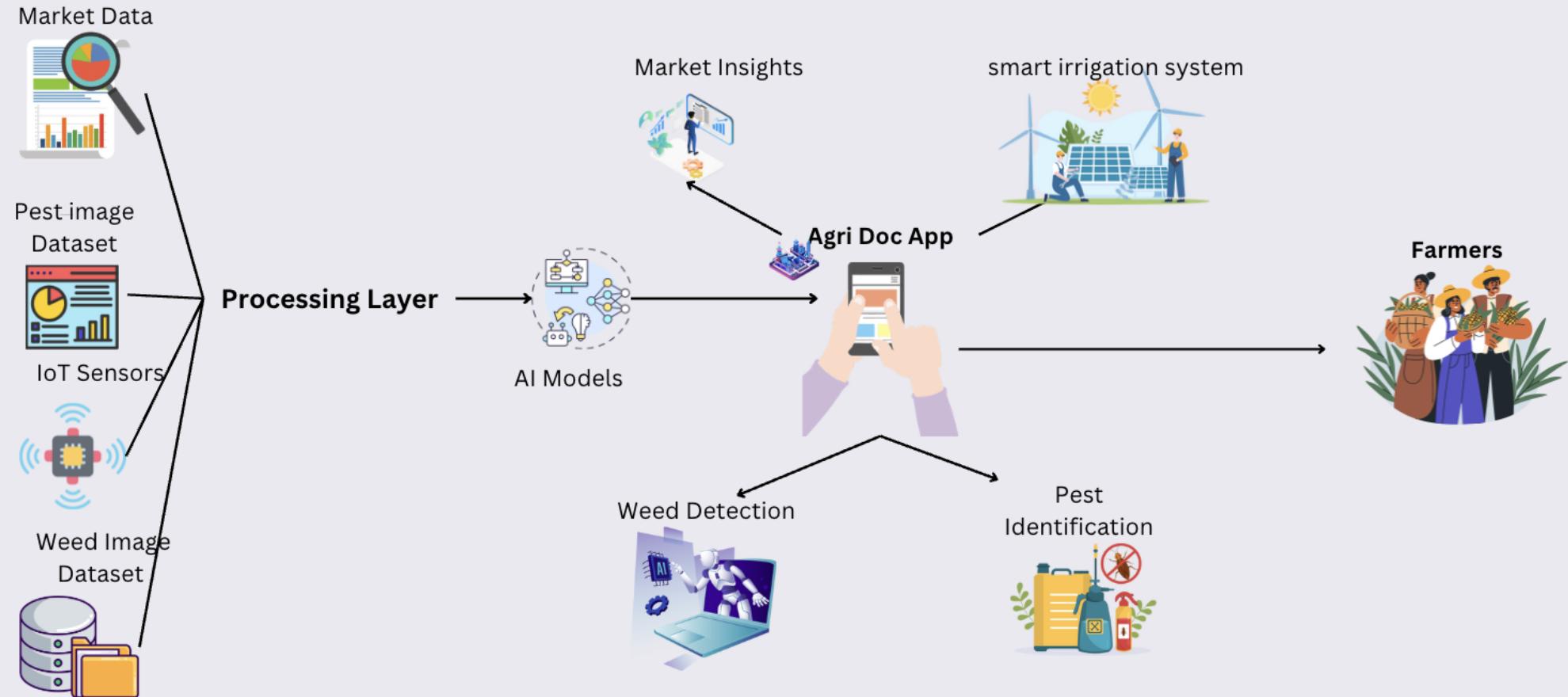
Research Objectives

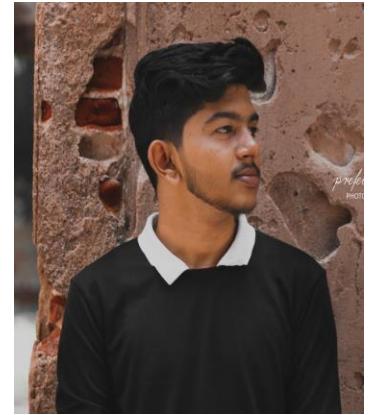


Develop *Agri Doc* app to empower paddy farmers with technology-driven solutions.

- Real-time irrigation monitoring using IoT sensors.
- AI-based weed identification for targeted control.
- Location-specific weather forecasts and alerts.
- Market insights with pricing and demand trends.

Diagram for Agri Doc





IT21829406 | U SUTHARSON

BSc(Hons) Information Technology Specializing in Information Technology



Market Data Analysis



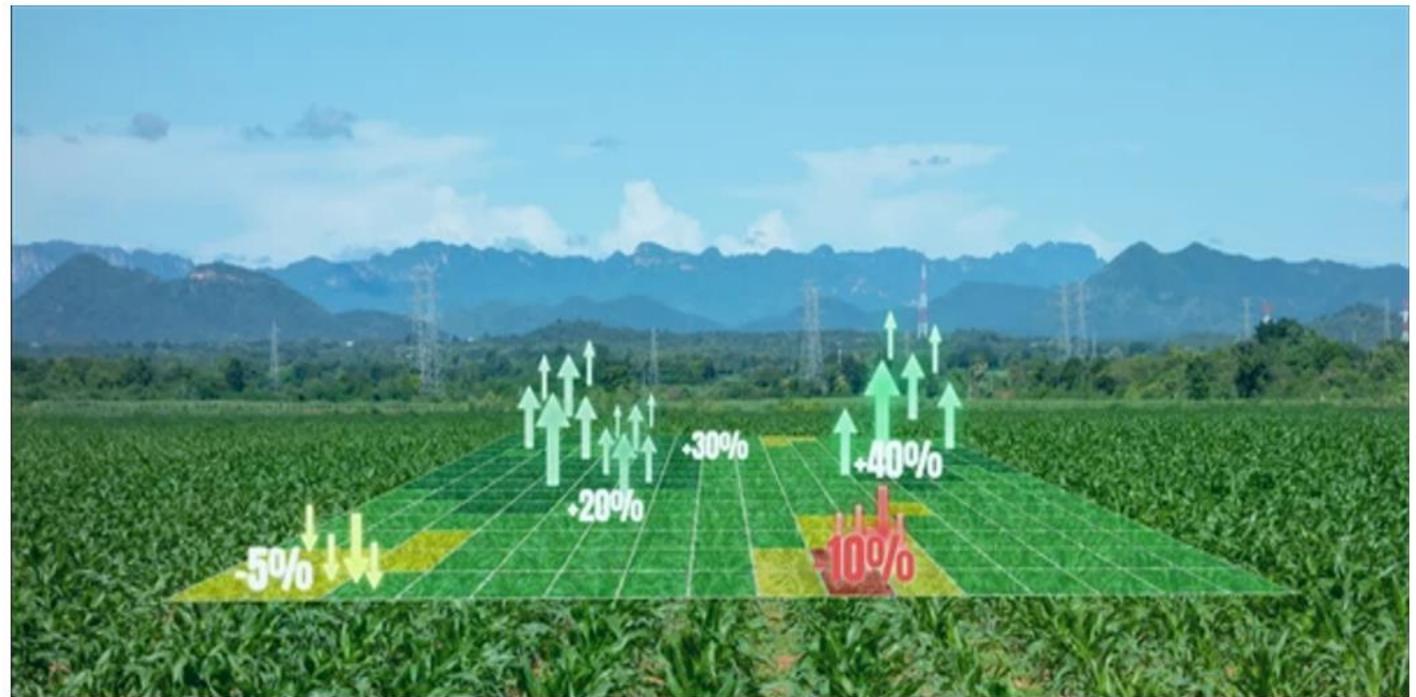
Introduction

Farmers often face challenges due to the lack of real-time market data, which leads to poor pricing decisions and financial losses. The market data analysis function addresses this issue by providing valuable insights into pricing trends and market demand. By leveraging data analytics, machine learning, and seamless market integration, this function equips farmers with accurate and timely information. These insights enable better decision-making, improve profitability, and contribute to enhanced financial stability for farmers.



Research Problem

- Farmers lack real-time, accurate market insights for pricing and sales decisions.
- This leads to economic instability and undervaluation of produce.
- Developing a predictable trend for price and demand can address these challenges effectively.



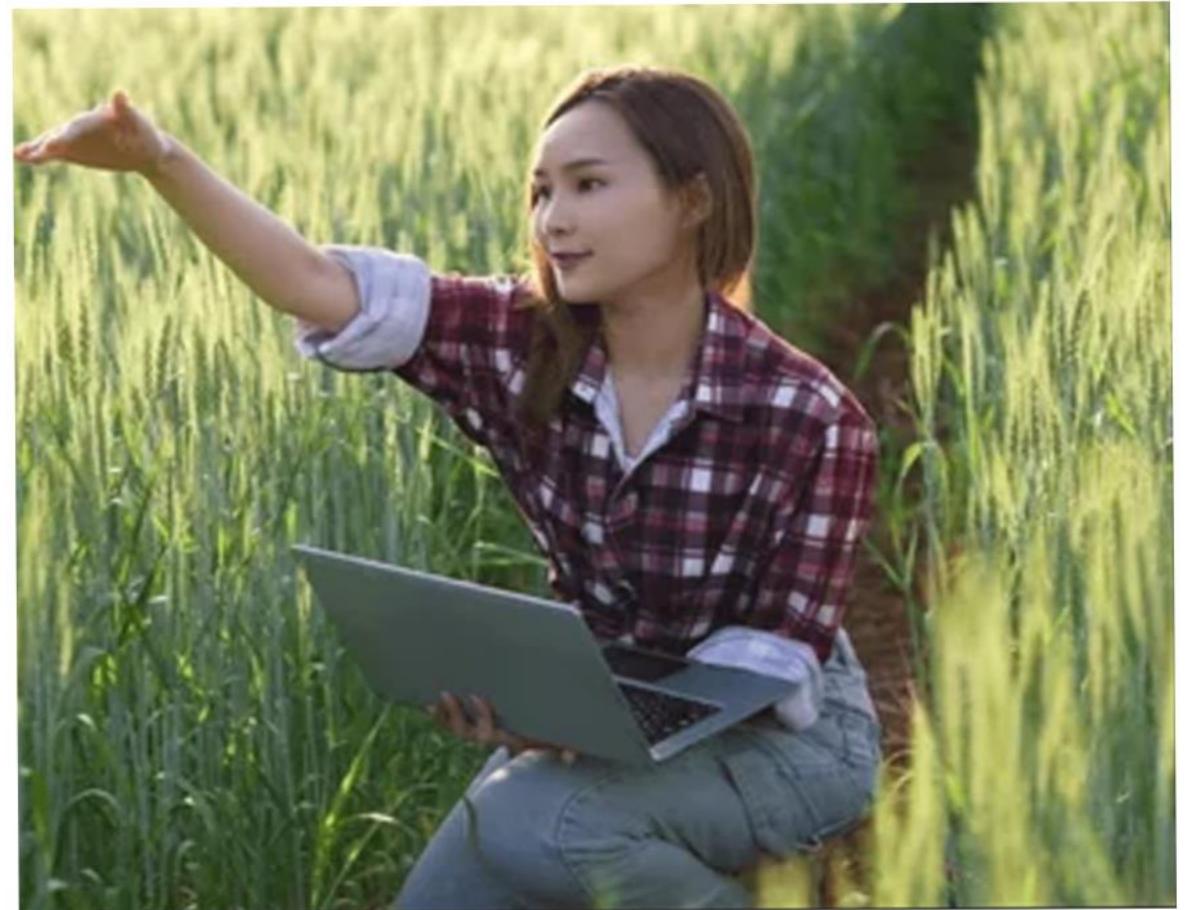
Research Gap



- Existing platforms focus primarily on crop cultivation, not market insights.
- No region-specific or predictive models tailored for paddy farmers.
- Previous apps and research have separate functions for prices or demands but lack a combined function integrating both price and demand.

Main Objective

To design a market data analysis module that equips farmers with actionable insights into pricing trends and optimal selling times.



Sub Objective

Collect and analyze

- Collect and analyze market data, including pricing and demand.

Develop

- Develop predictive models for future pricing trends and demand

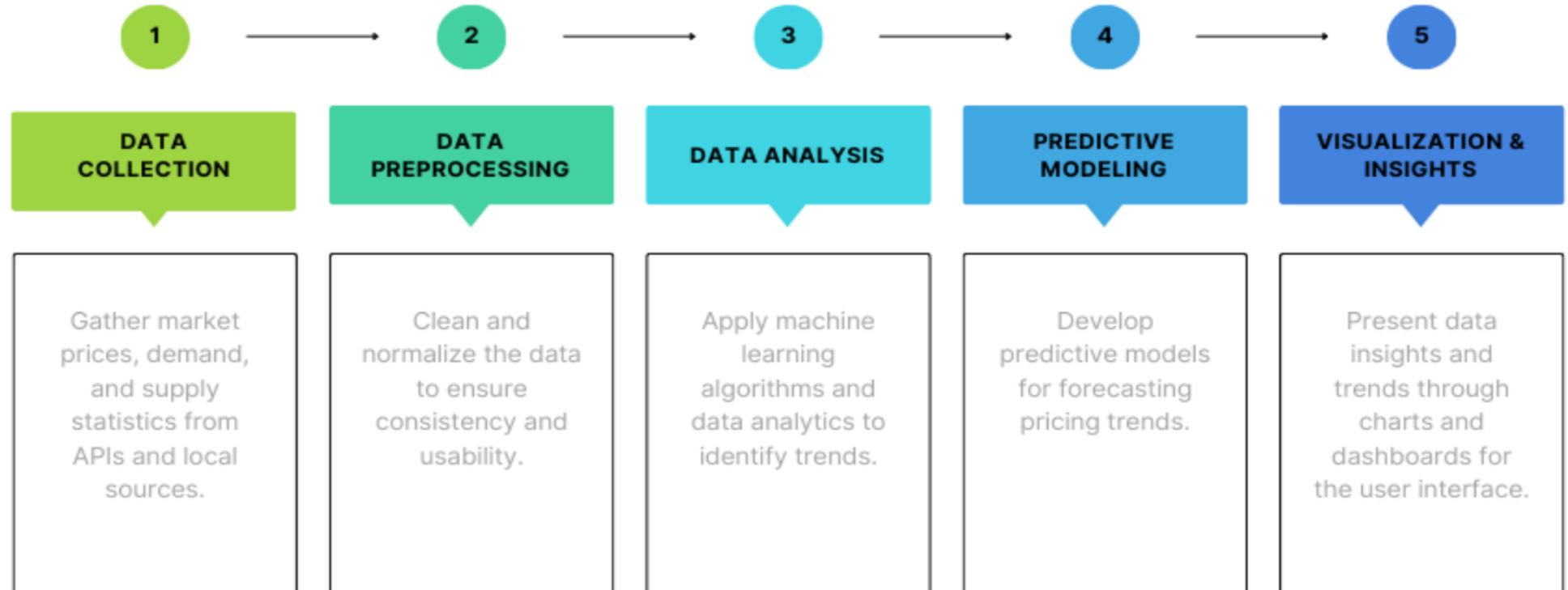
Provide

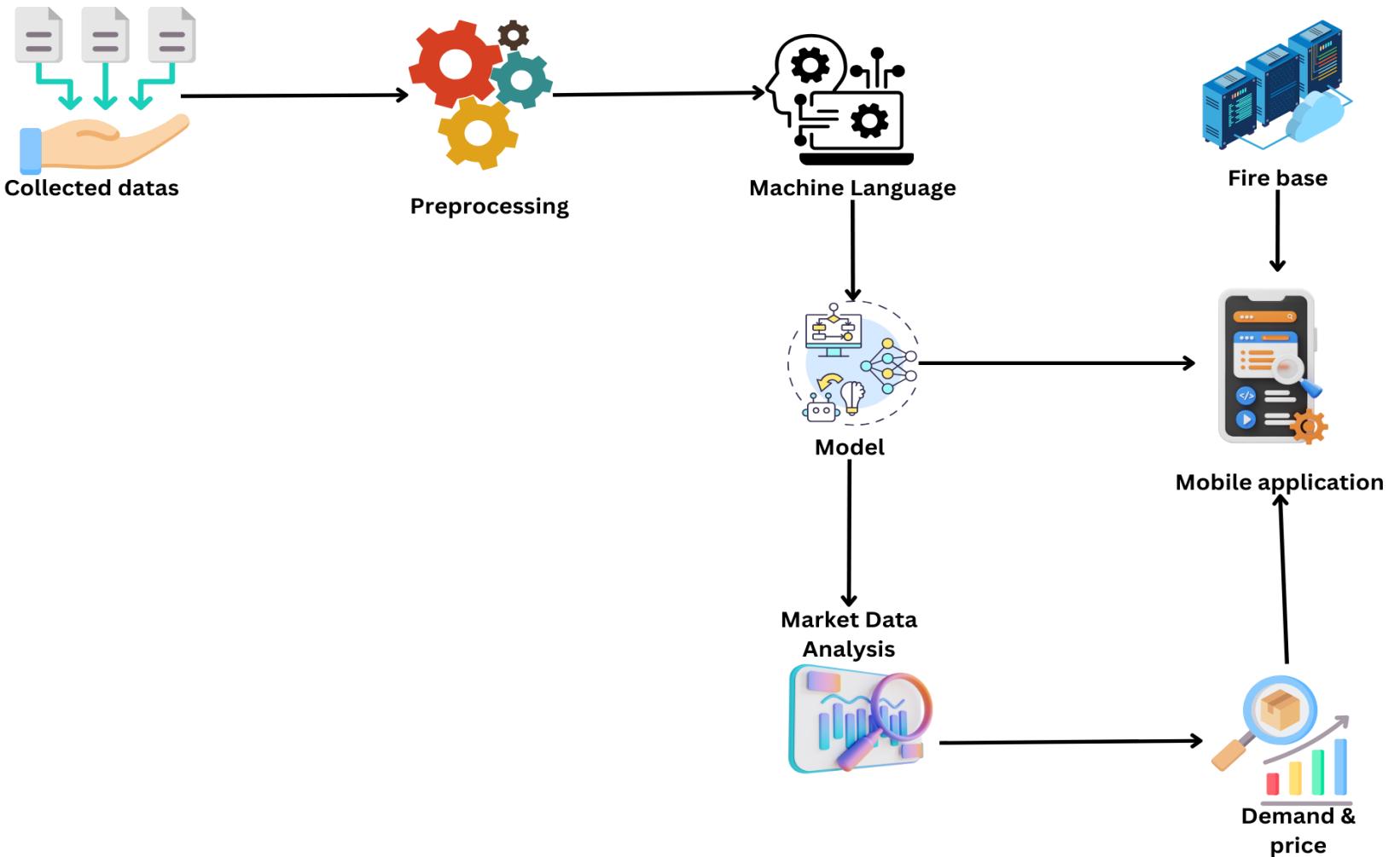
- Provide comparative insights into local and regional markets.

Enable

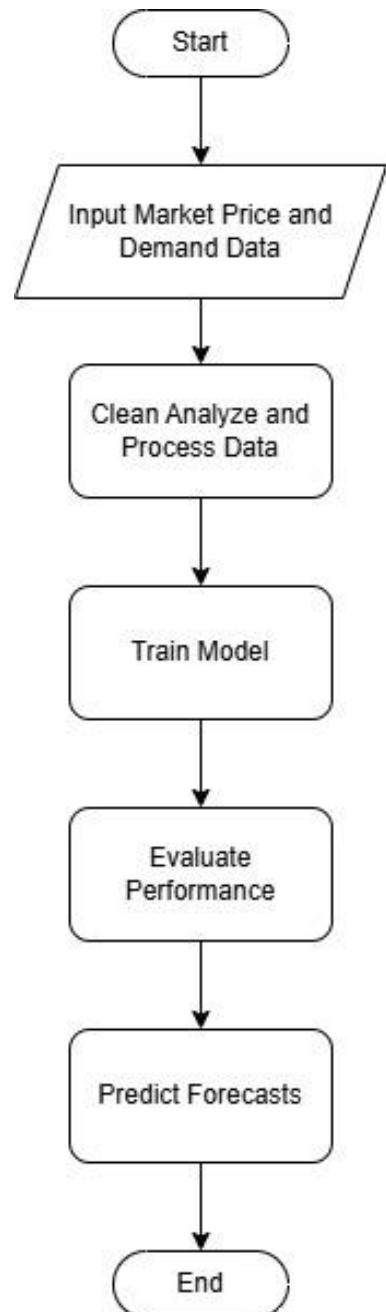
- Enable real-time tracking of market changes.

Methodology





Flow Chart Diagram



System Requirements



- **Hardware:**
Server: Intel Core i5, 8GB RAM, 500GB SSD, high-speed internet.
Mobile Devices: Android/iOS, 2GB RAM, 50MB storage.
- **Personal:**
System Administrator, Data Analyst, Software Developer.
- **Software:**
Development: Python, Dart, Firebase , Android Studio.
Machine Learning: Supervised Learning
Visualization: Flutter chart
API Integration: RESTful APIs for real-time data.

Technologies to Be Used



Programming Languages: Python (analytics), Dart (Flutter UI)



ML Frameworks: Supervised Learning



Database: Firebase



Dataset: Farmers



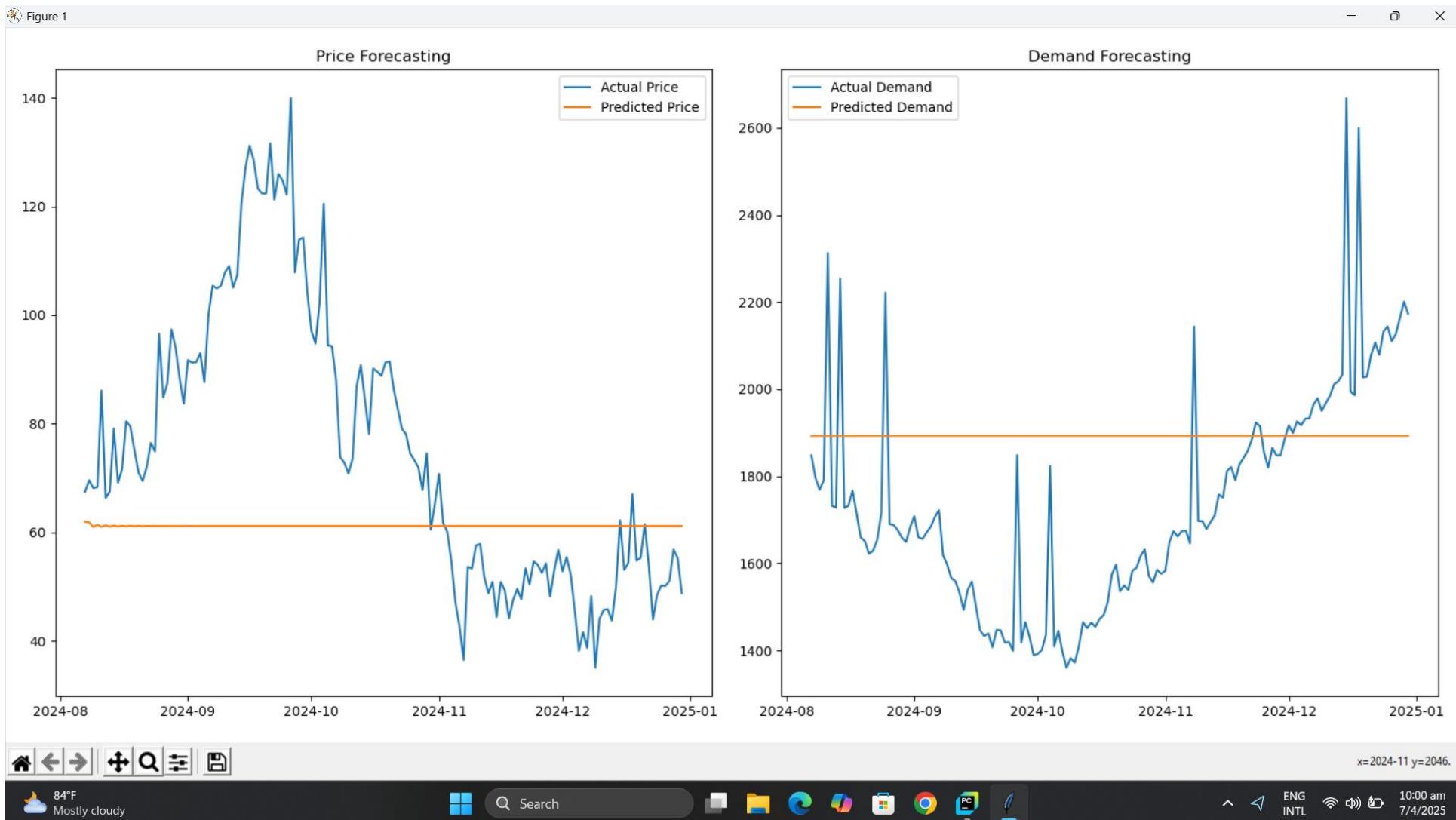
Visualization Tools: Flutter chart



Commercialization Strategy

- **Target Audience:** Small to medium-scale paddy farmers, agricultural cooperatives.
- **Revenue Model:**
 - Freemium (basic insights free, advanced analytics premium).
 - Partnerships with governments/NGOs for subsidies.
 - Advertisement revenue from agricultural products.
- **Distribution:**
 - Mobile app stores (Google Play, Apple App).
 - Partnerships with farming associations.
 - Workshops and awareness campaigns.
- **Growth:**
 - Localized market data and languages.
 - Expand to other crops/agriculture industries.
 - Integrate with e-commerce platforms.

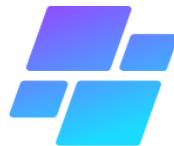
Training Forecast



Prediction Forecast

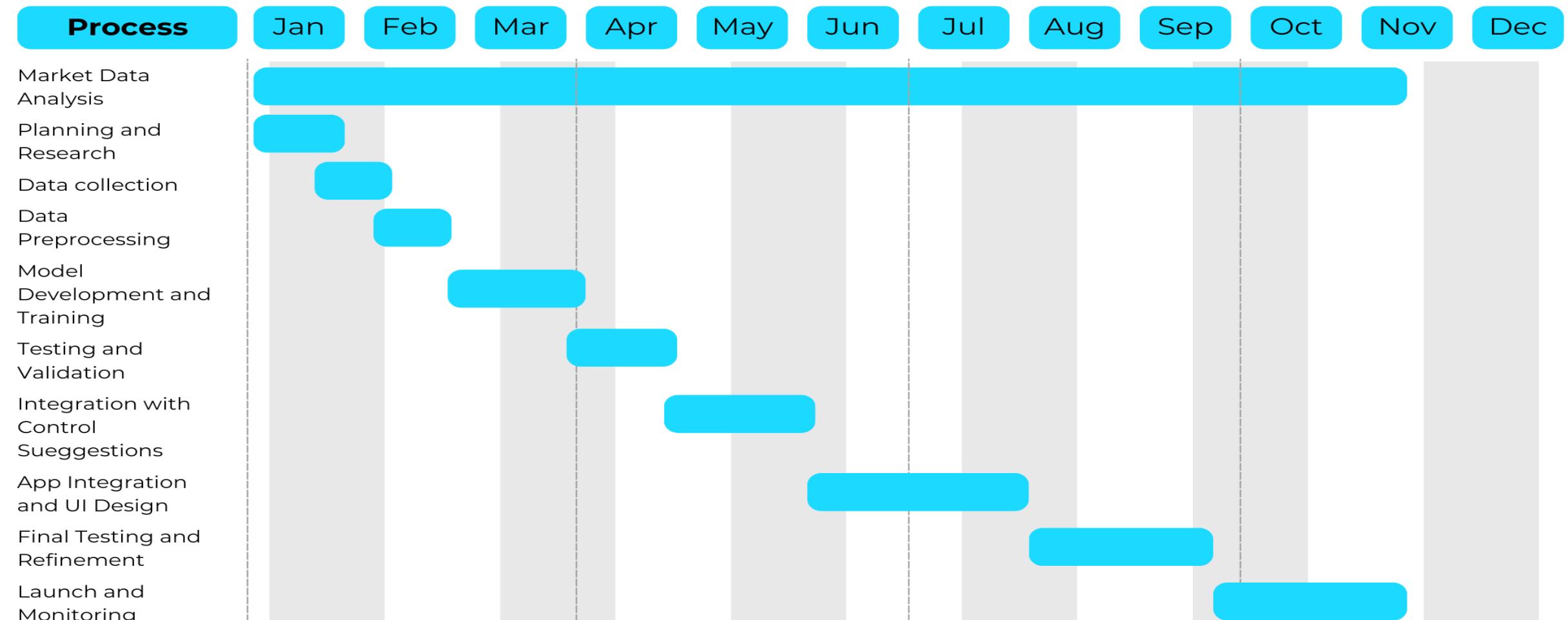
The screenshot shows a Python development environment with the following details:

- Project Structure:** A folder named "My Final [Market Price & Demand Forecast]" contains files: "demand_forecast_model.pkl", "market_data.csv", and "predict_forecast.py".
- Code Editor:** The file "predict_forecast.py" is open, displaying code related to price and demand forecasting.
- Output Window:** The terminal output shows:
 - "Next 7 Days Forecast :" followed by a table of date, price_forecast, and demand_forecast values.
 - "Forecast with Percentage Changes:" followed by a table of date, price_forecast, price_change%, demand_forecast, and demand_change% values.
 - "Process finished with exit code 0"
- System Status Bar:** Shows the time (17:53), file path (My Final > predict_forecast.py), and system status (84°F, Mostly cloudy).



Gantt Chart

R25-057 | Agri Doc App



References

- Department of Agriculture, "Welcome to the Department of Agriculture," [Online]. Available: <https://doa.gov.lk/>. [Accessed: Jan. 24, 2025].
- Department of Census and Statistics, "Paddy Crop Cutting Survey 2022," [Online]. Available: <https://www.statistics.gov.lk/Resource/en/Agriculture/Publications/PaddyCropCuttingSurvey2022.pdf>. [Accessed: Jan. 24, 2025].
- Department of Census and Statistics, "Paddy Statistics 2019-2020 Maha Season," [Online]. Available:

- International Water Management Institute (IWMI), "Improving Water Productivity in Sri Lankan Paddy Lands," [Online]. Available: <https://publications.iwmi.org/pdf/H013481.pdf>. [Accessed: Jan. 24, 2025].
- D. Hettige, "Paddy Cultivation Practices in Sri Lanka," [Online]. Available: <https://dh-web.org/botany/Paddycultivation.pdf>. [Accessed: Jan. 24, 2025].
- Library of the University of Ruhuna, "Research on Paddy Cultivation," [Online]. Available: <http://ir.lib.ruh.ac.lk/bitstream/handle/iruor/14880/37%20%2885-104%29.pdf?sequence=1&isAllowed=y>. [Accessed: Jan. 24, 2025].
- Department of Agriculture, "Agriculture Statistics Handbook," [Online]. Available: <https://doa.gov.lk/wp-content/uploads/2020/05/AgstatBK.pdf>. [Accessed: Jan. 24, 2025].



IT21809460|LAVANYA.M

BSc (Hons) Information Technology Specializing in Information Technology



WEED IDENTIFICATION



Introduction

Background

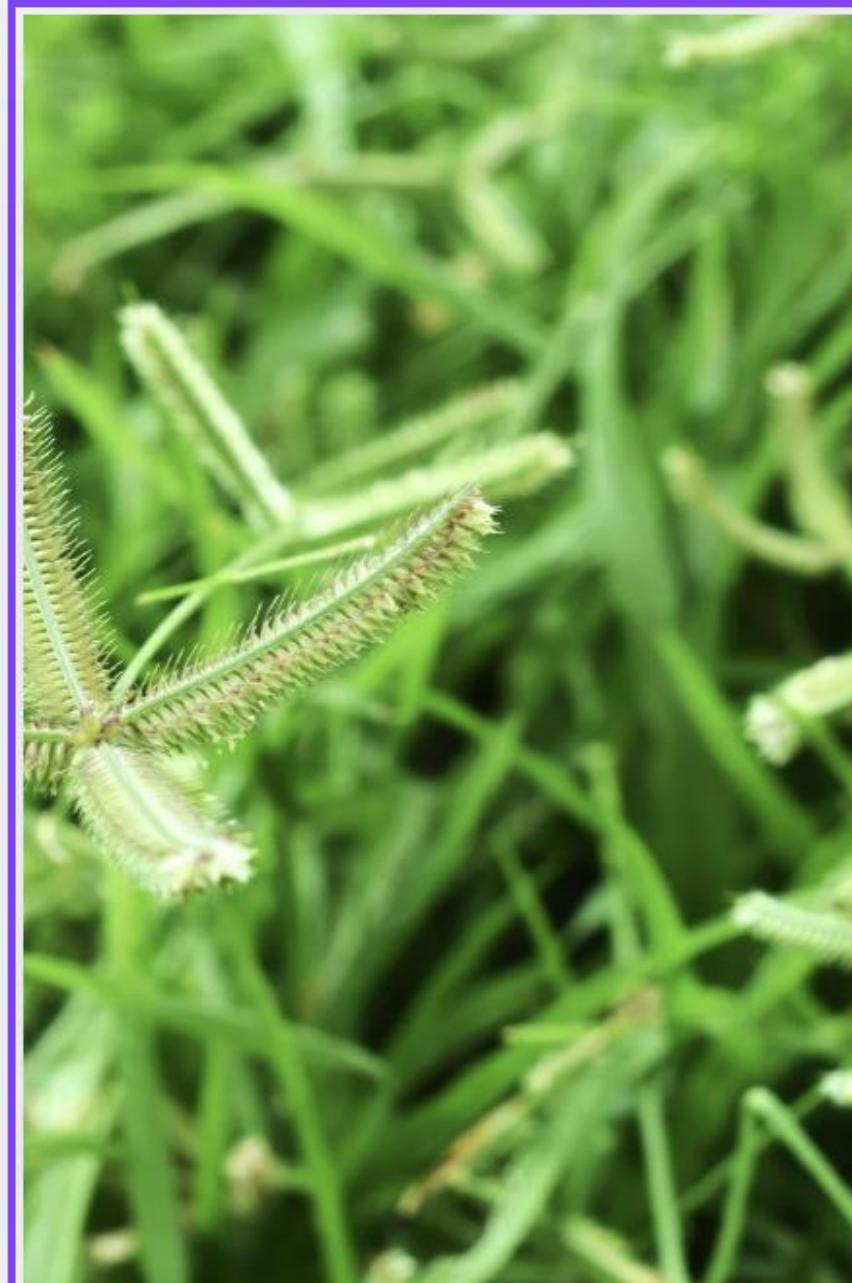
- Weed management boosts crop yield by reducing competition for resources. Traditional methods are labor-intensive and inefficient for large-scale farming.
- Manual detection is often inaccurate, and excessive herbicide use harms the environment. Advanced tools for precision weed control are lacking.
- AI and ML enhance weed detection through image recognition, classify weeds efficiently, and provide tailored recommendations for effective control.
- AI systems reduce herbicide use, save time, promote sustainability, and improve crop yield.



Research problem

How can AI-based image recognition techniques be employed to develop an adaptive system that effectively identifies and classifies weeds in paddy fields, providing real-time recommendations for environmentally sustainable weed management?





Specific and Sub Objectives

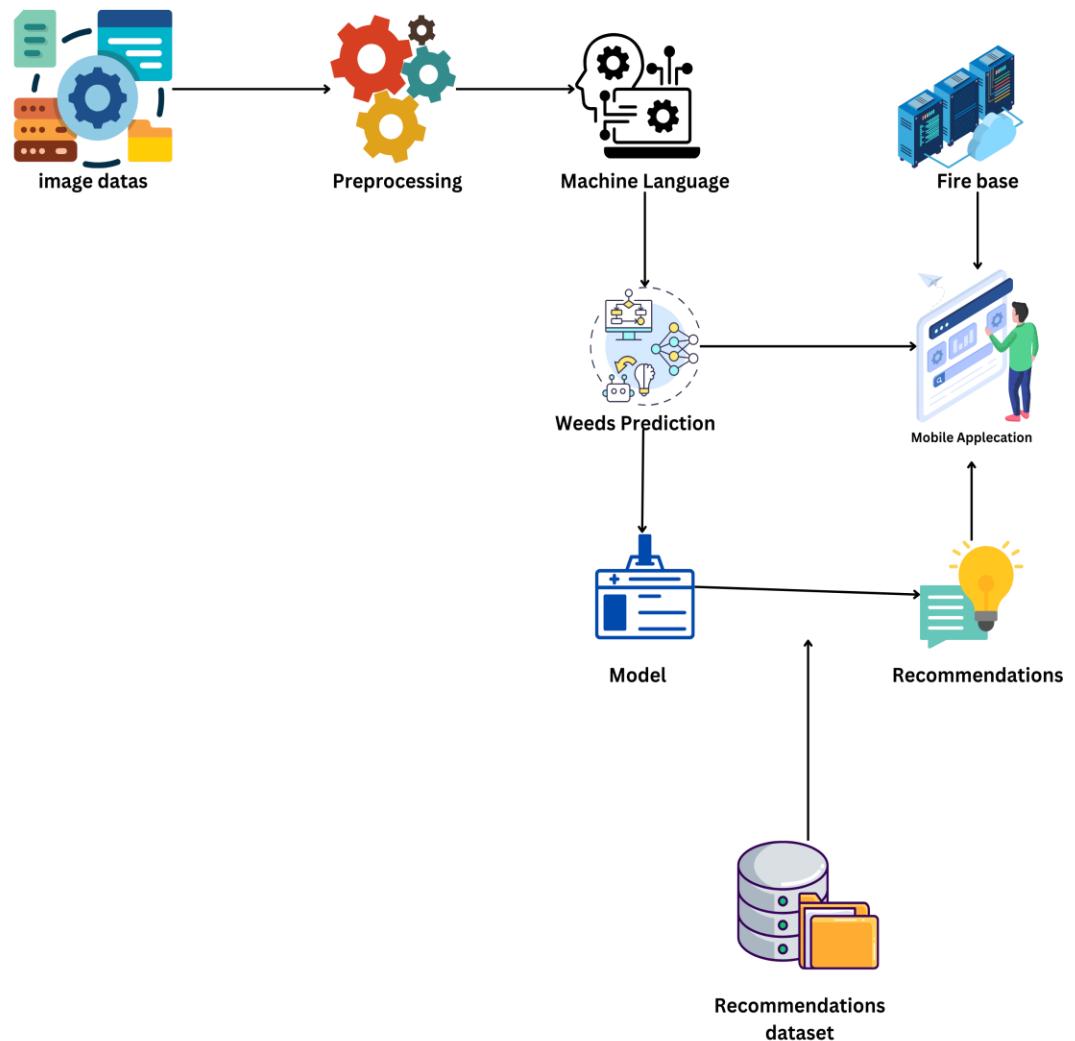
Specific Objectives

- Develop an Accurate Weed Identification Model
- Optimize for Real-Time Image Processing
- Provide Sustainable Weed Control Recommendations

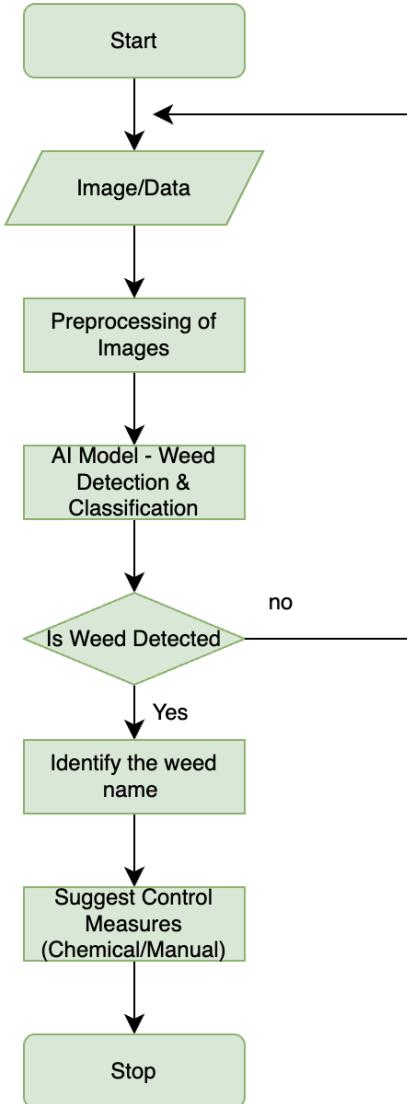
Sub-Objectives

- Data Collection & Labeling
- Model Training & Evaluation
- Cloud-Based Real-Time Alerts
- User Interface Design

System Diagram



Flow chart



Tools & Technologies



Technologies

- OpenCV, Flutter, Firebase, FastAPI , TensorFlow, VGG16

Storage

- Firebase Cloud Storage

Programming Languages

- Python (training the machine learning model)
- Dart

Functional

Identify and classify weed species from field images.

Notify farmers through the mobile app about detected weeds.

Provide real-time weed control recommendations.

Non-Functional

Fast processing

◦High availability

◦Multilingual support.



Weeds Identification System



Create an easy-to-use mobile or web app with accurate AI-based weed detection, integrating it with agricultural tools like drones or automated sprayers.

Develop the Product:



Identify target customers (farmers, agribusinesses, governments), analyze competitors, and tailor the product to meet market needs

Market Research



Offer subscription plans, freemium services, or B2B licensing to generate revenue.

Revenue Models



Collaborate with agricultural equipment manufacturers, research institutions, and government bodies.

Partnerships



Promote through case studies, digital marketing, trade shows, and referral programs.

Marketing



Scale globally with cloud technology, localize for different regions, and integrate with global agricultural data.

Global Expansion



Highlight environmental benefits by reducing herbicide use and promoting sustainable farming practices.

Sustainability

Model training

The screenshot shows a Jupyter Notebook environment with a dark theme. On the left is a file tree containing various Python files and image files. The central area contains a code cell with the following Python script:

```
...  
20180322-114158-2.jpg  
20180322-115953-2.jpg  
20180322-133822-1.jpg  
abc.png  
augmentation.py  
Dataset.txt  
FlaskAPI.py  
hh.jpg  
plots.py  
Predict.py  
species_label_encoder.pkl  
weed_species_classifier_vg  
WeedModel_Train.py  
> External Libraries  
└ Scratches and Consoles  
    └ Extensions  
        Database Tools and SQL  
64  
65  
66  
67  
68 ▷ 69  
70  
71  
72  
73  
74  
75  
76
```

```
except Exception as e:  
    print(f"Error processing {filename}: {str(e)}")  
return results  
  
def _display_prediction(self, img_path, class_name, confidence): 1 usage  
    img = load_img(img_path, target_size=(300, 300))  
  
    plt.figure(figsize=(8, 6))  
    plt.imshow(img)  
    plt.title(f"Prediction: {class_name}\nConfidence: {confidence:.2%}")  
    plt.axis('off')  
    plt.show()  
  
if __name__ == "__main__":  
    classifier = WeedClassifier()  
  
    # Single image prediction  
    test_image = "20180322-114158-2.jpg"  
    if os.path.exists(test_image):  
        class_name, confidence = classifier.predict(test_image)  
        print(f"\nPrediction for {test_image}:")  
        print(f"Class: {class_name}")
```

Below the code cell is a 'Run' button followed by the name of the script being run: 'WeedModel_Train'. The output pane shows the following logs:

```
Run WeedModel_Train ×  
Run Cell :  
Epoch 1: val_accuracy improved from -inf to 0.5782, saving model to weed_species_classifier_vgg16.h5  
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  
438/438 1061s 2s/step - accuracy: 0.5108 - loss: 1.5979 - val_accuracy: 0.5782 - val_loss: 1.1861  
Epoch 2/20  
438/438 0s 2s/step - accuracy: 0.5531 - loss: 1.2622
```

Accuracy 86.19%

The screenshot shows a Python development environment with the following components:

- Project View:** Shows a file structure for a project named "RP". It includes files like "ion.py", "plots.py", "WeedModel_Train.py", "Predict.py", "species_label_encoder.pkl", "weed_species_classifier_vg", and "WeedModel_Train.py".
- Code Editor:** The "WeedModel_Train.py" file is open, displaying Python code for training a model. The code imports various libraries such as os, pandas, numpy, matplotlib, sklearn, tensorflow, and joblib. It defines paths for the dataset and labels, and sets a random seed.
- Plots:** Two line graphs are displayed in the main window:
 - Accuracy over epochs:** Shows Train Accuracy (blue line) increasing from ~0.55 to ~0.70, and Validation Accuracy (orange line) increasing from ~0.58 to ~0.75.
 - Loss over epochs:** Shows Train Loss (blue line) decreasing from ~1.4 to ~0.8, and Validation Loss (orange line) decreasing from ~1.1 to ~0.75.
- Run Log:** The "Run" tab shows the output of the "WeedModel_Train" script. It includes a traceback for an AttributeError related to "data_Validation" and a message indicating the process finished with exit code 1.
- Status Bar:** The bottom status bar shows the current time (15:113), file encoding (CRLF), character set (UTF-8), indentation (4 spaces), Python version (Python 3.9), and other system information.

Relative Topic Suggestion

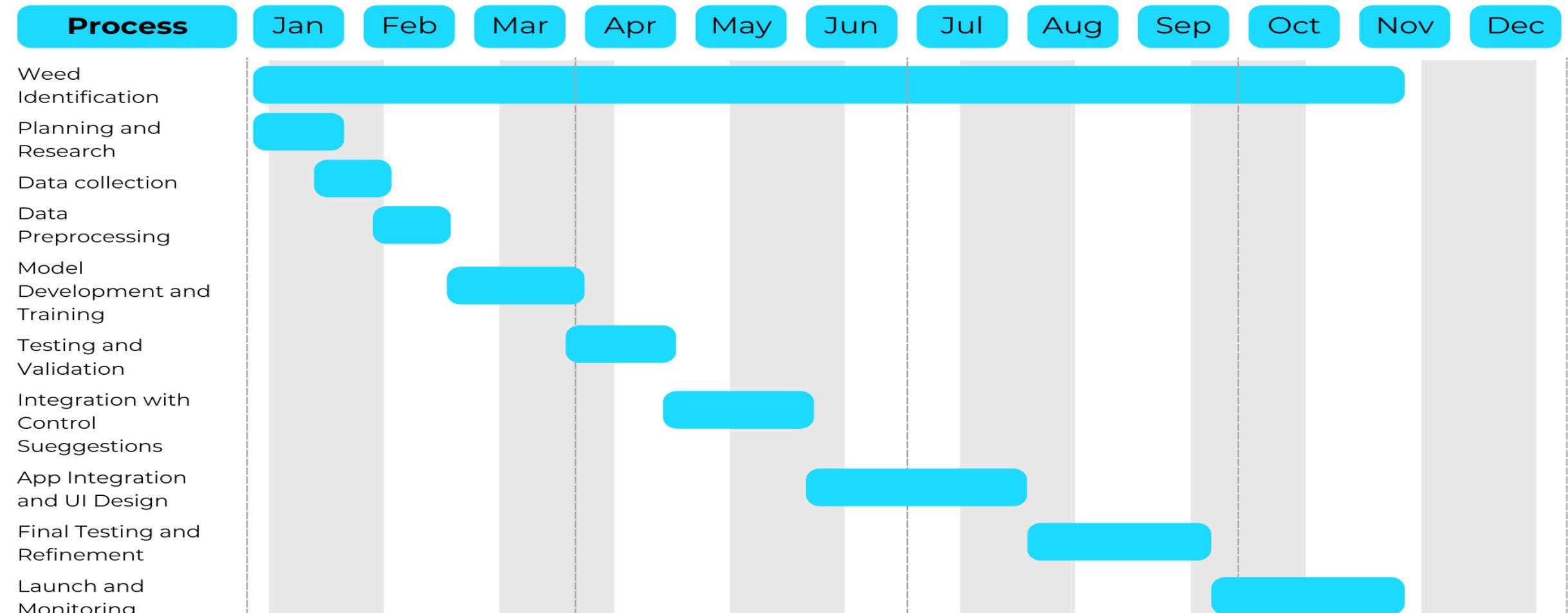
The screenshot shows a Python development environment, likely PyCharm, with the following details:

- Project Structure:** The project is named "RP" and contains a "Dataset" folder with "images" and "labels.csv", a file "abc.png", and several Python files: "augmentation.py", "Dataset.txt", "FlaskAPI.py", "hh.jpg", "plots.py", "Predict.py", "species_label_encoder.pkl", and "weed_species_classifier_vgg16.h5".
- Code Editor:** The "Predict.py" file is open, showing code for training a model and evaluating it. It includes dumping a label encoder and printing class mapping.
- Run Tab:** The "Predict" configuration is selected, and the command is set to run "Predict.py". The output window shows the command being run and the prediction for "abc.png" which is "Lantana".
- Plots Tab:** A "Plots" tab is visible on the right, showing a grid of six small images labeled 1 through 6, each with a caption below it. The first image is labeled "Prediction: Lantana Confidence: 0.9999".



Gantt Chart

R25-057 | Agri Doc App



References

- M. Kamilaris and F. Prenafeta-Boldú, "Deep learning in agriculture: A survey," *Computers and Electronics in Agriculture*, vol. 147, pp. 70–90, Apr. 2018. [Online]. Available: <https://doi.org/10.1016/j.compag.2018.02.016>
- P. D. Dobermann, "AI and Machine Learning in Agriculture: Opportunities for Precision Weed Management," *Advances in Agronomy*, vol. 168, pp. 1–35, 2021. [Online]. Available: <https://doi.org/10.1016/bs.agron.2021.06.004>
- Food and Agriculture Organization, "Artificial Intelligence in Agriculture," FAO, Rome, Italy, 2021. [Online]. Available: <https://www.fao.org>
- International Crops Research Institute for the Semi-Arid Tropics (ICRISAT), "Sustainable Farming Practices and AI Applications," ICRISAT, Hyderabad, India, 2020. [Online]. Available: <https://www.icrisat.org>
- Agriculture Department of Sri Lanka, "Weed Management Practices and Modern Applications of AI in Agriculture," Colombo, Sri Lanka, 2023. [Online]. Available: <https://www.doa.gov.lk>
- Sri Lanka Agrarian Service Centers, "Field Observations on Weed Management Using Advanced Technologies," Sri Lanka Agrarian Service Centers, 2024.

- B. Marambe, "Planning for Effective Weed Management: Lessons from Sri Lanka," *ResearchGate*, 2013. [Online]. Available: <https://www.researchgate.net/publication/236847755>
- S. Mendis, "Weed Management Practices in Rice Cultivation," *Tropical Agricultural Research and Extension*, vol. 22, no. 1 & 2, pp. 19–25, 2019. [Online]. Available: [https://tare.agri.ruh.ac.lk/pdf/V-22.1%20&%202/4.%20TARE%2022\(1%20&%202\)%20WEED%20MANAGEMENT%20PRACTICES%20IN%20RICE%20CULTIVATION%20CS.19.01%20MENDIS%20APS.pdf](https://tare.agri.ruh.ac.lk/pdf/V-22.1%20&%202/4.%20TARE%2022(1%20&%202)%20WEED%20MANAGEMENT%20PRACTICES%20IN%20RICE%20CULTIVATION%20CS.19.01%20MENDIS%20APS.pdf)
- N. Sahota, "Automated Weed Detection: AI-Enhanced Solution for Weed Control," 2023. [Online]. Available: <https://www.neilsahota.com/automated-weed-detection-ai-enhanced-solution-for-weed-control/>
- M. Gazzard et al., "WeedScout: Real-Time Autonomous Blackgrass Classification and Mapping Using Dedicated Hardware," *arXiv preprint arXiv:2405.07349*, 2024. [Online]. Available: <https://arxiv.org/abs/2405.07349>
- Y. Du et al., "Deep-CNN Based Robotic Multi-Class Under-Canopy Weed Control in Precision Farming," *arXiv preprint arXiv:2112.13986*, 2021. [Online]. Available: <https://arxiv.org/abs/2112.13986>



IT21819506 | V ABILAXSHAN

BSc (Hons) Information Technology Specializing in Information Technology

Smart Irrigation System



"SMART" Irrigation

INTRODUCTION

- Agriculture is a cornerstone of Sri Lanka's economy, with paddy farming being one of the primary crops cultivated.
- Irrigation is essential for paddy farming, but traditional methods are often inefficient, relying on fixed schedules rather than real-time water needs.
- Global efforts in smart agriculture have demonstrated the potential of IoT and machine learning in optimizing irrigation systems, yet these technologies are underutilized in Sri Lanka.





Research gaps

Technological Gaps

- Current systems lack real-time monitoring of environmental and soil conditions.
- Limited use of machine learning models to predict irrigation needs based on data trends and environmental factors.

Farmer Adoption Challenges

- Most existing solutions are either too complex or expensive for local farmers to adopt.
- Lack of mobile-friendly platforms tailored to Sri Lankan farmers' needs and literacy levels.

Localized Solutions

- Few solutions address the unique environmental and socio-economic conditions of Sri Lankan paddy fields, such as tank irrigation systems and irregular rainfall



- **Research problem**

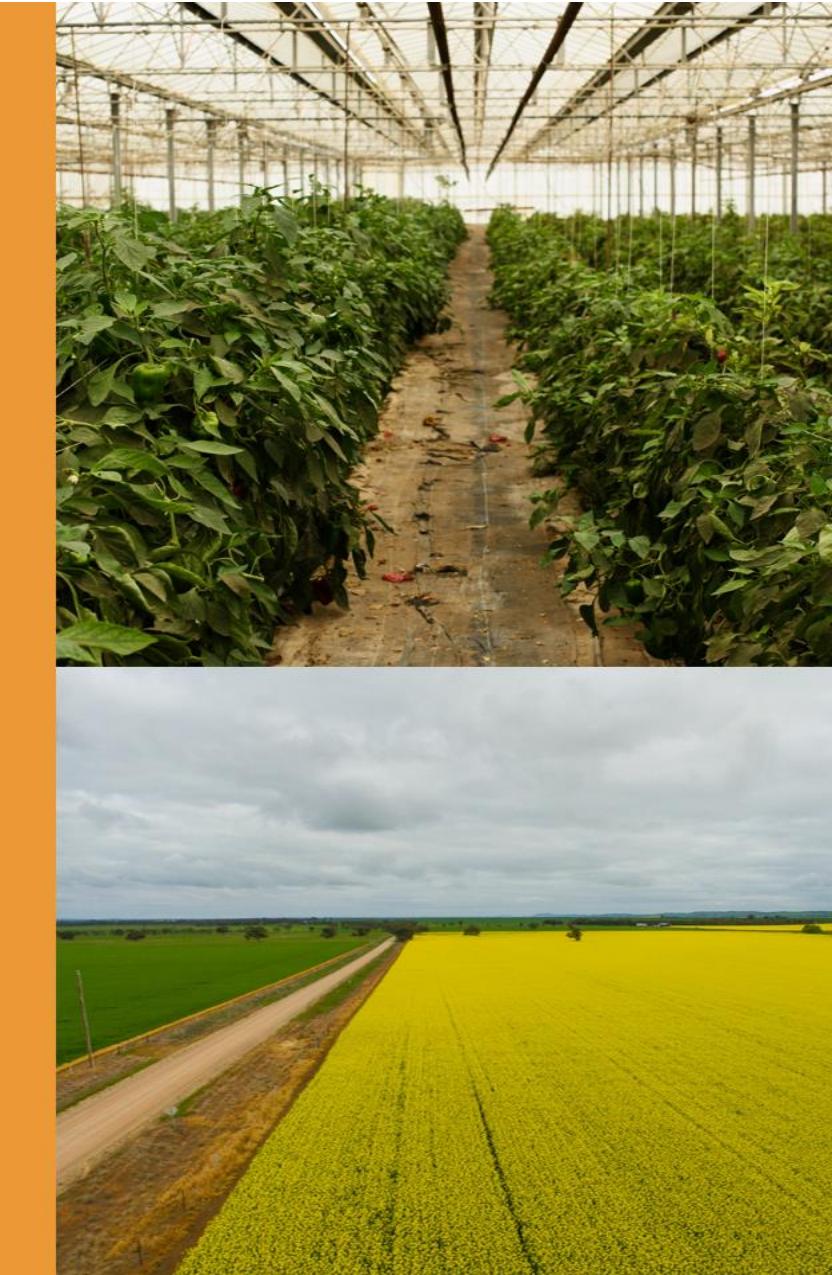
Main Problem:

- Farmers need **reduce human interfere** and based **cost-effective, easy-to-use, and data-driven system** that enables them to make precise irrigation decisions based on real-time data, reducing water wastage and improving crop yields

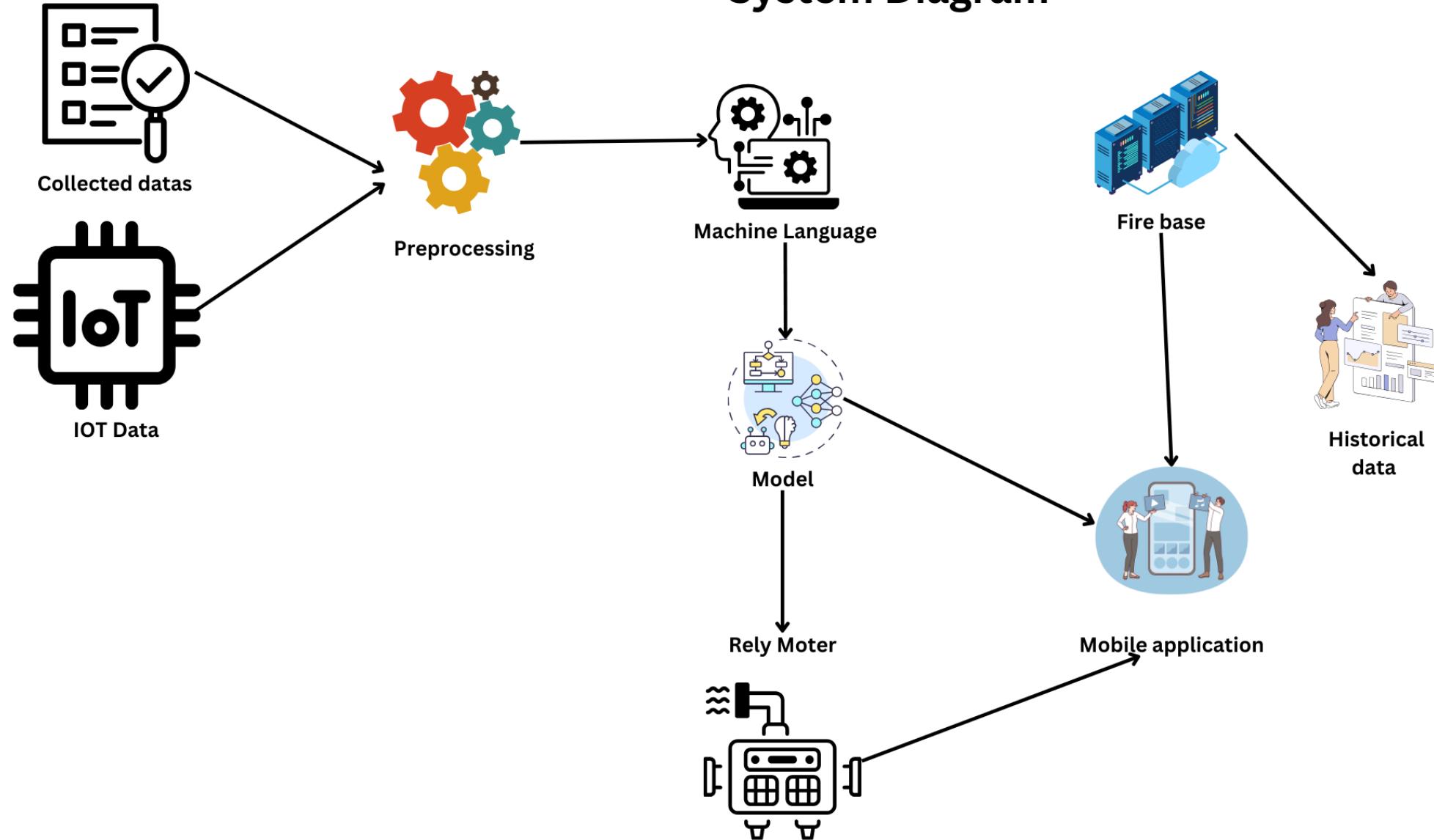
.

- **Core Challenges:**

1. Designing a system that integrates data from **IoT sensors** and external environmental datasets (soil moisture sensor, environment temperature sensor,)
2. Ensuring accurate and reliable predictions using **machine learning** models (trained on real-world and historical data).
3. Balancing **technical sophistication** with **simplicity and affordability** to ensure broad adoption among Sri Lankan farmers.



System Diagram



Project Methodology

Phase 1: Requirements Gathering

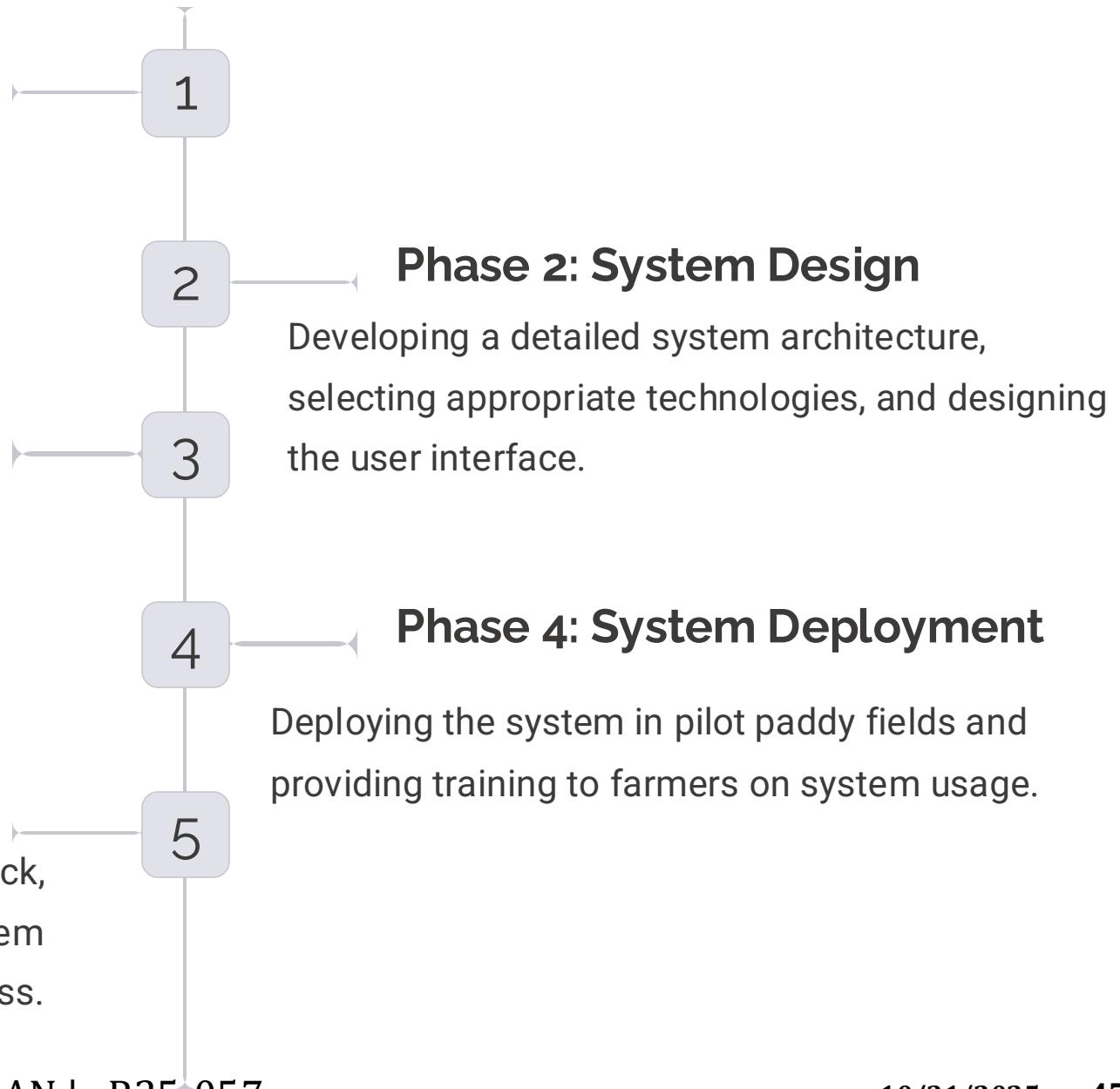
Conducting field research to understand farmer needs and environmental conditions.

Phase 3: Prototype Development

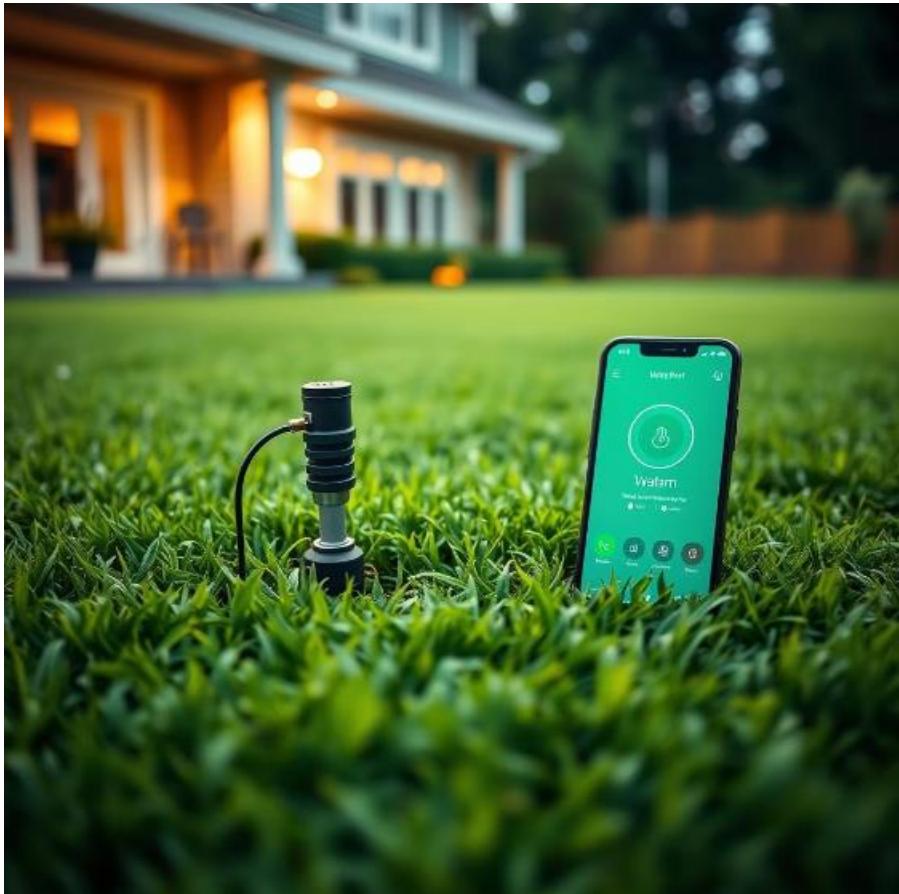
Developing a functional prototype of the system and conducting field testing to ensure accuracy and effectiveness.

Phase 5: Evaluation & Optimization

Monitoring system performance, collecting user feedback, and making necessary adjustments to optimize system efficiency and effectiveness.



System Overview



IoT Sensors

Collect real-time soil moisture data, providing insights into the actual water needs of the paddy field.

Data Analysis

Analyzes collected data, environmental factors (temperature, humidity, rainfall), and historical data.

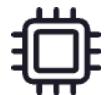
Irrigation Recommendations

Generates dynamic irrigation schedules and notifications based on real-time and historical data.

User Interface

Provides a user-friendly platform for farmers to access real-time water usage data, irrigation recommendations, and system performance.

System Requirements



Hardware

Server: Intel Core i5, 8GB RAM, 500GB SSD, high-speed internet.

Mobile Devices:
Android/iOS, 2GB RAM,
50MB storage



Personnel

System Administrator,
Data Analyst, Software
Developer.



Software

Development: Python, Dart, MySQL, Android Studio.

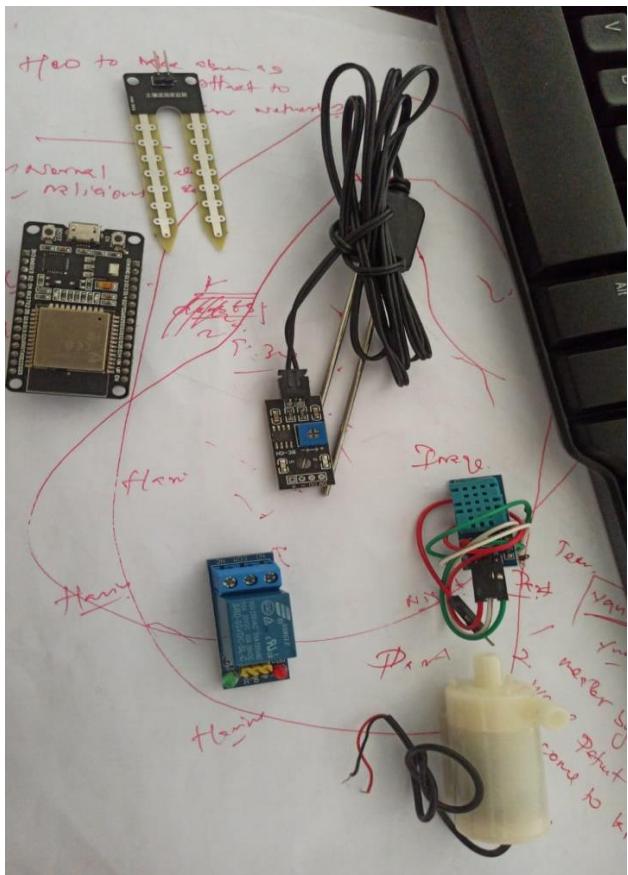
Machine Learning:

Supervised Learning
Visualization: Chart.js.

API Integration: RESTful APIs
for real-time data.



Key Technologies



Internet of Things (IoT)

Utilizes wireless sensor networks for continuous monitoring of soil moisture and other environmental parameters.

Machine Learning

Develops predictive models to analyze data and provide accurate irrigation recommendations.

Using python

Cloud Computing

Enables secure data storage, analysis, and communication between system components.

Mobile App

Provides farmers with a user-friendly interface to access real-time data, irrigation schedules, and system alerts.

Using flutter

Commercialization Strategy



1 Target Audience

Paddy farmers, agricultural cooperatives, and government agencies involved in promoting sustainable farming practices.



2 Value Proposition

Reduced water consumption, increased crop yields, and improved profitability for farmers, while promoting sustainable agriculture.



3 Marketing & Sales

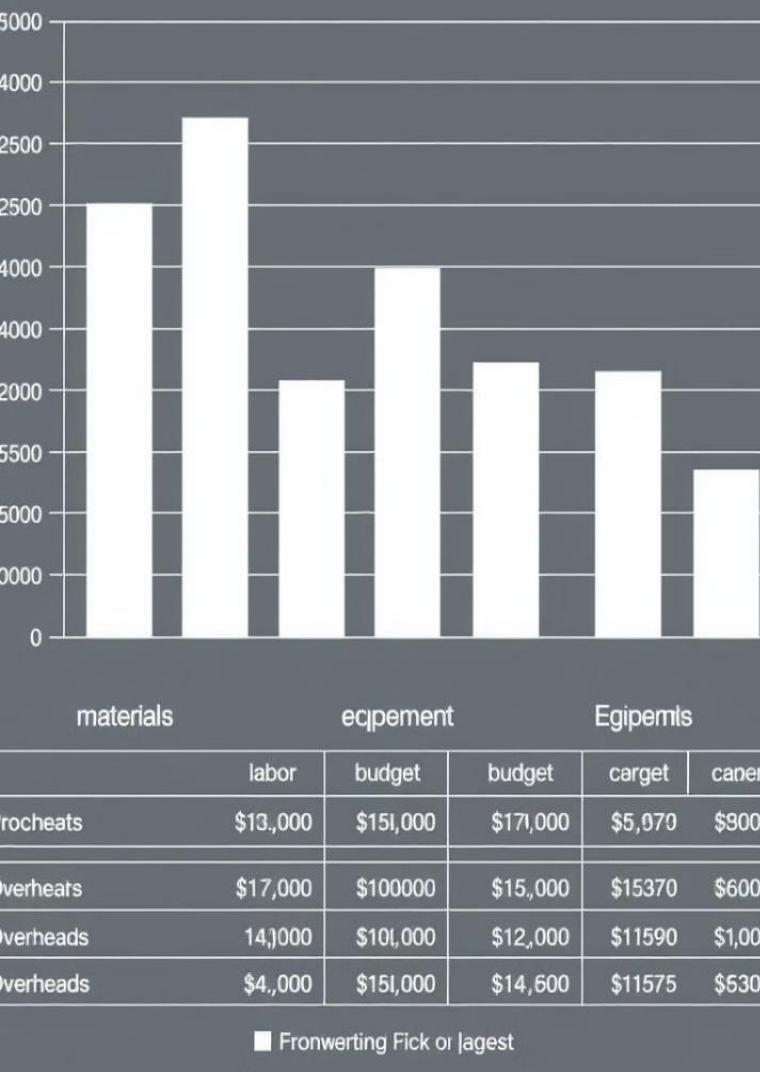
Targeted marketing campaigns, partnerships with agricultural organizations, and demonstrations in pilot paddy fields.



4 Pricing Strategy

Subscription-based model with flexible pricing plans tailored to the specific needs of different farmer groups.





Budget & Funding

35k

Development Costs

Hardware, software,
and development
team expenses.

500K

Marketing & Sales

Marketing
campaigns,
partnerships, and
training programs.

25k

Pilot Deployment
System deployment
costs and farmer
training.

Train model screen shots



Prediction screen shot

The screenshot shows the PyCharm IDE interface with the following details:

- Project:** my project [IoT-Based Automated System] C:\Users\V.Asvika
- Files:** Train.py, Predict.py, soil_watering_data.csv, watering_duration_model.pkl, watering_scaler.pkl
- Panels:**
 - Editor:** Displays the Predict.py code. The code defines a predict_watering function that loads a model and scaler, handles FileNotFoundError, creates features, scales them, and makes predictions. It also prints a message if the model is not found.
 - Run:** Shows the terminal output of the Predict.py run.
- Terminal Output:**

```
Maximize your efficiency with pandas
Try PyCharm Professional Dismiss
15 def predict_watering(soil_moisture, temperature, humidity): 3 usages
16     model = joblib.load('watering_duration_model.pkl')
17     scaler = joblib.load('watering_scaler.pkl')
18 except FileNotFoundError:
19     print("Model not found. Please train the model first by running train.py")
20     return None
21
22 # Create features
23 moisture_deficit = max(0, 40 - soil_moisture) # 40 is our target moisture level
24 features = pd.DataFrame( data=[[soil_moisture, temperature, humidity,
25                                 temperature / (humidity + 1),
26                                 moisture_deficit]],
27                               columns=['soil_moisture', 'temperature', 'humidity',
28                                         'temp_humidity_ratio', 'moisture_deficit'])
29
30 # Scale and predict
31 features_scaled = scaler.transform(features)
32
Predict
Predictions:
1. Dry conditions (25%, 28°C, 40%): {'soil_moisture': 25, 'temperature': 28, 'humidity': 40, 'watering_needed': True, 'duration_minutes': 20.5, 'recommendation': 'Water now'}
2. Moderate conditions (35%, 22°C, 65%): {'soil_moisture': 35, 'temperature': 22, 'humidity': 65, 'watering_needed': True, 'duration_minutes': 2.5, 'recommendation': 'Water later'}
3. Moist conditions (45%, 20°C, 75%): {'soil_moisture': 45, 'temperature': 20, 'humidity': 75, 'watering_needed': False, 'duration_minutes': 0, 'recommendation': 'No water needed'}
```
- Status Bar:** 28:48 CRLF UTF-8 4 spaces Python 3.8

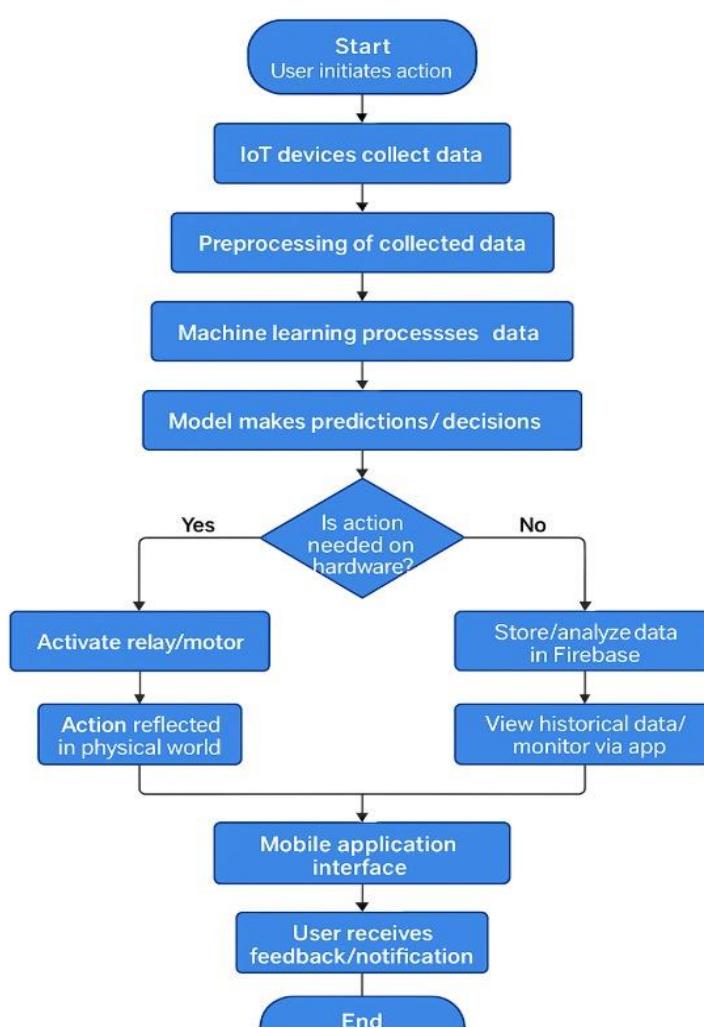
Arduino readings

The screenshot shows the Arduino IDE interface with a dark theme. The top menu bar includes File, Edit, Sketch, Tools, and Help. A toolbar below the menu contains icons for saving, running, and connecting. The sketch name is "sketch_apr7a.ino". The code reads environment temperature and humidity from DHT sensors and soil moisture from an analog pin, printing the results to the Serial Monitor. The Serial Monitor window shows the output of the code, including sensor values and a progress bar for package download.

```
sketch_apr7a | Arduino IDE 2.0.4-nightly-20230216
File Edit Sketch Tools Help
✓ → ↻ ESP32 Dev Module
sketch_apr7a.ino
51     } else {
52         Serial.println("Error: Failed to read from DHT sensor");
53     } else {
54         Serial.print("Environment Temperature: ");
55         Serial.print(envTemperature);
56         Serial.println(" °C");
57         Serial.print("Environment Humidity: ");
58         Serial.print(humidity);
59         Serial.println(" %");
60     }
61
62     // Read soil moisture value
63     int soilMoistureValue = analogRead(soilSensorPin);
64     Serial.print("Soil Moisture Level: ");
65     Serial.println(soilMoistureValue);
66
67     Serial.println("-----");
68
69
70     delay(2000);
71 }

Output Serial Monitor ×
Message (Enter to send message to 'ESP32 Dev Module' on 'COM3')
Soil Temperature: 28.30 °C
Environment Temperature: 28.30 °C
Environment Humidity: 68.00 %
Soil Moisture Level: 4095
-----
Soil Temperature: 28.44 °C
Environment Temperature: 28.30 °C
Environment Humidity: 68.00 %
Soil Moisture Level: 4095
-----
Soil Temperature: 28.44 °C
New Line 115200 baud
Downloading index: package_esp8266com_index.json
Ln 47, Col 37 ESP32 Dev Module on COM3 21:01 07/04/2025
28°C Mostly cloudy Search
```

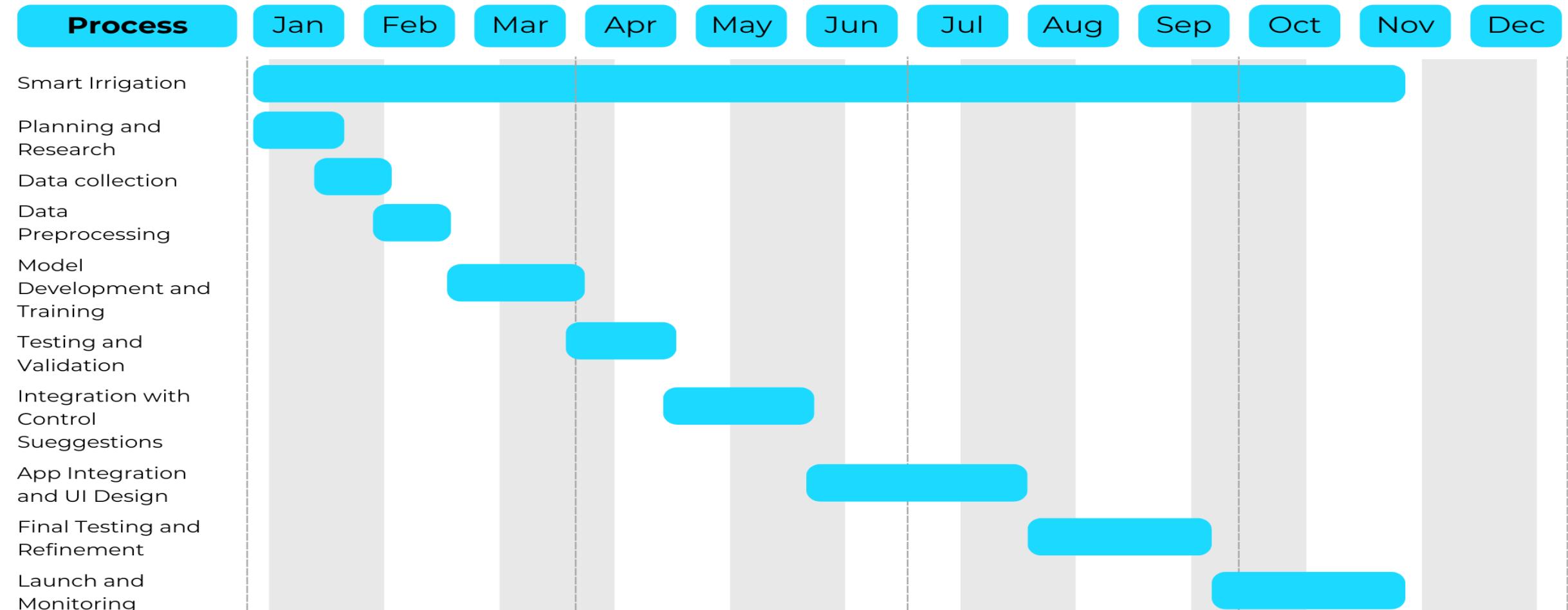
Flow chart





Gantt Chart

R25-057 | Agri Doc App



References

- Department of Agrarian Development, Sri Lanka, "Tank-Level Monitoring Report 2023," Colombo, Sri Lanka, 2023.
- Department of Meteorology, Sri Lanka, "Annual Weather Report," Colombo, Sri Lanka, 2023. [Online]. Available: <http://www.meteo.gov.lk>
- M. Perera and S. Fernando, "A Study on Tank Irrigation Systems in Dry Zones of Sri Lanka," *Journal of Agricultural Sciences*, vol. 58, no. 2, pp. 120–130, Aug. 2022.
- D. Silva, "IoT-based Tank Level Monitoring for Sustainable Water Management," presented at the *International Conference on Agriculture and Environment (ICAE)*, Kandy, Sri Lanka, 2022.
- United Nations Food and Agriculture Organization, *Climate-Smart Agriculture in Sri Lanka: Practices and Strategies*, 1st ed., Rome, Italy: FAO Publishing, 2021.
- N. Jayasinghe et al., "Predicting Rainfall Patterns Using Machine Learning for Agricultural Optimization," *IEEE Transactions on Smart Agriculture*, vol. 15, no. 4, pp. 89–95, Dec. 2021.
- Department of Meteorology, Sri Lanka, "Monthly Rainfall Data," Colombo, Sri Lanka, 2023. [Online]. Available: <http://www.meteo.gov.lk>

- Agrarian Development Department, "Tank Level Monitoring Report," Sri Lanka, 2023.
- A. Sharma and S. Patel, "Real-Time Water Level Monitoring in Irrigation Tanks Using IoT Sensors," presented at the *IEEE International Conference on Smart Agriculture (ICSA)*, Pune, India, Oct. 2022.
- J. Lee, B. Park, and H. Kim, "Wireless Sensor Network Deployment for Agricultural Irrigation Management," *Sensors*, vol. 21, no. 6, pp. 1504–1515, Mar. 2021.
- M. R. Hasan, A. Saha, and M. K. Hasan, "IoT-Based Automated Irrigation System for Efficient Water Usage," *IEEE Transactions on Agriculture and Food Systems*, vol. 12, no. 1, pp. 89–96, Jan. 2023.
- N. Gupta, R. Verma, and K. Das, "Smart Agriculture Using IoT: A Sensor-Based Approach," *International Journal of Advanced Computer Science and Applications (IJACSA)*, vol. 10, no. 4, pp. 451–456, Apr. 2019.
- S. Ahmed et al., "IoT-Based Water Tank Monitoring System: A Case Study on Rural Farming," *Proceedings of the International Conference on IoT and Applications*, Hyderabad, India, Dec. 2021, pp. 78–85.



IT21813320 | A.SHIVAPHIRIYAN

BSc (Hons) Information Technology Specializing in Information Technology

PEST IDENTIFICATION & CONTROL



Introduction

This research project focuses on developing a pest identification and control system for paddy farmers. It leverages image processing and machine learning to identify pests, determine severity, and recommend suitable fertilizers. The system is designed to support sustainable farming through timely and accurate decision-making.



Research Problem

Paddy cultivation in Sri Lanka is heavily impacted by pest infestations, yet most small-scale farmers lack access to timely and accurate pest identification methods. Traditional approaches are manual, depend on expert knowledge, and often result in delays in diagnosis and improper use of fertilizers. This leads to reduced crop yield, overuse of chemicals, and long-term soil degradation.





Research Gap

Current agricultural advisory systems lack automation in pest detection, struggle with accuracy in rural field conditions, and provide generalized recommendations. This project addresses these issues using AI and image analysis for tailored, real-time insights.



Objectives



Main Objective



To develop an AI-powered pest identification and control system using image processing to improve paddy farming productivity.

Sub-Objectives



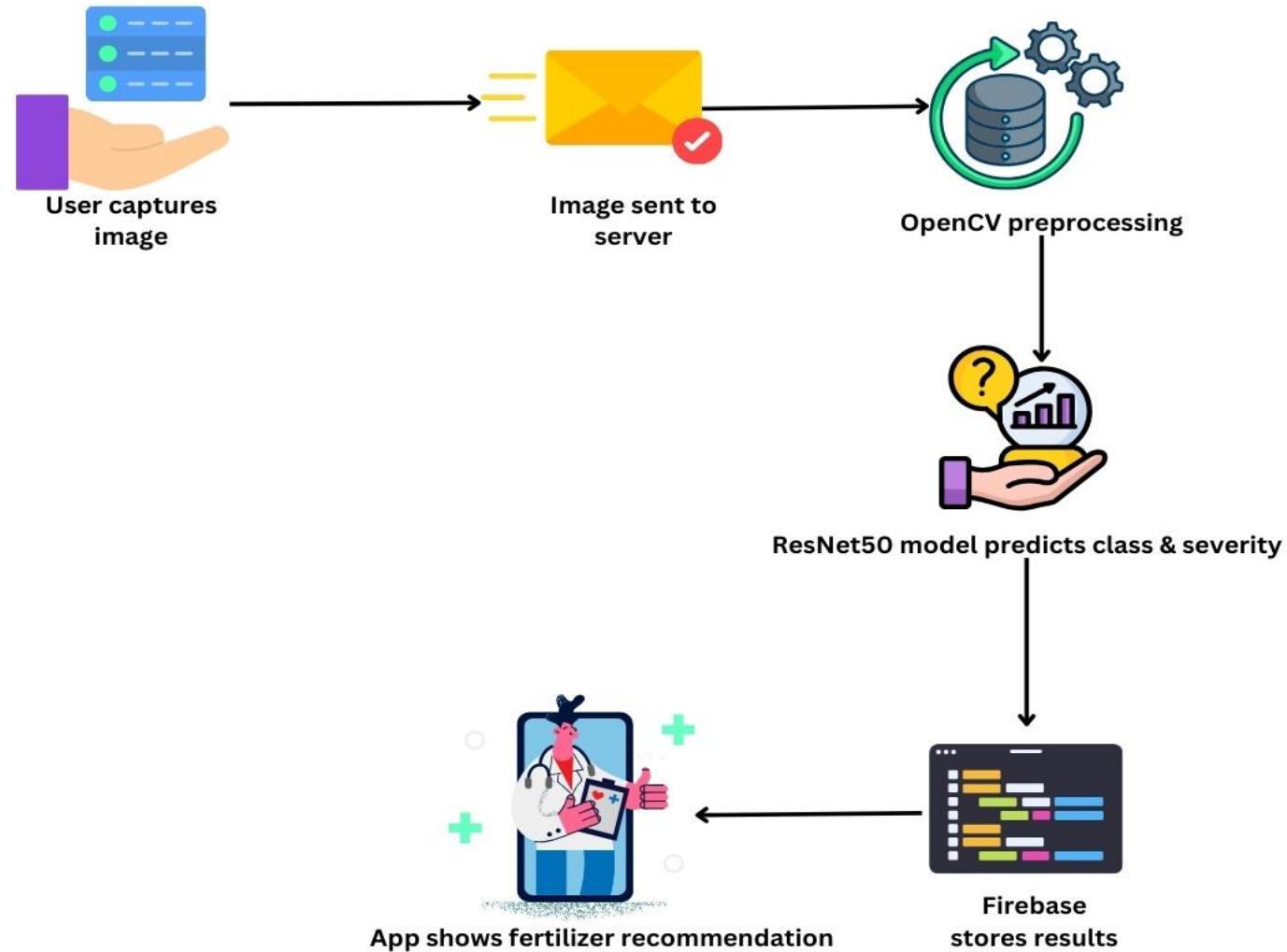
1. Identify pests and classify them as harmful or harmless.
2. Determine severity of infestation (low, mild, severe).
3. Recommend suitable fertilizers (organic or artificial).
4. Analyze leaf affliction severity for further treatment.

Project Methodology

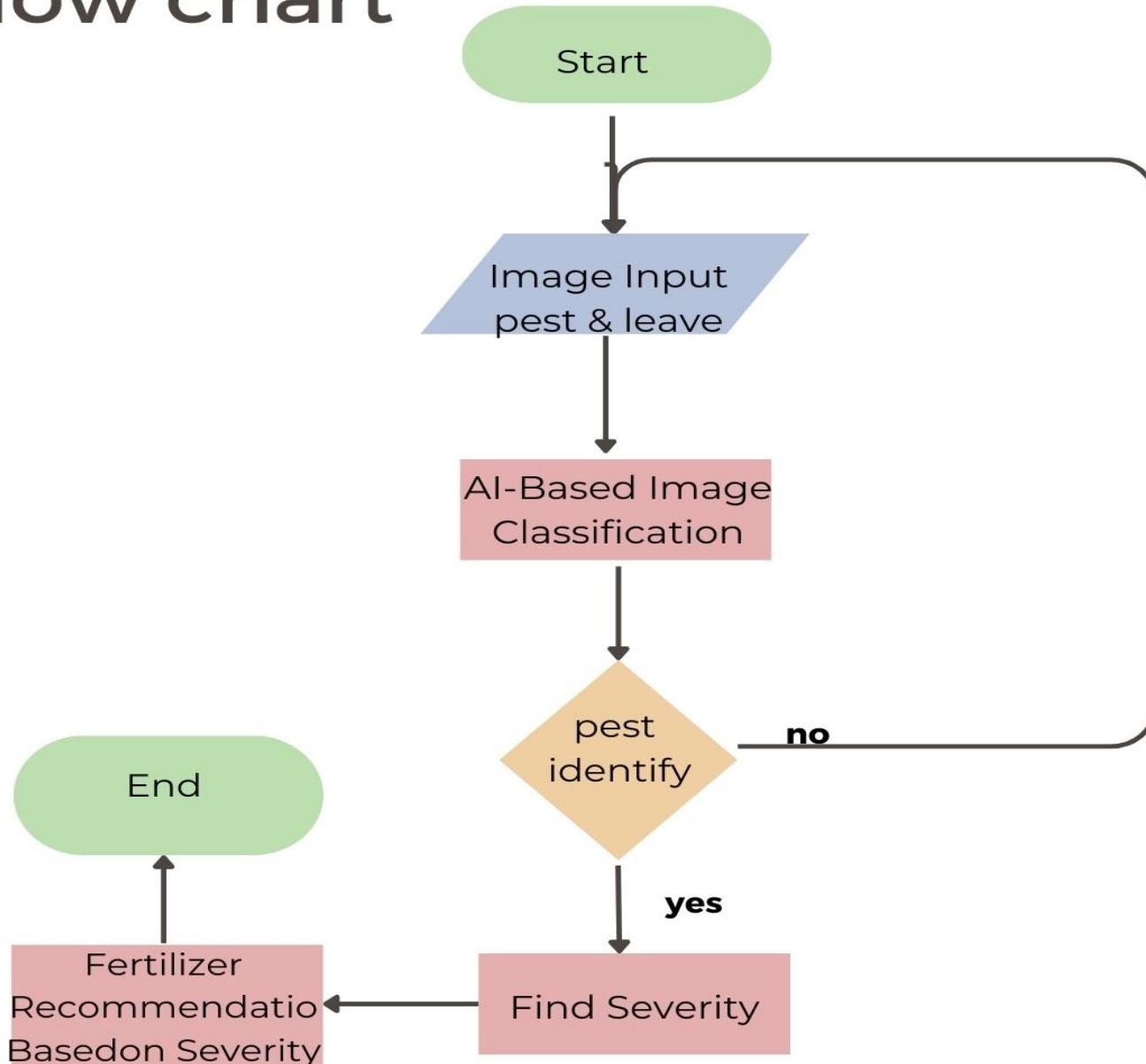
- Phase 1: Data Collection - Gather pest and leaf images from field and datasets.
- Phase 2: Image Processing - Use OpenCV for image preprocessing (resizing, noise removal).
- Phase 3: Model Training - Train ResNet50 for pest classification and severity detection.
- Phase 4: Integration - Connect trained model with a mobile-friendly backend using Python and Firebase.
- Phase 5: Evaluation - Test system accuracy and usability.



System Diagram



Flow chart



System Overview

Mobile app (Flutter) for uploading images and receiving recommendations.



Python backend handles logic and model integration.

Firebase stores images, user data, and model outputs.



System Requirements

Hardware:

- Android/iOS phone with camera
- Server: 8GB RAM, 100GB SSD, Python environment

Software:

- OpenCV for image processing
- TensorFlow with ResNet50
- Firebase for data storage and model output



Key Technologies

Frontend:

Flutter

Image Processing:

OpenCV

Model:

ResNet50 (Pretrained CNN for classification)

Backend:

Python

Database:

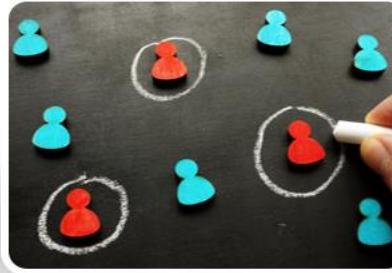
Firebase Realtime DB

ML Approach:

Supervised Learning (Image classification)



Commercialization Strategy



Target Audience:
Paddy farmers, agricultural extension officers



Value Proposition:
Fast, accurate pest detection and tailored fertilizer guidance



Marketing:
Farmer workshops, extension programs, digital outreach



Pricing:
Freemium app with paid tier for detailed analytics and offline features

ResNet50 Model (accuracy-15%-30%)

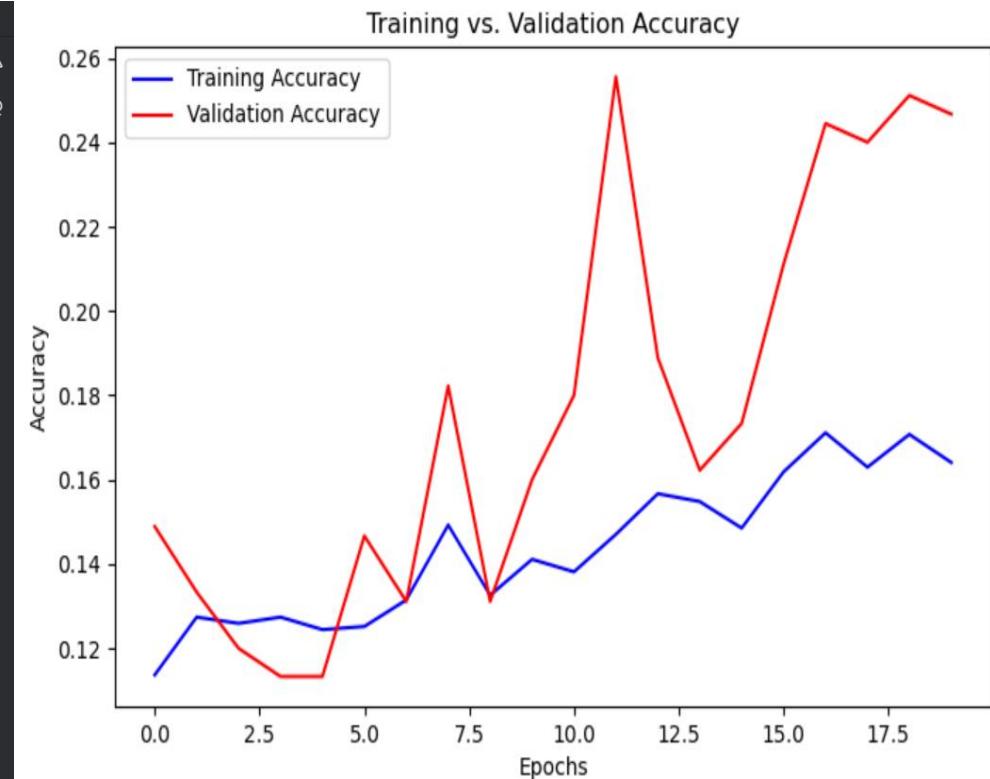
The screenshot shows the PyCharm IDE interface. The project is named 'MyFYP [Pest Identification_Fertilizer Recommendation]'. The 'predict_image.py' file is open, containing the following code:

```
def predict_image(img_path):    # Rescale image    img_array = img_array / 255.0    predictions = pre.model.predict(img_array)    # Get the predicted    predicted_class_idx = np.argmax(predictions, axis=1)[0]    confidence_score = np.max(predictions, axis=1)[0]    return predicted_class_idx, confidence_score
```

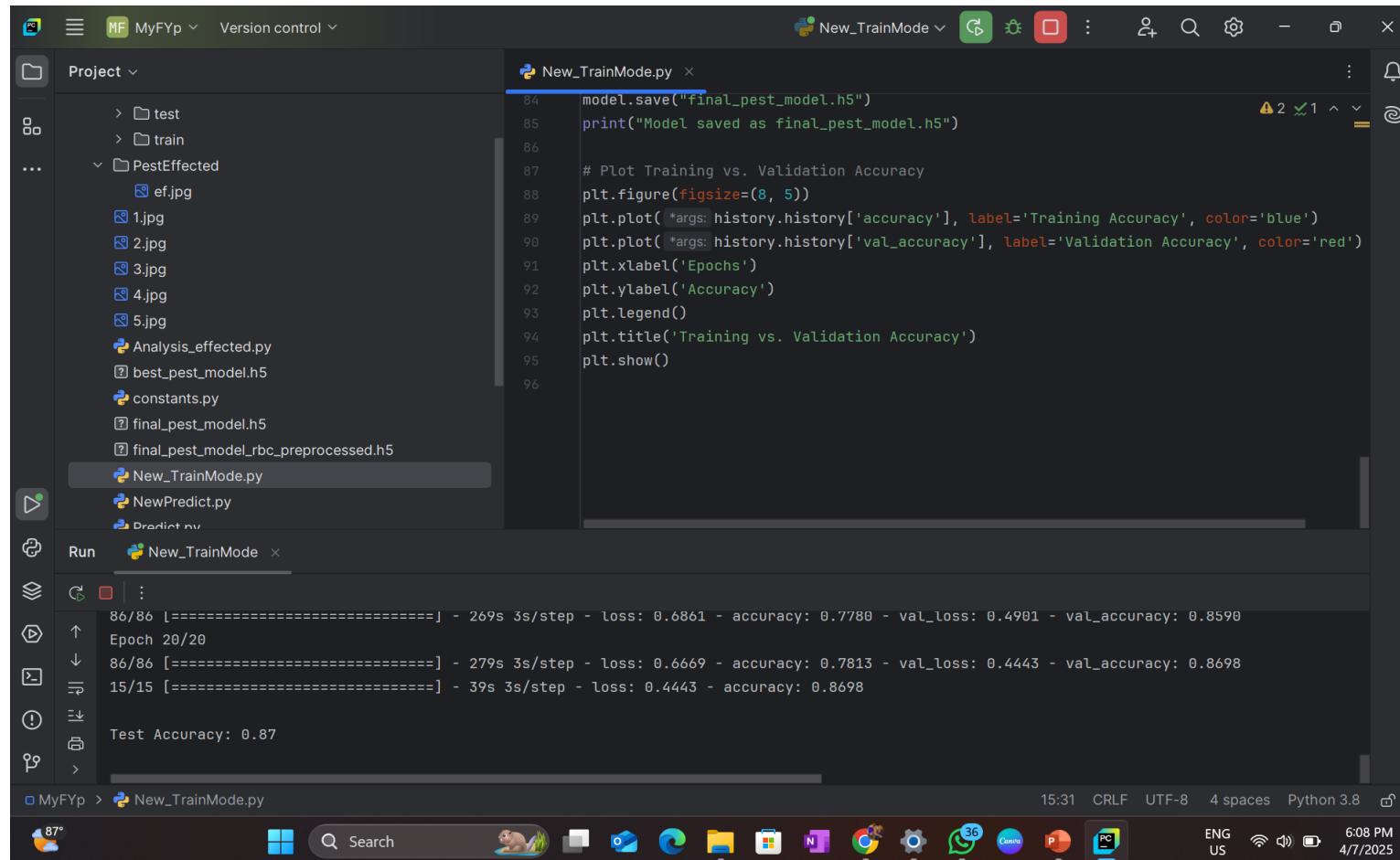
The 'Run' tab shows the command 'TrainMode_Pest' is selected. The terminal output shows:

```
Epoch 27/150  
7/7 [=====] - 10s 1s/step - loss: 2.3801 - accuracy: 0.1000 - val_loss: 2.3628 - val_accuracy: 0.1273  
2/2 [=====] - 2s 827ms/step - loss: 2.3491 - accuracy: 0.1455  
Test Accuracy: 0.15  
Model saved as final_pest_model.h5
```

The status bar at the bottom indicates the time is 18:34, and the Python version is 3.8.



VGG16 Model(accuracy-87%)



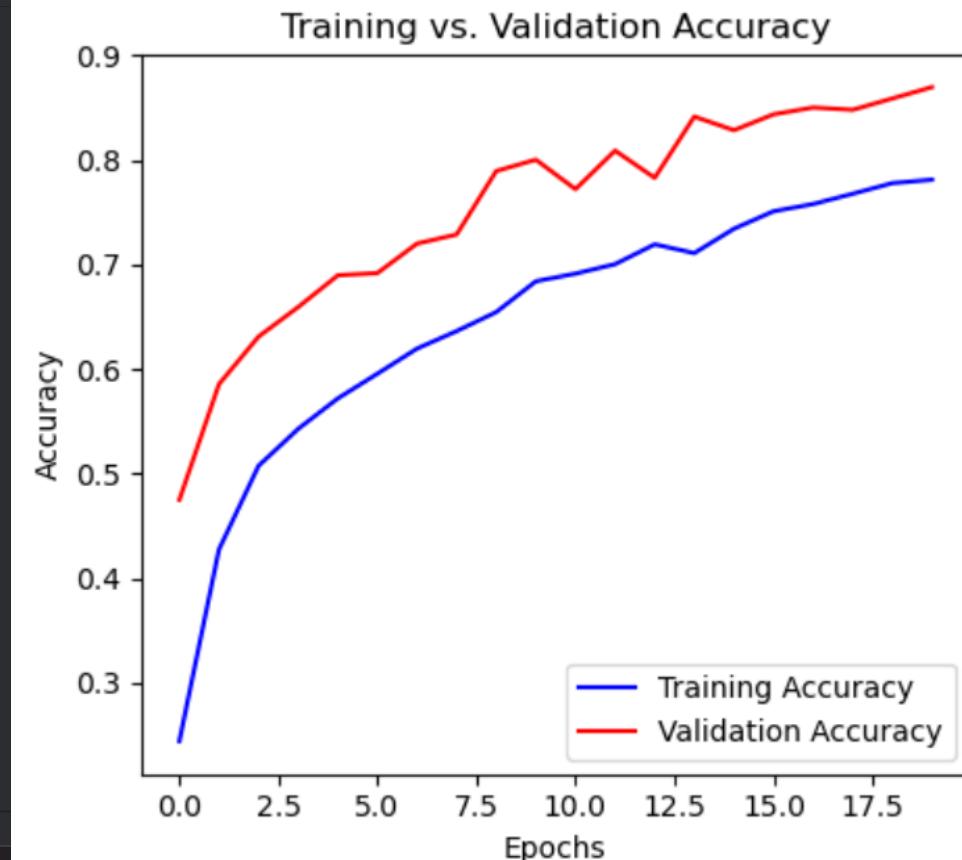
The screenshot shows a code editor window with a Python script named `New_TrainMode.py`. The script includes code to save a model and plot training and validation accuracy. The terminal below shows the execution of the script, displaying training and validation metrics across 20 epochs. The status bar at the bottom indicates the system is running at 87% CPU usage.

```
model.save("final_pest_model.h5")
print("Model saved as final_pest_model.h5")

# Plot Training vs. Validation Accuracy
plt.figure(figsize=(8, 5))
plt.plot(args.history.history['accuracy'], label='Training Accuracy', color='blue')
plt.plot(args.history.history['val_accuracy'], label='Validation Accuracy', color='red')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()
plt.title('Training vs. Validation Accuracy')
plt.show()
```

```
86/86 [=====] - 269s 3s/step - loss: 0.6861 - accuracy: 0.7780 - val_loss: 0.4901 - val_accuracy: 0.8590
Epoch 20/20
86/86 [=====] - 279s 3s/step - loss: 0.6669 - accuracy: 0.7813 - val_loss: 0.4443 - val_accuracy: 0.8698
15/15 [=====] - 39s 3s/step - loss: 0.4443 - accuracy: 0.8698
Test Accuracy: 0.87
```

87%



Pest Identification Result

The screenshot shows a Windows desktop environment with a dark-themed IDE (PyCharm) and a terminal window.

Project Structure:

- MyFYP
- Version control
- Project
- test
- train
- PestEffected
 - ef.jpg
 - 1.jpg
 - 2.jpg
 - 3.jpg
 - 4.jpg
 - 5.jpg
- Analysis_effected.py
- best_pest_model.h5
- constants.py
- final_pest_model.h5
- final_pest_model_rbc_preprocessed.h5
- New_TrainMode.py
- NewPredict.py
- Predict.py

NewPredict.py (Code Editor):

```
1 > import ...
2
3 # Constants
4 MODEL_PATH = "final_pest_model.h5"
5 IMG_PATH = "3.jpg"
6 IMG_SIZE = (224, 224)
7 TEST_PATH = "PestDataset/test"
8
9 # Load model
10 model = tf.keras.models.load_model(MODEL_PATH)
11 print("Model loaded successfully.")
12
13 # Get class names from folder structure
14 class_names = sorted(os.listdir(TEST_PATH))
15 class_names = [folder for folder in class_names if folder != "test"]
16 print(f"Classes detected: {class_names}")
17
18 # Load and preprocess the image
19
20
21
22
```

Terminal Output:

```
2025-04-07 18:11:45.413597: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:176] host
2025-04-07 18:11:45.414074: I tensorflow/core/platform/cpu_feature_guard.cc:151] This Tensor
To enable them in other operations, rebuild TensorFlow with the appropriate compiler flags.
Model loaded successfully.
Classes detected: ['aphids', 'armyworm', 'beetle', 'bollworm', 'grasshopper', 'mites', 'mosquito', 'sawfly', 'stem_borer']
Predicted Class: armyworm
```

Figure 1 (Prediction Window):

Prediction: armyworm



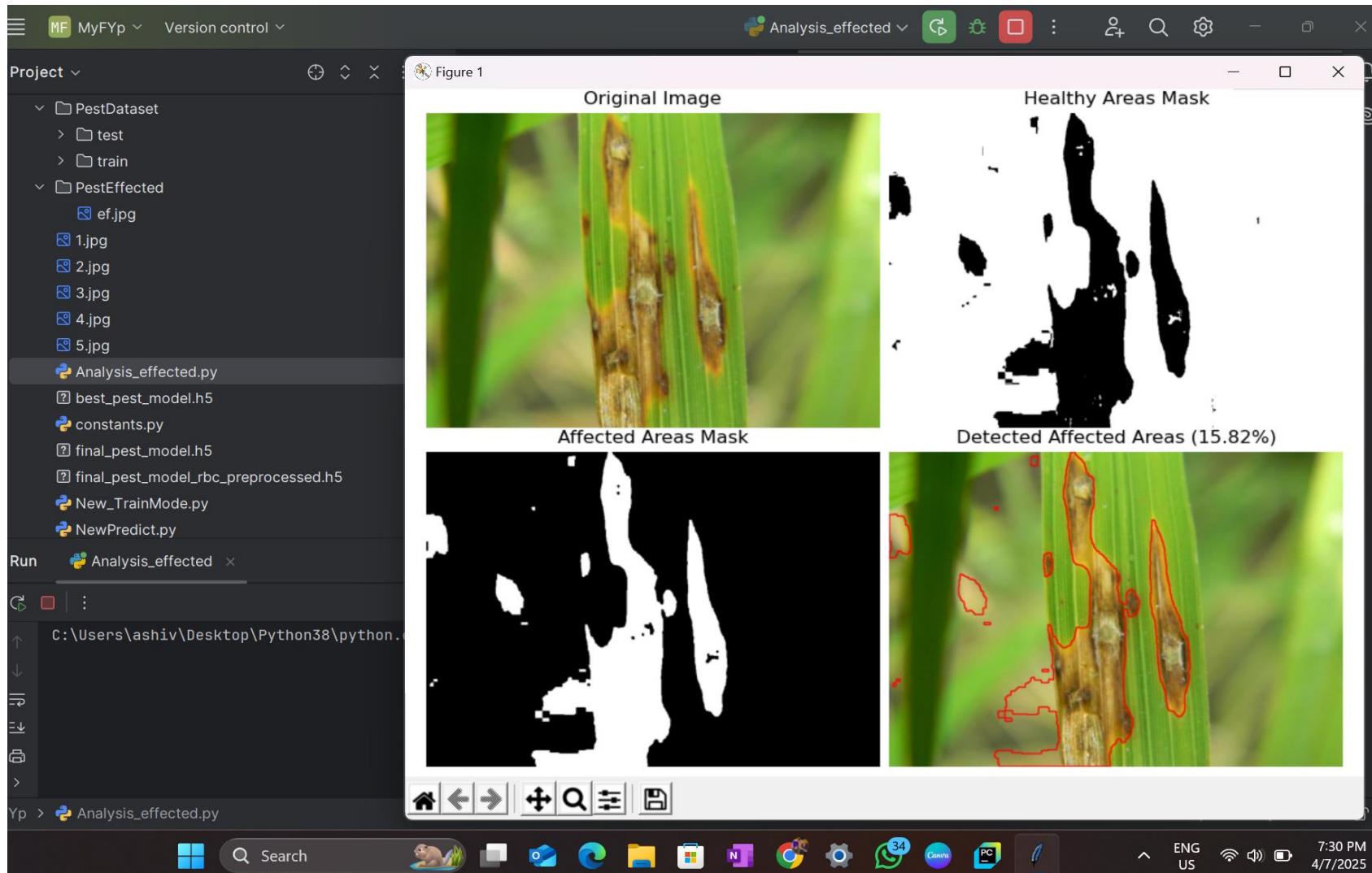
The figure shows a caterpillar with a segmented body, greenish-yellow with brown stripes, crawling on a green plant stem against a blurred background.

Model loaded successfully.

Classes detected: ['aphids', 'armyworm', 'beetle', 'bollworm',

Predicted Class: armyworm)

Effected Leaves Results (with Severity Level)



1. OpenCV preprocessing

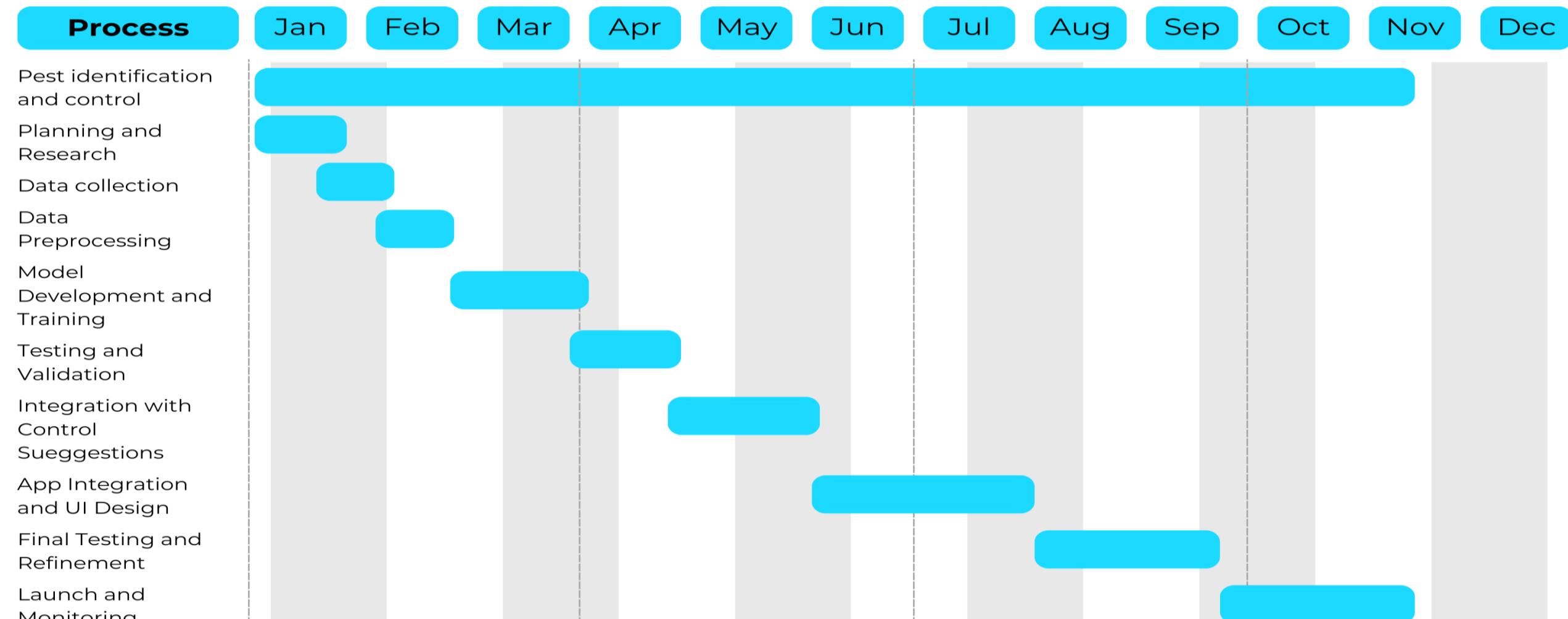
Used to identify and segment affected areas on leaves

2. Calculates affected percentage (mild(<10%) , moderate(10-30%), severe(>30%)

```
C:\Users\ashiv\Desktop\Python38\python
Percentage of affected area: 15.82%
Number of affected spots: 11
Severity: Moderate
```

Gantt Chart

R25-057 | Agri Doc App



References

- Department of Agriculture, Sri Lanka, "RRDI Pest Management Resources," [Online]. Available: [Let's Nurture the Earth for Generations to Come..](#) [Accessed: Apr. 06, 2025].
- International Rice Research Institute (IRRI), "Insects – Pests and Diseases – Step-by-Step Production," [Online]. Available: [Let's Nurture the Earth for Generations to Come..](#) [Accessed: Apr. 06, 2025].
- International Rice Research Institute (IRRI), "Pests and Diseases – Step-by-Step Production," [Online]. Available: [Let's Nurture the Earth for Generations to Come..](#) [Accessed: Apr. 06, 2025].
- S. Kamilaris and F. X. Prenafeta-Boldú, "Deep learning in agriculture: A survey," Computers and Electronics in Agriculture, vol. 147, pp. 70–90, Apr. 2018. doi: 10.1016/j.compag.2018.02.016.
- P. Ferentinos, "Deep learning models for plant disease detection and diagnosis," Computers and Electronics in Agriculture, vol. 145, pp. 311–318, Feb. 2018. doi: 10.1016/j.compag.2018.01.009.
- A. Mohanty, D. P. Hughes, and M. Salathé, "Using deep learning for image-based plant disease detection," Frontiers in Plant Science, vol. 7, p. 1419, Sept. 2016. doi: 10.3389/fpls.2016.01419.



References

- L. Wang, B. Li, Z. Huang, and Z. Liu, "Identification of rice diseases using deep convolutional neural networks," *Neurocomputing*, vol. 267, pp. 378–384, Dec. 2017. doi: 10.1016/j.neucom.2017.06.023.
- H. Brahimi, S. Boukhalfa, and A. Moussaoui, "Deep learning for tomato diseases: Classification and symptoms visualization," *Applied Artificial Intelligence*, vol. 31, no. 4, pp. 299–315, Apr. 2017. doi: 10.1080/08839514.2017.1315516.
- N. Sladojevic, M. Arsenovic, A. Anderla, D. Culibrk, and D. Stefanovic, "Deep neural networks based recognition of plant diseases by leaf image classification," *Computational Intelligence and Neuroscience*, vol. 2016, Article ID 3289801, 2016. doi: 10.1155/2016/3289801.





Thank For
Your Attention

End Presentation

R25-057

