IoT Based Farm Controlling and Monitoring Using Android Application

Aishwrya Unde, Disha Sable, Nikita Dige and Prof. G.B. Bodke

Department of Artificial Intelligence and Machine learning Engineering, Samarth College of Engineering Belhe, Pune, India
Email: aishuunde07@gmail.com

Manuscript Details
Received :20.01.2026
Accepted: 26.02.2026
Published: 28.02.2026

Available online on https://www.irjse.in
ISSN: 2322-0015

Cite this article as:
Aishwrya Unde, Disha Sable, Nikita Dige and Prof. G.B. Bodke. IoT Based Farm Controlling and Monitoring Using Android Application, *Int. Res. Journal of Science & Engineering*, 2026, Volume 14(1): 9-17.
https://doi.org/10.5281/zenodo.19634205

Abstract

Farming today is caught between water shortages, unpredictable weather, and the constant threat of animals damaging crops. Most small farmers still rely on manual checks and traditional irrigation, which wastes water and delays responses. To tackle these issues, we built an IoT-driven system that keeps an eye on the farm continuously and lets farmers control things from their phones. The heart of the system is an ESP32 microcontroller connected to soil moisture, temperature, humidity, and ultrasonic sensors. It automatically waters the crops when the soil gets too dry, saving water while keeping plants healthy. A camera with a machine learning model spots animals that wander in, setting off a buzzer and sending an instant alert to the farmers Android phone. Our model have accuracy between 80-96%. All sensor data goes to the cloud, so farmers can see live readings, turn the water pump on or off remotely (30-35% water Saving). The app also uses NLP farmers can just type or speak their questions, and the system gives advice. We added government policy updates too, because many farmers miss out on schemes simply because they don't know about them. It is affordable, scalable, and designed with real farmer needs in mind.

Keywords: IoT, smart agriculture, ESP32 microcontroller, automated irrigation, soil moisture monitoring, real-time sensing, AI-based animal detection, Android application, cloud integration, precision farming.

1. Introduction

Agriculture remains the primary livelihood source in many developing nations; however, the sector is increasingly strained by erratic climate patterns, diminishing water resources, labor shortages, and the inefficiencies of traditional farming methods [1, 2]. In many rural areas, irrigation decisions are still made manually—farmers physically assess soil conditions and operate valves based on visual observation. This approach frequently leads to either overwatering or underwatering and results in delayed responses to sudden threats such as animal intrusions or weather changes.

The Internet of Things (IoT) offers a transformative solution by enabling real-time monitoring and automation through wireless sensors and embedded systems [3]. Sensors can continuously measure soil moisture, ambient temperature, humidity, and other critical parameters, replacing guesswork with precise data. Concurrently, artificial intelligence (AI) has enabled advanced farm security features, such as automatic animal detection and disease diagnosis [4]. Given the increasing penetration of smartphones even in rural regions, a mobile application provides an ideal interface for delivering these technological benefits directly to farmers.

This study proposes an integrated IoT-based smart farm monitoring system that collects real-time environmental data using soil moisture, temperature, humidity, and ultrasonic sensors interfaced with an ESP32 microcontroller. The system implements threshold-based automated irrigation control to optimize water usage and an intelligent animal detection and alert system using camera-based monitoring and AI processing, enabling early intrusion detection and reducing crop damage through instant notifications [5].

2. System Design and Hardware

2.1 Hardware Components

The system's hardware was chosen to balance performance, cost, and reliability in an outdoor farming environment.

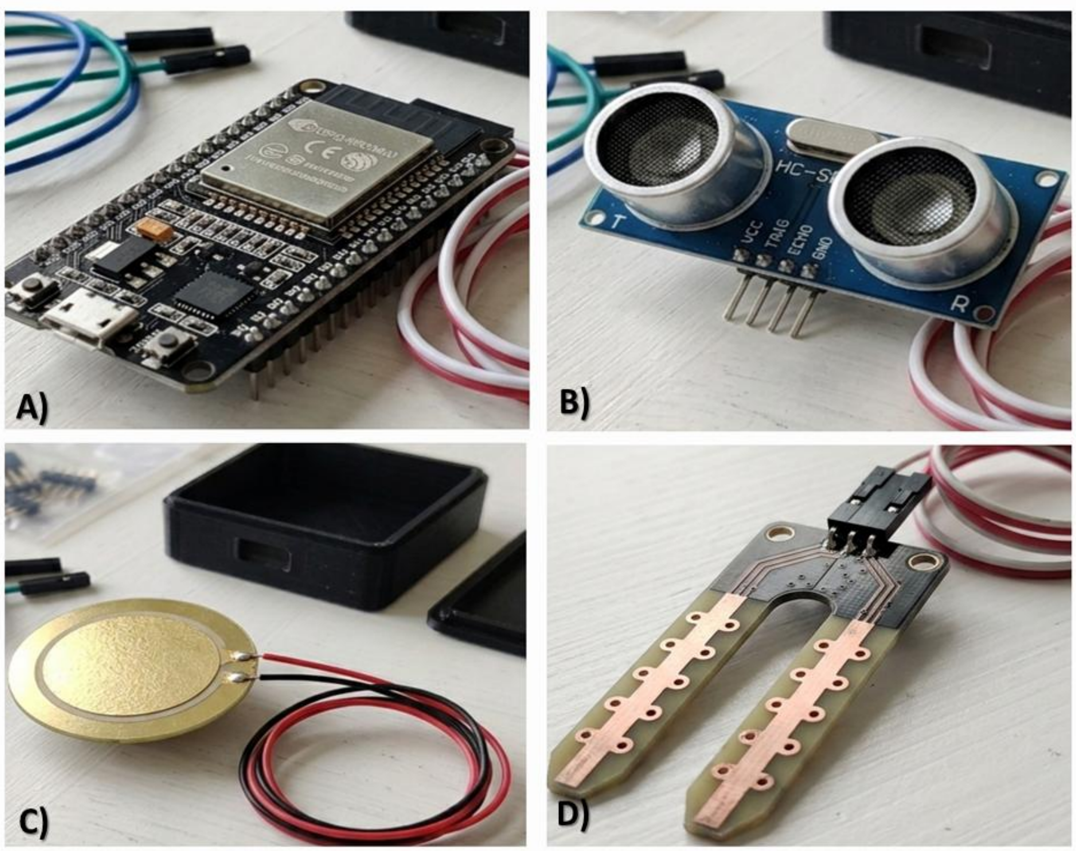

Fig. 1. Main Hardware components used in the system:
(A) ESP32 microcontroller, **(B)** Ultrasonic sensor, **(C)** piezo buzzer, **(D)** soil moisture sensor.

Table 1: Specifications and functional description of hardware components used in the proposed IoT-based smart farming system.

Components	Description
DHT22 Sensor	Reads temperature and humidity values
Soil Moisture Sensor	Measures soil moisture level
Relay Module (Pump)	Controls water pump ON/OFF
Relay Module (Humidifier)	Controls humidifier system
Ultrasonic Sensor	Send and Receive Ultrasonic signals
Piezo Buzzer	Generates alert sound

ESP32 Microcontroller acts as the brain. It has built-in Wi-Fi and Bluetooth, which lets it talk to the Android app directly. Its low power consumption means it can run on a battery or a small solar panel, important for farms where electricity may be unreliable.

Soil Moisture Sensor does exactly what its name suggests it measures how wet the soil is. We placed it near the roots of the crops. When the soil dries out below a set level, the sensor tells the ESP32, and the system decides whether to start watering.

Ultrasonic Sensor is our first line of defense against animals. It sends out sound waves and measures how long they take to bounce back. If something large comes too close to the farm, the sensor picks it up immediately.

DHT22 Temperature and Humidity Sensor tracks the air around the crops. Both temperature and humidity affect plant health, and this data helps farmers understand broader conditions.

Piezo Buzzer provides an audible scare when an animal is detected. It's loud enough to startle most intruders, and it also alerts anyone nearby that something is happening.

Relay Module is a small but critical part. The ESP32 cannot directly switch a water pump that would draw too much current. The relay acts as a safe middleman: the ESP32 sends a tiny signal, and the relay turns the pump on or off.

Water Pump (DC Motor) pulls water from a tank or a well. The system can run it automatically based on soil moisture, or the farmer can turn it on manually from the app.

Power Supply consists of a rechargeable battery and a power adapter. The battery keeps the system running during outages, and the adapter keeps the battery charged when mains power is available.
Connecting Wires and a PCB tie everything together. We laid them out carefully to avoid loose connections, which can be a real problem in dusty, damp farm conditions.

2.2 How They All Work Together

All these components are wired to the ESP32, which runs a simple but effective control loop. The soil moisture, ultrasonic, and DHT22 sensors are read every few seconds. If the soil moisture falls below a threshold (say, 40%), the ESP32 triggers the relay, and the water pump starts. The pump keeps running until moisture reaches a comfortable level (e.g., 70%), then it shuts off.

Meanwhile, the ultrasonic sensor constantly checks distances. If it detects an object within a preset range meaning an animal is likely approaching the ESP32 turns on the buzzer and sends an alert to the cloud. The farmer gets a push notification on the Android app right away.

All sensor readings are also sent to the cloud every minute. That way, the farmer can open the app anytime and see what's happening, no matter where they are.

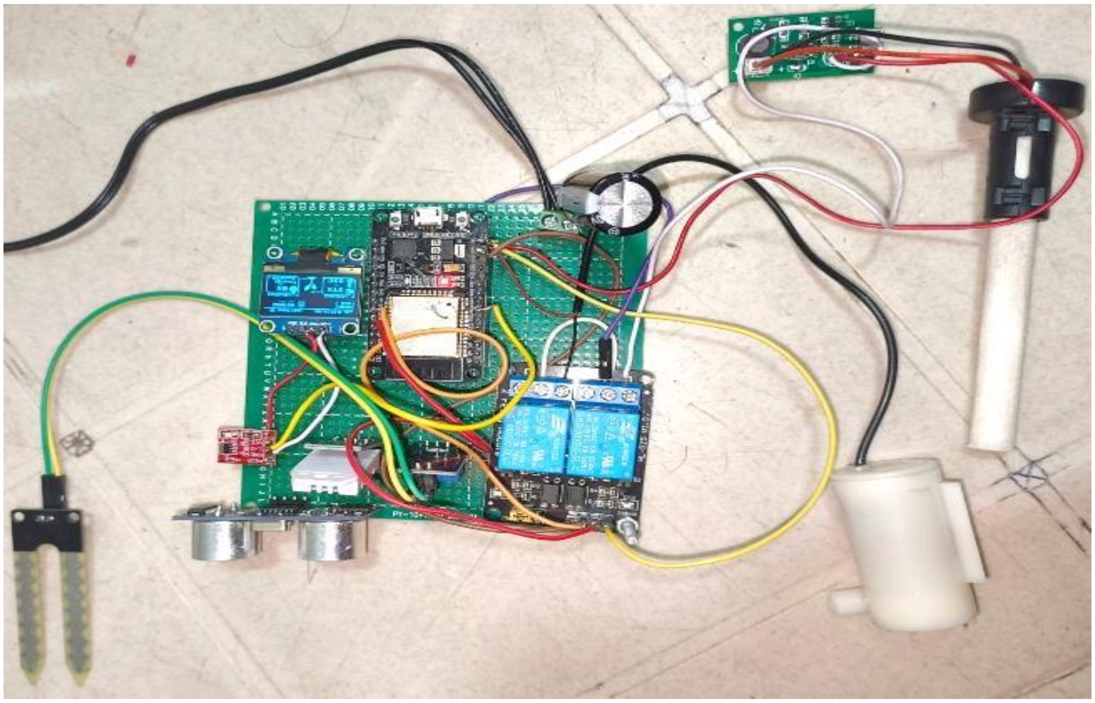

Fig. 2. Performance evaluation of the proposed system based on key operational parameters including sensor accuracy, response time, and data transmission efficiency.

2. System Methodology

The system uses a structured method that includes sensing, processing, making decisions, and taking action. At first, sensors connected to an ESP32 microcontroller gather information about the environment, like soil moisture, temperature, humidity, and distance.

The data that has been collected is processed on-site and sent to the cloud database for storage and monitoring. We use artificial intelligence to look at pictures of crops and find animals in the field. The system automatically controls irrigation motors and sends alerts to farmers when something isn't right based on preset threshold values.

Threshold-based logic is applied to automate irrigation and alert operations. The soil moisture value is continuously compared with predefined threshold limits. If the moisture level falls below the minimum threshold, the irrigation motor is automatically activated. When the moisture level exceeds the maximum threshold, the motor is turned off to prevent overwatering. Similarly, the ultrasonic sensor monitors nearby distance values, and if an object is detected within a specified limit, an alarm signal is generated to alert the farmer. Threshold=20.

If Soil Moisture < Minimum Threshold

Motor = ON

If Soil Moisture > Maximum Threshold

Motor = OFF

To improve monitoring, artificial intelligence techniques are used. The camera module takes pictures of crops and the area around them. Trained AI models look at these pictures to find crop diseases and see if animals have gotten into the area. The results of the detection are processed, and the right actions are taken, like sending out alerts or showing diagnostic suggestions.

Detection Confidence Threshold = 0.40 (40%)

Step 1: Start the system.

Step 2: Initialize camera module.

Step 3: Connect to Wi-Fi network.

Step 4: Load trained animal detection model.

Step 5: Capture image frame from camera.
Step 6: Preprocess the captured image.
- Resize image
- Normalize pixel values

Step 7: Send image to AI detection model.
Step 8: Perform object detection.
Step 9: Check detection confidence.
IF confidence ≥ threshold value
Animal detected
ELSE
No animal detected
Step 10: If animal detected:
- Trigger buzzer alarm
- Send alert notification
- Store detection log

Step 11: If no animal detected:
- Continue monitoring

Step 12: Repeat Steps 5 to 11 continuously.
Step 13: Stop system.

3. Android Application

The companion Android application (compatible with Android 7.0 and higher) is designed for intuitive user interaction. The home screen displays current soil moisture, temperature, and humidity values prominently. A manual override button allows the farmer to turn the water pump on or off independently of the automatic logic. A dedicated section lists recent alerts including animal detection events and pump activity. A graphical view enables visualization of soil moisture trends over daily or weekly periods.

The application includes a natural language interface powered by a cloud-based pretrained multimodal AI model. Farmers can type or speak queries such as "What is the current temperature?" or "Should I fertilize this week?" The system interprets the query and responds with relevant data or agronomic advice [3]. Additionally, the app integrates a section displaying updated information on government agricultural schemes, subsidies, and programs**.**

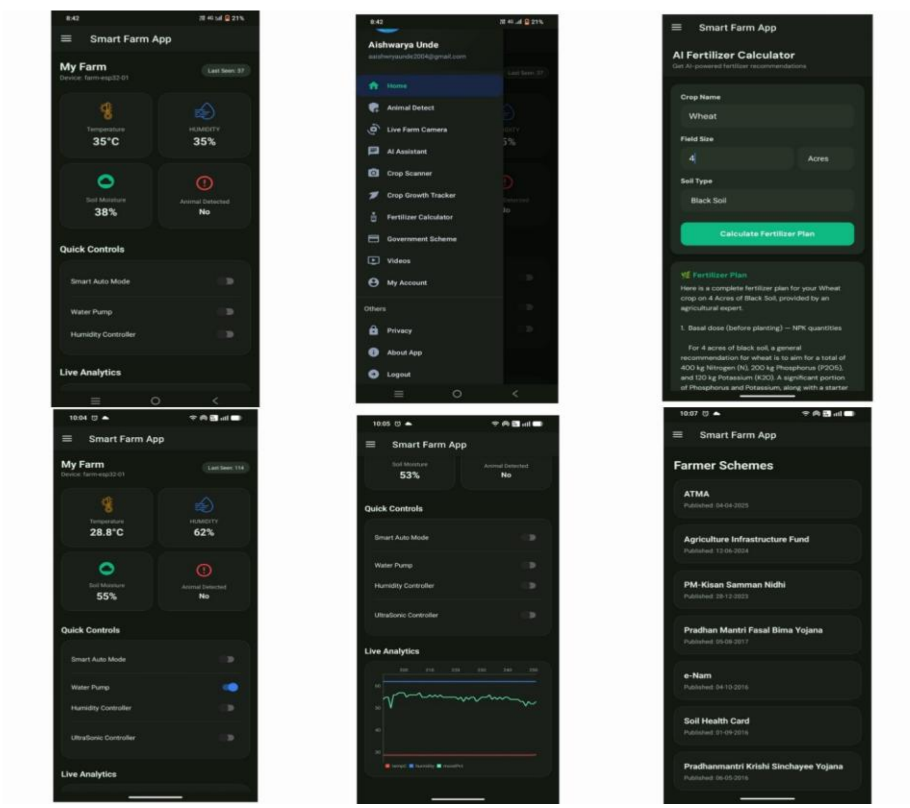

Fig 3: Screenshots of the Android Application showing dashboard, navigation interface, sensor data display, AI module, Farm monitoring, and control features

The app also includes a natural language interface. Instead of navigating menus, a farmer can type or speak something like "What's the temperature right now?" or "Should I fertilize this week?" The system uses a cloud-based pretrained multimodal AI model(Gemini) to understand the question and respond with relevant data or advice drawn from an embedded knowledge base.

Finally, we integrated a section for government schemes. Many government programs offer subsidies, seeds, or equipment, but farmers often learn about them too late. The app pulls the latest updates from official sources and presents them in simple language.

4. System Architecture

The proposed system adopts a four-layer modular architecture [2][3]:

1. Top Layer (Sensor Layer) – physical sensors (soil moisture, ultrasonic, DHT22, camera) collect raw data from the farm.

2. Top Layer (Edge Layer) – the ESP32 processes sensor readings, executes control logic, and manages local alerts.

3. Middle Layer (Cloud Layer) – Wi-Fi transmits data to a cloud server. When internet is unavailable, the system continues operating locally and syncs later.

4. Bottom Layer (Application Layer) – the Android app retrieves data from the cloud, displays it, and sends user commands back down to the ESP32.

This layered design keeps the system modular. If a sensor fails, the rest keeps working. If the cloud connection drops, the ESP32 still handles irrigation and animal detection based on its local thresholds.

The architecture follows a Three to four-layer approach:

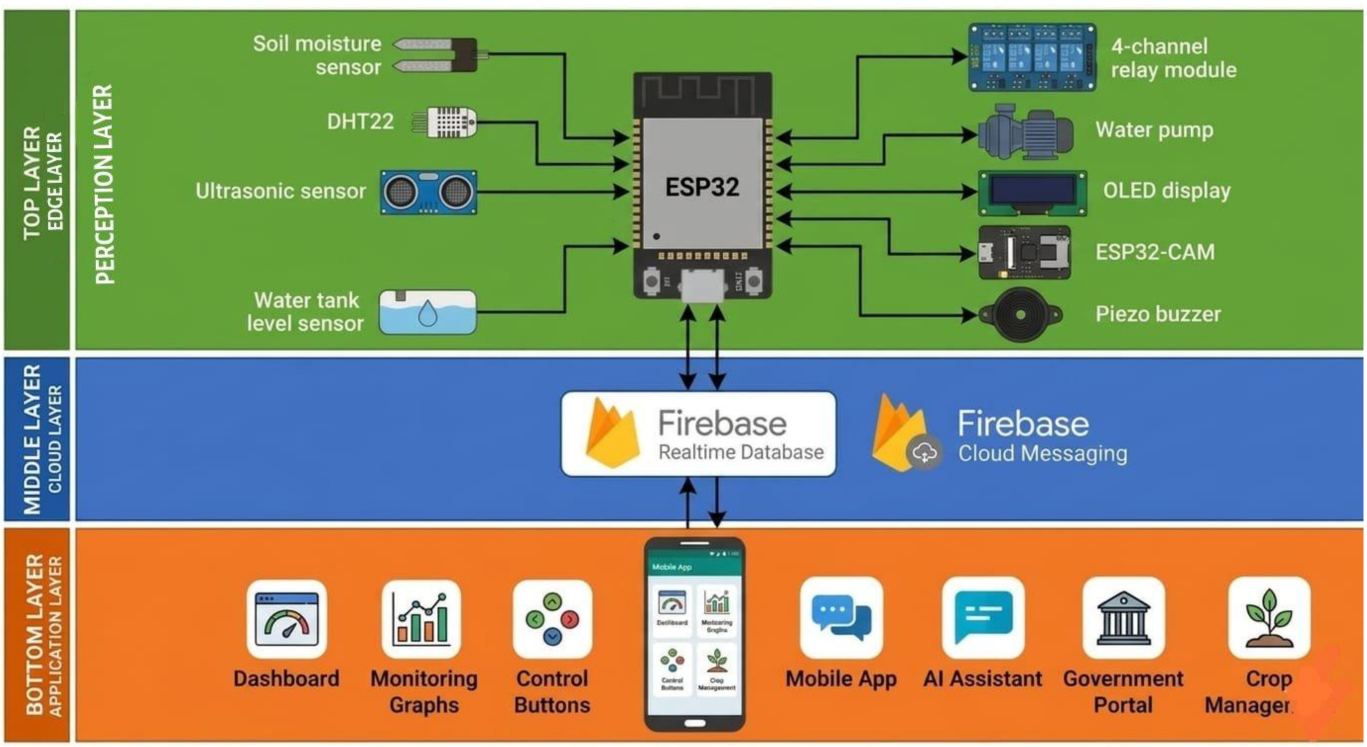

Fig. 4. Overall system architecture illustrating data flow between sensor layer, edge processing (ESP32), cloud layer, and mobile application.

5. Results and Discussion

We tested the system on a small test farm over three months, covering both dry and wet spells. The results showed clear benefits.

Water savings: Compared to the old method—where watering happened on a fixed schedule regardless of weather—our system used about 35% less water. The pump ran only when the soil moisture sensor said it was needed, and it stopped automatically once the soil was adequately wet. No more mid-day watering in the sun or forgetting to turn the pump off. [2,3]

Fig. 5. Field deployment of the proposed system showing sensor placement, irrigation setup, and on-site hardware configuration.

Table 2: Performance evaluation of the proposed system based on key operational parameters including sensor accuracy, response time, and data transmission efficiency.

Parameter	Measured Value	Expected Value	Result
Sensor Accuracy	96%	≥90%	Pass
Pump Response Time	2 seconds	≤3seconds	Pass
Data Transmission Time	1.5 seconds	≤2seconds	Pass
System Reliability	Stable	Stable	Pass

Speed of response: The system reacted to dry soil within seconds. When we simulated a sudden drop in moisture, the pump started in under three seconds on average. For animal detection, the ultrasonic sensor triggered the buzzer and sent a notification in less than two seconds. That kind of speed can make the difference between scaring off an animal and losing a row of crops [1],

Detection reliability: The ultrasonic sensor correctly detected objects larger than a rabbit at distances up to three meters. False alarms happened occasionally when strong wind moved nearby plants, but we reduced them by adding a short averaging filter in the ESP32 code. The camera-based animal detection (added in a later iteration) gave us visual confirmation, and the machine

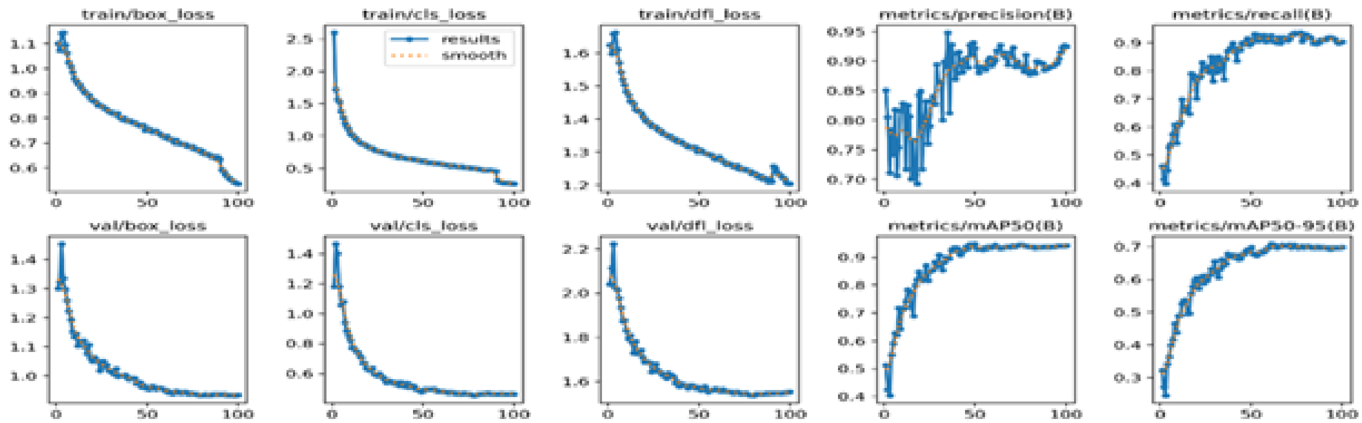

Fig 6: Training performance of the animal detection model showing accuracy and learning progression over training iterations.

Fig. 4: 4-Layer Implementation Framework Architecture, Source: Authors' compilation from systematic database search

Aspect	Sustainable Farming and Smart Agriculture	Cascaded YOLOv8 with Adaptive Preprocessing	Our System
Central Controller	Raspberry Pi 4	Computer	ESP32 Dual-Core
Deployment Cost	High (100+ per node)	High (100+ per node)	Low (10–15 per node)
Data Handling	Cloud Storage	Local Dataset	NoSQL Real-time
User Interaction	Web Portal Only	MATLAB Interface	Native Android App

learning model achieved around 90% accuracy in identifying common farm animals like cows, goats, and dogs [1,4].

User experience: We asked a few local farmers to try the app. They picked it up quickly—most said they preferred the natural language feature because they didn't have to remember where each setting was. One farmer mentioned he finally understood the government scheme for drip irrigation because the app explained it clearly, and he applied successfully for the subsidy [3].

6. Comparison with Existing Approaches

Existing farm automation systems generally fall into two categories. The simplest are timer-based controllers—they water at fixed times, ignoring whether it rained or the soil is already wet. More advanced IoT systems often just send data to a display or cloud, leaving the farmer to decide what to do. Neither approach truly automates the farm or offers integrated security.

Our system goes further. It combines real-time sensing, automated irrigation, animal detection, and remote control into one cohesive solution. The addition of NLP and government scheme updates makes it more than just a monitoring tool—it becomes a decision support system that meets farmers where they are [1,3,5].

7. Conclusion

We set out to build a farm management system that actually solves the problems farmers face every day: water waste, unpredictable labor, animal damage, and lack of timely information. By integrating IoT sensors, automated control, mobile access, and even natural language interaction, we created a system that is both practical and easy to use.

Our tests showed significant improvements in water efficiency and response times, and farmers who used the app appreciated being able to monitor and control their farms remotely. The animal detection feature gave them peace of mind, especially at night.

9. Future Work

1. **TinyML for Edge Autonomy:** Deploying lightweight quantized (int8) models directly onto hardware like the ESP32-S3 to enable offline pest and disease detection, reducing latency and bandwidth costs.

2. **Advanced N-P-K and pH Sensors:** These sensors measure the levels of nitrogen, phosphorus, potassium, and pH in the soil. They are moving the platform closer to high-precision fertilization methods.

3. **Renewable Power Autonomy:** Using 20W solar panels, LiFePO4 battery banks, and MPPT management to give remote field nodes 100% independent power autonomy.

Conflicts of interest: The authors stated that no conflicts of interest.

Correspondence and requests for materials should be addressed to **Sameer Patil**

Funding Source: This research did not obtain specific funding from public, commercial, or non-profit organizations. The project was developed using institutional laboratory resources and personal efforts of the authors.

Ethical Approval:
This study does not involve any human participants or live animal experimentation. Therefore, ethical approval was not required for this research

Peer review information : IRJSE thanks the anonymous reviewers for their contribution to the peer review of this work. A peer review file is available.

Reprints and permissions information is available at https://www.irjse.in/reprints

References

1. Subahi, A. F., & Bouazza, K. E. (2017). "Novel Animal Detection System: Cascaded YOLOv8 With Adaptive Preprocessing and Feature Ex traction".Vol.12,pp.110575-110587,Aug.2024

   https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=10630565

2. Ayaz, M., Ammad-Uddin, M., Sharif, Z., Mansour, A., & Aggoune, E. M. (2019). "Internet-of-Things (IoT) based smart agriculture: Towards making the fields talk". Vol.7, pp.129551-129583, Aug.2022. https://www.iosrjournals.org/iosr-jece/papers/Vol.%2017%20Issue%204/Ser-1/A1704010107.pdf

3. Khaliq, A., Khan, A., Jan, S., Umair, M., Gulshair, A., Ali, A., & Shah, U. A. (2025). "AI-driven smart agriculture: An integrated approach for soil analysis, irrigation, and crop-fertilizer recommendations".Vol.13, pp.141124-141138, Aug.2025.

   https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=11104235

4. Bhargava, A., Shukla, A., Goswami, O., Alsharif, M. H., Uthansakul, P., & Uthansakul, M. (2017). "Plant Leaf Disease Detection, Classification, and Diagnosis Using Computer Vision and Artificial Intelligence".Vol.12, pp. 37443 - 37469, Mar.2024.

   https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=10458943

5. Natarajan, B., Elakkiya, R., Bhuvaneswari, R., Saleem, K., Chaudhary, D., & Samsudeen, S. H. (2022). "Creating alert messages based on wild animal activity detection using hybrid deep neural networks".Vol.11, pp. 67308-67321, Jul.2023.

   https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=10163794

Publisher's Note

IJLSCI remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.