Mohamed Khider University of Biskra
Faculty of Science and Technology
Department of Electrical Engineering

MASTER THESIS

Science and Technology
Communication
Networks and Communications
Réf.:

Presented and defended by:

Mimoune Jouheina – Merzougui Basma

On: Thursday, June 12, 2025

Water level control based on IoT System

Jury:

BENAKCHA Abdelhamid	Pr	University of Biskra	President
Ameid Sofiane	MAA	University of Biskra	Examiner
Bekhouche khaled	MCA	University of Biskra	Supervisor

Academic year : 2024 – 2025

Mohamed Khider University of Biskra
Faculty of Science and Technology
Department of Electrical Engineering

MASTER THESIS

Science and Technology
Communication
Networks and Communications

Réf.:

Water level control based on IoT System

In :……

Presented by: Mimoune Jouheina – Merzougui Basma

Favorable opinion of the supervisor: Dr. Bekhouche khaled

Favorable opinion of the jury president

……..

Stamp and signature

إهداء

الحمد لله حتى يبلغ الحمد منتهاه و أخيرا من قال أنا لها نالها

إلى من زين إسمي بأجمل الألقاب و من دعمني بلا حدود إلى من علمني أن الدنيا كفاح و سلاحها العلم و المعرفة ...داعمي الأول في مسيرتي و سندي و ملاذي بعد الله والدي .

إلى من جعل الله الجنة تحت أقدامها و من سهلت علي الشدائد بدعائها سر نجاحي و جنتي والدتي .

إلى من ساندوني بكل حب عند ضعفي و أزاحوا عن طريقي المتاعب , إلى من شد الله عضدي بهم إخوتي ريان , نهاد,هناء ,رانيا و سندي خليل إلى كل عائلتي .

إلى رفيقة الخطوة و شريكة الرحلة الدراسية و نعم الرفيقة بسمة شكرا على التمام و العون و الختام.

إلى صديقاتي ريان و سهام و نرجس وصفية و صفاء الداعمين لي في كل مرة.

إلى كل من ساهم في نجاح هذا المشروع الأستاذ و المشرف خالد بخوش و الزميلين غيلوبي إمام و رحاب محمد أمين .

أهديكم بكل فخر و حب ثمرة تخرجي و نجاحي و في الأخير الحمد لله على التمام و البلوغ و الختام.

ميمون جهينة

إهداء

ما سلكنا البدايات إلا بتيسيره وما بلغنا النهايات إلا بتوفيقه وما حققنا الغايات إلا بفضله

أهدي تخرجي الى الراحل الباقي في قلبي أبي

والى من أنارت ل طريق حياتي

أمي حبيبتي

والى من علمتني أن العطاء ليس له حدود

الى أمي الثانية التي لم تنجبني

الى سندي في هذه الحياة ومصدر الأمان

عائلتي

الى رفاق الخطوة الأولى و الخطوة ما قبل الأخيرة

شكرا لنفسي التي صبرت واجتهدت الى أن حققت

بسمة مرزوقي

Acknowledgement

First and foremost, all praise and thanks are due to God Almighty for illuminating my path and enabling me to complete this work.

I would like to express my profound gratitude and appreciation to my supervisor, Dr. Bekhouche Khaled. His patience, diligent guidance, and fruitful support were instrumental in bringing this research to its required form.

My sincere gratitude is also extended to the honorable members of the jury members Pr. BENAKCHA Abdelhamid and Dr. Amied Sofiane.

I thank them for graciously agreeing to evaluate this humble work and for dedicating their precious time to its reading and assessment.

I must also not forget to thank all the faculty members of the Department of Electrical Engineering and the Dean's office of the Faculty of Science and Technology at the University of Mohammed Khider- Biskra, for the knowledge and wisdom they have imparted to us throughout our academic journey.

الملخص

تعتبر الإدارة الذكية لموارد المياه حلاً بارزًا لمواجهة تحديات الكفاءة والاستدامة، حيث تقدم أنظمة إنترنت الأشياء (IoT) بديلاً فعالاً لطرق التحكم التقليدية. في هذا السياق، تكتسب أنظمة التحكم في مستوى المياه عن بعد أهمية كبيرة نظرًا لقدرتها على توفير المراقبة الفورية والتشغيل التلقائي. يهدف هذا المشروع إلى تصميم وتنفيذ نظام متكامل ومتعدد الاستخدامات لمراقبة والتحكم في مستوى المياه في الوقت الفعلي، مع التركيز على الموثوقية وكفاءة الطاقة وسهولة الاستخدام.

في هذه المذكرة، تم تصميم وتطوير نظام تحكم في مستوى المياه قائم على إنترنت الأشياء، يتمركز حول المتحكم الدقيق ESP32-Sبدأ العمل باختيار المكونات المادية المناسبة، حيث تم استخدام حساس الموجات فوق الصوتية HC-SR04 للقياس، ووحدة مرحل للتحكم في المضخة. لضمان اتصال موثوق، تم تطبيق استراتيجية اتصال متعددة النماذج تشمل Wi-Fi للتواصل مع منصة ThingSpeakالسحابية، و Bluetooth للتحكم المحلي المباشر، ووحدة SIM800L GSM كقناة تحكم احتياطية عبر الرسائل القصيرة(SMS) .

تم تطوير برنامج مدمج (Firmware) مخصص للمتحكم يقوم بتنفيذ منطق التحكم التلقائي بناءً على عتبات ديناميكية، مع تطبيق مرشح برمجي لزيادة دقة قراءات الحساس. بالإضافة إلى ذلك، تم تطوير تطبيق هاتف مخصص ليكون الواجهة المركزية والحصرية للمستخدم، مما يتيح له المراقبة والتحكم اليدوي. لضمان استمرارية العمل، تم تحسين استهلاك الطاقة بشكل كبير من خلال استخدام وضع السكون العميق للمتحكم واستبدال شاشة LCD الأولية بشاشة OLED. أثبتت الاختبارات النهائية نجاح النموذج الأولي في تحقيق جميع الوظائف المطلوبة، مع استجابة موثوقة في كل من أوضاع التحكم التلقائي واليدوي.

Abstract

Smart water resource management has become a prominent solution for addressing efficiency and sustainability challenges, with Internet of Things (IoT) systems offering an effective alternative to traditional control methods. In this context, remote water level control systems are of great importance due to their ability to provide real-time monitoring and automatic operation. This project aims to design and implement an integrated and versatile system for real-time water level monitoring and control, with a focus on reliability, power efficiency, and user-friendliness.

In this thesis, an IoT-based water level control system centered around the ESP32-S microcontroller was designed and developed. The work began by selecting appropriate hardware components, utilizing an HC-SR04 ultrasonic sensor for measurement and a relay module for pump actuation. To ensure reliable connectivity, a multi-modal communication strategy was implemented, incorporating Wi-Fi for communication with the ThingSpeak cloud platform, Bluetooth for direct local control, and a SIM800L GSM module as a backup control channel via SMS.

Custom firmware was developed for the microcontroller to execute automatic control logic based on dynamic thresholds, with a software filter implemented to increase sensor reading accuracy. Furthermore, a dedicated mobile application was developed to serve as the exclusive central user interface, allowing for monitoring and manual control. To ensure operational longevity, power consumption was significantly optimized by utilizing the microcontroller's Deep Sleep Mode and replacing an initial LCD with an OLED screen. Final tests demonstrated the prototype's success in achieving all required functionalities, with reliable performance in both automatic and manual control modes.

LIST OF FIGURES

Chapter I: IoT Based System

Chapter II: Hardware and Software Description

Chapter III: System Design and Implementation

List of Tables

List of Abbreviations

ADC Analog-to-Digital Converter

AMQP Advanced Message Queuing Protocol

API Application Programming Interface

BLE Bluetooth Low Energy

CAD Computer-Aided Design

CoAP Constrained Application Protocol

DAC Digital-to-Analog Converter

DDS Data Distribution Service

DRC Design Rule Check

EEPROM Electrically Erasable Programmable Read-Only Memory

GPIO General-Purpose Input/Output

GPRS General Packet Radio Service

GSM Global System for Mobile Communications

HAL Hardware Abstraction Layer

HTTP HyperText Transfer Protocol

HTTPS HyperText Transfer Protocol Secure

I2C Inter-Integrated Circuit

IoT Internet of Things

MCU Microcontroller Unit

MQTT Message Queuing Telemetry Transport

PCB Printed Circuit Board

SIM Subscriber Identity Module

SMS Short Message Service

SPI Serial Peripheral Interface

TCP Transmission Control Protocol

TLS Transport Layer Security

ToF Time-of-Flight

UI User Interface

Wi-Fi Wireless Fidelit

Table of contents

Table of contents

Chapter III: System Design and Implementation

Table of contents

General Introduction

General Introduction

In recent years, the rapid evolution of digital technologies has led to the emergence and widespread adoption of the Internet of Things (IoT), revolutionizing how devices communicate, analyze, and respond to real-world environments. IoT has become a cornerstone of modern automation systems by enabling seamless interaction between physical objects and cloud-based platforms. Through the integration of sensors, microcontrollers, and wireless communication technologies, IoT systems now facilitate remote monitoring, data analysis, and automated control in real time.

Among the many areas that have benefited from IoT, water resource management has become a key focus due to increasing global concerns over water scarcity, leakage, and inefficient distribution. Traditional water monitoring methods are often limited by manual operations, fixed timers, and lack of feedback, resulting in excessive water waste and energy consumption. Addressing these limitations, modern IoT-based water level monitoring systems offer intelligent, automated, and energy-efficient solutions.

The state of the art in IoT-based water level monitoring systems integrates advanced components such as ultrasonic distance sensors (e.g., HC-SR04), cloud platforms (e.g., ThingSpeak), and microcontrollers (e.g., ESP32, STM32), supported by robust wireless technologies including Wi-Fi, Bluetooth Low Energy (BLE), and GSM. These systems not only detect water levels accurately but also enable real-time data visualization, remote control of pumps, and alert notifications through mobile applications.

Recent developments have emphasized multi-protocol communication for redundancy and reliability ensuring system operability in diverse environments, even in the absence of internet access. Furthermore, open-source platforms such as MIT App Inventor have democratized mobile app development, allowing tailored interfaces for end-users to interact with their water management systems intuitively.

This project positions itself within this modern landscape by designing and implementing a complete IoT solution for smart water level control. It combines low-power embedded hardware, versatile communication modules, and cloud-based analytics into a unified and scalable system. The result is a reliable, cost-effective platform adaptable for domestic, agricultural, and industrial applications, contributing to the global push toward smart and sustainable resource management.

The thesis is structured into four main chapters. The first chapter provides a general overview of the Internet of Things (IoT) and its various applications. The second chapter delves into the wireless communication technologies employed in the project, including Wi-Fi, Bluetooth, and GSM. The third chapter presents a detailed description of all the hardware components and software tools selected for the system. Finally, the fourth chapter thoroughly explores the practical implementation phases, from circuit design and programming to prototype assembly, highlighting the challenges encountered and discussing the test results.

Chapter I: IoT based system

I.1 Introduction

In recent years, the rapid advancement of digital technologies has led to the emergence of the Internet of Things (IoT) as a groundbreaking innovation capable of reshaping how we interact with our environment. The IoT represents a shift from isolated, standalone devices to interconnected systems that communicate, analyze, and respond to real-world conditions in real time. From smart homes to precision agriculture and intelligent infrastructure, IoT technologies are becoming essential to modern life by enabling automation, efficiency, and remote management across various domains.

In this context, water management has emerged as one of the most critical applications of IoT. With the growing challenges of water scarcity, inefficient usage, and climate-induced fluctuations, it has become increasingly important to monitor and control water resources intelligently. Traditional methods often rely on manual inspections or fixed timers, which can result in water waste, energy inefficiency, and system failures. However, integrating IoT into water level control systems offers a practical and sustainable solution to these limitations.

This chapter lays the foundation for understanding how IoT systems operate and why they are particularly suited for applications such as automated water level monitoring and control. It begins by defining the IoT and its components, including sensors, microcontrollers, client-server communication, and cloud platforms. It then explores the architecture that underpins most IoT applications and examines how these elements come together to form a smart, responsive system. It also highlights the main challenges of IoT implementation, such as network reliability, data security, power consumption, and scalability. Finally, it discusses the specific implementation of IoT technology within our project, illustrating the real-world relevance of theoretical concepts.

I.2 IoT system

I.2.1 Overview of the IoT

The genesis of the Internet of Things (IoT) can be traced to 1982 with the internet-connectivity of a Coca-Cola vending machine. A pivotal development occurred in 1998 with the initial application of Radio Frequency Identification (RFID), a technology that subsequently became a foundational component of IoT. The term "Internet of Things" was officially coined by Kevin Ashton in 1999. The following years witnessed significant milestones, including the first mention of the term by the International Telecommunication Union (ITU) in its 2005 annual reports and the inaugural conference on IoT in 2008. This spurred nations such as Belgium (2009) and China (2010) to establish national action plans.

The last decade has been characterized by exponential growth and substantial commercial interest. Cisco projected the potential financial value of IoT could reach $14 trillion, reporting 25 billion connected devices before 2020. Projections indicated 29.4 billion connected devices in 2023. Intel forecasted that the IoT market value would reach $6.2 trillion by 2025, while Strategy Analytics predicted 38 billion connections by the end of 2025 and 50 billion by 2030.

Sensor production is expected to expand significantly across various sectors, including energy and mining (33%), power and utilities (32%), automotive (31%), industrial use (25%), health (22%), and retail (20%).

In terms of market valuation, the IoT sector was estimated at $25 billion in 2020 with a forecast to reach $6 trillion in 2025. Another market analysis valued the global IoT industry at $1.90 billion in 2018, predicting it will grow to $11.03 billion by 2026. Furthermore, IoT surpassed "Big Data" as a topic of discussion in 2014, with references to the term increasing from 15,000 in 2013 to 45,000 in 2014. It is anticipated to be the primary application area to benefit from future 5G and 6G networks [1].

I.2.2 IoT definition

The term "Internet of Things" (IoT) refers to a system of technological solutions that enable the identification of physical entities, as well as the collection, processing, and transmission of data within physical environments . The core of IoT lies in the interconnection of embedded devices within everyday objects through the Internet, allowing these devices to automatically send and receive data. In this context, "things" refer to physical devices equipped with wireless or cellular communication capabilities, enabling them to interact with one another. IoT offers multiple connectivity options, thereby enhancing network accessibility on a broad scale [2].

I.2.3 How the IoT works

At the foundation of the IoT is Internet Protocol (IP) and Transmission Control Protocol (TCP). These standards and rules form the basis for sensors, devices, and systems to connect with the Internet and with each other. The IoT processes data from the devices and communicates the information via wired and wireless networks, including Ethernet, Wi-Fi, Bluetooth, 5G and LTE cellular, radio frequency identification (RFID), and near field communication (NFC). Typically, IoT devices connect to IoT gateways or edge devices that collect data. They feed data to and from cloud computing environments, which store and process the information. A broad array of networking standards ensure that the data is then sharable and reaches the correct “thing,” thereby connecting the physical world with the digital.

Two basic types of connected devices exist: digital-first and physical-first. The former consists of machines and devices specifically designed with built-in connectivity, such as smartphones, streaming media players, mobile payment terminals, agricultural combines, and jet engines. Digital-first devices generate data and communicate with other machines through machine-to-machine (M2M) communications. Physical-first devices, on the other hand, include a microchip or a sensor with communication capabilities. For example, a key chain, a vehicle, or a medical device in a hospital may contain a chip added after it was manufactured that makes the object or product newly functional and traceable. Some observers classify products according to a more detailed spectrum of interactivity, consisting of not two categories but five, ranging from the pure digital (followed by digital first, dual use, and physical first) to pure device (without any digital capabilities).

The IoT allows people and systems to share data and content through social media and other online methods; monitor and control events remotely; and interact with others through mobile devices and other systems, such as gaming devices. For example, during the pandemic, connected

thermometers allowed epidemiologists to better understand the spread of COVID-19 by tracking people with fevers [3].

I.2.4 IoT application

1. Smart Homes

IoT enables the automation of household devices, such as lights, thermostats, and security systems. Devices like Amazon Echo and Google Nest allow users to control home functions remotely, **ex:** Smart thermostats (e.g., Nest), smart locks, and voice-controlled assistants (e.g., Alexa).

2. Smart Cities

IoT is applied in urban planning to manage resources efficiently, monitor traffic, reduce energy consumption, and improve public safety, **ex:** Smart parking systems, waste management, and street lighting optimization.

3. Healthcare

IoT devices help in remote health monitoring, fitness tracking, and even chronic disease management through wearable tech, **ex:** Fitness trackers (e.g., Fitbit), connected glucose monitors, and remote ECG monitoring.

4. Smart Transportation

IoT improves transportation systems with real-time vehicle tracking, fleet management, and predictive maintenance tools, **ex:** Connected cars, ride-sharing apps (e.g., Uber), and smart traffic systems.

5. Industrial IoT

IoT in manufacturing industries enables predictive maintenance, process automation, and real-time analytics, **ex:** Industrial robots, condition monitoring sensors, and supply chain optimization.

6. Smart Agriculture

IoT is used to monitor crop health, automate irrigation, and track livestock, **ex:** Soil moisture sensors, GPS-enabled tractors, and drone-based crop monitoring.

7. Smart Grid

IoT helps in monitoring and managing energy usage efficiently in power grids, reducing wastage and optimizing distribution, **ex:** Smart meters, demand response systems, and renewable energy integration.

8. Retail IoT

IoT is used in retail for inventory tracking, personalized customer experiences, and automated checkout systems, **ex:** Smart shelves, beacons for customer targeting, and RFID systems

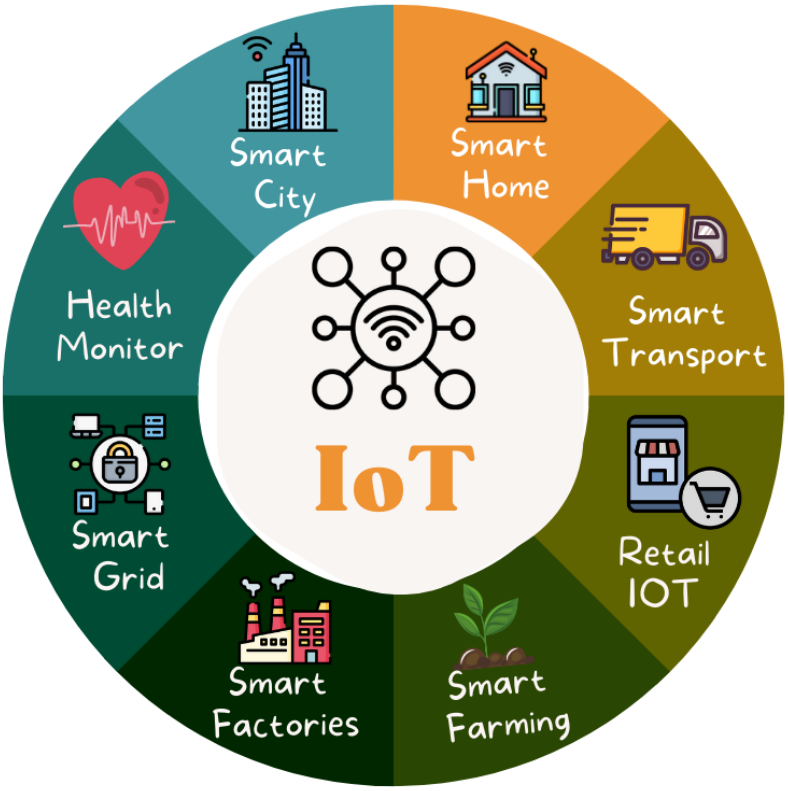


Figure I.1: IoT application [4]

I.2.5 IoT Architecture

The architecture of the Internet of Things (IoT) is typically organized into four main layers: the Sensing Layer, the Network Layer, the Data Processing Layer, and the Application Layer. Each layer plays a specific role in enabling the smooth operation of IoT systems.

I.2.5.1 Sensing Layer

This is the first and most fundamental layer in the IoT architecture. Its primary function is to gather data from the physical environment. It uses a variety of sensors and actuators to monitor parameters such as temperature, humidity, light, and sound. These devices are usually connected to the network through wired or wireless communication technologies, enabling the transmission of collected data to the next layers.

1.General Role of Sensors in IoT

Sensors are core components of any IoT system, acting as an interface between the physical world and digital systems. Their primary functions include:

- **Data Collection:** Measuring physical/chemical variables (e.g., temperature, humidity, pressure).

- **Signal Conversion:** Transforming analog signals into digital formats for processing.
- **Data Transmission:** Sending data to controllers or cloud platforms via communication protocols [5].

2.Classification of IoT Sensors

Sensors can be categorized based on their measurement type:

- **Environmental Sensors**: Monitor ambient conditions (e.g., air quality) Ex : DHT11 for temperature/humidity.
- **Mechanical Sensors**: Detect motion, pressure, or flow Ex: BMP280 pressure sensors.
- **Electromagnetic Sensors**: Track changes in electric/magnetic fields Ex : Ultrasonic proximity sensors.
- **Chemical Sensors**: Analyze chemical compositions (e.g., water quality) Ex : pH sensors [5].

3.Actuators

These are the counterparts to sensors. Instead of collecting data, they take action on the environment based on commands received from the system. They convert a digital signal back into a physical action. Examples include:

- An electric switch to turn a light on or off
- A motor to open or close a valve
- A servo to adjust a robotic arm
- A thermostat to change the room temperature

I.2.5.2 Network Layer

The network layer handles the transmission of data between devices and to external systems. It ensures that devices are properly connected and able to communicate efficiently. Technologies commonly used at this layer include Wi-Fi, Bluetooth, Zigbee, and cellular networks like 4G and 5G. Additionally, this layer may involve gateways and routers that manage data traffic and provide security features such as data encryption and user authentication to ensure safe and secure communication.

1.Gateways

- **IoT boards**

Programmable boards are small electronic devices that contain a processor, memory, and input/output (I/O) pins. They are used to build interactive electronic systems, automation, and Internet of Things (IoT) applications. These boards can be programmed to perform specific tasks using programming languages such as C/C++, Python, and others [6].

Figure I.2: type of IoT boards [7]

- **The Role of the IoT Board**

In IoT systems, the main board functions as the central unit that connects the physical environment to the digital world. It acts as an interface between hardware components—such as sensors and actuators and digital platforms like cloud services and user applications. The main board is responsible for processing the data collected by sensors and either transmitting it to cloud platforms for further analysis or initiating immediate responses through actuators based on predefined logic.

Moreover, the main board is essential for enabling network connectivity. It typically supports a variety of communication technologies, including Wi-Fi, LTE, Zigbee, and LoRa, which facilitate efficient and reliable data exchange between IoT devices and cloud-based systems.

Selecting an appropriate main board depends on various factors, including the level of processing power required, available connectivity options, physical size, energy efficiency, and overall cost. For example, Arduino boards are well-suited for basic projects with limited computational demands, Raspberry Pi is ideal for applications that require more advanced processing or graphical output, and the ESP32 is a popular choice for projects that need wireless communication with low power consumption.

2.Communication Protocols

- **Short-Range Protocols**

Wi-Fi (Wireless Fidelity)

Based on the IEEE 802.11 standard, the Wi-Fi protocol operates within a star network topology to connect devices to a central access point. It is characterized by its ability to provide very high data throughput, making it ideal for IoT applications that require transmitting large amounts of data, such as video streaming. However, its high power consumption makes it suitable primarily for devices connected to a permanent power source, such as security cameras, smart home systems, and advanced industrial sensors.

Bluetooth and Bluetooth Low Energy (BLE)

Bluetooth, and specifically its Low Energy (BLE) version designed under the IEEE 802.15.1 standard, is a fundamental protocol for creating Wireless Personal Area Networks (WPAN). BLE has been optimized for IoT applications by enabling devices to operate for years on a single battery through the intermittent transmission of small data packets. This makes it the dominant technology in wearables, medical sensors, indoor positioning beacons, and any application requiring energy-efficient wireless connectivity.

Zigbee

The Zigbee protocol, based on the IEEE 802.15.4 standard, is a specialized solution for creating low-power control networks, with its primary strength lying in its support for a mesh networking topology. This architecture allows each device in the network to act as a signal repeater, creating an extensive, robust, and self-healing network capable of finding alternative routes if a node fails. Due to its low power consumption and high reliability, it is considered the optimal choice for home and building automation (BMS) applications and industrial control networks.

NFC (Near Field Communication)

NFC is a wireless communication protocol that operates via electromagnetic induction over an extremely short distance of no more than a few centimeters, which is considered a fundamental security feature rather than a limitation. This characteristic requires intentional physical proximity, making NFC ideal for secure transactions like contactless payments, identity verification and access control systems, and simplifying the device commissioning process with a single touch.

- **Low-Power Wide-Area Network (LPWAN) Protocols**

LoRaWAN

LoRaWAN is a Low-Power Wide-Area Network protocol designed to achieve an exceptional communication range of several kilometers with a battery life that can exceed ten years. The protocol operates on top of the LoRa physical layer, which uses a spread spectrum modulation technique, and within a star-of-stars architecture. This makes it an ideal solution for applications that transmit small amounts of data infrequently across vast geographical areas, such as smart agriculture, smart cities, and asset tracking.

NB-IoT (Narrowband IoT)

As a 3GPP standard that operates on licensed cellular networks, NB-IoT is specifically designed to support a massive number of static IoT devices at a low cost and with minimal power consumption. By utilizing a narrow bandwidth, it provides superior signal penetration capabilities inside buildings and in underground locations, making it the preferred technology for applications like smart water and gas meters and urban infrastructure sensors.

LTE-M (LTE for Machines)

LTE-M, another 3GPP standard, serves as an evolution of traditional LTE networks to support IoT applications requiring higher performance than NB-IoT. It offers higher data rates, lower latency, and supports full mobility and even voice communications (VoLTE). This makes it suitable for more complex applications such as mobile asset trackers, medical wearables, and point-of-sale (PoS) systems.

- **Cellular and Satellite Protocols**

Cellular (4G/5G)

Traditional cellular communication networks, such as 4G and 5G, provide the high-reliability, high-bandwidth connectivity essential for critical IoT applications. While 4G provides sufficient speeds for video streaming, 5G introduces transformative capabilities like Ultra-Reliable Low-Latency Communication (URLLC) for controlling robotics and autonomous vehicles, and massive Machine-Type Communications (mMTC) to support immense device densities, albeit at a higher cost and power consumption.

Satellite

Satellite communication provides the ultimate solution to the coverage problem, ensuring connectivity at nearly any point on Earth, including oceans and remote areas unserved by terrestrial networks. Despite its higher costs and greater latency, it is an indispensable technology for applications such as global maritime shipping tracking, monitoring energy pipelines, and precision agriculture in geographically isolated regions.

I.2.3.3 Data Processing Layer

At this stage, raw data collected by the sensing layer is processed and analyzed. The data processing layer includes both hardware and software components that organize, interpret, and extract useful information from the data. Tools like data analytics platforms, data management systems, and machine learning algorithms are often used here. For instance, a data lake a centralized storage system is commonly employed to hold large volumes of raw data for further analysis and decision-making.

1. IoT Platforms

These are not just "software suites," but rather integrated systems offered as a Platform as a Service (PaaS), specifically designed to accelerate and simplify the development of IoT applications. These platforms provide a ready-made infrastructure that handles complex and repetitive tasks, allowing developers to focus on the application logic itself. Their primary functions include:

- **Device Management:** Registering and authenticating devices, monitoring their status, securing communication, and providing Over-the-Air (OTA) software updates.
- **Data Ingestion:** Establishing secure and scalable channels to receive data streams from millions of devices simultaneously.
- **Rules Engine:** A software component that allows for the definition of simple "If-This-Then-That" (IFTTT) logic, which triggers automated actions based on incoming real-time data, such as sending an alert when a temperature exceeds a certain threshold.
- **Integration:** Providing Application Programming Interfaces (APIs) to easily connect IoT data with other cloud services (such as storage, machine learning, and business intelligence tools).

2.Data Storage

The nature of IoT data characterized as time-series data produced in massive volumes and at high velocity requires specialized storage solutions. For this reason, **NoSQL (non-relational) databases** are often used because they offer flexibility and horizontal scalability. Among them, **time-series databases** like InfluxDB or TimescaleDB stand out. They are specifically designed to optimize the performance of storing and retrieving data associated with a timestamp, which is the essence of sensor data.

3. Data Analytics and Processing

This is the component that extracts actual value from the data. The analysis process can be divided into two main paths:

- **The Hot Path (Stream Processing):** This involves the immediate, real-time analysis of data as it flows in to make quick decisions. This path is responsible for instant alerts and automated responses, such as shutting down a machine upon detecting an abnormal vibration.
- **The Cold Path (Batch Processing):** This involves storing data for later analysis in batches. This path is used for complex analyses that do not require an immediate response, such as training machine learning models for predictive maintenance, analyzing historical patterns to optimize operations, or generating Business Intelligence (BI) reports.

4. Edge and Fog Computing

As a modern and important evolution of this layer, not all processing is done exclusively in the cloud anymore. The concept of **Edge Computing** involves moving a portion of the storage and analysis processes closer to the data source (i.e., at the device or local gateway). The goal is to reduce latency, lessen the burden on the network, and enable immediate decision-making without needing to connect to the cloud. **Fog Computing** acts as an intermediate layer between the "edge" and the "cloud," performing more complex analytical tasks on a local level. This hybrid Cloud-Edge model has become the standard in critical applications that require millisecond response times, such as autonomous vehicles and industrial robotics.

I.2.5.4 Application Layer

The application layer is the interface between the IoT system and its end-users. It delivers functionalities and services that allow users to interact with and manage connected devices. This layer includes mobile apps, web dashboards, and other user interfaces designed to provide real-time insights and control. It also supports middleware solutions that enable interoperability between different IoT systems. Furthermore, advanced analytics tools, such as visualization platforms and machine learning models, are often integrated to help transform complex data into actionable insights.

1.Core Functions:

- **Visualization and Monitoring:** This is the most direct function, where digital data is converted into easily understandable visual representations. This is not limited to numbers and text but extends to advanced data visualization techniques such as interactive dashboards, time-series graphs, heatmaps, and geospatial representations (GIS Maps) that display asset locations and statuses in real-time.
- **Control and Actuation:** This function enables the user to "close the control loop" in the IoT system. Through control interfaces, whether on a mobile application or a web-based control panel, the user can send commands to actuators in the physical layer—such as turning on a pump, adjusting a thermostat, or unlocking a door—transforming the system from a passive monitoring tool into an active control system.
- **Analytics and Reporting:** This function goes beyond real-time monitoring to provide historical and contextual perspective. Through these interfaces, users can generate custom reports, analyze long-term trends, compare performance, and extract Business Intelligence (BI) that supports strategic decision-making.
- **Alerting and Event Management:** This is a proactive mechanism to inform users of important events or anomalies that require their attention. In academic and advanced applications, this concept goes beyond simply sending an email; it includes defining different severity levels for alerts, routing them to the relevant personnel based on their roles, and providing a complete event log to facilitate audits and post-incident analysis.

2.Academic Design Considerations:

- **User Experience (UX) and Human-Computer Interaction (HCI):** Interfaces must be designed with a focus on the end-user. This requires a deep understanding of the context of use and the user's role (e.g., a technician's interface in the field should be simple and suitable for small screens, while an analyst's interface in the office can be rich with data and details).
- **Role-Based Access Control (RBAC):** In multi-user systems, it is essential to implement robust security models that ensure each user can only access the data and functions for which they are authorized according to their job role. This protects data privacy and prevents operational errors.
- **Scalability & Multi-tenancy:** The Application Layer must be built to scale horizontally to support a growing number of users and devices without performance degradation. In

commercial applications, it is often designed to support a multi-tenant architecture, where multiple independent organizations can use the same application instance while keeping their data securely isolated.

- **API-First Approach:** Modern application layers are often designed around Application Programming Interfaces (APIs). This approach not only facilitates the development of the system's own user interfaces (like web and mobile) but also opens the door for integration with external systems (such as ERP systems) and enables the creation of an entire ecosystem around the IoT data.

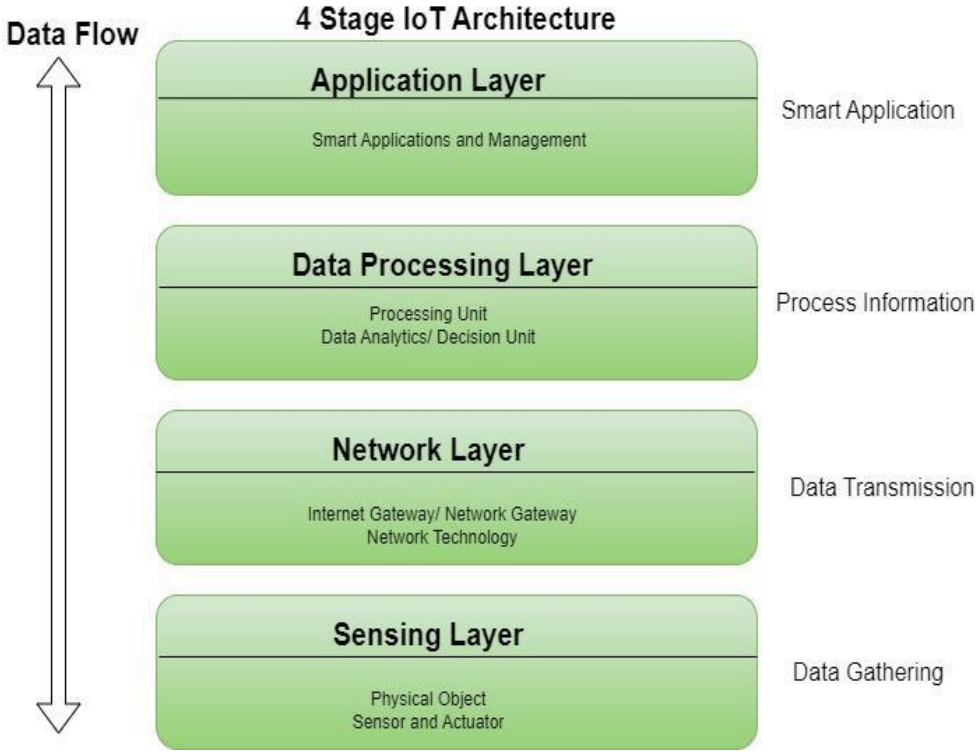


Figure I.3 : IoT Architecture[8]

I.2.6 Communication Models in IoT

The Internet of Things (IoT) relies on various communication models that define how data is exchanged between devices, servers, and users. These models are essential to the operation of IoT systems, as they determine the architecture and efficiency of communication flows. Depending on the nature of the application, different models may be adopted to ensure optimal performance. The four primary communication models in IoT are: **Device-to-Device (D2D)**, **Device-to-Server (D2S)**, **Device-to-Gateway (D2G)**, and **Server-to-Server (S2S)**.

1.Device-to-Device (D2D) Communication

In the Device-to-Device model, communication occurs directly between IoT devices without involving a central server or intermediary. This model is typically used in local or small-scale environments, such as smart homes, where, for example, a motion sensor may trigger a security camera when movement is detected. Common protocols employed in this model include **Bluetooth**, **ZigBee**,

Z-Wave, and **Wi-Fi Direct**. The D2D model offers low latency and reduced dependency on cloud infrastructure, making it ideal for applications that require fast, localized response.

2.Device-to-Server (D2S) Communication

The Device-to-Server model involves sending data directly from IoT devices to a centralized server or cloud platform. This model is widely adopted in scenarios where large-scale data processing, storage, and analysis are required such as in environmental monitoring systems or remote health diagnostics. Communication protocols like **HTTP/HTTPS**, **MQTT**, and **CoAP** are commonly used in this context. The main advantages of this model include scalability, accessibility of data from remote locations, and the ability to leverage cloud computing resources for advanced analytics.

3. Device-to-Gateway (D2G) Communication

In the Device-to-Gateway model, devices communicate with a local gateway that performs preliminary data processing before forwarding the data to a server or cloud platform. This approach is especially useful in systems with numerous low-power sensors, such as in precision agriculture, where devices send information to an edge device (e.g., a Raspberry Pi) acting as the gateway. The gateway may aggregate, filter, or encrypt data, thereby reducing bandwidth usage and enhancing security. Moreover, it helps extend the battery life of connected devices by offloading computational tasks.

4. Server-to-Server (S2S) Communication

Server-to-Server communication refers to the exchange of data between cloud servers or backend systems, typically after the data has been collected from IoT devices. This model is essential in complex, large-scale systems such as smart cities, where various subsystems like traffic control, energy management, and environmental monitoring need to interoperate. By enabling integration across platforms, S2S communication facilitates centralized decision-making, supports big data analytics, and enhances overall system intelligence and efficiency.

I.2.7 IoT Protocols

Communication protocols are essential for the functioning of Internet of Things (IoT) systems, as they enable diverse devices to exchange data. The varied requirements of IoT applications have led to the development of specialized protocols. This section will review the most significant of these protocols, with prominent examples including:

1. HTTP/s

Hyper Text Transfer Protocol is an application-layer protocol for transmitting hypermedia documents, such as HTML, across the Internet. It is the foundation of data communication, enabling web browsers (clients) and web servers to communicate effectively.It enables web browsers (clients) and servers to communicate using a request-response model. In this model, a client sends an HTTP request to a server, which then returns an HTTP response containing the requested content or an error message.HTTP is stateless, meaning each request is independent and doesn't retain information about previous interactions. It supports various methods like GET (to retrieve data) and POST (to submit data). Secure communication is facilitated through HTTPS, which incorporates encryption protocols

like TLS (Transport Layer Security) to ensure data integrity and confidentiality during transmission [9].

➢ **Structure of HTTP Messages**

- **Start Line:** Indicates the nature of the message.
- **Request Line (for requests):** Contains the HTTP method, Request URI, and HTTP version (e.g., GET /index.html HTTP/1.1).

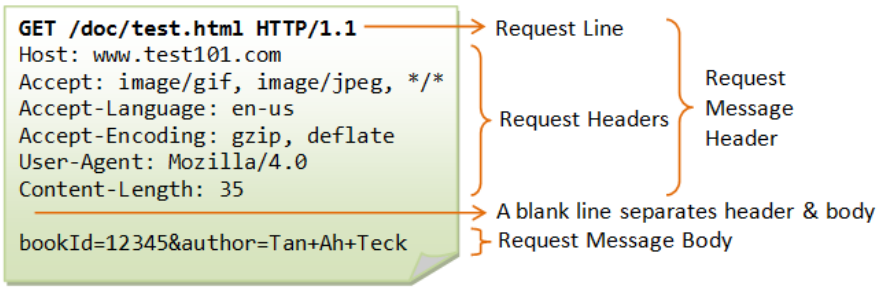


Figure I.4 : sample HTTP request message [10]

- **Status Line (for responses):** It contains the HTTP version, status code, and reason phrase (e.g., HTTP/1.1 200 OK).

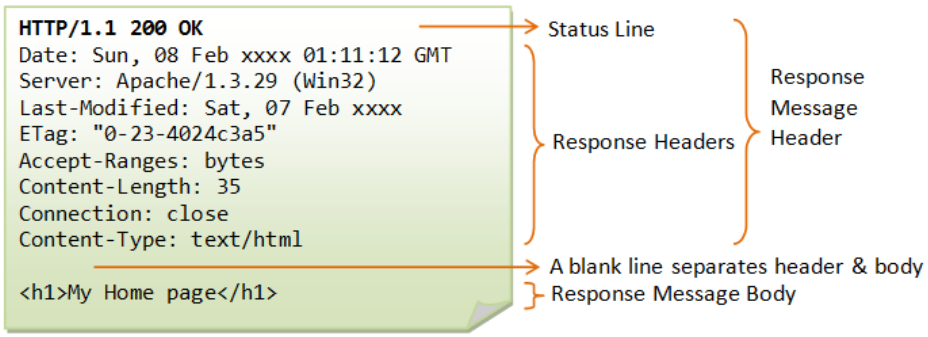


Figure I.5 : sample HTTP response message [10]

- **Headers:** Key-value pairs providing additional information about the request or response (e.g., Content-Type: text/html).
- **Blank Line:** A mandatory empty line indicating the end of the header section.
- **Message Body:** Optional part containing the data to be transmitted (e.g., HTML content, JSON data) [9].

2. CoAP

CoAP is a lightweight protocol designed for constrained IoT devices with limited processing, memory, and power. It operates over UDP, offering a simpler, low-overhead alternative to HTTP for

basic data exchange and commands. It also includes mechanisms for communication reliability in constrained networks [11].

3. AMQP

Advanced Message Queuing Protocol (AMQP) is an open standard application layer protocol for business messaging and reliable communication between different systems. It ensures interoperability with features like message queuing, routing, reliable delivery, transactions, and security. It's used in complex enterprise environments and IoT scenarios needing high

reliability [12].

4. DDS

Data Distribution Service (DDS) is an open standard from OMG for real-time systems needing high performance, scalability, and reliability. It uses a data-centric, publish-subscribe model, often operating without a broker. DDS is common in industrial, defense, aerospace, and healthcare applications [13].

5. WebSocket Protocol

The WebSocket protocol provides a full-duplex, persistent communication channel over a single TCP connection. Established via an HTTP "Upgrade," it allows for low-latency, low-overhead, bidirectional data flow, bypassing the traditional HTTP request-response model. It's suitable for real-time web apps, online gaming, and interactive IoT scenarios [14].

I.2.8 Challenges of Using IoT in Real-World Projects

While IoT technology offers significant advantages, its implementation in real-world systems also poses several challenges, particularly in resource management applications. Some of the main limitations include:

- **Network Reliability**: IoT systems rely heavily on stable internet connectivity. Any disruption can lead to data loss or delayed responses.
- **Data Security and Privacy**: Transmitting data over networks, especially through cloud services, increases the risk of cyberattacks or data breaches.
- **Power Consumption**: Many IoT devices, especially those operating on batteries in remote areas, require optimized energy usage to extend operational life.
- **Hardware Compatibility**: Integrating various sensors and communication modules can be difficult, especially when components are sourced from different manufacturers.
- **Scalability**: As the system expands (e.g., more sensors or users), ensuring consistent performance and reliable data processing becomes more complex.

I.2.9 Applications of IoT in Water Level Control Systems

Water level control systems are among the fields that have directly benefited from the integration of Internet of Things (IoT) technologies. By combining smart sensors and modern communication modules, effective solutions have been developed to manage and monitor water resources in real time. These systems rely on installing sensors designed to measure the water level in tanks or wells, connected to small processing units capable of collecting data and transmitting it to digital platforms or applications.

These solutions enable users to remotely monitor the water level via the Internet and to automatically control pumps or valves based on sensor readings, which contributes to improving operational efficiency, reducing waste, and preventing unnecessary consumption.

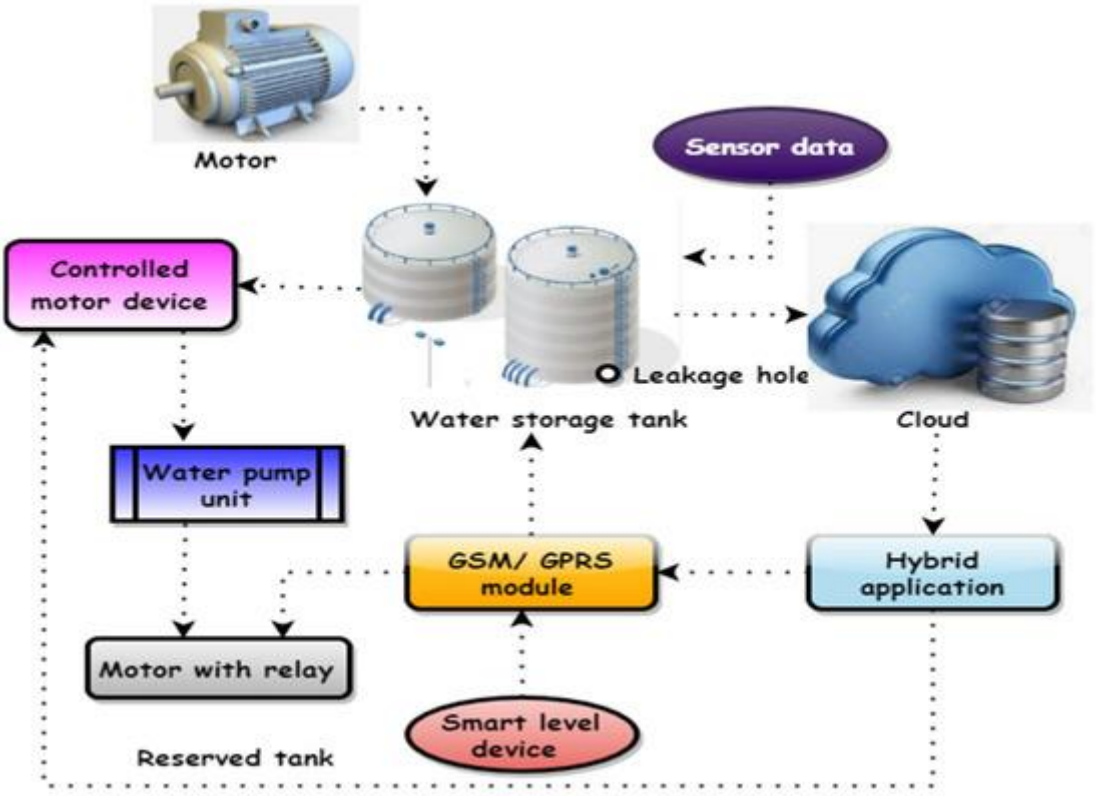


Figure I.6: Smart water tank using IoT [15].

IoT applications in water level control systems are diverse and cover various sectors, including:

1.Household tank management

Many households adopt smart systems to monitor tank levels and automatically refill them when they reach a predefined minimum threshold, sending alerts via text messages or notifications if the tank becomes empty or full.

2.Agricultural Sector

These systems enable smart irrigation by controlling water pumps and scheduling refilling times based on water level and crop requirements, helping to optimize water consumption and improve productivity.

3.Industrial and Commercial Facilities

Numerous factories and facilities require continuous monitoring of the water used in production processes. IoT systems provide precise tracking and record essential data for periodic reporting, in addition to automatically operating pumps when needed.

4.Public Water Resource Management

Municipalities and relevant authorities use IoT solutions to monitor water levels in reservoirs, dams, and distribution stations, ensuring a continuous supply of water and preventing emergencies caused by sudden drops or leaks.

These systems have several important features, including:

- **Real-time Supervision:** The ability to monitor water levels anytime and anywhere through web interfaces or mobile applications.
- **Automatic Alerts:** Notifications are sent when critical thresholds are reached or when a system malfunction occurs.
- **Automated Control:** Pumps and valves operate based on sensor readings without the need for constant human intervention.
- **Data Analysis:** Measurement data is stored and analyzed to improve resource management strategies.

With the continuous development of digital technologies, IoT applications in water level control systems are expected to expand further, both in terms of improved analysis and artificial intelligence capabilities and in the adoption of renewable energy sources to sustainably power sensors and end devices.

I.2.10 The future of IoT

Addressing these challenges requires a combination of robust hardware selection, secure communication protocols, and well-designed software solutions.

While the ability to connect physical objects and devices introduces increased efficiencies and, in some cases, cost savings, scaling up those connection points and networks creates greater possibilities, though not without some great risks and challenges. For example, a smart car that connects with a smartphone can already integrate mapping, entertainment, voice commands, and other functions that transform the vehicle into a computer on wheels, but a network of connected vehicles and infrastructure could potentially allow vehicles not only to avoid crashes while driving but also to "see" around corners and avoid collisions with a bicyclist or a pedestrian. In addition, sensors in bridges, tunnels, roads, and other infrastructure could indicate when repairs are necessary or when failure is imminent. Putting such innovations into practice, however, can be challenging. Current autonomous vehicles, for example, are already burdened with safety concerns and susceptibility to hackers.

Smart utilities and even smart cities could allow societies to use energy resources and transportation systems more effectively and at a lower cost than in the past. Connected devices inside and outside the body could revolutionize the way people monitor health conditions, allowing smart-connected devices to release the right amount of medication at the right place and time, and tiny robotic devices injected into the human body could detect and fix medical problems. Although no one can predict the exact course that these connected technologies will take, and the challenges and social concerns they may spur, it is clear that the IoT will continue to have a profound impact on lives and culture in the years ahead [3].

I.3 Conclusion

This chapter has provided a comprehensive overview of the Internet of Things and its relevance to modern system design. It explored key concepts such as IoT architecture, core components including sensors, microcontrollers, and cloud services, and examined their practical implementation in our water level monitoring system. By highlighting both the capabilities and challenges of IoT, this chapter forms a critical foundation for understanding how interconnected technologies can enhance resource management, efficiency, and automation in real-world applications.

Chapter II: Hardware and Software Description

II.1 Introduction

The success of any IoT-based totally system relies upon largely on the proper selection and integration of each hardware and software program components. This bankruptcy makes a speciality of the technical surroundings in which the water degree tracking and manipulate system become developed. It introduces the important thing hardware modules including the ESP32 microcontroller, water level sensors, and output actuators, and explains their roles within the system. Furthermore, the bankruptcy outlines the software tools and improvement systems used mainly the Arduino IDE and App Inventor to put in force the common sense and interface of the machine. By detailing the interplay between hardware and software, this chapter lays the groundwork for understanding how the system operates in actual-time to attain the desired capability.

II.2 Hardware

II.2.1 Microcontroller Selection

The choice of the microcontroller is an important solution in this project. This is because it has a direct impact on performance, energy consumption and development flexibility. Two strong candidates are ESP32-S and STM32 Blue Pill . ESP32-S provides built-in Wi-Fi and Bluetooth connections, so it's ideal for dual-core processors for IoT applications and multitasking. Meanwhile, STM32 Blue Pill (based on STM32F103C8T6) provides a reliable core of a well-known ecosystem for ARM Cortex-M3 Bark, a wide range of surrounding support and built-in development, but has no wireless opportunities. The final selection depends on the requirements of the project depends on the more important wireless connection processing ability and the control of the peripherals.

II.2.1.1.ESP-32S

The NodeMCU-32S is an ESP32-based development board featuring a dual-core processor, integrated Wi-Fi/Bluetooth connectivity, and extensive I/O capabilities (GPIO, ADC, PWM, SPI, I2C, UART). Its breadboard-compatible design facilitates rapid prototyping, while support for multiple development frameworks (Arduino, MicroPython, ESP-IDF) enhances its versatility for IoT applications. The board's low-power modes further optimize its suitability for embedded systems, making it a practical choice for industrial prototyping [16].

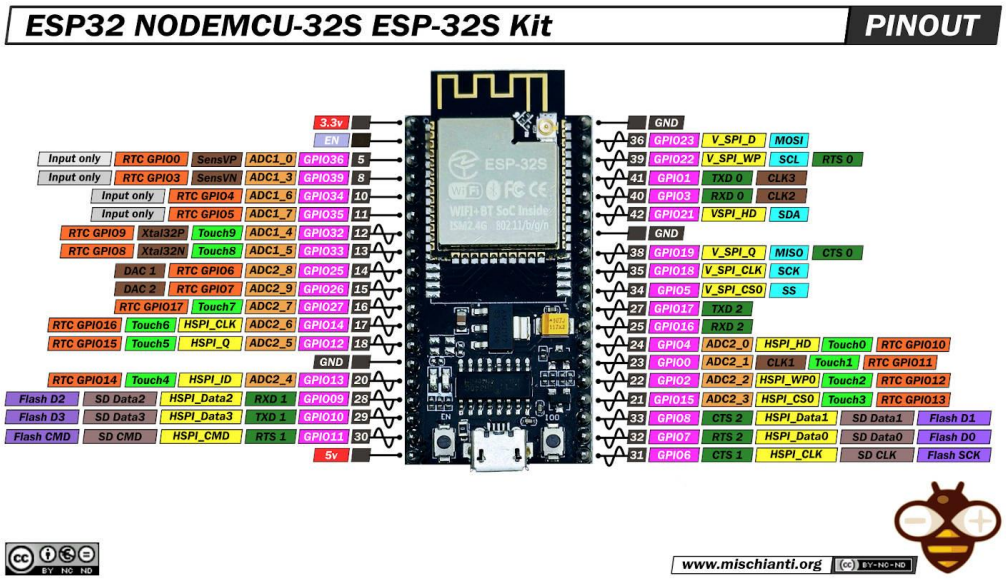


Figure II.1: ESP-32S Board pinout [16].

Table II .1: ESP-32S Technical Details

Property	Specification
Microcontroller	ESP32 Dual-core Tensilica LX6
Flash Memory	4MB
SRAM	520KB
Wi-Fi	802.11 b/g/n
Bluetooth	Bluetooth v4.2 BR/EDR and BLE
USB to Serial	CH340C
Operating Voltage	3.0V to 3.3V
Input Voltage (Vin)	5V via micro USB
Digital I/O Pins	34
ADC Channels	18 (12-bit)
DAC Channels	2 (8-bit)
PWM Channels	16
SPI/I2C/I2S/UART	Supported
GPIOs	Up to 34
Dimensions	Approximately 25.5mm x 48mm

II.2.1.2.STM Blue pill

STM32F103C8T6 Blue Pill Development Board contains a 32-bit Cortex-M3 RISC ARM core with an internal oscillator of 4 -16 MHz. It is a CMOS flash technology chip. This chip has 37 GPIO pins and 10 Analog pins. It has some modern communication interfaces like a CAN and a USB port. The peripherals give outstanding control of the board as it operates at very low voltage, so it is suitable for low-power applications. It also comes with an integrated watchdog and a window watchdog timer for the proper execution of instructions [17].

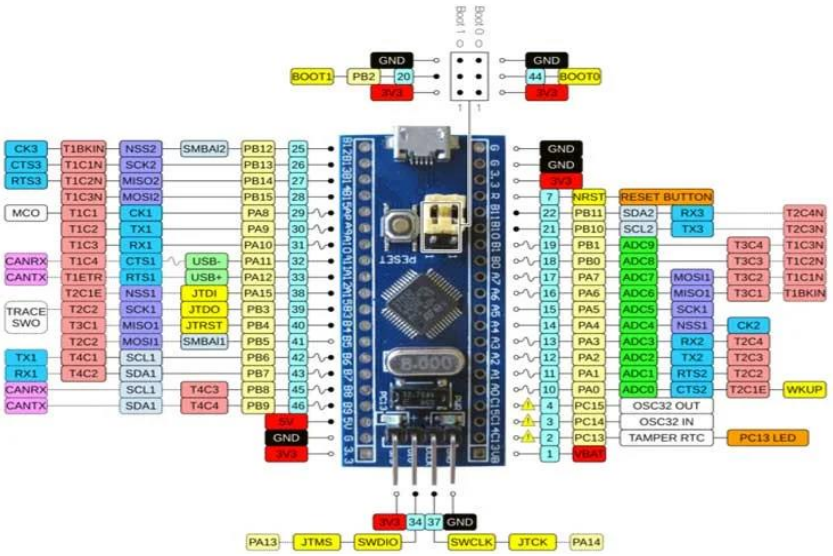


Figure II.2: STM Blue pill Board pinout [17].

Table II.2: STM32 Blue Pill Technical Details

Property	Specification
Microcontroller	STM32F103C8T6
Flash Memory	64KB
SRAM	20KB
Operating Voltage	3.3V
Input Voltage (Vin)	5V via USB or external
Digital I/O Pins	37
ADC Channels	10 (12-bit)
PWM Channels	15
SPI/I2C/UART	Supported
GPIOs	Up to 37
Clock Speed	72 MHz
USB	Full Speed USB 2.0
Dimensions	Approximately 18mm x 53mm

II.2.1.3.Detailed Comparison of Development Boards

The meticulous selection of core components, foremost among which is the microcontroller unit (MCU), plays a crucial role in determining the performance and reliability of any Internet of Things (IoT) based system. Therefore, we present below a technical comparison between the ESP-32S and STM32 Blue Pill boards, both popular choices in this field, to highlight their respective capabilities.

Table II.3: Comparative Analysis of the ESP-32S and STM32 Blue Pill Boards

Feature	ESP-32S	STM32 Blue Pill (STM32F103C8T6)
Built-in IoT Capabilities	Excellent (Wi-Fi 802.11 b/g/n, Bluetooth v4.2 BR/EDR & BLE)	No built-in wireless communication
Ease of Programming (Common Environments)	Easy to Medium (Arduino IDE, MicroPython, Espressif IDF)	Medium to Advanced (Arduino IDE with STM32 core, STM32CubeIDE, Keil MDK, PlatformIO)
Processor (Core & Speed)	Dual-core Tensilica LX6, up to 240MHz	Single-core ARM Cortex-M3, up to 72MHz
Floating Point Unit (FPU)	Yes (for both cores)	No (software emulation if needed, slower)
Memory (Flash/RAM)	Typically 4MB-16MB Flash, ~520KB SRAM	Typically 64KB-128KB Flash, 20KB SRAM
Power Consumption (Operating Modes)	Medium (tens of mA in active mode). Multiple sleep modes (Light Sleep, Deep Sleep) down to µA range.	Low to Medium (tens of mA in active mode). Sleep modes (Sleep, Stop, Standby) down to µA range.
Key Peripherals (I/O)	GPIOs (multifunctional), ADC (12-bit), DAC (8-bit), SPI (x3), I2C (x2), UART (x3), I2S, CAN, Touch Sensor, Hall Sensor, built-in Temperature Sensor (low accuracy)	GPIOs, ADC (12-bit x2), SPI (x2), I2C (x2), UART (x3), CAN, Timers (advanced and multiple)
Operating Voltage	3.0V - 3.6V (typically 3.3V)	2.0V - 3.6V (board typically 3.3V, MCU is 5V tolerant on some pins)

On-board Power Management	Limited; requires external circuitry for advanced battery charging/management.	None; requires entirely external circuitry.
Debugging Capabilities	Via UART (for serial printing), JTAG (requires setup). OTA debugging possible.	SWD (Serial Wire Debug) - well-supported with ST-Link, JTAG.
Cost	Low to Medium	Very Low
Community & Online Support	Very Large and Active (forums, libraries, extensive project examples)	Large (especially within Arduino and general STM32 communities)
Integration with Other Systems	Easy due to built-in connectivity, wide support from IoT platforms and MQTT.	Good, relies on external modules for connectivity, but strong for direct industrial control.
Learning Curve for Advanced Features	Medium (especially with built-in FreeRTOS and ESP-IDF)	Medium to Steep (especially with STM32CubeIDE and direct HAL/LL library usage)

II.2.2 Sensors

II.2.2.1.Distance Sensors

1.Ultrasonic Sensors

An ultrasonic sensor is an electronic device that measures the distance of a target object by emitting ultrasonic sound waves, and converts the reflected sound into an electrical signal. Ultrasonic waves travel faster than the speed of audible sound (i.e. the sound that humans can hear). Ultrasonic sensors have two main components: the transmitter (which emits the sound using piezoelectric crystals) and the receiver (which encounters the sound after it has travelled to and from the target) [18].

The basic principle of work:

- Using IO trigger for at least 10us high level signal.
- The Module automatically sends eight 40 kHz and detect whether there is a pulse signal back.
- IF the signal back, through high level , time of high output IO duration is the time from sending ultrasonic to returning [19].

The formula for this calculation is **D = ½ T x C** (where D is the distance, T is the time, and C is the speed of sound ~ 343 meters/second) [18].

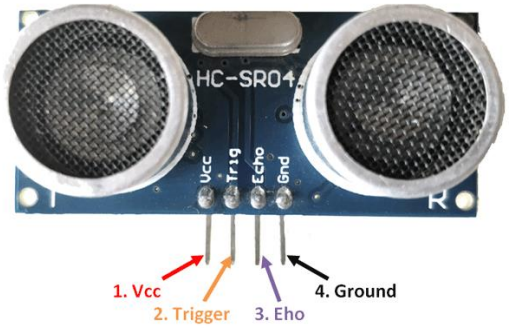


Figure II.3: HC-SR04 Ultrasonic Sensor Pinout [19].

Table II.4: HC-SR04 Technical Details [19]

Working Voltage	DC 5 V
Working Current	15Ma
Working Frequency	40Hz
Max Range	4m
Min Range	2cm
MeasuringAngle	15 degree
Trigger Input Signal	10uS TTL pulse
Echo Output Signal	Input TTL lever signal and the range in proportion

2.US-100

An ultrasonic distance sensor similar to the HC-SR04, but with higher accuracy. It supports both UART and PWM interfaces and includes a built-in temperature sensor to compensate for environmental temperature variations in distance measurements.

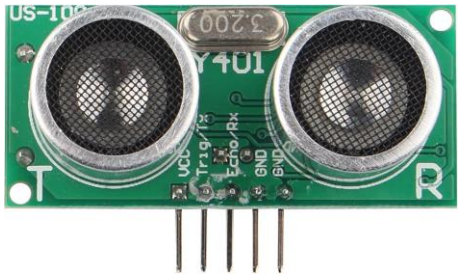


Figure II.4: US-100 Ultrasonic Sensor Module [20].

3.JSN-SR04T

A waterproof ultrasonic sensor designed for outdoor applications. It operates on the same principle as the HC-SR04 and features temperature measurement capability to adjust distance readings based on ambient conditions.

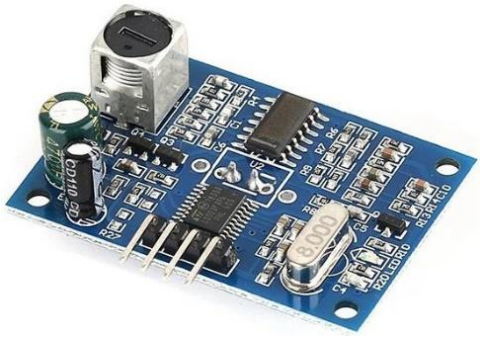


Figure II.5: JSN-SR04T Ultrasonic Sensor Module [21].

2.LiDAR - Light Detection and Ranging Sensors

- **TF-Luna**

The TF-Luna is a single-point ranging LiDAR sensor based on the Time-of-Flight (ToF) principle. It features a unique optical and electronic design and utilizes an 850nm infrared light source to deliver stable, accurate, and highly sensitive distance measurements.

Equipped with built-in adaptive algorithms, the TF-Luna is capable of performing reliably across a variety of environments and target types. It also supports adjustable configurations and parameters, allowing it to be tailored to specific application needs. These features enable the TF-Luna to maintain excellent ranging performance even in complex and challenging scenarios, making it suitable for a wide range of industrial and technological applications [22].

Figure II.6: TF-Luna Sensor Pinout [23].

- **TF-mini**

TF-Mini is a compact, LiDAR sensor that also uses the Time-of-Flight (ToF) principle. It offers reliable short-range distance measurements and is widely used in applications such as robotics and obstacle avoidance due to its small size and low power consumption.

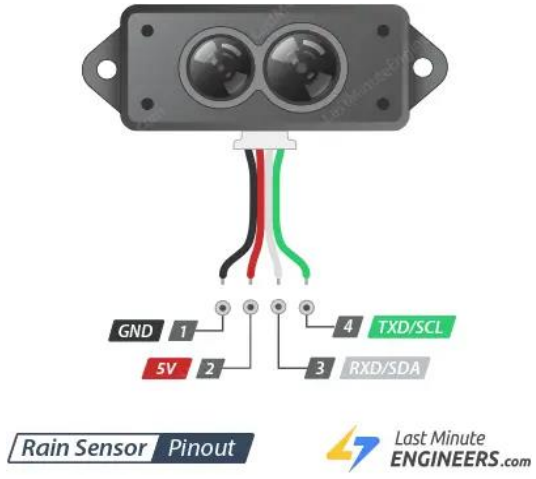


Figure II.7: TFMini-S Sensor Pinout [24].

II.2.2.2.Water Pollution Sensors

1.Turbidity sensor

The turbidity sensor evaluates water quality by measuring turbidity levels, which indicate the concentration of suspended particles in the water. It operates by analyzing light transmission and

scattering, both of which vary in response to the total suspended solids (TSS) present. As the concentration of TSS increases, the turbidity of the water correspondingly rises.

These sensors are widely employed in various applications, including monitoring water quality in rivers and streams, assessing wastewater and effluent, controlling settling pond operations, conducting sediment transport studies, and performing laboratory analyses.

The sensor offers both analog and digital output modes. In digital mode, the threshold level can be adjusted, allowing compatibility with different microcontroller units (MCUs) [25].

Figure II.8: SEN0189 Sensor Module [26].

Figure II.9: TSD-10 Sensor Module [27].

2.PH Sensor

A pH sensor is an electrochemical device used to measure the hydrogen ion concentration in water, indicating whether the solution is acidic or alkaline. It typically consists of a glass electrode that generates a voltage corresponding to the pH level. pH sensors play a vital role in water quality monitoring, as pH influences chemical solubility, biological activity, and the effectiveness of treatment processes. They are widely used in environmental research, drinking water systems, agriculture, aquaculture, and industrial water management [28].

Figure II.10: SEN0161 Sensor Module [29].

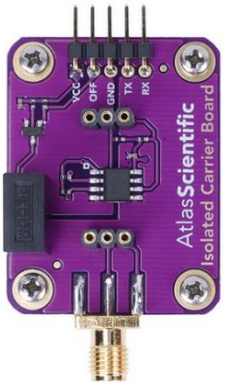


Figure II.11: Atlas Scientific pH Sensor Module[30].

3.TDS (Total Dissolved Solids) sensors

A TDS (Total Dissolved Solids) sensor measures the concentration of dissolved substances in water, such as salts, minerals, and organic matter. These sensors help determine water purity and are widely used in drinking water analysis, agriculture, aquaculture, and water treatment systems. High TDS levels can indicate contamination or poor water quality, while low levels suggest higher purity [31].

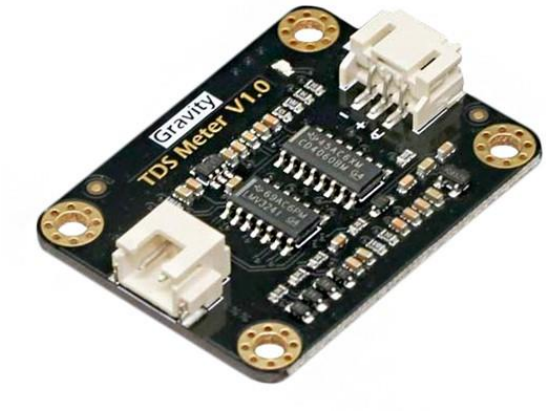


Figure II.12: DFRobot SEN0244 Sensor Module [32].

II.2.3 Actuators

1.Current sensor (ACS712)

The **ACS721 current sensing module** is a high-precision sensor designed to measure both AC and DC currents up to 20A. Built upon the reliable ACS712 chip, it offers seamless integration with microcontrollers via its analog I/O interface. This module is ideally suited for applications such as over-current protection circuits, battery charging systems, and switch-mode power supplies, providing accurate current detection and reliable monitoring [33].

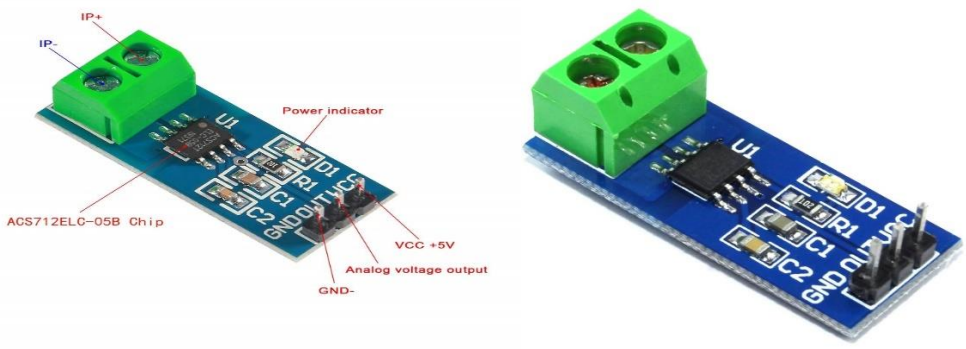


Figure II.13: ACS712 Current Sensor Module [34].

2. Relay module

A relay is an electromechanical device that functions as an electrically controlled switch, allowing microcontrollers such as Arduino to control high-voltage or high-current loads like lights, motors, or pumps. This type of relay typically operates with a **12V power supply** to activate its internal coil. It features three terminal connections: Common (COM) Normally Open (N.O.) and Normally Closed (N.C.) When a **HIGH signal** is applied to the control pin, the relay switches its internal contacts, enabling safe and efficient control of high-power circuits using low-voltage signals [35].

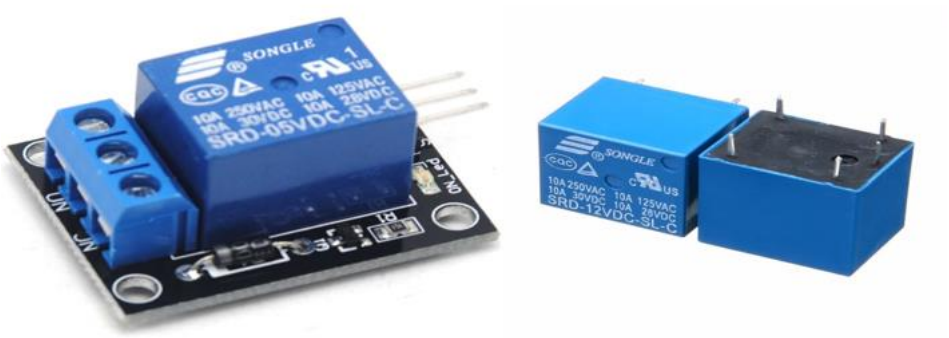


Figure II.14: Relay Module [36].

3.Keyboard 5 key

A 5-key keyboard is a simple input device consisting of five push buttons used for manual control of specific functions within electronic systems. In the "Water Control Based on IoT System" project, this keypad plays a crucial role by allowing the user to switch between different communication modes (Bluetooth, WiFi, GSM), enter the SIM card number, and modify user data easily. It provides a simple and effective local control interface, ensuring flexibility and autonomy in system operation even when the application or network connection is unavailable .

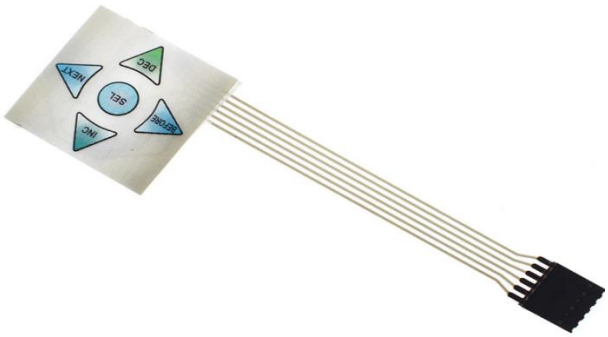


Figure II.15: Keyboard 5 key [37].

4. printed circuit board (PCB)

A printed circuit board (PCB) is a fundamental component used to assemble and connect electronic components within a unified electrical circuit, allowing electric current to flow between them. The board's base is typically made of a rigid non-conductive material, but it can also be manufactured using flexible materials or a combination of both rigid and flexible substrates, depending on the design requirements. Electronic components such as diodes, inductors, and transistors are mounted on the surface of the board and are connected through fine copper traces that act as electrical pathways. PCBs are used in a wide range of electronic devices of various types and sizes, including Internet of Things (IoT) systems [38].

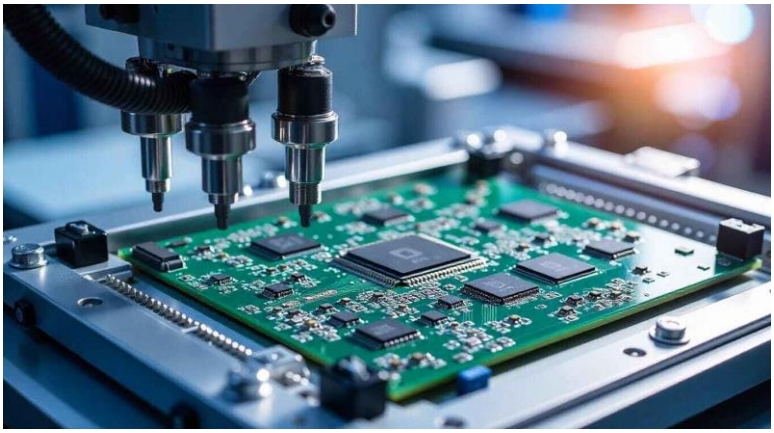


Figure II.16: printed circuit board (PCB) [39].

5.OLED I2C (1.3 inch)128×64 3-5V

A 128x64 OLED display is a low-size display used to display text and graphics in high resolution of 128 pixels wide by 64 pixels high. The display has OLED technology, which offers high quality and low power consumption compared to normal LCD displays. The display is compatible with I2C communication, and therefore it is easy to connect it to microcontrollers such as the Arduino and ESP8266. It is generally used in electronics projects to display real-time data such as sensor values or alarms. In the "Water Level Control Using an IoT System" project, the display can be used to display water level, network status, and alarms directly [40].

Figure II.17: OLED I2C 128 [41].

6.Cables (Electrical Wires)

Connectors used to connect electronic additives including Arduinos, LEDs, and resistors. They are available in a lot of colours and lengths, and jumper wires are regularly used for breadboard experiments.

Figure II.18: Cables.

7.Resistance

A digital component used to lessen the flow of electrical current in a circuit, assisting to guard touchy components together with LEDs.
It is measured in ohms (Ω) and its cost is selected based on the circuit requirements.

Figure II.19: Resistance [42].

8.LED (Light-Emitting Diode)

A digital factor that emits mild whilst current flows in the precise route.
It includes a superb terminal (anode) and a negative terminal (cathode), and is often paired with a resistor to limit the contemporary.

Figure II.20: LEDs.

9.Breadboard

A board designed for quickly connecting and trying out digital circuits without the want for soldering.

It is used to facilitate the construction and change of digital tasks at some point of layout and testing.

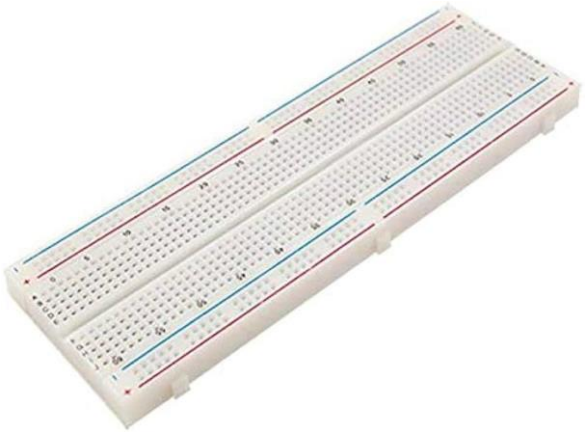


Figure II.21: Breadboard [43].

II.2.4.Communication Module

II.2.4.1.ESP32 (Bluetooth)

The ESP32 microcontroller module has built-in Bluetooth functionality, which makes it an ideal option for short-range communications and low-energy wireless communication in embedded systems. However, in this project—Water Level Control Using an IoT System—we are using Bluetooth to support direct communications between the ESP32 and a smartphone or near field device without using the Internet. With Bluetooth communications the user can access water levels, get notifications, or set up the system without the Internet. The mobile user has an accessible wireless connection from sitting next to the local device, that is otherwise not available if relying on Wi-Fi or GSM where a signal may be erratic. Bluetooth can provide an experience that is fast and energy efficient, allowing the user to monitor equipment in real-time, or exchange data faster than other methods [44].

Figure II.22: Communication Module ESP32 [45].

II.2.4.2.ESP32 (Wi-Fi)

The ESP32 microcontroller comes with Wi-Fi connectivity built in, and due to that, it can connect to wireless networks and exchange information over the internet. In the "Water Level Control Using an IoT System" project,Wi-Fi support allows the device to send water level data to mobile apps or cloud servers in real time. This facilitates remote monitoring and control of water systems from-anywhere over the internet.
The power efficiency and accurate wireless connection of the ESP32 make it perfect for round-robin IoT applications requiring stable internet connections .

Table II.05: Comparison Between Wi-Fi and Bluetooth Capabilities in ESP32

Feature	Wi-Fi (802.11 b/g/n)	Bluetooth (v4.2 BR/EDR + BLE)
Frequency	2.4 GHz	2.4 GHz
Supported Modes	Station, Soft-AP, Station + Soft-AP	BR, EDR, BLE
Max Theoretical Speed	Up to 150 Mbps	Up to 3 Mbps (BR/EDR), lower for BLE
Low Power Support	Supported	Supported (BLE)
Security	WPA/WPA2/WPA3, WEP	BLE encryption, pairing modes
Multiple Connections	Multiple clients supported	Multiple device connections (depending on role)
Protocol Stack	Built-in TCP/IP stack	Built-in Bluetooth stack
Common Use Cases	Internet access, data transfer (HTTP, MQTT)	Device-to-device communication, notifications, remote control

II.2.4.3.GSM (SIM800L / SIM900L)

The SIM800L and SIM900L are well-known GSM communication modules, connecting to cellular networks for Internet of Things (IoT) applications. In this project, titled "Water Level Control Using an IoT System," the SIM800L and SIM900L will serve as a means of transmitting data and/or sending alerts via SMS or GPRS when Wi-Fi and Bluetooth networks are unavailable. Standard SIM cards work with these modules, meaning the technology will operate within the range of mobile networks. There are some situations that require the flexibility of using mobile networks instead of Wi-Fi or Bluetooth. Due to their power efficiency and small size, the SIM800L and SIM900L are an excellent choice for battery-powered water level monitoring systems that require reliable and efficient long-range communications [46].

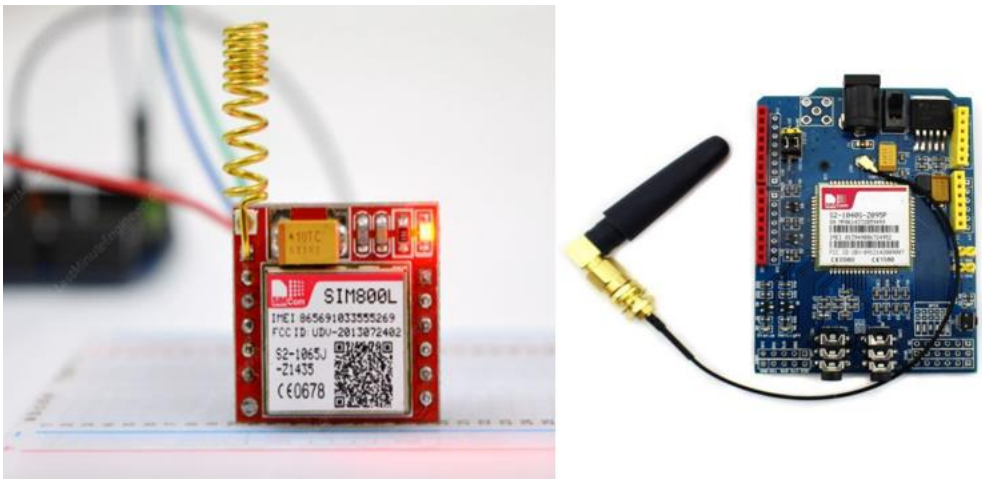


Figure II.23: GSM Module (SIM800L / SIM900L) [46].

II.3 Software

II.3.1.Interfacing ESP32 with ARDUINO

- **ESP32 Programming Methodology Using the Arduino IDE:** The ESP32 module changed into selected because the number one microcontroller for the water stage screen due to its excessive wi-fi connectivity competencies and assist for a couple of communique protocols consisting of Wi-Fi and Bluetooth.

To enforce the required programming, the following systematic steps had been observed:

1. Setting up the improvement surroundings (Arduino IDE)

The Arduino IDE software become downloaded and established. Support for the ESP32 module became then delivered with the aid of coming into its repository hyperlink in the Boards Manager settings and installing an appropriate package.

2. Connecting the ESP32 to the laptop

The ESP32 module become connected to the computer thru a USB cable, ensuring that the serial port (COM port) changed into correctly defined in the application.

3. Writing this system

C/C code changed into evolved to read the sensor statistics (water degree), method it, after which send it to the mobile application. The code additionally protected commands to manipulate the pump's on and rancid based totally on commands acquired from the software.

4. Integrating Communication Protocols

Libraries for Bluetooth and Wi-Fi had been blanketed, allowing the module to seamlessly transmit and receive statistics with the App Inventor app at the telephone.

5. Uploading the Code and Testing Performance

After verifying the code's integrity, it became uploaded to the ESP32 the use of the Arduino IDE. The gadget changed into then examined to ensure correct records exchange and quick command response.Following these steps helped obtain seamless integration between the embedded machine and the cellular app, ensuring the device's stability for the duration of actual-world operation.

Figure II.24: Interfacing ESP32 with ARDUINO [47].

II.3.2.STM Programmer Tool

Programming tools for STM32 microcontrollers are crucial components within the improvement of embedded structures. The term "STM Programmer" normally refers to a fixed of device that consists of the professional STM32CubeProgrammer software program and the ST-LINK hardware programmer. This setup allows builders to add firmware to STM32 microcontrollers produced through STMicroelectronics and gives bendy conversation with the chip through a couple of interfaces collectively with SWD, JTAG, UART, and USB, making it appropriate for a sizeable variety of industrialandacademicprograms.

These equipment guide advanced functions together with memory examine and erase operations, safety settings configuration, and external memory programming while to be had. The ST-LINK device available in the famous V2 and V3 variations acts as the bodily interface, connecting to the pc via USB and to the microcontroller through SWDIO, SWCLK, and GND pins. Using the graphical interface of STM32CubeProgrammer, users can hook up with the microcontroller, load a firmware report in .Hex or .Bin format, and add it to the internal memory the usage of the "Start Programming" characteristic, streamlining the development and checking out manner with reliability and performance.

Figure II.25: STM Programmer [48].

For more information and to download the STM32CubeProgrammer, please visit:
https://www.st.com/en/development-tools/stm32cubeprog.html

Documentation for the ST-LINK programmer is available at:
https://www.st.com/en/development-tools/st-link-v2.html

II.3.3.ThingSpeak Platform

ThingSpeak is an Internet of Things (IoT) analytics platform service that was utilized in this project not only for data collection and visualization but also as a conduit for relaying control commands to the system. The platform supports data transmission and reception via REST API and MQTT protocols and offers integration with MATLAB for advanced online data processing and analysis [49].

In this water level control system project, the ThingSpeak platform was employed in a bidirectional manner:

1.Data Acquisition Channel (Water Level Reading)

- A dedicated channel was established on ThingSpeak to receive water level data collected by the HC-SR04 sensor, which is connected to the ESP32-S microcontroller.
- The ESP32-S microcontroller periodically transmits level readings to this channel over the internet (using the ThingSpeak API via a Wi-Fi connection).
- This channel is used to store water level records and enables real-time data display through interactive visualizations such as charts and gauges. This displayed data is also accessed and presented through the mobile application developed as the primary user interface for the system.

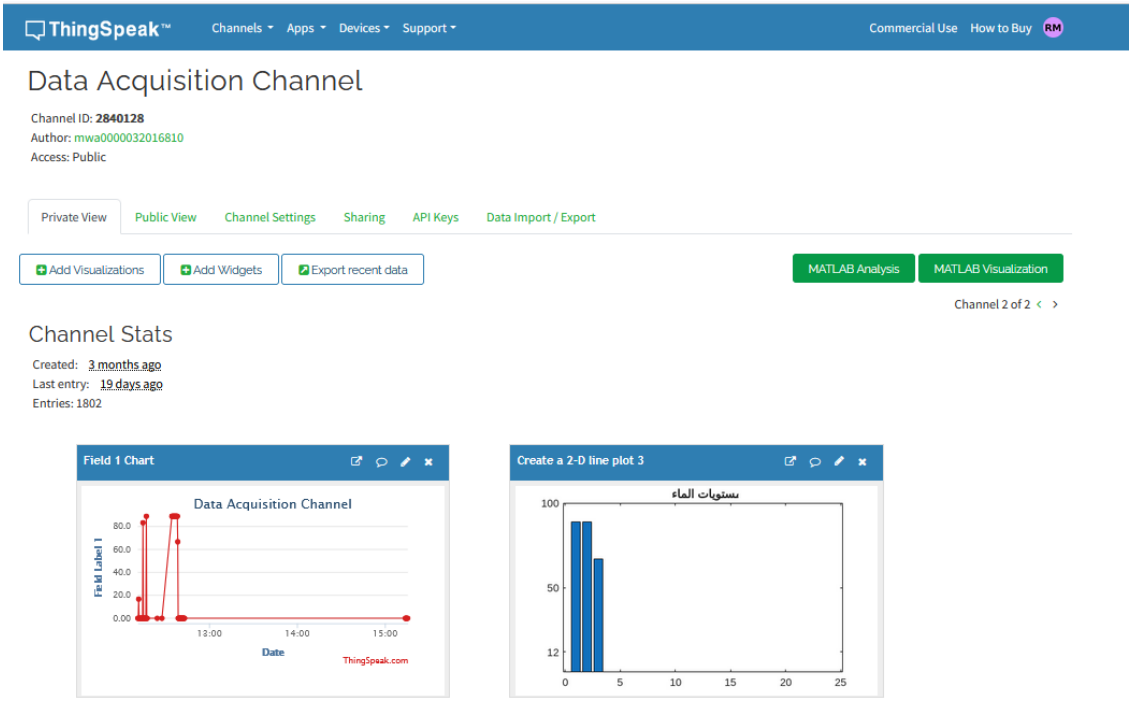


Figure II.26: Real-time water level visualization on the ThingSpeak platform [50].

2.System Control Channel

- In addition to data reading, a second channel was created on ThingSpeak specifically for sending control commands to the water level monitoring system.
- Commands, such as manually activating/deactivating the pump, are sent from the mobile application to this channel on ThingSpeak.
- The ESP32-S microcontroller periodically monitors this channel. Upon receiving a new command, it executes the corresponding action, for example, operating the relay to control the pump.
- In this manner, the control channel acts as an intermediary for transferring commands from the user interface (the mobile application) to the microcontroller at the device's location.

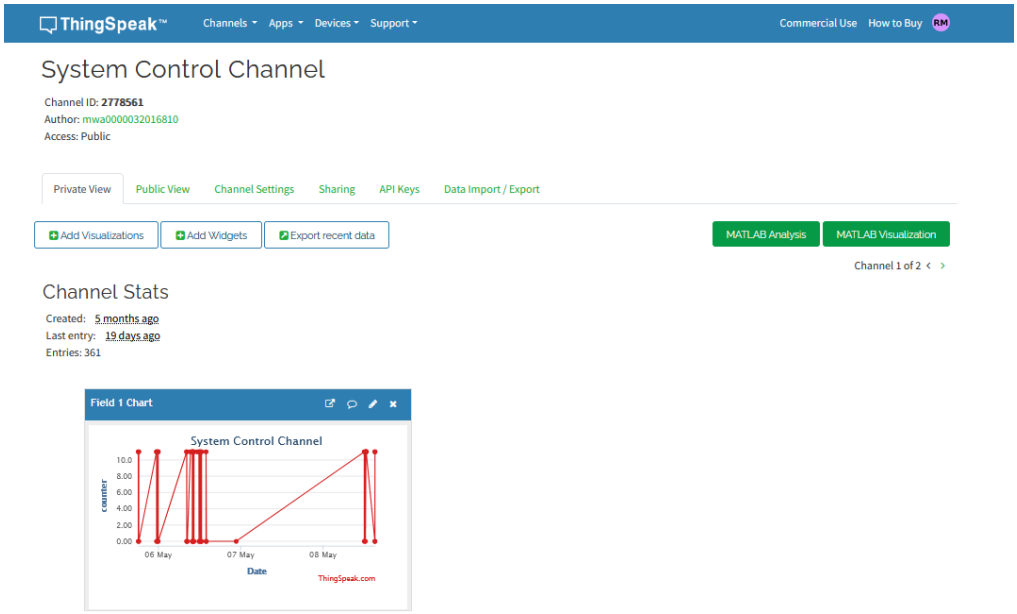


Figure II.27: ThingSpeak system control channel interface [51].

This dual utilization of the ThingSpeak platform provides significant flexibility to the system, enabling remote data monitoring and system control from any location via the mobile application.

II.3.4.Android Application

II.3.4.1.App Inventor (MIT)

App Inventor is an open-source, visual development environment created by the Massachusetts Institute of Technology (MIT) that aims to provide a means for users, primarily non-programmers, to create simple and meaningful Android mobile applications in one development environment in a straightforward and efficient manner. App Inventor is a block-based programming environment, in which coding is expressed in visual blocks that can be manipulated, stacked and arranged in a logical

way to express the logic of an application without writing a script in a traditional programming language.

Figure II.28: App Inventor (MIT) [52].

II.3.4.2.Development Environment

The App Inventor environment consists of two main parts:

- **Designer View:** Creates the user interface by adding components such as buttons, text, images, and communication modules.
- **Blocks View:** Defines the application's business logic using visual blocks, making programming easy and error-proof.

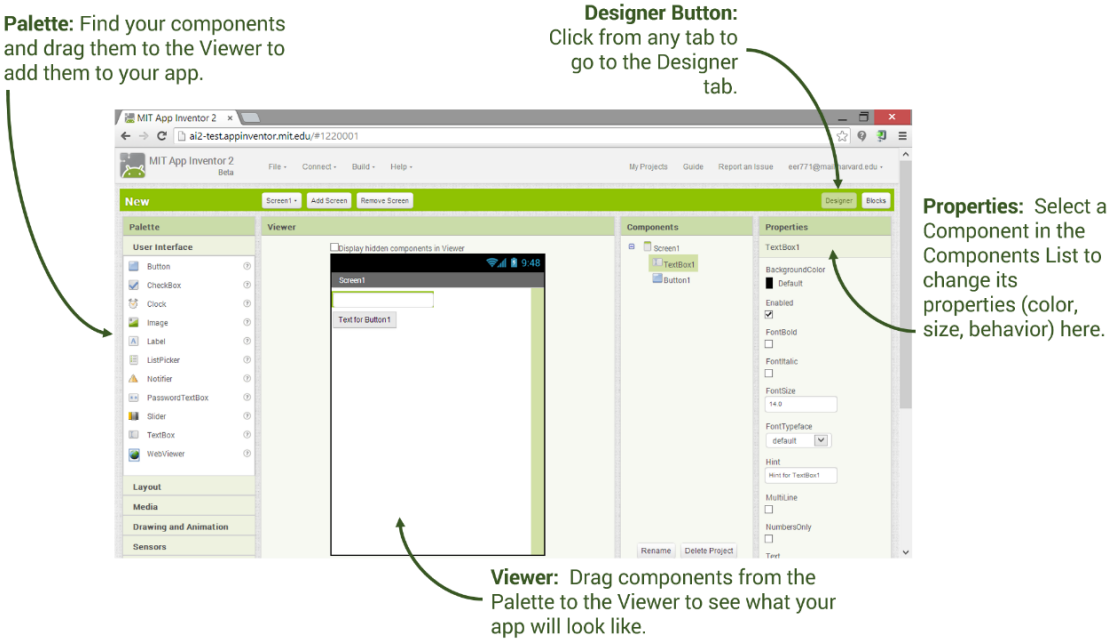


Figure II.29: Designer View [53].

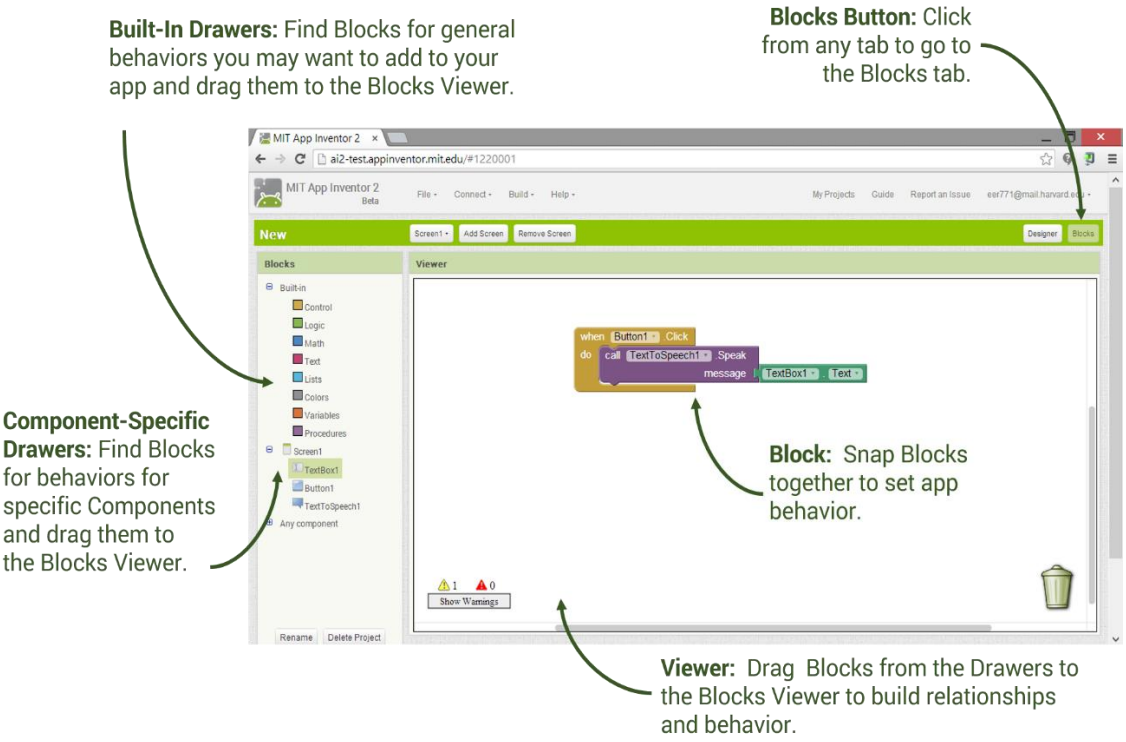


Figure II.30: Blocks View [53].

II.3.4.3.Designing the User Interface (UI)

- **Our LOGO**

The image shows the interface of an Android app developed using the App Inventer platform. The app aims to control a water pump and monitor the water level via three methods: Bluetooth, Wi-Fi and GSM.

Figure II.31: Our LOGO.

II.3.4.4.A Remote Water Level Control App Using Bluetooth, Wi-Fi, and GSM

➢ **Screen 1: Bluetooth Connection (Direct Local Control)**

This screen is used when the user desires to manage a nearby tool the usage of Bluetooth generation. After setting up a connection, the app presents the water degree facts obtained from the sensors and permits the user to show the pump on or off based on those readings. This method now does not require a web connection and is suitable for locations near the tank.

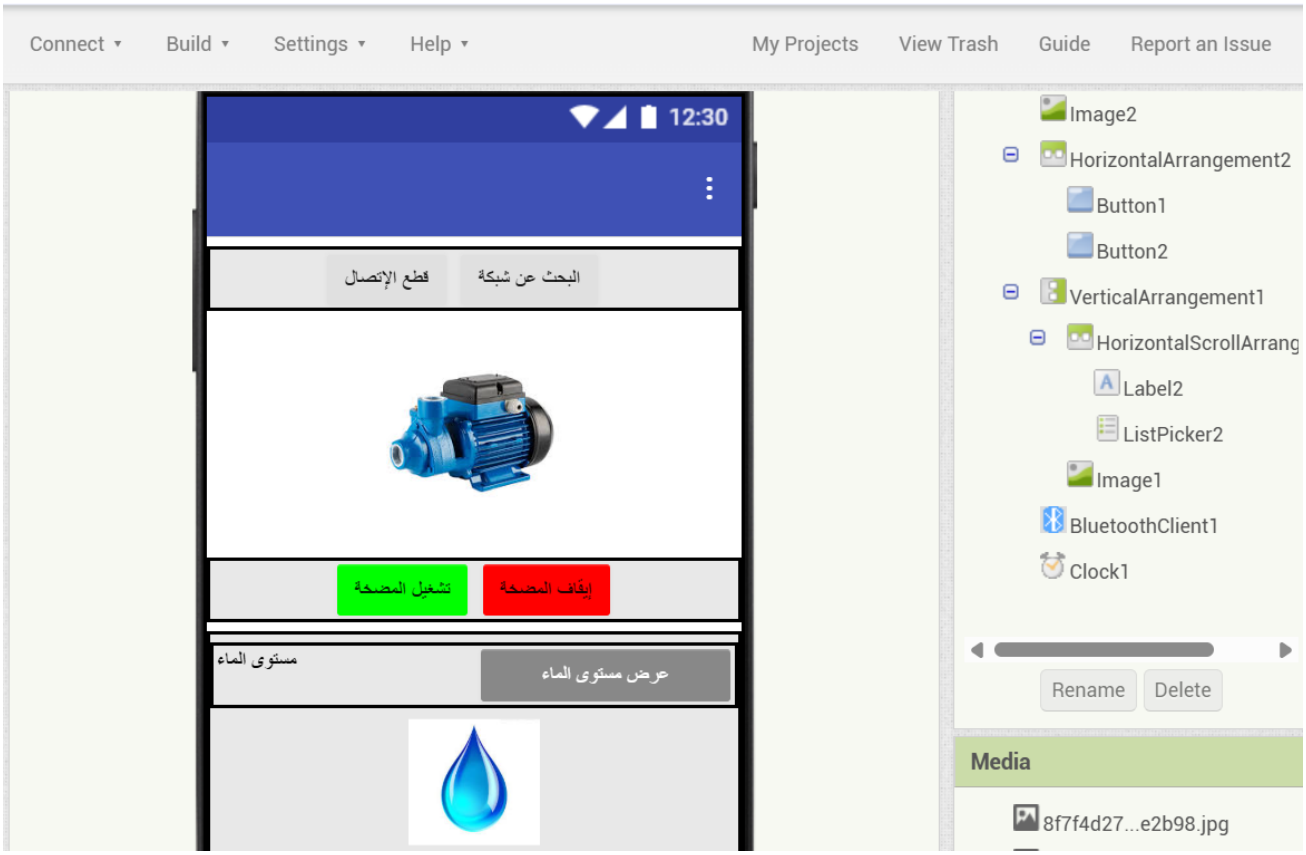


Figure II.32: Screen Bluetooth Connection [54].

➢ **Screen 1: Wi-Fi Connection (Cloud Monitoring and Control)**

This display presentations an introductory interface whilst the app is launched. It is used to set up a cloud reference to the device via Wi-Fi. The app retrieves water stage facts from the tank through the internet (as an example, using ThingSpeak) and shows it to the consumer. Based on this statistics, the user can determine to show the pump on or off remotely. This method is suitable for networked environments.

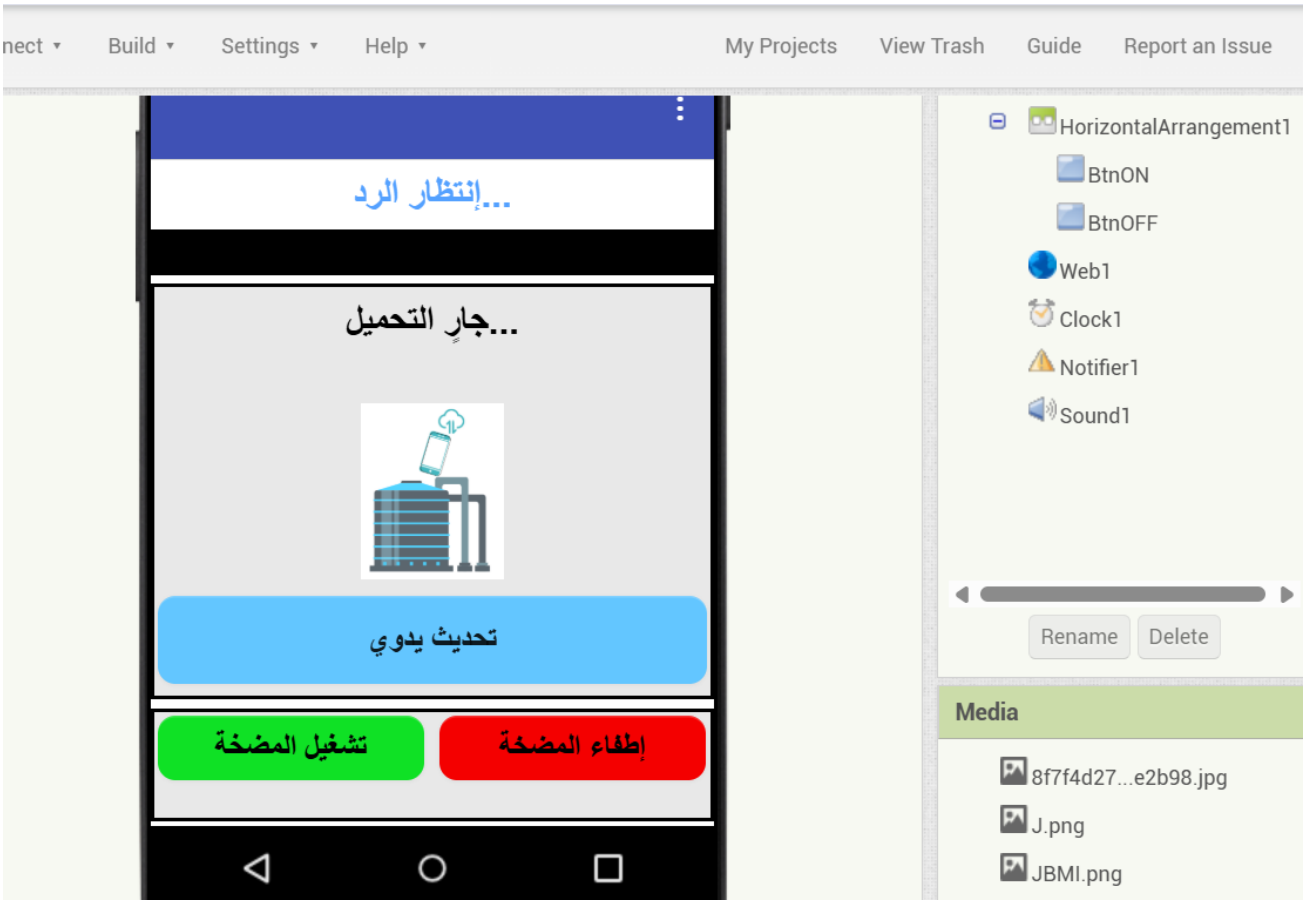


Figure II.33: Screen Wi-Fi Connection [54].

➢ **Screen 3: GSM Connection (SMS Control)**

This display permits the consumer to manipulate the device via sending and receiving textual content messages via the GSM mobile network. The app sends quick messages to a controller (including the SIM900L), soliciting for a water degree analysis or a pump on/off command. Responses are obtained as messages displayed on the screen. This method is suitable for far flung, isolated places, or those without an internet connection.

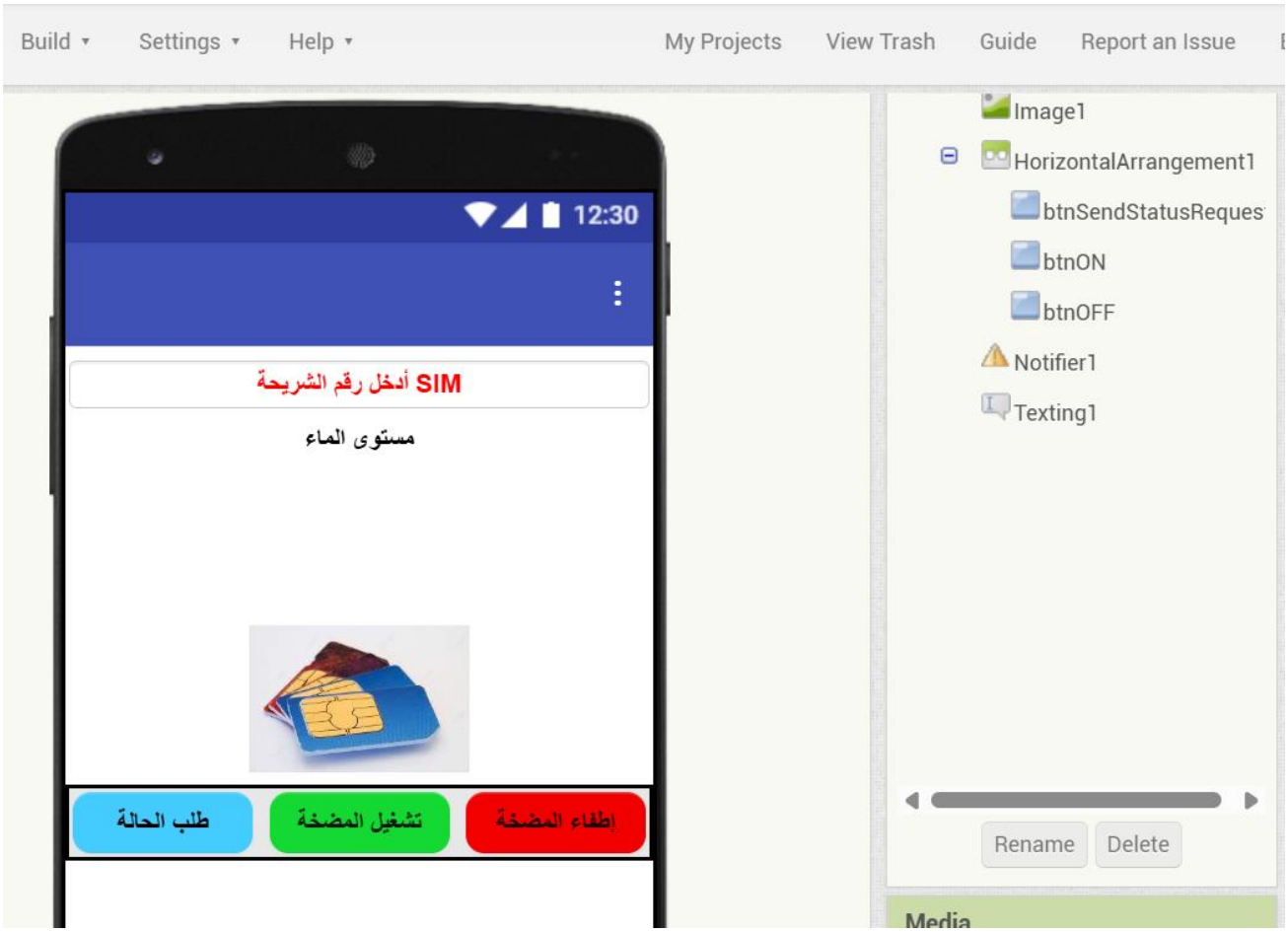


Figure II.34: Screen GSM Connection [54].

- **Wi-Fi block explanation**

Wi-Fi Control Phase (ThingSpeak): In this segment, the software is predicated on Wi-Fi and the ThingSpeak cloud platform to accumulate records and send commands. When the devoted display is opened, the variables used to shop the incoming information are initialized. The Clock1.Timer block periodically executes a GET request to the ThingSpeak API to retrieve the state-of-the-art water stage facts. The user also can manually refresh this statistics through the btnRefresh button. When the response arrives, it's miles activated with the aid of the Web1.GotText block, which first confirms the connection became a success (reaction code 2 hundred) and then approaches the retrieved JSON content to extract the water stage price. This cost is displayed within the user interface, with the status coloured based totally on severity (inexperienced for everyday, blue for low, or pink for crucial), and an audible alert is activated while wished. To manage the pump, the btnON and btnOFF buttons are used, each of which sends an HTTP request to ThingSpeak to alternate the pump's nation based totally at the sent values (1 for on, zero for off), using a stable API key.

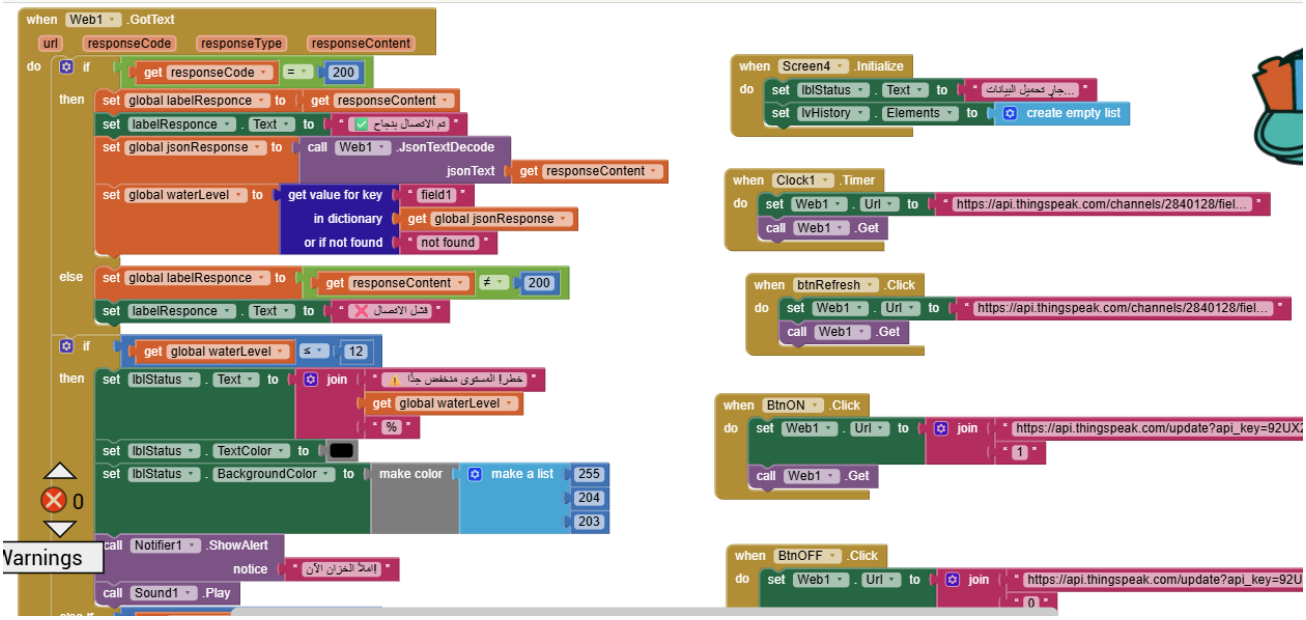


Figure II.35: Wifi block [54].

- **Bluetooth block explanation**

Bluetooth Control Phase (Local) : In this section, the software enables direct manipulate of the device via a local Bluetooth connection. The consumer selects the device to be connected thru the ListPicker element, which shows all to be had devices the usage of the Before Picking block. Direct connection is then executed after the choice via AfterPicking. After a a success connection, the relationship status is up to date in the interface to verify the pairing technique. Clock1.Timer is likewise right here to reveal the information glide from the software's controller, so any incoming data is displayed in real time in the Label2 detail. It additionally enables direct instructions to be sent to the device through two buttons: Button1 sends a price of "1" to show the pump on, and Button2 sends a value of "zero" to turn it off, in the shape of simple instructions understood by way of the controller. Button3, but, serves as the relationship area, updating the user interface to an "offline" country, including complete manage of the local consultation.

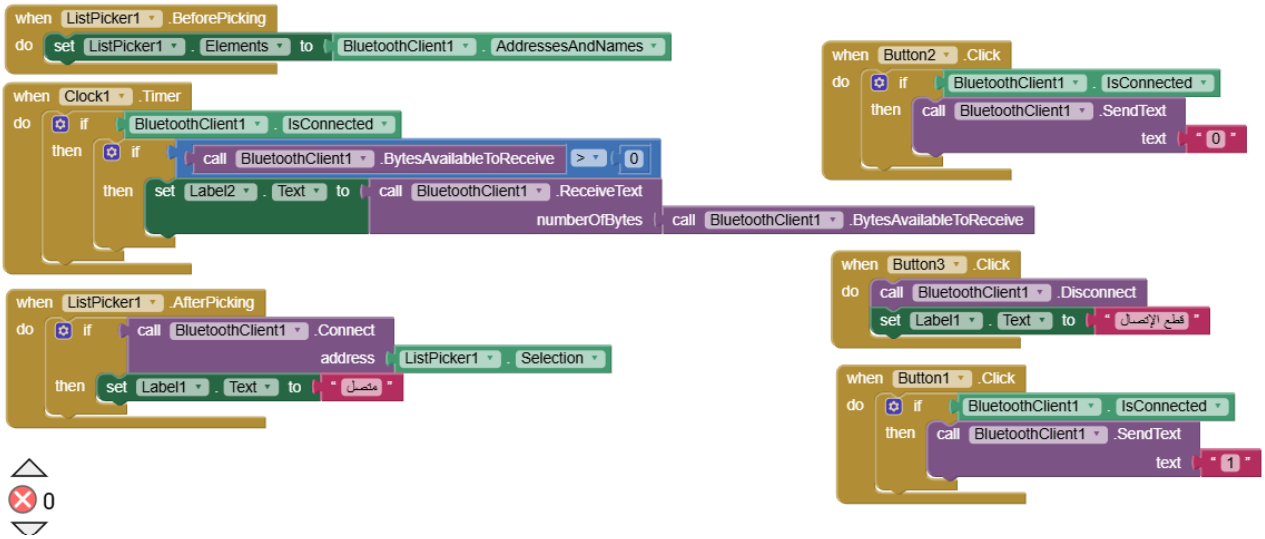


Figure II.36: Bluetooth block [54].

- **GSM block explanation**

mobile application developed using MIT App Inventor provides a user-friendly interface for controlling the water pump via GSM communication. The app features three main buttons: **ON**, **OFF**, and **Status Request**, which send SMS commands to the microcontroller (ESP32 with a SIM800L module).

When the user presses the **ON** button, the application sends an SMS with the message "ON" to the target phone number, triggering the microcontroller to activate the pump. Similarly, pressing the **OFF** button sends an "OFF" message to turn the pump off. The **Status Request** button sends an "S" message, prompting the microcontroller to respond with the current status of the system.

Additionally, the application is capable of receiving SMS responses. When an SMS message is received, the app extracts the message content and displays it in a status label, providing real-time feedback to the user. This logic enables remote control and monitoring of the pump using simple text messages, making the system accessible even without internet connectivity.

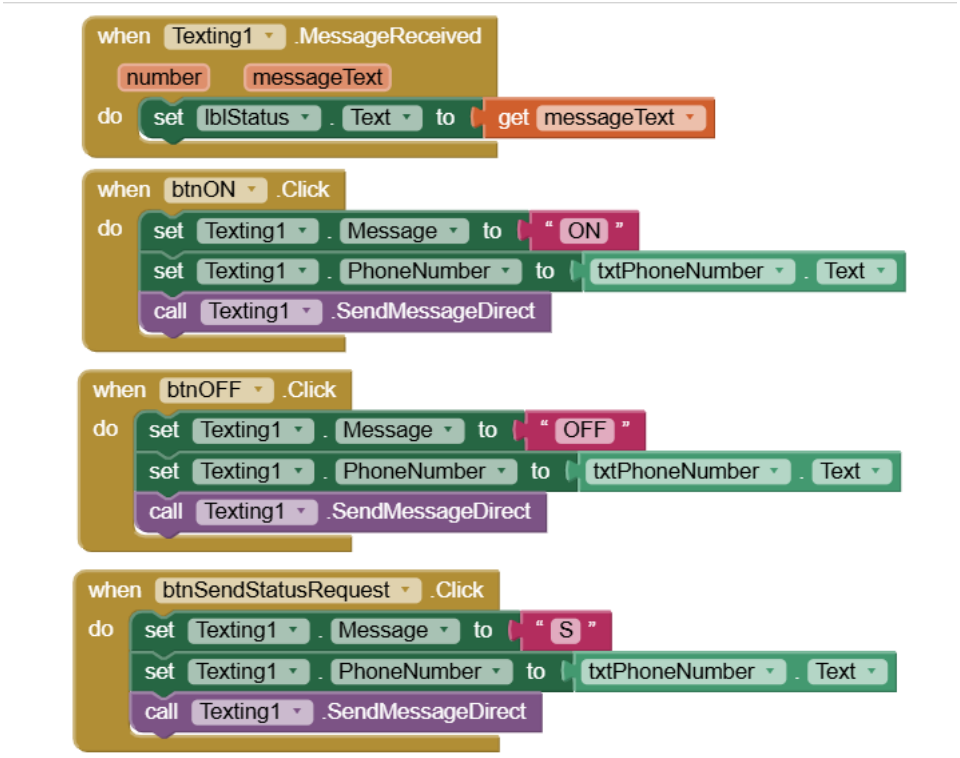


Figure II.37: GSM block [54].

II.3.5.EasyEDA

II.3.5.1. EasyEDA std software

The practical implementation of this project, which involves the design and execution of an IoT-based water level control system, necessitated an effective tool for electronic circuit and Printed Circuit Board (PCB) design. EasyEDA Std was selected for this purpose due to its nature as a comprehensive, web-based Electronic Design Automation (EDA) platform. This choice obviated the need for complex software installations and offered the flexibility of working across various operating systems. Furthermore, its intuitive interface and gentle learning curve contributed significantly to accelerating the design process.

Within this project, EasyEDA was utilized to develop the schematic diagram for the circuitry integrating the ESP-32S microcontroller with the HC-SR04 ultrasonic sensor. The ESP-32S was specifically chosen for its embedded Wi-Fi capabilities and dual-core processing power, making it well-suited for Internet of Things applications. The schematic also incorporated other essential control elements, such as relay module for pump actuation. Subsequently, the PCB was designed using EasyEDA to ensure a compact and reliable assembly of these components into a functional system.

II.3.5.2. Project Design Steps in the EasyEDA std Environment

To design this system, EasyEDA software was utilized for its integrated environment for schematic and Printed Circuit Board (PCB) design. The electronic circuit design for the project involved two primary stages using this software:

II.3.5.2.1. Schematic Design

The design process for the electronic circuit commenced with the creation of the schematic diagram using **EasyEDA**. This stage is considered the cornerstone for the subsequent Printed Circuit Board (PCB) design, aiming to define all necessary electronic components and their logical interconnections to efficiently achieve the system's functionalities.

This was executed by following these steps:

1.Component Selection and Placement: Using the search tool and the built-in component library (Component Tool) in **EasyEDA**, the schematic symbols for the essential components were identified and inserted into the design workspace. These components included:

- **ESP32-S:** microcontroller, it was chosen for its dual-core processor and built-in Wi-Fi support, making it suitable for Internet of Things applications.
- **SIM800L EVB:** communication module to provide connectivity via the GSM/GPRS network (for sending alerts or receiving remote commands if Wi-Fi is unavailable).
- **HC-SR04:** ultrasonic sensor for measuring the distance to the water surface, thereby determining its level.
- **Relay Module**: To control the switching of the water pump based on signals from the ESP32-S microcontroller.
- **LM2596:** voltage regulator DC-DC step-down converter module, used to regulate the voltage from either of the two power sources (battery or adapter) and provide a stable **5V** supply for all circuit components.
- **Two Diodes**: Employed to create a power source selection (Power OR-ing) circuit, allowing the system to be powered by either the battery or an AC adapter, while preventing reverse current flow between the sources.
- **5-Key Keypad**: To facilitate direct command input or settings adjustment on the system.
- **Three Capacitors**: Used for power supply filtering and improving circuit stability.
- **LED (Light Emitting Diode)**: To indicate the system's operational status or any other alerts.
- **Jack Connector**: For connecting the external power supply to the system.
- **Battery**: As a power source.

2.Wiring Components

After placing all components in the design workspace, the Wire Tool was used to connect the terminals of these components according to the circuit's logical design. Particular attention was given to the following critical connections:

- Connecting the HC-SR04 sensor pins (Trig and Echo) to the 5 and 18 digital Input/Output (GPIO) pins.
- Connecting the SIM800L EVB module to the ESP32-S microcontroller via serial communication (UART: TX, RX) ports.
- Connecting the 5-Key Keypad to 12,13,14,27 and 32 digital input (GPIO) pins.
- Connecting the signal input of the Relay Module to 23 digital output (GPIO) pin.
- Accurately wiring the power supply circuit:
  - Connecting the outputs of both power sources (battery and adapter), each through a diode, to the input of the LM2596 voltage regulator module.
  - Ensuring the LM2596 module's output (**5V**) directly supplies all system components.

3.Defining Properties and Adding Labels

The Properties Bar was used to define any necessary values or properties for the components. The Text Tool was also employed to add descriptive labels for major components, net labels to facilitate reading the schematic and tracing electrical paths, and any other notes essential for understanding the design.

Figure [**II.38**] illustrates the final schematic diagram of the water level control system designed in the **EasyEDA** environment, which includes all the aforementioned components and connections.

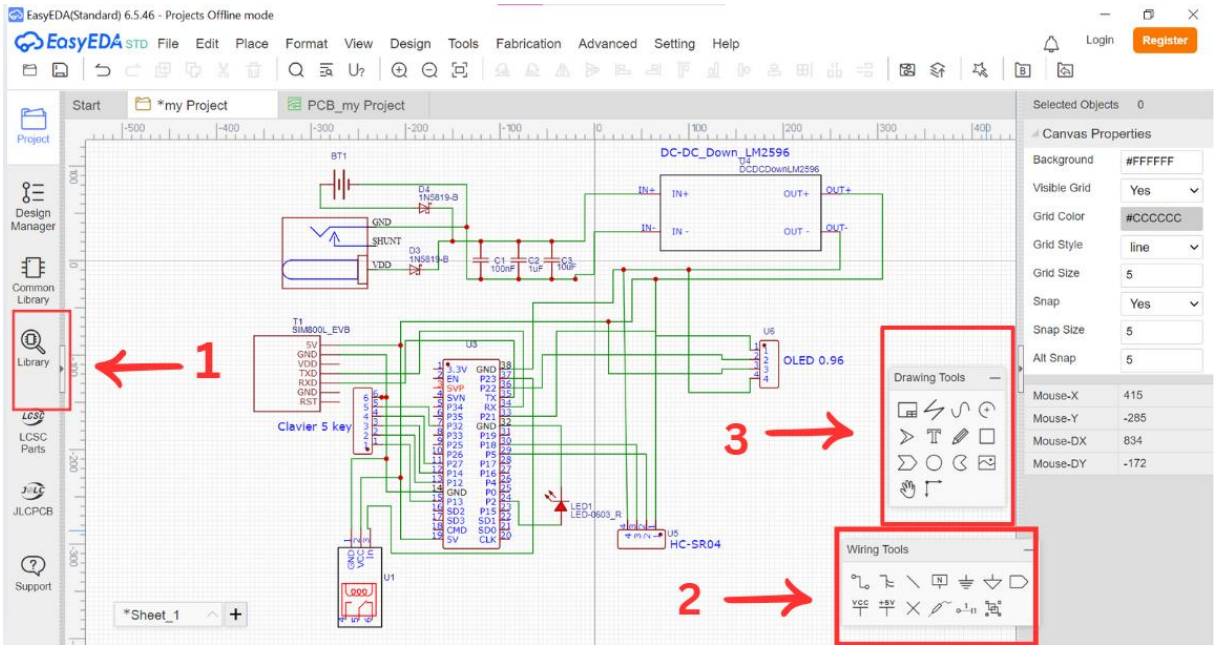


Figure II.38: System Schematic Diagram [55].

II.3.5.2.2.Printed Circuit Board (PCB) Layout Design

After completing the schematic design and verifying the logical connections between components, the second and essential phase was undertaken: Printed Circuit Board (PCB) Layout design using EasyEDA software. This phase aims to transform the theoretical schematic into a manufacturable physical design, considering the physical dimensions of components, the efficiency of electrical connections, noise reduction, and ease of assembly and maintenance.

The PCB was designed following these steps:

1.Transferring Design to PCB Editor: EasyEDA software provides a tool to directly convert the completed schematic into the PCB design environment. Upon doing so, all component footprints and the initial connections (Ratsnest) that link the component terminals needing connection are brought into the PCB design workspace, as can be observed in Figure III.39.

2.Defining Board Outline: The dimensions and shape of the PCB were determined based on the expected size of the system, its mounting location, and the number and size of components. The board outline was drawn with the appropriate shape and dimensions to accommodate all elements.

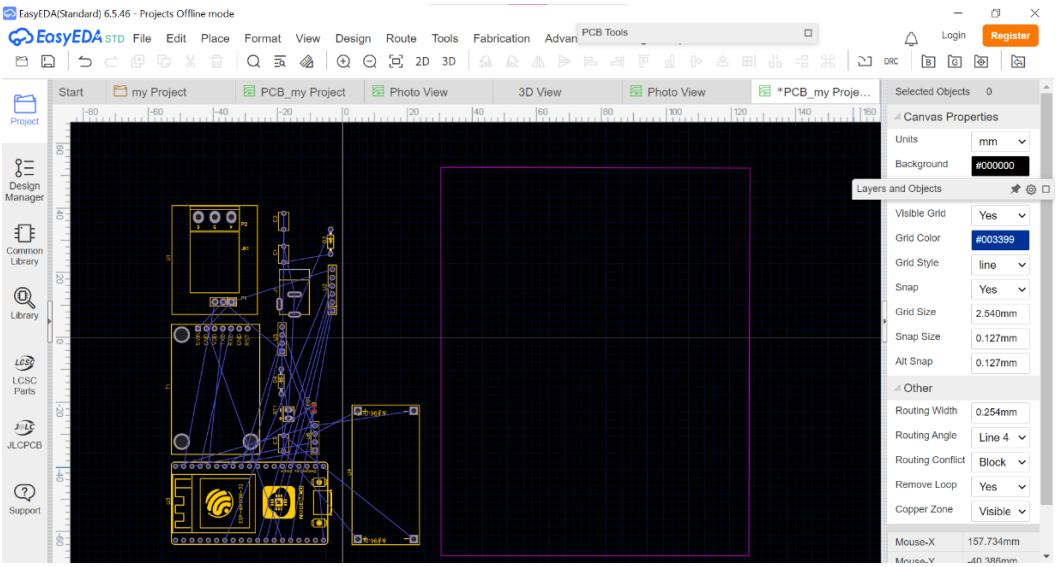


Figure II.39: Defining the Printed Circuit Board (PCB) dimensions and outline [55].

3.Component Placement

This is one of the most critical steps in PCB design. The actual component footprints were arranged on the board surface using the Placement Tool. Several factors were considered during placement, including:

- Grouping interconnected components closely to shorten copper trace lengths.
- Separating components that might cause interference, such as the SIM800L module or the LM2596 voltage regulator, from sensitive traces.
- Facilitating access to connectors like the power jack.
- Providing adequate space for larger components and securing them properly. illustrates the physical placement of components on the board.

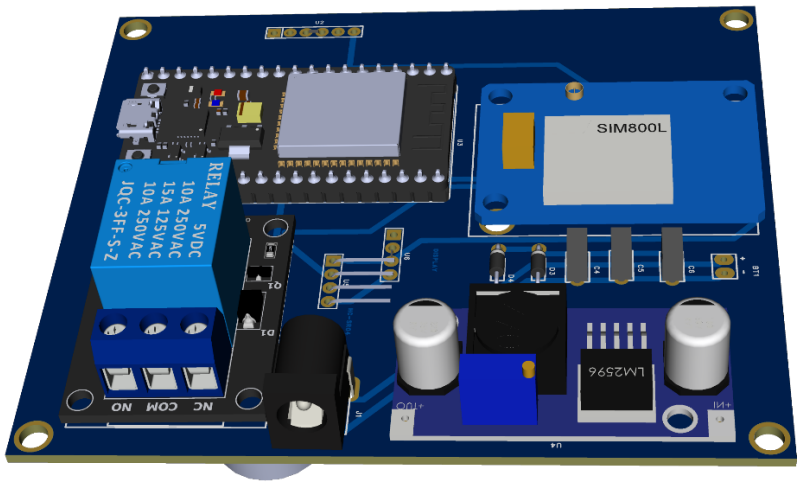


Figure II.40: 3D visualization of the prototype's main electronic board [55].

4.Routing Traces: After finalizing component placement, the Routing Tool was used to create the actual copper connections between component terminals based on the ratsnest. The board was designed as a Two-Layer PCB (Top Layer and Bottom Layer) to increase routing flexibility and reduce interference. The following were considered during routing:

- **Trace Width:** Appropriate trace widths (1mm) were selected based on the expected current, especially for main power and pump traces.
- Minimizing trace length as much as possible.
- **Avoiding sharp (90-degree) angles:** in traces and using 45-degree angles or curved traces.
- **Using Copper Pours / Polygons:** for the ground (GND) plane to improve circuit stability and reduce noise.
- **Adding Silk Screen Text and Markings:** The Silkscreen Layer was used to add component designators, values (if necessary), polarity markings, the project name, and any other information that might facilitate board assembly and testing. These markings can be clearly seen in the 3D views of the board
- **Design Rule Check (DRC):** Before sending the design for manufacturing, a Design Rule Check (DRC) was performed using the built-in tool in **EasyEDA**. This check helps detect potential design errors such as traces being too close, incomplete connections, or other violations of standard manufacturing rules. Any errors found were corrected to ensure the board's manufacturability.
- **Generating Manufacturing Files:** After ensuring the design was error-free, standard manufacturing files (Gerber files) and a drill file were generated. These are the files sent to a PCB manufacturer to produce the actual boards.

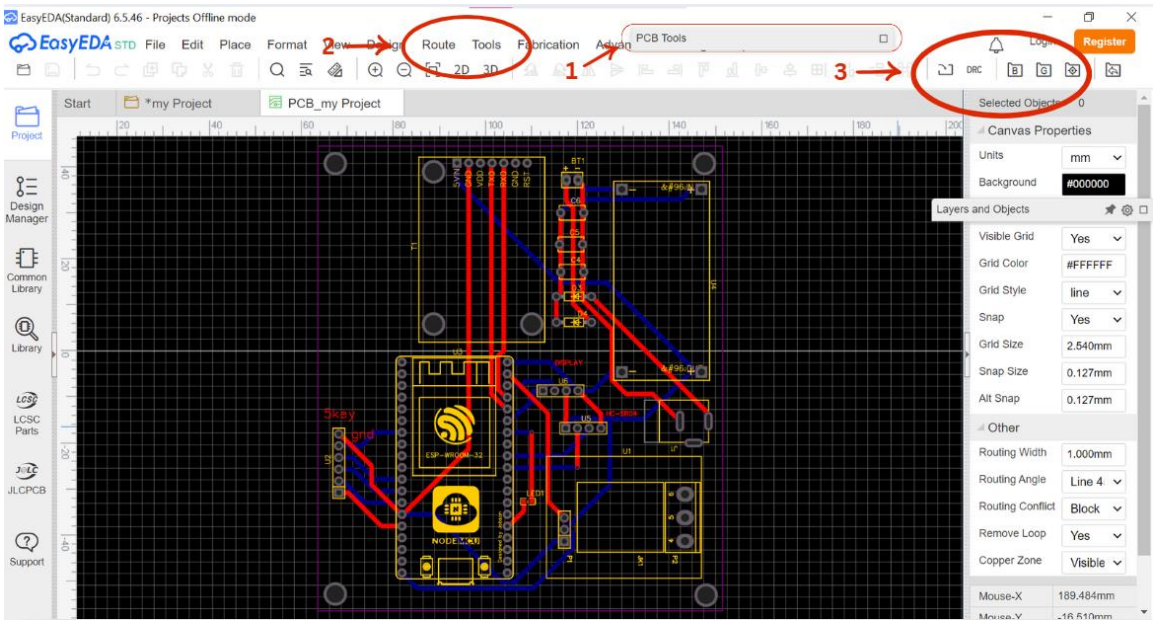


Figure II.41: Final PCB design interface, displaying copper trace layers and component footprints[55].

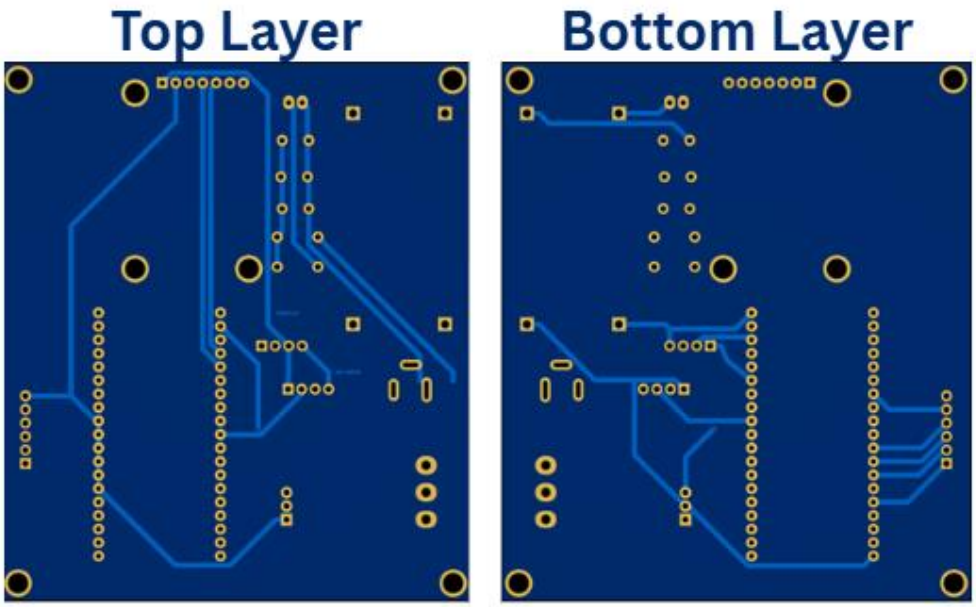


Figure II.42: Complete copper trace layout for both the Top and Bottom Layers of the PCB [55].

II.3.6. System Enclosure Design using Autodedsk Fusion 360

Following the design and testing of the Printed Circuit Board (PCB) for the water level control system, a need arose to design a suitable enclosure. This enclosure would protect the electronic components, provide an integrated final appearance for the system, and facilitate its installation in the target environment. For this purpose, the 3D mechanical design software Autodesk Fusion 360 was employed.

The enclosure was designed by following these steps and principles, with each part of the enclosure being designed separately to ensure precision, ease of modification, and manufacturability:

1.Importing PCB Model

To design an accurate and compatible enclosure, the process began by importing the 3D model of the completed PCB into the Autodesk Fusion 360 workspace. This allowed the actual dimensions of the board and the locations of prominent components to be used as a primary reference for the design, as shown in Figure **II**.43.

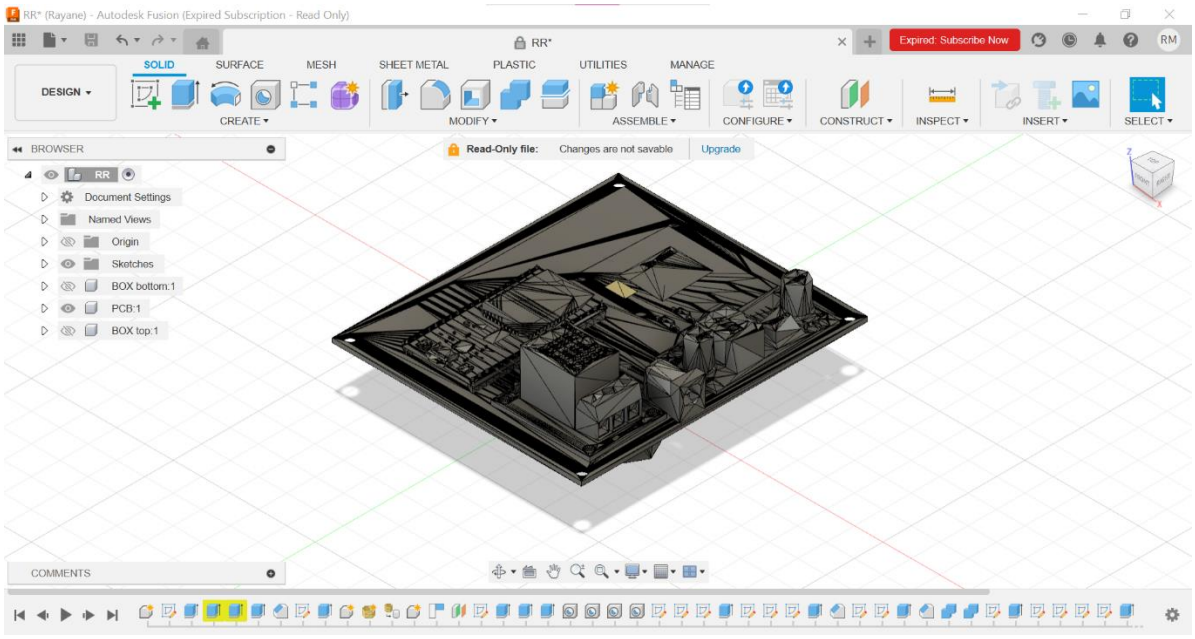


Figure II.43: 3D PCB model within the Autodesk Fusion 360 interface.

2.Designing the Main Enclosure Body

The main structure of the enclosure, consisting of a base and a lid, was designed with each part as a separate component. The design considered:

- **Internal Dimensions:** To comfortably accommodate the PCB.
- **Wall Thickness:** To achieve the required durability.
- **Mounting Bosses:** Custom-designed mounting bosses were added to the base of the enclosure to align with the mounting holes on the PCB. These bosses allow the board to be securely fastened inside the enclosure using screws. Figure **II**.44 illustrates the design of the enclosure's base with the mounting bosses.

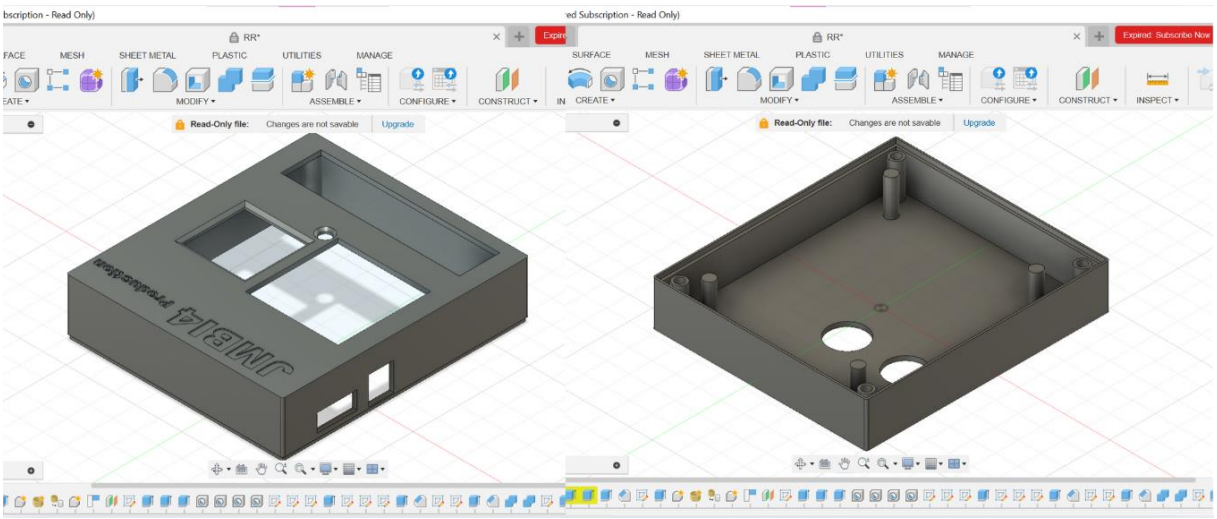


Figure II.44 : Designed enclosure parts (lid and base) prior to virtual assembly.

3.Creating Openings and Ports

Precise openings were created in the enclosure walls (in both the base and the lid) to accommodate components requiring external access or connection. These included:

- An opening for the Jack Connector (power supply).
- Openings for the pump connection terminals.
- An opening for the HC-SR04 ultrasonic sensor.
- Openings for the 5-Key Keypad buttons.
- An opening for the LED indicator. These openings can be seen in the assembled design of the enclosure, as depicted in Figure **II**.44.

4.Enclosure Closing Mechanism

A mechanism was designed to securely fasten the base and lid of the enclosure together using screws. Recesses or dedicated spots for screws were incorporated into both parts to ensure a tight seal.

4.1.Virtual Assembly and Verification

After designing all parts (the enclosure base, lid, and the PCB), they were virtually assembled in Fusion 360. This step was crucial to verify dimensional compatibility, ensure there was no interference between components, and confirm that all openings and mounting bosses were correctly positioned. Figure III.44 shows the final assembly of the system within the designed enclosure.

4.2.Preparing for 3D Printing

It was ensured that the design of each enclosure part was suitable for 3D printing technology. The final models were exported as STL files in preparation for printing.

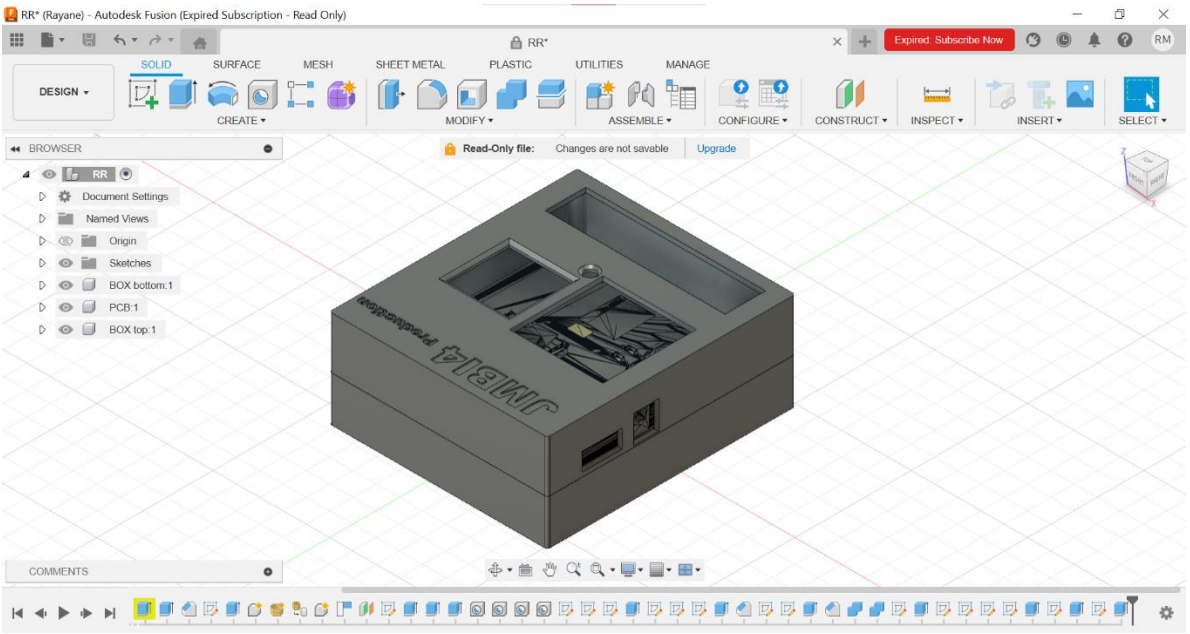


Figure II.45: Final virtual assembly of the system within its designed enclosure in Fusion 360.

II.4 Conclusion

This chapter has mentioned the vital hardware and software components that constitute the muse of the water stage control gadget. From the choice of the ESP32 microcontroller for its wi-fi capabilities to the use of the ThingSpeak platform for far off data monitoring and manipulate, every issue changed into cautiously selected and integrated to fulfill the particular functional requirements of the assignment. The systematic design and implementation of the sensors, actuators, and verbal exchange protocols make sure correct statistics acquisition and effective user interplay thru a cell software. Together, these factors exhibit how embedded systems and cloud-based platforms can be synergistically mixed to construct a dependable and scalable IoT solution.

Chapter III: System Design and Implementation

III.1 Introduction

This chapter presents a comprehensive overview of the design and implementation process of the proposed smart water level monitoring and control system. It covers both the hardware and software aspects of the project, beginning with a detailed description of the overall system architecture and operational logic. The chapter then transitions into the practical phases of circuit prototyping, printed circuit board (PCB) fabrication, microcontroller programming, and mobile application development. A significant focus is also given to the integration of multiple communication methods Bluetooth, Wi-Fi, and GSM which collectively ensure reliable and flexible system control in a variety of scenarios. In addition, this chapter discusses system assembly, testing procedures, the challenges encountered during development, and the practical solutions implemented to address them. The goal is to document how theoretical design concepts were successfully translated into a functional, real-world prototype.

III.2 System Description

III.2.1 System Structure and Component Interaction

The figure illustrates the overall system architecture and the interaction between the main components of an IoT-based water level monitoring and control system.

At the core of the system is an **ultrasonic sensor (HC-SR04)** mounted on top of the water tank, which continuously measures the distance to the water surface to determine the current water level. The measured data are transmitted to the **microcontroller**, which serves as the central processing unit of the system.

The microcontroller, implemented using a development board (e.g., ESP32), performs multiple tasks concurrently:

- It **analyzes the sensor readings** and determines whether the water level falls below or exceeds predefined thresholds.
- Based on this analysis, it **controls the water pump** by activating or deactivating the **relay module**, which functions as an electronic switch isolating the high-voltage circuit from the microcontroller's low-voltage outputs.
- In parallel, the microcontroller transmits real-time status updates to a **mobile application** via Wi-Fi or Bluetooth connectivity, enabling remote monitoring and control.
- Additionally, an **OLED display (0.96-inch)** is integrated to locally visualize the water level information and the pump status without the need for a mobile device.

The **mobile app** serves as a user interface that provides notifications, visualizations of the tank’s status, and manual control capabilities if required.

The diagram demonstrates how all system components interact in a coordinated manner:

- The **sensor** continuously supplies measurement data.
- The **microcontroller** processes and evaluates the data, updates the display, and issues commands to the relay.
- The **relay** physically switches the pump on or off according to control logic.
- The **mobile app** provides the user with both monitoring and control functionalities through wireless communication.

Overall, this integrated design achieves autonomous operation of the water pump based on real-time level monitoring while ensuring transparency and user control via both local and remote interfaces.

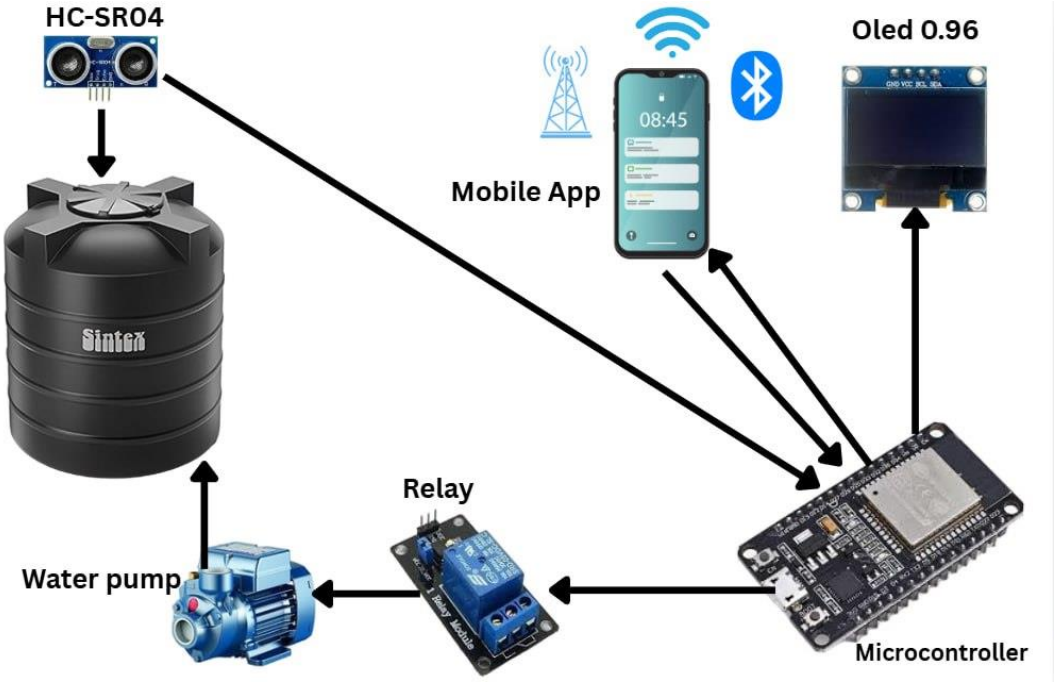


Figure III.1: System Structure and Component Interaction.

III.2.2 System Architecture

Figure **III.2** illustrates the overall architecture of the water level control system. The system is composed of two primary subsystems: the On-site System, which is responsible for the actual measurement and control, and the Cloud & Remote Interface, which enables user monitoring and interaction. The following description details the flow of data and commands among these components.

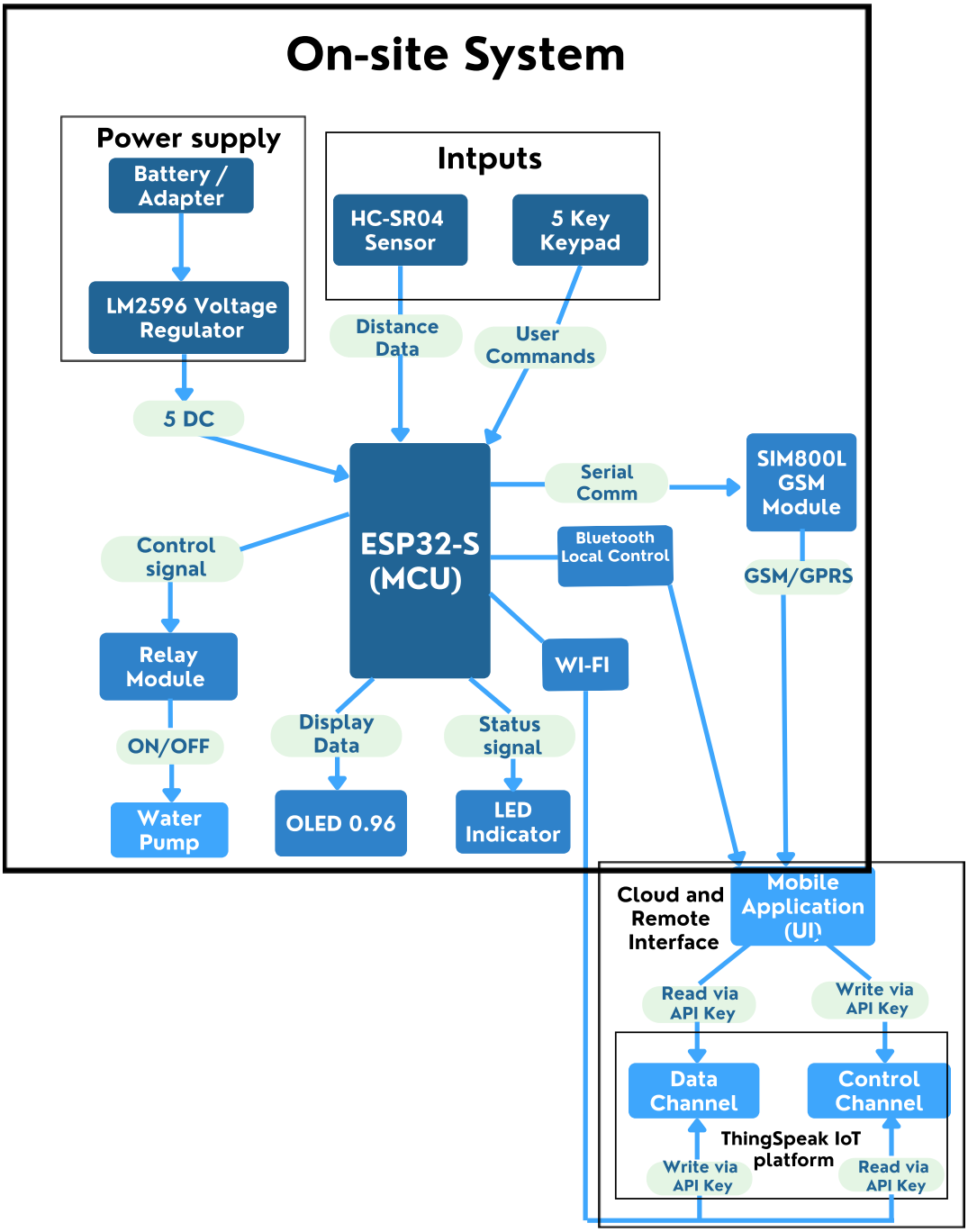


Figure III.2: System Block Diagram.

III.2.3 Operational Logic and Workflow

Following the presentation of the system's general architecture in the block diagram, this section details how data and commands flow through the various components to achieve the monitoring and control functions. The system's lifecycle begins with a mandatory initial setup stage, followed by a main operational loop that combines automatic and manual control.

1.Initial System Setup Stage

Upon first power-on, the system enters a Setup Mode. In this mode, the user is guided via the OLED display to input the essential information required for operation and connectivity. Using the 5-Key Keypad, the user enters:

- Wi-Fi credentials (SSID and password).
- The total height of the tank (in centimeters), which is used as the primary reference for dynamically calculating all control thresholds.

After this data is entered and confirmed, it is saved to the microcontroller's non-volatile memory (EEPROM), and the system then proceeds to the main operational loop.

2.Data Acquisition and Monitoring Flow

Once in operational mode, continuous monitoring begins. The HC-SR04 ultrasonic sensor measures the distance to the water's surface, and the ESP32-S microcontroller then processes this reading to calculate the actual water level. Subsequently, the microcontroller connects to the internet via Wi-Fi and periodically transmits the level value to a dedicated Data Acquisition Channel on the ThingSpeak platform. The mobile application, serving as the exclusive monitoring interface, accesses this channel to fetch the latest reading and display it to the user in numerical and graphical formats.

3.Automatic Control Logic

In the absence of manual commands from the user, the system operates in its default Automatic Mode. This logic relies on two operational thresholds that are calculated as a percentage of the tank's height:

- Low Threshold (12%): When this level is reached, the controller sends a signal to the Relay Module, which in turn automatically activates the pump.
- High Threshold (90%): When this level is reached, the controller sends a signal to the Relay Module to automatically deactivate the pump.

Additionally, a 50% threshold is used as an informational reference point only. When the water level reaches this mark, the mobile application can display a "Tank Half Full" status, but this threshold does not trigger any direct control action on the pump.

4.Manual Control Methods

The user can override the automatic mode at any time and control the pump manually through the mobile application via three distinct methods. In all cases, the controller executes the command by sending the appropriate signal to the Relay Module:

- **Remote Control (via Internet):** The user sends a command (ON/OFF) from the application to the Control Channel on ThingSpeak, which is then read and executed by the microcontroller.
- **Local Control (via Bluetooth):** For immediate response when in close proximity, the application sends commands directly to the microcontroller via a Bluetooth connection.
- **Backup Control (via SMS):** As an alternative method in case of internet failure, the user can send commands via SMS text messages to the SIM800L module, where the microcontroller will parse the message and execute the command.

4.Local Status Indication

Throughout all these operations, the OLED display and the LED indicator are used to provide basic system status to a local observer.

III.3 Electronic Circuit Design and Fabrication

At this stage of the project, the transition from theoretical conception to actual implementation of the electronic management system was achieved. The work was divided into two main phases: the first involved building a prototype on a board to demonstrate the overall performance of the components and the integrity of the connections, as well as designing a custom printed circuit board (PCB) for more efficient and usable field use. This sequence facilitated a smooth transition from the experimental phase to a high-quality, fully functional phase.

III.3.1 Breadboard Prototyping

In the initial phase of hardware implementation, the entire system was constructed and tested on a breadboard. The ESP32 microcontroller was connected to various sensors including the ultrasonic sensor (HC-SR04). The output actuator (a water pump ,ACS 712) was connected via a relay module. Jumper wires were used for all connections. The setup was powered through a USB cable from a laptop, and serial data was monitored using the Arduino IDE.

- Each component was tested individually to verify its operation. Then, the system was tested as a whole to validate logic flow between input sensing and pump actuation. This stage allowed for adjustments before committing to the final PCB design.

1.Integration of HC-SR04 with OLED and Wi-Fi for ThingSpeak-Based Distance Monitoring

The HC-SR04 sensor and the OLED screen were connected directly to the ESP32 microcontroller on the breadboard. A program was written to execute the following tasks sequentially in a continuous loop:

- Reading: Acquire the distance measurement from the HC-SR04 sensor.
- Local Display: Immediately display the measured value on the OLED screen.

- Connection and Transmission: Connect to a Wi-Fi network and then send the same value to the dedicated data channel on the ThingSpeak platform.

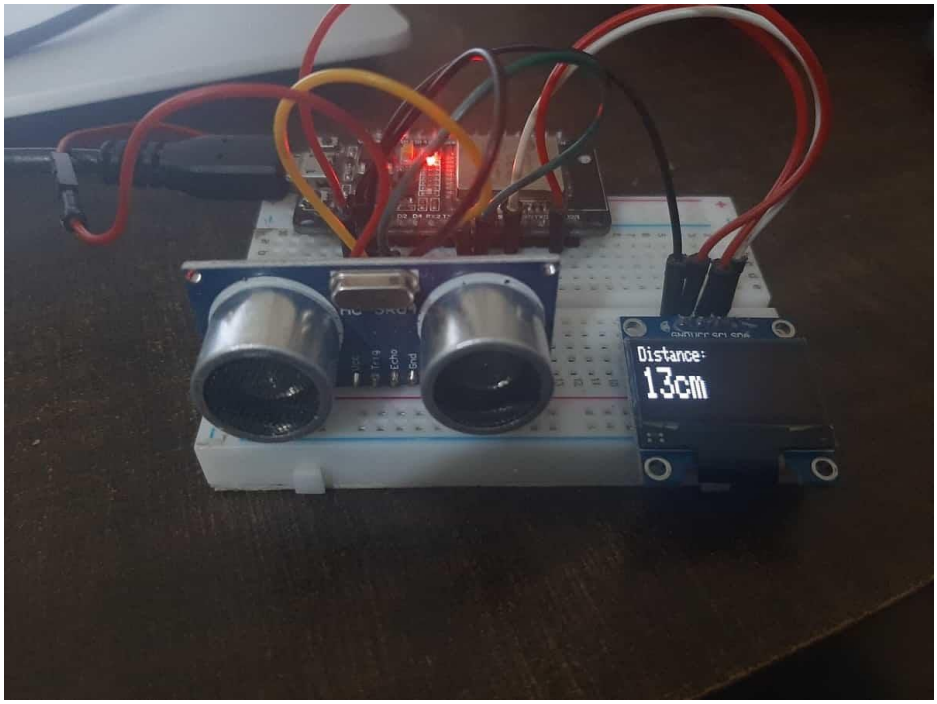


Figure III.3: HC-SR04 with OLED for distance Monitoring.

2. ACS712 Current Sensor (Diagnostic and Monitoring)

One key element in the diagnostic phase is the use of the ACS712 current sensor module. This sensor was integrated to monitor the electrical current consumed by the water pump during operation. It serves as a real-time diagnostic tool for identifying abnormalities such as overcurrent conditions that could indicate motor blockage, wear, or power anomalies.

By reading the analog output of the ACS712 through the ESP32 microcontroller, the system continuously tracks current usage. If the current exceeds predefined thresholds, the system is programmed to shut down the pump and alert the user via the mobile application or SMS (when using GSM). This helps prevent damage to the hardware, enhances safety, and provides clear visibility into system performance. In addition, current data can be logged and sent to a cloud platform for long-term analysis and preventive maintenance planning.

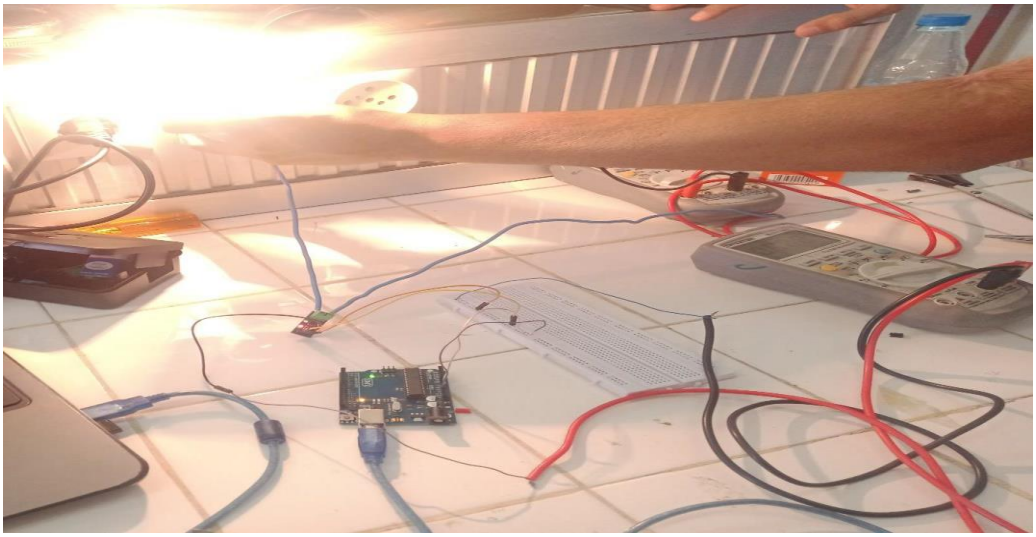


Figure III.4: Testing ACS712 Current Sensor.

- **Detection of Phase Loss in 3-Phase Systems**

In the extended scope of this project, the ACS712 current sensor is proposed not only for single-phase monitoring but also as a practical diagnostic tool for detecting faults in three-phase systems especially for the protection of motors and pumps.

When operating three-phase equipment, a condition known as 'phase loss' or 'manque de phase' can occur, in which one of the three phases becomes disconnected or fails. This results in severe mechanical stress and overheating in electric motors, leading to potential burnout or permanent damage if not detected in time.

To address this, three ACS712 sensors can be integrated one per phase (R, S, T) with each connected to a separate ADC input on the ESP32. By continuously monitoring the current flowing through each phase, the system can compare the measured values in real-time. A significant drop in current on any single phase, while the others remain within normal range, is a strong indicator of phase loss. For example, if current1, current2, and current3 represent the readings from the three phases:

```
cpp
if (abs (current1 - current2) > threshold || abs (current1 - current3) > threshold) {
  // Trigger alarm or shutdown
  Serial.println("Phase loss detected!");
}
```

The threshold value is determined based on the expected operating current and acceptable variations. Upon detection of phase imbalance, the system can immediately shut down the motor, preventing irreversible damage, and issue a real-time alert through the mobile app or SMS. This approach adds a robust layer of protection and aligns with industrial safety practices.

The collected current data can also be logged for future analysis and integrated into preventive maintenance strategies, helping to predict and avoid equipment failures before they occur.

3.Relay testing :

Several experiments were conducted to verify the system's effectiveness in controlling the water level using the ESP32 module and relay. Initially, a relay was connected to the ESP32 to control the pump's on and off based on signals received from the microcontroller. The relay represents the implementing element that responds to system commands. The relay's operation was tested by sending commands via the web interface (using the App Inventor application), verifying the activation sound (click) and the on/off of a small pump as a practical example.

A water level sensor was also connected to monitor the liquid level inside the tank, and the system was programmed to automatically turn on the pump when the water level drops below the permissible limit and turn it off when the tank is full. Various scenarios were tested by manually filling and emptying the tank, and it was observed that the system responded correctly by turning on the pump when needed, demonstrating the effectiveness of the integration of the relay, ESP32, and sensors within the IoT framework.

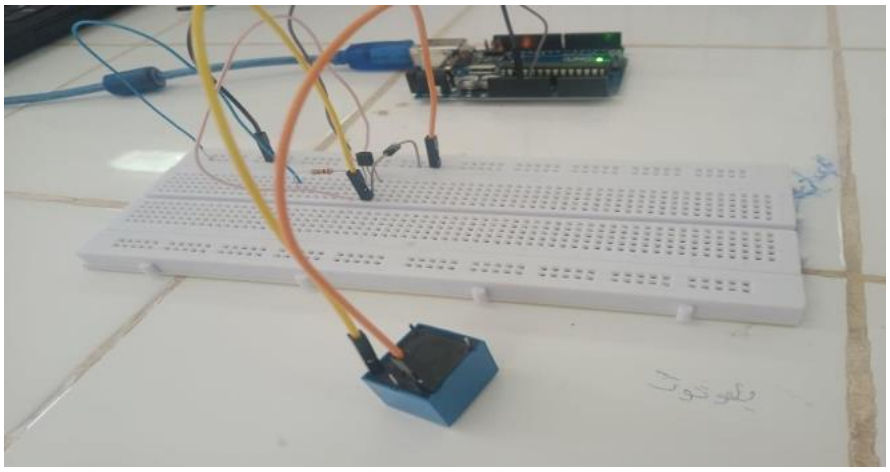


Figure III.5: Relay testing.

4.GSM Control Test

To test the backup control channel, the SIM800L module, the relay module, and an LED indicator were connected to the ESP32 microcontroller on the breadboard. Subsequently, text messages (SMS) containing simple commands such as "ON" and "OFF" were sent from a mobile phone to the phone number of the SIM card installed in the SIM800L module.

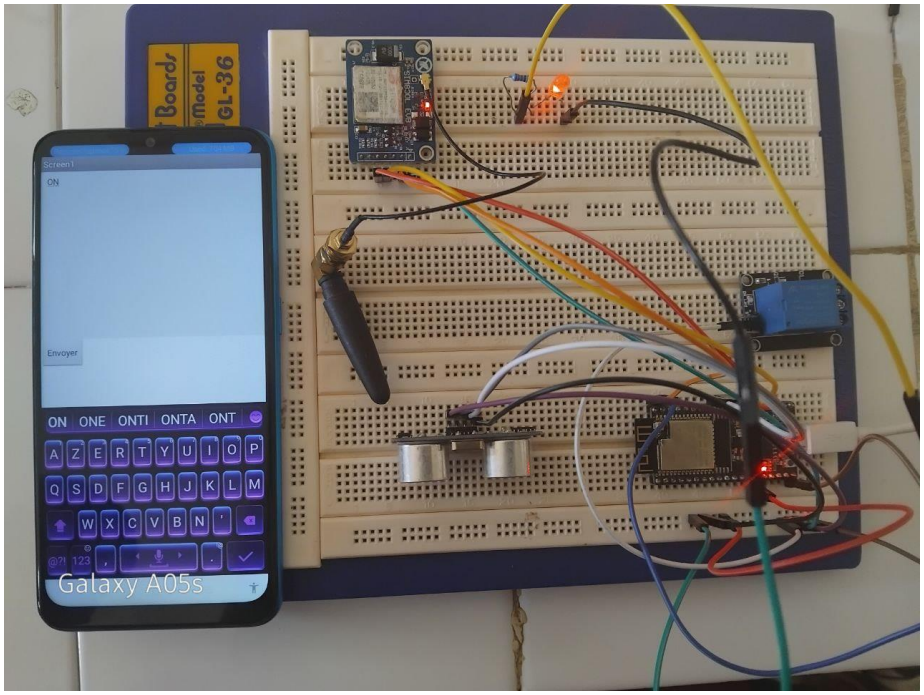


Figure III.6: Backup control test via SMS, illustrating the ON and OFF states.

4.Local Control via Bluetooth Test

To validate the direct, local control channel, the built-in Bluetooth module on the ESP32 microcontroller was activated. A wireless connection was established between a mobile phone and the microcontroller via Bluetooth.

Upon sending simple control commands (ON/OFF) from the phone, an LED indicator connected to the microcontroller responded instantly. This test successfully validated the concept of direct, local control without the need for an intermediary network like the internet.

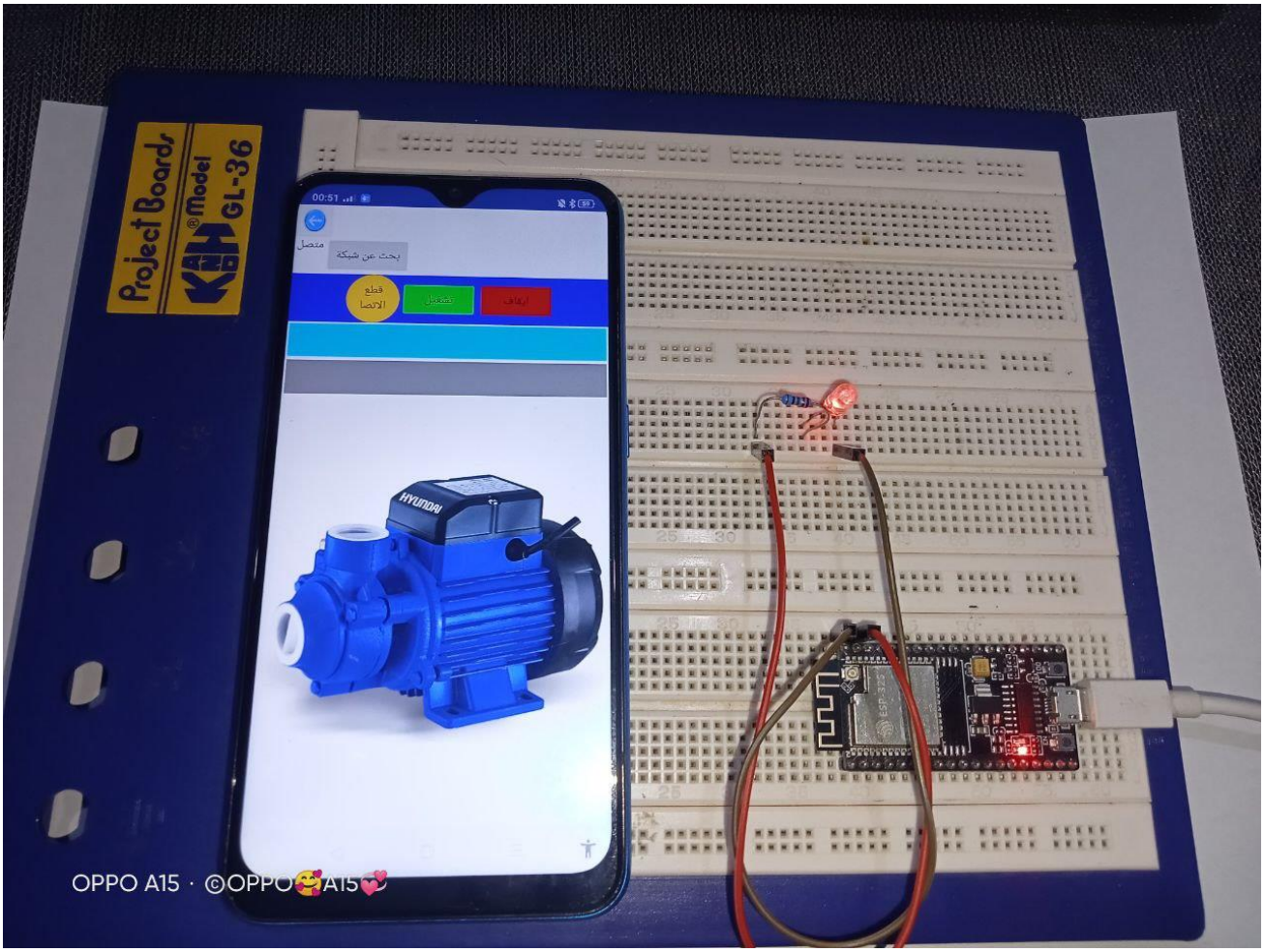


Figure III.7: Direct system control test via Bluetooth (ON and OFF states).

5.Local Network Control Test (Wi-Fi)

To verify the ESP32's ability to receive commands over a Wi-Fi network, a basic remote control test was performed. The microcontroller was programmed to act as a miniature web server on the local network.

After connecting to the server via a web browser, we were able to send direct ON and OFF commands. An LED indicator connected to the microcontroller responded instantly to these commands, which successfully validated the concept of remote control over the network as a preliminary step.

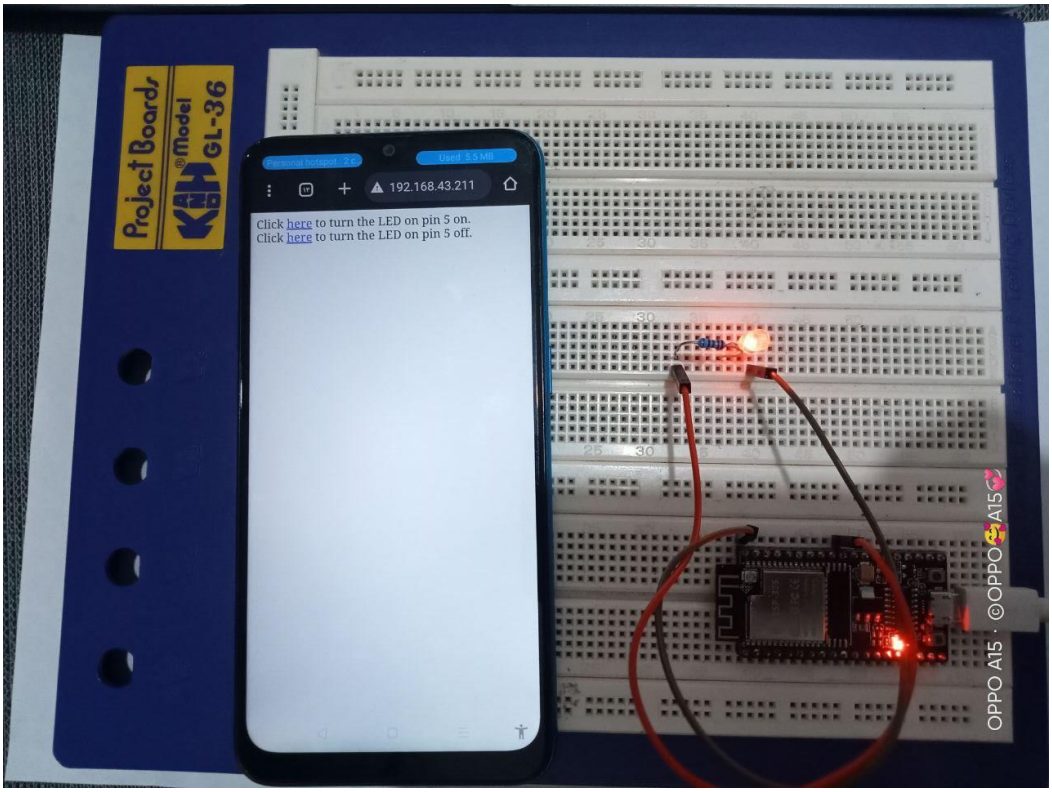


Figure III.8: Testing system control via Wi-Fi interface (ON and OFF) states.

III.3.2 PCB Layout Design

After verifying the efficiency of the experimental setup, a printed circuit board (PCB) was designed using **EasyEDA** software.

The process involved creating the **schematic diagram** for all components, followed by an effective layout that considered minimizing interference and optimizing current flow.

The placement of key components such as the **ESP32**, **relay**, **sensors**, and **voltage regulators** was carefully planned, with electrical traces routed in a structured and logical manner. The design was then previewed in both **2D and 3D views** before exporting the manufacturing files (**Gerber Files**).

- The manufacturing files were sent to a factory, and the PCB was received to specifications, allowing components to be installed and the system to be tested on the final circuit.

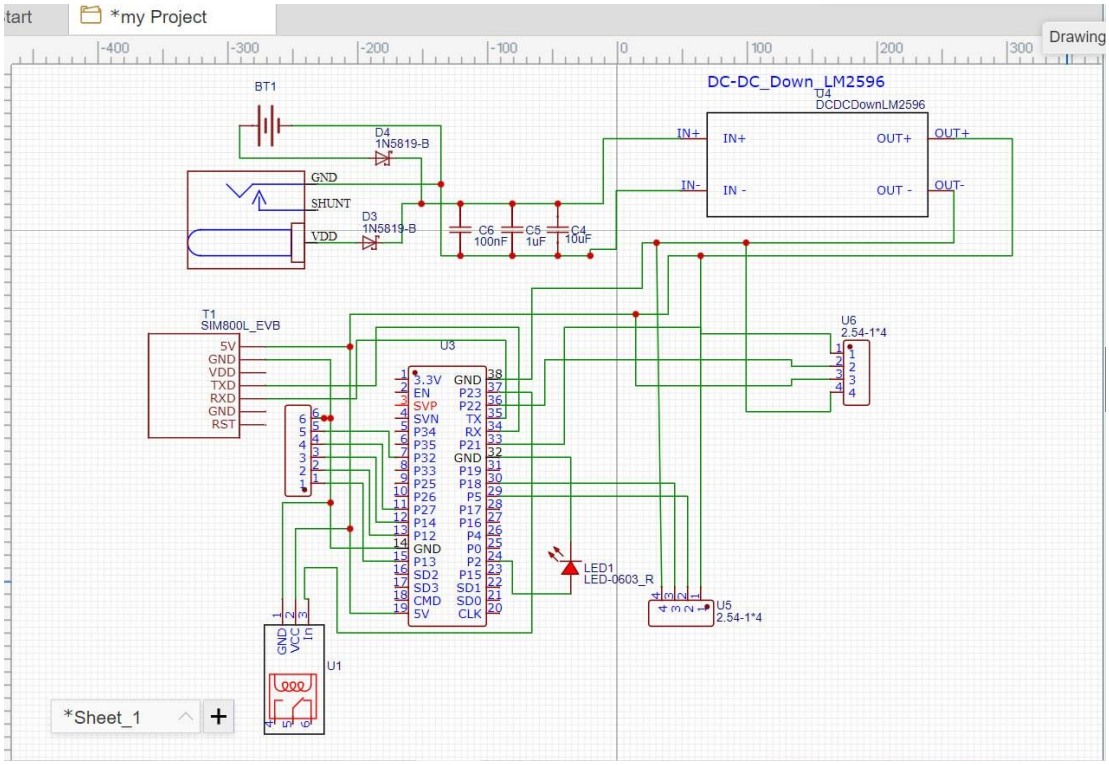


Figure III.9: the schematic diagram [55].

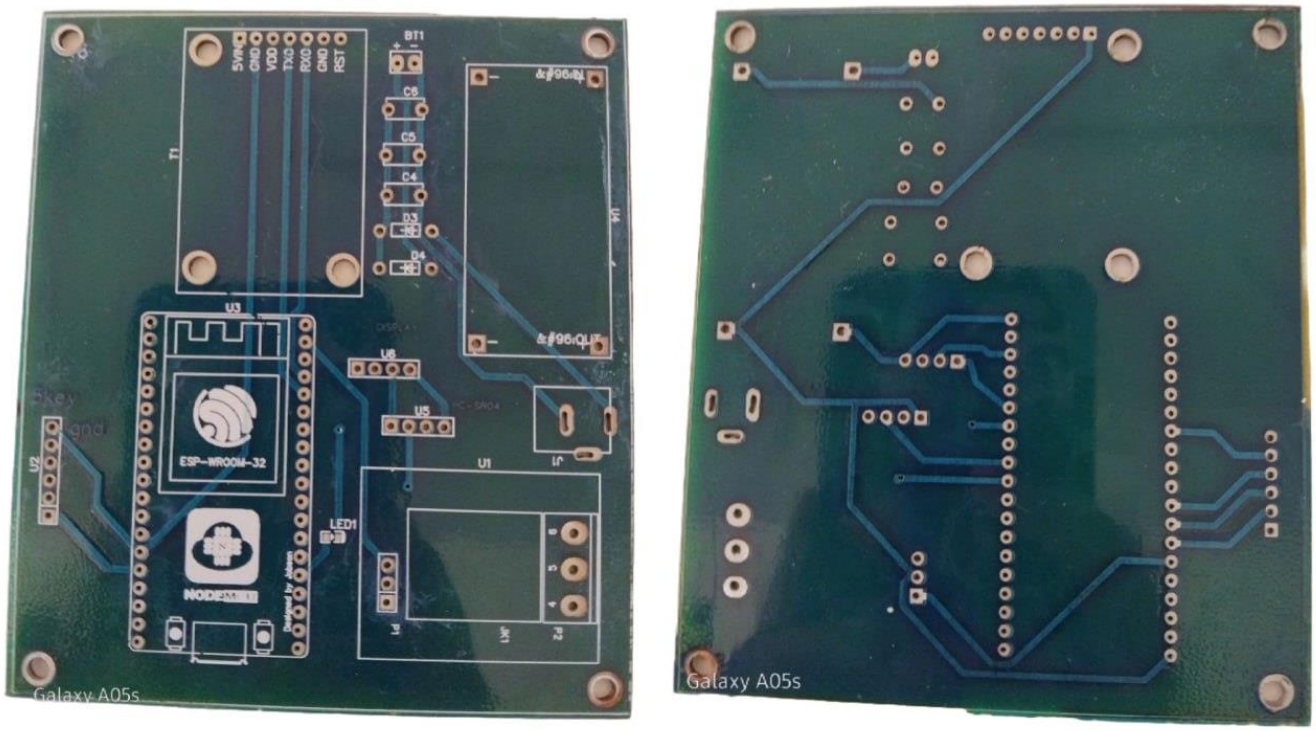


Figure III.10: the PCB board.

III.4 Software Development

III.4.1 Microcontroller Programming

1.Key Libraries Used

To achieve the system's required functionalities, several key software libraries were utilized, most notably:

- WiFi.h: For managing the microcontroller's connection to the Wi-Fi network.
- ThingSpeak.h: To facilitate the process of sending data to and receiving commands from the ThingSpeak platform.
- Adafruit_SSD1306.h: To control the OLED display.
- SoftwareSerial.h: To create a software-based serial connection for communicating with the SIM800L module.
- EEPROM.h: For saving user settings (such as network credentials and tank height) to the non-volatile memory.

2. Setup Phase (setup())

Upon system startup, the setup() function performs a series of one-time initialization tasks. It initializes the Serial Communication for debugging, sets the modes for the GPIO pins (as inputs or outputs), and starts the OLED display. It then checks the non-volatile EEPROM memory for previously saved settings. If no settings are found, the system enters the initial setup mode.

3.Main Operational Loop (loop())

Following the setup phase, the process enters a main, continuous operational loop, which is illustrated by the flowchart in Figure III.1.

In each cycle, the system reads the water level from the sensor and checks for any incoming manual commands from the communication channels (Wi-Fi, Bluetooth, SMS). Based on this, it makes a decision to activate or deactivate the pump, either in response to a manual command or based on the automatic threshold logic (12% and 90%). After executing the decision and updating the local outputs (OLED display and LED indicator), the data is sent to the ThingSpeak platform, and the microcontroller then enters deep sleep mode to conserve power before the next cycle begins.

4. Power Management

To achieve high power efficiency, Deep Sleep Mode was integrated into the operational loop. After completing each cycle (reading, checking, transmitting), the microcontroller enters a deep sleep state for a predefined duration (e.g., 15 minutes). It then wakes up automatically via an internal timer to repeat the cycle. This approach radically reduces power consumption and significantly extends the battery's operational life.

5.Code Snippets

- **HC-SR04 sensor**

Function Description

After obtaining the raw distance reading from the HC-SR04 sensor, it is processed and converted into a percentage that represents the tank's fullness. The following snippet shows the core code responsible for this important calculation.

```
// Get the raw distance reading from the sensor function.
float distance = getDistance();

// Calculate the fullness percentage based on tank dimensions.
// (maxDistance and minDistance are predefined constants)
float level_percentage = ((maxDistance - distance) / (maxDistance - minDistance)) * 100;

// Constrain the value to ensure it always stays between 0% and 100%.
level_percentage = constrain(level_percentage, 0, 100);
```

- **WIFI and thingspeak integration**

Function Description

To ensure a robust data transmission process, a function was written that first verifies the internet connection status. If the connection is lost, the function automatically attempts to reconnect to the network before sending data to ThingSpeak. This mechanism ensures the system's robustness and reliability.

Code Snippet: C++

```
// This function sends data and handles auto-reconnection
void sendDistanceData(float percentage) {
  // First, check if WiFi is still connected
  if (WiFi.status() == WL_CONNECTED) {
    HTTPClient http;
    // Construct the URL and send the data
    String url = "http://api.thingspeak.com/update?api_key=" + myWriteAPIKey + "&field1=" + String(percentage);
    http.begin(url);
    http.GET();
    http.end();
  } else {
    // If not connected, attempt to reconnect for the next cycle
    connectToWiFi();
  }
}
```

Receiving and Parsing SMS Commands

- **Function Description**

To illustrate the logic of the backup control via SMS, the following snippet shows the core part of the main operational loop (loop). This part continuously listens for data from the SIM800L module and, in case a new message arrives, parses it to execute the command.

```
// دالة تهيئة الوحدة بالأوامر الأساسية
void initSIM800() {
  sendAT("AT");               // اختبار الاتصال
  sendAT("ATE0");             // إيقاف صدى الأوامر
  sendAT("AT+CMGF=1");        // تفعيل وضع الرسائل النصية
  sendAT("AT+CNMI=2,2,0,0,0");// استقبال الرسائل مباشرة إلى السيريال
}

// دالة مساعدة لإرسال أمر AT وقراءة الرد
void sendAT(String command) {
  sim800.println(command);
  delay(500);
  while (sim800.available()) {
    Serial.write(sim800.read());
  }
}
```

III.4.2 Mobile Application

The mobile application was developed to enable the user to monitor the tank system and control the water pump remotely, using the communication method that best fits their needs and environment. To ensure maximum flexibility and efficiency, the application supports three different communication technologies: Bluetooth, Wi-Fi, and GSM/SMS, allowing the user to choose the most suitable option depending on network availability and physical distance from the system.

The application was built using MIT App Inventor, which provides a user-friendly visual environment and built-in support for communication components, including Bluetooth, Wi-Fi, and SMS. The app is designed with a simple and functional interface that allows users to easily switch between the different communication modes.

Supported Communication Modes

- Bluetooth Mode: Used when the mobile device is in close proximity to the system. This mode enables fast and stable command transmission without requiring an internet connection.
- Wi-Fi Mode: Used when both the system and the mobile device are connected to the same wireless network. Communication is established via internet protocols such as HTTP or MQTT.
- GSM/SMS Mode: Ideal for environments without Wi-Fi coverage or when the user is far from the system. Commands and responses are sent as text messages through a GSM module (e.g., SIM800L).

Main Functionalities of the Application:

- Connection Mode Selection: The user can choose the desired communication method (Bluetooth, Wi-Fi, or GSM) through a dropdown menu or buttons.
- SIM Card Number Input: When using GSM mode, the user inputs the system's SIM card number.
- Pump Control: The application can send commands such as "ON" to activate the pump or "OFF" to deactivate it.
- Status Request: The "STATUS" command is used to request the current tank or pump status.
- Response Display: Responses from the system are shown using the Notifier component or text labels.

User Interface Design

The interface is designed to adapt based on the selected communication mode. For instance, in Bluetooth mode, the app displays a list of nearby devices, while in GSM mode, a phone number input field is shown. System feedback or messages are displayed through clear notifications or text boxes.

Challenges and Solutions

Integrating multiple communication technologies in one application presented several technical challenges, particularly in managing concurrent events and handling different types of system responses. These were resolved by modularizing each communication mode and providing clear user feedback when switching modes or encountering communication errors.

Figure III.11: User Interface Design of Our App.

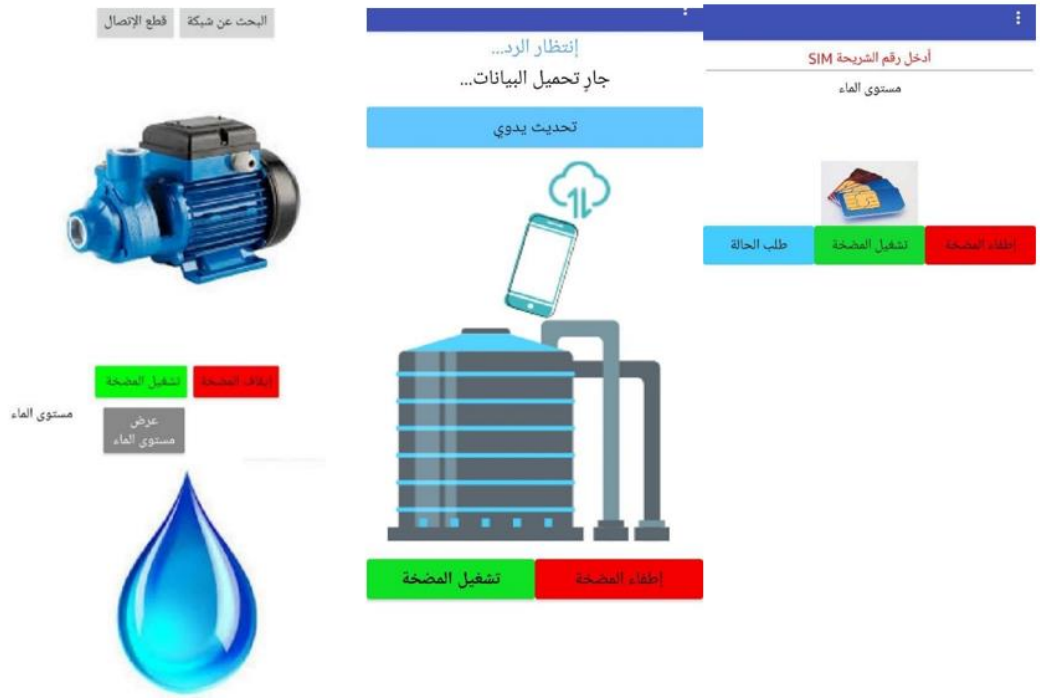


Figure III.11: User Interface Design of Our App.

- Some of the notifications when the application was tested on the system

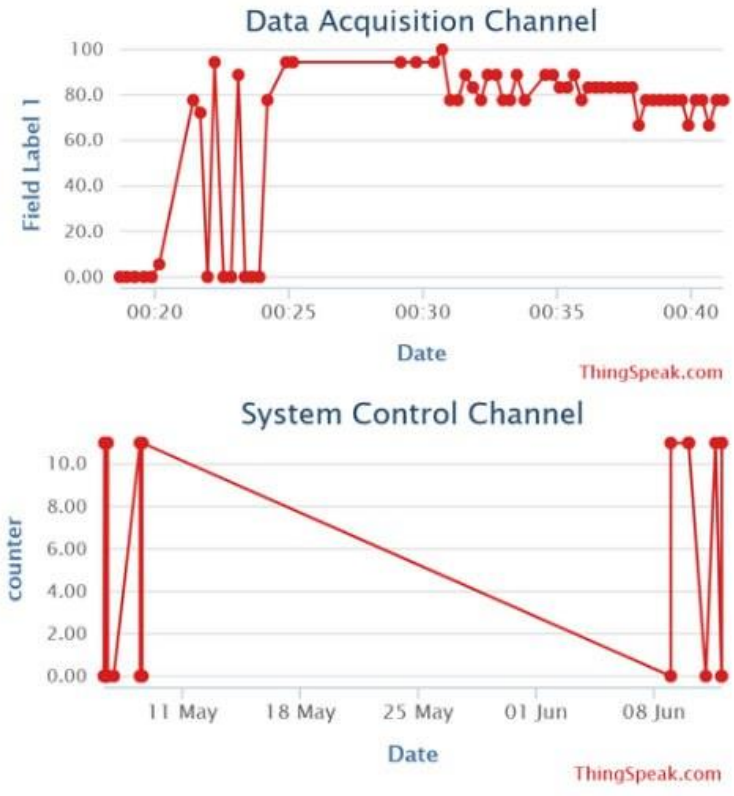


Figure III.12: Thingspeak channel interfaces [50] [51].

Figure III.13: Water Pump Alerts and Operation Commands.

III.5 Challenges and Solutions

During the practical implementation phases of the project, which ranged from initial circuit assembly to system programming and enclosure fabrication, a series of technical and practical challenges arose. Overcoming these obstacles required careful analysis and the implementation of appropriate solutions to ensure the system functioned as intended. This section outlines the most significant challenges faced and the solutions that were developed to resolve them.

III.5.1 Hardware and Software Integration Challenges

1. SIM800L Module Network Connection Failure

- **Problem Description :** An issue was encountered with the first version of the SIM800L EVB module that was utilized as shown in Figure IV. The module would continuously attempt to search for the GSM network without ever successfully registering or acquiring a signal, rendering it unusable.

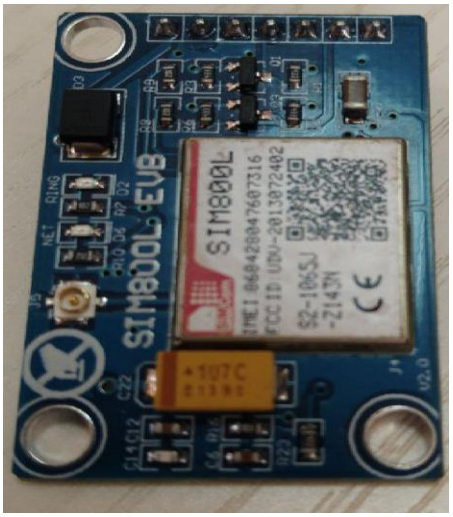


Figure III.14: The initial module with the insufficient capacitor.

- **Impact on System :** This malfunction prevented the system from utilizing its backup communication channel via SMS, significantly reducing the system's reliability in the event of an internet or Wi-Fi outage.
- **Implemented Solution :** After research and analysis, it was discovered that the problem is common in certain versions of this module and lies with the voltage-stabilizing capacitor on the development board. Its capacitance was insufficient to handle the high and sudden current spikes required by the module when searching for the network. The problem was solved by replacing the module with a different version as shown in a subsequent figure IV that includes a capacitor with a larger capacitance 1000µF. This resulted in stable module performance and its ability to reliably connect to the network.

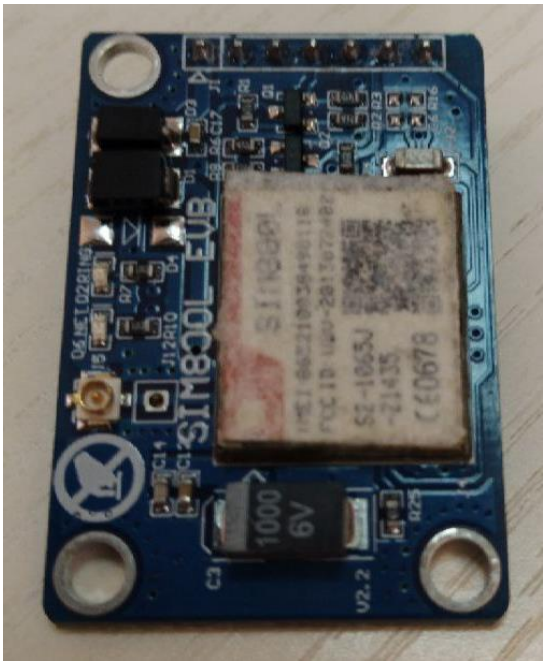


Figure III.15: The stable SIM800L module with the correct capacitor.

2.High Power Consumption and Rapid Battery Drain

- **Problem Description:** During tests relying on battery power, it was observed that the system continuously consumed a significant amount of energy. This led to the battery being depleted within a few hours, an insufficient duration for reliable operation when relying solely on battery power.
- **Impact on System:** The high power consumption limited the system's ability to operate independently on its battery for extended periods, reducing its practical utility in locations where a constant power source might be unavailable.
- **Implemented Solution:** To solve this issue, a multi-faceted power management strategy was implemented, encompassing both software and hardware optimizations:

    - **Software-Level Optimization:** The firmware was modified to utilize the Deep Sleep Mode of the ESP32 microcontroller. The controller wakes at set intervals to quickly perform its tasks (reading the sensor, transmitting data) and then returns to deep sleep.
    - **Hardware-Level Optimization:** A key design decision was made to replace the traditional LCD screen with an OLED screen. OLED displays are characterized by self-illuminating pixels, leading to lower overall power consumption.

These integrated solutions radically reduced the average power consumption and extended the battery's operational life from a few hours to several days.

3.Erroneous Readings from the Distance Sensor Due to Environmental Factors

- **Problem Description:** It was observed that the HC-SR04 ultrasonic sensor would occasionally provide suddenly erroneous readings (either extremely high or extremely low values), even when the water level was static. This could be caused by multiple echoes reverberating off the tank walls, water vapor condensation on the sensor, or the presence of foam on the water's surface.
- **Impact on System:** A single erroneous reading is sufficient to cause the system to make an incorrect decision, such as activating the pump when the tank is full, or deactivating it when the tank is empty. This could potentially lead to an overflow condition or a system shutdown.
- **Implemented Solution:** Instead of relying on a single raw reading, a software filter was implemented in the firmware. The microcontroller now takes several consecutive readings (e.g., 5 readings) at a time. It then sorts these values, discards the highest and lowest outliers, and calculates the average of the remaining values. This method, commonly known as a Median Filter or a trimmed-mean approach, ensures that anomalous readings are discarded, resulting in a stable and reliable measurement of the water level.

4.Selecting the Appropriate Microcontroller (STM32 vs ESP32)

- **Problem Description:**
  At the project's outset, the STM32 (Blue Pill) microcontroller was evaluated as a primary option. However, the first fundamental challenge was at the hardware level. This project essentially requires internet connectivity via Wi-Fi to communicate with the ThingSpeak platform and local connectivity via Bluetooth to communicate with the mobile application. The STM32 Blue Pill module lacks both of these features natively.
  To achieve these functionalities, it would have been necessary to add external wireless communication modules, a step that would have increased the circuit's complexity, size, and cost.
  In addition to the hardware challenge, another challenge emerged at the software level, as the official STM32CubeIDE development environment was found to require a steep learning curve and complex configuration.
- **Impact on Project:**
  These combined challenges (the need for additional hardware and programming difficulty) would have significantly slowed the project's development pace and increased the probability of integration issues between the different components.
- **Implemented Solution:**
  For these reasons, the optimal and strategic solution was to completely replace the STM32 platform and switch to the ESP32 platform. This decision provided a comprehensive solution to both problems:
  - **On the Hardware Side:** The ESP32 chip provided all required communication features (Wi-Fi and Bluetooth) natively, which eliminated the need for any external modules and reduced circuit complexity.
  - **On the Software Side:** The Arduino framework for the ESP32 provided a simple development environment with ready-to-use and user-friendly libraries, allowing the

focus to remain on the core system logic instead of the complexities of integration and low-level hardware configuration.

5.Selecting Appropriate Design Tools

- **Problem Description :**Initially ,KiCad software was chosen for the Printed Circuit Board (PCB) design and Blender for the mechanical enclosure design. However, after commencing work, it became apparent that both programs have complex user interfaces and steep learning curves, particularly for the rapid prototyping needs of this project.
- **Impact on System :**The complexity of these tools would have significantly slowed down the design phase. Furthermore, the process of verifying the fit between the PCB designed in KiCad and the enclosure designed in Blender would have been cumbersome, requiring multiple manual steps for exporting and importing models, which increases the likelihood of dimensional errors.
- **Implemented Solution :**To overcome this obstacle, a decision was made to switch the design toolset to a more integrated and user-friendly environment:
  - **For PCB Design :**A switch was made from KiCad to **EasyEDA** .EasyEDA is characterized by its simple and direct interface, extensive cloud-based libraries, and direct integration with manufacturing services, which greatly accelerated the board design process.
  - **For Enclosure Design :**Blender was replaced with Autodesk Fusion 360, Fusion 360 is more specialized and intuitive for mechanical design. Most importantly, it allowed for the easy importation of the 3D PCB model from EasyEDA to design the enclosure around it with precision, ensuring a perfect fit between the two parts.

This shift in tools resulted in a significantly streamlined and accelerated workflow, from concept to physical product.

6.Weak GSM Network Coverage and Connection Drops

- **Problem Description :**After resolving the hardware issue with the SIM800L module, another connectivity-related challenge emerged. It was observed that the system would sometimes fail to connect to the GSM network or the connection would drop frequently, even though the module itself was functioning correctly.
- **Impact on System :**The instability of the GSM connection meant that the backup control channel via SMS could not be relied upon, causing the system to lose one of its most important safety and reliability features in the event of an internet outage.
- **Implemented Solution :**It was determined that the problem was not with the device, but with the network coverage quality of the service provider for the SIM card being used in the testing area. To solve this, practical tests were conducted using SIM cards from the different locally available network operators. After comparing the signal strength and connection stability for each ,the SIM card belonging to the operator with the best coverage in the target location was selected ,which ensured a stable and reliable GSM connection for the system.

III.6 System Assembly and Final Prototype

III.6.1 Electronic Circuit Assembly

The first stage of assembly involved soldering all electronic components, such as the ESP32-S microcontroller, the SIM800L module, and the relay, onto the fabricated Printed Circuit Board (PCB). Figure 3.16 shows the board after the completion of this stage, ready for installation in the enclosure.

Figure III.16: The soldering of all electronic components.

III.6.2 Final Integration into the Enclosure

In the final stage, the completed PCB was mounted inside the 3D-printed enclosure base and secured with screws. After connecting external components such as the OLED display and the keypad, the lid was closed to form the final prototype. Figure 4.16 shows the external appearance of the system in its final form, ready for the testing phase.

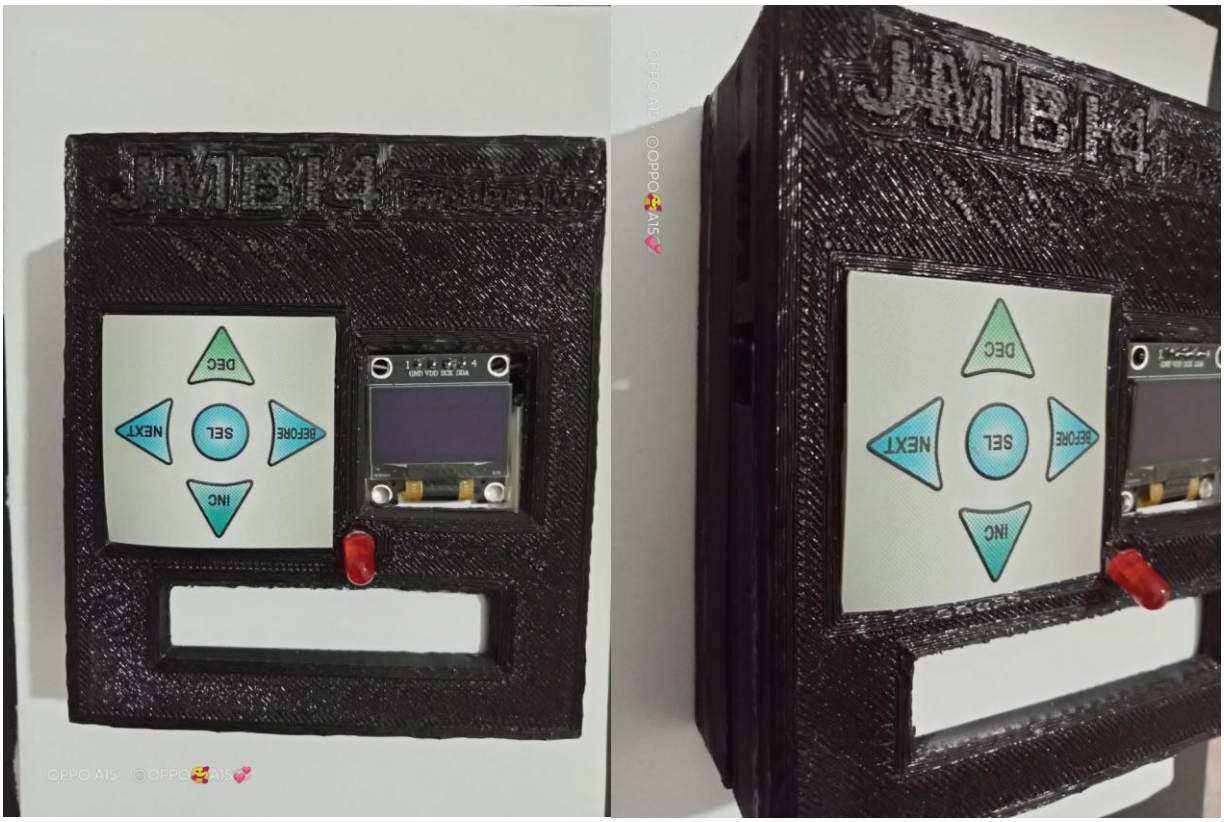


Figure III.17: Final Integration into the Enclosure.

III.7 Testing and Results

III.7.1 Testing Methodology

To validate the system's functionality and performance, a series of methodical tests were conducted on the final prototype, fully assembled within its enclosure. These tests were designed to first evaluate each of the system's functions individually, and subsequently to assess the system's performance as an integrated unit within an environment that simulates real-world operating conditions.

IV.7.2 Functional Testing

The focus of this section was to verify that each core function operates as expected. The results of these tests are summarized in the following table:

Table IV. : Summary of Core Functional Tests

Function Tested	*Test Procedure*	*Expected Result*	*Actual Result (Status)*
Automatic Control (Pump ON)	Water level was manually lowered below the 12% threshold.	The relay should activate and the pump should turn on.	Success
Automatic Control (Pump OFF)	Water level was manually raised above the 90% threshold.	The relay should deactivate and the pump should turn off.	Success
Remote Control (Wi-Fi)	An "ON" command was sent from the mobile app via the internet.	The pump should turn on.	Success
Local Control (Bluetooth)	An "OFF" command was sent from the app via Bluetooth.	The pump should turn off instantly.	Success
Control via SMS	An SMS containing the command " ON" was sent.	The pump should turn on.	Success
Cloud Data Upload	The data channel on ThingSpeak was monitored.	Water level readings should appear periodically.	Success
OLED Water Level Display	The water level in the tank was varied while monitoring the screen.	The screen should display the water level value correctly and in sync with the sensor readings.	Success

III.7.3 Performance Testing

1.Data Filter Effectiveness Test

- **Objective:** The objective of this test was to quantitatively evaluate the performance of the software filter applied to the HC-SR04 sensor readings and to measure its success in increasing data stability and reliability.
- **Test Procedure:** Two sets of data were recorded from the sensor under static conditions. The first set represents the percentages calculated from the raw readings (unfiltered), and the second set represents the final percentages after applying the software filter.
- **Results and Analysis:** The following table shows a direct comparison between the readings. Upon statistical analysis, the Standard Deviation, which measures data dispersion, was found to have dropped sharply from 8.34 for the raw percentages to just 0.15 for the filtered percentages

  Table IV: Comparison of Raw and Filtered Percentages

(%) Filtered Percentage	(%) Raw Percentage	Reading
81.0	77	1
81.2	66	2
80.9	83	3
81.1	88	4
81.0	83	5
80.8	72	6
0.15	8.34	Standard Deviation

To display this difference visually, the graph in Figure IV. shows the same data, where the sharp fluctuation of the line for the raw percentages is evident compared to the high stability of the line for the filtered percentages.

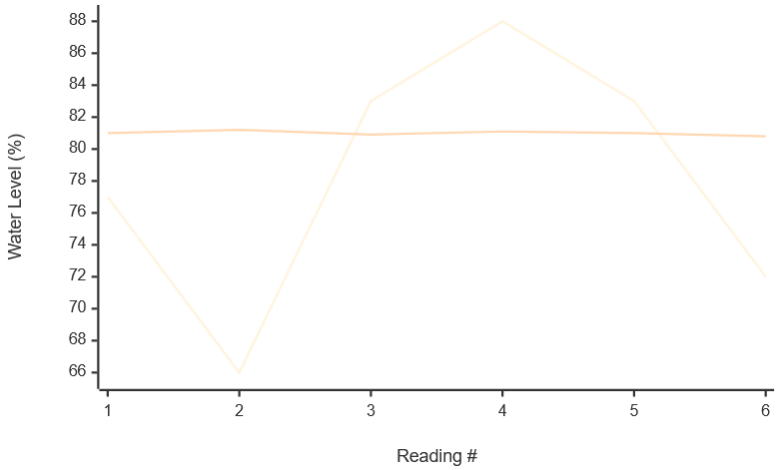


Figure III.18: Comparison of raw versus filtered water level data.

These numerical and visual results clearly demonstrate that the software filter was highly effective at eliminating outlier values and providing stable, reliable measurements. This is essential for ensuring the accuracy of the system's automatic control logic and preventing it from making incorrect decisions.

2.Command Response Time Test

- **Objective:** To measure the system's response speed to manual commands sent via the different communication channels.
- **Test Procedure:** The time elapsed between the moment a command was sent from the mobile application and the moment the relay responded was measured.
- **Results:**
  - **Via Wi-Fi:** The average response time was 2.48s, a duration dependent on the network and ThingSpeak servers.
  - **Via Bluetooth:** The response time was less than one second (nearly instantaneous).
  - **Via GSM (SMS):** The average response time was 5.67s, a time which is dependent on the GSM network's message delivery delays.

3.SMS Channel Reliability Test

- **Objective :**To measure the reliability of the backup control channel.
- **Test Procedure:** 20consecutive text messages containing "ON" and "OFF" commands were sent to the SIM800L module.

Result : The system successfully responded to 20 out of 20 commands, yielding a 100% success rate, which is a very good reliability rate for a backup communication channel.

III.8 Conclusion

In conclusion, this chapter presented the design and implementation process of a water level monitoring and pump control system, covering everything from system architecture and hardware development to microcontroller programming and the creation of a mobile application supporting Bluetooth, Wi-Fi, and GSM communication. The system has successfully demonstrated its core capabilities by monitoring the water level and controlling the pump automatically or manually based on defined conditions.

However, it is important to note that this prototype represents an initial design that is still under development. Several technical and practical challenges were identified during testing and integration phases, and ongoing efforts are being made to find effective solutions. The current design is intentionally flexible and can be modified or expanded based on the specific needs of the environment or user. Future improvements may include enhanced sensor accuracy, better power optimization, and a more advanced user interface within the mobile application.

Therefore, this system serves as a practical first step toward building a smart and reliable solution to the water level management problem, while remaining open to future enhancements and adaptation.

General Conclusion

General Conclusion

In this thesis, we have designed and implemented a comprehensive IoT-based water level monitoring and control system that integrates hardware, software, and cloud technologies to address the challenges of traditional water management methods. Starting with a thorough study of IoT architectures and communication protocols, we identified the ESP32 microcontroller as the core component due to its versatility and built-in wireless capabilities.

Through the integration of ultrasonic sensors, relay modules, and multiple communication interfaces including Wi-Fi, Bluetooth, and GSM, the system achieved real-time monitoring, automated control, and flexible connectivity modes. A dedicated mobile application was developed to serve as an intuitive user interface, enabling remote and local operation while displaying critical data and alerts.

During the development and testing phases, particular attention was given to power optimization, system reliability, and ease of use. The implementation of Deep Sleep modes and the replacement of LCD with OLED displays significantly improved energy efficiency. Final tests demonstrated the system's ability to perform all target functionalities effectively, with reliable switching between control modes and accurate data reporting.

Overall, this project highlights the potential of IoT solutions in modern resource management, offering scalable, cost-effective, and user-friendly tools to improve water conservation and operational efficiency. The developed prototype serves as a foundation for further enhancements, such as integrating machine learning algorithms for predictive maintenance and expanding the system to manage additional environmental parameters.

References

References

References

[1] Jabbar, M. A. (2024). *Internet Of Things (IOT): Origins, Embedded Technologies, Smart Applications and its Growth in the Last Decade.* ResearchGate. https://www.researchgate.net/publication/381689333_Internet_Of_Things_IOT_Origins_Embedded_Technologies_Smart_Applications_and_its_Growth_in_the_Last_Decade

[2] Benghozi, P.-J., Bureau, S., Massit-Folléa, F., Waroquiers, C., & Davidson, S. (2009). *L'internet des objets: quels enjeux pour l'Europe*. Éditions de la Maison des sciences de l'homme.

[3] Copeland, B. J. (2024). *Internet of Things.* Britannica. https://www.britannica.com/science/Internet-of-Things

[4] TechProgress. (n.d.). *Internet of Things overview*. Retrieved July 11, 2025, from https://techprogress.co/internet.php

[5] Vermesan, O., & Friess, P. (2013). *Internet of things: Converging technologies for smart environments*. River Publishers.

[6] Arduino. (n.d.). *Arduino official documentation*. Retrieved June 12, 2025, from https://www.arduino.cc/

Espressif. (n.d.). *ESP32 series*. Retrieved June 12, 2025, from https://www.espressif.com/en/products/socs/esp32

Raspberry Pi Foundation. (n.d.). *Raspberry Pi documentation*. Retrieved June 12, 2025, from https://www.raspberrypi.org/documentation/

[7] HashStudioz. (n.d.). *Top IoT development boards: How to select the right one for your project*. Retrieved July 11, 2025, from https://www.hashstudioz.com/blog/top-iot-development-boards-how-to-select-the-right-one-for-your-project/

[8] GeeksforGeeks. (2024). *Computer network architecture.* Retrieved July 8, 2025, from https://www.geeksforgeeks.org/computer-networks/architecture-of-internet-of-things-iot/

[9] Cavli Wireless. (2023, September 12). *How HTTP powers communication in IoT networks*. https://www.cavliwireless.com/blog/not-mini/how-http-powers-communication-in-iot-networks

[10] [4] Weixin_41835977. (2019). *IoT-A——物联网体系架构.* CSDN. Retrieved July 8, 2025, from https://blog.csdn.net/weixin_41835977/article/details/88915447

[11] Shelby, Z., Hartke, K., & Bormann, C. (2014). *The Constrained Application Protocol (CoAP)* (RFC 7252). Internet Engineering Task Force. https://doi.org/10.17487/RFC7252

[12] OASIS. (2012). *Advanced Message Queuing Protocol (AMQP) Version 1.0*. http://docs.oasis-open.org/amqp/core/v1.0/os/amqp-core-overview-v1.0-os.html

References

[13] Object Management Group. (2015, March). *Data Distribution Service (DDS) version 1.4* (Formal/2015-04-10). https://www.omg.org/spec/DDS/1.4/

[14] Fette, I., & Melnikov, A. (2011). *The WebSocket protocol* (RFC 6455). Internet Engineering Task Force. https://doi.org/10.17487/RFC6455

[15] Gupta, H., Tanwar, S., Tyagi, S., Kumar, N., & Obaidat, M. S. (2022). *Smart irrigation system using Internet of Things and cloud computing*. Processes, 10(11), 2462. https://www.mdpi.com/2227-9717/10/11/2462

[16] Mischianti, L. (n.d.). *ESP32 NodeMCU 32S ESP-32S kit: High resolution pinout, datasheet and specs*. Mischianti.org. Retrieved June 12, 2025, from https://mischianti.org/esp32-nodemcu-32s-esp-32s-kit-high-resolution-pinout-datasheet-and-specs/

[17] Microcontrollers Lab. (n.d.). *STM32F103C8T6 blue pill pinout, peripherals, programming features*. Retrieved June 12, 2025, from https://microcontrollerslab.com/stm32f103c8t6-blue-pill-pinout-peripherals-programming-features/

[18] Fierce Electronics. (n.d.). *What is an ultrasonic sensor?* Retrieved June 12, 2025, from https://www.fierceelectronics.com/sensors/what-ultrasonic-sensor

[19] SparkFun Electronics. (n.d.). *HC-SR04 ultrasonic sensor datasheet*. https://cdn.sparkfun.com/datasheets/Sensors/Proximity/HCSR04.pdf

[20] Amazon. (n.d.). *Ultrasonic Accuracy Measuring Module, Rangefinder 2cm–300cm*. Amazon.nl. Retrieved May 25, 2025, from https://www.amazon.nl/-/en/Ultrasonic-Accuracy-Measuring-Rangefinder-2cm-300cm/dp/B0CCS72MBG

[21] LED-Diode. (n.d.). *JSN-SR04T Integrated Ultrasonic Ranging Module*. Retrieved May 25, 2025, from https://ar.led-diode.com/power-module/jsn-sr04t-integrated-ultrasonic-ranging.html

[22] Waveshare. (n.d.). *TF-Luna LiDAR range sensor*. Retrieved June 12, 2025, from https://www.waveshare.com/wiki/TF-Luna_LiDAR_Range_Sensor

[23] Cirkit Designer. (n.d.). *TF-Luna LiDAR – Component Reference*. Retrieved May 25, 2025, from https://docs.cirkitdesigner.com/component/f682a215-2fff-41de-ad16-70661f4475ac/tf-luna-lidar

[24] LastMinuteEngineers. (n.d.). *In-Depth: Interfacing TFMini-S LiDAR Sensor with Arduino*. Retrieved May 25, 2025, from https://lastminuteengineers.com/tfmini-s-lidar-sensor-arduino-tutorial/

[25] DFRobot. (n.d.). *Turbidity sensor SKU: SEN0189*. https://media.digikey.com/pdf/data%20sheets/dfrobot%20pdfs/sen0189_web.pdf

References

[26] DFRobot. (n.d.). *Gravity: Analog Turbidity Sensor for Arduino*. Botland.store. Retrieved May 25, 2025, from https://botland.store/gravity-temperature-sensors/5318-dfrobot-gravity-analogue-turbidity-sensor-6959420909674.html

[27] Generic. (n.d.). *Turbidity Sensor TSD-10 – Genuine Robotics*. Amazon.in. Retrieved May 25, 2025, from https://www.amazon.in/Generic-Turbidity-Sensor-TSD-10/dp/B07S8LZC3T

[28] DFRobot. (n.d.). *Gravity: Analog pH sensor / meter kit V2*. Retrieved June 12, 2025, from https://wiki.dfrobot.com/Gravity__Analog_pH_Sensor_Meter_Kit_V2_SKU_SEN0161-V2

[29] Rapid Online. (n.d.). *Gravity: Analog pH Sensor / Meter Kit for Arduino (SEN0161)*. Retrieved May 28, 2025, from https://www.rapidonline.com/dfrobot-sen0161-gravity-analog-ph-sensor-meter-kit-for-arduino-75-0249

[30] Atlas Scientific. (n.d.). *Conductivity K 0.1 Kit*. Arduitronics. Retrieved May 28, 2025, from https://www.arduitronics.com/product/4504/conductivity-k-0-1-kit-%E0%B9%81%E0%B8%97%E0%B9%89%E0%B8%88%E0%B8%B2%E0%B8%81-atlas-scientific-made-in-usa

[31] DFRobot. (n.d.). *Gravity: Analog TDS sensor / meter for Arduino*. Retrieved June 12, 2025, from https://wiki.dfrobot.com/Gravity__Analog_TDS_Sensor___Meter_for_Arduino_SKU__SEN0244

[32] DFRobot. (n.d.). *Gravity: Analog TDS Sensor / Meter for Arduino (SKU: SEN0244)*. Retrieved May 29, 2025, from https://wiki.dfrobot.com/Gravity__Analog_TDS_Sensor___Meter_for_Arduino_SKU__SEN0244

[33] HIT. (n.d.). *ACS712 hall current sensor (5A/20A/30A)*. Retrieved June 12, 2025, from https://hit.ps/ar/product/acs712-hall-current-sensor-5a-20a-30a/

[34] Amazon.in. (n.d.). *Diitao Current DC0-25V Voltage Terminal*. Retrieved June 12, 2025, from https://www.amazon.in/Diitao-Current-DC0-25V-Voltage-Terminal/dp/B0BBLNVN8K?th=1

[35] Addicore. (n.d.). *1 channel relay module – active high control*. Retrieved June 12, 2025, from https://www.addicore.com/products/1-channel-relay-module-active-high-control

[36] Arduino Store. (n.d.). *1 Relay Module 5VDC 10A (Assembled)*. Retrieved June 12, 2025, from https://store.arduino.cc/products/1-relay-module-5-vdc-10a-assembled

[37] Mhtronic. (n.d.). *Matrix clavier 5 touches*. Retrieved June 12, 2025, from https://mhtronic.com/produit/matrix-clavier-5-touches/

[39] Freepik. (n.d.). *Componentes eletrônicos em PCB em fábrica de laboratório de alta tecnologia*. Retrieved June 12, 2025, from https://br.freepik.com/componentes-eletronicos-em-pcb-em-fabrica-de-laboratorio-de-alta-tecnologia_47771824.htm

References

[40] Universal Solder. (n.d.). *1.3 inch OLED I2C 128x64 3.5V*. Retrieved June 12, 2025, from https://www.universal-solder.ca/product/1-3-inch-oled-i2c-128x64-3-5v/

[41] Naylamp Mechatronics. (n.d.). *Display OLED 0.96 I2C 128x64 SSD1306*. Retrieved June 12, 2025, from https://naylampmechatronics.com/oled/850-display-oled-096-i2c-12864-ssd1306.html

[42] Finnley Electrical. (n.d.). *What is electrical resistance?* Retrieved June 12, 2025, from https://www.finnleyelectrical.com.au/what-is-electrical-resistance/

[43] AZ-Delivery. (n.d.). *Breadboard*. Retrieved June 12, 2025, from https://www.az-delivery.de/fr/products/breadboard

[44] Espressif Systems. (n.d.). *ESP32 Bluetooth features*. Retrieved June 12, 2025, from https://www.espressif.com/en/products/socs/esp32

[45] FishPoint. (n.d.). *ESP32 tutorial or content*. Retrieved June 12, 2025, from https://fishpoint.tistory.com/10192

[46] HnHCart. (n.d.). *Difference between SIM800L, SIM800A & SIM900A*. Retrieved June 12, 2025, from https://www.hnhcart.com/blogs/sensors-modules/difference-between-sim800l-sim800a-sim900a

[47] YouTube. (n.d.). *ESP module explanation video*. Retrieved June 12, 2025, from https://www.youtube.com/watch?v=P_yLXjCA_0c

[48] STMicroelectronics. (n.d.). *STM32CubeProgrammer*. Retrieved June 12, 2025, from https://wiki.stmicroelectronics.cn/stm32mpu/index.php?title=STM32CubeProgrammer&oldid=74437

[49] MathWorks. (n.d.). *ThingSpeak documentation*. Retrieved June 12, 2025, from https://www.mathworks.com/help/thingspeak/index.html

[50] ThingSpeak. (n.d.). *Channel 2840128*. Retrieved June 12, 2025, from https://thingspeak.mathworks.com/channels/2840128/private_show

[51] ThingSpeak. (n.d.). *Channel 2778561*. Retrieved June 12, 2025, from https://thingspeak.mathworks.com/channels/2778561/private_show

[52] MIT App Inventor. (n.d.). *Hour of Code*. Retrieved June 12, 2025, from https://appinventor.mit.edu/explore/hour-of-code

[53] MIT Center for Mobile Learning. (n.d.). *Designer and blocks editor overview*. Retrieved June 12, 2025, from https://mit-cml.github.io/explore/designer-blocks.html

[54] MIT App Inventor. (n.d.). *App Inventor project interface*. Retrieved June 12, 2025, from https://ai2.appinventor.mit.edu/#5632802149171200

[55] EasyEDA. (n.d.). *EasyEDA editor*. Retrieved June 12, 2025, from https://easyeda.com/editor

الجمهورية الجزائرية الديمقراطية الشعبية

وزارة التعليم العالي و البحث العلمي

جامعة محند خبضر بسكرة

عنوان المشروع:

جهاز ذكي لقياس مستوى الماء في الخزانات والتحكم في مضخة المياه عن بُعد باستخدام WiFi ، GSM و Bluetooth

مشروع لنيل شهادة مؤسسة ناشئة في اطار القرار الوزاري 1275

صورة العلامة التجارية

الاسم التجاري

JMBI4

السنة الجامعية

2025 _ 2024

بطاقة معلومات:

حول فريق الاشراف وفريق العمل

1- فريق الاشراف:

فريق الاشراف	
المشرف الرئيسي (01): بخوش خالد	التخصص: الكترونيك

2- فريق العمل:

فريق المشروع	التخصص	الكلية
الطالب: مرزوقي بسمة	شبكات واتصالات	علوم وتكنولوجيا
الطالب: ميمون جهينة	شبكات واتصالات	علوم وتكنولوجيا

فهرس المحتويات

عنوان المشروع : **جهاز ذكي لقياس مستوى الماء في الخزانات والتحكم في مضخة المياه عن بُعد باستخدام WiFi ، GSM وBluetooth**

المحور الأول: تقديم المشروع

المحور الثاني: الجوانب الابتكارية

المحور الثالث: التحليل الاستراتيجي للسوق

المحور الرابع: خطة الإنتاج والتنظيم

المحور الخامس :الخطة المالية

المحور السادس : النموذج الاولي التجريبي

عنوان المشروع : **جهازذكي لقياس مستوى الماء في الخز انات والتحكم في مضخة المياه عن بُعد باستخدام WiFi ، GSM وBluetooth**

المحور الاول: تقديم المشروع

عنوان المشروع : **جهازذكي لقياس مستوى الماء في الخز انات والتحكم في مضخة المياه عن بُعد باستخدام WiFi ، GSM وBluetooth**

المحور الأول: تقديم المشروع

مقدمة

مع تنامي الحاجة إلى ترشيد استهلاك المياه وتحسين إدارة الموارد المائية في مختلف القطاعات، أصبحت الحلول الذكية ضرورة ملحة في الحياة المعاصرة. في ظل هذا التحدي، يُقدم مشروع JMBI 4 نظامًا متكاملًا لمراقبة مستوى الماء في الخزانات والتحكم في تشغيل أو إيقاف المضخة عن بُعد، باستخدام ثلاث تقنيات اتصال رئيسية WiFi، GSM، و.Bluetooth.

يهدف المشروع إلى تقديم حل تقني موثوق ومرن يمكن استخدامه في البيوت، المزارع، البلديات، وحتى المؤسسات التي تعتمد على الخزانات لتخزين المياه. من خلال دمج تقنيات الاتصال الحديثة مع وحدات استشعار دقيقة، يمكن مراقبة الحالة اللحظية لمستوى المياه والتحكم الذكي في المضخة بناءً على بيانات فورية تُعرض في تطبيق هاتف ذكي أو تُرسل عبر رسائل نصية.

يمثّل هذا الابتكار مساهمة محلية في سوق يعاني من نقص في الحلول المتكاملة والبسيطة لإدارة المياه، ويوفر بديلًا موثوقًا للأجهزة المستوردة الباهظة الثمن

1. فكرة المشروع (الحل المقترح)

يُعاني الكثير من المستخدمين، سواء في المنازل أو المزارع أو المؤسسات، من صعوبة مراقبة خزانات المياه والتحكم في المضخة بشكل يدوي. يؤدي هذا غالبًا إلى مشاكل مثل فيضان الخزان، أو نفاد المياه دون إنذار مسبق، مما يؤثر على الحياة اليومية.

الحل المقترح هو تصميم جهاز ذكي منخفض التكلفة مزوّد بمستشعرات لمراقبة مستوى الماء، يمكنه إرسال بيانات دقيقة في الزمن الحقيقي باستخدام WiFi أوBluetooth ، أو إرسال تنبيهات وتحكمات عبر رسائل SMS باستخدام وحدة.GSM

يُتيح النظام تشغيل/إيقاف المضخة تلقائيًا أو يدويًا عبر تطبيق هاتفي بسيط، ويعرض إشعارات وتنبيهات فورية عند امتلاء أو فراغ الخزان، مما يساعد المستخدمين على اتخاذ قرارات سريعة دون الحاجة لتفقد الخزان ميدانيًا.

2. القيم المقترحة

- حل ذكي محلي الصنع لإدارة المياه.
- ترشيد استهلاك المياه والطاقة
- سهولة الاستخدام لجميع الفئات (فلاحون، عائلات، تقنيون...)
- تقنيات اتصال متعددة لتناسب كل البيئات (مناطق حضرية / نائية)
- سعر منخفض مقارنة بالأجهزة الأجنبية.
- تقليل هدر المياه وتحسين الكفاءة.
- توفير الراحة والتحكم عن بعد

3. فريق العمل

- ميمون جهينة .
- مرزوقي بسمة .

4. أهداف المشروع

- تقديم منتج موثوق وعالي الجودة للسوق المحلي بسعر تنافسي.
- تحقيق مبيعات أولية تغطي تكاليف الإنتاج خلال السنة الأولى.
- بناء علامة تجارية معروفة في مجال حلول إنترنت الأشياء المنزلية والزراعية.

5. جدول زمني لتحقيق المشروع:

	المهمة العملية	الشهر الأول	الشهر الثاني	الشهر الثالث	**نسبة الإنجاز المقدرة**
1	تحديد فكرة المشروع و توثيقها	1-2الأسبوع			100%
2	اعداد الدراسة التقنية الأولية	2-4الأسبوع	1-2الأسبوع		90%
3	تصميم الدارة الالكترونية و اختيار المكونات	3-4الأسبوع	1-2الأسبوع		80%
4	برمجة المستشعرات و ربطها بمنصة Thingspeak		2-4الأسبوع	1-2الأسبوع	60%
5	تصميم التطبيق	3-4الأسبوع		1-2الأسبوع	70%
6	تجربة الجهاز على حالات واقعية			2-3الأسبوع	50%
7	تقييم النتائج و التحسينات التقنية			3-4الأسبوع	40%
8	اعداد التقرير النهائي و العرض التقديمي			4الأسبوع	60%

عنوان المشروع : **جهازذكي لقياس مستوى الماء في الخزانات والتحكم في مضخة المياه عن بُعد باستخدام WiFi ، GSM وBluetooth**

المحور الثاني :الجوانب الابتكارية

عنوان المشروع : **جهازذكي لقياس مستوى الماء في الخز انات والتحكم في مضخة المياه عن بُعد باستخدام WiFi ، GSM وBluetooth**

المحور الثاني: الجو انب الابتكارية

1. طبيعة الابتكار

تعتبر طبيعة الابتكار لهذا المشروع من الابتكارات التكنولوجية(Technological Innovations) .

2. مجالات الابتكار

يمزج مشروع JMBI 4 بين تقنيات متعددة لجمع البيانات والتحكم عن بُعد، وهو ما يمنحه طابعًا ابتكاريًا من عدة جوانب:

- ابتكار تقني :يجمع المشروع بين تقنيات الاتصال اللاسلكيWiFi ، GSM، Bluetoothفي نظام واحد.
- ابتكار وظيفي :يوفر مراقبة فورية وتحكم مباشر أو آلي بالمضخة حسب المستوى.
- ابتكار في السوق :يُعد من أوائل المشاريع المحلية التي تقدم هذا النوع من الحلول الذكية منخفضة التكلفة.
- ابتكار في الاستعمال :لا يتطلب خبرة تقنية كبيرة ويعمل في بيئات حضرية أو ريفية.
- التحكم الديناميكي :النظام ليس ثابتًا، بل يسمح للمستخدم بتعريف أبعاد الخزان، ويقوم النظام بحساب عتبات التحكم تلقائيًا، مما يجعله قابلاً للتكيف مع أي بيئة .
- التصميم المتكامل :ابتكار منتج كامل من الألف إلى الياء، بدءًا من تصميم الPCB ، مرورًا بتصميم الغلاف الميكانيكي، وانتهاءً بتطوير البرمجيات وتطبيق الهاتف.

عنوان المشروع : **جهازذكي لقياس مستوى الماء في الخزانات والتحكم في مضخة المياه عن بُعد باستخدام WiFi ، GSM وBluetooth**

المحور الثالث : التحليل الاستراتيجي للسوق

المحور الثالث: التحليل الاستراتيجي للسوق

رغم توفر بعض الحلول المستوردة، إلا أن السوق الجزائري يفتقر إلى حلول محلية:

- **نقاط القوة:**
  - تكلفة منخفضة.
  - مرونة في الاتصال.
  - سهل التركيب والاستخدام.
- **نقاط الضعف:**
  - نقص المكونات الإلكترونية في السوق المحلي.
  - غياب الدعم المؤسساتي للمشاريع المائية.
- **الفرص:**
  - توجه الدولة نحو الرقمنة.
  - زيادة الوعي بأهمية ترشيد المياه.
- **التهديدات:**
  - المنتجات المستوردة الرخيصة.
  - تقلب أسعار المكونات

هيكل السوق

يعتمد السوق الجزائري أسا سا على الاستيراد مع ضعف واضح في الإنتاج المحلي، ما يخلق فرا غا يمكن أن تستغله المؤسسات الناشئة المحلية.

شدة المنافسة

المنافسة في السوق لا تزال ضعيفة إلى متوسطة، نظرا لقلة المشاريع المحلية وعدم توفر أجهزة ذكية مشابهة، ما يمنح فرصة كبيرة للابتكار.

عنوان المشروع : **جهازذكي لقياس مستوى الماء في الخز انات والتحكم في مضخة المياه عن بُعد باستخدام WiFi ، GSM وBluetooth**

قياس شدة المنافسة

- المنافسون المباشرون: لا يوجد حاليًا في السوق المحلي منافسون مباشرون يقدمون منتجًا متكاملاً بنفس الميزات (تطبيق هاتف + 3 طرق اتصال + إعدادات ديناميكية).
- المنافسون غير المباشرين: الحلول التقليدية مثل العوامات الميكانيكية، الدوائر الإلكترونية البسيطة التي تعتمد على مؤقتات، أو أنظمة مستوردة قد تكون باهظة الثمن.

المحور الرابع:
خطة الانتاج و التنظيم

عنوان المشروع : **جهازذكي لقياس مستوى الماء في الخز انات والتحكم في مضخة المياه عن بُعد باستخدام WiFi ، GSM وBluetooth**

المحور الرابع :خطة الإنتاج والتنظيم

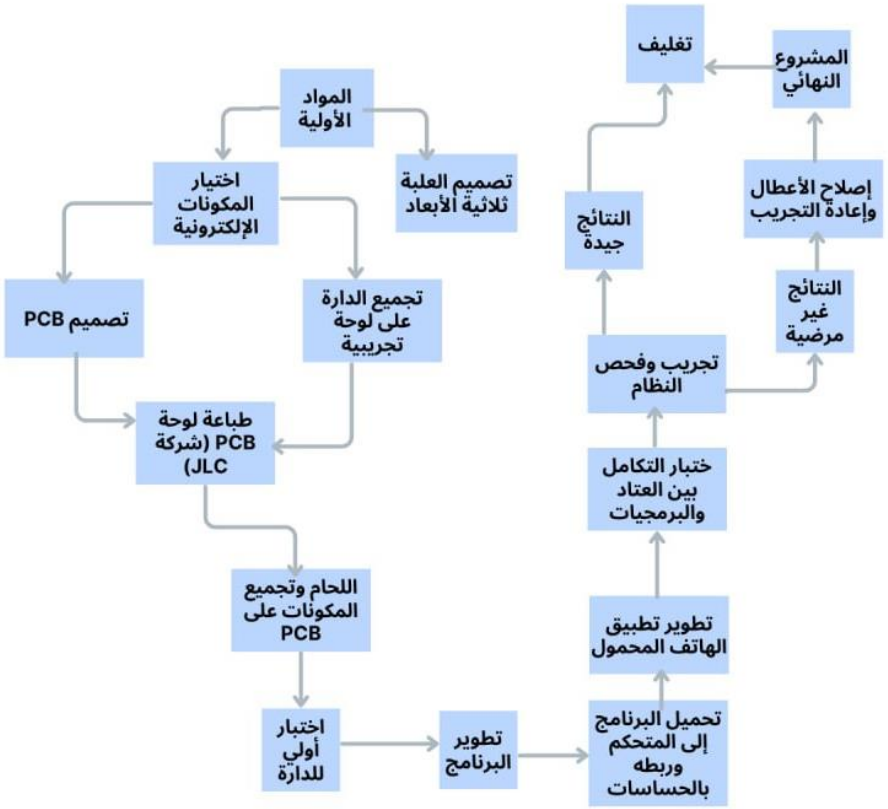
المرحلة 1: إعداد التصميم النظري والدارة الكهربائية.

المرحلة 2: اقتناء ESP32، حساس HC-SR04، وحدة GSM، Relay، إلخ.

المرحلة 3: برمجة النظام واختبار الاتصال بالتطبيق.

المرحلة 4: تجربة النظام ميدانيًا في سيناريوهات حقيقية.

المرحلة 5: جمع الملاحظات والتحسينات.



عنوان المشروع : **جهازذكي لقياس مستوى الماء في الخزانات والتحكم في مضخة المياه عن بُعد باستخدام WiFi ، GSM وBluetooth**

:

المحور الخامس

الخطة المالية

المحور الخامس: الخطة المالية

1. التكاليف والاعباء:

تتمثل تكاليف المشروع كما يلي:

.التكاليف الثابتة

التكاليف الادارية والعمومية : تقدر قيمة الشهرية للإيجار ب 20000DA وتقدر قيمة السنوية للإيجار ب 240000DA بالإضافة الى تكاليف الهاتف وانترنت والضرائب كما هو موضح في الجدو ل:

نوع العبء	القيمة التقديرية السنوية (دج)
إيجار مقر أو فضاء عمل	120,000دج
مصاريف نقل ولوجستيك	20,000دج
ضرائب ورسوم رمزية	24,000دج
أدوات مكتبية وتغليف	10,000دج
المجموع الكلي	174,000دج

تكلفة الكهرباء والماء: تتمثل تكلفة الكهرباء والماء كما موضح في الجدول:

نوع المصروف	القيمة التقديرية السنوية (دج)
فاتورة الكهرباء	18,000دج
فاتورة الماء	2,000دج
اشتراك الإنترنت	12,000دج
اشتراك الهاتف	6,000دج
المجموع	38,000دج

عنوان المشروع : **جهاز ذكي لقياس مستوى الماء في الخزانات والتحكم في مضخة المياه عن بُعد باستخدام WiFi ، GSM وBluetooth**

الرواتب: تتمثل رواتب الموظفين كما هو ممثل في الجدول التالي:

الوظيفة	عدد الأشخاص	الراتب الشهري (دج)	الراتب السنوي الإجمالي (دج)
مطور إلكترونيات وبرمجة	1	40,000	480,000
تقني تركيب وصيانة	1	30,000	360,000
مدير المشروع	1	45,000	540,000
موظف دعم وخدمة زبائن	1	25,000	300,000
المجموع الكلي	—	—	1,680,000دج

4.جدول تكاليف الإنتاج السنوية (50 وحدة)

البند	التكلفة للوحدة	الكمية	الإجمالي السنوي
لوحةESP32	2,500	50	125,000
حساس مستوى الماء HC-SR04	400	50	20,000
وحدةGSM SIM800L	2,000	50	100,000
وحدةRelay	600	50	30,000
بطارية وصندوق خارجي	1,500	50	75,000
شاشةLCD	1,800	50	90,000
أسلاك ومواد لحام	200	50	10,000
المجموع	—	—	450,000دج

عنوان المشروع: **جهاز ذكي لقياس مستوى الماء في الخزانات والتحكم في مضخة المياه عن بُعد باستخدام WiFi ، GSM وBluetooth**

تكاليف الانشاء:

جدول تكاليف الإنشاء الأولية

البند	الكمية	التكلفة لواحد	التكلفة الإجمالية
كراء أو تجهيز مقر العمل	1	120,000	**120,000**
شراء أدوات ومعدات لحام وتركيب	1	100,000	**100,000**
شراء حاسوب أو تجهيز برمجي	1	150,000	**150,000**
مكتب وكراسي وتجهيزات مكتبية	1	80,000	**80,000**
مصاريف تأسيس قانوني وتسجيل	1	50,000	**50,000**
أدوات حماية ولباس عمل	2	10,000	**20,000**
لافتة ولوحة تعريفية للمشروع	1	15,000	**15,000**
المجموع الكلي	—	—	**535,000 دج**

اجمالي التكاليف:

البند	القيمة (دج)
الرواتب السنوية للفريق	**1,680,000**
تكاليف الإنتاج السنوية (50 وحدة)	**450,000**
تكاليف الإنشاء الأولية	**535,000**
الإجمالي العام	**2,665,000دج**

عنوان المشروع: **جهازذكي لقياس مستوى الماء في الخزانات والتحكم في مضخة المياه عن بُعد باستخدام WiFi ، GSM وBluetooth**

حساب إيرادات المنتج الواحد

لحساب الإيرادات المتوقعة لكل وحدة من جهاز4 JMBI ، يتم أولًا تحديد التكلفة الإجمالية لتجميع وحدة واحدة، ثم إضافة هامش ربح بناءً على السوق والمنافسة.

.1التكلفة الإجمالية التقديرية للوحدة:

السعر التقديري (دج)	المكون
2,500دج	**لوحةESP32**
400دج	**حساسHC-SR04**
2,000دج	**وحدة(SIM800L) GSM**
600دج	**وحدةRelay**
1,500دج	**بطارية وعلبة خارجية**
1,800دج	**شاشة) LCD اختيارية(**
200دج	**أسلاك، لحام، إلخ**
10,800 – 9,000دج	**المجموع التقريبي**

.2تحديد هامش الربح والسعر المقترح للبيع:

بناءً على المنافسة وتكاليف المشروع، تم اقتراح أن يكون سعر البيع في السنة الأولى: .8000دج.

.3معادلة حساب الإيراد الصافي للوحدة:

إيرادات المنتج الواحد = سعر البيع - التكلفة الإجمالية للوحدة

مثال تطبيقي:

-سعر البيع =8000 دج

-التكلفة = 5000 دج

-الربح الصافي لكل وحدة = 3000 دج

هذه المعادلة تُساعد على تحليل ربحية المشروع مع التوسع، وتسمح بضبط السعر حسب تغير التكاليف أو ظروف السوق.

14.رقم الأعمال ومصادر الإيرادات

1.حساب رقم الأعمال(Chiffre d'affaires)

يُحسب رقم الأعمال بضرب عدد الوحدات المباعة في سعر البيع لكل وحدة:

السنة	عدد الأجهزة المباعة	سعر البيع للوحدة	رقم الأعمال المتوقع
1	50	8000دج	400,000دج
2	100	7000دج	700,000دج
3	150	6000دج	900,000دج
4	200	5000دج	1,000,000دج
5	250	4000دج	1,000,000دج

رقم الأعمال = عدد الأجهزة × سعر البيع

2.مصادر الإيرادات المتنوعة

-البيع المباشر للجهاز
-خدمات التركيب والصيانة
-اشتراكات دعم سنوي ذكي
-بيع بالجملة للبلديات أو الشركات
-تخصيص الجهاز حسب الطلب
-بيع التطبيقات المرافقة أو ترقياتها
-تكوينات ودورات تدريبية

هذه المصادر تُعزّز من قدرة المشروع على تحقيق مداخيل مستدامة وقابلة للتوسع في المستقبل.

عنوان المشروع : **جهازذكي لقياس مستوى الماء في الخز انات والتحكم في مضخة المياه عن بُعد باستخدام WiFi ، GSM وBluetooth**

جدول مصادر التموين - مشروع4 JMBI

الجدول التالي يعرض مصادر التموين، نوع التموين، والتكلفة السنوية التقديرية إلى جانب القيمة المضافة لكل مصدر:

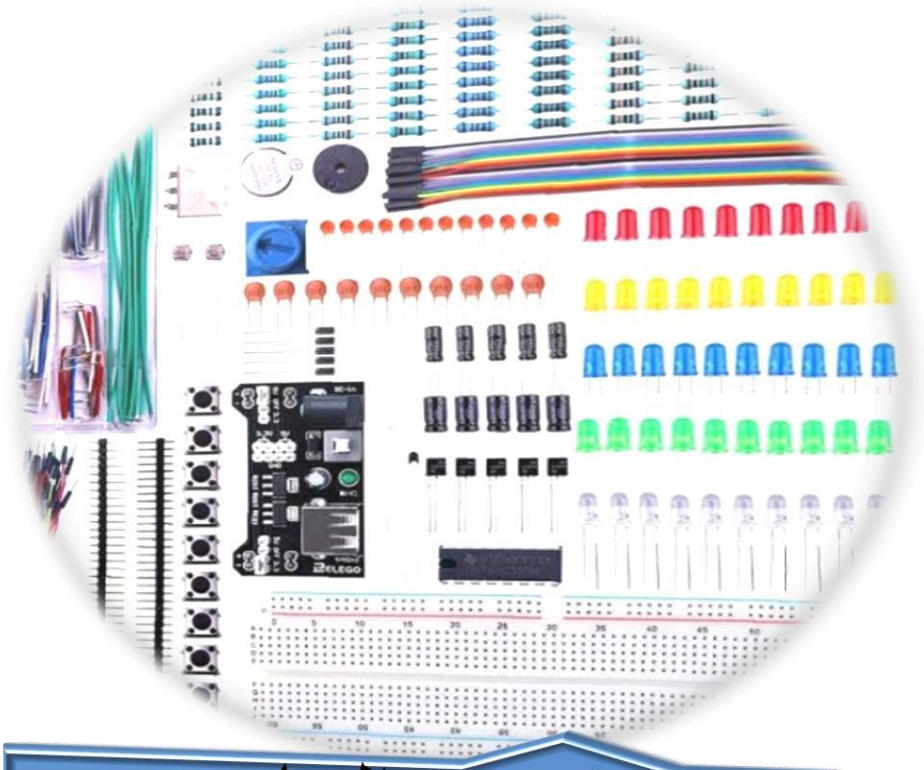
مصدر التموين	نوع التموين	التكلفة التقديرية السنوية (دج)	القيمة المالية للمساهمة أو التوفير
متاجر إلكترونيات محلية	قطع إلكترونية (ESP32)، حساسات(...	200,000دج	توفير فوري للوقت والتوصيل
مو اقع التجارة الإلكترونية	استيراد مكونات خاصة عبر الإنترنت	150,000دج	أسعار أقل لبعض المكونات بنسبة 15%
مؤسسات شريكة	دعم مالي أو تجهيزات أو فضاءات عمل	100,000دج	تقليل التكاليف التشغيلية
حاضنات أعمال	تمويل أولي، تكوين، مواكبة مشروع	300,000دج	تغطية تكاليف بدء المشروع
برامج دعم جامعية	دعم مادي وتقني لمشاريع التخرج	50,000دج	منح جامعية وتوفير المعدات
تمويل ذاتي	مساهمات داخلية من أعضاء الفريق	80,000دج	تغطية النفقات الأولية

عنوان المشروع : **جهاز ذكي لقياس مستوى الماء في الخزانات والتحكم في مضخة المياه عن بُعد باستخدام WiFi ، GSM وBluetooth**

جدول حسابات النتائج المتوقعة لخمس سنوات الأولى:

السنة	عدد الوحدات المباعة	سعر البيع للوحدة (دج)	الإيرادات (دج)	التكاليف الإجمالية (دج)	الربح الصافي (دج)
السنة 1	50	20,000	1,000,000	2,665,000	**-1,665,000 (خسارة)**
السنة 2	100	20,000	2,000,000	1,000,000*	**1,000,000**
السنة 3	150	20,000	3,000,000	1,200,000*	**1,800,000**
السنة 4	200	20,000	4,000,000	1,400,000*	**2,600,000**
السنة 5	250	20,000	5,000,000	1,600,000*	**3,400,000**

عنوان المشروع : **جهازذكي لقياس مستوى الماء في الخزانات والتحكم في مضخة المياه عن بُعد باستخدام WiFi ، GSM وBluetooth**



المحور السادس

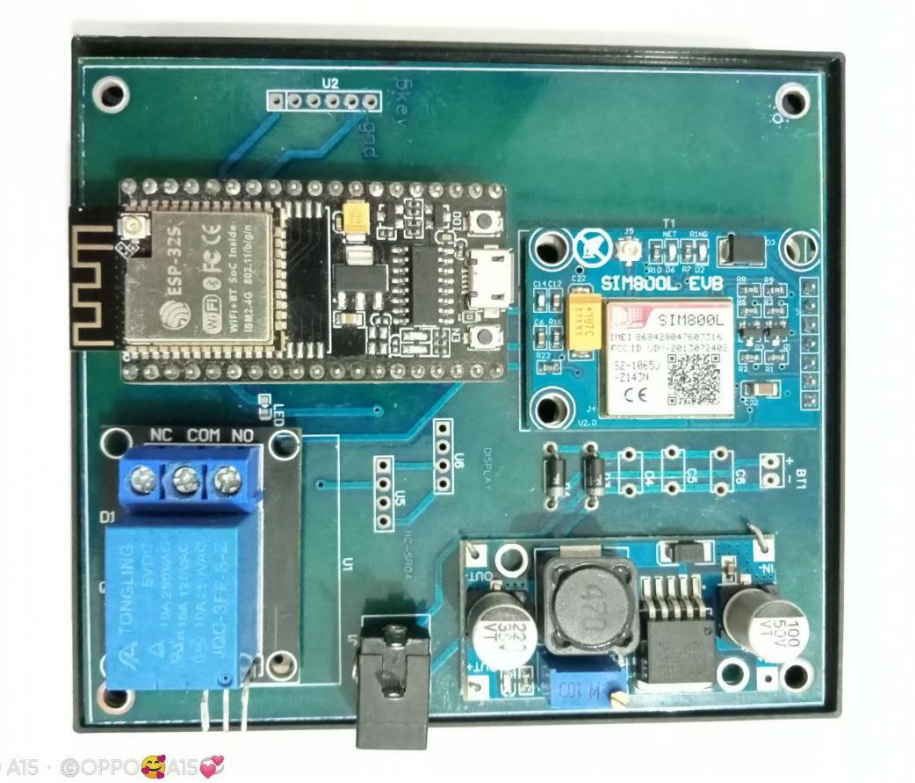
النموذج التجريبي

عنوان المشروع : **جهاز ذكي لقياس مستوى الماء في الخزانات والتحكم في مضخة المياه عن بُعد باستخدام WiFi ، GSM وBluetooth**

المحور السادس : النموذج الاولي التجريبي

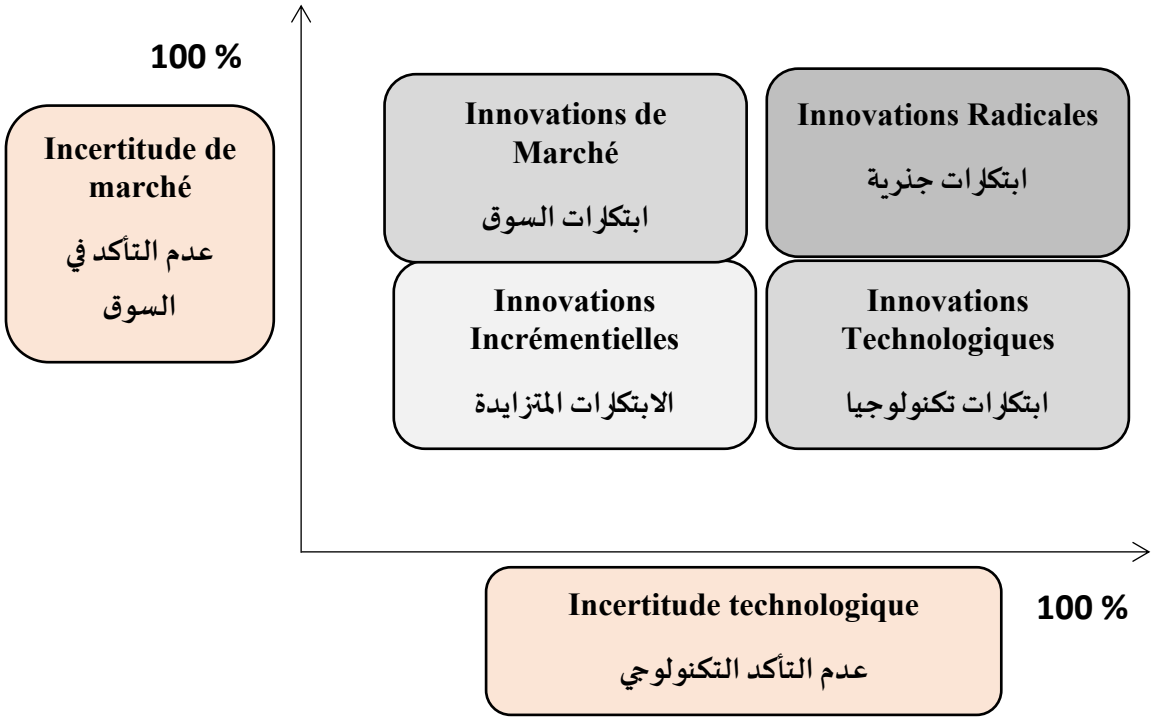
يُعد مشروع JMBI 4 خطوة واعدة نحو تطوير حلول ذكية لإدارة المياه في الجزائر، خصوصًا في ظل التحديات المتزايدة المرتبطة بترشيد الاستهلاك وضمان وفرة المياه في مختلف المجالات. من خلال هذا النموذج الأولي، تمكّنا من وضع الأسس التقنية لنظام مرن وفعال يدمج بين البساطة والذكاء، ويعتمد على تقنيات اتصال متعددة لتوفير تجربة تحكم ومراقبة متكاملة.

ورغم أن المشروع لا يزال في مراحله التجريبية، إلا أن نتائجه الأولية مشجعة وتفتح آفاقًا واسعة للتطوير المستقبلي، سواء من حيث الأداء أو التوسع في الوظائف. ومع تحسين الموثوقية وتوسيع التجارب الميدانية، يمكن لـ JMBI 4 أن يصبح أداة أساسية في إدارة الخزانات، وأن يساهم في ترشيد استهلاك المياه، وتقليل الخسائر، وتقديم خدمة ذات قيمة حقيقية للمستخدم الجزائري.



عنوان المشروع : **جهازذكي لقياس مستوى الماء في الخز انات والتحكم في مضخة المياه عن بُعد باستخدام WiFi ، GSM وBluetooth**

عنوان المشروع : **جهازذكي لقياس مستوى الماء في الخزانات والتحكم في مضخة المياه عن بُعد باستخدام WiFi ، GSM وBluetooth**



الملحق رقم 04: نموذج العمل التجاري

شرائح العملاء	العلاقات مع العملاء	القيمة المقترحة	الأنشطة الرئيسية	الشراكات الرئيسية
• المؤسسات الفلاحية. • العائلات في المنازل • الفلاحون وأصحاب المزارع • البلديات والمؤسسات • المدارس والمستشفيات	• التعاون مع الشركات الزراعية لتعزيز ع ر وضنا • توفير خدمات اصلاح الأعطال. • تقديم تجارب مجانية للمزارعين. تركيب ميداني عند الطلب **القنوات** • الإذاعات المحلية والتلفزة الوطنية . • مواقع التواصل الاجتماعي. • التوصيل بواسطة شركات التوصيل • التطبيق الهاتفي (MIT App Inventor) -رسائل SMS	مراقبة ذكية لمستوى المياه في الخزانات تحكم في المضخة عن بُعد - (WiFi/GSM/Bluetooth) تقليل الفيضانات ونفاد الماء - سعر مناسب وسهولة في التركيب - والاستخدام	-تصميم وتطوير النموذج -برمجة التطبيق وربط الشبكات -تجربة الجهاز ميدانيًا وتحسينه -التسويق والدعم الفني **الموارد الرئيسية** • الة طباعة ثلاثية الابعاد . • العناصر الالكترونية ومحركات وبطاريات. • ESP32 -، وحدات اتصال (WiFi, GSM, Bluetooth) -الحساسات (Ultrasonic) -فريق التطوير (برمجة	• الموردون للعناصر الالكترونية • شركة طباعة اللوحات الالكترونية • شركات التوصيل . • -متاجر بيع مكونات إلكترونية -مزارع، بلديات محلية للتجريب -مطوري تطبيقات -جمعيات بيئية

	(GSM) -نقاط بيع محلية/شركاء -منصات إلكترونية (Ouedkniss)، جوميا....		وهندسة إلكترونية) -البنية التحتية للاختبار والدعم.	

مصادر الإيرادات	**هيكلية التكاليف**
• **.البيع المباشر للجهاز :يمثل المصدر الأساسي، ويتم عبر المتاجر أو الطلبات المسبقة.** • **خدمات التركيب والصيانة :رسوم إضافية تقدم للمستخدمين الذين يفضلون الدعم الميداني.** • **اشتراكات دعم سنوي ذكي :تشمل التحديثات البرمجية والمراقبة عن بُعد.** • **بيع بالجملة للبلديات أو الشركات :عقود تزويد لخزانات متعددة دفعة واحدة.**	• تكلفة التصنيع ..... • تكلفة التصميم ..... • تكلفة الآلات والأجهزة المكتبية ..... • تكلفة كراء المحل ..... • -شراء المكونات الإلكترونية • تكلفة التطوير والتجريب • التسويق والدعم • -الرواتب والمصاريف التشغيلية الثابتة

3.معادلة حساب الإيراد الصافي للوحدة: إيرادات المنتج الواحد = سعر البيع - التكلفة الإجمالية للوحدة مثال تطبيقي: • سعر البيع = 20,000 دج • التكلفة = 10,000 دج • **الربح الصافي لكل وحدة = 10,000 دج**	