Talking Plant Automatic System Using ESP32

Microprocessor Systems and Interfacing

CEP Report

AYESHA BUTT	**CIIT//FA22-BCE-053/LHR**
MAHA CHAUDHARY	**CIIT//FA22-BCE-082/LHR**
RANA SAAD SAFADAR	**CIIT//FA22-BCE-093/LHR**

Project Supervisor: Engr. USMAN RAFIQ

Spring-2025

Department of Electrical & Computer Engineering

COMSATS UNIVERSITY ISLAMABAD

LAHORE CAMPUS, PAKISTAN

TABLE OF CONTENTS

TABLE OF FIGURES

ABSTRACT

In the era of smart agriculture, the “Talking Plant” project offers a novel approach to plant health monitoring by integrating IoT and voice response systems. This system is designed to monitor essential parameters of plant growth, namely moisture level, temperature, and humidity—using DHT11 and soil moisture sensors interfaced with an ESP32 microcontroller. A DF-Player Mini module is used to generate human-like voice responses that inform the user about the plant's needs. For example, if the soil moisture is low, the plant “speaks” to alert the user to water it. This project is especially useful for hydroponic systems where soilless cultivation requires precise environmental control. By providing real-time voice feedback, this project helps users with little agricultural knowledge to maintain plant health efficiently. The cost-effectiveness and scalability of the system make it suitable for both home gardening and larger greenhouse setups. The prototype has been simulated and tested successfully using hardware components and produces consistent and reliable feedback.

.

1. ***INTRODUCTION***

The increasing demand for smart farming solutions, especially in home gardening and urban agriculture, calls for innovative, interactive systems. Traditional irrigation methods often lead to overwatering or underwatering due to lack of real-time feedback. This project introduces an intelligent, voice-enabled system that bridges the communication gap between plants and humans using sensors, IoT, and audio feedback mechanisms. It encourages smart gardening by ensuring that plants receive the right amount of water at the right time without requiring constant human supervision. Additionally, the use of voice alerts adds a unique emotional layer, making users more responsive and engaged with their plants. The ESP32 acts as the central controller, integrating inputs from sensors and triggering appropriate audio messages and irrigation actions. The system can be scaled for various applications, from small home setups to urban rooftop gardens. Its modular nature allows for enhancements such as remote monitoring through mobile apps or cloud integration.

Figure 1 *Internet of Things Based Smart Farming* ***[1]***

1.1. ***MOTIVATION***

Plants are living beings that require care, but they can't communicate their needs. By giving a plant a voice, we make it more relatable and easier to care for. This concept also promotes awareness about efficient water usage, sustainability, and plant well-being.

Moreover, in today's fast-paced world, people often forget to water their plants or lack the knowledge of when and how much to water them. This leads to plant neglect and wastage of water. The motivation behind this project is to make plant care more efficient and engaging by transforming the silent needs of plants into audible alerts. It aims to create a bond between humans and plants through technological innovation, making gardening a more intuitive and accessible experience for everyone, including children, busy professionals, and the elderly.

1.2. ***AIMS AND OBJECTIVES***

The aim is to automate the irrigation process while making the experience interactive and educational. This project focuses on using IoT and voice interaction to create a smarter plant care ecosystem.

1.2.1 OBJECTIVES

- To build an ESP32-based irrigation system that intelligently responds to plant needs.
- To monitor soil moisture, temperature, and humidity levels in real-time.
- To activate a water pump when moisture levels fall below a certain threshold.
- To use DF-Player Mini for playing pre-recorded voice alerts that represent the plant's status.
- To reduce water wastage by watering only when needed.
- To foster environmental responsibility and awareness through smart plant care.
- To encourage the adoption of home gardening and urban agriculture using smart tools.
- To enable future integration with mobile or cloud platforms for remote monitoring.

1.3. ***REPORT ORGANIZATION***

1. Section 1: Introduction and background
2. Section 2: Review of existing related projects
3. Section3:Implementationdetails including hardware, schematics.
4. Section 4: Real-world use cases
5. Section 5: Social contribution
6. Section 6: Environmental impact
7. Section 7: Conclusion and future work

2. ***RELATED WORK***

Existing systems utilize technologies like soil moisture sensors and IoT for irrigation. This project differentiates itself by integrating advanced features such as:

1. **Energy-Efficient Design:**
    - Optimizes resource usage to reduce power consumption, making it more sustainable and cost-effective.
2. **Real-Time Monitoring and Adaptive Watering:**
    - Provides precise, data-driven irrigation schedules, responding dynamically to soil moisture levels for better efficiency and plant health.
3. **Voice Feedback Integration:**
    - Offers an emotionally engaging, conversational interaction with the plant, which most existing systems lack.
4. **Educational Utility:**
    - Designed to serve as an educational tool for understanding embedded systems, environmental science, and sustainable practices.

Previous research in IoT-based plant monitoring emphasizes sensor accuracy and remote control. This Talking Plant project advances the field by combining sensory data with voice feedback, filling a significant interactional gap in automated gardening solutions.

3. ***PROJECT IMPLEMENTATION***

The system is implemented using an ESP32 board that receives data from various sensors and triggers corresponding actions. When soil moisture falls below a threshold, a relay module turns on a water pump. Simultaneously, the DF-Player Mini activates a voice message informing the user that the plant needs water.

1. Components:

Component	Description
ESP32 Node-MCU (30-pin, CP2102)	Core microcontroller used for processing sensor data and controlling outputs
DHT22 Temperature & Humidity Sensor	Measures ambient temperature and humidity with higher precision than DHT11
Soil Moisture Sensor	Detects moisture level in soil to determine when watering is needed
LDR (Light Dependent Resistor)	Measures light intensity to assess plant light exposure
DF-Player Mini MP3 Module	Plays pre-recorded voice messages stored on SD card
Speaker (Connected to DF-Player)	Outputs the plant's "voice" messages
8GB Micro SD Card	Stores MP3 files for DF-Player Mini
OLED Display	Displays sensor values like temperature, humidity, and moisture in real-time
Button 6x6x13.5 mm	Used for manual triggering of voice or irrigation functions
1/4W Resistors	Used for voltage division and current limiting (e.g., with LDR or buttons)
Breadboard	For prototyping and organizing component connections
Jumper Wires	For electrical connections between modules
Power Supply (Adapter)	Provides required 5V–12V DC power to all modules

2. *System Workflow:*

- Sensors measure soil moisture, temperature, humidity, and light.
- ESP32 evaluates sensor data and triggers irrigation if required.
- DF-Player Mini provides appropriate voice feedback.

3. *Software Implementation:*

- Development environment: Arduino IDE
- Programming language: C / C++
- Operating Systems: Windows, Linux, or macOS

4. *Arduino IDE*

- The Arduino Integrated Development Environment (IDE) is a cross-platform software application written in Java and based on C/C++. It is used for writing, compiling, and uploading programs to Arduino and compatible boards like ESP32. With the help of third-party board definitions (cores), it can support a wide variety of microcontrollers. It provides a simple interface for embedded development with a wide range of libraries for sensors, modules, and IoT tools.

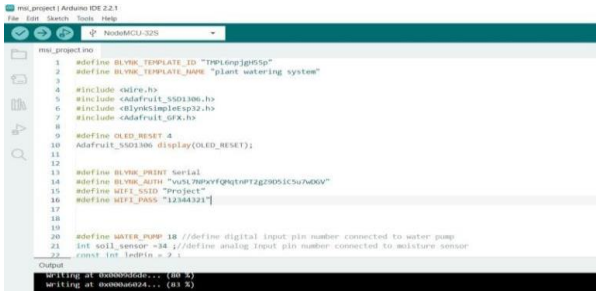

Figure 2 Arduino IDE

3.1. ***COMPONENTS DETAILS***

1. ESP32 Module

• A powerful dual-core microcontroller with built-in Wi-Fi and Bluetooth, ideal for IoT applications. It serves as the brain of the system, managing sensors and controlling other components

Figure 3 ESP32 Module [3]

2.DHT22Temperature&HumiditySensor

• Measures ambient temperature and humidity with high accuracy and sends the data to ESP32 for environment monitoring.

Figure 4 DHT22 Temperature & Humidity Sensor

3. Soil Moisture Sensor

• Measures the moisture content in soil. ESP32 uses this data to decide whether to activate the water pump.

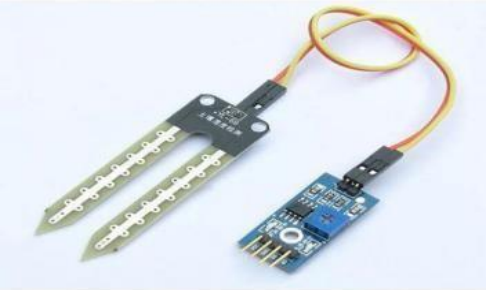

Figure 5 Soil Moisture Sensor [4]

4.DFPlayer Mini MP3 Module

• A tiny MP3 module used to play voice feedback or audio alerts using an SD card. Controlled by the ESP32 via serial communication.

Figure 6 DF-Player Mini MP3[5]

5.0.96"OLEDDisplay(I2C,NoTouch)

• A compact, energy-efficient screen that displays real-time data such as temperature, humidity, light intensity, and system status. Offers better contrast and readability than TFTs, especially in low-light environments. Connects via I2C for easier wiring with ESP32.

Figure 7 OLED[6]

3.2. *SCHEMATIC DIAGRAM*

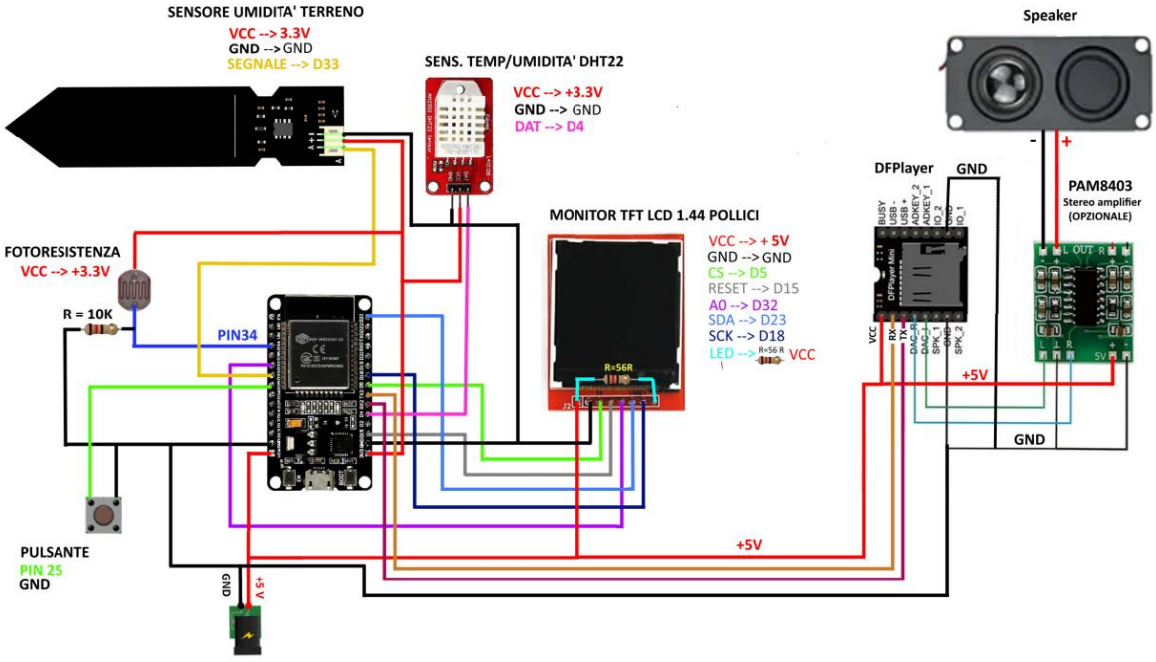

Figure 8 Schematic Diagram

3.3.***WORKING OF PROJECT***

1. **Data Collection**

- Sensors such as the **soil moisture sensor**, **DHT22 temperature/humidity sensor**, and **LDR** (light sensor) continuously collect environmental data from the plant's surroundings.
- The **ESP32** processes this data in real-time and stores important thresholds like low water levels, high temperature, or excessive dryness.

2. **Decision-Making**

- The ESP32 compares sensor values with predefined thresholds.
- If the soil is too dry, light is too low, or the plant is overheating, the ESP32 triggers an alert mechanism.
- Based on the type of condition, a pre-recorded **audio message** is selected.

3.**Communication (Talking Feature)**

- The **DF-Player Mini MP3 module** is controlled by the ESP32 to play voice messages stored on the **SD card** (e.g., “I need water!”, “It’s too hot!”).
- These audio outputs are amplified and played through a connected **speaker**.
- This gives the effect that the plant is “talking” about its condition.

4.**User Interaction**

- An **OLED display** shows live values such as temperature, humidity, and moisture levels in a readable format for visual feedback.

5.Power Management

- The system is powered by using a **DC adapter**, ensuring consistent supply.

3.4 Final Hardware Picture:

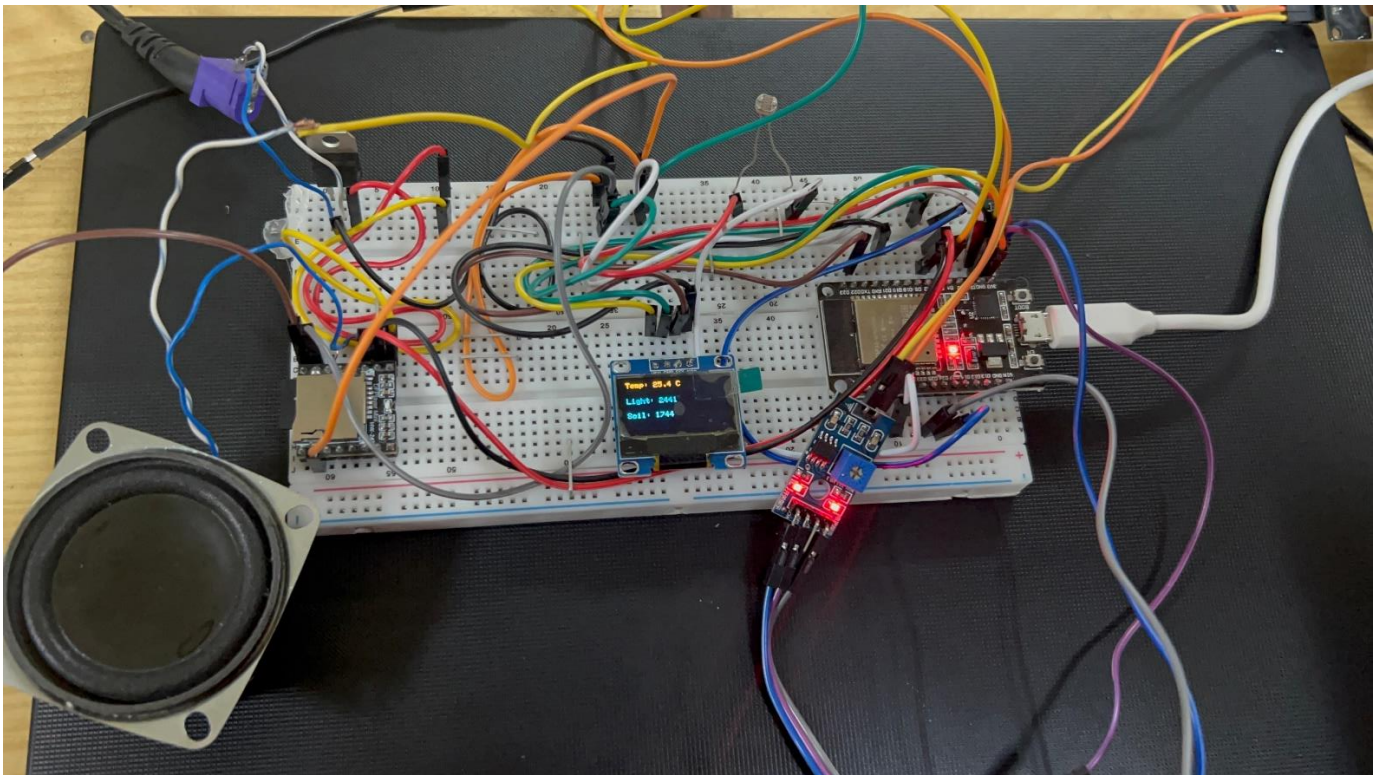

Figure 9 Picture of Hardware

4. PRACTICAL APPLICATION:

The automatic plant irrigation system you're developing has various practical applications that can benefit different sectors. Here are some practical applications:

1. Home Gardening:

For individuals growing plants indoors, in balconies, or backyards, this system ensures optimal watering based on real-time data. The voice interaction adds an emotional element, helping household members, especially children and elderly—engage with plants regularly. It simplifies plant care for busy individuals who may forget watering schedules.

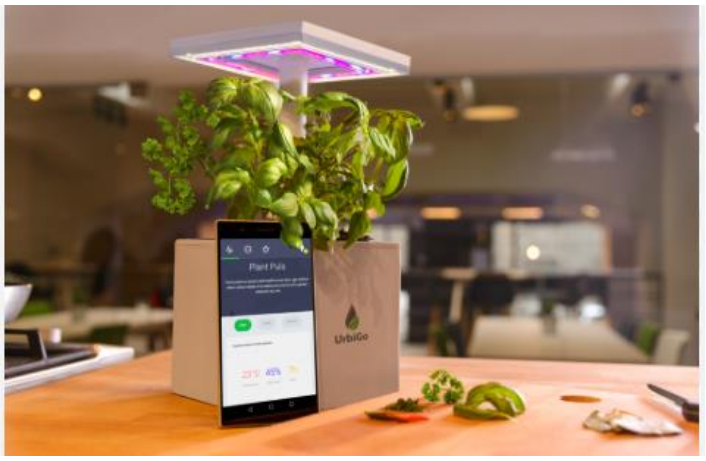

Figure 10 Smart Home Gardening [7]

2.UrbanFarming:

In city settings with limited space like rooftops or community gardens, managing multiple plants efficiently is a challenge. This smart system enables precise, automated irrigation and health updates, reducing manual labor. Urban farmers benefit from conserving water and improving yield by responding to actual plant conditions rather than using fixed schedules.

Figure 11 Urban Farming[8]

3.Smart City Agriculture

Integrated into smart city infrastructure, these systems can be deployed in public parks, green belts, or municipal gardens. They can communicate plant needs directly to city management teams via IoT networks, promoting better care of urban vegetation and contributing to greener cities.

Figure 12 Smart City Agriculture

4.RemoteAreaFarmingAutomation:

In rural or hard-to-access regions, farmers can face challenges with consistent

crop monitoring. This system allows for autonomous irrigation and verbal alerts, reducing the need for constant human presence. If paired with wireless or mobile network connectivity, it can relay data remotely, enabling farming automation in remote locations with minimal cost.

Figure 13 ***RemoteAreaFarmingAutomation***

5. Contribution of this project to the society (PLO 6):

This project contributes to society by:

- Promotes sustainable water usage
- Encourages smart agriculture
- Enhance Environmental Awareness
- Engages children and elders through voice-based plant interaction

6. PROJECT ANALYSIS WITH REFERENCE TO ENVIRONMENT AND SUSTAINABILITY (PLO 7)

Aligned with PLO 7, the system integrates sustainability by:

- Reduce water waste through smart irrigation
- Minimize energy consumption with efficient components
- Encourages planting and greenery in urban spaces
- Supports sustainable living practices

7. CONCLUSION:

This project demonstrates the effective fusion of IoT, sensors, and audio feedback to create a responsive, intelligent plant care system. It proves useful for all age groups and encourages sustainable practices in gardening and farming. With further enhancements like mobile integration, cloud storage, and AI-based decision-making, it can be expanded into a larger-scale agricultural solution.

The inclusion of real-time environmental monitoring, voice interactivity, and remote control bridges the communication gap between humans and plants. It not only improves plant care experience but also promotes awareness about plant well-being. Future developments may also include multilingual voice interaction, solar-powered modules for off-grid use, and integration with weather APIs for predictive irrigation strategies.

This scalable system has the potential to revolutionize both domestic gardening and commercial farming by transforming plants into active participants in their care and maintenance.

REFERENCES

[10] https://www.youtube.com/watch?v=ME9OcxJVRoY

[11] https://randomnerdtutorials.com/esp32-soil-moisture-sensor/

[12] https://lastminuteengineers.com/esp32-dht22-web-server-tutorial/

[13] https://www.instructables.com/Smart-Irrigation-System-Using-Blynk-and-ESP32/

[14] https://circuitdigest.com/microcontroller-projects/soil-moisture-based-automatic-irrigation-system-using-esp32

[15] https://www.hackster.io/news/smart-gardens-and-automated-irrigation-made-easy-with-esp32-and-blynk-dfefab

[16] https://maker.pro/esp32/projects/plant-monitoring-and-irrigation-system-using-esp32

[17] https://create.arduino.cc/projecthub/projects/tags/esp32-irrigation

APPENDIX

```
/*
  Calibration and Testing for:
  - Capacitive Soil Moisture Sensor V2
  - LDR (Photoresistor)
  - DHT22 Temperature and Humidity Sensor

  Displays data on Serial Monitor and 0.96" OLED Display (I2C)
  Uses ESP32 WROOM-32
*/
    #include <Wire.h>
    #include <Adafruit_GFX.h>
    #include <Adafruit_SSD1306.h>
    #include <DHT.h>
    #include <DFRobotDFPlayerMini.h>

    // OLED display setup
    #define SCREEN_WIDTH 128
    #define SCREEN_HEIGHT 64
    #define OLED_RESET -1
    Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT,
&Wire, OLED_RESET);

    // Sensor pins
    #define DHTPIN 4
    #define DHTTYPE DHT22
    #define SOIL_PIN 33
    #define PHOTO_PIN 34
    DHT dht(DHTPIN, DHTTYPE);

    // DFPlayer setup
    HardwareSerial dfSerial(1);  // Use UART1
    DFRobotDFPlayerMini myDFPlayer;

    // Sensor values
    float temperatura;
    int umidterreno;
    int luminosit;

    void setup() {
    Serial.begin(115200);
    dfSerial.begin(9600, SERIAL_8N1, 16, 17);  // RX=16, TX=17
(ESP32 UART1)

    // OLED init
    if (!display.begin(SSD1306_SWITCHCAPVCC, 0x3C)) {
    Serial.println(F("❌ SSD1306 init failed"));
    while (1);
    }
    display.clearDisplay();
    display.display();

    // DFPlayer init
    Serial.println("🔌 Connecting to DFPlayer...");
    if (!myDFPlayer.begin(dfSerial)) {
```

```
    Serial.println("❌ DFPlayer Mini not found!");
    while (1);
    }
    Serial.println("☑ DFPlayer connected.");
    myDFPlayer.volume(30);
    Serial.println("🔊 Volume set to 30 (max)");

    // Sensor setup
    dht.begin();
    pinMode(SOIL_PIN, INPUT);
    pinMode(PHOTO_PIN, INPUT);

    delay(1000);
    }

    void loop() {
    // Read sensors
    temperatura = dht.readTemperature();
    umidterreno = analogRead(SOIL_PIN);
    luminosit = analogRead(PHOTO_PIN);

    // Serial debug
    Serial.println("🌡  Temp: " + String(temperatura) + "°C");
    Serial.println("💡 Light: " + String(luminosit));
    Serial.println("🌱 Soil: " + String(umidterreno));
    Serial.println("------------------------");

    // Display on OLED (clean block layout)
    display.clearDisplay();
    display.setTextColor(SSD1306_WHITE);

    // Temp block
    display.setCursor(0, 0);
    display.setTextSize(1);
    display.print("Temp: ");
    display.setTextSize(2);
    display.print(temperatura, 1);
    display.println("C");

    // Light block
    display.setCursor(0, 24);
    display.setTextSize(1);
    display.print("Light: ");
    display.setTextSize(2);
    display.println(luminosit);

    // Soil block
    display.setCursor(0, 48);
    display.setTextSize(1);
    display.print("Soil: ");
    display.setTextSize(2);
    display.println(umidterreno);

    display.display();

    // --- Audio messages ---
```

```
    if (umidterreno > 3300) {
    Serial.println("🔊 Playing: I need water");
    myDFPlayer.play(1);  // 0001.mp3
    delay(5000);
    } else if (temperatura > 35.0) {
    Serial.println("🔊 Playing: It's getting hot");
    myDFPlayer.play(2);  // 0002.mp3
    delay(5000);
    } else if (luminosit < 1000) {
    Serial.println("🔊 Playing: Need sunlight");
    myDFPlayer.play(3);  // 0003.mp3
    delay(5000);
    } else {
    Serial.println("🔊 Playing: Thank you");
    myDFPlayer.play(4);  // 0004.mp3
    delay(5000);
    }

    delay(3000);
    }
```