**DEERWALK INSTITUTE OF TECHNOLOGY**

****

**Tribhuvan University**

**Faculty of Humanities and Social Sciences**

**WATER LEVEL TRACKING SYSTEM**

**A PROJECT REPORT**

**Submitted To:**

**Department of Computer Application**

**DWIT College**

***In partial fulfillment of the requirements for Bachelors in Computer Application***

**Submitted By:**

**Bipashree Aryal**

**6-2-1175-55-2020**

**DWIT COLLEGE**

**DEERWALK INSTITUTE OF TECHNOLOGY**

****

**Tribhuvan University**

**Faculty of Humanities and Social Sciences**

**DWIT College**

# **SUPERVISORS RECOMMENDATION**

I hereby recommend that this project prepared under my supervision by BIPASHREE ARYAL entitled **“WATER LEVEL TRACKING SYSTEM”** in partial fulfillment of the criteria for the degree of Bachelor's in Computer Applications to be taken further for evaluation.

Shyam Sundar Khatiwada

Project Supervisor

DWIT College, Sifal, Kathmandu

****

**Tribhuvan University**

**Faculty of Humanities and Social Sciences**

**DWIT College**

# **LETTER OF APPROVAL**

This is to certify that this project prepared by BIPASHREE ARYAL entitled **“WATER LEVEL TRACKING SYSTEM”** in partial fulfillment of the requirements of the degree of Bachelors in Computer Application has been well studied. It is gratifying in scope and quality as a project for the required degree.

|  |  |
| --- | --- |
| Shyam Sundar Khatiwada  Project Supervisor  DWIT College | Roshan Tandukar  Examiner  FoHSS, Tribhuvan University |
| Shyam Sundar Khatiwada  Project Coordinator  DWIT College | Hitesh Karki  Campus Chief  DWIT College |

# **ABSTRACT**

The Water Level Tracking System is an Internet of Things (IoT) project that combines the ESP32 Microcontroller and HC-SR04 Sonar Module to address water level management challenges in household and commercial tanks. This cost-effective solution provides real-time insights into water levels and removes the hassle of manual water filling tasks. By utilizing an iterative prototyping model and evaluating multiple microcontrollers, including Raspberry Pi, Arduino Uno, and ESP32, it was determined that the ESP32 offers the best value for the money.

The Water Level Tracking System fills a market void by providing an affordable and accessible smart solution for homeowners and tenants in Nepal. With its ability to accurately track water levels, this system helps alleviate the concerns associated with water management. This project has implications for a wide range of applications, benefiting both residential and commercial sectors that rely on water tanks in Nepal.

Overall, the Water Level Tracking System contributes to water conservation efforts and promotes efficient water usage in Nepal. It provides an effective tool for monitoring water levels and alleviating the risks of manual checking of water in tall tanks and the waste caused by overflow of water, ultimately simplifying the water management process for users.

**Keywords:** Internet of Things (IoT); ESP32 Microcontroller; HC-SR04 Sonar Module; Remote On/Off Functionality; Iterative Prototyping; Water Tanks.

# **ACKNOWLEDGEMENT**

I would like to take this opportunity to express my heartfelt gratitude to all those who have supported and assisted me throughout the completion of this individual project, the Water Level Tracking System.

Primarily, I extend my sincere appreciation to my supervisor, Mr. Shyam Sundar Khatiwada for their guidance, expertise, and invaluable feedback. Their continuous support and encouragement have been instrumental in shaping the project and ensuring its successful completion.

I would like to acknowledge the assistance and encouragement provided by my friends and family. Their unwavering belief in my abilities and their constant motivation have been a source of inspiration throughout this project.

I am grateful to Deerwalk Institute of Technology (DWIT College) for providing the necessary resources and facilities for the project.

I would like to express my thanks to Mr. Saroj Dhakal who participated in the testing phase of the Water Level Tracking System. His willingness to provide feedback and insights greatly contributed to the refinement and improvement of the system.

Lastly, I would like to acknowledge the contributions of all the researchers and developers in the field of IoT, whose work and advancements have laid the foundation for projects like mine.

Once again, I am sincerely thankful to all those who have played a part, big or small, in the completion of this project. Your support and assistance have been invaluable.

Bipashree Aryal

Roll No.: 6-2-1175-55-2020

# **TABLE OF CONTENTS**

[SUPERVISORS RECOMMENDATION i](#_Toc152186966)

[LETTER OF APPROVAL ii](#_Toc152186967)

[ABSTRACT iii](#_Toc152186968)

[ACKNOWLEDGEMENT iv](#_Toc152186969)

[TABLE OF CONTENTS v](#_Toc152186970)

[LIST OF FIGURES vii](#_Toc152186971)

[LIST OF TABLES viii](#_Toc152186972)

[LIST OF ABBREVIATIONS ix](#_Toc152186973)

[CHAPTER 1: INTRODUCTION 1](#_Toc152186974)

[**1.1.** **Overview** 1](#_Toc152186975)

[**1.2.** **Problem Statement** 1](#_Toc152186976)

[**1.3.** **Objectives** 2](#_Toc152186977)

[**1.4.** **Scope and Limitation** 2](#_Toc152186978)

[**1.4.1.** **Scope** 2](#_Toc152186979)

[**1.4.2.** **Limitations** 3](#_Toc152186980)

[**1.5.** **Report Organization** 3](#_Toc152186981)

[CHAPTER 2: BACKGROUND STUDY AND LITERATURE REVIEW 4](#_Toc152186982)

[**2.1.** **Background Study** 4](#_Toc152186983)

[**2.2.** **Literature Review** 4](#_Toc152186984)

[CHAPTER 3: SYSTEM ANALYSIS 5](#_Toc152186985)

[**3.1.1.** **Requirement Analysis** 5](#_Toc152186986)

[**i.** **Hardware Requirements** 5](#_Toc152186987)

[**ii.** **Functional Requirements** 5](#_Toc152186988)

[**iii.** **Non-Functional Requirements** 5](#_Toc152186989)

[**3.1.2.** **Feasibility Analysis** 6](#_Toc152186990)

[**i.** **Technical Feasibility** 6](#_Toc152186991)

[**ii.** **Operational Feasibility** 7](#_Toc152186992)

[**iii.** **Economic Feasibility** 7](#_Toc152186993)

[**3.1.3.** **Data Modeling** 8](#_Toc152186994)

[**3.1.4.** **Process Modeling** 9](#_Toc152186995)

[**3.2.** **SYSTEM DESIGN** 10](#_Toc152186996)

[**3.2.1.** **Circuit Design** 10](#_Toc152186997)

[CHAPTER 4: IMPLEMENTATION AND TESTING 11](#_Toc152186998)

[**4.1.** **IMPLEMENTATION** 11](#_Toc152186999)

[**4.1.1. Tools Used** 11](#_Toc152187000)

[**4.1.2. Implementation Details of Modules** 12](#_Toc152187001)

[**4.2.** **TESTING** 16](#_Toc152187002)

[**4.2.1. Test Cases for Unit Testing** 16](#_Toc152187003)

[**4.2.2. Test Cases for System Testing** 16](#_Toc152187004)

[CHAPTER 5: CONCLUSION AND FUTURE RECOMMENDATION 17](#_Toc152187005)

[**5.1.** **LESSON LEARNT/OUTCOME** 17](#_Toc152187006)

[**5.2.** **CONCLUSION** 18](#_Toc152187007)

[REFERENCES 19](#_Toc152187008)

# **­­LIST OF FIGURES**

[**Figure 1: Data-Flow Diagram for WLTS 8**](#_Toc152225107)

[**Figure 2: Block-Diagram for WLTS 8**](#_Toc152225108)

[**Figure 3: Flowchart for WLTS 9**](#_Toc152225109)

[**Figure 4: Circuit Diagram for WLTS 10**](#_Toc152225110)

[**Figure 5: Operating Principle of HC-SR04 [9] 12**](#_Toc152225111)

[**Figure 6: Components and Pinout of HC-SR04 [10] 13**](#_Toc152225112)

[**Figure 7: ESP32 Microcontroller [12] 14**](#_Toc152225113)

[**Figure 8: Icon for WLTS App 15**](#_Toc152225114)

[**Figure 9: Splash Screen for WLTS 16**](#_Toc152225115)

[**Figure 10: Home Screen for WLTS 16**](#_Toc152225116)

# **LIST OF TABLES**

[**Table 1: Cost of Development for WLTS** 7](#_Toc152186869)

[**Table 2: Symbols for Circuit Diagram** 11](#_Toc152186870)

[**Table 3: Test Cases for Unit Testing WLTS** 17](#_Toc152186871)

[**Table 4: Test Cases for System Testing WLTS** 17](#_Toc152186872)

# **LIST OF ABBREVIATIONS**

**IoT**: Internet of Things

**WLTS**: Water-Level Tracking System

**ESP32**: Espressif Systems-32

**HC-SR04**: High-Conductance Ultrasonic Sensor

**NPR**: Nepalese Rupee

**IDE**: Integrated Development Environment

**Wi-Fi**: Wireless Fidelity

**IO**: Input/Output

**RTDB**: Real-Time Database

**API**: Application Programming Interface

# **CHAPTER 1: INTRODUCTION**

## **Overview**

The Water Level Tracking System project was initiated due to personal experiences and concerns regarding the hassle of manually controlling water pumps and the decreasing groundwater levels during winter seasons. The project aims to address the problem of inefficient water level management in household and commercial water tanks.

The primary objective of the project is to develop a smart solution that simplifies water level management and optimizes the utilization of precious groundwater resources. providing real-time insights into water consumption patterns, users can make informed decisions regarding their water usage.

The scope of the project focuses on creating a user-friendly dashboard that accurately tracks water levels and offers real-time motor status. While the project aims to enhance water management practices, it is important to note that it does not inherently decrease water consumption. Rather, it empowers users with valuable information to better manage their water usage and promote conscious water management practices.

## **Problem Statement**

The current problem lies in the inefficient management of water levels in household and commercial water tanks, compounded by the hassle of manually controlling water pumps. This issue arises due to the lack of convenient and automated solutions that provide real-time monitoring of water levels. The existing systems require constant manual intervention, making it challenging for users to effectively manage their water resources. The evidence of this problem can be seen in the frustrating and time-consuming process faced by individuals in maintaining optimal water levels in tanks and the resulting wastage or shortage of water.

The Water Level Tracking System project proposes a solution to address this problem by developing a smart system that automates the control of water pumps and offers real-time monitoring of water levels. By incorporating advanced sensors and microcontroller technology, the project enables users to remotely monitor the water levels in their tanks. The system will provide accurate and timely information on water levels, empowering users to make informed decisions and take proactive measures to manage water efficiently.

## **Objectives**

1. Develop a Water Level Tracking System that automates water pump control based on real-time water level data, simplifying water level management in household and commercial water tanks.
   1. Integrate an ESP32 Microcontroller and HC-SR04 Sonar Module for accurate water level monitoring.
   2. Implement a Relay system for automatic execution of the water motor respective to the water level in the tank.
2. Provide insights into water levels through the Water Level Tracking System Flutter Application.
   1. Display Real-Time Water Levels on the Application Dashboard.
   2. Provide Water Pump Controls for Remote Control functionality of the Pump.

## **Scope and Limitation**

### **Scope**

1. The project focuses on the development of a Water Level Tracking System using the ESP32 Microcontroller and HC-SR04 Sonar Module.
2. The Application Dashboard will provide Real-Time Monitoring of water levels in household and commercial water tanks.
3. The System will automatically turn the Motor On/Off respective to the amount of water in the Tank.
4. The System will also provide Remote Control Functionality for the Water Pump.

### **Limitations**

1. The project’s scope is limited to monitoring the water level and fill the water automatically when needed. It does not include mechanisms to directly reduce water consumption.
2. The accuracy of the water level measurements may be affected by external factors such as temperature changes, tank material and sensor calibration.
3. The project assumes a stable internet connection for remote access and control of the Water Level Tracking System.

## **Report Organization**

This Project Report is Organized as follows:

1. Preliminary Section:

This Section consists of the Title Page, Abstract, Acknowledgement, Table of Contents, List of Figures, List of Tables, and the List of Abbreviations.

1. Introduction Section:

This Section contains the Overview, Problem Statement, Objectives, Scope, and the Limitation of the project.

1. Background Study and Literature Review Section:

In this section, the Background Study and the Literature Review are performed before the selection of the project is discussed.

1. System Analysis:

This section consists of the Requirement Analysis (Functional and Non-Functional), Feasibility Analysis (Technical, Operational, Economic), Data Modeling, and System Design.

1. Implementation and Testing:

The Implementation of the project and the testing conducted on the project along with the Test Cases, Input Data and Output Expected are discussed in this section.

1. Conclusion and Future Recommendation:

The Outcome of the Project, Conclusion, and the Future Recommendations to further improve the scope of the project are discussed in this section.

# **CHAPTER 2: BACKGROUND STUDY AND LITERATURE REVIEW**

## **Background Study**

The HC-SR04 Sonar Module [1] is an ultrasonic sensor that calculates distance by measuring the time it takes for ultrasonic waves to travel and return. By emitting ultrasonic waves and detecting their reflections, the module can determine the distance to an object. This functionality is essential for accurately measuring the water level in a tank, as the module can be positioned at the top of the tank to measure the distance to the water surface.

The ESP32 microcontroller [2], known for its Wi-Fi capabilities, can be used to post data to a Firebase Realtime Database [3]. Firebase provides cloud-based services, including a Realtime Database that allows for the storage and retrieval of data in real-time. By establishing a Wi-Fi connection and utilizing the Firebase API [4], the ESP32 can send HTTP POST requests containing relevant data. This data is stored in the Firebase Realtime Database, enabling users to access and monitor the water level information in real-time through the Water Tracking Level Dashboard.

The combination of the HC-SR04 Sonar Module's distance calculation capabilities and the ESP32's data posting functionality to Firebase Realtime Database forms the foundation of the Water Level Tracking System project. These technologies enable accurate water level measurements and seamless data communication, facilitating efficient monitoring and control of water levels in tanks.

## **Literature Review**

After conducting both internet and market research, it was discovered that there are no "Smart" solutions accessible in the market. Existing methods employ a Static On/Off Flow Control Valve, which turns on or off the Inlet Pipe dependent on the level of the water, but this does not manage the condition of the electric motor, resulting in long-term damage to the costly component.

Additional investigation in prominent online retailers yielded no results. Ivent Solutions Limited, situated in New Zealand, sells a comparable product named Smart-Water OnlineTM. Which provides identical functionality to the Water Level Tracking System but costs roughly $435 for the starting pack alone, which is a staggering 29-Times the price of the hardware that the Water Level Tracking System utilizes [5].

# **CHAPTER 3: SYSTEM ANALYSIS**

## **Requirement Analysis**

### **Hardware Requirements**

* ESP32 Microcontroller
* HC-SR04 Sonar Module
* Relay Module
* Water Pump

### **Functional Requirements**

* **Water Level Measurement**: The System must accurately measure the water level in the tank using the HC-SR04 Sonar Module. It should provide real-time data on the water level, enabling users to monitor and track changes.
* **Data Visualization**: The dashboard should present the water level information in a clear and intuitive manner. It should include visual representation such as graphs or charts that allow users to easily interpret and analyze the data.
* **Remote Control Functionality:** The system should provide the capability to remotely control the Water Pump and based on the water level in the tank. Users should be able to turn the pump On or Off through the application interface.

### **Non-Functional Requirements**

* **User-Friendly Interface**:The dashboard should have a user-friendly interface that is easy to navigate and understand. It should be intuitive and require minimal technical expertise to operate.
* **Real-Time Data Updates**: The system should provide real-time updates of water level measurements and pump status. Users should be able to access the latest information without significant delays.
* **Reliability and Accuracy**: The system should be reliable, providing accurate measurements and ensuring the proper functioning of remote-control functionality. It should minimize errors and maintain consistency in data reporting.
* **Security and Privacy**: The system should implement appropriate security measures to protect user data and ensure privacy.

## **Feasibility Analysis**

### **Technical Feasibility**

* **Hardware Availability**: The required hardware components, including the ESP32 Microcontroller and HC-SR04 Sonar Module, are readily available on the market for a combined amount of less than Rs. 1,500 NPR. These components are widely used and have established compatibility with each other, ensuring the feasibility of integrating them into the system.
* **Software and Programming**: The ESP32 Microcontroller can be programmed using various development environments, such as Arduino IDE or Platform-IO. These environments provide extensive libraries and resources for interfacing with sensors and implementing Wi-Fi connectivity for no additional cost whatsoever. The availability of these software tools and the documentation surrounding them make the development process feasible.
* **Communication and Data Management**: The integration of the ESP32 with the Firebase RTDB presents technical feasibility. The Firebase API provides the necessary functions for data posting and retrieval, allowing for real-time communication between the microcontroller and database. The availability of robust documentation and community support for Firebase facilitates the implementation of data management and synchronization features.
* **Sensor Accuracy and Performance**: The HC-SR04 Sonar Module is a widely used and reliable sensor for distance measurement. It offers satisfactory accuracy and performance, making it feasible for accurately measuring the water level in a tank. Furthermore, for improved accuracy, the system can be equipped with a DHT11/DHT 22 Sensor to measure the temperature and humidity of the tank and the distance can be calculated relative to the values obtained. However, this falls out of scope in this condition.
* **Wi-Fi Connectivity**: The ESP32 Microcontroller’s built-in Wi-Fi capabilities allow for seamless connectivity to the internet, facilitating real-time data transmission and remote-control functionality. The widespread availability of Wi-Fi networks ensures technical feasibility in establishing the required connections.

### **Operational Feasibility**

* **Install and Forget**: The WLTS is a One-time Install and Use Forever solution. This means that once the system is installed and configured, it can be utilized continuously without the need for recurring installations or significant maintenance efforts. This nature of the system offers operational benefits by reducing ongoing operational costs and minimizing disruptions to daily operations.
* **Ease of Use**: The dashboard should have a user-friendly interface that is intuitive and requires minimal technical expertise. Users should be able to navigate through the dashboard, access water level and motor data with ease.
* **Integration with Existing Systems**: The WLTS should seamlessly integrate with existing water management processes. It should be compatible with standard water tanks used in household/commercial settings. The system should not require significant modifications to the existing infrastructure, minimizing disruption to operations during installation and implementation.

### **Economic Feasibility**

* **Cost of Development**: Listed below is the total cost related to the hardware components used in the development of the Water Level Tracking System:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S/N** | **Particulars** | **Quantity** | **Rate** | **Price** |
| 1 | ESP32 Microcontroller | 1 | Rs. 900 | Rs. 900 |
| 2 | HC-SR04 Sonar Module | 1 | Rs. 350 | Rs. 350 |
| 3 | Relay Module | 1 | Rs. 150 | Rs. 150 |
| 4 | Printed Circuit Board (PCB) | 1 | Rs. 50 | Rs. 50 |
| **Total** | | | | Rs. 1,450 |

**Table 1: Cost of Development for WLTS**

* **Cost Savings**: WLTS offers potential cost savings by enabling efficient water management. By accurately monitoring water levels and providing insights to consumption patterns, users can optimize their water usage and minimize wastage. Additionally, this also provides a fail-proof measure to mitigate overflows in water tanks resulting in reduced wastage of water and consequently reduced water bills eventually.
* **Energy Efficiency**: The remote-control functionality of WLTS allows users to efficiently manage the water pump’s operation resulting in energy cost savings as well.

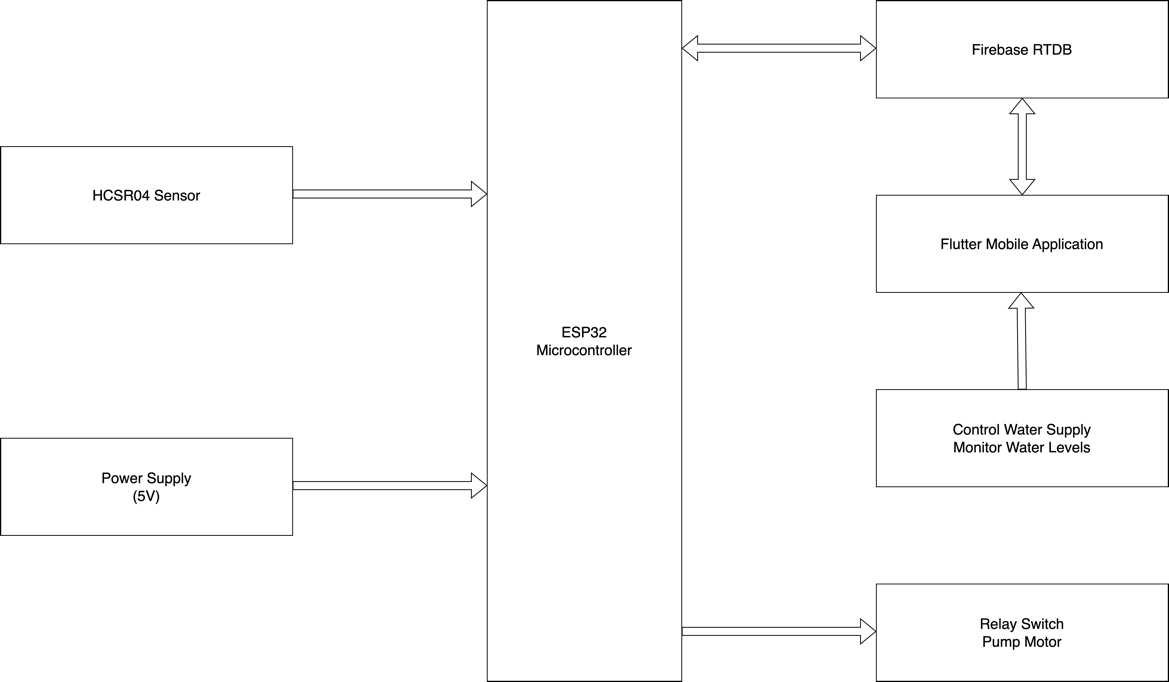
## **Data Modeling**

A black background with white rectangular labels

Description automatically generated

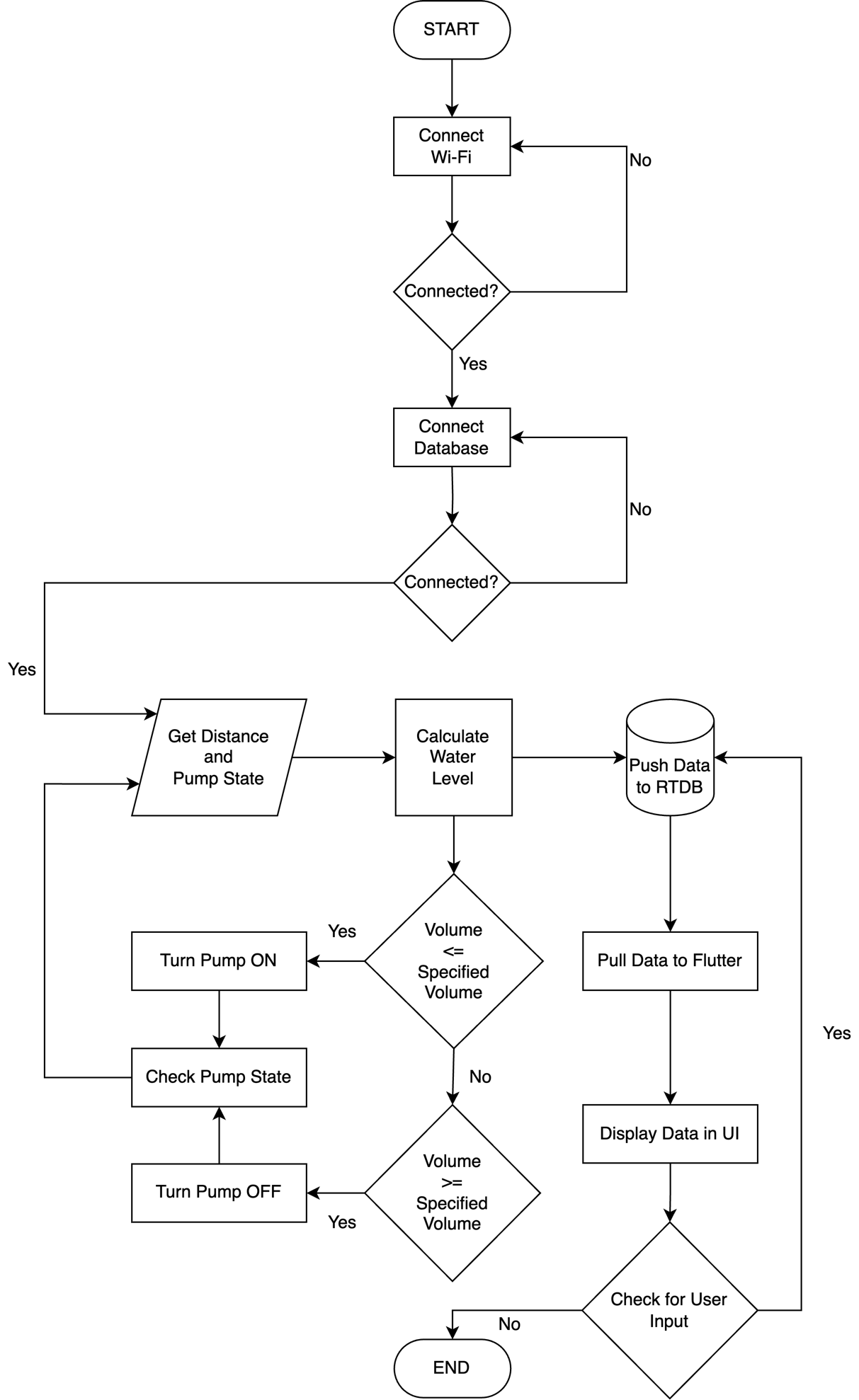
**Figure 1: Data-Flow Diagram for WLTS**

As Illustrated in the Data Flow Diagram above, the Water Level Tracking System works as follows: The water tank stores water, which is pumped in by the water pump. The water pump is controlled by the relay module, which receives on/off commands from the ESP32 microcontroller. The ESP32 microcontroller also receives water level data from the HC-SR04 sensor. If the water level is low, the ESP32 microcontroller sends an on command to the relay module, which turns on the water pump. If the water level is high, the ESP32 microcontroller sends an off command to the relay module, which turns off the water pump. The Firebase RTDB stores the water level and pump state data, which can be accessed by the user device to monitor the system.



**Figure 2: Block-Diagram for WLTS**

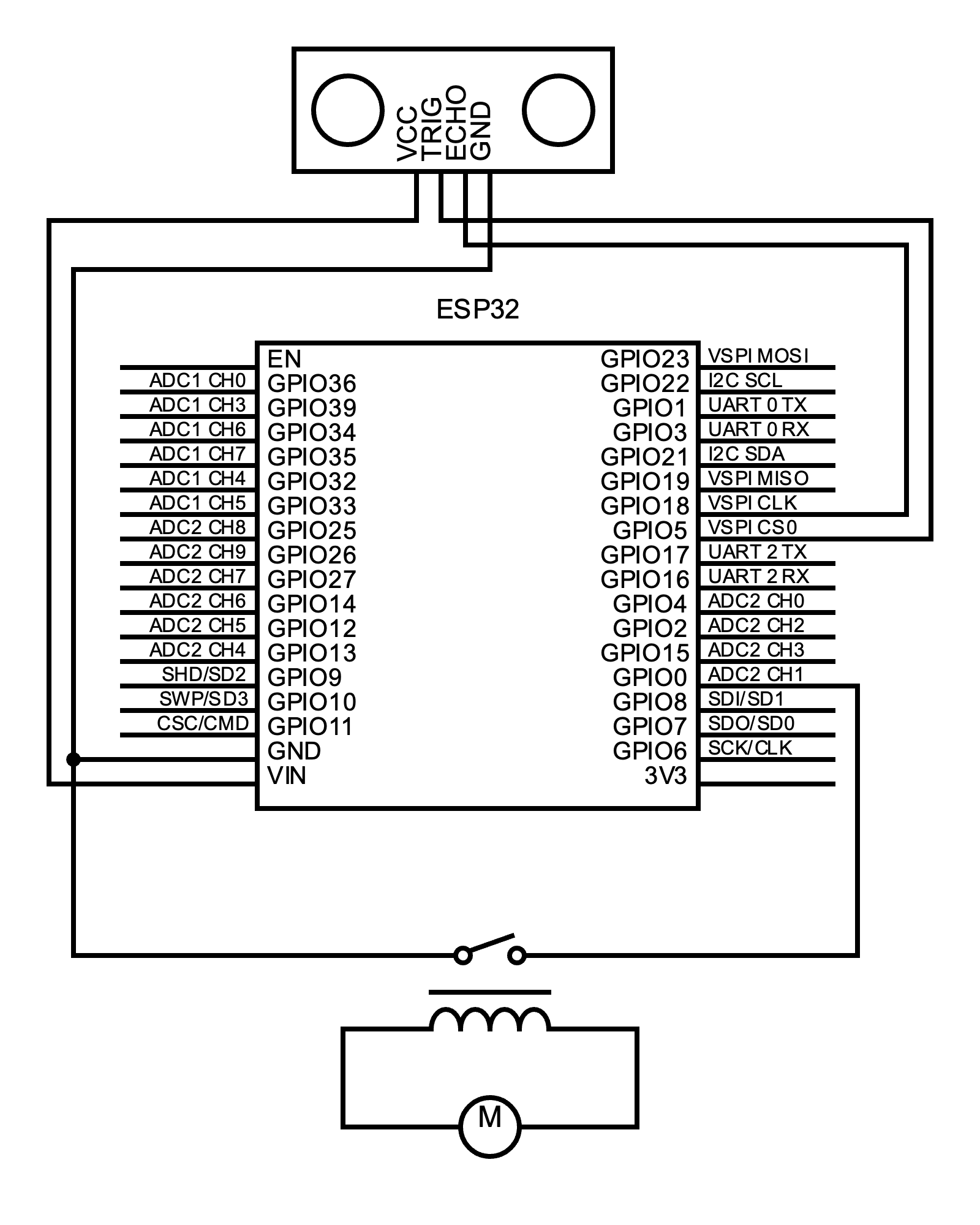
## **Process Modeling**



**Figure 3: Flowchart for WLTS**

## **SYSTEM DESIGN**

### **Circuit Design**



**Figure 4: Circuit Diagram for WLTS**

|  |  |
| --- | --- |
| Symbol | Name |
|  | ESP32 Microcontroller |
|  | HC-SR04 Module |
|  | Normally Open Relay |
|  | Water Pump Motor |

**Table 2: Symbols for Circuit Diagram**

# **CHAPTER 4: IMPLEMENTATION AND TESTING**

## **IMPLEMENTATION**

After evaluating diverse options, it was determined that the Raspberry Pi [6], while capable, is expensive and computationally overkill for the project's scale.

The Arduino Uno [7], on the other hand, is a reliable microcontroller but lacks built-in Wi-Fi or Bluetooth capabilities. To make it suitable for IoT-based projects, the addition of a wireless module is required. However, this not only increases the cost but also introduces technical complexities.

Considering these factors, the ESP32 [2] microcontroller emerged as the ideal choice for this project. It offers a great package for the price, with built-in wireless features that eliminate the need for additional modules. The ESP32 provides the necessary capabilities for seamless data communication and connectivity [8].

However, flashing the code to the ESP32 can be a tedious process. It has two modes: Download mode, used for flashing data to the chip, and Hardware mode, used for utilizing the ESP32 as a HID (Human Interface Device) Class device. One notable advantage of the ESP32 over the Arduino Uno is its ability to function as a HID device without requiring additional hardware.

During the code flashing process, it is necessary to hold the boot button on the ESP32 module until the computer establishes a connection with the chip. However, this inconvenience can be addressed by placing a 10μF capacitor between the Enable (EN) and Ground pins of the ESP32 module.

### **4.1.1. Tools Used**

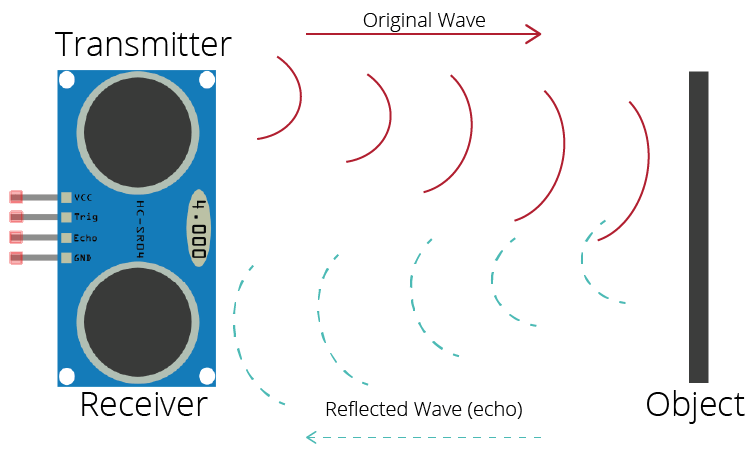
1. **Hardware Components:** ESP32 Microcontroller, HC-SR04 Sonar Module, 2-Channel Relay Module, Breadboard, Jumper Wires, Micro-USB Cable.
2. **Backend Code:** Arduino Programming Language (A Variation of C++).
3. **Frontend Code:** Flutter (A Framework of Dart Programming Language).
4. **Database Platform:** Google Firebase Real-Time Database.
5. **IDEs/Code Editors:** Arduino IDE, Visual Studio Code.
6. **Serial Monitoring Software:** Open Serial Port Monitor.
7. **Diagrams and Charts:** Draw.io, TeamGantt, Circuit-Diagram.org.
8. **Documentation:** Microsoft Word, Microsoft PowerPoint.

### **4.1.2. Implementation Details of Modules**

The Implementation of Modules in WLTS can be differentiated into multiple layers as discussed below:

1. Hardware Details:
   * **HC-SR04 Sonar Module**
     1. Operating Principle:

The HC-SR04 operates on the principle of ultrasonic echo-ranging. It emits ultrasonic pulses and measures the time it takes for the pulses to bounce back after hitting an object.



**Figure 5: Operating Principle of HC-SR04 [9]**

* + 1. Specifications:
       - Operating Voltage: The HC-SR04 typically works in the range of 5V DC.
       - Operating Frequency: The ultrasonic waves are usually around 40KHz.
       - Measuring Range: The HC-SR04 can measure distances ranging from 2cm to 400cm with an accuracy of about 3mm.
    2. Components:



**Figure 6: Components and Pinout of HC-SR04 [10]**

* + - * Ultrasonic Transmitter:

The ultrasonic transmitter is responsible for emitting short ultrasonic pulses. These pulses travel through the air until they encounter an obstacle or a surface. The typical operating frequency of the ultrasonic transmitter in the HC-SR04 module is around 40KHz. This frequency is chosen for its balance between distance accuracy and interference reduction.

* + Ultrasonic Receiver:

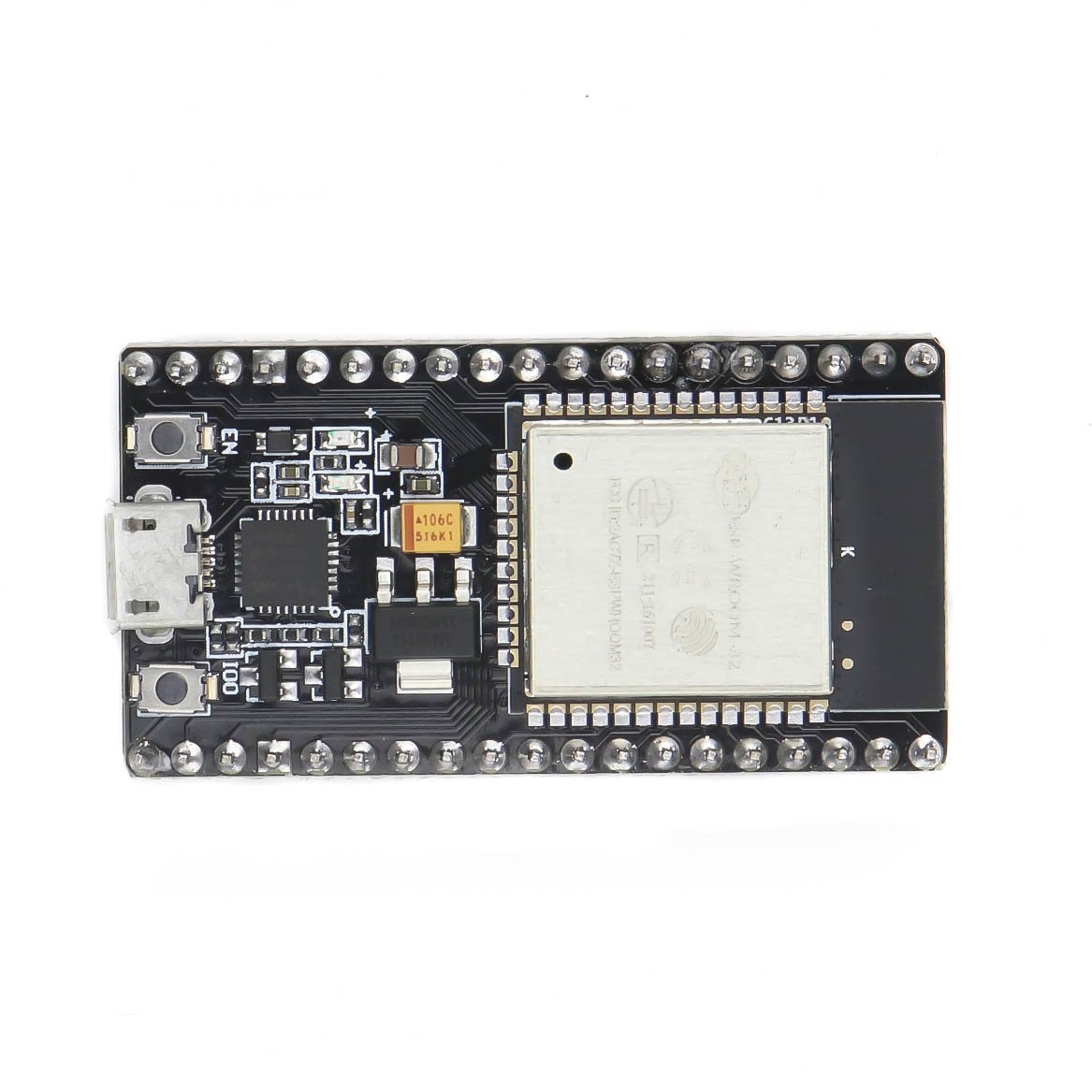
The ultrasonic receiver in the HC-SR04 module detects the echoes of the ultrasonic pulses. When the emitted pulses hit an object, they bounce back as echoes, and the receiver captures these echoes. The sensitivity of the receiver is designed to capture the reflected signals and convert them into electrical signals and convert them into electrical signals that can be processed by the control unit.

* + Control Circuit:

The control circuit of the HC-SR04 module manages the timing and signal processing. It controls the emission of ultrasonic pulses, measures the time it takes for the echoes to return, and process this information to calculate the distance. The control circuit also consists of a Crystal Oscillator of 4MHz as an internal system clock.

1. Connection Pins:
   * VCC: Power Supply (5V)
   * Trig (Trigger): Trigger Input for initiating the measurement.
   * Echo: Output Pin that produces a pulse proportional to the distance measured.
   * GND: Ground Connection.
   * **ESP32 Microcontroller:**
     1. Microcontroller Overview:

The ESP32 is a powerful and versatile microcontroller developed by Espressif Systems [11]. It is a part of the ESP family, known for its integrated Wi-Fi and Bluetooth capabilities.



**Figure 7: ESP32 Microcontroller [12]**

* + 1. Key Features:
  + Dual-Core Processor:

The ESP32 features a dual-core Tensilica Xtensa LX6 processor [13], providing high processing power and multitasking capabilities.

* + Wireless Connectivity:

The ESP32 has built-in Wi-Fi, Bluetooth Classic and Bluetooth Low Energy (BLE), allowing for versatile communication options.

* + Peripheral Interfaces:

The ESP32 provides various interfaces, including GPIO (General Purpose Input/Output), I2C (Inter-Integrated Circuit), SPI (Serial Peripheral Interface), UART (Universal Asynchronous Receiver-Transmitter) and more.

* + Low Power Features:

The ESP32 is designed to be power-efficient, with various low-power modes for energy conservation.

1. Hardware Implementation:

* **Sensor Integration:** Connection between the HC-SR04 Sonar Module and the ESP32 Microcontroller.
* **Pump Control Interface:** Interfacing between the ESP32 with the Relay Module to enable automated functionalities.

1. Communication Implementation:

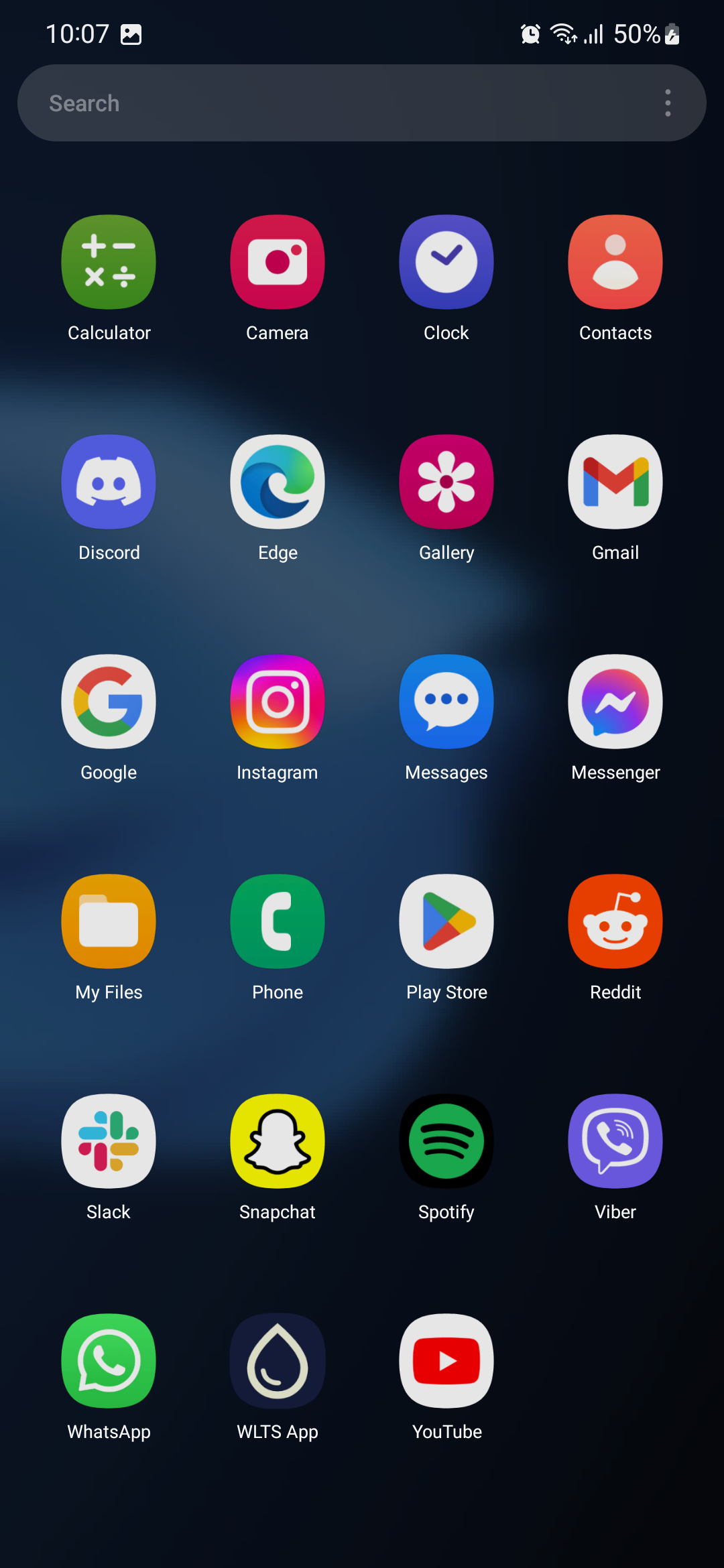
* **Wi-Fi Module:** Utilizing the built-in Wi-Fi capabilities of the ESP32 along with the Wi-Fi Library for connection to the internet and subsequently the Firebase Database.
* **Firebase Integration:** Interfacing the ESP32 with the Firebase Realtime Database (RTDB) to store and retrieve water level data and pump control status using the Firebase-ESP32 Library.

1. User Interface Implementation:

* **Flutter Application**

The Flutter application serves as the user interface, providing a visually appealing and user-friendly experience. It fetches data from the Firebase RTDB, allowing users to monitor water levels and control devices remotely.

* + 1. Application User Interface
       - **Application Icon**



**Figure 8: Icon for WLTS App**

* + - * **Splash Screen**

A blue background with a white drop of water

Description automatically generated

**Figure 9: Splash Screen for WLTS**

* **Home Screen**

**A screenshot of a phone

Description automatically generated**

**Figure 10: Home Screen for WLTS**

* **Real-Time Updates:** Using Flutter’s Real-Time Listeners to receive live updates whenever the water level changes.

## **TESTING**

### **4.2.1. Test Cases for Unit Testing**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| S/N | Test Case | Test Input | Expected Outcome | Test Result |
| 1 | Sensor Reading | Simulated Water Level Data | Correct water reading is captured | PASS |
| 2 | Pump Control | System turns the pump On. | Pump state changes to ON | PASS |
| System turns the pump Off. | Pump state changes to OFF | PASS |
| 3 | Data Synchronization | New water level recorded | Real-Time update of water level on UI | PASS |
| 4 | Database Connectivity | Valid Database Credentials | Successful connection to the Firebase RTDB | PASS |
| Invalid Database Credentials | Connection error message displayed | PASS |

**Table 3: Test Cases for Unit Testing WLTS**

### **4.2.2. Test Cases for System Testing**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| S/N | Test Case | Test Input | Expected Outcome | Test  Result |
| 1 | Sensor Reading | Place the sensor in a water tank with known water levels | Water Level data is accurately captured and displayed. | PASS |
| Verify the displayed water level readings with the actual water levels. | Water level readings match the actual levels | PASS |
| 2 | Pump Control | The Water Level is below the Minimum Threshold | Pump state changes to ON | PASS |
| Water Level is above the Maximum Threshold | Pump state changes to OFF | PASS |
| 3 | Data Synchronization | Simulate changes in water level data | Real-Time update of water level on UI | PASS |

**Table 4: Test Cases for System Testing WLTS**

# **CHAPTER 5: CONCLUSION AND FUTURE RECOMMENDATION**

## **LESSON LEARNT/OUTCOME**

During the Water Level Tracking System project, several valuable lessons were learned that contributed to the overall development and implementation process. These lessons provided insights and guidance for future projects of a similar nature.

Firstly, the importance of selecting the right microcontroller for the project was highlighted. Through the evaluation of different options such as Raspberry Pi, Arduino Uno, and ESP32, it was realized that the ESP32 microcontroller proved to be the most suitable choice. It offered a good balance between cost, functionality, and ease of use, with its built-in wireless capabilities and affordability. This experience emphasized the significance of considering specific project requirements and constraints when selecting hardware components.

Additionally, the process of flashing code to the ESP32 highlighted the need for thorough testing and troubleshooting. Understanding the different modes of the ESP32, such as the download mode and hardware mode, was crucial in ensuring successful code deployment. The discovery of using a 10μF capacitor to address the boot button holding issue demonstrated the importance of resourcefulness and creative problem-solving when encountering technical challenges.

Lastly, the project highlighted the significance of thorough testing at different stages. From unit testing the individual modules to system testing the overall functionality, conducting comprehensive tests helped identify and rectify issues early on. The importance of a robust testing strategy became evident in ensuring the reliability, accuracy, and performance of the Water Level Tracking System.

In conclusion, the lessons learned throughout the project encompassed hardware selection, troubleshooting, and rigorous testing. These insights will undoubtedly guide future endeavors, emphasizing the significance of thoughtful decision-making, adaptability, and continuous improvement in achieving successful project outcomes.

## **CONCLUSION**

In conclusion, the Water Level Tracking System project has successfully tackled the challenge of water level management in household and commercial settings. By leveraging the ESP32 microcontroller and the HC-SR04 Sonar Module, this project has offered a valuable solution for monitoring water levels and controlling water pumps.

Through iterative prototyping and comprehensive testing, the project demonstrated the feasibility and effectiveness of using the ESP32 microcontroller as a cost-efficient and feature-rich platform. Its built-in wireless capabilities and affordability make it an ideal choice for implementing IoT-based water level tracking systems.

The successful implementation of the Water Level Tracking System project highlights the potential of IoT technologies in addressing real-world challenges. It offers a practical and affordable solution for water level monitoring and management, benefiting homeowners and industries in Nepal and beyond. The project's findings and insights can serve as a foundation for future endeavors in smart water management, fostering a more sustainable approach to water resource utilization.

## **FUTURE RECOMMENDATION**

# **REFERENCES**

|  |  |
| --- | --- |
| [1] | ElecFreaks, "HC-SR04 Ultrasonic Module User Guide," [Online]. Available: https://www.elecfreaks.com/blog/post/hc-sr04-ultrasonic-module-user-guide.html. |
| [2] | E. Systems, "Arduino-ESP32 Libraries," 2020. [Online]. Available: https://github.com/espressif/arduino-esp32/blob/master/libraries/WiFi/src/WiFi.h. |
| [3] | Google Inc., "Firebase Documentation," Google Inc., [Online]. Available: https://firebase.google.com/docs. [Accessed 03 04 2023]. |
| [4] | Mobizt, "Firebase-ESP32," 19 04 2019. [Online]. Available: https://github.com/mobizt/Firebase-ESP32. |
| [5] | "Smart Water Online," 16 02 2021. [Online]. Available: https://smartwateronline.com. |
| [6] | The Pi Foundation, "Raspberry Pi Documentation," [Online]. Available: https://www.raspberrypi.com/documentation/. [Accessed 23 03 2023]. |
| [7] | Arduino Inc., "Arduino Docs," [Online]. Available: https://docs.arduino.cc/. [Accessed 23 03 2023]. |
| [8] | J.-L. Aufranc, "CNX-Software," [Online]. Available: https://www.cnx-software.com/2020/03/24/know-the-differences-between-raspberry-pi-arduino-and-esp8266-esp32/. [Accessed 21 03 2023]. |
| [9] | R. Santos, "Random Nerd Tutorials," 2013. [Online]. Available: https://randomnerdtutorials.com/complete-guide-for-ultrasonic-sensor-hc-sr04/. |
| [10] | D. G. Chris, "Hackster," 2017 August 20. [Online]. Available: https://www.hackster.io/diyguyChris/ultrasonic-sensor-hc-sr04-arduino-project-021d11. |
| [11] | Espressif Systems, "ESP32 Technical References," [Online]. Available: https://www.espressif.com/sites/default/files/documentation/esp32\_technical\_reference\_manual\_en.pdf. [Accessed 22 03 2023]. |
| [12] | "Robu," Robu, [Online]. Available: https://robu.in/product/esp-wroom-32-esp32-wifi-bt-ble-mcu-module/. |
| [13] | Cadence Design Systems Inc., "Mirrobo," September 2014. [Online]. Available: https://mirrobo.ru/wp-content/uploads/2016/11/Cadence\_Tensillica\_Xtensa\_LX6\_ds.pdf. |
| [14] | A. N. R. L. Steven Sachio, "CORE," 14 12 2018. [Online]. Available: https://www.core.ac.uk. |
| [15] | F. Dev, "Flutter Documentation," [Online]. Available: https://docs.flutter.dev/. |
| [16] | M. M. A. K. V. M. Shweta Karad, "IoT Based Real Time Water Monitoring System for Smart City," *International Journal of Innovative Science and Research Technology,* p. 251, 2018. |
| [17] | S. C. Priya Jayakrishnan, "Water Level Monitoring System Using IoT," *International Research Journal of Engineering and Technology (IRJET),* p. 1817, 2017. |