جامعة ابن طفيل
Ibn Tofaïl University
Faculté des Sciences

IoT Project

ESP32-Based Irrigation System

Réalisé par:

EL Mahdi ATMANI

Instructeur: Pr. Abdessalam AIT MADI

Université Ibn Tofail
Informatique et Intelligence Artificielle

Oct 2024

Contents

1 Introduction **4**
- 1.1 Motivation . . . 5
  - 1.1.1 Problem Statement . . . 5
  - 1.1.2 Key Motivation for the Project . . . 5
  - 1.1.3 Technological Motivation . . . 5
  - 1.1.4 Contrast with Traditional Systems . . . 6

2 Literature Review **7**
- 2.1 Introduction to Smart Irrigation Systems . . . 7
- 2.2 Role of IoT in Agriculture . . . 7
- 2.3 Architecture of IoT-based Irrigation Systems . . . 7
- 2.4 Advances in Sensor Technologies . . . 8
- 2.5 Cloud-based IoT Platforms for Irrigation Management . . . 8
- 2.6 Benefits of IoT-based Smart Irrigation Systems . . . 8
- 2.7 Challenges in Implementing IoT-based Irrigation Systems . . . 8
- 2.8 Conclusion . . . 8

3 System design and components **10**
- 3.1 Wiring Diagram . . . 10
- 3.2 ESP32 Microcontroller . . . 10
  - 3.2.1 GPIO Control . . . 11
- 3.3 Moisture Sensor (MH-Sensor-Series) . . . 11
  - 3.3.1 Moisture Sensor Pin Configuration . . . 11
- 3.4 GPIO Reader . . . 12
- 3.5 Relay Module (JQC3F-05VDC-C) . . . 12
  - 3.5.1 Relay Pin Configuration . . . 12
  - 3.5.2 Breadboard Usage in the Project . . . 12
- 3.6 Water Pump Integration in the Project . . . 13
- 3.7 System Overview . . . 14
- 3.8 Component Selection . . . 14
  - 3.8.1 ESP32 Microcontroller . . . 14
  - 3.8.2 Soil Moisture Sensor . . . 15
  - 3.8.3 Relay Module . . . 15
  - 3.8.4 Water Pump . . . 15
- 3.9 Wiring Setup . . . 15
  - 3.9.1 ESP32 to Relay Wiring . . . 15
  - 3.9.2 Soil Moisture Sensor to ESP32 Wiring . . . 16
- 3.10 Control Logic . . . 16

4 Software Implementation **17**
4.1 Code Overview . . . 17
4.1.1 Moisture Sensor and Water Pump Control Code Explanation . . . 18

5 Testing and Results **23**
5.1 Testing Methodology . . . 23
5.1.1 Initial Testing Phase . . . 23
5.1.2 Test Environment Setup . . . 23
5.2 Test Results and Challenges . . . 24
5.2.1 Communication Testing . . . 24
5.2.2 Voltage Compatibility Testing . . . 24
5.3 Analysis of Results . . . 24
5.3.1 Successful Implementations . . . 24
5.3.2 Identified Limitations . . . 25
5.4 Testing Conclusions . . . 25

6 Future Improvements **26**
6.1 Sensor Enhancement . . . 26
6.1.1 Robust Moisture Sensor Algorithm . . . 26
6.2 Power System Upgrades . . . 26
6.2.1 High-Voltage Device Support . . . 26
6.2.2 Enhanced Power Management . . . 27
6.3 Connectivity Enhancements . . . 27
6.3.1 Advanced Wi-Fi Implementation . . . 27
6.4 Additional Feature Proposals . . . 28
6.4.1 Smart System Integration . . . 28
6.4.2 User Experience Improvements . . . 28

7 Conclusion **29**

Abstract

Efficient water management is essential for agriculture and horticulture, where overwatering or underwatering can lead to significant resource wastage and reduced crop yields. Traditional irrigation systems, which often rely on fixed schedules, lack adaptability to real-time environmental conditions, resulting in inefficient water usage. This project presents the development of a smart irrigation system using an ESP32 microcontroller, integrated with a soil moisture sensor, a relay-controlled water pump, and an IoT platform. The system automates irrigation by analyzing real-time soil moisture data and allows for remote control via cloud services, enhancing both water efficiency and user convenience.

The proposed solution offers a data-driven approach to irrigation, activating the water pump only when soil moisture falls below a predefined threshold or when manually triggered through an IoT-based interface. The use of the ESP32 platform ensures low-power consumption, built-in Wi-Fi for seamless cloud integration, and scalability for various agricultural settings. Initial testing shows the system performs effectively for low-power devices, such as a 3.3V light bulb, with challenges encountered for higher voltage devices due to relay limitations. Future iterations will address these challenges by improving power management and extending the system's capability for larger agricultural applications.

Overall, this project demonstrates a practical application of IoT in smart agriculture, with potential long-term benefits in resource conservation, sustainability, and scalability for larger farms or commercial use. The system provides a foundation for future enhancements, such as weather API integration, higher voltage device support, and machine learning-based irrigation optimization.

Chapter 1

Introduction

Water is one of the most vital resources in agriculture, serving as the foundation for crop growth, soil fertility, and overall farm productivity. However, increasing global population, climate change, and unpredictable weather patterns are putting significant pressure on water resources. In many regions, including Morocco, agricultural water usage is inefficient, leading to excessive waste, degradation of natural ecosystems, and reduced availability of water for future use. Sustainable water management is therefore critical for ensuring long-term food security, environmental health, and economic viability, particularly in water-scarce countries like Morocco.

Morocco has been facing a severe water crisis in recent years, driven by prolonged droughts, over-extraction of groundwater, and the impacts of climate change. Agriculture, which accounts for nearly 80% of the country's water consumption, is one of the sectors most vulnerable to these changes. Water shortages have caused reductions in crop yields, increased competition for limited water resources, and heightened food insecurity in some regions. The Moroccan government has recognized the importance of water management by implementing national strategies like the National Water Plan and encouraging the use of more efficient irrigation systems. However, the challenge remains substantial, as traditional irrigation methods—such as flood and surface irrigation—are still prevalent and inefficient.

In this context, advanced technologies like the Internet of Things (IoT) can play a transformative role in addressing Morocco's water crisis. IoT solutions enable real-time monitoring and control of water resources, allowing farmers to optimize water usage for irrigation and minimize waste. Smart irrigation systems, equipped with IoT sensors, can monitor critical environmental variables such as soil moisture, temperature, and weather conditions, and automatically adjust water delivery based on crop needs. This technology helps farmers save water while maintaining or even improving crop yields, making it a promising tool in the fight against Morocco's water challenges.

The objective of this report is to explore how IoT can be leveraged to optimize irrigation practices, improve agricultural productivity, and conserve water, specifically in the context of water-scarce regions like Morocco. By analyzing the potential of IoT in irrigation management, this report aims to highlight how smart technologies can contribute to a more sustainable and resilient agricultural sector in the face of ongoing water shortages.

This project focuses on automating irrigation systems using microcontrollers such as the ESP32 to optimize water usage and improve plant health, whether for agricultural crops or small-scale setups like balcony gardens. The system leverages an ESP32 to monitor soil moisture levels through a sensor and control a relay that activates a water pump accordingly. This process is managed via the cloud using a mobile application or with Google Home for smart home automation, providing a seamless, user-friendly approach to efficient irrigation.

1.1 Motivation

1.1.1 Problem Statement

Efficient water management is a critical challenge in agriculture and gardening today. Overwatering or underwatering can harm plant health and waste valuable resources. Traditional irrigation systems, which typically operate on fixed schedules, do not account for real-time soil moisture levels or changing weather conditions. This leads to either water wastage or insufficient irrigation, both of which have negative economic and environmental consequences.

1.1.2 Key Motivation for the Project

The motivation behind this project is to develop a **smart irrigation system** that addresses these inefficiencies by:

- **Improving water usage efficiency**: Water is only used when necessary, based on real-time soil moisture data, preventing wastage.
- **Reducing manual intervention**: The system automates the irrigation process, significantly reducing the need for human oversight.
- **Enabling remote monitoring and control**: Through IoT integration, users can monitor soil moisture levels and control the water pump remotely, making it ideal for small farmers, gardeners, and hobbyists.
- **Sustainability**: Preventing over-irrigation contributes to sustainable water management, particularly in areas facing water scarcity.

1.1.3 Technological Motivation

The ESP32 platform is central to the system's design, offering:

- Low power consumption and built-in Wi-Fi connectivity.
- Compatibility with a range of sensors, actuators, and cloud services.
- An affordable, versatile solution for developing a scalable smart irrigation system, accessible to small-scale farmers and hobbyists.

1.1.4 Contrast with Traditional Systems

Traditional irrigation systems rely on fixed schedules to water plants. These systems often result in inefficient water use due to their inability to adapt to real-time environmental conditions. By contrast, the proposed system is data-driven, responsive, and integrated with IoT technology. Table 1.1 highlights the key differences:

Feature	**Traditional Systems**	**Smart Irrigation System (Proposed)**
Watering Method	Based on fixed schedules	Based on real-time soil moisture data
Resource Efficiency	Often inefficient, can lead to water wastage	Water used only when necessary, reducing waste
Remote Control	Manual, no remote control	IoT-enabled, accessible remotely
Environmental Awareness	No adaptation to weather changes	Can integrate weather APIs and sensors
Manual Intervention	High, requires frequent adjustments	Low, fully automated operation
Cost of Water Usage	High due to inefficiency	Reduced due to data-driven irrigation

Table 1.1: Comparison of Traditional and Smart Irrigation Systems

Chapter 2

Literature Review

2.1 Introduction to Smart Irrigation Systems

In recent years, the integration of Internet of Things (IoT) technology into agriculture has gained widespread attention, offering a pathway to more efficient and sustainable farming practices. Traditional irrigation methods, often labor-intensive and reliant on manual control, can lead to significant water wastage and inefficiencies. IoT-based smart irrigation systems aim to address these issues by utilizing real-time data from sensors to automate and optimize water usage, enhancing both agricultural productivity and water conservation.

2.2 Role of IoT in Agriculture

IoT technology is increasingly being applied in precision agriculture, particularly in irrigation management, where real-time data collection and remote monitoring are essential. Studies have shown that IoT-based systems allow for the collection of environmental data such as soil moisture, humidity, temperature, and water levels, enabling farmers to make informed decisions on irrigation scheduling. Sensors, such as soil moisture sensors, temperature and humidity sensors, and water level sensors, are pivotal in this regard. These devices provide data that is processed and transmitted to cloud-based platforms, such as ThingsBoard or AWS IoT, where analytics and decision-making algorithms can optimize irrigation strategies.

2.3 Architecture of IoT-based Irrigation Systems

Most IoT-based smart irrigation systems are structured into three key layers: the sensor layer, the communication layer, and the application layer. Sensors placed in the field collect data on environmental conditions. The communication layer, typically using protocols such as HTTP or MQTT, facilitates secure data transmission to the cloud platform. Finally, the application layer provides dashboards and user interfaces for real-time data visualization and system control. Studies such as [1] have demonstrated the efficiency of this layered architecture in ensuring seamless data flow, improving water use efficiency, and providing actionable insights into farm management.

2.4 Advances in Sensor Technologies

The advancement in sensor technologies has significantly contributed to the development of smart irrigation systems. Various sensors like the DHT22 (for humidity and temperature) and soil moisture sensors play a crucial role in monitoring field conditions. These sensors can provide real-time, precise data, which can be integrated with algorithms to make data-driven decisions on when and how much to irrigate. Several studies [1] highlight the importance of sensor calibration to ensure accurate data, which is critical for the effective functioning of smart irrigation systems.

2.5 Cloud-based IoT Platforms for Irrigation Management

The use of cloud computing in IoT-based irrigation systems has emerged as a key enabler of smart farming. Cloud platforms such as ThingsBoard, AWS IoT, and Microsoft Azure IoT offer robust solutions for data storage, real-time analytics, and remote monitoring. These platforms allow farmers to visualize environmental data in real-time, set irrigation schedules, and receive notifications or alerts for critical conditions, such as low soil moisture or high water levels. Cloud-based irrigation systems have been shown to reduce water consumption by optimizing irrigation schedules based on predictive analytics and real-time data analysis.

2.6 Benefits of IoT-based Smart Irrigation Systems

The implementation of IoT-based irrigation systems has several key benefits. First, it improves water use efficiency by automating the irrigation process and ensuring that crops receive the optimal amount of water. This can lead to water savings of up to 50%, as demonstrated in various case studies [1]. Second, these systems reduce the labor required for irrigation management, as remote monitoring and control can be performed via mobile devices or computers. Finally, IoT systems contribute to sustainable farming practices by preventing over-irrigation and minimizing the impact on natural water resources.

2.7 Challenges in Implementing IoT-based Irrigation Systems

While IoT-based irrigation systems offer numerous benefits, several challenges remain. These include the high initial cost of system installation, particularly for small-scale farmers, and the need for reliable internet connectivity in rural areas. Additionally, the complexity of managing large datasets and ensuring data security are ongoing concerns. Future research should focus on developing more cost-effective solutions and addressing data privacy issues in IoT-enabled farming systems.

2.8 Conclusion

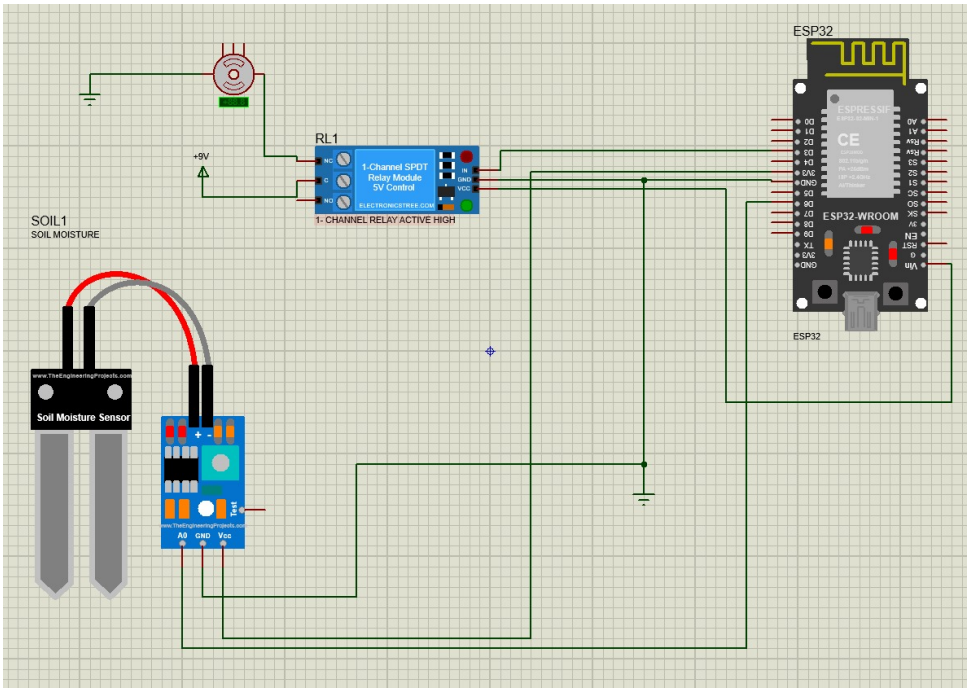
IoT-based smart irrigation systems represent a significant advancement in agricultural technology, offering the potential to enhance water use efficiency, reduce labor, and

improve crop yields. However, ongoing challenges related to cost, connectivity, and data management must be addressed to ensure widespread adoption. Future innovations in sensor technologies, cloud computing, and machine learning algorithms are likely to further optimize irrigation management, contributing to more sustainable and efficient farming practices.

Chapter 3

System design and components

3.1 Wiring Diagram



3.2 ESP32 Microcontroller

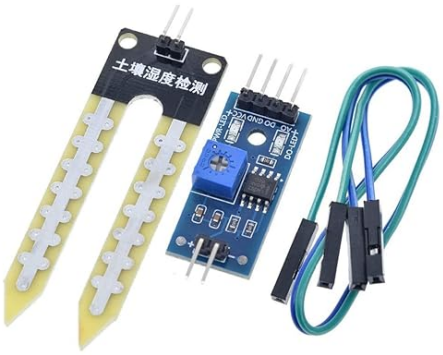
The ESP32 is a powerful and widely used microcontroller with built-in Wi-Fi and Bluetooth capabilities, making it an ideal choice for IoT applications. The ESP32 can interface with sensors, relays, and other peripherals through its GPIO pins.

3.2.1 GPIO Control

We used GPIO 26 (G26) on the ESP32 to control the relay responsible for switching the water pump. The ESP32's GPIO pins output 3.3V, which can drive the control signal for the relay.

3.3 Moisture Sensor (MH-Sensor-Series)

The MH-Sensor-Series moisture sensor measures the volumetric water content in soil. The sensor outputs an analog signal that the ESP32 reads via one of its analog GPIO pins, allowing us to trigger actions based on moisture levels.



3.3.1 Moisture Sensor Pin Configuration

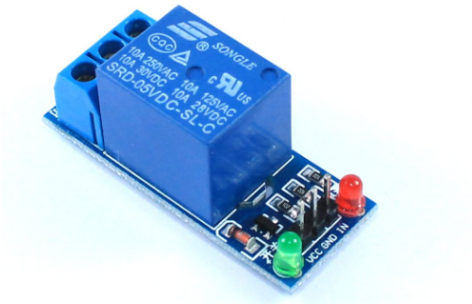
- `VCC` ——> 3.3V from ESP32
- `GND` ——> Ground
- `A0 (Analog Output)` ——> Connected to GPIO 34 (G34)

3.4 GPIO Reader

GPIO Pin 32 (G32) of the ESP32 microcontroller was utilized to read data from the connected moisture sensor. The sensor's analog signal is captured by the ESP32's GPIO pin, which converts the raw input into a corresponding numerical value. This processed value represents the moisture level detected by the sensor, enabling the system to effectively monitor soil conditions in real-time. The ESP32, with its built-in ADC (Analog-to-Digital Converter), accurately interprets the sensor signal and provides data for further analysis or display.

3.5 Relay Module (JQC3F-05VDC-C)

A relay acts as an electrically operated switch, allowing low-voltage signals to control higher voltage or current devices. The relay we used is powered by 5V and triggered by the ESP32 to turn the pump or light bulb on and off.



3.5.1 Relay Pin Configuration

- `VCC` ——> 5V from ESP32
- `GND` ——> Ground
- `IN` ——> Connected to GPIO 26 (G26)
- `COM (Common)` ——> Connected to the 3.3V power line
- `NO (Normally Open)` ——> Connected to the light bulb or pump

3.5.2 Breadboard Usage in the Project

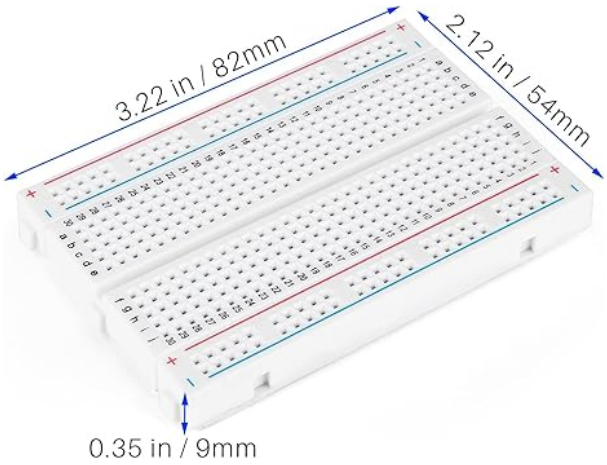
In this project, the breadboard is a critical tool for assembling and testing the circuit without permanent soldering. It provides a flexible way to connect components such as the ESP32, relay module, and moisture sensor, enabling easy modification, replacement, and debugging of the circuit.

Key Features:

- **Power Rails:** The breadboard has two vertical power rails, typically marked as + (positive) and – (negative). These are used to distribute power (e.g., 3.3V or 5V) and ground (GND) across the board for all components.

- **Terminal Strips:** The horizontal rows in the center of the breadboard are used for connecting components. Each row consists of five holes electrically connected, making it simple to link different components.

The breadboard allows quick connections of the ESP32's GPIO pins to the relay and moisture sensor using jumper wires. This setup is invaluable for verifying that sensor readings are accurate and the relay is functioning correctly. Once testing and adjustments are complete, the circuit can be transitioned to a permanent solution, such as a soldered PCB.



3.6 Water Pump Integration in the Project

In this project, the water pump is an essential part of the automated irrigation system, responsible for delivering water to the plants. The pump is controlled through the relay module, which is connected to the ESP32. The water pump operates as follows:

Functionality:

- The water pump is powered by a separate power source and connected to the `NO (Normally Open)` terminal of the relay.
- The ESP32 controls the relay via GPIO 26. When the soil moisture sensor detects that the soil is dry (below a set threshold), the ESP32 sends a signal to the relay, closing the circuit and activating the pump.
- Once the soil reaches the desired moisture level, the ESP32 signals the relay to open the circuit, turning off the pump and stopping the water flow.

This automated system ensures efficient water usage by turning the water pump on and off based on real-time sensor data. The breadboard simplifies the connection between the ESP32, relay, and pump, allowing for easy testing and adjustments during the development phase.

This chapter presents the design and architecture of the automated soil moisture monitoring and control system. The system uses an ESP32 microcontroller to gather data from a soil moisture sensor, process the information, and control a water pump via a relay module. The section details the wiring, component selection, and the logic behind the code implementation.

3.7 System Overview

The system is designed to automate the irrigation process in a garden or farm by monitoring soil moisture levels. The core components include a soil moisture sensor, a relay module, an ESP32 microcontroller, and a water pump. The moisture level is continuously monitored, and when the soil is dry, the system automatically activates the water pump, providing irrigation as needed.

The high-level overview of the system functionality is as follows:

- The soil moisture sensor measures the resistance between two probes to assess the water content in the soil.
- The ESP32 reads the sensor data and compares it with predefined threshold values.
- If the soil moisture level falls below the threshold, the ESP32 sends a signal to the relay to turn on the water pump.
- Once the moisture level is restored to the desired value, the pump is turned off.

3.8 Component Selection

The components selected for the system were chosen based on their compatibility with one another, ease of use, and cost-effectiveness. Below is a summary of the key components and their roles:

3.8.1 ESP32 Microcontroller

The ESP32 is a powerful, low-cost microcontroller with built-in Wi-Fi and Bluetooth, making it suitable for IoT applications. Its role in this system is to:

- Read analog data from the soil moisture sensor.

- Process the data to determine whether the soil is dry or wet.
- Control the relay module to switch the water pump on or off.

In this system, the ESP32 operates at 3.3V logic, but it can communicate with the 5V relay module without the need for logic level shifting.

3.8.2 Soil Moisture Sensor

The soil moisture sensor measures the soil's water content by determining the resistance between two conductive probes. This sensor has three pins: VCC, GND, and an analog output pin (A0). The analog output provides a continuous signal proportional to the moisture level, which the ESP32 reads and interprets.

- When the soil is dry, the resistance increases, causing the sensor output to drop.
- When the soil is wet, the resistance decreases, raising the output voltage.

3.8.3 Relay Module

The relay module used in this system is a 1-channel SPDT (Single Pole Double Throw) relay with 5V control. It acts as an electronically controlled switch, allowing the ESP32 to control a higher voltage device like a 9V water pump or other actuators.

- The relay operates in an active-high configuration, meaning it is triggered when the control signal from the ESP32 goes high.
- The relay's Normally Open (NO) pin is connected to the water pump, while the Common (COM) pin is connected to the 9V power supply.

3.8.4 Water Pump

A 9V DC water pump is connected to the system via the relay. When the relay is activated by the ESP32, the pump is powered on to provide irrigation to the soil. This configuration ensures that the pump is isolated from the low-power components like the ESP32, protecting the microcontroller from high-current loads.

3.9 Wiring Setup

3.9.1 ESP32 to Relay Wiring

The ESP32's G26 pin is connected to the relay's input (IN) pin, which allows the microcontroller to control the relay state. The relay's common (COM) pin is wired to the external 9V power supply, while the normally open (NO) pin is linked to the positive terminal of the water pump. The ground of the power supply and the ESP32 share a common ground.

The wiring setup works as follows:

- When the ESP32 sends a HIGH signal (3.3V) to the relay, the relay switches on, completing the circuit and turning on the water pump.

- The pump remains active as long as the signal remains HIGH.
- When the signal goes LOW, the relay opens the circuit, turning the pump off.

This design ensures that the ESP32 can control the water pump without directly handling higher voltage or current, thus safeguarding the microcontroller.

3.9.2 Soil Moisture Sensor to ESP32 Wiring

The soil moisture sensor is connected as follows:

- The VCC pin of the sensor is connected to the 3.3V output of the ESP32.
- The GND pin is connected to the ground (GND) of the ESP32.
- The analog output pin (A0) of the sensor is wired to an analog input pin (A0) on the ESP32.

The analog signal from the sensor represents the soil moisture level. The ESP32 reads this signal and converts it into a digital value ranging from 0 to 1023. Based on predefined thresholds for dry and wet soil conditions, the ESP32 determines whether the soil requires irrigation.

3.10 Control Logic

The system's control logic is implemented through a simple decision-making process:

1. The ESP32 continuously reads the soil moisture level from the sensor.
2. If the moisture level falls below a defined threshold (indicating dry soil), the ESP32 sends a signal to the relay to activate the water pump.
3. The pump remains active until the moisture level rises above another threshold (indicating sufficiently moist soil), at which point the ESP32 turns the pump off.
4. The system repeats this cycle, ensuring optimal soil moisture levels.

The thresholds can be fine-tuned in the code based on the specific soil type and environmental conditions.

Chapter 4

Software Implementation

4.1 Code Overview

The following code is responsible for switching the relay on and off in intervals to simulate an irrigation cycle. The ESP32 reads moisture sensor data (future implementation), and based on predefined conditions, it triggers the relay to activate the pump.

```
// Include necessary libraries
#include "thingProperties.h"

const int moistureSensorPin = 32;  // Moisture sensor connected to GPIO pin 32
const int relayPin = 26;           // GPIO pin for controlling the relay (G26 on the ESP32)
int moistureLevel = 0;             // Variable to store the sensor value

void setup() {
  // Initialize serial communication
  Serial.begin(115200);
  pinMode(relayPin, OUTPUT);       // Set the relay pin as output
  digitalWrite(relayPin, LOW);     // Ensure the relay is off initially (pump off)

  // Initialize IoT Cloud properties
  initProperties();

  // Connect to the cloud
  ArduinoCloud.begin(ArduinoIoTPreferredConnection);

  // Wait for connection to the cloud
  setDebugMessageLevel(2);
  ArduinoCloud.printDebugInfo();
}

void loop() {
  // Update IoT data
  ArduinoCloud.update();

  // Read moisture level
  moistureLevel = analogRead(moistureSensorPin);

  // Convert sensor value to percentage (0 to 100)
  int moisturePercent = map(moistureLevel, 4095, 910, 0, 100);
```

```
  MOISTURE = moisturePercent;
  Serial.print("Moisture level: ");
  Serial.print(moistureLevel);
  Serial.print(" | Percentage: ");
  Serial.println(moisturePercent);

  // Check the switch state and moisture level
  if (SWITCH || moisturePercent < 50) {  // Using the _switch_ variable
    Serial.println("The switch is ON");
    digitalWrite(relayPin, HIGH);  // Turn the relay on (pump active)
    Serial.println("Pump activated");
  } else {
    digitalWrite(relayPin, LOW);   // Turn the relay off (pump inactive)
    Serial.println("Pump deactivated");
  }

  delay(2000);  // 2-second pause before the next check
}

/*
  Since MOISTURE is a READ_WRITE variable, onMOISTUREChange() is
  executed every time a new value is received from IoT Cloud.
*/
void onMOISTUREChange()  {
  // Add your code here to act upon MOISTURE change
}

/*
  Since SWITCH is a READ_WRITE variable, onSWITCHChange() is
  executed every time a new value is received from IoT Cloud.
*/
void onSWITCHChange()  {
  // Add your code here to act upon SWITCH change
}
```

Listing 4.1: Arduino Code

4.1.1 Moisture Sensor and Water Pump Control Code Explanation

This project uses an ESP32 microcontroller to control a water pump based on the readings from a soil moisture sensor. The system is integrated with the IoT Cloud, allowing for remote monitoring and control of the water pump.

Pin Configuration:

- `moistureSensorPin`: The moisture sensor is connected to GPIO 32 of the ESP32. This pin reads the analog signal from the sensor, representing the soil moisture level.
- `relayPin`: The relay, which controls the water pump, is connected to GPIO 26. This pin is configured as an output to switch the relay on and off.

Setup: The `setup()` function initializes the system:

- Serial communication is started using `Serial.begin(115200)` to allow debugging and monitoring of sensor values.
- The relay pin is set as an output with `pinMode(relayPin, OUTPUT)`, and the relay is turned off initially by setting `digitalWrite(relayPin, LOW)`, ensuring that the water pump is off when the system starts.
- IoT Cloud properties are initialized using `initProperties()`, and a connection is established with the cloud using `ArduinoCloud.begin()`.
- Debugging information is enabled to monitor the cloud connection using `setDebugMessageLeve`

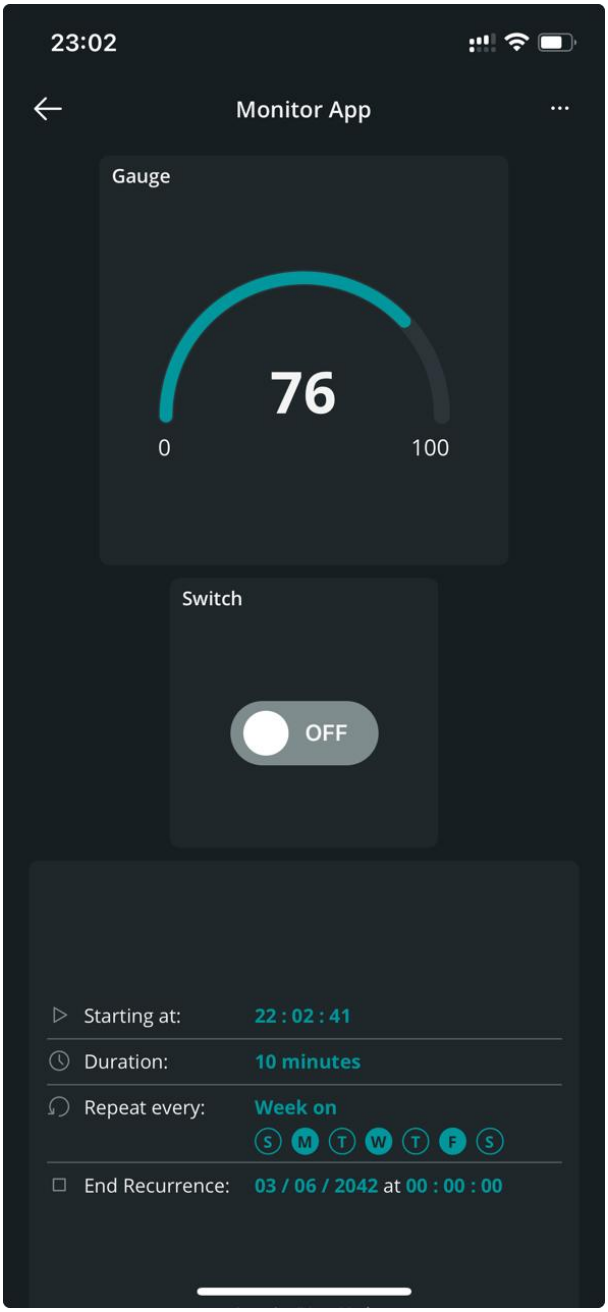
Main Loop: In the `loop()` function:

- The function `ArduinoCloud.update()` keeps the system synchronized with the IoT Cloud, updating any variables or properties linked to the cloud.
- The soil moisture is read using `analogRead(moistureSensorPin)`, which gives a raw value from the sensor. This value is then mapped to a percentage (0-100) using the `map()` function, where 4095 represents dry soil and 910 represents wet soil.
- The calculated moisture percentage is printed to the serial monitor for debugging purposes.
- If the moisture level is below 50% or if the user manually turns on the pump using the cloud-based switch, the relay is activated (`digitalWrite(relayPin, HIGH)`), turning the pump on. If the moisture level is sufficient or the switch is turned off, the pump is turned off (`digitalWrite(relayPin, LOW)`).
- A delay of 2 seconds (`delay(2000)`) ensures the system waits before checking the moisture level again.

IoT Cloud Integration: Two key variables, `MOISTURE` and `SWITCH`, are synchronized with the IoT Cloud. The function `onMOISTUREChange()` is triggered whenever the moisture level is updated from the cloud, while `onSWITCHChange()` reacts to changes in the state of the manual switch. These functions enable remote control and monitoring of the system through the cloud.

This system allows real-time monitoring of soil moisture levels and control of the water pump, ensuring efficient irrigation based on the current environmental conditions. Remote control via the IoT Cloud adds convenience, allowing the user to activate or deactivate the pump from anywhere.

Mobile App Cloud Integration:



Google Voice Assistant Integration:

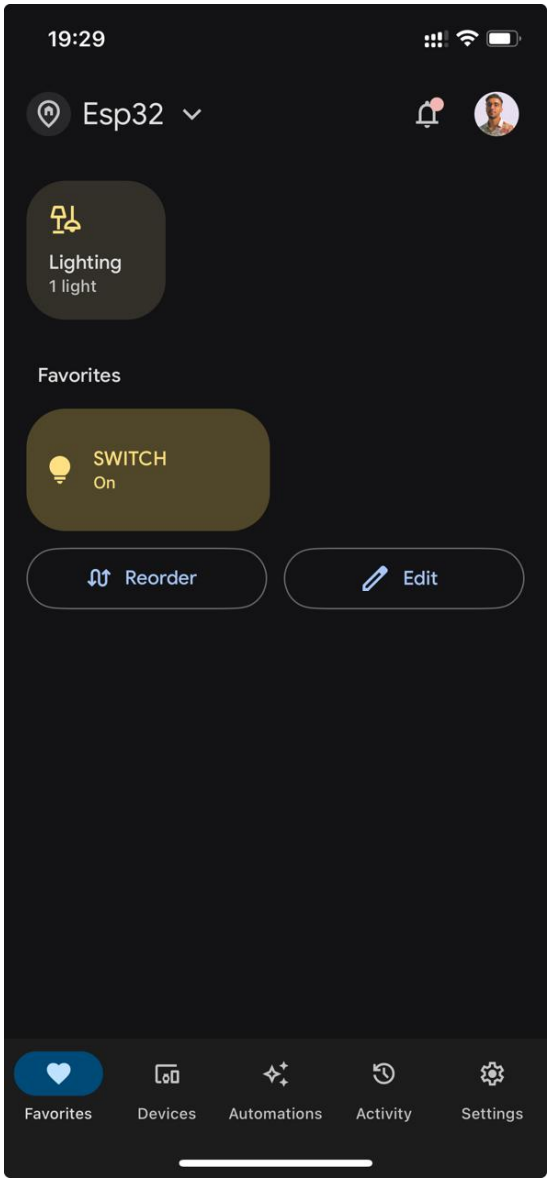


figure Google Home Image 1

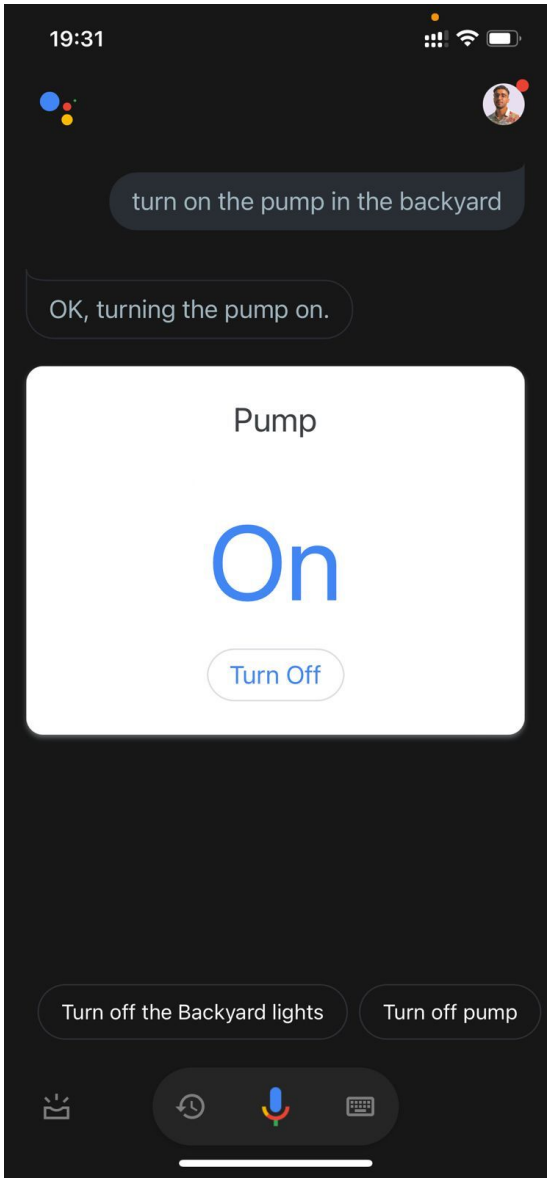


figure Google Home Image 2

The project incorporates Google Voice Assistant capabilities to create an intelligent irrigation system with advanced control features. The implementation merges hardware automation with user-friendly voice and mobile interfaces, enhancing the overall functionality and accessibility of the system.

Interface Implementation

The mobile application interface serves as the central control hub, featuring a sophisticated moisture monitoring dashboard. This dashboard includes a prominent circular gauge displaying soil moisture levels on a scale of 0-100%, with real-time readings clearly visible against a dark-themed interface. A manual toggle switch provides direct control over the system's operation.

Voice command functionality is implemented through Google Assistant integration, allowing users to control the system through natural language commands. Users can issue simple instructions such as "turn on the switch" or "turn off the backyard pump," with the system providing immediate feedback on device status and command execution.

Technical Implementation

The technical implementation ensures seamless integration between the ESP32 microcontroller and Google Home infrastructure, utilizing cloud-based command processing and local network communication protocols. This architecture enables robust synchronization between the mobile application and hardware components, creating a responsive and reliable automation system.

Users benefit from both automated control through moisture sensing and manual override capabilities, accessible through voice commands and the mobile interface. This dual-control approach ensures maximum flexibility and user convenience while maintaining the system's core automation features.

Integration Summary

The implementation demonstrates the successful merger of IoT hardware with cloud-based voice assistant technology, creating a sophisticated yet user-friendly irrigation control system. This integration represents a significant advancement over traditional irrigation systems, offering enhanced control options and improved user interaction capabilities.

Chapter 5

Testing and Results

This chapter presents the comprehensive testing procedures undertaken and the results obtained during system implementation. The testing phase revealed both successes and challenges, providing valuable insights for system optimization.

5.1 Testing Methodology

5.1.1 Initial Testing Phase

The testing process was conducted in multiple stages to evaluate different aspects of the system:

- Communication protocol verification
- Component functionality testing
- Voltage compatibility assessment
- System integration testing

5.1.2 Test Environment Setup

Testing was conducted using:

- ESP32 development board
- Various test loads:
  - 3.3V light bulb
  - 9V water pump
- Relay module
- Serial monitoring tools

5.2 Test Results and Challenges

5.2.1 Communication Testing

Initial communication testing revealed several challenges:

- Serial Monitor displayed corrupted data due to incorrect baud rate configuration
- Resolution achieved by:
  - Adjusting baud rate to 115200
- Final implementation demonstrated stable communication

5.2.2 Voltage Compatibility Testing

Low Voltage Device Testing

Testing with 3.3V devices showed positive results:

- Successful control of 3.3V light bulb
- Reliable relay switching
- Consistent performance over extended operation
- No signal degradation observed

Higher Voltage Device Testing

Tests with 9V components revealed limitations:

- Unreliable relay switching for 9V pump
- Issues identified:
  - Insufficient current from ESP32's 3.3V signal
  - Relay switching threshold limitations
  - Potential EMF interference

5.3 Analysis of Results

5.3.1 Successful Implementations

- Stable communication protocol established
- Effective control of low-voltage devices
- Reliable serial monitoring after configuration
- Successful integration of basic control functions

5.3.2 Identified Limitations

- Voltage mismatch between control signal and high-voltage devices
- Relay switching limitations for higher power applications
- Initial communication configuration challenges

5.4 Testing Conclusions

The testing phase revealed both the system's capabilities and limitations:

- Successful implementation for low-voltage applications
- Identified need for improvements in high-voltage control
- Established baseline for future enhancements
- Demonstrated importance of proper configuration

These results provide valuable insights for future development iterations and system improvements. The testing process has highlighted both the current system's capabilities and areas requiring additional development focus.

Chapter 6

Future Improvements

Based on the current implementation and identified opportunities for enhancement, several key areas for future improvement have been identified. These improvements aim to increase system reliability, expand functionality, and improve user experience.

6.1 Sensor Enhancement

6.1.1 Robust Moisture Sensor Algorithm

- Implementation of advanced filtering algorithms for more accurate readings
- Development of calibration routines for different soil types
- Integration of temperature compensation for moisture readings
- Addition of multiple sensor support for averaged readings
- Implementation of predictive maintenance alerts based on sensor degradation patterns

6.2 Power System Upgrades

6.2.1 High-Voltage Device Support

- Integration of industrial-grade relay modules capable of handling:
  - Higher voltage pumps (up to 240V AC)
  - Multiple pump systems
  - Variable speed pump control
- Implementation of MOSFET-based switching circuits for:
  - Improved energy efficiency
  - Reduced wear on mechanical components
  - Silent operation

6.2.2 Enhanced Power Management

- Solar power integration capabilities:
  - Solar panel charging circuits
  - Battery management system
  - Power consumption optimization
- Low-power operation modes:
  - Deep sleep implementation
  - Wake-on-sensor-change capability
  - Scheduled activation periods
- Backup power systems:
  - UPS integration
  - Automated failover mechanisms
  - Power loss notification system

6.3 Connectivity Enhancements

6.3.1 Advanced Wi-Fi Implementation

- Web Interface Development:
  - Historical data visualization
  - Remote control capabilities
  - Mobile-responsive design
- Cloud Integration:
  - Data backup and storage
  - Cross-device synchronization
  - Weather data integration
  - Automated scheduling based on weather forecasts
- Security Implementations:
  - SSL/TLS encryption
  - User authentication system
  - Access control levels
  - Secure OTA updates

6.4 Additional Feature Proposals

6.4.1 Smart System Integration

- Machine Learning Implementation:
  - Predictive watering schedules
  - Plant health monitoring
  - Resource optimization algorithms
- Environmental Monitoring:
  - Temperature sensors
  - Humidity monitoring
  - Light level detection
  - pH measurement
- Automation Expansion:
  - Multiple zone control
  - Custom watering profiles
  - Season-based adjustments
  - Integration with home automation systems

6.4.2 User Experience Improvements

- Mobile Application Enhancement:
  - Push notifications
  - Customizable alerts
  - Detailed system analytics
  - User-friendly configuration interface
- Documentation and Support:
  - Interactive setup guide
  - Troubleshooting wizard
  - Community forum integration
  - Regular feature updates

These proposed improvements represent a comprehensive roadmap for system evolution, focusing on reliability, functionality, and user experience. Implementation priority should be based on user feedback and resource availability, with emphasis on maintaining system stability while introducing new features.

Chapter 7

Conclusion

The IoT-based smart irrigation system developed in this project represents a significant step forward in addressing the inefficiencies of traditional irrigation methods. By leveraging the ESP32 microcontroller, soil moisture sensors, and cloud integration, the system automates irrigation, ensuring that water is used only when necessary. This not only improves water efficiency but also enhances crop health and reduces the labor required for irrigation management.

The system successfully integrates sensor data with real-time control of a water pump, providing a foundation for scalable solutions in both small-scale and large-scale agricultural applications. Initial testing demonstrated the system's effectiveness in controlling low-voltage devices and highlighted areas for improvement, particularly in managing higher-voltage components. These challenges can be overcome with further iterations focused on improving power management and expanding the system's capabilities to support more demanding agricultural environments.

Future enhancements should include the integration of weather data, machine learning algorithms for predictive irrigation scheduling, and expanded connectivity options to ensure the system's adaptability in various contexts. Furthermore, improvements in sensor technology, power systems, and user interfaces will contribute to a more robust and user-friendly system.

Overall, this project demonstrates the potential of IoT technology in creating sustainable and efficient irrigation systems. With further development, it can contribute to solving global water management challenges, particularly in water-scarce regions like Morocco, while supporting the growth of more resilient agricultural practices.

Bibliography

[1] A. Morchid, et al., *IoT-based smart irrigation management system to enhance agricultural water security using embedded systems, telemetry data, and cloud computing*, Results in Engineering, 23, 102829, 2024.

[2] K. Pawlak, & M. Kołodziejczak, *The role of agriculture in ensuring food security in developing countries: Considerations in the context of the problem of sustainable food production*, Sustainability, 12(13), 5488, 2020.

[3] Food and Agriculture Organization (FAO), *The State of Food and Agriculture 2020: Overcoming Water Challenges in Agriculture*, Rome, Italy, 2020.

[4] T. Khokhar, *Chart: Globally, 70% of freshwater is used for agriculture*, World Bank Blogs, 2017.

[5] A. Maroli, V.S. Narwane, & B.B. Gardas, *Applications of IoT for achieving sustainability in the agricultural sector: A comprehensive review*, Journal of Environmental Management, 298, 113488, 2021.