

RÉPUBLIQUE ALGÉRIENNE DÉMOCRATIQUE ET POPULAIRE  
MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR ET DE LA  
RECHERCHE SCIENTIFIQUE

ÉCOLE NATIONALE POLYTECHNIQUE



المدرسة الوطنية المتعددة التقنيات  
Ecole Nationale Polytechnique



Département d'Électronique

## Final Year Project

For the degree of State Engineer in Electronics

---

Development of an IoT System for Real-Time Environmental Data  
Acquisition and Control in Agriculture

---

**BOUHOUNALI Abdennour & FOUTIA Abderrahmane**

Presented and defended publicly on (13/06/2024)

### Composition of the Jury:

President:	Pr. Cherif LARBES	ENP
Examiner:	Pr. BOUSBIA SALAH Hicham	ENP
Supervisor:	Pr. ADNANE Mourad	ENP
Supervisor	Dr. BOUHOUN Salah	URAER

ENP 2024



RÉPUBLIQUE ALGÉRIENNE DÉMOCRATIQUE ET POPULAIRE  
MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR ET DE LA  
RECHERCHE SCIENTIFIQUE

ÉCOLE NATIONALE POLYTECHNIQUE



المدرسة الوطنية المتعددة التقنيات  
Ecole Nationale Polytechnique



Département d'Électronique

## Final Year Project

For the degree of State Engineer in Electronics

---

Development of an IoT System for Real-Time Environmental Data  
Acquisition and Control in Agriculture

---

**BOUHOUNALI Abdennour & FOUTIA Abderrahmane**

Presented and defended publicly on (13/06/2024)

### Composition of the Jury:

President:	Pr. Cherif LARBES	ENP
Examiner:	Pr. BOUSBIA SALAH Hicham	ENP
Supervisor:	Pr. ADNANE Mourad	ENP
Supervisor	Dr. BOUHOUN Salah	URAER

ENP 2024

RÉPUBLIQUE ALGÉRIENNE DÉMOCRATIQUE ET POPULAIRE  
MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR ET DE LA  
RECHERCHE SCIENTIFIQUE

ÉCOLE NATIONALE POLYTECHNIQUE



المدرسة الوطنية المتعددة التقنيات  
Ecole Nationale Polytechnique



Département d'Électronique

Mémoire de projet de Fin d'Études

Pour l'obtention du diplôme d'Ingénieur d'État en Électronique

---

Développement d'un système IoT pour l'acquisition et le contrôle  
en temps réel des données environnementales en agriculture

---

**BOUHOUNALI Abdennour & FOUTIA Abderrahmane**

Présenté et soutenu publiquement le (13/06/2024)

**Composition du Jury:**

Président:	Pr. Cherif LARBES	ENP
Examineur:	Pr. BOUSBIA SALAH Hicham	ENP
Promoteur:	Pr. ADNANE Mourad	ENP
Co-Promoteur	Dr. BOUHOUN Salah	URAER

ENP 2024

## ملخص

يهدف نظام أنترنت الأشياء لتوفير مراقبة و تحكم فعال للعوامل المناخية خاصة في تطبيقات الزراعة ويكمن التحدي في تقديم بشكل آني للبيانات مثل درجة الحرارة، الرطوبة و عوامل أخرى من مختلف المواقع.

مشروعنا يسعى للتصدي لتحديات المراقبة عن بعد في الزراعة من خلال حل شامل . يشمل تصميم عقد الإستشعار للحصول على البيانات و عقد محركات التحكم ، مع المحطة الرئيسية لإدارة تدفق البيانات . الموقع الإلكتروني يوفر للمستخدم واجهة مستخدم لمراقبة آنية للبيانات مع قابلية التحكم عن بعد لمحركات التحكم .

كلمات مفتاحية :

نظام أنترنت الأشياء - المراقبة المناخية - الفلاحة - الحصول على البيانات - محركات التحكم - البيانات الآنية - عقد الإستشعار اللاسلكية - منصة الويب - التحكم عن بعد

## Résumé

Le système IoT répond au besoin spécifique d'une surveillance et d'un contrôle efficaces des paramètres météorologiques en agriculture. Le défi réside dans l'acquisition de données en temps réel pour la température, l'humidité et d'autres paramètres pertinents dans divers endroits. Notre projet vise à relever les défis de surveillance agricole grâce à une solution complète. Il implique la conception de nœuds capteurs pour l'acquisition de données et les nœuds d'actionneurs, ainsi qu'une station de base pour gérer le flux de données. La plateforme Web offre aux utilisateurs une interface conviviale pour la surveillance en temps réel des paramètres météorologiques et la possibilité de contrôle à distance des actionneurs.

**Mots-clés :** Système IoT - Surveillance météorologique - agriculture - Acquisition de données - Contrôle d'actionneurs - Données en temps réel - WSN (réseaux de capteurs sans fil) - Plateforme Web - Contrôle à distance.

## Abstract

The IoT system addresses the need for efficient monitoring and control of meteorological parameters, specifically for agricultural applications. The challenge lies in acquiring real-time data for temperature, humidity, and other relevant parameters across various locations. Our project aims to tackle the monitoring challenges in agriculture through a comprehensive solution. It involves designing sensor nodes for data acquisition and actuator nodes for control, along with a base station to manage data flow. The web platform provides users with a user-friendly interface for real-time monitoring of meteorological parameters and the ability to remotely control the actuators.

**Keywords:** IoT System - Meteorological Monitoring - Agriculture - Data Acquisition - Actuator Control - Real-time Data - WSN - Web Platform - Remote Control.

## Dedication

To those who have given meaning to my journey,

It is with deep gratitude that I dedicate this final project of my studies to all the individuals who have contributed to its completion. The path to this moment has not been without challenges, but thanks to your support, love, and encouragement, I have been able to overcome each step with determination and perseverance.

To my beloved family, to my dear parents, my unwavering pillars who have consistently supported and encouraged my perseverance throughout these years of study. Thank you from the bottom of my heart for your support and exceptional contribution.

To my cohort, "Classmate," whose companionship will forever remain etched in my memory, and with whom I've journeyed through these intense three years, brimming with a spectrum of emotions, from moments of joy to challenges. I am profoundly grateful for the invaluable contributions each of you has made. To my friends and my schoolmates, I want to thank you, especially: Rahal, Abdelbaki, Aya, Abderezak, Said, Walid, Ferial, Wissal, Mehdi, Loubna, Karine, Mehdi, Sabrina, Bouchra, Soumia, Imed, Amira, Yousra, Mahmoud, Oussama, Serine, Ilhem, Sarah, and Sonia for your valuable advice and support.

*Abderrahmane.*

## **Dedication**

To those who have given meaning to my journey,

It is with deep gratitude that I dedicate this thesis to all who contributed to its completion.

To my beloved parents, whose memories and spirits inspire and guide me every day. Though you are no longer with us, your love and wisdom have been my constant companions.

To my brothers and uncles, for your steadfast support and encouragement. To my friends, whose friendship and understanding have been invaluable. To my teachers, whose guidance and knowledge have been instrumental in my academic growth.

Thank you all for being my pillars of strength.

*Abdenmour.*

## Acknowledgments

This thesis marks a decisive turning point in our lives, the culmination of a long educational journey, and the beginning of new adventures. It is the result of our hard work, realized through our efforts and the invaluable assistance of those around us. Therefore, we would like to thank everyone who has contributed, directly or indirectly, to the success of this work.

Of course, we would like to express our sincere gratitude to our supervisors, Dr. Salah BOUHOUN, Prof. Mourad ADNANE, and the entire research team at UARER, as well as the NABTAKIR Association, who supported us throughout this work. We thank them for their patience, motivation, and unparalleled dedication to making this work valuable. They have always been available when we needed them throughout this project. We could not have asked for better guidance. We owe them immense and infinite gratitude.

In conclusion, we would like to thank all our teachers in the electronics department who have supported us during these three years of study and until the completion of this project.

*Abderrahmane and Abdennour*



# Contents

List of Tables

List of Figures

List of Acronyms

<b>General Introduction</b>	<b>18</b>
<b>1 General Synoptics: Context and State of the Art</b>	<b>20</b>
1.1 Introduction . . . . .	20
1.2 Historical Context . . . . .	20
1.3 Problematic of Our System . . . . .	21
1.4 General Synoptic of Our Solution . . . . .	21
1.5 State of the Art . . . . .	22
1.6 Conclusion . . . . .	24
<b>2 Material and Technology Used</b>	<b>25</b>
2.1 Component . . . . .	25
2.1.1 STM32 Series F103C8T6 . . . . .	25
2.1.2 Arduino Nano . . . . .	26
2.1.3 ESP32 . . . . .	27
2.1.4 Sensors Used : . . . . .	28
2.1.4.1 SHT31 Sensor . . . . .	28
2.1.4.2 DHT22 Sensor . . . . .	29
2.1.4.3 Calibration . . . . .	30
2.1.4.4 Resistive Soil Sensor . . . . .	32

---

2.1.4.5	Capacitive Soil Sensor . . . . .	33
2.1.4.6	Calibration . . . . .	34
2.1.4.7	Gaz Sensor . . . . .	36
2.1.4.8	Solar Irradiance Sensor . . . . .	37
2.1.4.9	Calibration . . . . .	38
2.1.4.10	Ambient light Sensor . . . . .	39
2.1.5	Actuator Used . . . . .	40
2.1.5.1	Solenoid Valve . . . . .	41
2.1.5.2	Micro Diaphragm Pump . . . . .	41
2.1.5.3	DC Submersible Water Pump . . . . .	42
2.1.6	DC to DC converter . . . . .	43
2.1.7	GSM Module . . . . .	45
2.1.8	SD Card Module . . . . .	46
2.1.9	Relay Module . . . . .	47
2.1.10	Specifications . . . . .	47
2.1.10.1	Relay Description . . . . .	48
2.2	Technology . . . . .	48
2.2.1	Wireless technology . . . . .	48
2.2.2	LoRa . . . . .	49
2.2.3	LoRa Network Architecture . . . . .	49
2.2.4	PHP: . . . . .	52
2.2.5	Laravel Framework: . . . . .	53
2.2.6	MySQL: . . . . .	54
<b>3</b>	<b>Proposed Solutin</b>	<b>55</b>
3.1	Platform . . . . .	55
3.1.1	Introduction . . . . .	55
3.1.2	Web Application Interface . . . . .	58
3.1.3	Login Page . . . . .	58
3.1.3.1	Dashboard Page . . . . .	58
3.1.3.2	Data Page . . . . .	59

---

---

3.1.3.3	Control Parameters Interface . . . . .	60
3.1.3.4	Temperature Control . . . . .	60
3.1.3.5	Humidity Control . . . . .	61
3.1.3.6	CO <sub>2</sub> Control . . . . .	61
3.1.3.7	Soil Humidity Control . . . . .	61
3.1.3.8	Data Acquisition Frequency . . . . .	61
3.1.3.9	Example Parameter Configuration . . . . .	61
3.1.3.10	Control Page . . . . .	62
3.1.3.11	Responsive View . . . . .	63
3.1.4	Security Measures . . . . .	64
3.2	Base Station . . . . .	64
3.2.1	Introduction . . . . .	64
3.2.2	Base Station Architecture . . . . .	65
3.2.3	Description of the Base Station Components . . . . .	65
3.2.3.1	ESP32 . . . . .	65
3.2.3.2	Memory . . . . .	65
3.2.3.3	LoRa Module . . . . .	65
3.2.3.4	GSM SIM800L Module . . . . .	66
3.2.3.5	Wi-Fi . . . . .	66
3.2.3.6	LCD . . . . .	66
3.2.4	System Workflow . . . . .	66
3.2.4.1	Key Features . . . . .	67
3.2.5	Base Station interface . . . . .	67
3.2.6	PCB Design . . . . .	67
3.3	Node Architecture . . . . .	69
3.3.1	Introduction . . . . .	69
3.3.2	Type of nodes . . . . .	70
3.3.2.1	Sensor nodes . . . . .	70
3.3.2.2	Sensors Interface . . . . .	71
3.3.2.3	Node Design . . . . .	71

---

3.3.3	Actuator nodes . . . . .	72
3.3.3.1	Actuator Interface . . . . .	73
3.3.3.2	PCB design . . . . .	73
3.4	Power management . . . . .	75
3.5	Conclusion . . . . .	77
<b>4</b>	<b>Communication Protocol</b>	<b>79</b>
4.1	Introduction . . . . .	79
4.2	Communication within the System . . . . .	79
4.2.1	SPI Protocol . . . . .	79
4.2.1.1	SPI Communication Setup . . . . .	80
4.2.1.2	Data Flow . . . . .	80
4.2.2	I2C Protocol . . . . .	81
4.2.2.1	I2C Softawar . . . . .	81
4.3	Genaral communication . . . . .	82
4.3.1	Initialization Phase . . . . .	82
4.3.2	Main Operation Phase . . . . .	83
4.3.3	Recurrent Operations . . . . .	83
4.3.3.1	How Command sent to Sensor Node: . . . . .	83
4.3.4	Data Handling . . . . .	85
4.3.4.1	Message Format . . . . .	85
4.3.5	Manual and Update Operations . . . . .	86
4.4	Conclusion . . . . .	87
<b>5</b>	<b>Results, Challenges and Limitations</b>	<b>88</b>
5.1	Verification and Validation . . . . .	88
5.1.1	Sensor Node Validation . . . . .	89
5.1.2	Actuator Node Validation . . . . .	90
5.2	Time Controlling Issues . . . . .	91
5.3	LoRa Range Limitations . . . . .	93
5.3.1	Static Nature of the Monitoring Platform . . . . .	93

5.4	Flash Memory saturation . . . . .	94
5.4.1	Proposed solutions . . . . .	94
5.5	Future Work . . . . .	94
5.6	Conclusion . . . . .	95
<b>6</b>	<b>General Conclusion</b>	<b>96</b>
<b>7</b>	<b>Annex</b>	<b>97</b>
	<b>Bibliography</b>	<b>99</b>

# List of Tables

- 2.1 Comparison between SHT31 and DHT22 sensors . . . . . 30
- 2.2 Calibration Result . . . . . 39
- 2.3 Comparison of LPWAN Technologies Characteristics . . . . . 52
  
- 3.1 List of Components . . . . . 67
- 3.2 List of sensors used in the system . . . . . 71
- 3.3 List of actuators used in the system . . . . . 73

# List of Figures

- 1.1 General synoptic of the System . . . . . 21
  
- 2.1 STM32 BLUE PILL Pinout . . . . . 25
- 2.2 Arduino Nano Pinout . . . . . 26
- 2.3 ESP32 Pinout . . . . . 27
- 2.4 SHT31 Sensor . . . . . 28
- 2.5 DHT22 Sensor . . . . . 29
- 2.6 Data recording inside the greenhouse . . . . . 30
- 2.7 Environment metre . . . . . 31
- 2.8 The plot of the data from the Temperature sensor . . . . . 31
- 2.9 The coefficients of the regression functions. . . . . 31
- 2.10 The plot of the data from the Humidity sensor . . . . . 32
- 2.11 Resistive soil sensor . . . . . 32
- 2.12 Capacitive soil sensor . . . . . 33
- 2.13 Soil Sensor Module Schematic . . . . . 34
- 2.14 Weight of the soil before drying . . . . . 34
- 2.15 Drying the soil in the oven . . . . . 34
- 2.16 Weight of the soil after drying . . . . . 35
- 2.17 Starting the calibration . . . . . 35
- 2.18 The plot of the data Humidity from Soil sensor . . . . . 36
- 2.19 Gaz Sensor MQ-135 . . . . . 36
- 2.20 Solar Irradiance Sensor . . . . . 37
- 2.21 Calibration of Solar Irradiance Sensor . . . . . 38
- 2.22 Irradiation Calibration Graphs . . . . . 39

---

2.23 Ambient light Sensor . . . . .	39
2.24 Modbus communication . . . . .	40
2.25 Solenoid valve Actuator . . . . .	41
2.26 Micro Diaphragm Pump Actuator . . . . .	42
2.27 DC Submersible Water Pump Actuator . . . . .	43
2.28 DC to DC buck converter . . . . .	43
2.29 Buck converter representation . . . . .	44
2.30 GSM Module Pinout . . . . .	45
2.31 SD Card Module . . . . .	46
2.32 2 Channel Relay Module . . . . .	47
2.33 Relay Functio Mode . . . . .	48
2.34 LoRa Connectivity Schematic . . . . .	49
2.35 LoRa Sensor . . . . .	50
2.36 Wireless Technology used in IoT . . . . .	51
2.37 PHP logo . . . . .	52
2.38 Laravel logo . . . . .	53
2.39 MySQL logo . . . . .	54
3.1 Larevl Architecture . . . . .	56
3.2 Platform Architecture . . . . .	57
3.3 Login Page of Agritech-DZ Platform . . . . .	58
3.4 Dashboard Page of Agritech-DZ Platform . . . . .	59
3.5 Data Page of Agritech-DZ Platform . . . . .	60
3.6 Control Parameters Page of Agritech-DZ Platform . . . . .	60
3.7 Control Page of Agritech-DZ Platform . . . . .	62
3.8 Responsive Page of Agritech-DZ Platform . . . . .	63
3.9 Base Station Architecture . . . . .	65
3.10 Base Station Circuit . . . . .	68
3.11 Solder PCB . . . . .	69
3.12 Actuator and Sensor nodes . . . . .	70
3.13 Power circuit . . . . .	70



3.14	Sensor Node . . . . .	72
3.15	Actuator Node Architecture . . . . .	73
3.16	Actuators PCB design . . . . .	74
3.17	soldered PCB . . . . .	74
3.18	Actuator Power diagram block . . . . .	75
3.19	Solar System Diagram . . . . .	76
3.20	grid Power . . . . .	77
3.21	Solar Power . . . . .	77
4.1	SPI Communication Protocol . . . . .	80
4.2	SPI connection between the microcontroller,SD card, and LoRa Module . . . . .	80
4.3	I2C Communication Protocol . . . . .	81
4.4	Communication Protocol . . . . .	82
4.5	LORA PACKET FORMAT [23] . . . . .	83
4.6	Sensor Message Format . . . . .	86
4.7	Actuator Message Format . . . . .	86
4.8	Communication Protocol . . . . .	86
5.1	Data Collecting the Platform . . . . .	89
5.2	Actuator Stat . . . . .	90
5.3	Controlling the Actuator from the Platform . . . . .	91
7.1	Steps to establish a node . . . . .	97

---

# List of Acronyms

- **IOT**: Internet of Things
- **WiFi**: Wireless Fidelity
- **I2C**: Inter-Integrated Circuit
- **SPI**: Serial Peripheral Interface
- **SDA**: Serial Data
- **SCL**: Serial Clock
- **USB**: Universal Serial Bus
- **CAN**: Controller Area Network
- **ADC**: Analog-to-Digital Converter
- **DAC**: Digital-to-Analog Converter
- **MCU**: Microcontroller Unit
- **GPIO**: General Purpose Input Output
- **Rx**: Receive
- **Tx**: Transmit
- **IP**: Internet Protocol
- **LUX**: Unité d'éclairement lumineux (Unit of illuminance)
- **LoRa**: Long Range
- **PHP**: Hypertext Preprocessor
- **SQL**: Structured Query Language
- **API**: Application Programming Interface
- **DC**: Direct Current
- **PPM**: Parts per million
- **PMOS**: P-Channel MOS (Metal-Oxide Semiconductor)
- **NMOS**: N-Channel MOS (Metal-Oxide Semiconductor)

- **MOSI**: Master Out Slave In
  - **MISO**: Master In Slave Out
  - **CS**: Chip Select
  - **HTTP**: Hypertext Transfer Protocol
  - **PCB**: Printed Circuit Board
  - **PFM**: Pulse Frequency Modulation
  - **PWM**: Pulse Width Modulation
  - **RS485**: Recommended Standard 485
-

# General Introduction

In 2050, a study conducted by the FAO projects a staggering 70% surge in global food demand, based on the High-Level Expert Forum titled 'How to Feed the World in 2050.' However, this ambitious target faces formidable challenges, including the impacts of climate change, a rapidly increasing population, and the shrinking availability of arable land. Addressing these challenges is crucial for ensuring food security on a global scale. Consequently, adopting advanced agricultural technologies, particularly greenhouses, and vertical farming, is becoming indispensable to fortify both food production and quality.

The demand for sophisticated systems is pressing in this rapidly evolving agricultural landscape. Farmers require tools that enhance production, optimize costs, and simplify farm management. Our project, focusing on developing an IoT system, stands at the forefront of meeting these needs. The significance of this project in the market is underscored by the transformative impact it can have on agricultural practices. With the advent of IoT technology, farmers can make informed decisions based on real-time data, leading to improved crop yields and resource utilization. This contributes to meeting the growing demand for food and enhances the overall efficiency and sustainability of agricultural operations.

In a market where efficiency, sustainability, and precision agriculture are becoming paramount, our project addresses a critical need. It aligns with the evolving priorities of modern agriculture, making it a significant and timely contribution to the market. As the global population continues to rise, and environmental challenges persist, the importance of innovative solutions like our IoT system cannot be overstated, positioning it as a key player in the future of sustainable and efficient food production.

This thesis is structured into 5 chapters as follows :

Pour cela, plusieurs travaux se sont portés dans le but de pouvoir classer des patients atteints d'Alzheimer de patients sains ou encore de parvenir à classer les différents patients suivant leurs niveaux de démence, travaux remontant aux années 80, jusqu'à récemment avec, entre autres, les travaux de Syed et al. [?], N. Khan et al. [?], H. Nawaz et al. [?] et A. Loddo [?] atteignant respectivement 98.6%, 99.36%, 99.21% et 97.71% de précision sur les bases de données qu'ils ont traités.

**Chapitre 1 :** Presents the scientific context of our subject by tracing the impact of meteorological parameters in monitoring agriculture. It outlines the stages constituting a general description of the system and its different components, including the stages they contain. Additionally, it highlights the work carried out in this area.

**Chapitre 2 :** In Chapter 2, we compare various sensors and actuators, providing an overview of the materials and technologies employed in the study.

**Chapitre 3 :** This chapter presents our proposed solution, which integrates advanced IoT technologies to address the challenges of modern agriculture. By enhancing data-driven decision-making and enabling proactive management of agricultural environments, our

system aims to optimize productivity and promote sustainability in farming practices.

**Chapter 4:** In Chapter 6, we delve into the communication protocol within the system, exploring how data flows between its various components.

**Chapter 5:** We address the challenges encountered throughout our work, proposing some solutions, while also elucidating the limitations of the system.

Finally, we conclude this thesis with a general conclusion and some perspectives.

## About the Research Centre

The Unit for Renewable Energy in Arid Regions (URAER), attached to the Renewable Energy Development Center (CDER), is located in the beautiful city of Ghardaïa (600 km from Algiers) and just 1 kilometer from the airport. The significant investments in training and research in the field of renewable energies, such as the development of specialized laboratory equipment, have enabled Algeria to become a trustworthy and experienced partner in this field. The creation of URAER has further confirmed Algeria's role in promoting and developing renewable energies. URAER's ambition is to become an international platform for experimentation and a communication hub for regional achievements in renewable energies. Through its research programs, URAER aims to contribute to the development and mastery of renewable energy technologies. The unit's human potential supports national research and training efforts by collaborating with universities and other research centers, and by offering high-quality training in renewable energies, from the level of mastery up to specialized post-graduate levels.



# Chapter 1

## General Synoptics: Context and State of the Art

### 1.1 Introduction

In agriculture, it's vital to keep an eye on factors like soil moisture, temperature, and weather to ensure crops grow well. Data systems help with this by keeping track of these factors in real-time. This helps farmers make smart choices and improve their farming methods, leading to better crop yields and more sustainable practices. These systems are important in many industries and gather data from different sources, like sensors and software. In agriculture, they help farmers by keeping tabs on weather conditions, allowing them to use advanced technology and data analysis to make informed decisions, ultimately leading to better crop yields and more sustainable farming.

### 1.2 Historical Context

The evolution of data acquisition systems dates back to the mid-20th century when advancements in electronics and computing enabled the development of more sophisticated systems. Initially used in industrial settings for process control and monitoring, these systems gradually found applications in diverse fields like agriculture, environmental monitoring, health-care, and research. Over time, with the advent of microprocessors and digital technology, data acquisition systems became more compact, efficient, and capable of handling complex data streams. In agricultural contexts, the integration of meteorological parameter monitoring into these systems revolutionized farming practices by providing real-time insights into environmental conditions crucial for crop growth. Today, modern data acquisition systems leverage cutting-edge technologies like IoT, AI, and cloud computing to offer advanced functionalities such as remote monitoring, predictive analytics, and seamless data integration. These systems continue to play a pivotal role in optimizing agricultural processes, ensuring sustainable practices, and maximizing productivity in the ever-evolving landscape of farming and environmental management.

### 1.3 Problematic of Our System

The design and implementation of a data acquisition system to monitor environmental parameters in agricultural applications present several challenges. Ensuring the accuracy and reliability of collected meteorological data, properly selecting and calibrating sensors, establishing reliable data transmission methods in remote areas, and providing continuous power while minimizing energy consumption, sending farm data with no internet connection are the main challenges one faces. Additionally, effective data management, analysis for extracting useful information, environmental durability of components, and seamless integration with existing agricultural practices are crucial. For these reasons, our work proposes a method for acquiring various parameters with enhanced precision and in real time. Additionally, it offers the capability of Connecting isolated areas from the internet, thereby reducing energy consumption.

### 1.4 General Synoptic of Our Solution

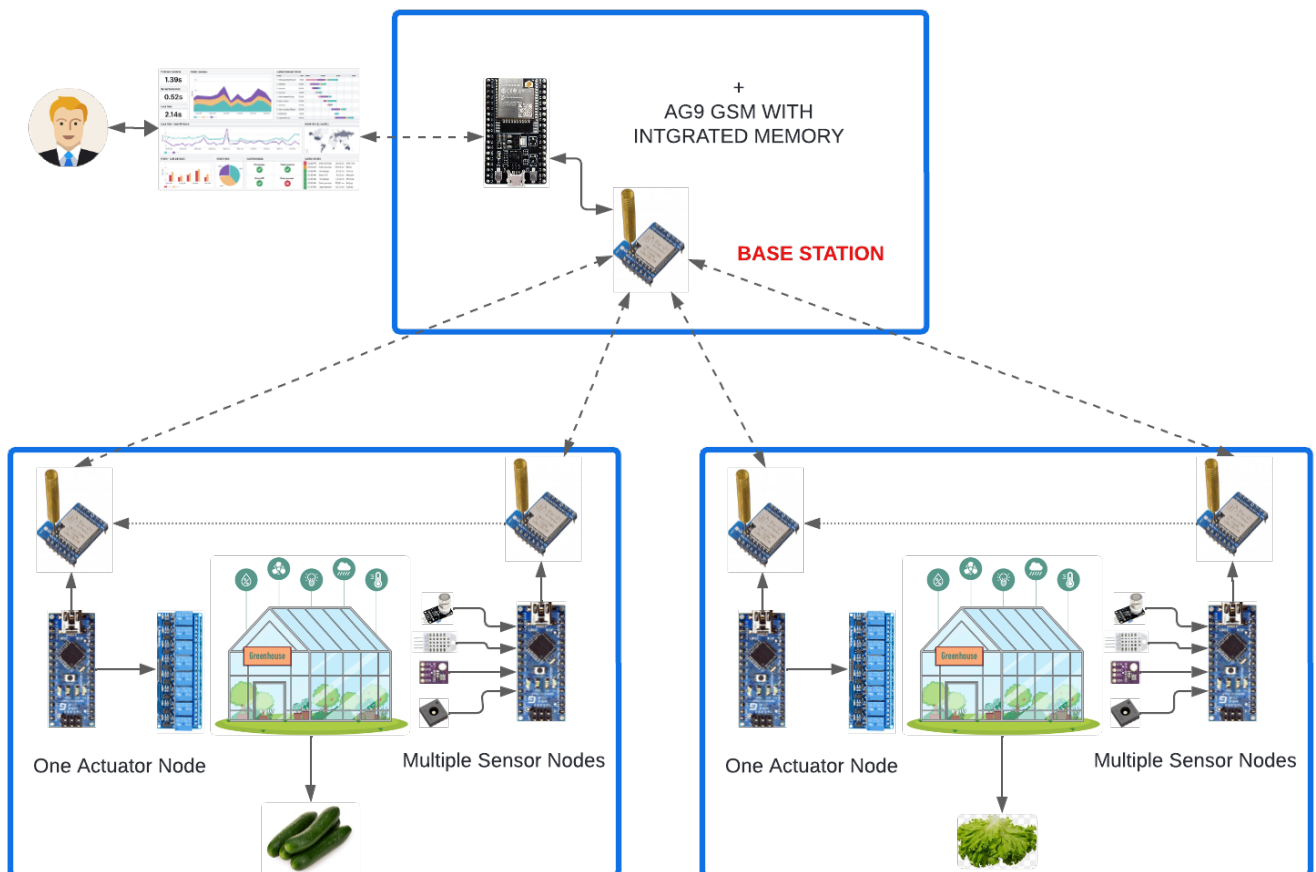


Figure 1.1: General synoptic of the System

The diagram illustrates an IoT system designed for monitoring and controlling a greenhouse environment, with a focus on optimizing the cultivation of specific products. The system is centered around an ESP32 Wi-Fi Microcontroller, which acts as the base station unit to coordinate with different nodes, including actuators and sensors. The ESP32 connects to the

platform using Wi-Fi and communicates with the nodes using LoRa technology. Additionally, multiple sensor nodes are placed in different sections of the greenhouse, along with one actuator node.

Remote monitoring and control functionalities are provided through a Web application. Additionally, the system architecture follows a three-layered approach:

**- Node Layer:**

- Individual nodes distributed throughout the greenhouse.
- Nodes feature sensors for environmental data collection and actuators for control.
- Responsible for local data collection and control actions based on predefined rules.
- Each node contains a microcontroller connected to different sensors for data acquisition and processing, and it's connected with the base station using a LoRa module.

**- Base Station Layer:**

- Central hub coordinating communication with nodes.
- Includes a central processing unit for managing data transfer
- Collects data from the nodes and transmits it to the platform, retrieves the status of the platform's actuators, and relays it back to the nodes.

**- Platform/Connectivity Layer:**

- Provides infrastructure for data transmission, storage, and remote access.
- Facilitates communication between the base station and external devices, supporting remote monitoring and control.
- Supports data aggregation, visualization, and analysis for optimizing greenhouse operations.
- Includes an authentication system for admin users and ensure the security of data transfer from the devices.

This three-layered architecture enhances system organization, enabling efficient communication, data management, and control of the greenhouse environment to improve productivity and crop management.

## 1.5 State of the Art

The integration of data acquisition systems and monitoring systems in smart agriculture has led to remarkable improvements in resource management, crop yield, and environmental sustainability, Yongchao Song and all [1]. Recent statistics indicate that farms implementing advanced data acquisition technologies have witnessed up to a 30% increase in crop productivity and a notable reduction of up to 50% in water usage, Abdennabi Morchid and all [2]. Additionally, studies have revealed a significant 20% decrease in fertilizer and pesticide usage, contributing to environmental preservation. Real-time data analysis facilitated by cloud



platforms has further empowered farmers to make timely decisions, resulting in a 25% decrease in crop loss due to adverse weather conditions and pest infestations. These outcomes highlight the pivotal role of data acquisition and monitoring systems in modern agriculture, not only in boosting efficiency and productivity but also in advancing sustainable farming practices [1][2][3].

Furthermore, monitoring systems in agriculture leverage sensors and IoT technology to gather real-time data on environmental conditions, crop health, and resource utilization. Key applications of these systems include optimizing irrigation, precise fertilizer administration, early detection of crop diseases, yield monitoring, and climate condition monitoring. This integration of IoT technology has the potential to revolutionize farming practices, increasing efficiency, reducing waste, promoting sustainability, and improving product quality through real-time data insights and automation, M. Mohammad El-Basioni and all [4].

The most critical parameters in agriculture, as highlighted by various studies and techniques, revolve around temperature, humidity, CO<sub>2</sub> concentration, and pH level. These parameters play a vital role in ensuring optimal conditions for crops and livestock. Sensors tailored for agriculture are adept at detecting a wide array of parameters, including soil NPK (Azote (N), Phosphor (P) et Potassium (K)) levels, moisture content, nitrate levels, light intensity, weather conditions, water level, plant disease detection, and more. However, the focus on temperature, humidity, CO<sub>2</sub> concentration, and pH level underscores their significance in maintaining a conducive environment for agricultural activity [4].

For instance, The technique used for data acquisition in agricultural monitoring systems often involves wireless sensor networks (WSNs) and the Internet of Things (IoT), Alexander T. Demetillo and all [5]. WSNs consist of spatially distributed autonomous sensors that monitor physical or environmental conditions, such as temperature, humidity, and soil moisture, and transmit this data wirelessly to a central hub or gateway. The IoT enables the integration of these sensor networks with the Internet, allowing for real-time data analysis and remote monitoring.

For example, in an agricultural monitoring system, sensors can be placed in fields to collect data on soil moisture, temperature, and other environmental factors [5], [6], Joao Tagaio [7]. This data is then transmitted wirelessly to a central hub, where it can be analyzed to optimize irrigation, fertilization, and pest control. The IoT enables farmers to remotely monitor their crops and take action based on real-time data, improving efficiency and reducing waste.

In addition to WSNs and the IoT, other wireless communication technologies, such as Zigbee and GSM, can be used for data acquisition in agricultural monitoring systems. Zigbee is a low-power wireless communication protocol that is commonly used for short-range communication between sensors and a central hub. GSM is a cellular communication protocol that can be used for long-range communication between sensors and a central hub, Haider Mahmood Jawad and all [8]. ,

Overall, the use of wireless communication technologies and the IoT in agricultural monitoring systems has the potential to revolutionize farming practices, increasing efficiency, reducing waste, promoting sustainability, and improving product quality through real-time data insights and automation.

## 1.6 Conclusion

The chapter underscores the critical importance of agricultural data acquisition systems in monitoring and analyzing environmental parameters in real-time. From their inception to their modern integration with IoT and cloud computing, these systems have revolutionized agricultural practices. While challenges persist, such as data accuracy and energy consumption, ongoing research and innovation aim to address them. Illustrated by a cloud-based greenhouse management system, these technologies optimize communication, data management, and crop control. With substantial growth projected in the global smart agriculture market, data acquisition systems continue to drive agricultural innovation towards efficiency and sustainability.

# Chapter 2

## Material and Technology Used

### 2.1 Component

#### 2.1.1 STM32 Series F103C8T6

The STM32 Blue Pill is an ARM Cortex-M3 development board clocked at 72 MHz, offering sufficient computational power to process sensor data, perform complex calculations, and execute real-time control algorithms. Its high frequency enables fast response times in agricultural applications. It supports multiple communication protocols, including UART, SPI, and I2C, facilitating the collection and transmission of sensor data. Its ability to execute tasks in real time makes it ideal for agricultural applications requiring an instant response to environmental changes.

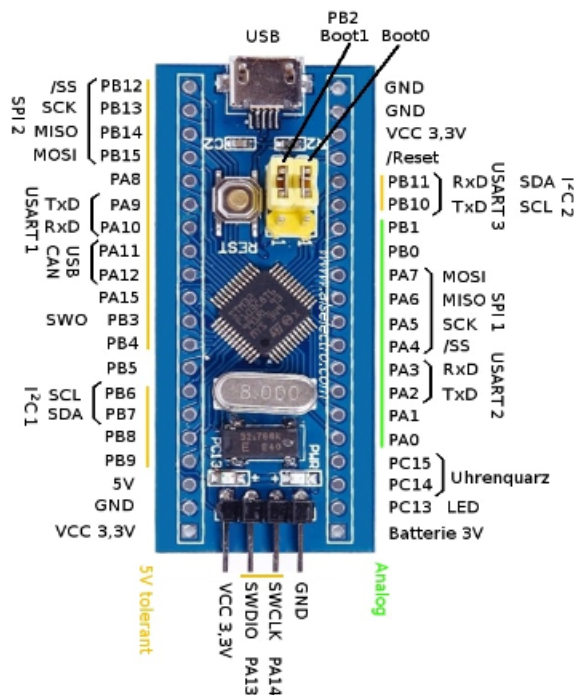


Figure 2.1: STM32 BLUE PILL Pinout

Features:

Parameter	Meaning
Architecture	32-bit ARM Cortex M3
Operating Voltage	2.7V to 3.6V
CPU Frequency	72 MHz
Number of GPIO pins	37
Number of PWM pins	12
Analog Input Pins	10 (12-bit resolution)
I2C Peripherals	2
SPI Peripherals	2
CAN 2.0 Peripheral	1
Timers	3(16-bit), 1
Flash Memory	32KB
RAM	20kB

### 2.1.2 Arduino Nano

The Arduino Nano is a compact, breadboard-friendly microcontroller board based on the ATmega328P, offering versatile functionality for a wide range of DIY electronics projects and prototyping.

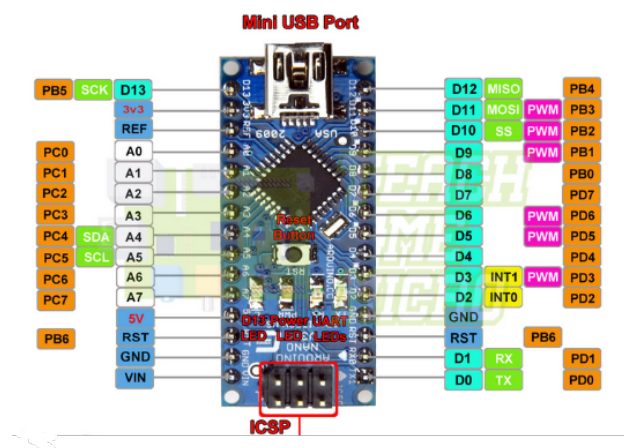


Figure 2.2: Arduino Nano Pinout

### Specifications

- **Microcontroller:** ATmega328P
- **Operating Voltage:** 5V
- **Input Voltage (recommended):** 7-12V
- **Input Voltage (limit):** 6-20V

- **Digital I/O Pins:** 14 (of which 6 provide PWM output)
- **Analog Input Pins:** 8
- **DC Current per I/O Pin:** 40 mA
- **DC Current for 3.3V Pin:** 50 mA
- **Flash Memory:** 32 KB (of which 2 KB used by the bootloader)
- **SRAM:** 2 KB
- **EEPROM:** 1 KB
- **Clock Speed:** 16 MHz

### 2.1.3 ESP32

The ESP32 is a low-cost, low-power system on a chip (SoC) with integrated Wi-Fi and Bluetooth capabilities. It is commonly used for IoT and embedded applications due to its high performance and versatility.

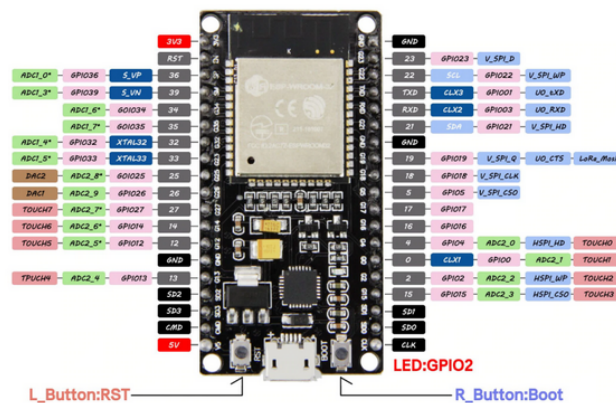


Figure 2.3: ESP32 Pinout

### Specification

- Input Voltage: 5VDC
- Operating Voltage: 2.3 3.6V
- Operating Current: 80mA
- Clock Frequency: 80 240MHz
- Flash Memory: 2MB
- Data Rate: 54Mbps
- SRAM Memory: 512KB

## 2.1.4 Sensors Used :

The choice of the right sensors for the project took place over a relatively long period, and this for various reasons.

- The large number of available sensors to compare and test was one of the reasons behind
- Ensuring the reliability and accuracy of the system.

### 2.1.4.1 SHT31 Sensor

In this section, we will begin discussing the first meteorological parameter, which is considered one of the most interesting parameters "Temperature and Humidity" inside the room. We will start by introducing the SHT31 sensor.



Figure 2.4: SHT31 Sensor

The SHT31 sensor, produced by Sensirion, is renowned for its accuracy and dependability in measuring temperature and humidity. It employs a capacitive humidity sensor and band-gap temperature sensor to ensure precise readings suitable for environmental monitoring and HVAC systems. Its broad operating voltage range and low power consumption contribute to fast response times.

**HVAC** : stands for Heating, Ventilation, and Air Conditioning.

#### **Specification**

- Operating Voltage: 3.3~ 5V.
- Operating Current: < 1.5mA.
- Humidity Detection Range: 0%RH ~100%RH .
- Humidity Accuracy:  $\pm 2\%$  RH .
- Temperature Detection Range:  $-40^{\circ}\text{C} \sim 125^{\circ}\text{C}$ .

- Temperature Accuracy:  $\pm 0.2^{\circ}\text{C}$ .
- Communication: I2C .
- Cable Length: about 1m .

#### 2.1.4.2 DHT22 Sensor

The DHT22, also known as RHT03, is a reliable temperature and humidity sensor manufactured by Aosong. It is known for its accuracy and stability in measuring environmental parameters. Equipped with a capacitive humidity sensor and a thermistor, it provides precise readings for indoor and outdoor conditions. With its wide operating voltage range and low power consumption.



Figure 2.5: DHT22 Sensor

#### Specification :

- Operating Voltage: 3 ~ 5V.
- Max Current During Measuring = 2.5mA.
- Humidity Detection Range: 0%RH ~100%RH .
- Humidity Accuracy:  $\pm 2\%RH \sim \pm 5\%RH$  ..
- Temperature Detection Range:  $-40^{\circ}\text{C} \sim 125^{\circ}\text{C}$ .
- Temperature Accuracy:  $\pm 0.5^{\circ}\text{C}$ .

For agricultural applications, the SHT31 is the preferred choice due to its higher accuracy and reliability in measuring temperature and humidity. This ensures precise monitoring of environmental conditions crucial for plant growth and health. Additionally, its compatibility with the I2C protocol simplifies integration into agricultural monitoring systems, making it the best option for such applications.

Table 2.1: Comparison between SHT31 and DHT22 sensors

Feature	SHT31	DHT22
Communication Protocol	I2C	One wire
Input Voltage	2.4V-5.5V	3.3-5V
Temperature Measurement	Yes	Yes
Humidity Measurement	Yes	Yes
Temperature Measurement Accuracy	$\pm 0.2^{\circ}\text{C}$	$\pm 0.5^{\circ}\text{C}$
Humidity Measurement Accuracy	$\pm 2\% \text{RH}$	$\pm 2-5\% \text{RH}$

### 2.1.4.3 Calibration

To calibrate the temperature sensor, we placed the different sensors to confirm the repeatability of our sensors. For this, we used an SHT31 temperature sensor and a DHT22 temperature sensor. We added an SD card to the system for data recording. As a reference, we used an environmental meter device with a temperature probe. Additionally, we incorporated a thermocouple sensor connected to the device to confirm accuracy further.



Figure 2.6: Data recording inside the greenhouse





Figure 2.7: Environment metre

After 1 hour of collecting data for all the sensors and saving them in a CSV file, we found the function that gives the temperature values as a function of time. The figures below show the results we obtained.

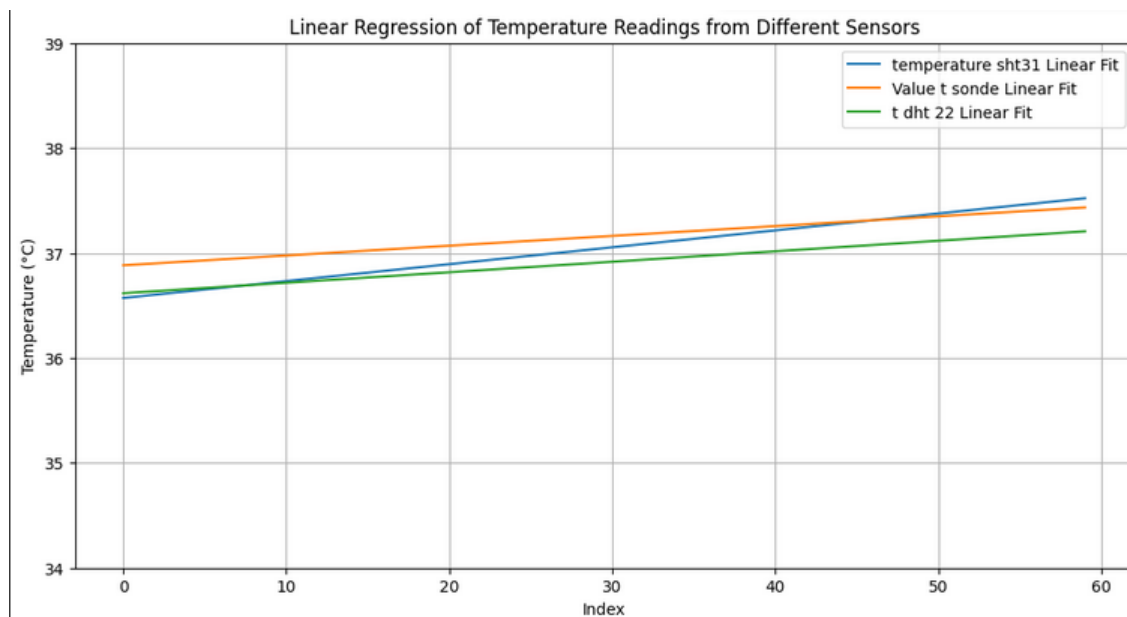


Figure 2.8: The plot of the data from the Temperature sensor

Here are the prediction values of the slope after using a simple Python code.

```
Prédiction de la température pour l'index 70 (temperature sht31) : 1.07 °C
Prédiction de la température pour l'index 70 (Value t sonde) : 0.59 °C
Prédiction de la température pour l'index 70 (t dht 22) : 0.55 °C
```

Figure 2.9: The coefficients of the regression functions.

For the calibration of humidity, this is the calibration curve figure 2.10. In conclusion, we can say that, as we can see, the graphs of the two sensors are relatively identical, confirming the repeatability of our sensors.

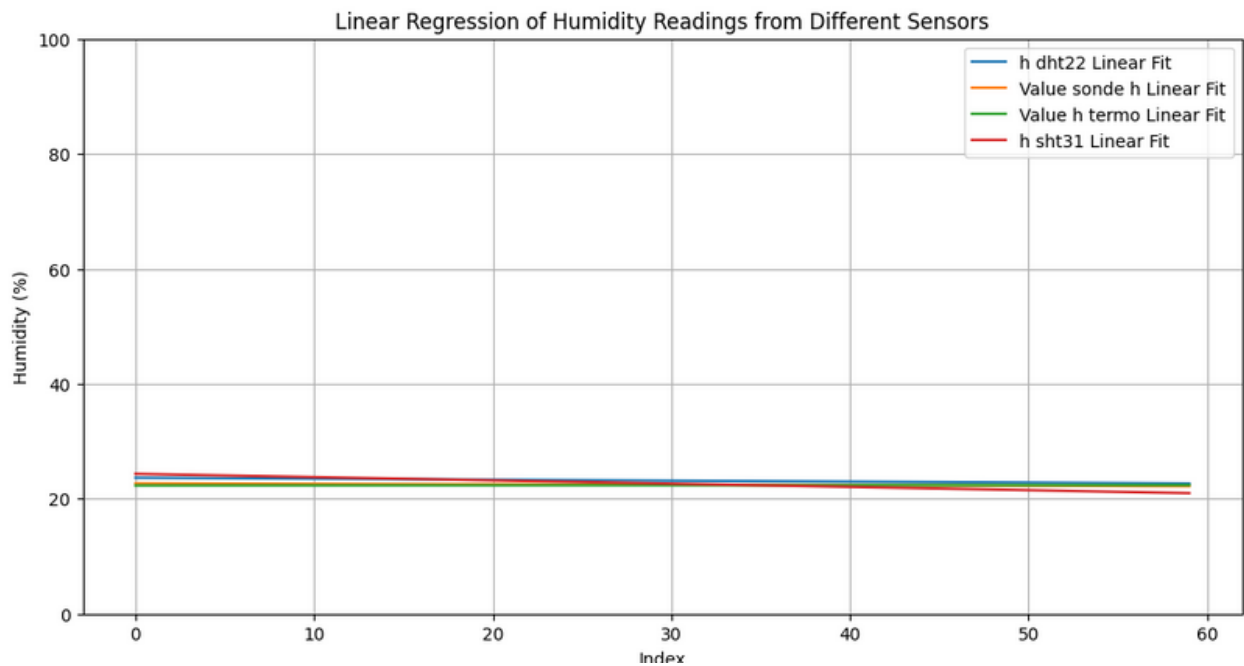


Figure 2.10: The plot of the data from the Humidity sensor .

#### 2.1.4.4 Resistive Soil Sensor

After soil moisture, the next consideration is the choice between resistive and capacitive technologies. Resistive technology, prone to oxidation like all metals, has a short lifespan, sometimes lasting only a few days. Consequently, we opted for the 'Soil Moisture Sensor v1.2,' as it represents the most readily available and cost-effective capacitive soil moisture sensor on the market.

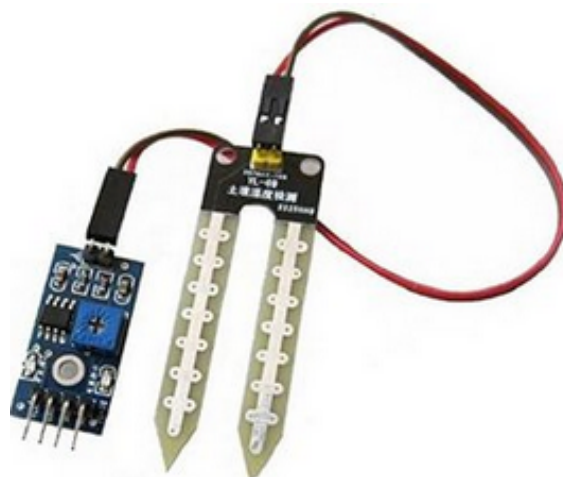


Figure 2.11: Resistive soil sensor

#### Specifications:

- Operating Voltage: 3.3V - 5V.
- Small Size: 1.6cm x 3cm .

- Dual Output Mode.
- Adjustable Sensitivity with Integrated Potentiometer on Dashboard.
- Power Indicator (Red LED) and Digital Output Indicator (Green LED).

**Theory of Operation :**

The soil moisture sensor has two conducting plates. The first plate is connected to the +5Volt supply through a series resistance of  $10K\Omega$  and the second plate is connected directly to the ground.

- It simply acts as a voltage divider bias network, and output is taken directly from the first terminal of the sensor pin, which is shown in the figure above.
- The output will change in the range of 0 – 5 Volt, in proportion to the change in the content of water in the soil.
- Deally, when there is zero moisture in the soil, the sensor acts as an open circuit which means infinite resistance. For this condition. For this condition, we get 5V at the output.

**2.1.4.5 Capacitive Soil Sensor**

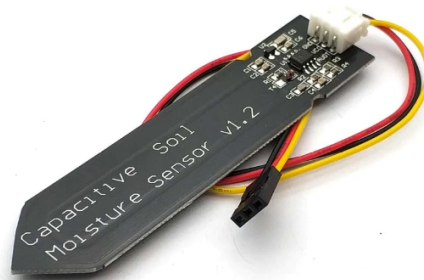


Figure 2.12: Capacitive soil sensor

The Capacitive Soil Moisture Sensor Module determines the amount of soil moisture by measuring changes in capacitance to determine the water content of the soil. This can be used in an automatic plant watering system or to signal an alert of some type when a plant needs watering.

**Specifications**

- Vcc Range 3.3 to 5.5V.
- I(typ) < 5mA.
- Vout Analog Output Range @ 5V 1.2V to 3V (typ)

**Theory of Operation :**

- The module uses a TL555I CMOS timer to create a 1.5MHz clock. The TL555I is similar to the ubiquitous NE555 but is newer CMOS technology with a higher frequency capability and other improvements.
- A peak voltage detector converts the waveform from the TL555I into a DC voltage that

can be read by the ADC input of a microcontroller.

- When the probe is exposed to moisture, it affects the capacitance of the circuit which in turn affects the peak amplitude of the signal and therefore the DC voltage output that is being monitored by the MCU.
- Higher moisture = lower DC voltage output.

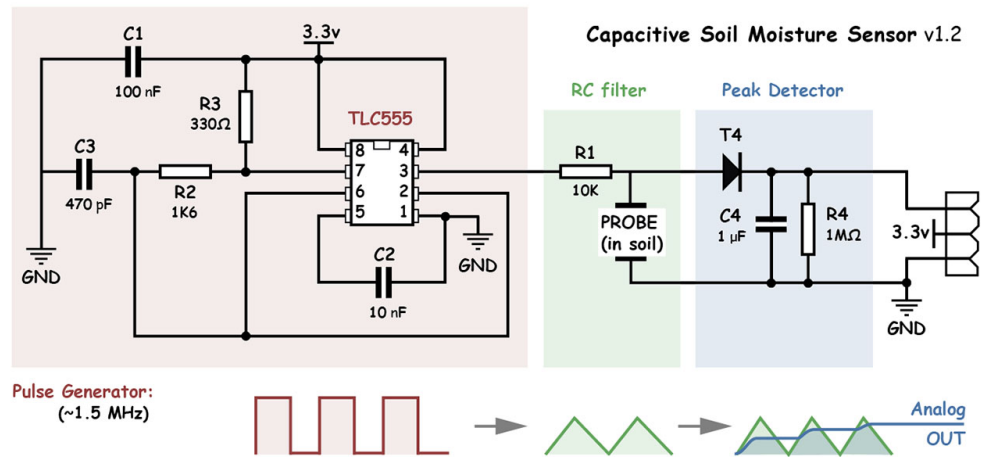


Figure 2.13: Soil Sensor Module Schematic

### 2.1.4.6 Calibration

To calibrate this sensor, we started by preparing the soil. We mixed potting soil with regular soil in a 50:50 ratio as shown in Figure 2.8 with a weight of 4 kg, then heated it in an oven for 24 hours at 106 degrees to evaporate all the moisture inside.[13]



Figure 2.14: Weight of the soil before drying



Figure 2.15: Drying the soil in the oven

After 24 hours of drying, we measured the weight of the soil and found that the weight had decreased due to the drying process, as clearly shown in the photo below.



Figure 2.16: Weight of the soil after drying

Next, we measured repetitive values every 4 minutes, adding 70 ml each time until we reached a total quantity of 1 liter. For this, we used a resistive soil sensor and a capacitive soil sensor. As reference values, we used an SHT31 sensor and a measurement device used by the center connected with a thermocouple.



Figure 2.17: Starting the calibration

After collecting the experiments' data, we saved them in a CSV file. Then, we performed a simple polynomial regression to find the function that gives the soil moisture values as a function of the output voltages. Figures 4.18 show the results we obtained.

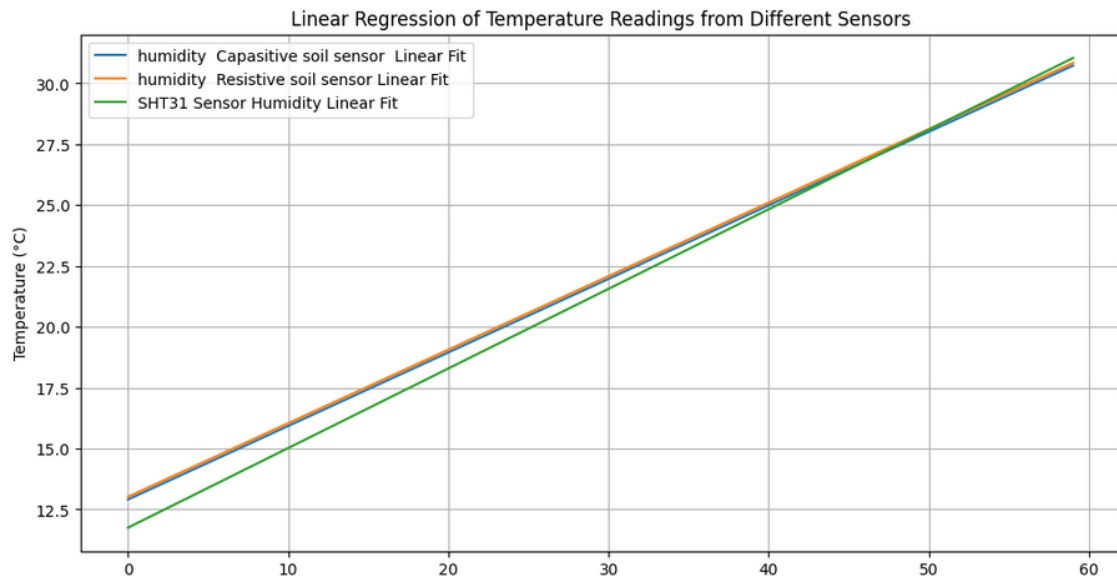


Figure 2.18: The plot of the data Humidity from Soil sensor

#### 2.1.4.7 Gaz Sensor



Figure 2.19: Gaz Sensor MQ-135

#### Specifications

- **Power Supply:** From 2.5 V to 5 V
- Analog and Digital Output

#### 2.1.4.8 Solar Irradiance Sensor



Figure 2.20: Solar Irradiance Sensor

This sensor operates on the principle of photosensitive elements converting ultraviolet rays into measurable electrical signals, enabling online monitoring of ultraviolet radiation. It finds broad applications in environmental monitoring, meteorological observation, agriculture, forestry, and other settings. Capable of measuring ultraviolet radiation in both atmospheric and artificial light source environments, it offers versatility and utility across various domains.

#### Specifications

- Input Voltage: 10-30 VDC
- Power Consumption:
  - o RS485 Output Type: 0.06 W
  - o Analog Output Type: 0.6 W
- Load Capacity:
  - o 4-20mA Output Type: Load Capacity 600
  - o 0-5V/0-10V Output Type: Output Resistance 250
- Work Temperature: -25°C to +60°C
- UV Intensity Measuring Range: 0-15 mW/cm<sup>2</sup>
- Resolution: 0.01 mW/cm<sup>2</sup>
- Precision: ±10
- UV Index Measuring Range: 0-15
- Measurement Wavelength Range: 290-390 nm
- Response Time:

- UV Intensity: 0.2 s
- UV Index: 0.2 s
- Linearity:  $\pm 1$
- Long-term Stability:  $\pm 3$
- Output: RS485 (Modbus RTU protocol)/4-20mA/0-5V/0-10V
- IP Protection: IP67

### 2.1.4.9 Calibration

In our study, we propose the following calibration method for our solar irradiation sensor: we place it alongside a reference sensor for 8 hours. Both sensors record solar irradiation values during this time, which are then saved onto our data platform. Subsequently, we extract the recorded data in CSV format for analysis. Using the collected data, we generate a graphical representation to visualize the solar irradiation readings from both sensors over the 8 hours.

Upon analysis, we compute the mean solar irradiation values obtained from each sensor. For the reference sensor, the mean value is calculated to be 761.18, while for our sensor, which reads in analog, the mean value is 684.34. The bias between the two sensors is determined by the difference in their mean values, resulting in a bias of 76.84.

This calibration process is crucial for ensuring the accuracy and reliability of our sensor readings. However, further investigation is warranted to explore potential sources of bias and to refine our calibration technique. Possible avenues for future research include examining environmental factors that may influence sensor performance and exploring advanced calibration methods to minimize bias and improve accuracy.



Figure 2.21: Calibration of Solar Irradiance Sensor



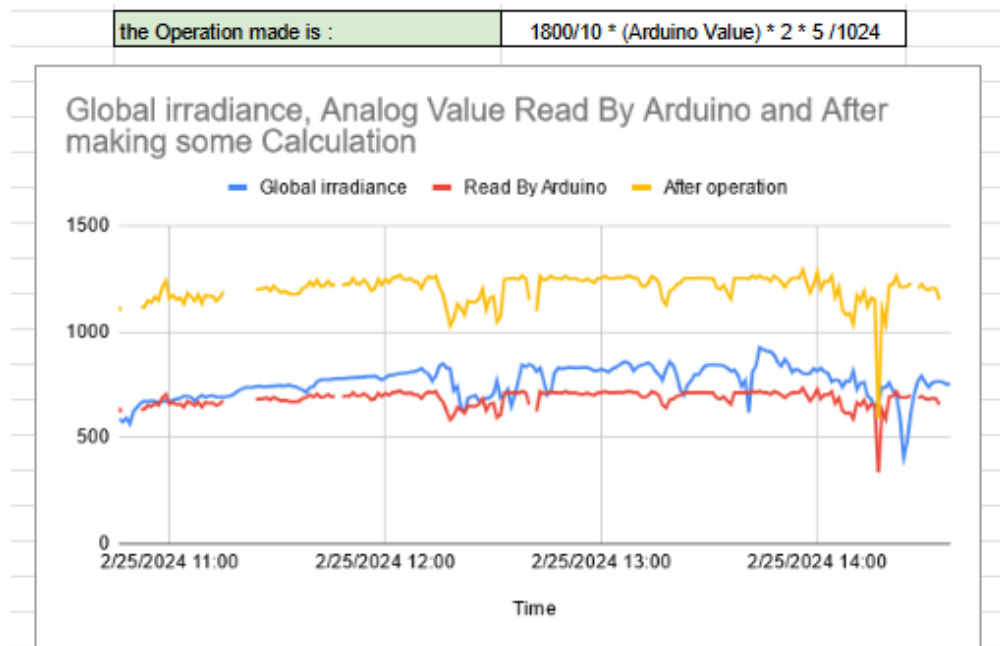


Figure 2.22: Irradiation Calibration Graphs

Sensor	Mean Solar Irradiation (W/m <sup>2</sup> )
Reference Sensor	761.18
Our Sensor (Analog)	684.34
Bias	76.84

Table 2.2: Calibration Result

#### 2.1.4.10 Ambient light Sensor



Figure 2.23: Ambient light Sensor

## Specifications

- **DC power supply (default):** 12-24V DC
- **Power consumption:** Light intensity  $< 0.15W$  (@12V DC, 25°C)
- **Light intensity accuracy:**  $\pm 5\%$  (25°C)
- **Long-term stability (light intensity):**  $< 5\%/y$
- **Output signal:** RS485, 4-20mA, 0-5V, 0-10V
- **Working pressure range:** 0.9-1.1 atm

The luminometer, also known as a light sensor Modbus( Master-Slave technique), measures the light intensity in its environment by converting captured light into an electrical signal. This signal is then electronically processed and transmitted digitally via an RS485 output. RS485 is a serial communication interface that enables reliable data transfer between electronic devices over long distances, using a differential signal configuration on two wires. RS485 supports bidirectional communication and can be used with various protocols, making it a popular choice for many industrial and commercial applications.

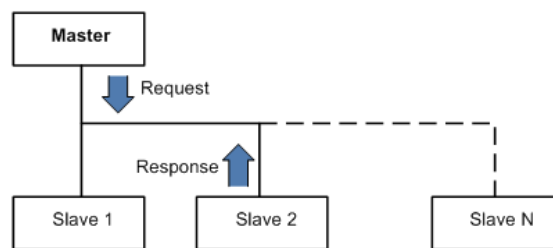


Figure 2.24: Modbus communication

### 2.1.5 Actuator Used

In this section, we will define the actuators provided by the center for our project. It has been found that these actuators are among the most important ones for maintaining optimal weather conditions in the greenhouse, thereby increasing productivity and reducing energy consumption costs. We will present each actuator individually.

### 2.1.5.1 Solenoid Valve



Figure 2.25: Solenoid valve Actuator

We used the solenoid valve in our system to control greenhouse irrigation

#### Specifications

- **Type:** Solenoid valve.
- **Design:** 1-chamber valve.
- **Ports:** 3/2 ports.
- **Operation Mode:** Direct control.
- **Function:** Normally closed (NC) at rest.
- **Mounting:** Any position.
- **Fluids:** Neutral gases.
- **T-Fluid:** 5 - 98 °C.
- **T-Ambient:** 5 - 60 °C. item **Nominal voltage** 12 VDC.
- **Nominal Power:** 4.8 W.

### 2.1.5.2 Micro Diaphragm Pump

Is a compact device used for the precise and efficient application of small quantities of fertilizers, pesticides, or irrigation water.



Figure 2.26: Micro Diaphragm Pump Actuator

### Specifications

- **Voltage** : 12Vdc.
- **Power(max)** : 60W.
- **Pressure(max)**:0.8Mpa.
- **Flow(max)**: 5L/min.

#### 2.1.5.3 DC Submersible Water Pump

Submersible pumps are the pumping machines that are designed to submerge into liquids. The DC submersible pump refers to the submersible pumps that use the direct current electricity as the power source.



Figure 2.27: DC Submersible Water Pump Actuator

### 2.1.6 DC to DC converter



Figure 2.28: DC to DC buck converter

The buck converter is used to convert a higher input voltage into a lower output voltage. It employs two power switches to transfer charge to the output capacitor. Figure 3.4 shows a block diagram that provides a simple representation of a buck converter. The output voltage is regulated by adjusting the duration each switch is on.

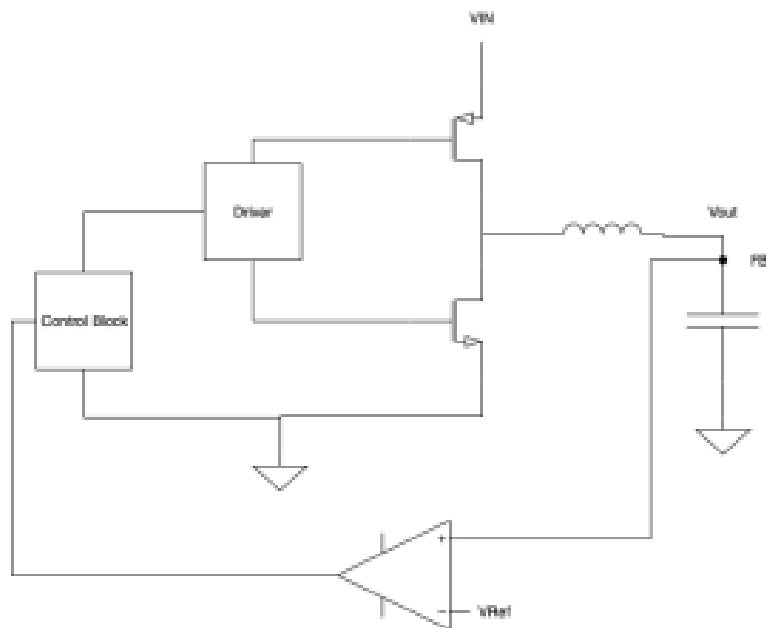


Figure 2.29: Buck converter representation

The power switch is turned on. When the PMOS is on, the NMOS is off, causing the current in the inductor to rise. Conversely, when the NMOS is on, the PMOS is off, and the current in the inductor falls. There are two modes of operation: PFM and PWM.

In PFM mode, the inductor current pulses are spaced in time with the same amplitude and duration, meaning the frequency changes but the pulse width remains constant. When the load increases, PFM mode cannot keep up because the pulse width does not change and the frequency cannot be increased. Therefore, PWM mode is used.

In PWM mode, the pulse width changes while the period remains constant. If the load increases, the on-time of the PMOS switch increases. If the load decreases, the on-time of the PMOS decreases and the on-time of the NMOS switch increases. The equation for the output voltage is derived based on these principles.

$$\text{Efficiency} = \frac{V_{OUT} \times I_{OUT}}{V_{OUT} \times I_{OUT} + P}$$

The equation represents the efficiency of a buck converter, which is determined by various power losses, including those from the PMOS and NMOS MOSFETs, the capacitor, the inductor, and the energy required for the control loop. The efficiency can reach up to 96%, as demonstrated by the TPS627431 from Texas Instruments [14].

## 2.1.7 GSM Module



Figure 2.30: GSM Module Pinout

### Specifications

- **Dimensions:** 2.2 cm x 1.8 cm.
- **Power Supply:** Requires a voltage between 3.4V and 4.4V.
- **Current Requirement:** Requires a peak current of approximately 2A.

### Features

- Automatically starts and searches for the network.

### Power Supply Considerations

- The 5V power supply from the Arduino is not suitable for the SIM800L.
- To address the power supply issue, a 1N4007 diode is added between the 5V pin of the Arduino and the VCC pin of the SIM800L.

### 2.1.8 SD Card Module

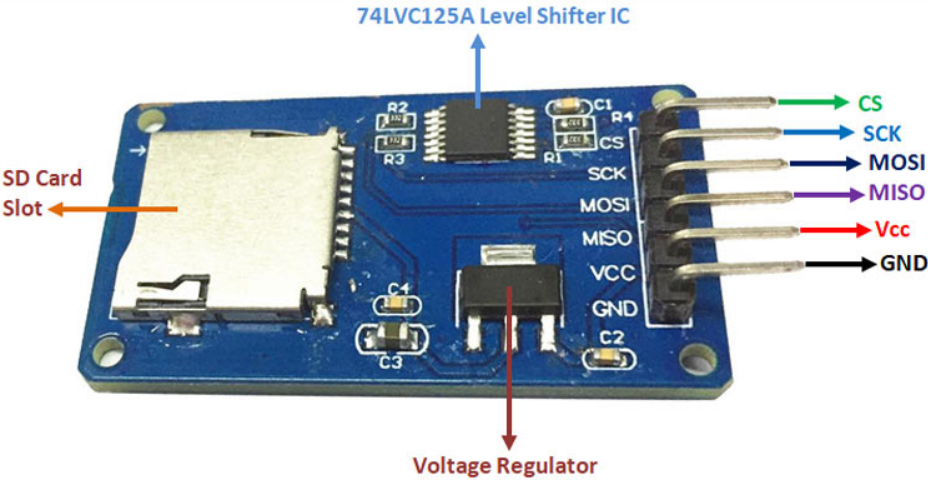


Figure 2.31: SD Card Module



## Specifications

- **Operating Voltage:** 4.5V - 5.5V DC.
- **Current Requirement:** 0.2-200 mA.
- **Voltage Regulator:** 3.3V on-board.
- **File System:** Supports FAT file system.
- **Micro SD Support:** Up to 2GB.
- **Micro SDHC Support:** Up to 32GB.
- **Communication Protocol:** SPI

### 2.1.9 Relay Module



Figure 2.32: 2 Channel Relay Module

### 2.1.10 Specifications

- **Voltage Rating:** 250VAC 10A, 28VDC 10A
- **Features:** Indicator LEDs
- **Control Voltage:** 5V

### 2.1.10.1 Relay Description

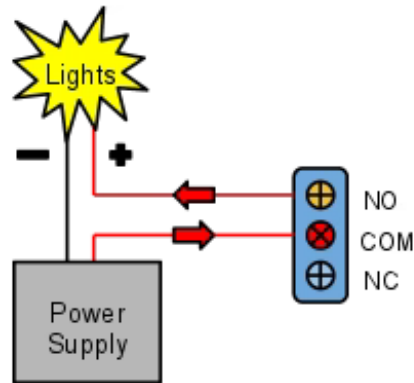


Figure 2.33: Relay Function Mode

A relay is an electrically operated switch. Many relays use an electromagnet to mechanically operate a switch, but other operating principles are also used, such as solid-state relays. Relays are used where it is necessary to control a circuit by a separate low-power signal, or where several circuits must be controlled by one signal.

Relays work on electromagnetism. When the relay coil is energized, it acts like a magnet and changes the position of a switch. The circuit that powers the coil is completely isolated from the part that switches ON/OFF. This provides electrical isolation. This is the reason we can control a relay using 5V from an MCU, while the other end of it could be running a 220 to 240V appliance. The 240V end is completely isolated from the 5V Arduino circuitry.

## 2.2 Technology

### 2.2.1 Wireless technology

It refers to communication methods that transmit data without the need for physical cables or wired connections. In IoT systems, wireless technologies play a crucial role in enabling connectivity between devices. Various wireless technologies like Wi-Fi, Bluetooth, Zigbee, Z-Wave, LoRa, and cellular networks are used based on factors such as range, power consumption, and data rate. Wi-Fi offers high-speed connectivity, Bluetooth is ideal for short-range applications, Zigbee and Z-Wave are suited for low-power smart home devices, and LoRa provides long-range coverage with low power. Cellular networks like NB-IoT, a protocol dedicated to low-power wide-area networks and the Internet of Things, and LTE-M, a low-power wide-area communication technology standard, offer wide-area coverage. These technologies facilitate automation, monitoring, and control across diverse domains, driving efficiency and innovation in industries. In our project, we have chosen to work with LoRa for several reasons, which we will explain in the next subsection.

## 2.2.2 LoRa

LoRa, or Long Range Radio, is a wireless modulation technology tailored for IoT and M2 (Machine to Machine) networks, emphasizing extended connectivity and energy efficiency[9]. It facilitates the integration of multiple applications within shared networks, promoting cost-effective solutions for diverse IoT deployments. With its multi-layered protocol ensuring secure communications and network-level encryption, LoRa enables individual gateways to manage vast numbers of nodes, supporting scalable implementations. Notably, its long-range transmission capabilities reduce infrastructure requirements, accelerating network deployment while minimizing costs.

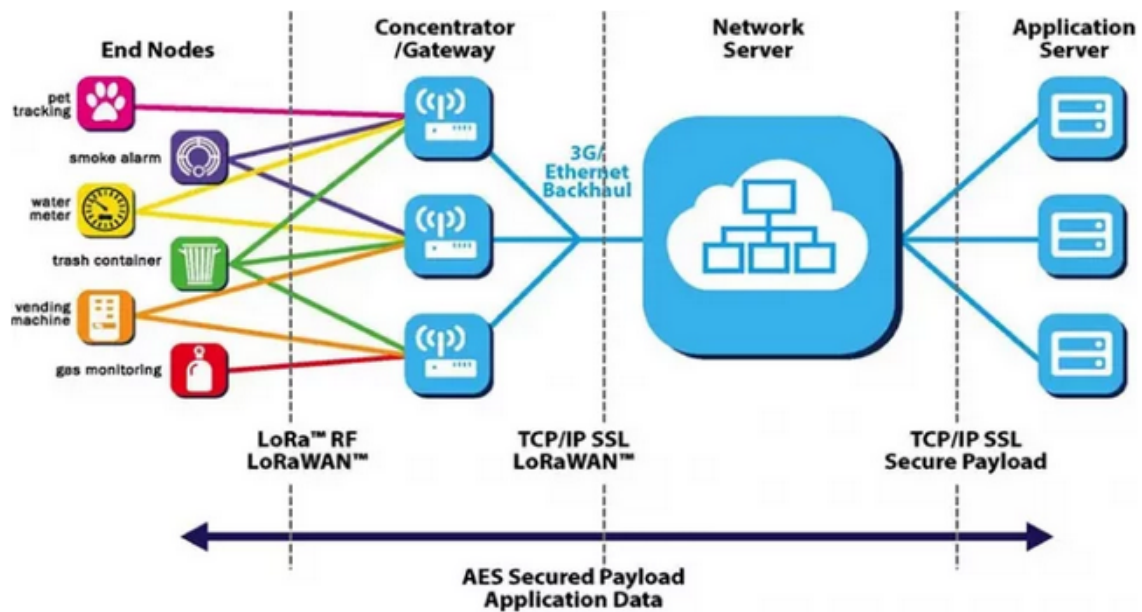


Figure 2.34: LoRa Connectivity Schematic

## 2.2.3 LoRa Network Architecture

LoRaWAN architecture consists of end nodes, gateways, the network server, and the application server. In terms of the authentic architecture for the LoRa network, the nodes are typically in a star-of-stars topology with gateways forming a see-through bridge. These relay messages between the central network server and end devices in the backend. Communication to endpoint nodes is usually bi-directional, but it is also possible to support multicast operation, and this is useful for features such as the other mass distribution messages, or software upgrades.



Figure 2.35: LoRa Sensor

**Specifications:**

- Communication distance: up to 15 km.
- Sensitivity: down to -148dBm.
- Programmable bit rates: up to 300bps to 50 kbps.
- payload of each message is 243 bytes.
- RSSI dynamic range: 127dB .
- Wireless frequency: 433MHz.
- Working voltage: 1.8-3.7v.
- Working temperature: -40 - +80.

**Why did we choose Lora ?**

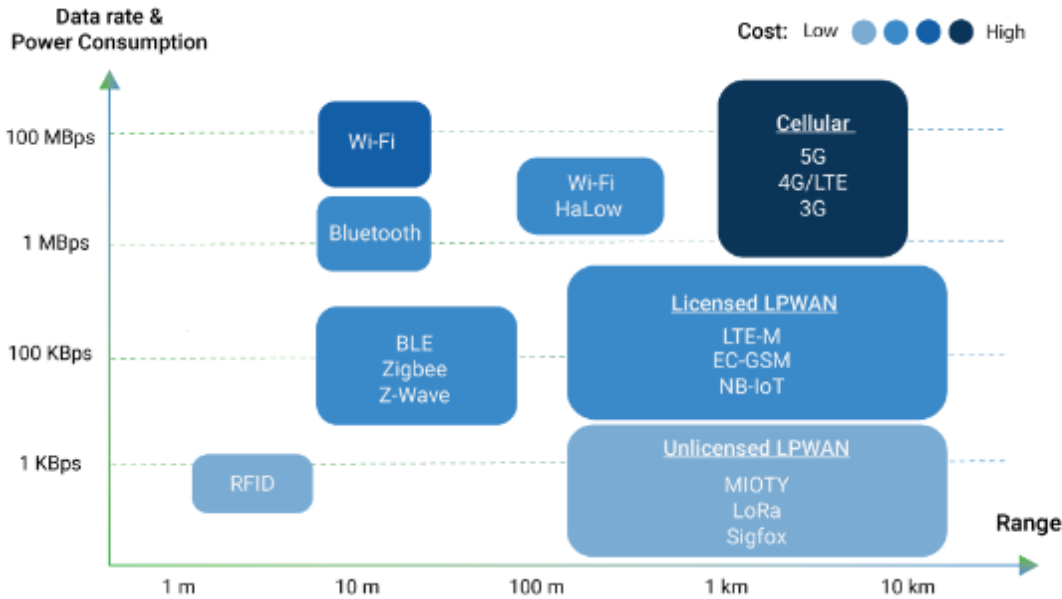


Figure 2.36: Wireless Technology used in IoT

Figure 2.19 depicts a comparison of various categories discussed earlier. The vertical axis represents the data rate and power consumption of each protocol. A higher position indicates a higher data rate and power consumption, while a lower position signifies a lower data rate and power consumption.

The horizontal axis illustrates the range of each protocol. Protocols with greater range are positioned towards the right, whereas those with shorter range are towards the left. Notably, protocols like LoRa, Sigfox, and MIOTY are highlighted due to their extensive range and relatively low power consumption.

For these reasons, we justify our choice of the LoRa module over other technologies by examining its strengths and weaknesses in comparison to other available options. Table 2.3 summarizes our comparison.

After analyzing the comparison of LoRaWAN technology in the table and taking into account our project requirements, we have pinpointed three crucial characteristics that help us in selecting the most appropriate technology for our application.

- **Long-Range Coverage for Remote Farms:** LoRa’s coverage range is well-suited for agriculture, providing reliable data acquisition from remote farms, even in rural areas with limited infrastructure. This ensures comprehensive monitoring and control over vast agricultural landscapes.
- **Adaptive Data Rate for Variable Environments:** LoRa’s adaptive data rate feature optimizes communication based on signal conditions, making it adaptable to the variable and dynamic environments found in agriculture. This ensures efficient data acquisition in changing conditions such as crop growth stages or weather patterns.
- **Localization Support for Precision Agriculture:** Techniques like RSSI enable localization capabilities in LoRa, enhancing precision and accuracy in data acquisition.

Characteristics Type of LPWAN	SigFox	LoRa (LoRaWAN)	NB-IoT
Modulation	BPSK	CSS	QPSK
Frequency	Unlicensed ISM bands	Unlicensed ISM bands	Licensed LTE bands
Bandwidth	100 Hz	250 kHz and 125 kHz	200 kHz
Maximum data rate	100 bps	50 kbps	200 kbps
Bidirectional	Limited / Semi-duplex	Yes / Semi-duplex	Yes / Semi-duplex
Maximum messages per day	140 (UL), 4 (DL)	Unlimited	Unlimited
Maximum payload length	12 bytes (UL), 8 bytes (DL)	243 bytes	1600 bytes
Coverage Range	10 km (urban), 40 km (rural)	5 km (urban), 20 km (rural)	1 km (urban), 10 km (rural)
Interference immunity	Very high	Very high	Low
Authentication & encryption	Not supported	Yes (AES 128b)	Yes (LTE encryption)
Adaptive data rate	No	Yes	No
Handover	End devices do not join a single base station	End devices do not join a single base station	End devices join a single base station
Localization	Yes (RSSI)	Yes (TDOA)	No (under specification)
Allow private network	No	Yes	No
Standardization	Sigfox company is collaborating with ETSI on the standardization of Sigfox-based network	LoRa-Alliance	3GPP

Table 2.3: Comparison of LPWAN Technologies Characteristics

tion. This is crucial for precision agriculture applications, allowing farmers to monitor specific areas, optimize resource usage, and implement targeted interventions based on localized data.

LoRa’s long-range coverage, adaptive data rate, and localization support make it a preferred choice for data acquisition in agriculture systems, enabling farmers to make informed decisions for efficient and sustainable farming practices.

### 2.2.4 PHP:



Figure 2.37: PHP logo

PHP (Hypertext Preprocessor) is a popular open-source server-side scripting language widely used for web development. It is designed to create dynamic web pages and applications by embedding code within HTML. PHP code is executed on the server, and the resulting output is sent to the client’s web browser.

PHP is known for its ease of use, versatility, and extensive library support. It integrates seamlessly with various databases, making it a popular choice for building data-driven web applications. Additionally, PHP has a large and active community, which contributes to its continuous development and the availability of numerous third-party libraries and frameworks.

One of the key strengths of PHP is its cross-platform compatibility, allowing developers to deploy applications on different operating systems and web servers. It also offers robust security features and supports a wide range of protocols, including HTTP, HTTPS, FTP, and SMTP[10].

### 2.2.5 Laravel Framework:



Figure 2.38: Laravel logo

**Laravel** is a free, open-source PHP web application framework that follows the Model-View-Controller (MVC) architectural pattern [11]. It aims to simplify the development of robust and maintainable web applications by providing a set of tools and conventions out of the box. Key features of Laravel include:

- **Elegant Syntax:**Laravel offers an expressive and readable syntax, making the code more concise and developer-friendly.
- **Modular Structure:**Laravel promotes a modular and decoupled codebase, enabling better code organization, reusability, and testability.
- **Powerful Tools:**Laravel provides a comprehensive set of tools and features, such as routing, templating, database abstraction, authentication, caching, and more.
- **Command-Line Interface (CLI):**Laravel includes Artisan, a powerful CLI tool that simplifies common development tasks like creating controllers, models, and migrations.
- **Security Features:**Laravel offers built-in security features, including protection against common web vulnerabilities like CSRF and SQL injection, as well as user authentication and authorization mechanisms.
- **Third-Party Integration:**Laravel supports seamless integration with popular PHP packages and libraries, making it easier to extend functionality.

- **Active Community:** Laravel has a large and active community, ensuring continuous development, updates, and availability of third-party packages.

With its wide range of features, tools, and conventions, Laravel aims to streamline the development process and promote best practices, making it a popular choice for building modern, scalable, and secure web applications in PHP[11].

**In another hand**, PHP is a static UI language, requiring full page refreshes for any UI updates, leading to interruptions in user experience and limited interactivity, in contrast to dynamic UI updates enabled by dynamic languages like JavaScript.

### 2.2.6 MySQL:



Figure 2.39: MySQL logo

MySQL is an open-source relational database management system (RDBMS) known for its reliability, scalability, and ease of use. It follows the relational model, allowing data to be organized into tables with rows and columns. MySQL supports SQL for querying and managing data and operates on a client-server architecture, enabling multiple clients to connect to a MySQL server. It is cross-platform compatible and offers features for scalability, performance, security, and high availability. MySQL has a large and active community and is backed by Oracle Corporation, providing both community and commercial support.



# Chapter 3

## Proposed Solution

Our solution is divided into three main parts: the Platform, the Base Station, and Node Architecture. In this chapter, we discuss the architecture and the role of each component.

### 3.1 Platform

#### 3.1.1 Introduction

In terms of security measures, our system does not employ data encryption; however, it incorporates an API for data storage. While data encryption is not implemented, the API provides a secure means for storing and accessing data, ensuring reliability and integrity in data management.

The Web-based platform serves as the central hub for data aggregation, visualization, and control in this IoT greenhouse monitoring system. Emphasizing its importance as a central hub for data aggregation, visualization, and control, it provides a secure infrastructure for transmitting data from the on-site base station to remote users. Through an intuitive web application, users can access real-time and historical environmental data, enabling comprehensive monitoring of greenhouse conditions. The platform supports remote control of actuators, allowing for automated adjustments to irrigation, climate systems, and more. Although data encryption is not implemented, the platform ensures data security through robust user authentication mechanisms and employs an API for secure data storage. Its scalable architecture accommodates integration with advanced analytics tools, facilitating data-driven decision support for optimizing crop management strategies[15].

**Architecture Overview** To understand the architecture of the platform, we need to understand the architecture of the Laravel Framework. As it's illustrated in Figure 3.2.

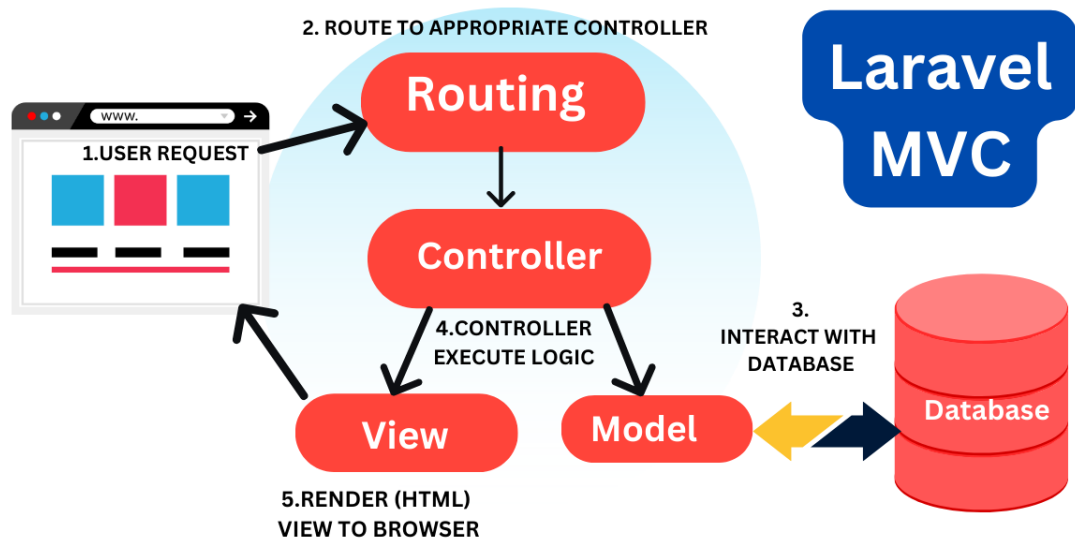


Figure 3.1: Larevl Architecture

1. **User Request:** A user initiates a request.
2. **Route to Appropriate Controller:** Laravel routes the request to the appropriate controller based on the URL.
3. **Interact with Database:** The controller interacts with the database using either Eloquent ORM (Object-Relational Mapping) or Query Builder.
4. **Controller Executes Logic:** The controller executes application logic, interacts with models, and prepares data.
5. **Render HTML View to Browser:** Controller loads a view, passes data, and renders an HTML response.

The architecture of the platform in our system is illustrated in this figure :

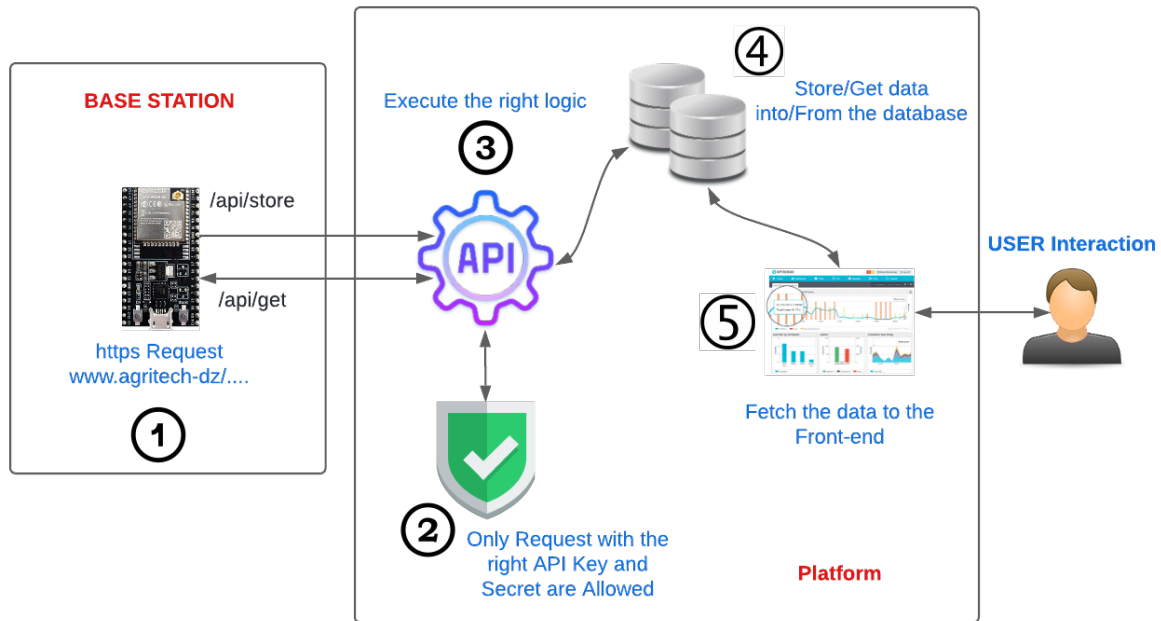


Figure 3.2: Platform Architecture

- 1. ESP32 Sends HTTPS Request:** The ESP32 device initiates an HTTPS request to the API server, specifying whether it is a POST or GET request. This request typically includes data to be processed by the API.
- 2. API Middleware Validates Request:** Upon receiving the request, the middleware of the API server checks whether it contains the correct API key and API secret. These credentials are usually stored in the database. If the request includes valid credentials, it proceeds to the next step; otherwise, it rejects the request and returns an error response.
- 3. API Executes Request Logic:** After passing through the middleware, the API server executes the logic associated with the request. This logic can involve various operations depending on the type of request received, such as processing sensor data, retrieving or storing information in the database, or performing specific actions based on the request parameters.
- 4. Database Interaction Based on Request Type:**
  - If the request is a store function (e.g., a POST request to save data), the API server stores the data received from the ESP32 device into the database.
  - If it's a GET request, the API server retrieves the requested data from the database based on the parameters provided in the request. This data is then formatted and send it back to the ESP32 device.
- 5. Data Sent to User-Side:**

Finally, the data fetched from the API is displayed on the front end of the platform, appearing on the screen of the user. This completes the communication cycle between the ESP32 device, the API server, and the user interface.

At any time the User can control the actuators of his greenhouses and their states will be changed in the database.

### 3.1.2 Web Application Interface

The URL to our Platform is : [www.agritech-dz.com](http://www.agritech-dz.com)

### 3.1.3 Login Page

The root page of our platform is a login form, where users can access their accounts by providing their email and password. The login form serves as the entry point to the platform and ensures secure access for registered users.

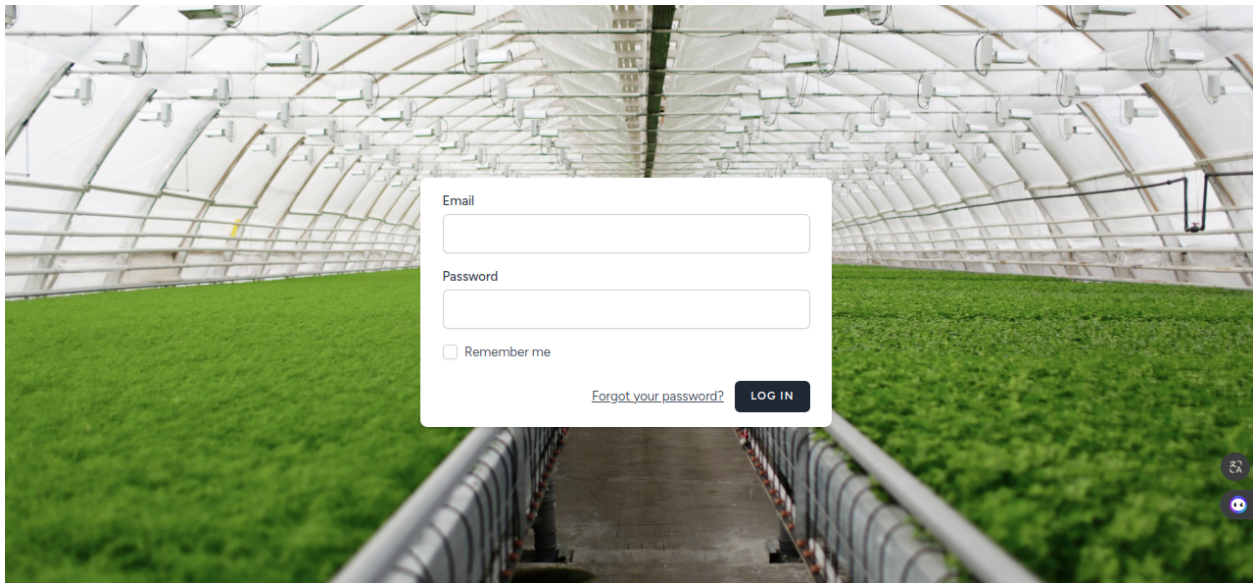


Figure 3.3: Login Page of Agritech-DZ Platform

#### 3.1.3.1 Dashboard Page

After a user successfully authenticates, he is directed to his personalized dashboard. The dashboard provides real-time insights into the conditions of their greenhouses, allowing them to monitor critical parameters such as temperature, humidity, CO<sub>2</sub> levels, soil humidity, and irradiation.

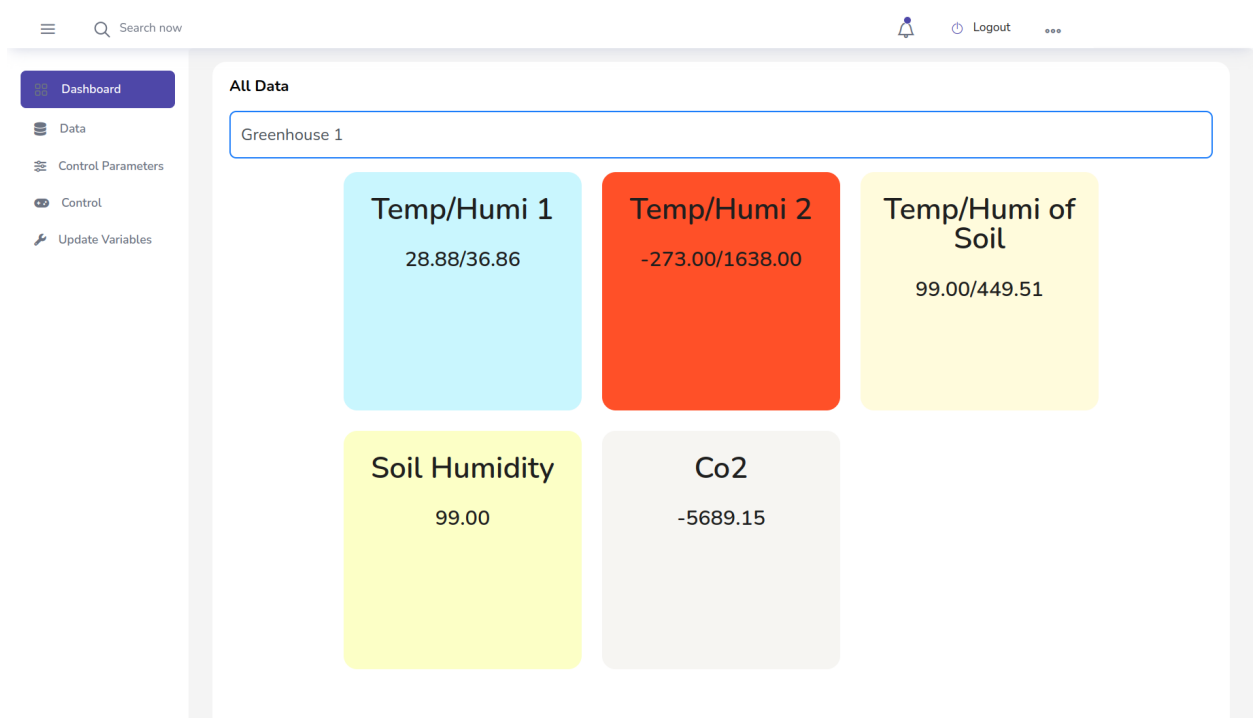


Figure 3.4: Dashboard Page of Agritech-DZ Platform

**Select Greenhouse :** At the top of the dashboard, users are presented with a select bar where they can choose the greenhouse they want to monitor. This dropdown menu enables users to switch between their different greenhouse locations easily.

**Real-Time Data Visualization :** The dashboard features visually appealing and informative half-circle charts to display real-time data. Each parameter (e.g., temperature, humidity) is represented by a separate half-circle chart. The current value of the parameter is displayed prominently in the center of the chart, making it easy for users to quickly grasp the current status.

### 3.1.3.2 Data Page

The historical data page provides users with comprehensive access to the historical data of all their greenhouses. It presents the acquired data along with the acquisition time in a sophisticated table format, allowing users to analyze trends and patterns over time.

**Table Display:** The data is displayed in a well-organized table format, with each row representing a data entry and columns corresponding to different parameters (e.g., temperature, humidity) and the acquisition time. Users can scroll through the table to explore historical data entries.

**Search Bar:** To facilitate data exploration, the page includes a search bar that allows users to filter data entries based on specific criteria. Users can enter keywords or filter parameters to quickly locate relevant data entries within the table.

**Download Button:** For further analysis or offline use, the historical data page includes a Download button that enables users to export the displayed data in an Excel file format. This feature provides users with the flexibility to perform in-depth analysis using external tools or share the data with colleagues or stakeholders.

Greenhouse Name	Temperature	Humidity	CO2	Soil Humidity	Wind Velocity	Irradiation	Acquisition Time
Greenhouse1	0	0	0	0	2	3.52	2024-02-25 09:15:37
Greenhouse1	0	0	0	0	630.01	1107.43	2024-02-25 09:46:05
Greenhouse1	0	0	0	0	634.01	1114.46	2024-02-25 09:52:22
Greenhouse1	0	0	0	0	634.01	1114.46	2024-02-25 09:53:24
Greenhouse1	0	0	0	0	652.01	1146.11	2024-02-25 09:54:26
Greenhouse1	0	0	0	0	647.01	1137.32	2024-02-25 09:55:28
Greenhouse1	0	0	0	0	662.01	1163.68	2024-02-25 09:56:29

Figure 3.5: Data Page of Agritech-DZ Platform

### 3.1.3.3 Control Parameters Interface

Parameter Settings

Frequency:

Maximum Temperature:  Minimum Temperature:

Maximum Humidity:  Minimum Humidity:

Maximum Soil Humidity:  Minimum Soil Humidity:

Maximum Co2:  Minimum Co2:

Figure 3.6: Control Parameters Page of Agritech-DZ Platform

The Control Parameters interface allows users to customize the operational settings of the system’s actuators, ensuring optimal environmental conditions. This interface includes the following adjustable parameters:

#### 3.1.3.4 Temperature Control

- **Maximum Temperature:** Defines the upper threshold for temperature. Exceeding this limit activates cooling mechanisms.

- **Minimum Temperature:** Defines the lower threshold for temperature. Dropping below this limit activates heating mechanisms.

### 3.1.3.5 Humidity Control

- **Maximum Humidity:** Sets the upper limit for relative humidity. Exceeding this value triggers dehumidification processes.
- **Minimum Humidity:** Sets the lower limit for relative humidity. Falling below this value activates humidifiers.

### 3.1.3.6 CO<sub>2</sub> Control

- **Maximum CO<sub>2</sub> Concentration:** Establishes the upper threshold for CO<sub>2</sub> levels. Surpassing this limit activates ventilation systems.
- **Minimum CO<sub>2</sub> Concentration:** Establishes the lower threshold for CO<sub>2</sub> levels. Dropping below this limit reduces ventilation.

### 3.1.3.7 Soil Humidity Control

- **Maximum Soil Humidity:** Sets the upper limit for soil moisture content. Exceeding this level activates drainage systems or reduces irrigation.
- **Minimum Soil Humidity:** Sets the lower limit for soil moisture content. Falling below this level activates irrigation systems.

### 3.1.3.8 Data Acquisition Frequency

- **Data Acquisition Frequency:** Specifies how often the system collects sensor data. Higher frequencies provide more current information but may increase system load and energy consumption.

### 3.1.3.9 Example Parameter Configuration

- **Maximum Temperature:** 30°C
- **Minimum Temperature:** 18°C
- **Maximum Humidity:** 70%
- **Minimum Humidity:** 40%
- **Maximum CO<sub>2</sub> Concentration:** 1000 ppm
- **Minimum CO<sub>2</sub> Concentration:** 300 ppm
- **Maximum Soil Humidity:** 60%
- **Minimum Soil Humidity:** 20%

- **Data Acquisition Frequency:** Every 5 minutes

By adjusting these parameters, users can ensure the system operates efficiently and maintains optimal conditions for specific requirements. This flexibility is essential for adapting to varying operational needs and achieving desired environmental quality.

### 3.1.3.10 Control Page

The control page empowers users to interact with the actuators of their greenhouses, enabling them to monitor and control the real-time status of these devices.

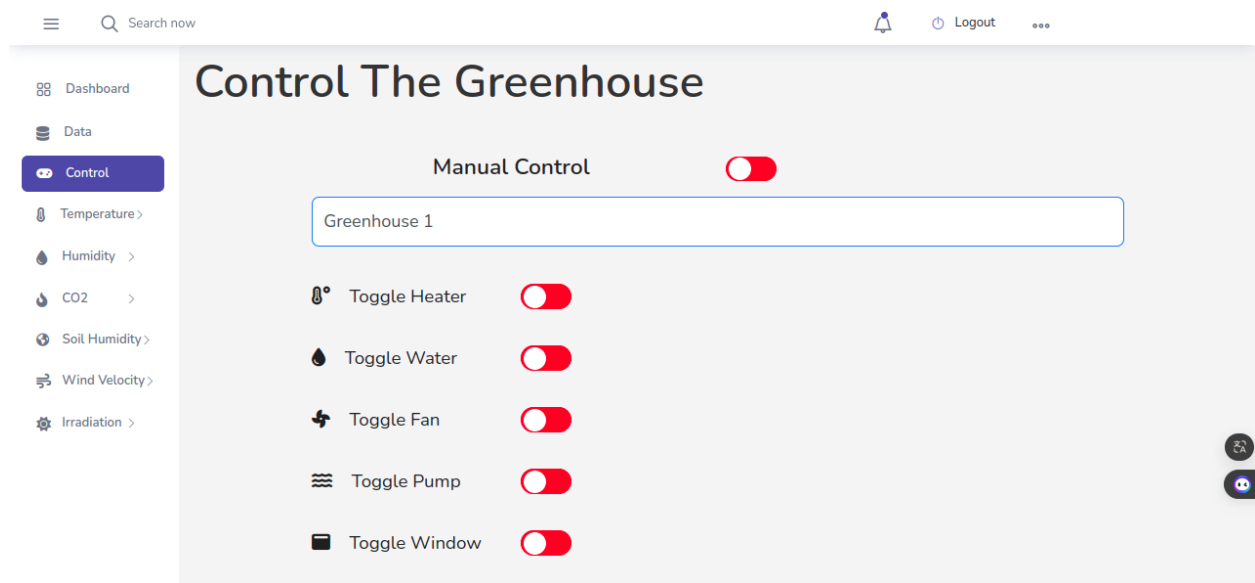


Figure 3.7: Control Page of Agritech-DZ Platform

**Select Greenhouse:** Users can select the greenhouse they want to control from a drop-down menu (select bar). Upon selecting a greenhouse, the real-time status of the actuators associated with that greenhouse is displayed.

**Actuator Status Display:** The real-time status of the actuators is visually represented, with red indicating that the actuator is turned off and green indicating that it is turned on. This visual cue allows users to quickly assess the status of each actuator at a glance.

**Interacting with Actuators:** Users can interact with the actuators by clicking on buttons corresponding to each device. Clicking a button toggles the state of the actuator and updates the database accordingly. For example, clicking a button to turn on a heater will update the database to reflect the change in status.

**Automatic Mode:** Additionally, users have the option to enable an "Automatic" mode by clicking a switch. In automatic mode, the actuators are controlled based on predefined rules or conditions. For example, if the temperature drops below a certain threshold, the heater will automatically turn on to maintain the desired temperature level.

**ESP32 Integration:** The control page is designed to work seamlessly with ESP32 devices. When the ESP32 sends a GET request to the server (typically every 3 seconds), the server retrieves the latest actuator states from the database and sends the corresponding commands



to the devices. This ensures that the actuators respond promptly to changes made by the user or automated rules.

### 3.1.3.11 Responsive View

The platform is designed to provide optimal user experience across various screen sizes, including smartphones, tablets, and desktops. The responsive design ensures that users can access and interact with the platform seamlessly, regardless of the device they are using.

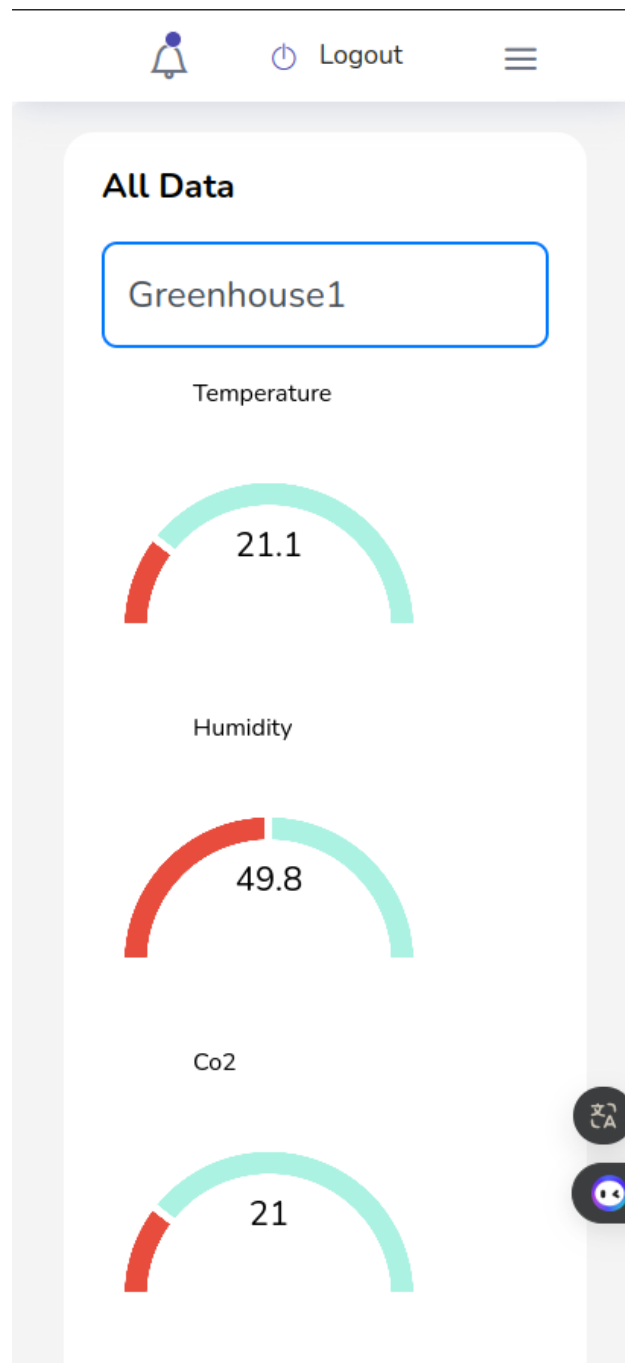


Figure 3.8: Responsive Page of Agritech-DZ Platform

### 3.1.4 Security Measures

The security of the platform is paramount to ensure the integrity and confidentiality of data. Several measures have been implemented to safeguard the system:

#### 1. Authentication Layer

- The platform employs a robust authentication layer to verify the identity of users before granting access to the system. This ensures that only authorized personnel can interact with the greenhouse monitoring application.

#### 2. Middleware Group for Route Protection

- Middleware groups have been configured to protect all routes of the application. These middleware functions act as a filter, intercepting incoming requests and enforcing access control policies. By implementing middleware at the application level, vulnerabilities are minimized, and unauthorized access attempts are thwarted.

#### 3. Encryption of Passwords and Sensitive Data

- Passwords and sensitive data stored in the database are encrypted to prevent unauthorized access. Utilizing industry-standard encryption algorithms, such as bcrypt, ensures that even if the database is compromised, the data remains unintelligible to malicious actors.

#### 4. API Middleware for Security

- Specialized middleware has been implemented for the API endpoints to prevent unexpected hacking attempts. These middleware functions validate incoming requests, sanitize input data, and enforce data validation rules to mitigate the risk of API-based attacks, such as SQL injection or cross-site scripting (XSS).

By integrating these security measures into the platform's architecture, the system maintains a high level of protection against potential threats, safeguarding the integrity, confidentiality, and availability of greenhouse monitoring data.

## 3.2 Base Station

### 3.2.1 Introduction

The Base Station serves as the central nerve of the IoT greenhouse monitoring system, managing the intricate relationships between sensor and actuator nodes while facilitating seamless communication with the web platform. Its pivotal role encompasses the collection, aggregation, and transmission of environmental data, enabling real-time control and adjustment of greenhouse conditions. Acting as a centralized communication hub, the Base Station bridges the physical and digital realms, ensuring fluid information flow and system coherence through adept management of communication protocols. Its reliability and robustness lay the foundation for optimized cultivation practices, enhanced productivity, and sustainable agricultural outcomes within the greenhouse environment.

### 3.2.2 Base Station Architecture

The architecture is illustrated in this figure:

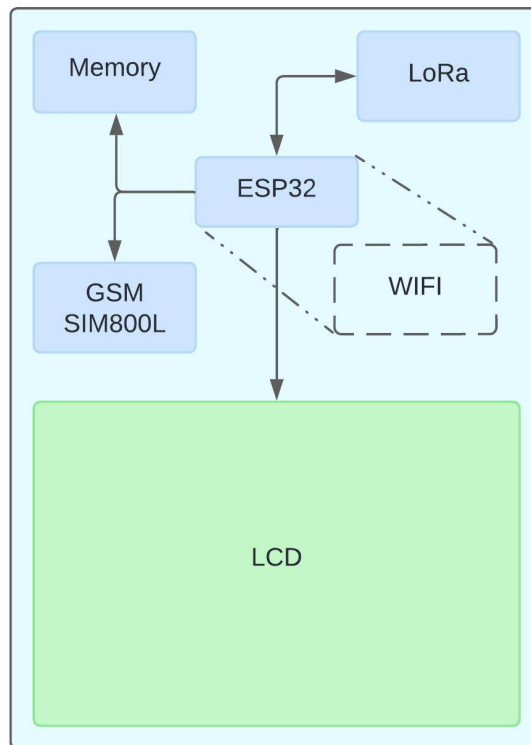


Figure 3.9: Base Station Architecture

### 3.2.3 Description of the Base Station Components

#### 3.2.3.1 ESP32

- **Function:** The ESP32 microcontroller is the central unit of the base station, responsible for managing data from various peripherals, executing control algorithms, and facilitating communication.
- **Features:** It includes dual-core processing capabilities, integrated Wi-Fi, and multiple I/O pins for interfacing with other components.

#### 3.2.3.2 Memory

- **Function:** Provides storage for data logging.
- **Connection:** Directly connected to the ESP32, it enables the system to store large amounts of data locally, which is crucial for offline cases or scenarios with intermittent connectivity.

#### 3.2.3.3 LoRa Module

- **Function:** Enables long-range, low-power wireless communication.

- **Connection:** Connected to the ESP32, it allows the base station to communicate with remote nodes over long distances, making it suitable for wide-area network applications.

#### 3.2.3.4 GSM SIM800L Module

- **Function:** Provides cellular communication capabilities.
- **Connection:** Connected to the ESP32, it facilitates data transmission over cellular networks, enabling remote monitoring and control when Wi-Fi is unavailable.

#### 3.2.3.5 Wi-Fi

- **Function:** Provides high-speed wireless communication.
- **Connection:** Managed by the ESP32's integrated Wi-Fi capabilities, it allows the base station to connect to local Wi-Fi networks for data transmission and remote access.

#### 3.2.3.6 LCD

- **Function:** Displays real-time data and system status.
- **Connection:** Connected to the ESP32, it provides a user-friendly interface for monitoring system performance and viewing data without needing an external device.

### 3.2.4 System Workflow

#### 1. Collect Data and Control Actuators

- **Data Collection:** The base station collects data from various sensor nodes using the LoRa module. These sensor nodes could be measuring parameters like temperature, humidity, soil moisture, Co2, etc.
- **Control Actuators:** The base station also sends commands to actuator nodes (e.g., irrigation systems, motors) to control their states based on the predefined rules or the commands of the client in the platform.

#### 2. Send Data to the Platform

- **Wi-Fi Communication:** If a Wi-Fi network is available, the ESP32 uses its integrated Wi-Fi capabilities to send the collected data to a remote server or cloud platform for centralized monitoring and analysis.
- **GSM Communication:** If Wi-Fi is not available, the ESP32 switches to using the GSM SIM800L module to send the data via cellular networks. This ensures continuous data transmission even in remote areas without Wi-Fi coverage.

#### 3. Store Data in Memory

- The ESP32 stores the collected and processed data in its connected memory module. This local storage acts as a backup, ensuring data is not lost during transmission failures and can be used for later analysis.

#### 4. Display Data on the LCD

- The ESP32 drives an LCD screen to display real-time data and system status. This allows users to monitor the performance and data metrics directly from the base station without needing additional devices.

##### 3.2.4.1 Key Features

- **Redundancy in Communication:** Ensures continuous data transmission through dual communication options (Wi-Fi and GSM).
- **Local Data Backup:** Prevents data loss by storing data locally in memory.
- **Real-Time Monitoring:** Provides immediate data visibility through an integrated LCD screen.
- **Versatile Control:** Supports both data collection from sensors and control of actuators, enabling comprehensive automation solutions.

##### 3.2.5 Base Station interface

Before designing the PCB for the base station, we mapped out all the components required, along with their respective communication protocols and voltage consumption. This step was essential for ensuring compatibility and efficient operation of the system, as depicted in the provided table 4.1.

Component	Operating Voltage	Communication Protocol
SD card	4.5V to 5.5V	SPI
GSM	3.2V to 4.5V	Digital cellular
LCD	3.3V to 5V	Analog
LoRa	3.3V	SPI

Table 3.1: List of Components

##### 3.2.6 PCB Design

After testing all the components, a PCB design was produced to connect the different elements, including the ESP32, transceiver module, SD card, GSM module, and LCD display, as shown in Figure 3.10. :

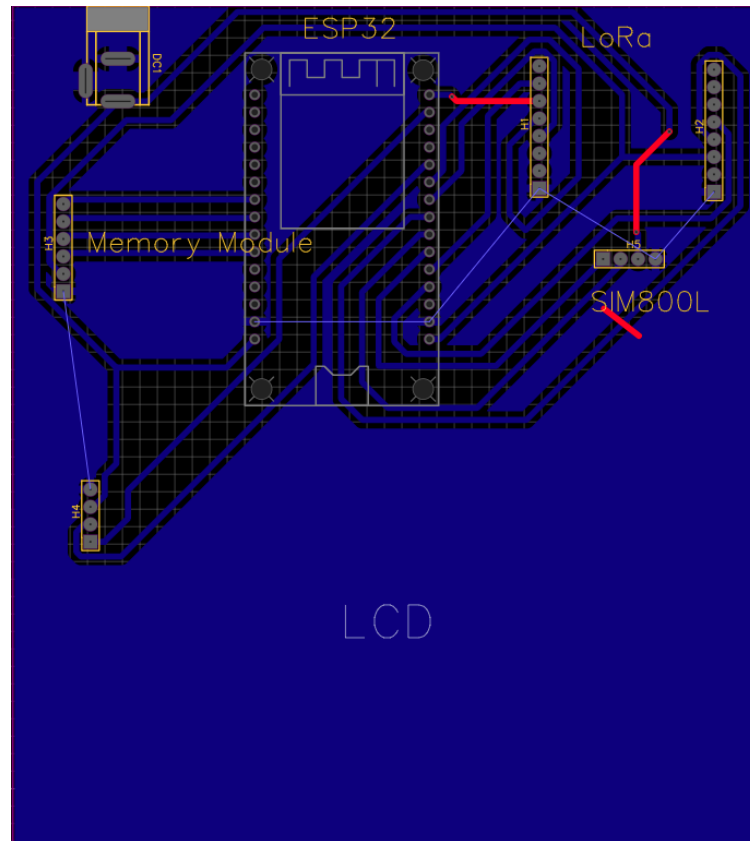


Figure 3.10: Base Station Circuit

In this circuit architecture, the ESP32 microcontroller acts as the brain. It is connected to the transmission module and the SD card for data storage in case of connection issues with the platform, ensuring that all data collected by the sensor nodes are saved. Additionally, there is a connection between the MCU and the GSM module, as well as another connection with the LCD display.

Close to the MCU, there is a 5V power input that supplies power to the MCU, the SD card, and the LCD. The transmission module is powered by the 3.3V output from the MCU. Lastly, the GSM module, which requires 3.7V, is powered by an external battery located beneath the LCD. Finally, it is presented a picture in Figure 4.3 taken to the solder PCB with all the components.

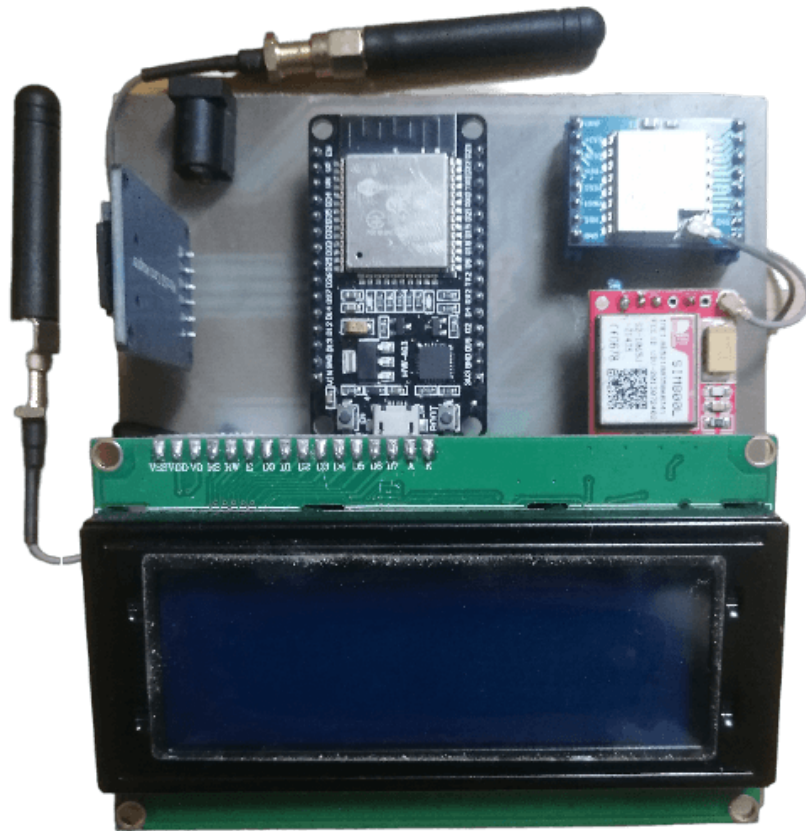


Figure 3.11: Solder PCB

## 3.3 Node Architecture

### 3.3.1 Introduction

In the market, a variety of node types are available for use in precision agriculture. These nodes come with multiple input and output ports, support for both digital and analog sensors, and compatibility with different types of sensors, actuators, and communication protocols such as I2C, SPI, etc. Many systems also rely heavily on a single power source. Initially, we adopted this methodology for our work.

Each node in our system is equipped with several sensors and actuators, along with a microcontroller and a LoRa module for data transmission. An external power supply board powers all of these components. Below are the circuit diagrams we fabricated for the initial testing of our system, following the aforementioned methodology.

The Sensor and Actuator Nodes in the IoT greenhouse monitoring system play critical roles in data collection, control, and communication. Sensor Nodes collect environmental data such as temperature, humidity, and CO<sub>2</sub> levels, while Actuator Nodes execute control actions such as irrigation and ventilation based on commands from the Base Station. Both

types of nodes rely on efficient communication protocols to interact with the Base Station, ensuring real-time monitoring and control of greenhouse operations. Together, they form an interconnected network that optimizes cultivation practices, enhances productivity, and promotes sustainable agriculture.

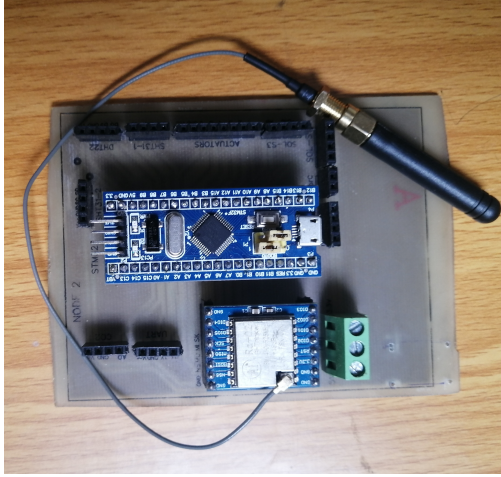


Figure 3.12: Actuator and Sensor nodes

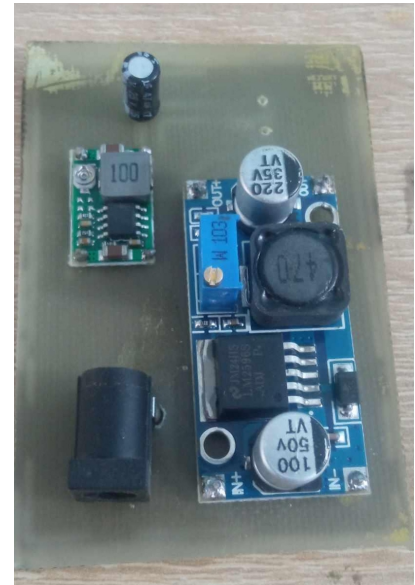


Figure 3.13: Power circuit

As depicted, the circuit of the nodes (Figure 3.12) is on the right side, while the circuit of the power (Figure 3.13) is on the left. The power circuit includes a DC buck converter to reduce the voltage from 12V to 5V for sensor power supply. Additionally, a mini DC buck converter is employed to further reduce the voltage from 5V to 3.3V for powering the microcontroller and sensors operating at 3.3V.

However, a significant challenge we encountered was energy consumption. To address this, we decided to separate the sensor and actuator nodes. Given the greenhouse's requirement for multiple sensor nodes and only a few actuator nodes, this approach helped reduce energy consumption and overall node costs.

### 3.3.2 Type of nodes

#### 3.3.2.1 Sensor nodes

The main role of sensor nodes is to collect environmental data and transmit it to the base station. A single greenhouse can contain multiple sensor nodes to ensure precise measurements and detect climate changes within the greenhouse. The hardware of the nodes includes:

- **Microcontroller:**
  - o STM32.
- **Communication Module:**
  - o LoRa 1278 module enables long-range, low-power communication with sensor nodes.



**- Sensors:**○ **Temperature:**

- \* SHT31: Supports I2C protocol communication.
- \* DHT22: Digital temperature and humidity sensor.

○ **Gas Sensor:**

- \* MQ-132: Capable of detecting various gases, including ammonia (NH<sub>3</sub>), nitrogen oxides (NO<sub>x</sub>), and benzene (C<sub>6</sub>H<sub>6</sub>).

○ **Soil Moisture Sensor:**

- \* Capacitive Soil Sensor: Measures soil moisture levels by detecting changes in capacitance caused by the presence of water in the soil.

○ **Light Sensor:**

- \* Ambient Light Sensor: Detects the amount of light present in the environment surrounding it.

**3.3.2.2 Sensors Interface**

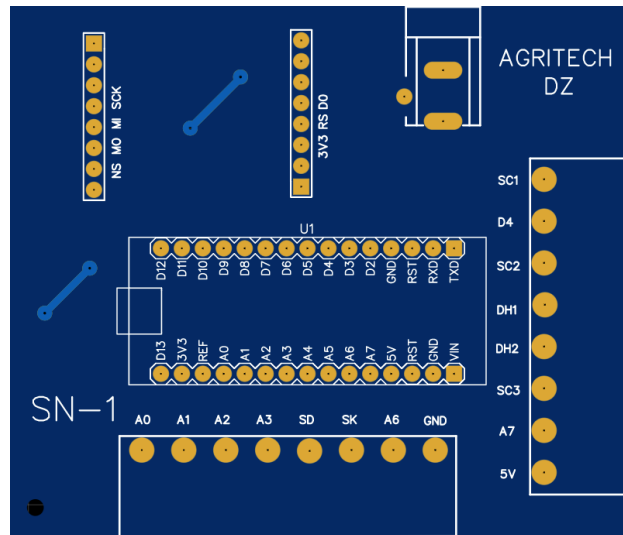
Since the microcontroller has various ports, both digital and analog, all types of sensors can be connected to the system. The only constraint is the voltage supply required by each sensor. The system provides only a 5V supply from the power source. Therefore, if a sensor requires a higher voltage, an external power supply will be necessary. For communication with the microcontroller, there are output pins for I2C, SPI, and digital pins that can be configured as outputs. For input, the system has comparators, analog pins, and digital pins that can receive or measure data from the sensors. As previously mentioned, this table presents some of the sensors that we used.

Sensor	Type of Sensor	Operating Voltage	Communication
MQ-135	Gas	3.3V to 5.5V	Analog
Capacitive Soil	Soil Moisture Sensor	3.3V to 5.5V	Analog
DHT22	Temperature and Humidity Sensor	3.3V to 5V	Digital
Light Sensor	Light Sensor	3.3V to 5.5V	Analog
SHT31	Temperature and Humidity Sensor	3.3V to 5.5V	I2C

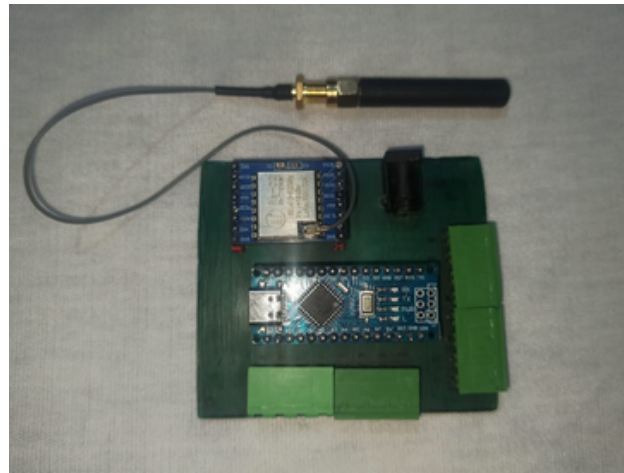
Table 3.2: List of sensors used in the system

**3.3.2.3 Node Design**

In order to connect the different sensors with a microcontroller and the LoRa module, a PCB design was produced.



(a) PCB Design



(b) Component Placement and Soldering

Figure 3.14: Sensor Node

### 3.3.3 Actuator nodes

The hardware setup for actuator nodes shares similarities with sensor nodes, albeit focusing on executing control actions rather than data collection. Actuator nodes are responsible for carrying out various control actions, such as adjusting irrigation levels or controlling ventilation systems, based on instructions received from the base station. The hardware of the nodes includes:

- **Microcontroller:**
  - o Arduino nano.
- **Communication Module:**
  - o LoRa 1278 module enables long-range, low-power communication with the base station.
- **Actuator:**

- Relay:Module Relay Shield 5V 2 Channels with Optocoupler

- **Power:**

- LM2596 DC to DC Buck Converter - Adjustable Step Down Power Supply Module

These components are connected as follow :

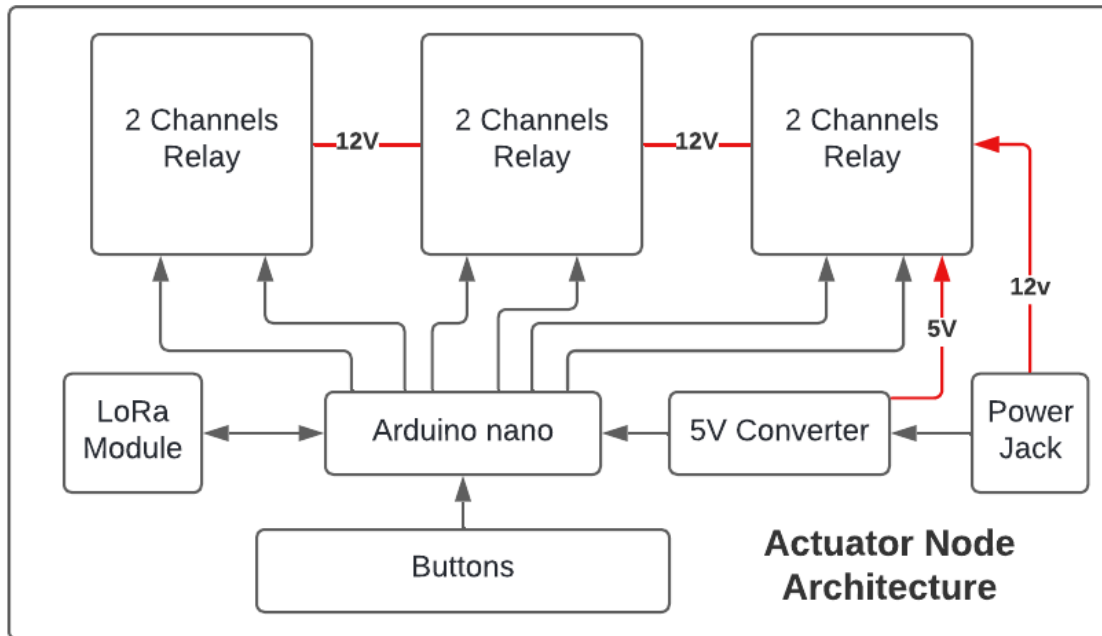


Figure 3.15: Actuator Node Architecture

### 3.3.3.1 Actuator Interface

Based on the conception of our system, we assembled the actuators into one system, with each relay connected to a specific actuator. For this purpose, we use a single 12V power supply for the entire system, and DC-DC converters to reduce the voltage as necessary. For communication, we use a LoRa module to send and receive the relay states from the base station. and the table above resume the actuator that we are used.

Actuator	Operating Voltage	Nominal Power
Solenoid Valve	11.4V to 12.6V	4.8W
Micro Diaphragm Pump	11.6V to 12.4V	60W
DC Submersible Water Pump	3.3V to 5V	3W
fan	220v	1000w

Table 3.3: List of actuators used in the system

### 3.3.3.2 PCB design

To control all actuators with a microcontroller, a PCB was designed and produced. Figure 5.4 presents a 3D image of the PCB, where all the different modules are identified.

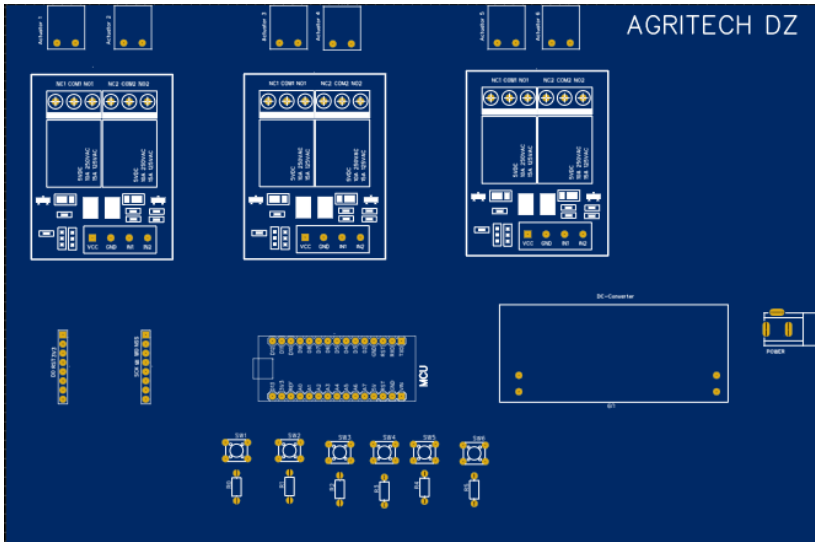


Figure 3.16: Actuators PCB design

The microcontroller serves as the system’s master, responsible for controlling the actuators. A receiver module is connected to the MCU, facilitating states’ exchange with the base station. The system incorporates 6 relays: four of which operate at 12V to power the actuators, one relay at 5V for a small pump, and the sixth relay controls a fan actuator at 220V using an external power supply. Additionally, six push buttons are provided for manual control when necessary, along with resistors and connectors for interfacing with the actuators.

We utilize a single jack and a DC-to-DC converter for the power supply. This allows us to reduce the voltage to 5V to power the circuit, excluding the various relays for the actuators. Finally, Figure 5.5 presents a picture of the soldered PCB with all the components.

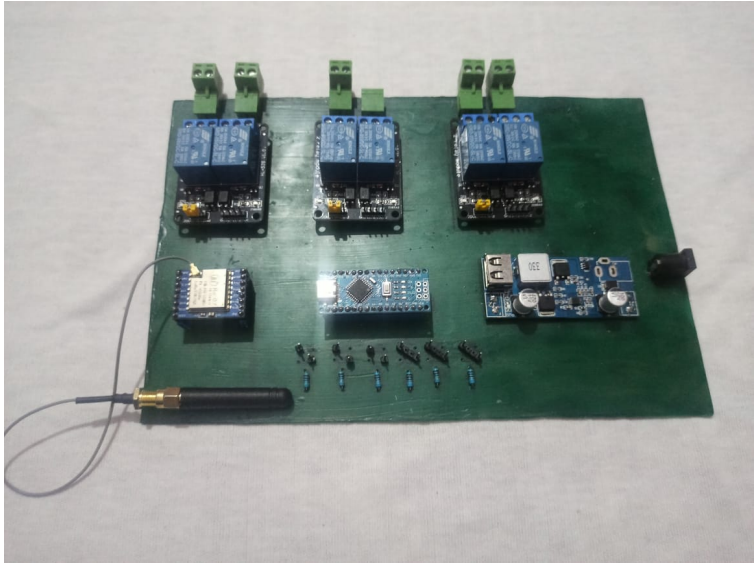


Figure 3.17: soldered PCB

### 3.4 Power management

As previously mentioned, the primary goal of this thesis is to develop an embedded system capable of operating for several years on a single battery charge. Consequently, optimizing power management is crucial.

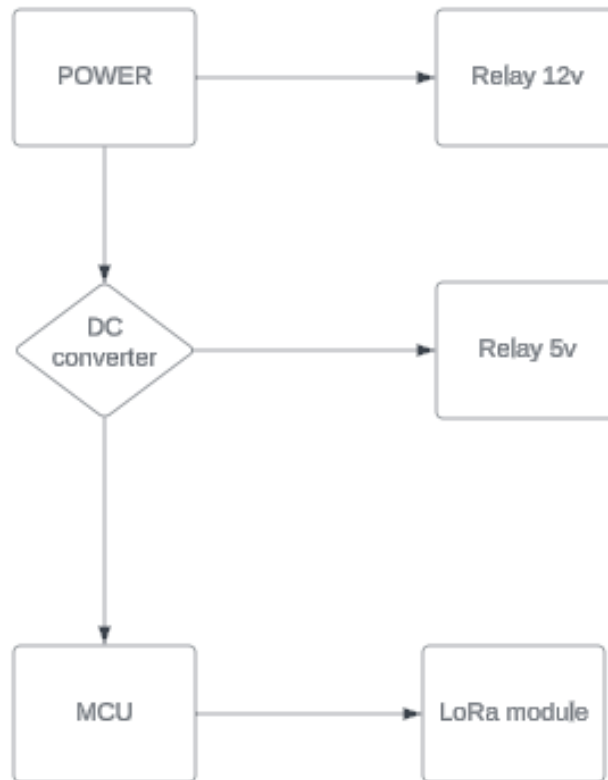


Figure 3.18: Actuator Power diagram block

Voltage regulators, which are essential for this optimization, perform various voltage translations. A voltage regulator is an electronic device designed to maintain a consistent output voltage. There are different types of regulators, such as DC-DC converters, which adjust direct current from one level to another. DC-DC converters can either step down a higher voltage to a lower voltage or step up a lower voltage to a higher voltage. The former is known as a buck converter as clearly illustrated in the sensor node Figure 5.6, while the latter is called a boost converter.

It is crucial to consider the efficiency of the regulators to be used, as well as their quiescent currents (the currents of the electronic system when there is no load at its output). Therefore, different options will be analyzed in terms of efficiency and quiescent current to select the best one for each regulator required by the system.

When integrating solar energy into battery-powered systems for sensor nodes Figure 5.5, one question that arises is whether to use voltage regulators. Many systems opt to avoid voltage regulators and instead supply power directly from the battery. This approach reduces the system's overall power consumption, as it eliminates losses and quiescent currents associated

with the regulators. However, without voltage regulation, the system experiences voltage fluctuations, ranging from the battery's fully charged higher voltage level to its lower voltage level as the battery discharges. This means that when the battery is fully charged, the system operates at a higher input voltage. Conversely, using regulators to convert the battery voltage to a lower, as more stable voltage allows the system to operate with consistent, lower input voltages. Consequently, the quiescent current is reduced, and the system's operation requires less energy.

Here is a brief demonstration of the solar power supply for the sensor nodes using a 3W solar panel, a voltage regulator, and a 12V battery. To power the sensor nodes, we added a DC-to-DC converter to step down the voltage to 5V, which then powers the nodes.

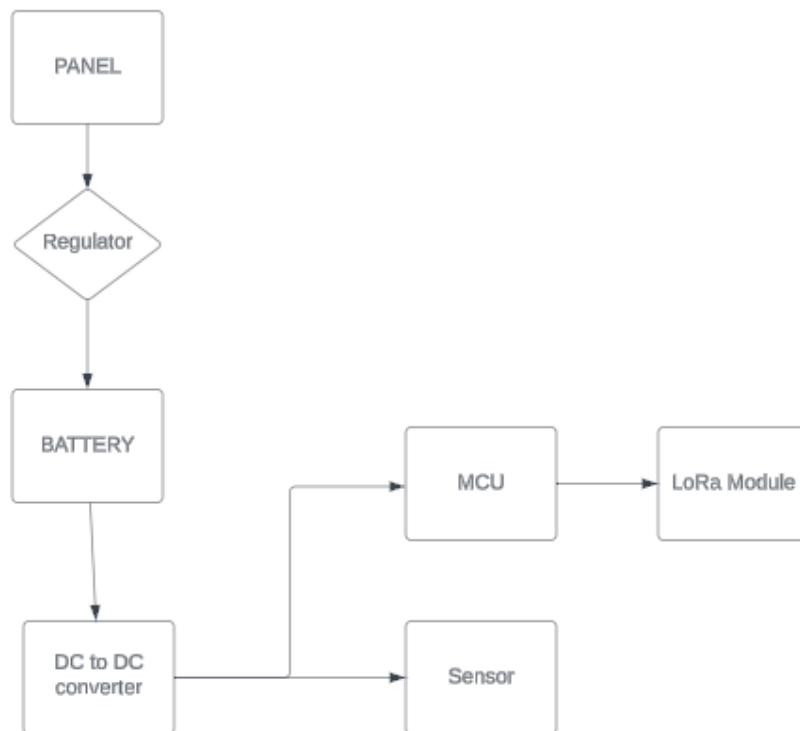


Figure 3.19: Solar System Diagram

Finally, to compare the results between solar power supply and grid power supply, here is a comparison of the consumption between the two methods, where the total current consumption is approximately 138 mA for both methods.

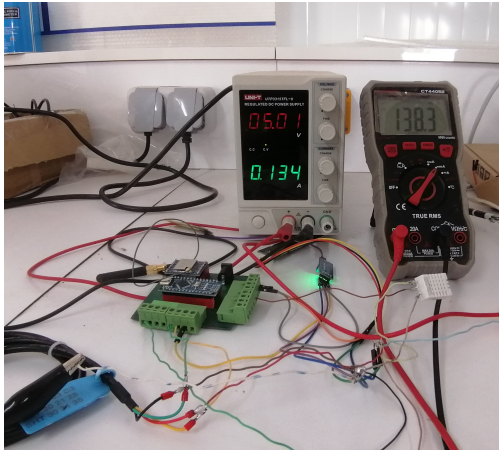


Figure 3.20: grid Power



Figure 3.21: Soler Power

### 3.5 Conclusion

In this chapter, we delved into the architecture and power management strategies of sensor and actuator nodes within our IoT-based greenhouse monitoring system. We meticulously detailed the design and implementation of these nodes, highlighting essential components such as microcontrollers, sensors, communication modules, and power circuits.

Our system employs distinct node types tailored for specific functions: sensor nodes gather environmental data, while actuator nodes execute control tasks based on commands from the base station. Both types of nodes feature efficient communication protocols to ensure seamless real-time monitoring and control.

Effective power management is pivotal for our system's long-term sustainability. We explored the use of voltage regulators and the trade-offs between direct battery supply and regulated power. Incorporating solar energy introduced additional considerations, including optimizing power consumption and system efficiency through voltage regulation or direct battery supply.

Our system aims to bolster the efficiency and sustainability of agricultural practices, aligning with the goals of precision agriculture and environmental conservation through meticulous design and strategic power management.

The base station architecture serves as the central nerve center, managing data collection, actuator control, and external communications in the IoT ecosystem. Key components such as the ESP32 microcontroller, memory module, LoRa module, GSM module, optional Wi-Fi, and LCD display synergize to ensure robust data handling, storage, transmission, and real-time display under diverse environmental conditions.

Workflow involves data collection from sensor nodes, actuator control based on predefined rules or user commands, data transmission via Wi-Fi or GSM, local data storage for redundancy, and real-time information display on the LCD screen. The architecture boasts redundancy in communication, local data backup, real-time monitoring, and versatile control capabilities.

The PCB design optimally integrates these components, offering a compact and efficient solution for base station construction. Overall, the base station architecture and PCB design are pivotal in guaranteeing the efficacy, reliability, and scalability of our IoT system for agricultural monitoring and control.

In conclusion, our developed platform stands as a cornerstone within the IoT greenhouse monitoring system, serving as a central hub for data aggregation, visualization, and control. Empowering users with intuitive interfaces, real-time and historical data access, and remote actuator control functionalities, the platform facilitates informed decision-making in greenhouse management.

Security measures, including authentication layers, middleware for route protection, data encryption, and API security, safeguard the integrity and confidentiality of greenhouse monitoring data from potential threats.

Moreover, the platform's scalability, facilitated by its modular design, cloud integration, and API accessibility, positions it for future advancements. Integration with advanced analytics and machine learning capabilities promises predictive analytics and intelligent decision support systems to optimize greenhouse operations further.

In essence, our platform embodies innovation and efficiency, propelling the capabilities of IoT-enabled agriculture. As it evolves to meet evolving greenhouse management needs, it remains steadfast in promoting sustainable and productive crop cultivation practices.



# Chapter 4

## Communication Protocol

### 4.1 Introduction

In modern IoT systems, efficient communication protocols are essential for ensuring seamless interaction between various devices such as sensors and actuators. This document outlines a specific communication protocol designed to manage these interactions using different types of network interfaces, namely LoRa, Wi-Fi, and GSM. The protocol facilitates data and command exchange with a central platform, ensuring reliable and continuous operation. Below are the details of the protocols used in this work, encompassing the initialization phase, main operation phase, recurrent operations, data handling, and manual and update operations.

### 4.2 Communication within the System

The communication within the system consists of two distinct parts:

- communication between MCU with different component
- communication between nodes and platform

and in this chapter, we will explain each communication and the way that we are used in our system

#### 4.2.1 SPI Protocol

The microcontroller communicates with both the LoRa module and the SD card using the Serial Peripheral Interface protocol. SPI is a synchronous serial communication protocol that enables high-speed data exchange between the MCU and peripheral devices

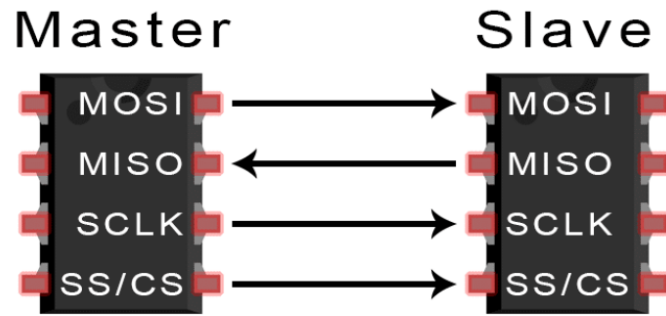


Figure 4.1: SPI Communication Protocol

#### 4.2.1.1 SPI Communication Setup

- **SCK**: The MCU provides a clock signal to synchronize data transfer.
- **MOSI**: Data line for sending data from the MCU to the peripheral devices.
- **MISO**: Data line for receiving data from the peripheral devices to the MCU.
- **CS**: Separate lines for each device (LoRa module and SD card) to select which device is currently communicating with the MCU.

#### 4.2.1.2 Data Flow

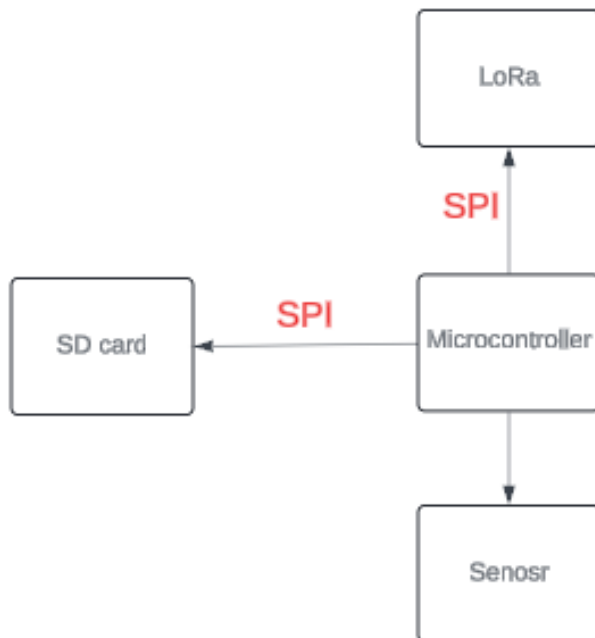


Figure 4.2: SPI connection between the microcontroller,SD card, and LoRa Module

The MCU initializes SPI communication by configuring the clock polarity, clock phase, and data order. When data needs to be sent to or received from the LoRa module or solely sent to the SD card, the MCU selects the respective device by setting its CS line low. Subsequently, data is transferred over the MOSI or MISO lines, synchronized with the clock signal. Once the data transfer is finished, the MCU sets the CS line high, deselecting the device.

## 4.2.2 I2C Protocol

I2C is a serial communication protocol for short-distance communication among multiple devices on the same PCB. It employs a two-wire interface consisting of SDA and SCL signals. Known for its low pin count, flexibility, and robustness, I2C enables multiple devices to communicate with a single controller or facilitates communication among multiple controllers and devices. In our project, we employed I2C software to manage multiple SHT31 sensors due to the limitation of both the STM32 and Arduino Nano microcontrollers, which support only one hardware-based I2C communication protocol.

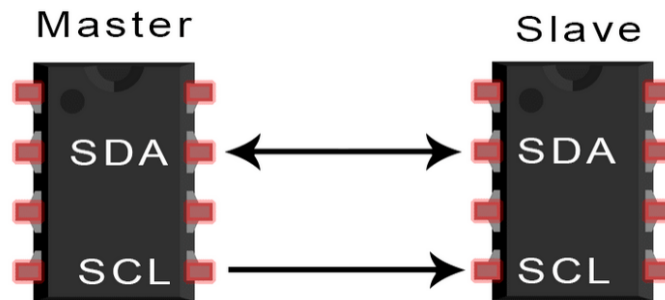


Figure 4.3: I2C Communication Protocol

### 4.2.2.1 I2C Softawar

We utilized the `<SoftwareI2C.h>` library available on the Arduino IDE to implement this method. Following the configuration of the SHT31 sensors, the necessary libraries were included:

- `#include <SoftwareWire.h>`
- `#include "SHT31_SW.h"`

We assign a unique address to each SHT31 sensor and connect them to the MCU as illustrated in the configuration below:

Listing 4.1: C++ code I2C Softwar

```
#define SHT31_ADDRESS 0x44
SoftwareWire sw1(6, 7);
SoftwareWire sw2(8, 9);

SHT31_SW sht1, sht2;
bool sht1_status = false, sht2_status = false;
```

### 4.3 General communication

For the general communication protocol of the system, we have attempted to outline the functional flow of the different parts, from the system initialization to the execution of the algorithm and data collection, as shown in the figure below. Additionally, we provide a description of each part of the synoptic diagram:

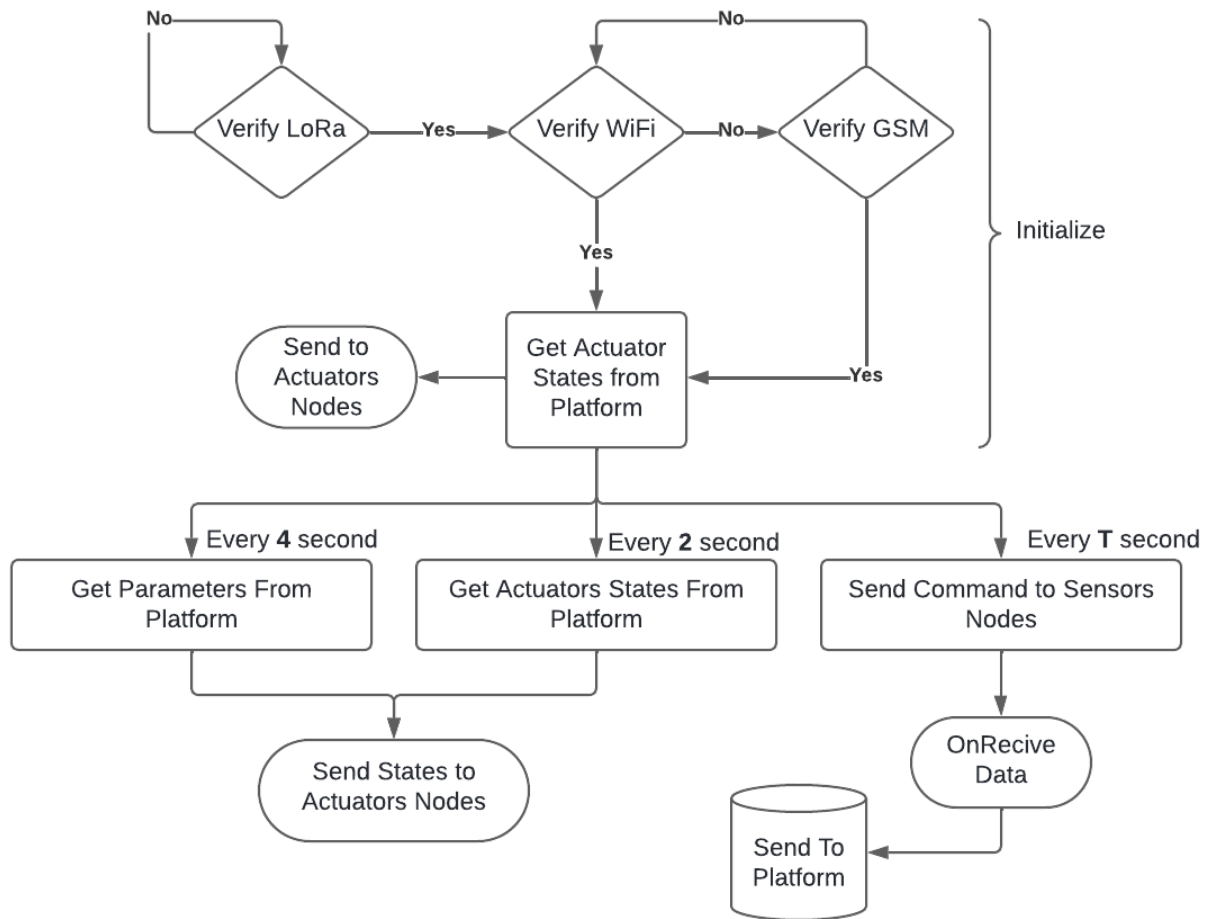


Figure 4.4: Communication Protocol

#### 4.3.1 Initialization Phase

The initialization phase begins by verifying the availability of the LoRa interface. If LoRa is available, the process proceeds to the next step. If LoRa is not available, the system attempts to verify the availability of WiFi. If WiFi is available, the process continues; if not, the system then verifies the availability of GSM. If GSM is available, the initialization process moves forward. If none of these interfaces are available, the initialization fails.

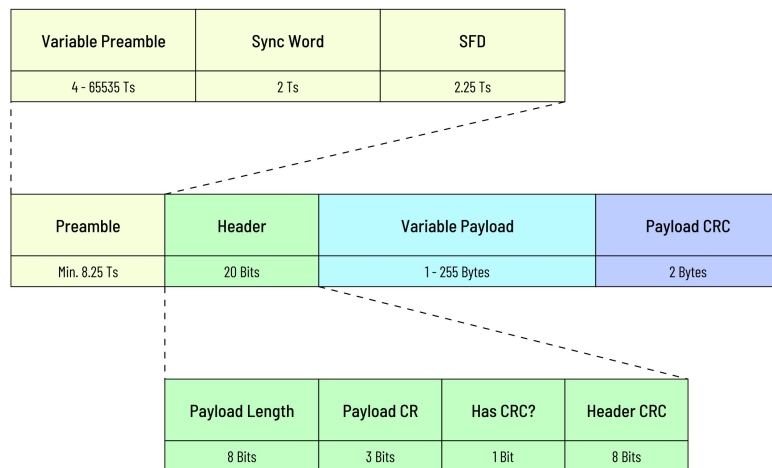


Figure 4.5: LORA PACKET FORMAT [23]

### 4.3.2 Main Operation Phase

During the main operation phase, the system retrieves the current states of the actuators from the central platform. Once these states are obtained, they are sent to the actuator nodes to ensure they have the latest information.

### 4.3.3 Recurrent Operations

Every four seconds, the system retrieves various operational parameters from the platform and sends these states, including the parameters, to the actuator nodes. Additionally, every two seconds, the system updates the actuator states from the platform to maintain up-to-date information. Furthermore, T is a parameter get from the platform and every T seconds, commands are sent to sensor nodes for data collection.

#### 4.3.3.1 How Command sent to Sensor Node:

##### LORA PACKET FORMAT:

- **The preamble** :is used to detect the start of the packet by the receiver.
- **The header** :(only in explicit header mode) is the default mode of operation.
  - o Destination Address.
  - o Source Address.
  - o Message Id.
  - o The payload length in bytes.
  - o The forward error correction code rate.
  - o The presence of an optional 16 bits CRC for the payload.

- **The payload** :is a variable-length field that contains the actual data coded at the forward error correction code rate either as specified in the header in explicit mode or fixed in implicit mode. An optional payload CRC may be appended

**Code Of the Base Station :**

The following function, `sendMessage`, is used to send a message via a LoRa (Long Range) communication module. Here is a brief explanation of what each line does:

```
void sendMessage(String Message, byte MasterNode, byte Node) {
    LoRa.beginPacket();           // start packet
    LoRa.write(Node);             // add Node address
    LoRa.write(MasterNode);       // add Base Station address Which is 0xFF
    LoRa.write(msgCount);         // add message ID
    LoRa.write(Message.length()); // add payload length
    LoRa.print(Message);          // add payload
    LoRa.endPacket();             // finish packet and send it
    msgCount++;                   // increment message ID
}
```

- **Function Definition:**

Defines a function named `sendMessage` that takes a `String` message and two `byte` values representing the addresses of the master node and the sensor node.

- **Start Packet:**

Initializes the beginning of a LoRa packet.

- **Add Sensor Node Address:**

Writes the address of the sensor node to the packet. This address specifies the intended recipient of the message.

- **Add Master Node Address:**

Writes the address of the master node (base station), typically set to `0xFF` for broadcast.

- **Add Message ID:**

Adds a message ID to the packet, which can be used for tracking or distinguishing messages.

- **Add Payload Length:**

Adds the length of the message payload, indicating how many bytes the message contains.

- **Add Payload:**

Writes the actual message content (payload) to the packet.

- **Finish Packet and Send It:**

Finalizes the packet and sends it over the LoRa network.

- **Increment Message ID:**

Increments the message ID counter, preparing it for the next message to be sent.

## Code Of The Node :

The following function, `onReceive` is called in the loop function and it is designed to handle the reception of data packets using a LoRa (Long Range) communication module. Here is a brief explanation of what each part of the code does:

```
void onReceive(int packetSize) {
    if (packetSize == 0) return;           // if there's no packet, return

    // read packet header bytes:
    int recipient = LoRa.read();           // recipient address
    byte sender = LoRa.read();             // sender address
    byte incomingMsgId = LoRa.read();      // incoming msg ID
    byte incomingLength = LoRa.read();     // incoming msg length

    String incoming = "";

    while (LoRa.available()) {
        incoming += (char)LoRa.read();
    }
}
```

- **Function Definition:** `void onReceive(int packetSize)` defines a function named `onReceive` that takes an integer parameter `packetSize`, which represents the size of the received packet.
- **Check for Empty Packet:** If `packetSize` is 0 (indicating no packet was received), the function returns immediately without doing anything.
- **Read Packet Header Bytes:** These lines read the first few bytes of the packet, which contain the header information:
  - o **recipient:** Address of the intended recipient of the packet.
  - o **sender:** Address of the sender of the packet.
  - o **incomingMsgId:** ID of the incoming message.
  - o **incomingLength:** Length of the incoming message payload.
- **Read and Store Incoming Message:** Initializes an empty `String` named `incoming` to store the message content. The `while (LoRa.available())` loop reads each byte of the incoming message from the LoRa buffer and appends it to the `incoming` string, converting each byte to a character.

## 4.3.4 Data Handling

When data is received from sensor nodes, it is promptly sent to the platform to ensure that the central system has the latest data for processing and decision-making.

Each time the Actuator States data is received from the Platform, it is sent to the Actuator States to update the states.

### 4.3.4.1 Message Format

The format of the messages sent and received is crucial for efficient communication. The data is formatted as strings to minimize the number of characters sent, reducing the likelihood of errors.

**Sensor Data Format:** For sensor data, the format is as follows:

Sensor 1	Comman	Sensor 2	Comman	Sensor 3	Comman	Sensor 4	Comman
25	,	10	,	30	,	-	,




Figure 4.6: Sensor Message Format

Each sensor data value is followed by a comma ','. This data is then decoded at the base station and sent to the platform.

**Actuator Node Data Format:** For actuator node data, after receiving the data from the platform, it is encoded in this format:

GreenhouseID	Comman	Actuator Name	Comman	State	semicolon	GreenhouseID	Comman
1	,	heater	,	1	;	1	,




Figure 4.7: Actuator Message Format

The format consists of:

- The first number after each semicolon ';' indicates the ID of the greenhouse.
- Followed by a comma ',' and the actuator name.
- Followed by another comma ',' and the state of the actuator: 0 for off, 1 for on.

This data is then decoded at the actuator node to change the states of the actuators.

**Reason for the Format:** The primary reason for using this format is to reduce the number of characters sent. The smaller the message, the lower the error rate during transmission.

### 4.3.5 Manual and Update Operations



Figure 4.8: Communication Protocol

Manual interaction with an actuator node is possible through a button press, which triggers an update. The base station acts as an intermediate step for sending data to the platform. Finally,



the system ensures that updated states are sent to the platform and that the state information is consistently updated.

The push buttons are also used for direct control of the relays on-site. For the system, they are considered as an interruption that will be handled by the MCU, and the state will be updated on the platform.

## 4.4 Conclusion

In this chapter, we detailed the communication protocols utilized within our IoT system, focusing on the efficient interaction between various components and the central platform. The SPI protocol is employed for high-speed data exchange between the MCU, LoRa module, and SD card, while the I2C protocol, implemented through software, enables seamless communication with multiple SHT31 sensors despite the hardware limitations of the STM32 and Arduino Nano microcontrollers.

We outlined the system's communication workflow, including the initialization phase, main operation phase, recurrent operations, data handling, and manual and update operations. These processes ensure reliable and continuous data exchange, maintaining up-to-date actuator states and sensor data on the platform.

Overall, the robust communication setup is crucial for the system's functionality, enabling efficient data collection, processing, and control in an IoT environment. This chapter underscores the importance of selecting and implementing appropriate communication protocols to achieve seamless integration and reliable operation within the system.

# Chapter 5

## Results, Challenges and Limitations

### Results

After designing and producing the proposed embedded system, it underwent testing to evaluate its performance and power consumption. The system was integrated into a LoRaWAN network, enabling the transmission of sensor data to the platform. Additionally, the system was tested for its ability to control actuators and its performance in collecting real-time data.

### 5.1 Verification and Validation

Following the soldering of components, basic tests were conducted to assess the various parts of the system. The initial test involved powering up the system and verifying the presence of all required voltages. Using a multimeter, we checked the voltages at the integrated circuits. Subsequently, we attempted to program the microcontroller with a basic program to ensure its proper functioning. We then developed the platform to verify the successful transmission of data to the platform and tested the control commands of the relay.

### 5.1.1 Sensor Node Validation

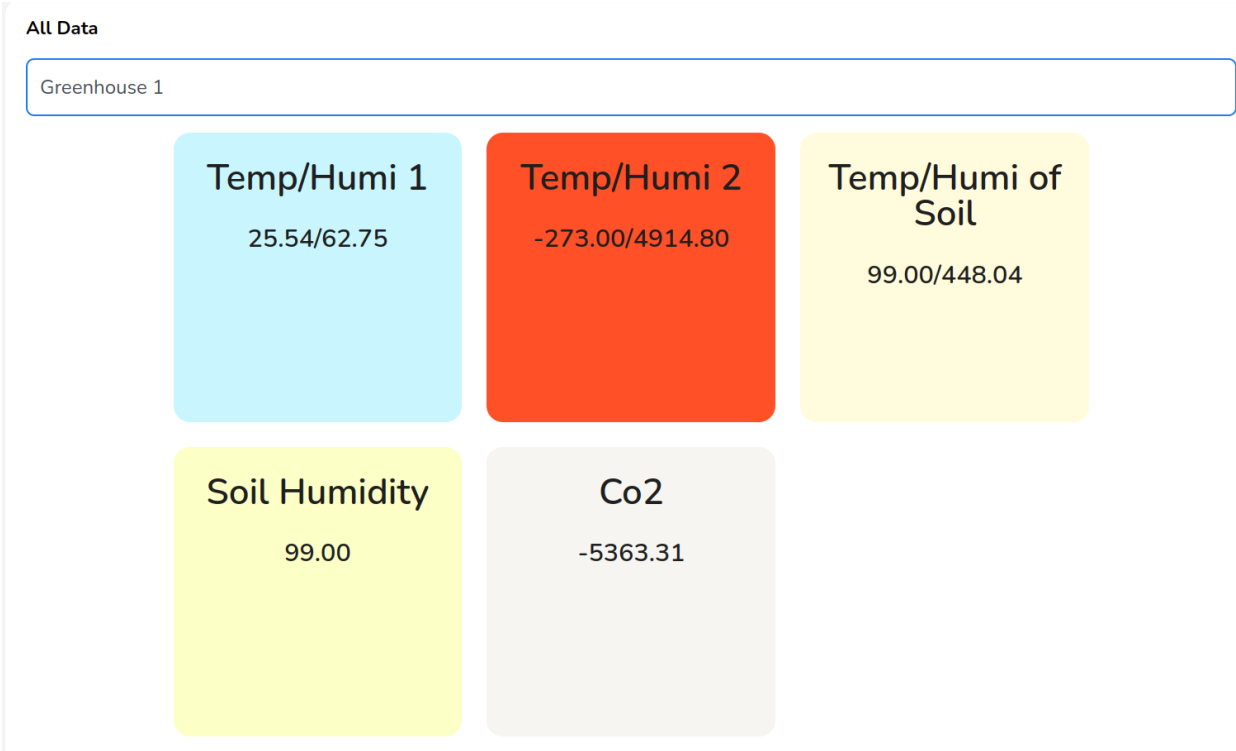


Figure 5.1: Data Collecting the Platform

To evaluate the real-time data collection from the sensor nodes, we connected all sensors to the system and verified the transmission between the base station, platform, and sensor nodes. Figure 8.1 shows the data collection in the Serial Monitor of the base station, while Figure 8.2 illustrates the data collection on the platform.

- The sensors are not all calibrated and some of them are damaged, so the data are not well.

## 5.1.2 Actuator Node Validation

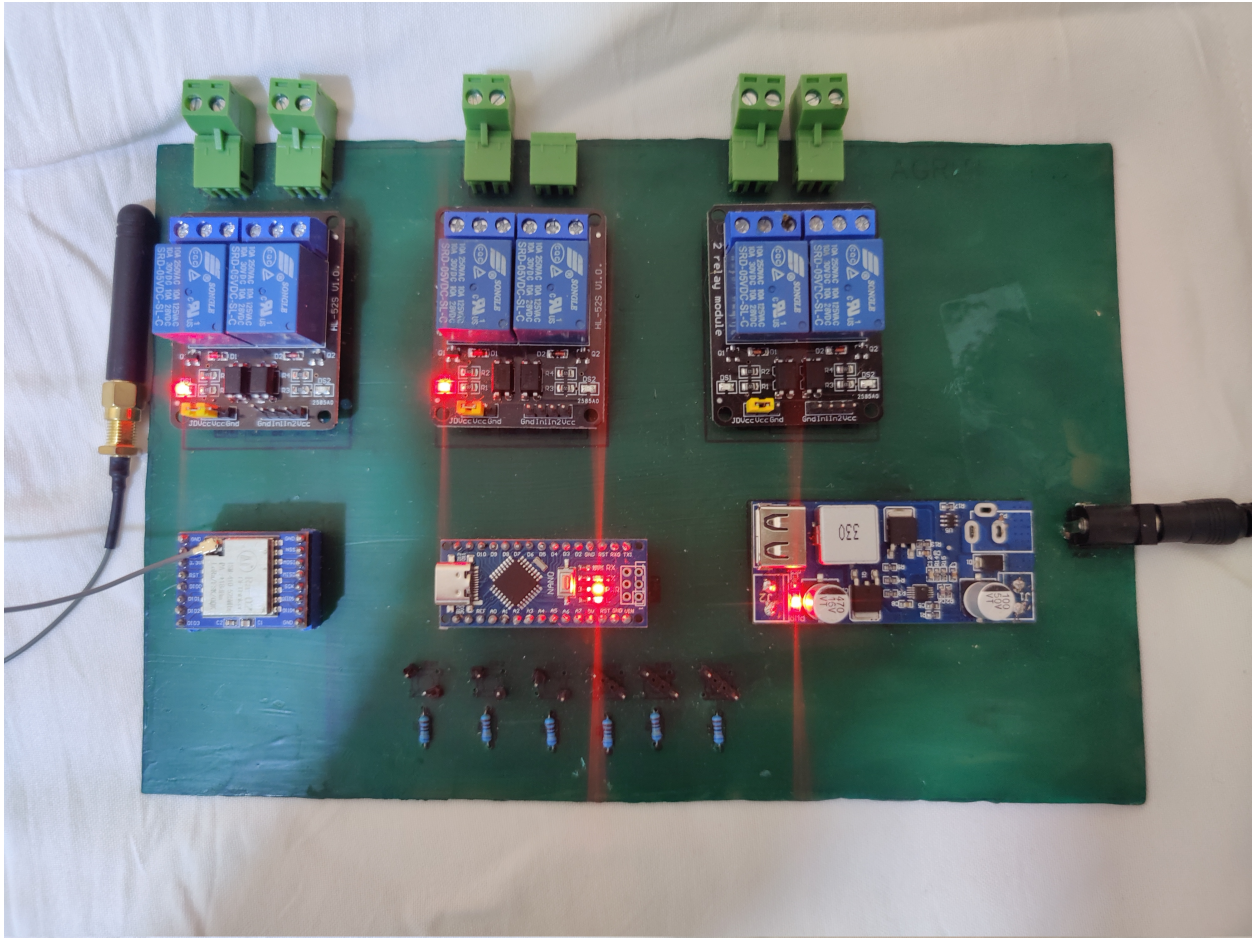


Figure 5.2: Actuator Stat

**Note :**

- Led Turn ON means the actuator is OFF.
- Led Turn OFF means the actuator is ON.

To validate our architecture, we connected all the components and verified the remote control of the actuator from the platform. Figure

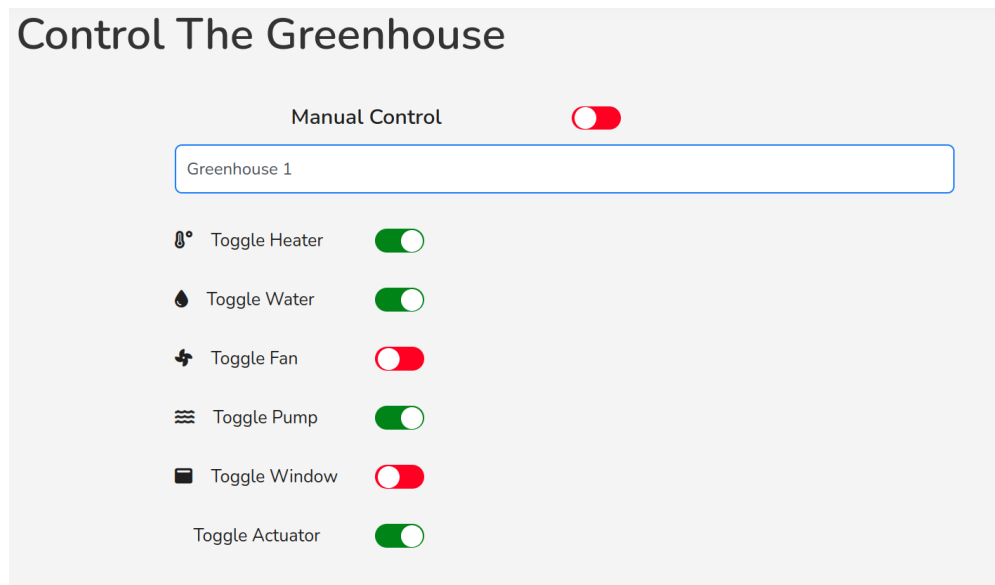


Figure 5.3: Controlling the Actuator from the Platform

## Challenges and Limitations

The development and implementation of the IoT system were accompanied by several challenges that impacted its performance and usability. Addressing these challenges was crucial for ensuring the system's reliability and effectiveness. This section highlights the key issues encountered during the project, providing a detailed analysis of each problem along with potential solutions. By understanding these challenges, we can better appreciate the complexities involved in IoT system development and identify areas for future improvement.

### 5.2 Time Controlling Issues

One of the significant challenges was managing the timing of tasks to avoid conflicts. Specifically, when the base station sends commands to sensor nodes to collect data, it takes some time for the nodes to respond. This delay interferes with the task of sending actuator states to actuator nodes.

- **Task Interference:** Commands sent to sensor nodes and actuator nodes often intersect, causing delays and potential data loss or duplication.
- **Response Time:** The time it takes for sensor nodes to respond to data requests introduces latency, impacting the synchronization of tasks.

### Our Solutions:

One of the significant challenges in the system implementation was managing the timing of tasks to avoid conflicts. Specifically, when the base station sends commands to sensor nodes to collect data, it takes some time for the nodes to respond. This delay interferes with the task of sending actuator states to actuator nodes.

To address this challenge, the system utilizes tags and conditions to execute functions at appropriate intervals. Here's how it works:

Listing 5.1: LoRa Parameter Configuration

```
1
2 currentMillis = millis();
3 currentsecs = currentMillis / 1000;
4
5 // Check if the interval has passed for sending commands to sensor nodes
6 if ((unsigned long)(currentsecs - previoussecsSensor) >= intervalSensor) {
7     previoussecsSensor = currentsecs;
8     sendToSensorNodes = 1;
9 }
10
11 // Check if the interval has passed for sending actuator states to
12 // actuator nodes
13 if ((unsigned long)(currentsecs - previoussecsActuator) >=
14     intervalActuator) {
15     previoussecsActuator = currentsecs;
16     sendToActuatorNodes = 1;
17 }
18
19 // Check if the interval has passed for getting parameters
20 if ((unsigned long)(currentsecs - previoussecsParameter) >=
21     intervalParameter) {
22     previoussecsParameter = currentsecs;
23     GetParameters = 1;
24 }
25
26 // Execute functions based on conditions
27 if (GetParameters && !sendToSensorNodes && !sendToActuatorNodes) {
28     getParameters(); // Executes function to retrieve parameters
29     GetParameters = 0;
30 }
```

**Explanation:**

- **millis():** Returns the number of milliseconds since the Arduino board started running the current program.
- **currentMillis** and **currentsecs:** Store the current time in milliseconds and seconds, respectively.
- **Interval Checks:** The code checks if specific intervals have passed since the last execution of tasks related to sending commands to sensor nodes, sending actuator states to actuator nodes, and getting parameters.
- **Flags:** If the interval has passed, flags (**sendToSensorNodes**, **sendToActuatorNodes**, **GetParameters**) are set to 1.
- **Function Execution:** Based on the conditions, functions **getParameters()** are executed to perform tasks such as retrieving parameters, then the flag returned to 0.

By utilizing these tags and conditions, the system ensures that tasks are executed at appropriate intervals, minimizing conflicts and ensuring smooth operation.

## 5.3 LoRa Range Limitations

The range of the LoRa communication module was another significant limitation. In building areas, the range was between 37-45 meters, and in open areas, it extended to 120 meters. This limited range affected the coverage and effectiveness of the system.

### Potential Solutions:

- **Parameter Optimization:** Adjusting parameters such as spreading factor, bandwidth, and transmission power in the LoRa module's code can potentially increase the range.

Listing 5.2: LoRa Parameter Configuration

```
1
2 // Define the LoRa frequency
3 #define LORA_FREQUENCY 915E6 // Use the frequency appropriate for
   our region 433E6
4 // LoRa parameters
5 #define SPREADING_FACTOR 12 // Range: 6-12, higher values increase
   range but decrease data rate
6 #define BANDWIDTH 125E3 // Range: 7.8E3 to 500E3, lower values
   increase range but decrease data rate
7 #define CODING_RATE 5 // Range: 5-8, higher values increase
   reliability but decrease data rate
8 #define SYNC_WORD 0x12 // Default is 0x12, change to differentiate
   your network
9 // in the void fucntion
10 // Set LoRa parameters
11 LoRa.setSpreadingFactor(SPREADING_FACTOR);
12 LoRa.setSignalBandwidth(BANDWIDTH);
13 LoRa.setCodingRate4(CODING_RATE);
14 LoRa.setSyncWord(SYNC_WORD);
```

- **Antenna Upgrade:** Using higher gain antennas can significantly enhance the communication range.
- **Repeater Nodes:** Deploying repeater nodes to extend the range and ensure reliable communication in larger areas.

### 5.3.1 Static Nature of the Monitoring Platform

The current monitoring platform is built using PHP with the Laravel framework, which is static. This architecture does not support auto-refresh when new data is submitted through post requests from the base station, leading to delays in data updates on the user interface.

### Potential Solutions:

- **Use of WebSockets:** Implementing WebSockets can enable real-time communication between the server and client, allowing for immediate data updates.
- **Switch to Node.js:** Reprogramming the platform using Node.js can provide better support for real-time data handling due to its non-blocking, event-driven architecture.

## 5.4 Flash Memory saturation

When working with the STM32 in the Arduino IDE, memory saturation becomes a notable challenge. Despite the STM32's 32kB flash memory capacity, attempting to execute a program often triggers a memory saturation error within the Arduino IDE. This contrasts sharply with the Arduino Nano, where such errors are absent, and programs compile seamlessly. Furthermore, when utilizing the Arduino IDE with the STM32, it generates four distinct file types, including: **Program Files:**

- `program.ino` The main source file of an Arduino program is referred to as the primary source file.
- `program.ino.BLUEPILL_F103C6.bin` is a binary file of the Arduino program specific to the Bluepill\_F103C6 board.
- `program.ino.eightanaloginputs.hex` is a hexadecimal file containing the binary code of the Arduino program.
- `program.ino.with_bootloader.eightanaloginputs.hex` is a hexadecimal file containing the binary code of the Arduino program with a bootloader included.

To address this issue, several solutions have been proposed. Initially, we attempted to optimize the program's storage by selectively retaining only the necessary lines of code within each sensor library. While this method did reduce the program's storage requirements, it proved insufficient to fully resolve the problem. The explanation for this issue lies in the abstraction layers created when coding STM32 with the Arduino IDE. These abstraction layers add overhead to the code, resulting in increased storage demands.

### 5.4.1 Proposed solutions

**SPI Communication:** During this optimization phase, we configured the STM32 for SPI communication, consequently eliminating the "program.ino.with\_bootloader.eightanaloginputs.hex" file. Subsequently, we converted "program.ino.eightanaloginputs.hex" to a binary file and compiled the program.

**External Flash Memory:** As part of this solution, we considered the addition of an external flash memory to the STM32. This could involve selecting an appropriate flash memory module compatible with the STM32 microcontroller and integrating it into the system. Additionally, we could explore options for interfacing and accessing the external flash memory from the STM32, ensuring seamless integration and expanded storage capacity.

## 5.5 Future Work

The platform's architecture is designed with scalability in mind, allowing it to accommodate future enhancements and growing demands. As the needs of greenhouse monitoring evolve, the platform can adapt and expand to meet these requirements. Several aspects contribute to the scalability of the platform and pave the way for future developments:



- **Enhanced Task Scheduling:** Developing a more advanced task scheduling mechanism to manage timing conflicts effectively.
- **Improved Communication Range:** Exploring different antenna designs and optimizing LoRa parameters to extend the communication range.
- **Real-Time Monitoring Platform:** Rebuilding the monitoring platform using Node.js or incorporating real-time communication technologies like WebSockets to provide immediate data updates.
- **Modular Design:**
  - o The platform is built with a modular design, allowing for easy integration of new features and functionalities. New modules can be developed independently and seamlessly integrated into the existing system without disrupting its operation.
- **Cloud Integration:**
  - o Integration with cloud services enables the platform to leverage scalable infrastructure resources. Cloud-based solutions provide flexibility in resource allocation, allowing the platform to handle increased data volumes and user traffic efficiently.
- **API for Integration:**
  - o The platform exposes APIs that enable integration with third-party applications and services. This facilitates interoperability and allows for the incorporation of new technologies and data sources into the system, enhancing its functionality and relevance.
- **Data Analytics and Machine Learning:**
  - o Future developments may include the integration of advanced data analytics and machine learning algorithms. By leveraging the vast amount of data collected by the platform, predictive analytics and intelligent decision-making capabilities can be implemented to optimize greenhouse operations further.

Looking ahead, the platform will continue to evolve to meet the changing needs of greenhouse management. Continuous enhancements and updates will ensure that the platform remains at the forefront of IoT-enabled agriculture, empowering users with innovative tools and insights for efficient and sustainable crop cultivation.

## 5.6 Conclusion

The challenges encountered during this project provided valuable insights into the complexities of developing a robust IoT system. The system can achieve better performance, reliability, and scalability by addressing these issues through future improvements, making it more effective for real-world applications.

# Chapter 6

## General Conclusion

In conclusion, our project highlights the transformative role of modern data acquisition systems and IoT integration in agriculture, significantly enhancing real-time monitoring and analysis of environmental parameters. Through the careful selection and integration of advanced sensors and platforms, we have developed a precise, reliable, and efficient system.

Our developed platform serves as a central hub in the IoT greenhouse monitoring system, offering real-time data aggregation, visualization, and control. With a focus on security, scalability, and a user-friendly interface, it empowers users to make informed decisions, ensuring the integrity and confidentiality of data. The platform's modular design and integration with advanced analytics and machine learning open avenues for predictive analytics and intelligent decision support, further optimizing greenhouse operations.

The base station architecture, comprising components such as the ESP32 microcontroller and communication modules, ensures reliable data collection and control even in challenging environments. The PCB design facilitates efficient integration of these components, enhancing the system's effectiveness and scalability.

Power management strategies, including the use of solar energy, are critical for the sustainable operation of sensor and actuator nodes. Through careful design and strategic power management, our system promotes the efficiency and sustainability of agricultural practices.

The robust communication protocols ensure efficient data exchange within the IoT system, maintaining the reliability of data collection, processing, and control. The SPI and I2C protocols facilitate seamless communication between various components, ensuring continuous and accurate monitoring.

The challenges encountered during this project provided valuable insights into the complexities of developing a robust IoT system. Addressing these challenges will lead to better performance, reliability, and scalability of the system.

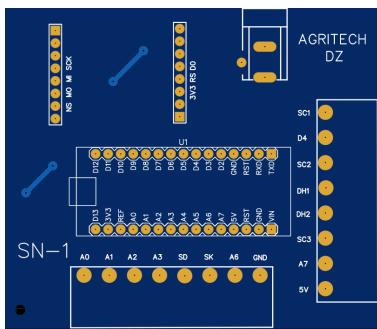
the project represents a significant step forward in IoT-enabled agriculture, driving innovation and efficiency. As the platform evolves, it will continue to support sustainable and productive farming practices, enabling greenhouse operators to maximize yields, minimize resource consumption, and navigate the complexities of modern agriculture with confidence.

# Chapter 7

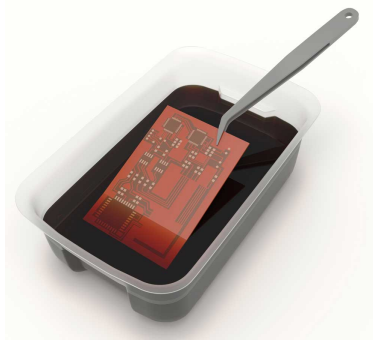
## Annex

### Node Design

In order to connect the different sensors with a microcontroller and the LoRa module, a PCB design was produced. In this section, we will explain each part of the design process.



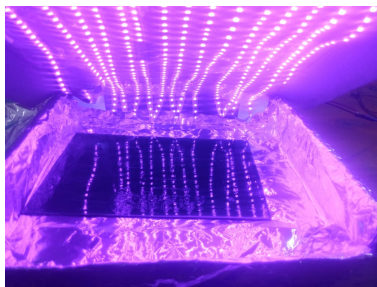
(a) PCB Design



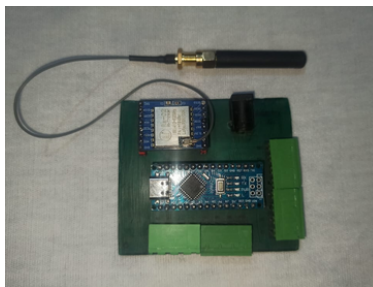
(b) Etching the PCB



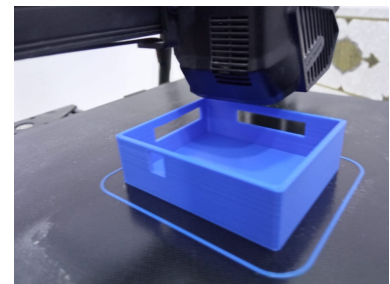
(c) Applying the Solder Mask



(d) UV Exposure and Development



(e) Component Placement and Soldering



(f) Enclosure Design

Figure 7.1: Steps to establish a node

1. **PCB Design:** The first step involves creating a detailed schematic and layout for the printed circuit board using EasyEDA software, which includes defining the electrical connections and component placements pawlowski2020greenhouse.
2. **Etching the PCB:** The design is transferred to a copper-clad board, which is then immersed in an acid solution to remove excess copper, forming the required electrical pathways

wolbert1998ldo.

3. **Applying the Solder Mask:** A solder mask is applied to protect the copper traces, leaving pads and holes exposed for soldering. This is done through applying a liquid photoimageable solder mask ti2015.
4. **UV Exposure and Development:** The PCB is exposed to ultraviolet light through a photolithographic mask to harden the solder mask in specific areas, followed by developing to remove the unhardened portions analog2014quadbuffer.
5. **Component Placement and Soldering:** Electronic components are placed on the PCB and soldered manually tagai2020smartnode.
6. **Enclosure Design:** The final step involves designing a protective enclosure for the PCB and its components, considering factors like mounting, accessibility, and heat dissipation, using CAD software and printing using a 3D printer intech2020.

# Bibliography

- [1] Yongchao Song, Jiping Bi, and Xuan Wang. Design and implementation of intelligent monitoring system for agricultural environment in iot. 2020.
- [2] Abdennabi Morchid, Rachid El Alami, Aeshah A. Raezah, and Yassine Sabbar. Applications of internet of things (iot) and sensors technology to increase food security and agricultural sustainability: Benefits and challenges. 2020.
- [3] Andrzej Pawlowski, Jose Luis Guzman, Francisco Rodríguez, Manuel Berenguel, José Sánchez, and Sebastián Dormido. Simulation of greenhouse climate monitoring and control with wireless sensor network and event-based control. 2020.
- [4] Iot in agriculture: 9 technology use cases for smart farming (and challenges to consider), 2020.
- [5] Alexander T. Demetillo, Michelle V. Japitana, and Evelyn B. Taboada. A system for monitoring water quality in a large aquatic area using wireless sensor network technology. 2020.
- [6] We are intech open, the world's leading publisher of open access books built by scientists, for scientists, 2020.
- [7] Joao Tagaio. Development of a smart node iot for agriculture appliances. Master's thesis, 2020.
- [8] Haider Mahmood Jawad, Rosdiadee Nordin, Sadik Kamel Gharghan, Aqeel Mahmood Jawad, and Mahamod Ismail. Energy-efficient wireless sensor networks for precision agriculture: A review. 2020.

[1] Yongchao Song, Jiping Bi, Xuan Wang. Design and implementation of intelligent monitoring system for agricultural environment in IoT.

[2] Abdennabi Morchid, Rachid El Alami, Aeshah A. Raezah , Yassine Sabbar. Applications of internet of things (IoT) and sensors technology to increase food security and agricultural Sustainability: Benefits and challenges.

[3 ] Andrzej Pawlowski, Jose Luis Guzman Francisco Rodríguez , Manuel Berenguel , José Sánchez and Sebastián Dormido , Simulation of Greenhouse Climate Monitoring and Control with Wireless Sensor Network and Event-Based Control

[4] IoT in Agriculture: 9 Technology Use Cases for Smart Farming (and Challenges to Consider)

[5] Alexander T. Demetillo, Michelle V. Japitana<sup>1</sup>, and Evelyn B. Taboada Research Open Access A system for monitoring water quality in a large aquatic area using wireless sensor network technology

- [6] We are Intech Open, the world's leading publisher of Open Access books Built by scientists, for scientists
- [7] Development of a smart node IoT for agriculture appliances. Master Thesis Joao Tagaio 78310-1.pdf.
- [8] Haider Mahmood Jawad , Rosdiadee Nordin , Sadik Kamel Gharghan , Aqeel Mahmood Jawad and Mahamod Ismail Energy-Efficient Wireless Sensor Networks for Precision Agriculture: A Review
- [9] <https://www.3glteinfo.com/lora/lora-architecture>
- [10] <https://softjournal.com/insights/pros-and-cons-of-php-programming-language>
- [11] <https://www.krishaweb.com/blog/best-laravel-security-features>
- [12] João Pedro Ramos Tagai. Development of a smart node IoT for agriculture appliances
- [13] Sourodip Chowdhury, Shaunak Sen, and S Janardhanan. Comparative Analysis and Calibration of Low-Cost Resistive and Capacitive Soil Moisture Sensor.
- [14] 300/400 mA High Efficiency Buck Converter with Ultra-low Quiescent Current. Texas Instruments, 6 2015. URL <http://www.ti.com/lit/ds/symlink/tps62743.pdf>. Revised May 2016.
- [15] R. Want, B. N. Schilit, and S. Jenson. Enabling the internet of things. *Computer*, 48(1):28–35, 2015. ISSN 00189162. doi: 10.1109/MC.2015.12.
- [16] B. Wolbert. Designing With Low-Dropout Voltage Regulators. 1(408):944–970, 1998
- [17] M. Cerchecci, F. Luti, A. Mecocci, S. Parrino, G. Peruzzi, and A. Pozzebon. A low power IoT sensor node architecture for waste management within smart cities context. *Sensors (Switzerland)*, 18(4), 2018. ISSN 14248220. doi: 10.3390/s18041282.
- [18] Single-Supply, Low Power, Precision FET Input Quad Buffer. Analog Devices, 12 2014. URL [https://www.microchip.com/stellent/groups/picmicro\\_s/documents/devicedoc/cn547043.pdf](https://www.microchip.com/stellent/groups/picmicro_s/documents/devicedoc/cn547043.pdf). Revision A.
- [19] LoRa™ Technology Module Command Reference User's Guide. Microchip, 2015. URL <https://ww1.microchip.com/downloads/en/DeviceDoc/40001784B.pdf>. Revision A.
- [20] TPS65021 Power Management IC For Li-Ion or Li-Polymer Powered Systems. Texas Instruments, 10 2005. URL <http://www.ti.com/lit/ds/symlink/tps65021.pdf>. Revision 2015.

[21] TLV755P 500-mA, Low IQ, Small Size, Low Dropout Regulator. Texas Instruments, 11 2017. URL <http://www.ti.com/lit/ds/symlink/tlv755p.pdf>. Revision 2018.

[22] emTech. Sx1261/2 - long range, low power, sub-ghz rf transceiver. URL <https://www.semtech.com/uploads/2019/03/Sx1261-2.pdf>. Access on 19 March 2019.

[23] By : CIRCUITSTATE Electronics LLP .URL : [https://www.circuitstate.com/tutorials/interfacing-ra-01-ra-02-sx1278-lora-modules-with-esp32-using-arduino/Package\\_Format](https://www.circuitstate.com/tutorials/interfacing-ra-01-ra-02-sx1278-lora-modules-with-esp32-using-arduino/Package_Format)