Arpit Kumar Gope2025, 13:4
ISSN (Online): 2348-4098
ISSN (Print): 2395-4752

International Journal of Science, Engineering and Technology

An Open Access Journal

Agrobotics: IoT Based Smart Agriculture System

[1]Arpit Kumar Gope, [2]Jadav Ansh Ashokbhai, [3]Kargavkar Karan Bharatbhai,[4]Kunbi Paras Vilash, [5]Dr. Nithya A

Computer Science & Engineering Parul University Vadodara, Gujarat

Abstract- Agriculture continues to be the backbone of most farm-based economies, both as a major economic sector and as a source of employment. Nevertheless, the sector still experiences major challenges like unpredictable climatic fluctuations, inadequate real-time monitoring of crops, poor resource use, and declining agricultural labor. The use of smart agriculture, which merges modern technologies with conventional farming, presents effective solutions to these challenges. This article proposes a smart agriculture system that utilizes the Internet of Things (IoT) and Artificial Intelligence (AI). IoT sensors allow for realtime environmental data acquisition, while AI facilitates predictive modeling and self-learning decision-making. Blockchain technology further helps in secure, transparent, and efficient management of agricultural supply chains. The suggested method increases productivity, minimizes wastage of resources, and optimizes supply chain processes, and thus supports sustainable and technology-based agriculture.

Keywords- Precision Farming, Agrobotics,Internet of Things (IoT), Artificial Intelligence (AI), Automated Irrigation, Blockchain, Supply Chain Transparency, Cloud Computing, Crop Surveillance, Sustainable Farming.

I. INTRODUCTION

The modern agriculture industry is confronted with several challenges that prevent efficiency, sustainability, and general development. Irregular climatic conditions have a direct impact on harvests, and a lack of qualified personnel raises the cost of operations and reduces agricultural activity. Traditional irrigation methods tend to cause overconsumption of water, subjecting natural resources to further pressure. In addition, the lack of real-time monitoring limits timely intervention, ultimately lowering productivity. Supply chain inefficiencies, such as low transparency and data manipulation, additionally lead to losses, delays, and unfair practices.

To mitigate these, the present research proposes a smart agriculture framework that incorporates high-end digital technologies like the Internet of Things (IoT) and Artificial Intelligence (AI). IoT-based sensors monitor environmental parameters like soil condition, humidity, and temperature continuously, thus enabling precision farming. Through the comparison of sensor values against set thresholds, the system facilitates quick decision-making and automates key functions like irrigation and pest control. Smart irrigation, specifically, maximizes water efficiency by providing the necessary volume at the optimal time. Blockchain technology also provides secure, transparent, and tamper-free agricultural supply chain management, thus supporting enhanced traceability and trust.

The use of all these technologies results in cost savings in farming, enhanced quality of crops, and honest trade practices. Overall, this method optimizes agriculture productivity and opens the door to data-driven, sustainable, and future-proofed farming.

II. EASE OF USE

The system of smart agriculture is crafted with ease of usage and flexibility in its design, so even those farmers with less technical knowledge can use it. IoT sensor-collected data are represented on an easy-to-use, web-based dashboard to allow farmers to monitor real-time soil nutrient, moisture, temperature, and humidity levels. The manual-intensive tasks of irrigation and fertilization are

carried out by automated modules, ensuring it does not require much human intervention. The system also provides timely notifications and suggestions in plain, easy-to-comprehend language, allowing farmers to make decisions with minimal technical expertise.

The plug-and-play nature of IoT devices, coupled with wireless connectivity, makes setup and maintenance easy. In addition, its modular nature enables scalability and simple integration of new features as and when required. This flexible and easy-to-understand platform reduces complexity and promotes uptake even by small and remote farmers.

III. LITERATURE SURVEY

Literature Survey

IoT in Agriculture

The adoption of IoT in agriculture has made it possible to monitor key parameters such as soil moisture, health of crops, temperature, and humidity continuously. These systems usually draw on wireless sensor networks and cloud platforms to harvest, store, and crunch agricultural data in a way that helps farmers make informed decisions. Most current solutions are still, however, narrow in scope and only provide simple monitoring and automation functionality. The fact that they are not integrated with Artificial Intelligence (AI) limits predictive power, limiting them to reactive functionality that responds to issues once they arise instead of actively preventing them.

AI for Precision Farming

Artificial Intelligence (AI) has become one of the pillars of precision farming through machine learning, deep learning, and computer vision-based technologies to enable data-driven decision-making. AI-based systems have the capability to predict future pest or disease outbreaks based on historical climate data coupled with real-time sensor data. This way, farmers can pre-emptively take steps to prevent damage to crops. In addition, AI optimizes resource utilization through analysis of soil and weather data to maximize water and fertilizer use. Image recognition models can diagnose early signs of nutrient deficiencies or stress in plants, which allows for early intervention. Models for predicting yields also help with strategic decision-making and planning, and resource allocation.

However, a few obstacles exist. AI deployment necessitates high computational power, standard datasets in agriculture are limited, and infrastructure access is mostly limited in smallscale farming. These constitute limiting factors to large-scale application of AI in agriculture.

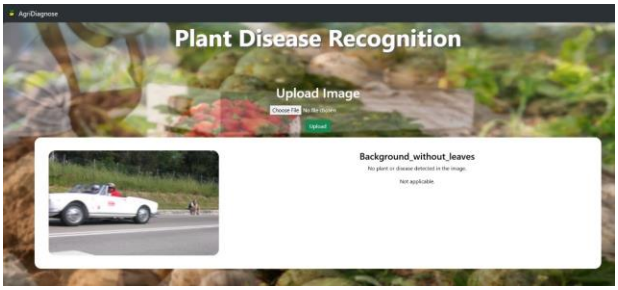


Figure. 1. Sample of AI model

Blockchain in Agribusiness Supply Chains

Blockchain technology offers a revolutionary answer to supply chain management in agriculture through transparency, immutability, and data protection. Every transaction placed on the blockchain is irreversible and tamper-proof, building trust among producers, distributors, retailers, and consumers. Traceability guarantees the authenticity and quality of agricultural products during the farm-to-market process. Blockchain also ensures less counterfeiting and enhances food safety by enabling quick identification and elimination of contaminated products.

Figure. 2. Sample of AI model

The use of smart contracts also goes a long way to automatize financial transactions, reduce paperwork, and speed up the payment settlements. Large-scale

adoption, though, is resisted by challenges like high setup costs, availability of qualified experts in short supply, and interoperability issues with different blockchain platforms.

Shortcomings in Existing Systems

Though with IoT, AI, and blockchain, there has been progress, various deficiencies remain in existing agricultural technology. Sophisticated technology like autonomous farm equipment and drones is still not generally used on farms. Cybersecurity weaknesses are also an issue, since most IoT platforms do not have strong protective cybersecurity measures. While blockchain provides an assured mechanism for secure exchange, its application in agriculture remains insignificant. High infrastructure expenses and technical knowhow often disallow small and marginal farmers from availing these technologies. These challenges can be overcome through innovative, cost-efficient, and farmer-friendly solutions that are inclusive and scalable in smart farming.

Watching the Weather in Real Time

Precise and real-time weather data are required for efficient farm management and risk planning. With live updates on temperature, rainfall, humidity, wind speed, and storm warnings, farmers can initiate precautionary measures to safeguard crops and maximize the use of resources. For example, irrigation schedules may be modified with a surprise rainfall, or safeguard measures may be taken during heavy winds. Weather-based information also assists in water savings, pest management, and disease control methods.

But third-party weather service provider reliance poses challenges like poor coverage in rural areas, periodic inaccuracies, and late notification. In spite of these limitations, making weather data part of IoT and AI-based systems allows for automated, adaptive adjustments, making real-time weather tracking an essential part of future smart agriculture systems.

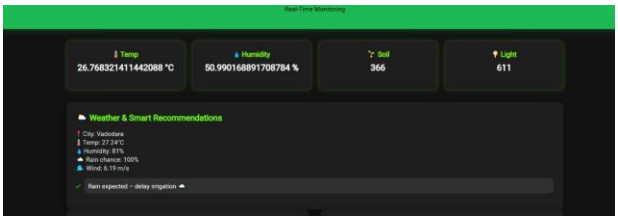


Figure. 3. Weather prediction API

IV. SOFTWARE REQUIRMENT SPECIFICATION

Software Requirement Specification (SRS) This sectionspecifies the functional and non-functional specifications of the envisioned IoT-based Smart Agriculture and Supply Chain Management System. The goal is to make traditional farming more modern by incorporating Internet of Things (IoT), Artificial Intelligence (AI), and blockchain technologies to facilitate higher productivity, sustainability, and transparency.

Introduction The main objective of this project is to develop and deploy an integrated smart farming platform that can automate agricultural activities based on real-time sensing and smart analytics. The platform is intended to enhance crop yield, optimize resource use, and enhance the credibility of agri-supply chains. The potential actors are farmers, agritech start-ups, research institutions, and policy-makers who need reliable instruments for monitoring, analysis, and decisionmaking in agriculture.

Functional Requirements The system will also offer the following essential features:

Real-time monitoring: Soil moisture, temperature, humidity, and livestock movement will be measured continuously through IoT-based sensors. Smart irrigation: An irrigations system powered by AI will automatically adapt water distribution in real-time according to soil conditions and crop needs. Security and surveillance: PIR motion sensors and camera modules will provide additional protection for livestock and assets. Blockchain-based supply chain: Every phase of the crop life cycle and distribution chain will be irrevocably logged, enhancing traceability and confidence. Predictive analytics: Machine learning-based programs will predict

weather patterns, pest infestations, and best farming practices. Centralized dashboard: A web- and mobile-accessible cloud interface will enable farmers and managers to remotely monitor, regulate, and oversee operations.

Non-Functional Requirements

Scalability: The system should accommodate deployment over vast fields of agricultural land with thousands of networked IoT devices. Security: Data integrity and confidentiality should be maintained via encryption and blockchainbased authentication. Reliability: The solution should provide consistent performance across varied environmental conditions while keeping high analytical accuracy. Usability: The interface should continue to be intuitive and easy to use even for non-technical users. Low Latency: Real-time processing must provide quick responses for auto-decision making and notifications.

Software and Hardware Requirements

Software Requirements: Operating Systems: Linux or Windows environments Programming Languages: Python (analytics, AI modules) and JavaScript (frontend/backend development) Database: Cloud-integrated databases like Firebase, MySQL, or MongoDB Cloud Platforms: AWS IoT Core, Google Cloud IoT, or Microsoft Azure for IoT and AI services Frameworks: React.js for frontend; Node.js for backend APIs and services

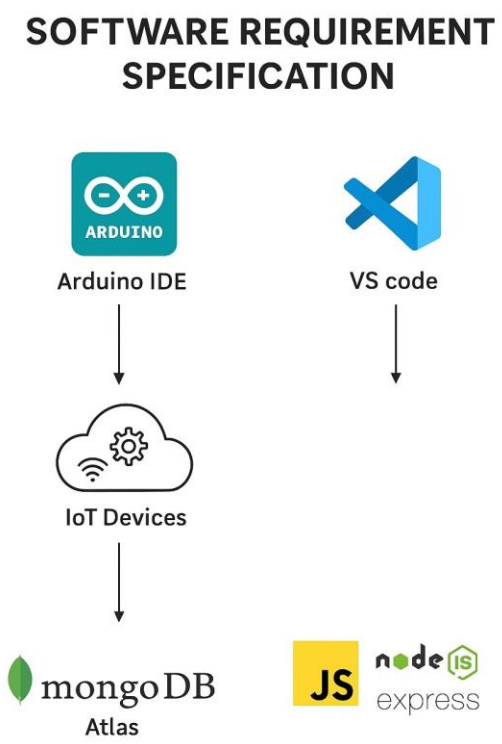


Figure. 4. Software requirments

Hardware Requirements: Sensors: PIR motion sensors, temperature, humidity, and soil moisture Controllers: Raspberry Pi and Arduino for data processing and acquisition Actuators: Irrigation valves and electric motors for automatic responses Communication: Zigbee, LoRa, or Wi-Fi modules based on the energy and range requirements.

Constraints The system's functionality relies on reliable internet connection for real-time synchronization and remote access. Upfront cost may be high because of the cost associated with IoT devices, blockchain infrastructure, and microcontroller hardware.

Assumptions and Dependencies End users will have access to smartphones or computers for

interacting with the dashboard.Third-party cloud services will be continuously available for data storage, AI computation, and device connectivity during the operational lifecycle.

V. METHODOLOGY

Development Approach

The Smart Agriculture and Supply Chain platform, driven by IoT, is designed using a modular and decentralized architecture. This design ensures scalability, automation, secure data sharing, and real-time monitoring of environmental and farming conditions. The development follows an Agile methodology, which allows for iterative progress where each stage is continuously tested, refined, and improved.The modular structure enables independent development and enhancement of the key components—namely smart farming techniques . Artificial Intelligence (AI) is integrated to process live sensor data, offering predictive insights for [5] precision agriculture. Additionally, a cloud-based infrastructure supports remote farm supervision by providing real-time data visualization, system control, and responsive feedback through an interactive dashboard.

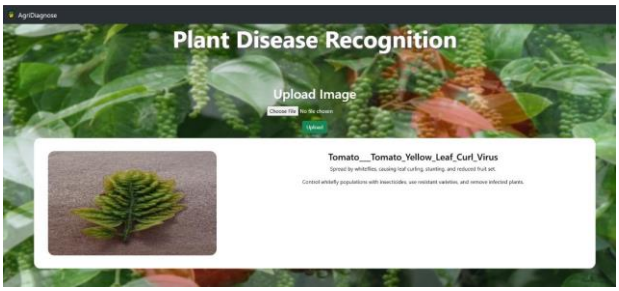


Figure. 5. Smart Agriculture Dashboard

Technology Stack

To implement the Smart Agriculture and Supply Chain system efficiently, a modern and comprehensive technology stack is employed, covering the frontend, backend, IoT modules, and blockchain layer.

Frontend User Interface

The user interface (UI) is developed using popular frameworks such as React.js, Vue.js, or Angular, selected based on specific project requirements. These frameworks allow for dynamic, interactive, and real-time dashboards that visualize agricultural and supply chain data. For responsiveness and styling, libraries such as Tailwind CSS, Material UI, or Bootstrap are integrated to ensure an engaging and user-friendly experience.

To manage application state efficiently, solutions like Redux, Vuex, or Zustand are applied, ensuring smooth synchronization of data across components. For blockchain-based features, wallet providers including MetaMask and WalletConnect are incorporated, enabling secure decentralized authentication and direct user interaction with smart contracts.

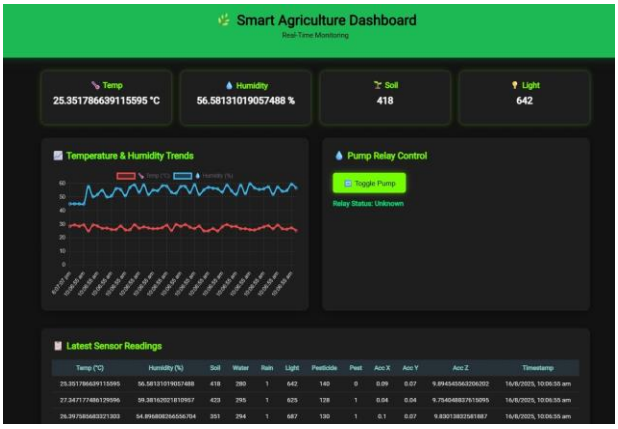


Figure. 6. Smart Agriculture Dashboard

Backend (Server-Side Logic)

The backend layer forms the central processing unit of the system and is typically implemented using Node.js with Express.js, or Python with Flask/Django, depending on the scalability and performance requirements. For data persistence, databases such as MongoDB, PostgreSQL, or Firebase are employed, offering both structured and unstructured data handling capabilities.

Data communication between frontend and backend is enabled through RESTful APIs or GraphQL, ensuring seamless and efficient data exchange. For security, authentication and access control mechanisms like JSON Web Tokens (JWT) or OAuth are integrated, safeguarding sensitive data and protecting system endpoints.

IoT-Enabled Smart Farming

The smart farming module leverages various sensors, including soil moisture detectors, DHT22 temperature and humidity sensors, and PIR sensors, to collect continuous environmental data and monitor farm microclimates. Alongside sens- ing, actuators such as automated irrigation systems, drones for aerial surveillance, and robotic tools are deployed. Guided by AI algorithms and real-time sensor feedback, these actuators enable autonomous decision-making in farming operations. For communication and connectivity, the system relies on lightweight protocols like MQTT and HTTP, along with longrange technologies such as LoRaWAN, ensuring stable data transfer even in remote agricultural regions..

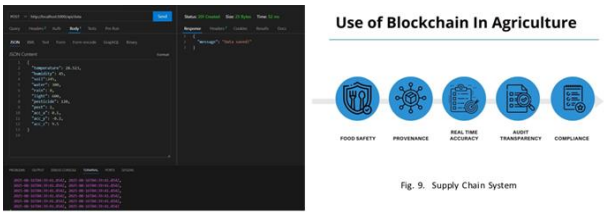


Figure. 7. Backend Rest APIs

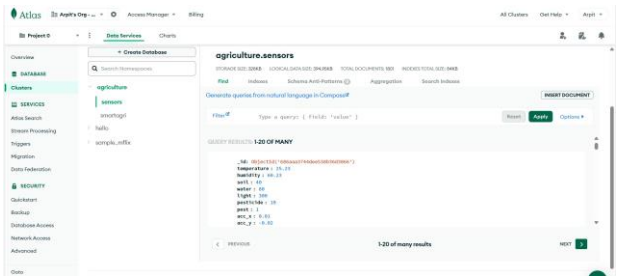


Figure. 8. MongoDB Atlas Data Gathering

Blockchain-Enabled Supply Chain Security

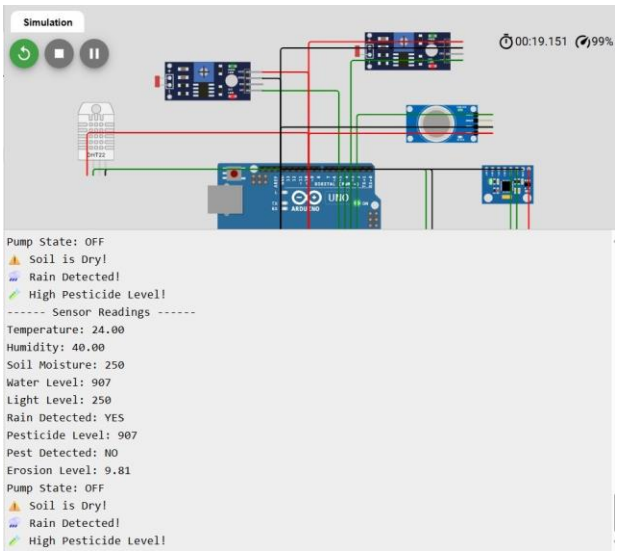
Blockchain is the key to supply chain security. By leveraging smart contracts, transactions and agreements are automatically and securely enforced, dispelling the necessity of intermediaries and encouraging greater trust among stakeholders. The decentralized ledger guarantees all records of transactions as immutable, transparent, and traceable across every step from production to final delivery.

In order to further improve security, cryptographic authentication mechanisms are implemented that enable only the legitimate parties to initiate or verify transactions. This method secures agricultural trade from fraudulent behavior, data tampering, and unauthorized access.

The platform operates as an uninterrupted series of processes, beginning from data collection and continuing to blockchain-driven traceability and automated feedback.

Data Acquisition and Processing

IoT sensors monitor soil health, climatic factors, and environmental parameters continuously, uploading the information to the cloud infrastructure. [8] The stream is monitored in real time by AI algorithms, which identify patterns and create predictions for irrigation routines, fertilizer needs, and crop management techniques.



Figurer. 10. Simulation and Monitoring

Crop Monitoring and Automated Irrigation

Upon detecting deviations from ideal conditions, the AI engine automatically turns on irrigation systems and fertilizer injectors, ensuring precision agriculture with minimal resource waste. To supplement this, aerial drones take multispectral images of the crops, [9] facilitating early disease, pest, and crop stress detection. Such insights prompt timely action, enhancing the health of crops and overall production.

Blockchain-Integrated Supply Chain
Once products are harvested, farm production information is all logged on the blockchain for immutability and transparency. Smart contracts are utilized to authenticate transactions, automate payments, and provide a tamper-proof record of supply chain activity, and establish trust with everyone involved.

Remote Access and Control
A web and mobile platform dashboard delivers real-time analytics, notifications, and recommendations to farmers. They can control irrigation manually from this portal, read AI-based advice, and monitor patterns in productivity at any time and from anywhere, making remote farm management efficient.

Summary
The suggested solution provides a secure, automated, and scalable agricultural system. IoT facilitates real-time monitoring, AI offers predictive intelligence, blockchain provides data integrity and traceability, and cloud computing provides remote accessibility and scalability. By integrating these technologies, the system fosters an integrated agricultural ecosystem that is efficient, transparent, and sustainable.

VI. HARDWARE USED

The Smart Agriculture and Supply Chain platform based on IoT is developed with cost-effective, low-power, and modular hardware components. The devices are chosen keeping in mind the cost factor, integration simplicity, and scalability requirements, so they will work efficiently in both research projects and small farms. The modules are designed for the lowest power consumption while enabling local control and cloud monitoring. Following is the description of the major hardware components:

Microcontroller (Arduino UNO / NodeMCU ESP8266)
The microcontroller is the master controller, tasked with reading sensor information and performing control actions. Arduino UNO and ESP8266 Wi-Fi module are options where Arduino processes data while ESP8266 performs wireless communication to the cloud or dashboard. NodeMCU ESP8266 delivers both processing power and wireless communication in a single platform, enabling wiring, hardware space, and cost overall are minimized.

Soil Moisture Sensor
This sensor detects the moisture level present in the soil and crops. transfers analog signals to the microcontroller. The system can automatically start irrigation based on predetermined levels, avoiding both over-watering and under-watering. This enhances water saving and promotes healthy plant growth, with less frequent human intervention—especially suitable for resource-poor rural regions.

DHT22 Sensor (Temperature and Humidity)
The DHT22 sensor measures temperature and humidity with greater accuracy and greater range than the DHT11. As these parameters are directly affecting crop development and controlling the greenhouse, the DHT22 is best suited for agricultural monitoring, where climatic changes play a significant role in yields.

Water Level Sensor
Mounted on the irrigation tank, this sensor gauges water availability and makes pumps function only when there is enough water. This avoids pump damage from dry running and allows for sustainable water management. When connected to the cloud dashboard, the system can notify farmers to fill up storage tanks in advance.

Relay Driver Module
The relay module serves as the interface for the low-voltage microcontroller (Arduino/NodeMCU) to high-voltage devices like water pumps. It offers protection and electrical isolation, preventing the damage to the controller by any variation in the pump's power supply. This facilitates switching of agricultural equipment safely and reliably both automatically and remotely. 6.6 Submersible Pump
A submersible pump is employed for irrigation using a lowvoltage DC. It is self-starting when the soil moisture content decreases below the predetermined value and directly supplies water to plant roots, ensuring minimum wastage. In advanced

models, variable-speed pumps can be integrated to regulate water flow based on immediate crop requirements.

Power Supply (5V DC Regulated Source)

The system is driven by a regulated 5V DC supply, which can be provided by an AC adapter or a rechargeable lithiumion battery. In field applications, the installation can be coupled with solar panels, rendering the system renewable and offgrid farming-friendly. Owing to low energy consumption, the system can operate for long periods without requiring regular battery replacement.

Indicators (LEDs and Buzzers)

Basic monitoring devices such as LED indicators and buzzers are also included for real-time local monitoring at the field location. LEDs indicate system status in terms of sensor activity, irrigation cycles, or connectivity status, while buzzers give a visual alarm for important events such as low water level or system faults. These simple yet efficient indicators enable farmers to monitor locally, without being wholly reliant on remote dashboards.

VII. SOFTWARE USED

Software Ecosystem , The envisaged Smart Agriculture and Supply Chain System based on IoT is supported not only by hardware modules but also by a solid software platform. The software stack supports integrated coordination among sensors, controllers, cloud infrastructure, and user interfaces. It integrates embedded programming, communication protocols, and cloud integration for providing real-time monitoring, automation, and secure data exchange. The key software components are as follows:

Development Environment

Programming and firmware installation are done through the Arduino Integrated Development Environment (IDE), which is commonly used in embedded system design and prototyping. [7] The IDE offers an easy-to-use interface for coding, compiling, and uploading to boards like Arduino UNO, NodeMCU ESP8266, and ESP32. Open-source and backed by a large developer community, it comes with comprehensive library support to allow quick prototyping. Moreover, its embedded debugging, monitoring, and firmware management capabilities simplify development, making it ideal for both lab experiments and real-world deployments.

Programming Language

The firmware of the system is mainly coded in Embedded C, whose direct hardware-level access ensures quick and efficient operation execution. Embedded C is used to execute : Sensor data acquisition (soil moisture, temperature, humidity, water level, etc.),Actuator control (relays, pumps, buzzers). Automation logic for irrigation control and supply chain notifications.In contrast to higher-level programming As a language, Embedded C is compact, quick in execution, and well-adapted to embedded environments. for microcontrollers with scarce resources, providing real-time execution and reliability to agricultural applications.

Communication Libraries

Specific Arduino libraries are employed to provide support for connectivity between IoT devices and cloud-based services. Typically used libraries are: WiFi.h → Enables wireless connection for ESP8266/ESP32 boards.HTTPClient.h → Allows devices to make HTTP/HTTPS requests to cloud APIs.ArduinoJson → Tiny JSON parser used for structuring sensor data prior to transmission.These libraries facilitate robust communication and data exchange, allowing IoT devices to easily interface with the internet for real-time decisionmaking.

Cloud Integration

To allow remote monitoring and control, the system is connected to cloud platforms like Firebase or ThingsBoard. These platforms serve as middleware and offer:Real-time storage and access of environmental information (soil moisture, humidity, temperature, water levels).Interactive dashboards for sensor trends and system performance visualization.Remote actuation, which allows the pumps and devices to be turned on/off using the dashboard.Supply chain tracking, providing transparent and secure recording of storage and distribution records.Cloud services widen the system's scope beyond local deployment to

worldwide accessibility, enabling users to observe and regulate farm activities from anywhere via web or mobile.

Debugging and Testing Tools

For debugging, the Serial Monitor of the Arduino IDE is utilized. It gives real-time feedback like:Sensor values.Microcontroller control commands.Error logs and network connectivity problems.This output enables developers to adjust sensor thresholds, debug communication issues, and test system accuracy prior to its release into actual agricultural fields.

User Interface (UI)

A web-based dashboard is implemented with HTML, CSS, and JavaScript to provide farmers and supply chain managers a clean and efficient platform for dealing with the system. The dashboard features:Visualization of environmental parameters (soil moisture, temperature, humidity).Real-time updates on system activity (water levels, pump operation).Remote control features, enabling irrigation and supply chain operations to be controlled directly.For enhancing interactivity and scalability, the UI can be augmented with frameworks such as React.js or Vue.js to provide a responsive and intuitive experience.

VIII. CODE IMPLEMENTATION

IoT Device Implementation (Arduino IDE)

```
#include <WiFi.h>
#include <HTTPClient.h>
#include <DHT.h>
#include <Wire.h>
#include <Adafruit_MPU6050.h>
#include <Adafruit_Sensor.h>
// --- DHT Sensor --#define DHTPIN 15
#define DHTTYPE DHT22
DHT dht(DHTPIN, DHTTYPE);
// --- Sensor Pins --#define SOIL_PIN      32
#define WATER_PIN      33
#define RAIN_PIN       25
#define LIGHT_PIN      34
#define PESTICIDE_PIN  35
#define PIR_PIN 14
#define RELAY_PIN      26
// --- I2C Pins for MPU6050 ---
#define SDA_PIN        21
#define SCL_PIN        22
Adafruit_MPU6050 mpu;
// --- WiFi Credentials --const char* ssid = "YOUR_WIFI_NAME"; const char* password = "YOUR_WIFI_PASSWORD";
// --- Backend URL --const char* serverURL = "http://YOUR_SERVER_IP_OR_NGROK/api/data";
void setup() {
Serial.begin(115200); dht.begin(); Wire.begin(SDA_PIN, SCL_PIN); if (!mpu.begin()) {
Serial.println("Failed to find MPU6050chip"); while (1); } pinMode(RAIN_PIN, INPUT); pinMode(PIR_PIN, INPUT); pinMode(RELAY_PIN, OUTPUT); digitalWrite(RELAY_PIN, LOW);
Serial.print("Connecting to WiFi"); WiFi.begin(ssid, password); while (WiFi.status() != WL_CONNECTED) {
delay(1000); Serial.print(".");
}
Serial.println("\nWiFi Connected!");
Serial.print("IP: ");
Serial.println(WiFi.localIP());
}
void loop() {
float temperature = dht.readTemperature(); float humidity = dht.readHumidity(); int soil = analogRead(SOIL_PIN); int water = analogRead(WATER_PIN); int rain = digitalRead(RAIN_PIN); int light = analogRead(LIGHT_PIN); int pesticide = analogRead(PESTICIDE_PIN); int pest = digitalRead(PIR_PIN);
sensors_event_t a, g, temp_event; mpu.getEvent(&a, &g, &temp_event);
String payload = "{";
payload += "\"temperature\":" + String(temperature) + ","; payload += "\"humidity\":" + String(humidity) + ",
"; payload += "\"soil\":" + String(soil) + ","; payload += "\"water\":" + String(water) + ","; payload += "\"rain\":" + String(rain) + ","; payload += "\"light\":" + String(light) + ","; payload += "\"pesticide\":" + String(pesticide) +
","; payload += "\"pest\":" + String(pest) + ","; payload += "\"acc_x\":" + String(a.acceleration.x)
```

```
+ ","; payload += "\"acc_y\":" + String(a.acceleration.y)
+ ",";
payload += "\"acc_z\":" + String(a.acceleration.z)
; payload += "}";
Serial.println("Sendingdata...");
Serial.println(payload);
if (WiFi.status() == WL_CONNECTED) { HTTPClient http; http.begin(serverURL);
http.addHeader("Content-Type", "application/json
"); int httpResponseCode = http.POST(payload);
Serial.print("HTTPResponse Code: ");
Serial.println(httpResponseCode); http.end();
} else {
Serial.println("WiFiDisconnected!");

}
delay(5000);
}
```

```
------ Sensor Readings ------
Temperature: 24.00
Humidity: 40.00
Soil Moisture: 250
Water Level: 907
Light Level: 250
Rain Detected: YES
Pesticide Level: 907
Pest Detected: NO
Erosion Level: 9.81
Pump State: OFF
⚠️ Soil is Dry!
🌧️ Rain Detected!
🧪 High Pesticide Level!
------ Sensor Readings ------
Temperature: 24.00
Humidity: 40.00
Soil Moisture: 250
Water Level: 907
Light Level: 250
Rain Detected: YES
Pesticide Level: 907
Pest Detected: NO
```

Figure. 11. Simulation and Monitoring

AI-Based Leaf Disease Detection and Analytics (Python)

```
from flask import Flask, render_template,request,
redirect,send_from_directory,url_for
import numpy as np import json import uuid import
tensorflow as tf
app = Flask(__name__) model =
tf.keras.models.load_model("models\
plant_disease_recog_model_pwp.keras")
label = ['Apple___Apple_scab',
'Apple___Black_rot',
'Apple___Cedar_apple_rust',
'Apple___healthy',
'Background_without_leaves',
'Blueberry___healthy',
'Cherry___Powdery_mildew',
'Cherry___healthy',
'Corn___Cercospora_leaf_spotGray_leaf_spot',

'Corn___Common_rust',
'Corn___Northern_Leaf_Blight',
'Corn___healthy',
'Grape___Black_rot',
'Grape___Esca_(Black_Measles)',
'Grape___Leaf_blight_(Isariopsis_Leaf_Spot)',
'Grape___healthy',
'Orange___Haunglongbing_(Citrus_greening)',
'Peach___Bacterial_spot',
'Peach___healthy',
'Pepper,_bell___Bacterial_spot',
'Pepper,_bell___healthy',
'Potato___Early_blight',
'Potato___Late_blight',
'Potato___healthy',
'Raspberry___healthy',
'Soybean___healthy',
'Squash___Powdery_mildew',
'Strawberry___Leaf_scorch',
'Strawberry___healthy',
'Tomato___Bacterial_spot',
'Tomato___Early_blight',
'Tomato___Late_blight',
'Tomato___Leaf_Mold',
'Tomato___Septoria_leaf_spot',
'Tomato___Spider_mitesTwo-spotted_spider_mite',
```

'Tomato___Target_Spot',
'Tomato___Tomato_Yellow_Leaf_Curl_Virus',
'Tomato___Tomato_mosaic_virus',
'Tomato___healthy']
with open("plant_disease.json",'r') as file: plant_disease = json.load(file)
print(plant_disease[4])
@app.route('/uploadimages/<path:filename>') def uploaded_images(filename):
return send_from_directory('./uploadimages', filename)
@app.route('/',methods = ['GET']) def home(): return render_template('home.html')
def extract_features(image):
image = tf.keras.utils.load_img(image, target_size=(160,160))
feature = tf.keras.utils.img_to_array(image) feature = np.array([feature]) return feature
def model_predict(image): img = extract_features(image) prediction = model.predict(img)
print(prediction) prediction_label = plant_disease[prediction.
argmax()]
return prediction_label
@app.route('/upload/',methods = ['POST','GET']) def uploadimage():
if request.method == "POST": image = request.files['img'] temp_name = f"uploadimages/temp_{uuid.uuid4
().hex}"
image.save(f'{temp_name}_{image.filename}') print(f'{temp_name}_{image.filename}') prediction = model_predict(f'./{temp_name}_{ image.filename}')
return render_template('home.html',result= True,imagepath = f'/{temp_name}_{image. filename}', prediction = prediction)
else: return redirect('/')
if __name__ == "__main__": app.run(debug=True)

Figure. 12. AI based Leaf Detection App

Code for the DASHBOARD(Javascript)

const express = require("express"); const mongoose = require("mongoose"); const cors = require("cors"); const SensorData = require("./models/SensorData"); const app = express(); const PORT = 5000;
// Middleware app.use(cors()); app.use(express.json());
app.use(express.static('public')); // Serve frontend files
// MongoDB Connection mongoose.connect("mongodb://127.0.0.1:27017/ agriculture", {
useNewUrlParser: true, useUnifiedTopology: true,
}).then(() => console.log(" MongoDB Connected")) .catch(err => console.error(" Mongo Error:", err))
;
// POST Sensor Data app.post("/api/data", async (req, res) => {
try {
const payload = req.body; if (payload.timestamp) delete payload.timestamp; const data = new SensorData(req.body); await data.save(); console.log(" Data saved:", req.body); res.status(201).json({ message: "Datasaved!" })

;
} catch (err) { console.error(" Save failed:", err); res.status(500).json({ error: "Savefailed" });

}
});
// GET All Sensor Data (latest 50) app.get("/api/data", async (req, res) => { try {
let data = await SensorData.find().sort({ _id: -1 }).limit(50); // Validate timestamps data = data.map(item => {
if (!item.timestamp || isNaN(new Date(item. timestamp).getTime())) {
item.timestamp = new Date();
} return item; });
console.log("Timestamps sent to frontend:", data .map(d => d.timestamp)); res.json(data);
} catch (err) { console.error("Errorfetchingdata:", err); res.status(500).json({ error: "Failed to fetch data" });

```
}
});
// POST Toggle Relay State let relayState = false;
app.post("/api/relay", (req, res) => {
relayState = !relayState; console.log("Relay State Changed:", relayState ? "
ON" : "OFF"); res.send(relayState ? "ON" : "OFF");
});
// Start Server app.listen(PORT, () => console.log(`
Server running
on http://localhost:${PORT}`));
```

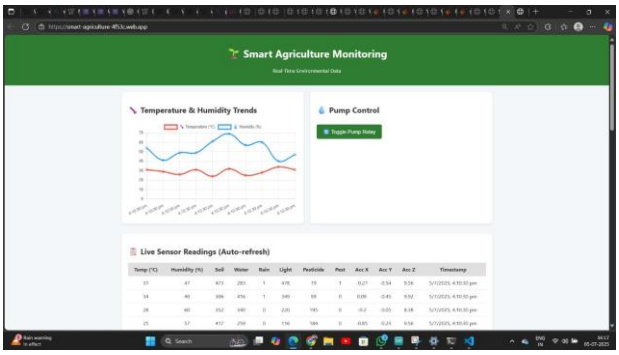


Figure. 13. Smart agriculture monitoring dashboard

IX. CONCLUSION

This research introduces an IoT-supported smart agriculture system that leverages real-time sensor readings to facilitate proactive decision-making in crop care. Integration of an early plant disease detection machine learning model coupled with data visualization dashboard and weather API integration significantly improves predictive and preventive features. Integration of IoT, machine learning, and data analytics showcases a viable framework for sustainable, efficient, and future-proof
agricultural practices.
We can then conclude that the combination of real-time sensing and intelligent prediction not only enhances productivity but also enables resource-efficient and sustainable agriculture.

X. FUTURE SCOPE

Smart Agriculture and Supply Chain System propelled by the IoT has immense scope for expansion in the future through the application of cutting-edge technologies. In agriculture, machine learning and deep learning can be used for predictive analytics for estimation of yields, pest, and disease detection early on, as well as adaptive cultivation practices based on detailed climate data. Blockchain integration can enhance supply chain integrity by providing tamper-evident farm-to-consumer traceability, facilitating automated transactions through smart contracts, and reducing risks of counterfeiting labeling. Integration of drones and autonomous robots can advance precision farming by facilitating targeted spraying of pesticides, round-the-clock monitoring of crops, and automated harvesting—reducing manual labor dependency and enhancing efficiency.Advanced networking technologies like 5G, edge computing, and low-power wide-area networks (LPWAN) can enable quicker, affordable, and reliable communication in rural regions. Multilingual mobile applications specific to farmers with AI-driven advice and direct market connections can offer actionable data, improving decision-making and profitability. Sustainability goals can be achieved through intelligent water management, carbon emissions monitoring, and organic certification through blockchain. At the policy level, integration with government systems enables automated subsidy distribution, simplified food safety compliance, and quick disaster response driven by IoT and drone-based inspections.Collectively, these innovations can transform the system into an end-to-end, smart, and sustainable agri-ecosystem.

REFERENCES

1. S. R. Nandurkar, V. R. Thool, R. C. Thool, " Design and Development of Precision Agriculture System Using Wireless Sensor Network ", IEEE transnational Conference robotization, Control, Energy and Systems (ACES), 2014
2. Y. Kim, R. Evans and W. Iversen, " Remote Sensing and Control of an Irrigation System Using a Distributed Wireless Sensor Network ", IEEE Deals dimension, pp. 1379 – 1387, 2008.
3. L. Li, K. Ota, and M. Dong, " Deep literacy for smart husbandry generalities, tools, operations, and openings, " IEEE Access, vol. 7, pp. 168 – 181, Jan. 2019.

4. Mendez, G. R., Yunus, M. A. M., Mukhopadhyay, S. C. 2012, May. A WiFi Grounded Smart Wireless Sensor Network for Monitoring and Agricultural Environment. In Instrumentation and Measurement Technology Conference(I2MTC), 2012 IEEE International IEEE. 26402645.
5. EIP- AGRI Focus Group, " Mainstreaming Precision Farming, " Nov. 2015.
6. OpenWeather, " Weather API Attestation, "(Online). Available https / openweathermap.org/api.(penetrated Aug. 24, 2025).
7. Arduino, " Arduino IDE Software, "(Online). Available https// www. arduino.cc/en/software.(penetrated Aug. 24, 2025).
8. TensorFlow, " TensorFlow An end- to- end open- source platform for machine literacy, "(Online). Available https// www.tensorflow.org.(Accessed Aug. 24, 2025).
9. S. R. Dubey, S. Jain, and A. Singh, " Leaf complaint discovery using mama - spine literacy ways A review, " International Journal of Computer operations, vol. 181, no. 1, pp. 22 – 26, Jul. 2018.
10. H. Y. Lam, W. T. Leong, and C. K. Leong, " IoT husbandry monitoring system, " in Proc. IEEE Int. Conf. on Consumer Electronics(ICCE), Las Vegas, USA, pp. 137 – 138, Jan. 2018.