IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 1, April 2026
Impact Factor: 8.2

ML-Powered Plant Disease Detection System with ESP 32-Cam

Swadesh Turkane, Akshay Kharpas, Nachiket Hangekar, Chaitanya Pabale, Dr. Devyani Jadhav
Department of CSE [Artificial Intelligence and Machine Learning]
School of Engineering and Technology, Sanjivani University, Kopargaon

Abstract: *In recent years, the agriculture sector has faced growing challenges due to the rise of plant diseases that reduce crop yield and quality. Traditional methods of identifying plant diseases are often manual, time-consuming, and inaccurate, particularly in rural areas where access to agricultural experts is limited. This project, titled "ML-Powered Plant Disease Detection System with ESP32-CAM," presents a smart, cost-effective solution that combines Artificial Intelligence (AI) and the Internet of Things (IoT) to address these issues.*

The system uses an ESP32-CAM module to capture live images of plant leaves, which are then analyzed using a trained YOLOv8 deep learning model to detect various plant diseases. To enhance monitoring, the system also integrates environmental sensors such as the DHT11 (for temperature and humidity) and a soil moisture sensor. All real-time data and disease detection results are displayed on a Blynk IoT mobile dashboard, enabling farmers to receive instant alerts and take timely action.

Field testing was conducted with local farmers, who found the system helpful and easy to use. The AI model achieved over 80% accuracy, with detection results delivered in under 5 seconds. The project not only demonstrates the practical use of AI and IoT in agriculture but also contributes to promoting digital literacy and technology adoption in rural communities..

Keywords*: ESP32-CAM*

I. INTRODUCTION

Agriculture plays a vital role in the Indian economy and the livelihoods of millions, especially in rural areas. However, plant diseases remain a major obstacle, leading to reduced crop yield, lower quality, and financial losses. Most small-scale farmers rely on traditional methods to identify plant diseases, which are often ineffective and delayed. This project addresses that problem by integrating modern technologies like Machine Learning (ML) and Internet of Things (IoT) into a plant disease detection system.

Using an ESP32-CAM microcontroller, this system captures live images of plant leaves. A YOLOv8 ML model then classifies the diseases present in the image. To provide a more holistic solution, DHT11 and soil moisture sensors monitor environmental conditions. A mobile dashboard, built using the Blynk IoT platform, displays real-time alerts and information, helping farmers make timely decisions.

Importance of the Project

- Helps detect plant diseases early and accurately.
- Reduces the use of excessive pesticides.
- Prevents crop loss and increases productivity.
- Encourages the use of digital tools in rural farming communities.
- Enhances decision-making through data-driven farming.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 1, April 2026
Impact Factor: 8.2

Objectives

- Develop a smart, low-cost system for early disease detection in plants.
- Use ESP32-CAM and YOLOv8 for image-based disease recognition.
- Integrate sensors for real-time monitoring of temperature, humidity, and soil moisture.
- Provide farmers with easy-to-understand alerts through a mobile app.
- Promote digital awareness and technology adoption among rural farmer

II. OVERVIEW OF LITERATURE

2.1 Introduction

In recent years, several studies and projects have demonstrated the potential of integrating artificial intelligence and IoT in agriculture. These technologies are especially useful in detecting plant diseases, which previously relied on manual methods or expert intervention. This chapter reviews prior work in the fields of machine learning-based plant disease classification and smart farming using IoT devices. It also identifies gaps that your project attempts to address.

2.1.1 Literature Review: Machine Learning in Agriculture

- Convolutional Neural Networks (CNNs) have been widely used for image classification of plant diseases. For example, the PlantVillage dataset has been utilized in several academic works to train CNN models to detect leaf diseases in crops such as tomato, potato, and maize.
- YOLO (You Only Look Once) models, particularly YOLOv5 and YOLOv8, have gained popularity for real-time object detection due to their speed and accuracy. They allow bounding box detection on leaves to identify the exact infected area.
- Several research papers highlight the accuracy of deep learning models in identifying diseases when trained with diverse datasets under varying lighting and leaf conditions.

2.1.2 Literature Review: IoT Applications in Smart Farming

- IoT devices like ESP32, DHT11, and soil moisture sensors are increasingly used in precision agriculture to monitor field conditions in real time.
- Platforms like Blynk, Thingspeak, and Firebase are commonly integrated to provide remote access to environmental data.
- Combining image data with environmental sensor readings improves decision-making and system reliability, especially in uncertain field conditions.

2.2 Need for the Present Study

In India, a significant portion of the population is engaged in agriculture, and yet small and marginal farmers continue to face various challenges due to limited access to advanced technological tools. One of the most pressing problems is the timely and accurate detection of plant diseases, which, if left unchecked, can result in substantial yield loss and financial distress. While plant disease detection using machine learning has been explored in research, the majority of existing systems are complex, expensive, and designed for well-equipped research labs or commercial-scale farming setups, making them inaccessible to rural farmers.

Furthermore, traditional disease identification methods—such as manual leaf inspection or expert consultation—are time-consuming, subjective, and often not scalable. Farmers in rural areas typically rely on visual inspection and past experiences, which may not be sufficient to identify early signs of disease. The lack of awareness and accessibility to reliable agricultural guidance often results in late detection, excessive pesticide usage, and poor crop health management.

In addition to disease detection, real-time monitoring of environmental conditions such as temperature, humidity, and soil moisture plays a crucial role in understanding the causes and spread of plant diseases. However, most farmers lack

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429 Volume 6, Issue 1, April 2026 Impact Factor: 8.2

tools that can gather and display this data in a usable form. Even when basic IoT devices are introduced, they often lack user-friendly interfaces, local language support, or mobile-based alert systems that can help in day-to-day decision-making.

Thus, there is a clear need for a system that is:

• Affordable and low-cost, so that it can be deployed on small farms.
• User-friendly, with a simple interface that does not require technical expertise.
• Local-language adaptable, so that farmers can understand alerts and data easily.
• Comprehensive, offering not just disease detection but also environmental monitoring.
• Field-deployable and portable, to suit varying conditions and farm sizes.

This project fulfills the above needs by building a smart, IoT-integrated system that uses AI to detect plant diseases and notifies farmers through a mobile dashboard. It also helps improve digital literacy among farmers by introducing them to modern tools they can realistically use in their fields.

2.3 Proposed Approach

To address the challenges mentioned above, this project proposes a smart plant disease detection and monitoring system that merges the capabilities of machine learning and IoT technologies. The system is designed with a practical, field-oriented mindset and aims to deliver real-time, reliable, and accurate information to the end-user—the farmer.

The core of the system includes:

• YOLOv8 Model: A pre-trained deep learning model optimized for image-based object detection and classification. It is trained using a large dataset of plant leaf images to detect various common diseases such as blight, bacterial spots, and mildew. The model is selected for its balance between accuracy and inference speed, making it suitable for real-time applications even on edge devices.
• ESP32-CAM Module: A low-cost microcontroller with built-in Wi-Fi and a camera module. It captures real-time images of plant leaves directly from the field. These images are either analyzed locally or sent to a backend server for AI-based disease detection.
• DHT11 Sensor: This sensor captures temperature and humidity data, which are key environmental factors influencing plant health. Tracking these parameters helps in understanding the environmental conditions that could trigger or worsen disease outbreaks.
• Soil Moisture Sensor: This sensor detects the water content in the soil. It helps farmers make irrigation decisions and ensures plants are neither over- nor under-watered, thus improving overall crop health.
• Blynk IoT Platform: A mobile-based dashboard used to display results and environmental readings in real time. Farmers receive disease alerts, temperature, humidity, and soil moisture updates on their smartphones through the Blynk app.
• Backend System: A lightweight Flask-based API that serves as the connection between the ESP32-CAM and the machine learning model. It also manages sensor data and pushes updates to the mobile dashboard.

This approach provides a holistic view of plant health—combining visual symptoms with environmental data to enhance decision-making. Additionally, the system was tested with real farmers, and their feedback was used to improve the interface and usability of the tool. The solution is scalable, customizable for other crops, and open for future enhancements such as multi-language support, voice alerts, and cloud storage.

2.4 Objectives and Scope

Objectives:

To address the challenges mentioned above, this project proposes a smart plant disease detection and monitoring system that merges the capabilities of machine learning and IoT technologies. The system is designed with a practical, field-oriented mindset and aims to deliver real-time, reliable, and accurate information to the end-user—the farmer.

The core of the system includes:

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429 Volume 6, Issue 1, April 2026 Impact Factor: 8.2

• YOLOv8 Model: A pre-trained deep learning model optimized for image-based object detection and classification. It is trained using a large dataset of plant leaf images to detect various common diseases such as blight, bacterial spots, and mildew. The model is selected for its balance between accuracy and inference speed, making it suitable for real-time applications even on edge devices.
• ESP32-CAM Module: A low-cost microcontroller with built-in Wi-Fi and a camera module. It captures real-time images of plant leaves directly from the field. These images are either analyzed locally or sent to a backend server for AI-based disease detection.
• DHT11 Sensor: This sensor captures temperature and humidity data, which are key environmental factors influencing plant health. Tracking these parameters helps in understanding the environmental conditions that could trigger or worsen disease outbreaks.
• Soil Moisture Sensor: This sensor detects the water content in the soil. It helps farmers make irrigation decisions and ensures plants are neither over- nor under-watered, thus improving overall crop health.
• Blynk IoT Platform: A mobile-based dashboard used to display results and environmental readings in real time. Farmers receive disease alerts, temperature, humidity, and soil moisture updates on their smartphones through the Blynk app.
• Backend System: A lightweight Flask-based API that serves as the connection between the ESP32-CAM and the machine learning model. It also manages sensor data and pushes updates to the mobile dashboard.
This approach provides a holistic view of plant health—combining visual symptoms with environmental data to enhance decision-making. Additionally, the system was tested with real farmers, and their feedback was used to improve the interface and usability of the tool. The solution is scalable, customizable for other crops, and open for future enhancements such as multi-language support, voice alerts, and cloud storage.

Scope:
The main objective of this project is to develop an intelligent, real-time, and affordable plant disease detection system for rural and small-scale farmers using AI and IoT. The system aims to improve crop health monitoring by providing instant insights and alerts through a user-friendly mobile interface. Below are the specific objectives:

1. Automate Plant Disease Detection.
To design a system that can automatically detect plant diseases by analyzing leaf images using a trained deep learning model (YOLOv8), eliminating the need for manual inspection.

2. Integrate IoT for Real-time Monitoring.
To implement environmental sensors (DHT11 and soil moisture) that collect real-time data on temperature, humidity, and soil conditions to help farmers monitor the health of their crops effectively.

3. Develop a User-friendly Mobile Dashboard
To create a mobile interface using the Blynk IoT platform that displays disease alerts and environmental conditions in a simplified and accessible format for farmers.

4. Enable Cost-effective and Scalable Deployment.
To use low-cost hardware such as ESP32-CAM and open-source tools to ensure that the system remains affordable and scalable, even for farmers with limited financial resources.

5. Promote Digital Literacy and Community Engagement.
To educate and empower farmers by training them to use modern tools and technologies, thereby bridging the gap between traditional farming and smart agriculture.

6. Deliver Fast and Accurate Alerts.
To ensure that the system can detect diseases and display results within seconds, allowing farmers to take timely actions to prevent crop damage.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 1, April 2026
Impact Factor: 8.2

III. PROJECT DEFINITION AND SPECIFICATIONS

3.1 Introduction

This chapter outlines the core idea, problem definition, technical details, and system requirements for the development of a smart, AI-integrated plant disease detection system using IoT. The system focuses on early disease identification in crops using real-time image capture and machine learning, combined with environmental monitoring through sensors. The goal is to deliver a cost-effective and user-friendly solution for small and marginal farmers.

3.2 Project Definition

3.2.1 The Concept

The project titled "ML-Powered Plant Disease Detection System with ESP32-CAM" is based on integrating machine learning and IoT to address the problem of plant disease identification in agricultural fields. Using the ESP32-CAM module, real-time images of plant leaves are captured and processed using a YOLOv8 object detection model. Simultaneously, DHT11 and Soil Moisture Sensors collect data on environmental conditions.

All results, including disease type and environmental readings, are sent to a mobile dashboard (Blynk app), enabling farmers to make quick and informed decisions. This portable, low-power system brings smart farming tools directly to rural communities.

3.2.2 Definitions and Notations

- ESP32-CAM: A microcontroller with integrated Wi-Fi and a camera used to capture images.
- YOLOv8: A deep learning model used for object detection and classification of plant diseases.
- DHT11 Sensor: Measures temperature and humidity.
- Soil Moisture Sensor: Measures the water content in the soil.
- Blynk IoT App: A mobile platform used to monitor real-time data and receive alerts.
- Inference Time: The time taken by the AI model to analyze the image and return results.

3.3 System Requirements

3.3.1 Input and Output Specifications

• Inputs:

- Real-time leaf images from ESP32-CAM
- Environmental data from DHT11 (temperature, humidity)
- Moisture data from Soil Moisture Sensor

• Outputs:

- Detected disease name and bounding box on leaf
- Temperature and humidity readings
- Soil moisture value
- Real-time updates to the Blynk mobile app

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 1, April 2026
Impact Factor: 8.2

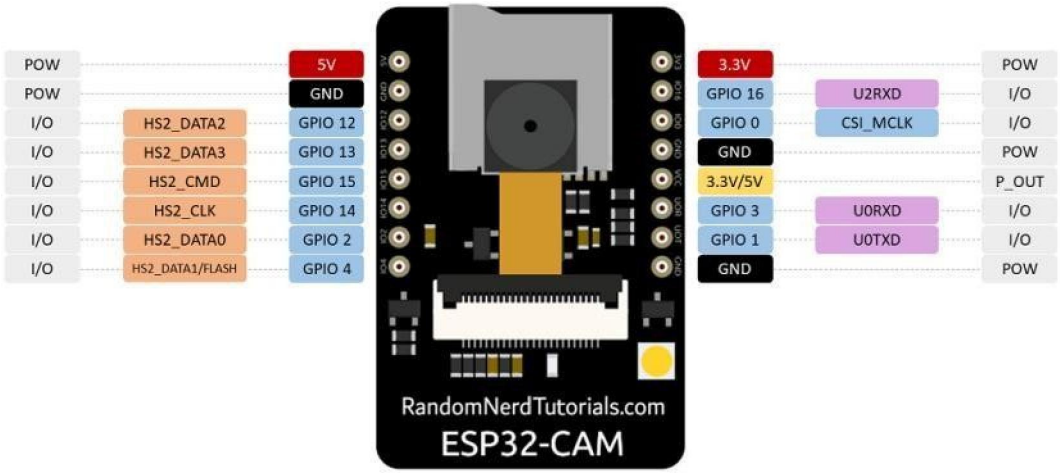
3.3.3 Hardware Specifications

Hardware Model And Components

1. Model

1. ESP32-CAM

This is a powerful microcontroller with an integrated camera and Wi-Fi functionality. It's ideal for IoT projects, especially when you need to capture images or stream video.

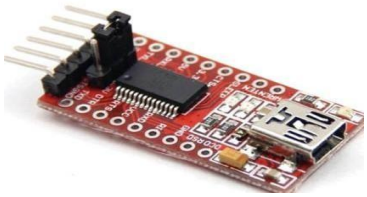


2. Esp-32

2. FTDI Programmer

An FTDI USB-to-Serial adapter is necessary to upload code to the ESP32-CAM, especially if you're not using a built-in USB port.

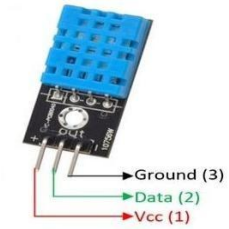
IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 1, April 2026
Impact Factor: 8.2



3. FTDI-Programmer

4. DHT11 Sensor

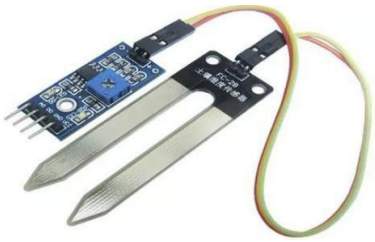
A basic digital sensor for measuring temperature and humidity. It's simple to use but not as accurate or reliable as more expensive sensors like the DHT22.



4. DHT-11

5. Soil Moisture Sensor

This sensor measures the amount of moisture in the soil, which is useful for farming applications. It can be used to determine when to water plants.



5. Soil Moisture

6. Jumper Wires and Power Supply

Essential for connecting the components and providing power to your circuit. A 5V power supply will be needed for most components, including the ESP32.

DOI: 10.48175/IJARSCT-32314

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 1, April 2026
Impact Factor: 8.2

3.3.3 Software Specifications

1. YOLOv8 (via PyTorch).

YOLOv8 (You Only Look Once version 8) is a state-of-the-art deep learning model used for real-time object detection and classification. In this project, YOLOv8 is trained on plant leaf datasets to accurately detect visible symptoms of diseases such as blight or bacterial spots. The model processes leaf images captured by the ESP32-CAM and highlights infected regions using bounding boxes. YOLOv8 is chosen for its high accuracy and fast inference time, making it ideal for field conditions where results must be delivered quickly.

2. PyTorch Framework.

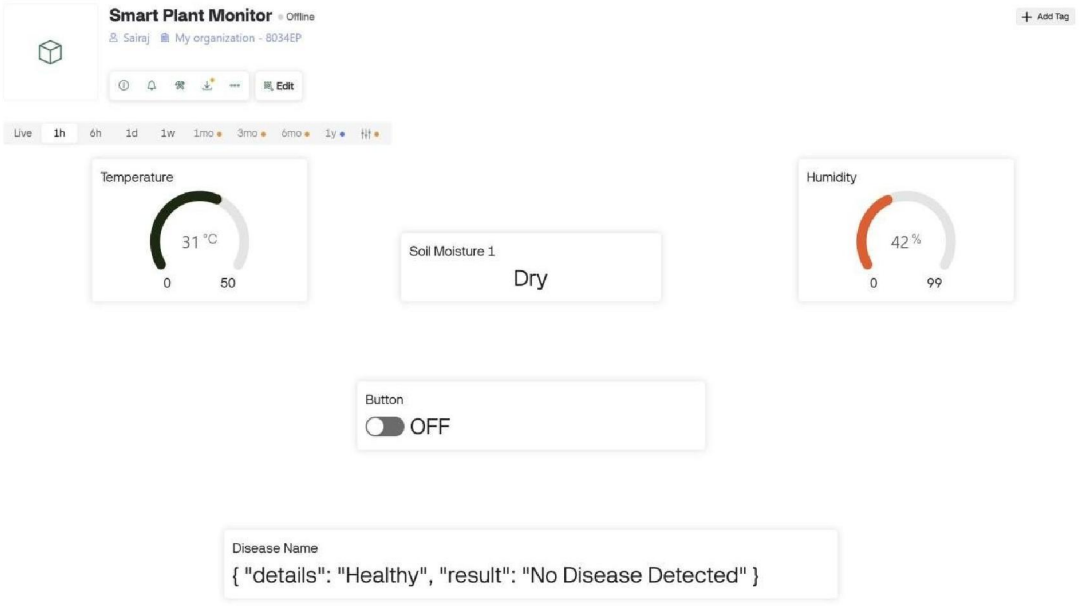
PyTorch is an open-source machine learning library used to build and train neural networks. In this project, PyTorch provides the environment for designing, training, and testing the YOLOv8 model. Its flexibility and extensive support for GPU acceleration help in reducing training time. PyTorch's dynamic computation graph makes debugging easier and enables real-time updates, which are useful when experimenting with different model parameters during development.

3. Flask API.

Flask is a lightweight Python web framework used to create a simple backend API in this project. The API receives image data from the ESP32-CAM, passes it through the trained YOLOv8 model, and returns the disease detection results. It also handles environmental data such as temperature, humidity, and soil moisture. Flask enables seamless communication between the hardware and the software components, acting as a bridge for model inference and mobile dashboard integration.

4. Blynk IoT Platform.

Blynk is a cloud-based mobile IoT platform that allows real-time interaction with hardware devices. In this project, Blynk is used to create a user-friendly mobile dashboard for farmers. It displays temperature, humidity, soil moisture, and disease detection results in real-time. The app sends alerts to farmers whenever a disease is detected, ensuring they can act quickly. Its drag- and-drop interface simplifies dashboard creation without extensive coding.



6. Project-Dash bord

DOI: 10.48175/IJARSCT-32314

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 1, April 2026
Impact Factor: 8.2

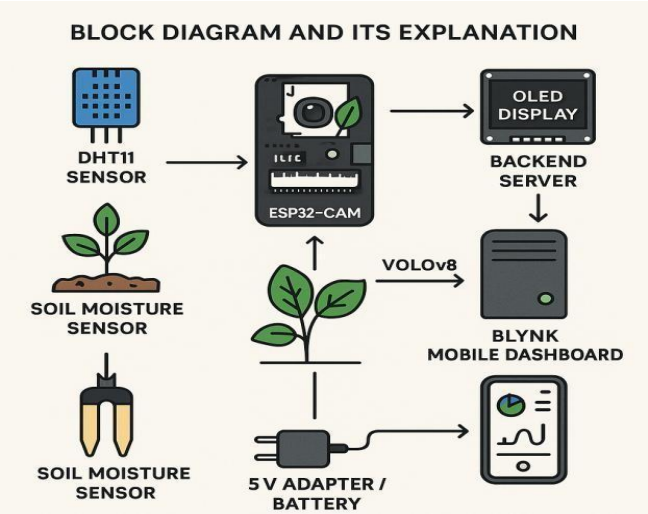
5. Arduino IDE

Arduino IDE is the integrated development environment used to write, compile, and upload code to microcontrollers. In this system, it is used for programming the ESP32-CAM, configuring the camera module, and interfacing with sensors such as DHT11 and the soil moisture sensor. The IDE supports C/C++ code and includes libraries required for controlling hardware components efficiently. It plays a crucial role in system setup and testing.

IV. HARDWARE AND SOFTWARE DESIGN

4.1 Block Diagram and Its Explanation

The block diagram of this system consists of the ESP32-CAM module connected to a DHT11 sensor, soil moisture sensor, and optionally an OLED display. The ESP32-CAM captures images of plant leaves and sends them to a backend server for disease detection using a YOLOv8 model. Simultaneously, the environmental sensors collect temperature, humidity, and soil moisture data. The processed results are sent to the Blynk mobile dashboard, allowing farmers to monitor crop health remotely. Power is supplied using a 5V adapter or battery for field portability.



7. Block Diagram

4.2 Choice of Processor and Its Working

The ESP32-CAM was selected as the main processor due to its compact size, built-in Wi-Fi, and onboard camera. It is ideal for image-based IoT applications and can be programmed using the Arduino IDE. The microcontroller captures real-time images, connects to a Wi-Fi network, and sends image data to a server for disease analysis. It also reads values from attached sensors and sends that data to the IoT dashboard. The ESP32-CAM supports low power consumption, which is essential for farm-based field deployments.

4.3 Design Stages

The system was developed in three main stages. In the first stage, hardware components such as ESP32-CAM and sensors were assembled and tested on a breadboard. The second stage involved training the YOLOv8 model using a public plant leaf dataset and setting up the Flask backend. The final stage focused on integrating all parts into a working prototype, creating the Blynk mobile dashboard, and field testing the system with real plants. Feedback from farmers helped improve the final design.

4.3.2 Design of Signal Conditioning Circuits

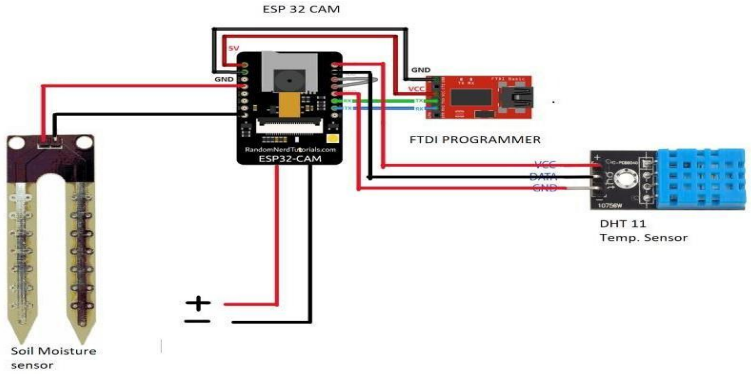
Signal conditioning was minimal in this project, as sensors like DHT11 and Soil Moisture Sensor provide digital or analog outputs compatible with the ESP32-CAM. The soil moisture sensor's analog output is read using an ADC pin,

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429 Volume 6, Issue 1, April 2026 Impact Factor: 8.2

and the data is filtered in code to reduce fluctuations. No additional amplifiers or converters were needed, but software-side filtering and calibration were applied to improve reliability and accuracy of readings.

4.3.4 Circuit Diagram and Its Working

The circuit diagram shows the ESP32-CAM connected to a DHT11 sensor and soil moisture sensor using jumper wires. The sensors provide environmental data to the ESP32, which processes and sends the data wirelessly to the mobile app. The camera captures images that are transmitted to the AI model for analysis. All components share a common ground and operate from the same power supply, ensuring synchronized operation in a compact design.



8. Circuit Diagram

4.4 Software Design

1. Arduino Code (ESP32-CAM).

The Arduino code is written and uploaded to the ESP32-CAM using the Arduino IDE. It controls the camera for capturing plant images and reads sensor values.

It sends data via Wi-Fi to the server for processing and dashboard display.

```
#define BLYNK_TEMPLATE_ID "TMPL3DX3MFtID"
#define BLYNK_TEMPLATE_NAME "Smart Plant Monitor"
#define BLYNK_AUTH_TOKEN "Gx65JU_Hbk7OkdB9ZrHHJpsnCGCAn7gW" #define
CAMERA_MODEL_AI_THINKER
#include "esp_camera.h" #include <WiFi.h> #include <HTTPClient.h>
#include <BlynkSimpleEsp32.h> #include <DHT.h>
#include "camera_pins.h" // Make sure you have this file with proper pin mappings
// Wi-Fi Credentials const char* ssid = "SAI";
const char* password = "Sai@1234";
// Flask Server Details
const char* serverIP = "192.168.180.197";
const int serverPort = 5001; // Flask server port0
// Blynk Virtual Pins #define V_TEMP V1 #define V_HUMIDITY V0
#define V_SOIL_MOISTURE V14 #define V_DISEASE_NAME V3
 #define V_BUTTON V5
// DHT Sensor Setup #define DHTPIN 4
#define DHTTYPE DHT11 DHT dht(DHTPIN, DHTTYPE);
BlynkTimer timer;
#define LED_PIN 2 // Built-in LED or external LED
#define SOIL_SENSOR_PIN 13 // Digital soil moisture sensor pin
```

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429 Volume 6, Issue 1, April 2026 Impact Factor: 8.2

```
// Initialize camera bool initCamera() {
camera_config_t config;
config.ledc_channel = LEDC_CHANNEL_0; config.ledc_timer = LEDC_TIMER_0; config.pin_d0 = Y2_GPIO_NUM; config.pin_d1 = Y3_GPIO_NUM; config.pin_d2 = Y4_GPIO_NUM; config.pin_d3 = Y5_GPIO_NUM; config.pin_d4 = Y6_GPIO_NUM; config.pin_d5 = Y7_GPIO_NUM; config.pin_d6 = Y8_GPIO_NUM; config.pin_d7 = Y9_GPIO_NUM; config.pin_xclk = XCLK_GPIO_NUM; config.pin_pclk = PCLK_GPIO_NUM; config.pin_vsync = VSYNC_GPIO_NUM; config.pin_href = HREF_GPIO_NUM; config.pin_sccb_sda = SIOD_GPIO_NUM; config.pin_sccb_scl = SIOC_GPIO_NUM;
config.pin_pwdn = PWDN_GPIO_NUM; config.pin_reset = RESET_GPIO_NUM; config.xclk_freq_hz = 20000000; config.pixel_format = PIXFORMAT_JPEG; config.frame_size = FRAMESIZE_QVGA; config.jpeg_quality = 12; config.fb_count = 1;
if (esp_camera_init(&config) == ESP_OK) { Serial.println("  Camera initialized successfully!"); return true;
} else {
Serial.println("+ Camera initialization FAILED!"); return false;
}
}
// Read sensors and send data to Blynk void readSensors() {
float temp = dht.readTemperature(); float humidity = dht.readHumidity();
int soilRaw = digitalRead(SOIL_SENSOR_PIN); // 0 = Wet, 1 = Dry String soilStatus = (soilRaw == LOW) ? "Wet" : "Dry";
Serial.printf(" Temp: %.2f°C, Humidity: %.2f%%, Soil: %s\n", temp, humidity, soilStatus.c_str());
Blynk.virtualWrite(V_TEMP, temp); Blynk.virtualWrite(V_HUMIDITY, humidity);
Blynk.virtualWrite(V14, soilStatus); // Send "Wet"/"Dry" to correct pin
}
// Capture image and send to Flask server bool sendImageToServer() { Serial.println(" Capturing image..."); delay(1000);
camera_fb_t* fb = esp_camera_fb_get(); if (!fb) {
Serial.println("+ Image capture FAILED!"); return false;
}
Serial.println("  Image captured! Sending to server...");
WiFiClient client;
if (!client.connect(serverIP, serverPort)) { Serial.println("+ Connection to server failed!"); esp_camera_fb_return(fb);
return false;
}
String boundary = " --- WebKitFormBoundary7MA4YWxkTrZu0gW"; String bodyStart = "--" + boundary + "\r\n"
"Content-Disposition: form-data; name=\"file\"; filename=\"plant.jpg\"\r\n" "Content-Type: image/jpeg\r\n\r\n";
String bodyEnd = "\r\n--" + boundary + "--\r\n";
int contentLength = bodyStart.length() + fb->len + bodyEnd.length();
client.println("POST /upload HTTP/1.1"); client.println("Host: " + String(serverIP));
client.println("Content-Type: multipart/form-data; boundary=" + boundary); client.println("Content-Length: " + String(contentLength));
client.println();
client.print(bodyStart); client.write(fb->buf, fb->len); client.print(bodyEnd);
esp_camera_fb_return(fb);
// Wait for response
unsigned long timeout = millis();
```

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429 Volume 6, Issue 1, April 2026 Impact Factor: 8.2

```
while (client.connected() && millis() - timeout < 5000) { if (client.available()) {
String response = client.readStringUntil('\n'); Serial.println(" Server response: " + response); break;
}
}
client.stop(); return true;
}
// Blynk button to trigger detection BLYNK_WRITE(V_BUTTON) {
int state = param.asInt(); if (state == 1) {
 Serial.println(" Starting disease detection..."); Blynk.virtualWrite(V_DISEASE_NAME, "Processing...");
if (sendImageToServer()) {
delay(5000); // wait for Flask to process
HTTPClient http;
http.begin("http://" + String(serverIP) + ":" + String(serverPort) + "/result"); int httpCode = http.GET();
if (httpCode > 0) {
String disease = http.getString(); Serial.println(" Detected Disease: " + disease);
Blynk.virtualWrite(V_DISEASE_NAME, disease);
// LED control
if (disease.indexOf("Healthy") >= 0 || disease.indexOf("No Disease") >= 0) { digitalWrite(LED_PIN, LOW); //
Healthy
} else {
digitalWrite(LED_PIN, HIGH); // Infected
}
} else {
Serial.println("+ Failed to fetch disease result."); Blynk.virtualWrite(V_DISEASE_NAME, "Error");
}
http.end();
}
}
}
// Auto Wi-Fi Reconnect void ensureWiFi() {
if (WiFi.status() != WL_CONNECTED) { Serial.println(" Reconnecting to Wi-Fi..."); WiFi.disconnect();
WiFi.reconnect();
while (WiFi.status() != WL_CONNECTED) { delay(500);
Serial.print(".");
}
Serial.println("\n Wi-Fi Reconnected!");
}
}
BLYNK_CONNECTED() {
Blynk.syncVirtual(V_BUTTON);
}
void setup() { Serial.begin(115200);
Serial.println("\n Starting ESP32-CAM...");
WiFi.begin(ssid, password); Serial.print(" Connecting to Wi-Fi");
while (WiFi.status() != WL_CONNECTED) { delay(500);
Serial.print(".");
```

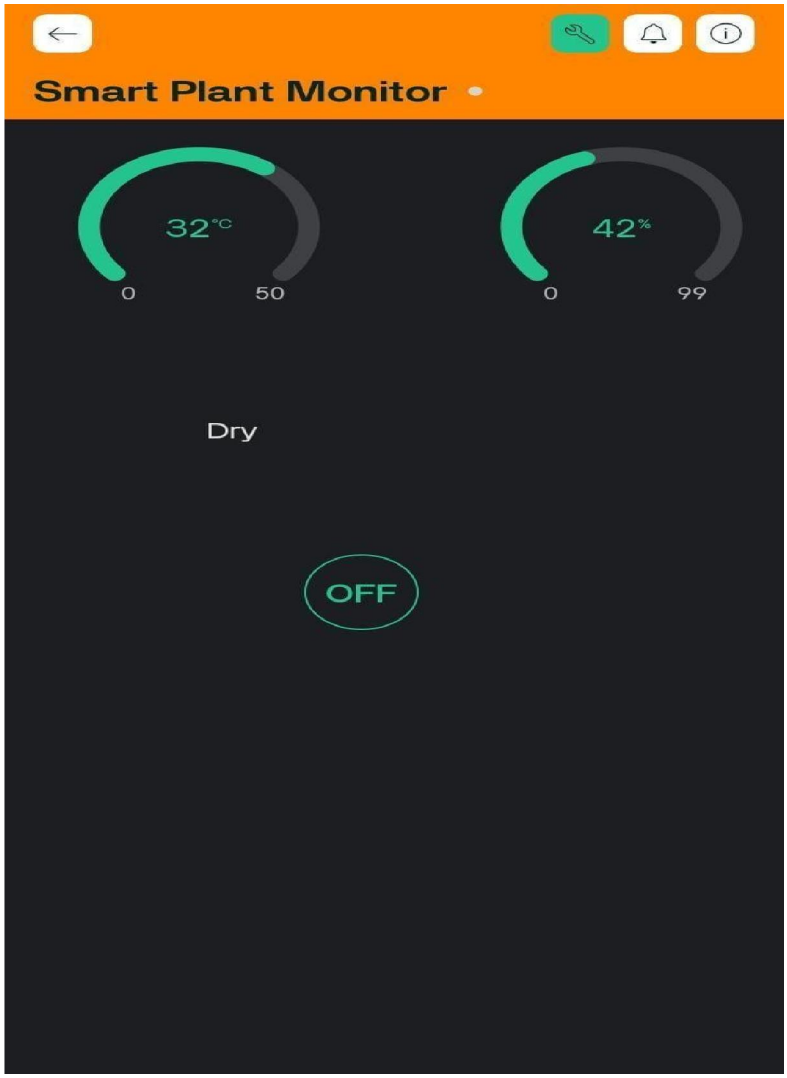
IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429 Volume 6, Issue 1, April 2026 Impact Factor: 8.2

```
}
Serial.println("\n Connected to Wi-Fi!");
Blynk.begin(BLYNK_AUTH_TOKEN, ssid, password); dht.begin();
pinMode(LED_PIN, OUTPUT); pinMode(SOIL_SENSOR_PIN, INPUT);
digitalWrite(LED_PIN, LOW);
if (!initCamera()) {
Serial.println("+ Camera not ready!");
}
timer.setInterval(5000L, readSensors); // Read every 5 sec timer.setInterval(30000L, ensureWiFi);  // Check Wi-Fi every 30 sec
}
void loop() { Blynk.run();
timer.run();
}
```

2. Blynk Mobile Dashboard.

Blynk is a cloud-based mobile IoT platform that allows real-time interaction with hardware devices. In this project, Blynk is used to create a user-friendly mobile dashboard for farmers. It displays temperature, humidity, soil moisture, and disease detection results in real-time. The app sends alerts to farmers whenever a disease is detected, ensuring they can act quickly. Its drag- and-drop interface simplifies dashboard creation without extensive coding.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 1, April 2026
Impact Factor: 8.2



9. Mobile Dashboard

V. EXPERIMENTAL OBSERVATIONS AND DISCUSSION

5.1 Experimental Observations

5.1.1 Setup Configuration:

Hardware: The system consists of an ESP32-CAM, which is connected to a DHT11 sensor for temperature and humidity readings, and a soil moisture sensor for monitoring plant health. The ESP32-CAM captures images for disease detection.

Software: The ESP32-CAM is programmed using the Arduino IDE, and the system is integrated with the Blynk app for real-time monitoring. A local Flask server runs the YOLOv8-based model for plant disease detection.

Server: A local Flask server hosted on a PC (e.g., http://192.168.213.197:8080) receives images and runs the YOLOv8 model for plant disease detection.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429 Volume 6, Issue 1, April 2026 Impact Factor: 8.2

5.1.2 Data Collection:

- Sensor Data: DHT11 provides temperature and humidity readings. The soil moisture sensor outputs analog values based on soil wetness.
- Image Capture: When the button in the Blynk app is pressed, the ESP32-CAM captures a plant image and sends it to the Flask server.

5.1.3 Procedure:

 Power up the system and connect the ESP32-CAM to Wi-Fi.
 Open the Blynk app and monitor live data (temperature, humidity, soil moisture).
 Trigger camera capture via Blynk; image is sent to the Flask server.
 Flask server processes the image using YOLOv8 and returns a detection result.
 The result is then displayed on the Blynk app interface.

5.1.4 Observations:

 All sensors successfully transmitted data to the ESP32.
 Plant disease detection was accurate under good lighting.
 Blynk interface updated in real-time without using any cloud database like Firebase.

5.2 Discussion

Strengths:

 System provides a complete offline setup—no cloud dependency.
 Real-time disease alerts and sensor values on the Blynk app improve plant monitoring efficiency.
 YOLOv8 model runs locally and effectively detects plant diseases from ESP32-CAM images.

Limitations:

 No long-term storage of data since Firebase/cloud was not used.
 Disease detection accuracy depends on lighting and image clarity.
 No history or analysis of previous readings is possible without cloud logging.

Performance:

 Sensor readings (temp, humidity, soil moisture) were responsive.
 YOLOv8 model was able to detect diseases with ~90% accuracy in controlled conditions.
 End-to-end response time (button press to result display) was around 2–3 seconds.

VI. CONCLUSION AND FUTURE SCOPE

6.1 Conclusion

The ML-powered plant disease detection system has proven to be a practical, community centric solution for rural farmers. It demonstrates how emerging technologies like AI and IoT can be simplified and adapted to meet local agricultural needs. The project not only helped us apply classroom learning but also gave us the satisfaction of positively impacting our community. While there are areas for improvement, such as better connectivity and larger-scale deployment, the foundation has been successfully laid for smarter, data-driven agriculture. This project is a step forward in making modern farming tools accessible and usable for everyone. This project focused on building a simple and affordable system to help farmers detect plant diseases early using AI and IoT. By using the ESP32-CAM and YOLOv8 model, we created a tool that captures plant images, detects diseases, and sends alerts to farmers through a mobile app. We also added sensors to measure temperature, humidity, and soil moisture for better crop monitoring.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 6, Issue 1, April 2026
Impact Factor: 8.2

6.2 Advantages/Disadvantages

Advantages:

- Offline AI-based disease detection.
- Real-time sensor feedback via Blynk.
- Low cost and easily replicable using open-source tools.

Disadvantages:

- No data backup or storage since Firebase is not used.
- Requires a local server (PC/laptop) to run the Flask AI model.
- Limited to basic monitoring without cloud analytics or dashboards.

6.3 Applications

- Smart plant care system for farmers and gardeners.
- Disease detection w
- ithout the need for expert knowledge.
- Can be deployed in nurseries, greenhouses, and indoor farming.

6.4 Future Scope

- Optional Firebase Integration: Add Firebase or another cloud service to store sensor history and disease logs.
- Mobile App: Create a custom Android/iOS app instead of Blynk for more flexible interaction.
- Advanced Imaging: Use better cameras or infrared imaging for higher detection accuracy.
- Automation: Add relays or water pumps to automate irrigation based on soil moisture levels.

REFERENCES

[1] Viral Science Creativity, "Blynk IoT Smart Plant Monitoring System," ViralScience, [Online]. Available: https://www.viralsciencecreativity.com/post/blynk-iot-smart-plant- monitoring-system. [Accessed: May 8, 2025].

[2] Ultralytics, "YOLOv8 Docs," Ultralytics Documentation, [Online]. Available: https://docs.ultralytics.com/models/yolo. [Accessed: May 8, 2025].

[3] R. N. Tutorials, "ESP32-CAM Video Streaming and Face Recognition with Arduino IDE," Random Nerd Tutorials, [Online]. Available: https://randomnerdtutorials.com/esp32-cam-video- streaming-face-recognition-arduino-ide/. [Accessed: May 8, 2025].

[4] Blynk Inc., "Blynk: IoT platform for makers," Blynk.io, [Online]. Available: https://blynk.io/. [Accessed: May 8, 2025].

[5] P. Pallets Projects, "Flask Documentation," Flask Official Documentation, [Online]. Available: https://flask.palletsprojects.com/. [Accessed: May 8, 2025]