AMERICAN JOURNAL OF MANAGEMENT AND IOT MEDICAL COMPUTING
Peer Reviewed, Referred & Indexed Journal
E-ISSN: 3069-0110
Vol.5, No.2(2) (2026)
www.ajmimc.com

Smart Plant Monitoring System

Kishore Das	Ashish Kumar Palei	Lect. Meenakshi Maharana
Student, (2201298098)	*Student, (2201298035)*	*Lecturer*
Dept. of CSE	*Dept. of CSE*	*Dept. of CSE*
GIFT Autonomous, Bhubaneswar	*GIFT Autonomous, Bhubaneswar*	*GIFT Autonomous, Bhubaneswar*

Abstract**—** *The Smart Plant Monitoring System is designed to improve the health, growth, and safety of plants using modern IoT and artificial intelligence technologies. The system provides real-time monitoring of soil moisture, temperature, humidity, and light intensity using sensor hardware, combined with AI-based plant disease detection through image processing. It helps farmers, plant enthusiasts, and agricultural researchers monitor plant conditions continuously and receive intelligent recommendations for better crop management. The system is developed using a Python Flask backend, PyTorch-based deep learning for disease detection, ESP32-CAM hardware, Claude AI integration, SQLite database, and a glassmorphism HTML/CSS/JS frontend with Chart.js visualizations. Overall, it improves plant health management, reduces manual monitoring effort, and provides a reliable intelligent solution for modern precision agriculture.*

Keywords**—** *Plant Monitoring, IoT, ESP32-CAM, Plant Disease Detection, Deep Learning, Flask, PyTorch, AI Agriculture, Smart Farming, Sensor Integration.*

I. Introduction

In modern agriculture and horticulture, maintaining the health and growth of plants is a primary challenge for farmers, gardeners, and agricultural researchers. Traditional plant monitoring methods mainly depend on manual observation and periodic inspection, which are time-consuming, inefficient, and often unable to detect plant diseases or environmental stress conditions at early stages. The absence of continuous monitoring facilities leads to significant crop losses and reduced agricultural productivity every year [1].

Recent advancements in Internet of Things (IoT) technologies, deep learning, and AI-based image processing have enabled the development of intelligent plant monitoring systems capable of providing continuous environmental sensing and automated disease detection [2]. IoT-based systems allow real-time collection of soil and environmental parameters such as moisture, temperature, humidity, and light intensity. Simultaneously, AI-based image analysis enables early identification of plant diseases before they cause severe damage to crops and vegetation [3].

The proposed system is designed as a centralized smart plant monitoring platform that integrates IoT-based environmental sensing, AI-powered plant disease detection, real-time data visualization, and intelligent agricultural recommendations. The platform continuously collects sensor data from ESP32-CAM hardware, processes plant images using a PyTorch-based deep learning model trained on the PlantVillage dataset with 38 disease categories, and integrates Claude AI for smart recommendations [4].

This paper focuses on the design and implementation of the Smart Plant Monitoring System. Section II discusses the challenges in existing plant monitoring approaches. Section III explains the proposed system architecture. Section IV describes the methodology and technologies used. Section V discusses system design and implementation. Section VI presents results and discussion, while Section VII concludes the paper.

II. Challenges in Existing System

Traditional plant monitoring and disease management systems face several technical and operational challenges due to the absence of real-time digital monitoring platforms. Most agricultural practices still depend on manual observation, periodic soil testing, and physical inspection, which are inefficient and often inaccurate for large-scale crop management [5].

A. Lack of Real-Time Environmental Monitoring

One of the major problems in traditional plant management is the absence of continuous real-time environmental monitoring. Farmers and plant owners generally assess plant conditions based on visual inspection or periodic manual measurements, which cannot capture dynamic changes in soil moisture, temperature, humidity, and light intensity [6]. These environmental parameters change continuously and significantly affect plant health and growth.

The absence of automated sensor-based monitoring reduces operational transparency and creates information gaps. In many cases, plants get damaged due to overwatering, dehydration, or temperature stress before the issue is identified manually.

B. Delayed Disease Detection

Plant diseases spread rapidly and can cause severe damage if not detected at an early stage. In traditional systems, disease identification mainly depends on the expertise of agricultural specialists or manual visual inspection [7]. This process is slow, error-prone, and often results in late diagnosis after the disease has already spread significantly across the crop field.

C. Manual Record Management

Agricultural monitoring involves maintaining detailed records of soil conditions, plant growth, disease history, and

treatment records. In traditional systems, these records are managed manually through paper registers or simple spreadsheets [8]. Manual record handling is time-consuming, prone to errors, and difficult to analyze for identifying long-term patterns in plant health.

D. Absence of Intelligent Recommendations

Traditional plant monitoring systems do not provide intelligent, AI-driven recommendations for plant care and disease management. Farmers often lack access to expert agricultural knowledge, especially in remote areas [9]. Without intelligent guidance systems, plant owners may apply incorrect treatments or inappropriate watering schedules that further damage plant health.

E. Need for an Intelligent Plant Monitoring Platform

The limitations of traditional systems highlight the need for a modern platform capable of integrating real-time environmental sensing, automated disease detection, historical data visualization, and AI-driven recommendations into a single unified system [10]. A smart plant monitoring platform can significantly improve plant health management and enable early disease intervention.

III. Proposed System

The proposed Smart Plant Monitoring System is designed to provide a comprehensive and intelligent solution for monitoring plant health, detecting diseases, analyzing environmental conditions, and delivering AI-powered agricultural recommendations in real time. The system addresses the limitations of traditional plant monitoring by integrating IoT-based sensor technology, deep learning-based disease detection, Claude AI integration, and a responsive web-based interface into a unified platform.

The system continuously collects environmental data including soil moisture, temperature, humidity, and light intensity using sensor hardware connected to the ESP32-CAM module. The collected data is transmitted to a Python Flask backend server, stored in a SQLite database, and visualized on the frontend using Chart.js for real-time trend analysis.

A. System Overview

The Smart Plant Monitoring System follows a client-server architecture where the frontend, Flask backend, SQLite database, and IoT hardware work together to provide comprehensive plant health monitoring. The frontend is developed using HTML, CSS with glassmorphism styling, and JavaScript, communicating with the backend through RESTful APIs.

The ESP32-CAM module continuously captures plant images and transmits sensor data to the Flask backend through Wi-Fi. The backend processes sensor readings, stores them in the SQLite database, and passes plant images to the PyTorch disease detection model. Claude AI integration provides intelligent recommendations based on detected diseases and sensor readings.

B. Sensor Monitoring Module

The sensor monitoring module is the data collection layer of the system. The ESP32-CAM hardware is equipped with sensors to measure soil moisture levels, ambient temperature, relative humidity, and light intensity in real time. This module continuously transmits sensor readings to the Flask backend through HTTP communication over Wi-Fi. Threshold-based alerts notify users when sensor values exceed safe ranges for optimal plant growth.

C. Disease Detection Module

The disease detection module uses a PyTorch-based convolutional neural network trained on the PlantVillage dataset, which contains thousands of plant images across 38 disease and healthy categories. When a plant image is submitted, the backend passes it to the disease detection model for analysis. The model returns the most probable disease category along with confidence scores, and Claude AI provides additional expert-level recommendations.

D. AI Recommendation Module

The AI recommendation module integrates Claude AI to provide intelligent, context-aware agricultural advice based on detected plant diseases and current environmental sensor readings. When a disease is detected or abnormal sensor values are recorded, the system sends relevant data to the Claude AI API, which generates customized recommendations for plant treatment, fertilization, watering schedules, and environmental corrections. A rule-based fallback mechanism ensures recommendations are provided even when the AI API is unavailable.

E. Working Flow of the System

The working process begins when the ESP32-CAM hardware continuously sends sensor readings and plant images to the Flask backend through Wi-Fi. The backend processes incoming data, stores environmental readings in the SQLite database, and passes plant images to the disease detection model. The frontend periodically fetches updated sensor data through REST APIs and dynamically updates Chart.js visualizations. When users submit a plant image, the system displays the predicted disease category, confidence score, and Claude AI generated recommendations.

IV. Methodology

The development of the Smart Plant Monitoring System follows a structured and modular approach to ensure efficient implementation, smooth IoT integration, and intelligent plant health management. The complete development process uses modern web technologies, deep learning frameworks, and IoT hardware, following the Agile software development methodology for iterative improvements and continuous testing.

A. Requirement Analysis

Requirement analysis is the foundational phase of system development. The functional requirements include real-time sensor monitoring, plant disease detection, AI-based

recommendations, historical data visualization, alert management, and centralized dashboard monitoring. Non-functional requirements focus on system responsiveness, user-friendly interface design, model accuracy, data reliability, and operational scalability.

B. Hardware Setup and IoT Integration

The ESP32-CAM module is configured to function as both the image capture unit and the IoT sensor data transmitter. The module is integrated with soil moisture sensors, DHT11 temperature and humidity sensors, and light-dependent resistors (LDR) for light intensity measurement. All sensor readings are collected by the ESP32-CAM firmware and transmitted to the Flask backend through HTTP POST requests over Wi-Fi connectivity.

C. Disease Detection Model Training

The plant disease detection model is developed using PyTorch and trained on the PlantVillage dataset, which contains labeled plant images across 38 categories covering common diseases and healthy conditions for various crop types. A convolutional neural network architecture is used for image classification, with transfer learning techniques applied to improve model accuracy and reduce training time [11]. Data augmentation techniques such as random cropping, horizontal flipping, and color jitter are applied during training to improve model generalization.

D. Backend Development

The backend of the system is developed using Python Flask, which acts as the core processing unit of the platform. It handles API requests, sensor data processing, database operations, disease detection inference, and Claude AI communication [12]. SQLite is used as the database for storing sensor readings, detection history, and plant records efficiently.

E. Frontend Development

The frontend is developed using HTML5, CSS3 with glassmorphism design principles, and JavaScript. Chart.js is integrated for real-time visualization of sensor data trends including soil moisture, temperature, humidity, and light intensity over time [13]. The user interface is designed to be visually appealing, intuitive, and responsive across different screen sizes and devices.

F. System Testing

System testing is performed to ensure proper functionality of all modules including sensor data collection, disease detection accuracy, API communication, Claude AI integration, and frontend visualization. The system is tested under different plant conditions and disease scenarios to verify stable performance and reliable detection accuracy across different crop types.

V. System Design and Implementation

The Smart Plant Monitoring System is implemented using a modular client-server architecture that integrates IoT hardware, deep learning inference, AI advisory, and real-time data visualization into a unified platform. Each component is developed and integrated carefully to provide seamless plant health monitoring and disease management capabilities.

A. System Architecture

The overall architecture consists of the ESP32-CAM IoT hardware layer, Python Flask backend layer, SQLite database layer, PyTorch disease detection model, Claude AI integration layer, and the glassmorphism frontend layer. Each layer performs specific functionalities while maintaining continuous communication with other system components through RESTful API interfaces.

B. Real-Time Sensor Monitoring

The real-time sensor monitoring module continuously tracks soil moisture, temperature, humidity, and light intensity through the ESP32-CAM sensor integration. Sensor readings are transmitted to the Flask backend at regular intervals and stored in the SQLite database with timestamps for historical analysis. The frontend displays live sensor values with color-coded indicators showing whether each parameter is within the optimal range for plant growth.

C. Plant Disease Detection Implementation

The disease detection module processes plant images through the trained PyTorch convolutional neural network model. Images submitted through the frontend upload interface or captured by the ESP32-CAM are preprocessed, resized, and normalized before being passed to the model for inference. The model returns the predicted disease category along with confidence scores for all 38 classes. All detection results are stored in the database for historical reference and trend monitoring.

D. Claude AI Integration

Claude AI is integrated into the system to provide expert-level agricultural recommendations based on detected plant diseases and environmental sensor readings. When a disease is identified or critical sensor threshold values are recorded, the system formulates a contextual prompt combining the detection result and current sensor data, and queries the Claude AI API for tailored recommendations covering treatment, fertilization, watering adjustments, and environmental corrections.

E. Admin Dashboard Implementation

The web-based dashboard serves as the central monitoring and management interface. Users can monitor live sensor readings, view historical trend charts, submit plant images for disease analysis, review AI recommendations, and access detection history through a unified glassmorphism-styled interface. All plant health data is stored digitally in the SQLite database, enabling easy access to historical records and data-driven agricultural decision-making.

F. Advantages of the Proposed System

The Smart Plant Monitoring System provides significant advantages over traditional manual plant monitoring approaches. The integration of real-time IoT sensing, AI-

powered disease detection, Claude AI recommendations, and interactive data visualization improves plant health management efficiency and agricultural productivity. The system reduces manual monitoring effort while enabling timely disease intervention before significant crop damage occurs.

Traditional System	Smart Plant Monitoring System
Manual visual inspection only	Real-time IoT-based sensor monitoring
No automated disease detection	AI-powered disease detection (38 classes)
No intelligent advisory	Claude AI-driven recommendations
Paper-based record management	Digital SQLite database management
No data visualization	Live Chart.js sensor trend visualization
Delayed disease intervention	Early detection with confidence scoring
High manual monitoring effort	Automated continuous IoT monitoring
No environmental threshold alerts	Automated alert system for critical values

Table 1. Comparison of Traditional System and Smart Plant Monitoring System

VI. Results and Discussion

The Smart Plant Monitoring System was successfully implemented and tested for real-time plant health monitoring, disease detection, and intelligent agricultural advisory. Different modules including IoT sensor monitoring, PyTorch-based disease detection, Claude AI recommendations, and Chart.js data visualization were tested under various plant conditions and disease scenarios to evaluate system performance and operational reliability.

A. System Performance Analysis

The overall system performance was analyzed based on sensor data responsiveness, API response time, disease detection inference speed, and database operations. The Flask backend maintained stable performance during continuous sensor data processing and concurrent disease detection requests. SQLite provided fast storage and retrieval of sensor records and detection history. The frontend interface remained responsive during real-time data updates and Chart.js rendering operations.

B. Disease Detection Accuracy

The PyTorch-based disease detection model demonstrated high accuracy during implementation testing across the 38 PlantVillage disease categories. The model successfully identified common plant diseases including bacterial blight, leaf rust, powdery mildew, and early blight with high confidence scores. Data augmentation techniques applied during training improved model generalization and detection robustness across images captured under different lighting conditions and angles.

C. Sensor Monitoring Efficiency

The IoT-based sensor monitoring module demonstrated accurate and continuous data collection during testing. The ESP32-CAM hardware successfully transmitted soil moisture, temperature, humidity, and light intensity readings to the Flask backend at regular intervals with minimal packet loss. Chart.js-based real-time visualization provided clear and intuitive trend analysis of sensor parameters over time.

D. AI Recommendation Quality

The Claude AI integration demonstrated effective generation of contextual agricultural recommendations based on detected diseases and sensor readings. Recommendations covered treatment protocols, watering adjustments, environmental corrections, and preventive care measures tailored to specific disease conditions. The rule-based fallback mechanism provided consistent basic recommendations when the AI API was unavailable.

E. User Experience and System Reliability

The glassmorphism-styled frontend provided a visually appealing and intuitive user experience for plant health monitoring and disease detection. Users were able to navigate between sensor monitoring, disease detection, and recommendation modules smoothly and efficiently. The system demonstrated stable performance across extended testing periods with consistent detection accuracy and sensor data reliability.

VII. Future Enhancements

The Smart Plant Monitoring System provides a strong foundation for developing more advanced and intelligent agricultural monitoring platforms in the future. Several additional features can further improve system intelligence, scalability, and user accessibility.

A. Mobile Application Integration

Currently, the platform is implemented as a web-based application. Future development can include dedicated Android and iOS mobile applications for easier accessibility and real-time push notifications, allowing farmers to receive instant alerts regarding disease detections, critical sensor threshold breaches, and AI recommendations directly on their smartphones.

B. Multi-Plant and Multi-Node Scalability

Future enhancements can extend the system to support multiple plant monitoring nodes across large agricultural fields or greenhouse environments. A centralized dashboard can aggregate sensor data and detection results from multiple ESP32-CAM nodes simultaneously, providing a comprehensive view of overall crop health across large cultivation areas.

C. Predictive Analytics and Growth Modeling

Machine learning-based predictive analytics can be integrated to forecast plant growth trajectories, disease outbreak probabilities, and seasonal crop performance based on historical sensor data. Growth modeling algorithms can

help farmers plan irrigation schedules, fertilization cycles, and harvesting timelines more effectively.

D. Cloud Deployment and Remote Access

Future implementation can include cloud deployment using platforms such as AWS or Google Cloud to improve system scalability, remote accessibility, and data storage capacity. Cloud-based infrastructure will support large-scale agricultural deployments involving multiple farms, diverse crop types, and real-time data aggregation across geographically distributed locations.

E. Integration with Agricultural Drones

Integration with agricultural drones equipped with multispectral cameras can extend the disease detection capabilities to large-scale open fields. Drone-captured images can be processed through the PyTorch disease detection model for aerial crop health assessment, enabling early detection of disease outbreaks across extensive cultivation areas.

VIII. Conclusion

The Smart Plant Monitoring System is an intelligent and centralized IoT-based plant health monitoring platform developed to improve agricultural efficiency, enable early disease detection, and provide AI-powered recommendations for modern plant care management. The system integrates real-time IoT sensing through ESP32-CAM hardware, deep learning-based plant disease detection using PyTorch and the PlantVillage dataset, Claude AI advisory integration, SQLite database management, and a glassmorphism HTML/CSS/JS frontend with Chart.js visualizations.

The implementation of modern IoT technologies and AI-driven intelligence improves plant health transparency, reduces manual monitoring effort, and enables timely disease intervention before significant crop damage occurs. The system helps farmers, agricultural researchers, and plant enthusiasts monitor plant conditions continuously and receive intelligent guidance for better crop management.

Overall, the Smart Plant Monitoring System provides a reliable, scalable, and user-friendly solution for modern precision agriculture, with strong future scope for mobile application integration, multi-node scalability, predictive analytics, cloud deployment, and drone-based aerial monitoring.

References

[1] P. Sharma and R. Kumar, "Challenges in Traditional Plant Disease Management and Monitoring Systems," International Journal of Agricultural Science and Research, vol. 10, no. 3, pp. 45–52, 2021.

[2] A. Singh and M. Patel, "IoT-Based Smart Plant Monitoring Systems: A Survey," IEEE Internet of Things Journal, vol. 8, no. 5, pp. 3210–3218, 2021.

[3] K. Verma and S. Das, "Deep Learning Approaches for Plant Disease Detection Using Convolutional Neural Networks," International Journal of Computer Vision and Image Processing, vol. 11, no. 2, pp. 78–89, 2021.

[4] Anthropic, Claude AI Documentation. [Online]. Available: https://docs.anthropic.com/

[5] T. Brown and A. Wilson, "Limitations of Manual Crop Monitoring Approaches in Precision Agriculture," International Journal of Agricultural Engineering, vol. 9, no. 2, pp. 110–118, 2020.

[6] D. Mishra and P. Roy, "Real-Time Environmental Sensing for Smart Agriculture Using IoT Technologies," IEEE Sensors Journal, vol. 21, no. 14, pp. 15680–15690, 2021.

[7] R. Mehta and S. Shah, "Automated Plant Disease Detection Using Machine Learning: Challenges and Opportunities," Journal of Agricultural Informatics, vol. 12, no. 1, pp. 33–44, 2021.

[8] K. Rao and V. Kumar, "Digital Record Management Systems for Agricultural Monitoring Applications," International Journal of Computer Applications, vol. 180, no. 12, pp. 22–28, 2020.

[9] J. Lee and H. Park, "AI-Driven Advisory Systems for Smart Farming and Crop Management," Computers and Electronics in Agriculture, vol. 185, p. 106142, 2021.

[10] S. Patel and A. Gupta, "Integrated IoT and AI Platforms for Modern Precision Agriculture," IEEE Access, vol. 9, pp. 66320–66335, 2021.

[11] PyTorch Documentation, Meta AI. [Online]. Available: https://pytorch.org/docs/

[12] Flask Documentation, Pallets Projects. [Online]. Available: https://flask.palletsprojects.com/

[13] Chart.js Documentation. [Online]. Available: https://www.chartjs.org/docs/

[14] PlantVillage Dataset, Penn State University. [Online]. Available: https://plantvillage.psu.edu/

[15] SQLite Documentation. [Online]. Available: https://www.sqlite.org/docs.html