



UNIVERSITY OF BALAMAND

FACULTY OF ENGINEERING

DEPARTMENT OF COMPUTER ENGINEERING

Greenhouse automation with AI for disease detection

Final Report

Students' Names:	Mark Hanna	A2110650
	Jad Zaiter	A2111535
	Julien Hayek	A2011611

Advisor: Dr. Nicolas Haddad

Moderators:

Dr. Issam Dagher

Dr. Mostapha El Hassan

Dr. Mohamad Khaldi

Course Code: ELCP392

Course Title: Senior Design II

Date Of Submission: May 2024

ABSTRACT

Greenhouse automation has become a popular approach in today's agriculture sector. Its importance is growing day by day due to the fascinating results it provides. It has a crucial role in saving the planet's resources by getting rid of old farming methods. Moreover, it provides higher crop yields since the plants are growing in their optimal environment. Lastly, it is powered by green energy which makes it environmentally friendly and does not produce pollution. This project aims to automate a greenhouse using microcontroller, sensors, and actuators. Furthermore, it aims at developing an AI that has the ability to check for disease in plants. This feature is missing in most of the modern-day greenhouse and could save the crops before it's too late. In addition to that a web-application will be developed so that the farmer can monitor the greenhouse, set the optimal conditions for the plants, and get a warning when a disease is detected.

Keywords: Greenhouse, Automation, AI, Disease, Plants, web-application.

TABLE OF CONTENTS

1.1	Introduction.....	11
1.2	Background Information.....	11
1.2.1	Agricultural Sector in Lebanon	11
1.2.2	Challenges in Traditional Agriculture.....	12
1.3	Sustainability	13
1.3.1	Why do we need sustainability?.....	13
1.3.2	Sustainable Development	15
1.4	Proposal	16
1.5	Design Criteria.....	16
1.6	Anticipated Outcomes.....	18
1.7	Scope of the project	18
1.7.1	General Constraints	19
1.7.2	Technical Constraints.....	19
1.7.3	Realistic Constraints.....	20
1.7.3.1	Time Constraints.....	20
1.7.3.2	Environmental Constraints.....	20
1.8	Conclusion	21
2.1	Introduction.....	22
2.2	Overview of Smart Technologies in Agriculture	23
2.3	Overview of Artificial Intelligence in Smart Agriculture	24
2.4	Internet of Things (IoT)	24
2.5	Sensor Technologies in Greenhouses.....	26
2.5.1	Types of Sensors	26
2.5.2	Sensor Monitoring	28
2.6	Data Transmission and Processing.....	28
2.6.1	Types of WSN Protocols.....	29
2.6.2	Computing	32
2.7	Energy conservation.....	33
2.8	Case studies.....	34
2.8.1	Case study: Automation and monitoring of greenhouse [14]	34
2.8.2	Case study: Smart Agriculture with an Automated IoT-Based Greenhouse System for Local Communities	36

2.8.3	Case study: IoT Smart Agriculture (University of Balamand)	39
2.9	Conclusion	40
3.1	Introduction	41
3.2	Block Diagram	41
3.3	Components and Sensors	43
3.3.1	Processor	43
3.3.2	Temperature sensor	46
3.3.3	Humidity sensor	47
3.3.4	Atmospheric pressure sensor	48
3.3.5	Carbon dioxide (CO ₂) sensor	49
3.3.6	Light intensity sensor	50
3.3.7	Soil Moisture sensor	51
3.3.8	Cameras	52
3.3.9	Wi-fi modules	53
3.3.9	Relay	54
3.3.10	Motors	55
3.3.11	Fans	56
3.3.12	Water pump	56
3.4	Standards and Protocols	57
3.4.1	Protocols	57
3.4.2	Standards	59
3.5	Conclusion	60
4.1	Introduction	61
4.2	AI for Disease Detection	61
4.2.1	Overview of the AI Disease Detector	61
4.2.2	Dataset	62
4.2.3	Training	63
4.2.3.1	Data Preparation	63
4.2.3.2	Model Definition	64
4.2.3.3	Model Training	65
4.2.3.4	Model Evaluation	66
4.2.3	Results	66
4.3	AI server	70
4.3.1	Overview of the AI server	70
4.3.2	Flowchart of the code	70
4.3.2.1	Server initializer	70

4.3.2.2 Request Handler	71
4.3.2.3 Image Processor	72
4.3.3 Breakdown of the Code	73
4.4 Main Server.....	74
4.4.1 Overview of the Main Server.....	74
4.4.2 Database.....	75
4.4.3 Website.....	77
4.4.3.1 Overview of the Website.....	77
4.4.3.2 Functionality	77
4.4.3.3 User Experience (UX).....	78
4.4.3.4 Technologies Used	79
4.4.3.5 Security	80
4.5 Conclusion	80
5.1 Introduction.....	81
5.2 ESP32 connected to sensors and actuators	82
5.2.1 Hardware connection of sensors to ESP32	82
5.3 ESP32-CAM.....	88
5.4 Results.....	89
5.3.2 ESP32-CAM – AI server.....	89
5.5 Future work.....	92
5.6 Conclusion	93
REFERENCES	94

LIST OF ABBREVIATIONS

Inter-Integrated Circuit	I2C
Serial Peripheral Interface	SPI
Graphical User Interface	GUI
Universal Asynchronous receiver / transmitter	UART
Artificial Intelligence	AI
Gross Domestic Product	GDP
Internet of Things	IoT
Carbon Dioxide	CO ₂
Wireless Sensor Network	WSN
Bluetooth Low Energy	BLE
Institute of Electrical and Electronics Engineers	IEEE
Radio Frequency Identification	RFID
Heating, Ventilation, and Air Conditioning	HVAC
Relative Humidity	RH
hectoPascals	hPa
Parts Per Million	ppm
Luminous Flux per Unit Area	lux
Analog to Digital Converter	ADC
Frames per Second	fps
Video Graphics Array	VGA
Ultra Extended Graphics Array	UXGA
Request for Comments	RFC
International Organization for Standardization	ISO

LIST OF TABLES

Table 1: Different types of physical sensors that are used in smart greenhouse farms [12]....	27
Table 2: WSN communication protocols typically used in smart greenhouses [12].....	30
Table 3: Comparison of microprocessors	43
Table 4: Comparison of temperature sensors.....	46
Table 5: Comparison of temperature sensors.....	47
Table 6: Comparison of barometric sensors	48
Table 7: Comparison of CO2 sensors	49
Table 8: Comparison of light intensity sensors.....	50
Table 9: Comparison of moisture sensors.....	51
Table 10: Comparison of cameras.....	52
Table 11: Comparison of wi-fi modules	54
Table 12: Comparison of protocols.....	59

LIST OF FIGURES

Figure 1: Agriculture's contribution to Lebanon's National GDP.....	12
Figure 2: An overview of integrated challenges majorly faced by the agriculture sector.	13
Figure 3: The 3 pillars of sustainable development.....	14
Figure 4: The 17 SDGs [6].....	15
Figure 5: Primary components of a typical greenhouse.....	22
Figure 6: IoT-Enabled Greenhouse.....	25
Figure 7: IoT General architecture for greenhouse applications.....	26
Figure 8: : Diagram of a monitoring system.....	28
Figure 9: Sensor node networks.....	29
Figure 10: Figure 10: Greenhouse combined with several renewable energy sources.....	34
Figure 11: Overview of components used.....	35
Figure 12: Picture of the Smart Greenhouse.....	36
Figure 13: Flowchart of case study.....	38
Figure 14: The framework of the application.....	39
Figure 15: Block Diagram of components.....	41
Figure 16: ESP32 architecture.....	45
Figure 17: ESP32 development board.....	45
Figure 18: ESP32-CAM AI Thinker development board.....	53
Figure 19: 4 5V relays.....	54
Table 20: Table of comparison motors.....	55
Figure 21: Motor driver.....	Error! Bookmark not defined.
Figure 22: Table of comparison fans.....	56

Figure 23: Water pump.....	57
Figure 24: InceptionV3 model.....	63
Figure 25: Logic of data preparation.....	64
Figure 26: Logic of model definition	65
Figure 27: Logic of model training	65
Figure 28: Logic of model evaluation.....	66
Figure 29: Cucumber confusion matrix.....	67
Figure 30: Strawberry confusion matrix	68
Figure 31: Bell Pepper confusion matrix.....	68
Figure 32: Eggplant confusion matrix	69
Figure 33: Tomato confusion matrix.....	69
Figure 34: Logic for initialization of the server	71
Figure 35: Logic for handling HTTP POST requests.....	72
Figure 36: Logic for image processing.....	73
Figure 37: Home page of the website displayed on different devices	79
Figure 38: Flowchart of the Greenhouse and its communication pathways.....	81
Figure 39: Hardware part of the system	82
Figure 40: Hardware connection of the TEMT6000 sensor	83
Figure 41: Hardware connection of the Hygrometer soil moisture sensor	83
Figure 42: Hardware connection of the DHT22 sensor	84
Figure 43: Hardware connection of the BMP180 sensor.....	84
Figure 44: Hardware connection of the CO2 sensor.....	85
Figure 45: Hardware connection of the water pump	85
Figure 46: Hardware connection of the fan.....	86

Figure 47: Hardware connection of DC motor.....	86
Figure 48: Flowchart of the code for ESP32 with sensors and actuators	87
Figure 49: Flowchart of the code for ESP32-CAM.....	88
Figure 50: Chart representing the Light environmental variable	89
Figure 51: Strawberry leaf 1 picture taken from ESP32-CAM.....	90
Figure 52: Strawberry leaf 1 predicted class and confidence	90
Figure 53: Strawberry leaf 2 picture taken from ESP32-CAM.....	90
Figure 54: Strawberry leaf 2 predicted class and confidence	91
Figure 55: Eggplant leaf 1 picture taken from ESP32-CAM.....	91
Figure 56: Eggplant leaf 1 predicted class and confidence	91
Figure 57: Eggplant leaf 2 picture taken from ESP32-CAM.....	92
Figure 58: Eggplant leaf 2 predicted class and confidence	92

CHAPTER 1

1.1 Introduction

Agriculture, which is essential to human existence and development, faces a slew of difficult problems in the twenty-first century. Climate change, a growing global population, and the strain on scarce resources are among them. The demand for creative and sustainable agriculture solutions in Lebanon is greater than ever. The need for locally grown products to feed an expanding population is critical. Hence, the country's agriculture industry must adapt to these new difficulties. The goal of this project is to create a solution to these problems: a smart greenhouse that will integrate cutting-edge technology to provide exact control over the growth environment. Our design incorporates automatic climate control, sophisticated irrigation systems, disease detection systems, and real-time monitoring. The outcomes will be as follows: year-round cultivation, better agricultural yields, and more effective resource usage. This project aims to assure a steady supply of locally grown produce while also paving the way for a more sustainable future for agriculture. In this chapter, we will go over the inception of the idea of the project and our team's vision of it in the future. In the second chapter, we will conduct a literature review on current smart greenhouse trends and technologies. In the third chapter, we will discuss the components we will use and the rationale behind choosing them.

1.2 Background Information

1.2.1 Agricultural Sector in Lebanon

Lebanon's agriculture business has a large potential to help assist the country's recovery from the economic crisis that began in October 2019. Figure 1 depicts the rise in GDP (on agricultural

goods) from 2013 to 2020. There was a -1.93% drop in GDP in 2018, which was further exacerbated in 2020 when it reached -20% [1]. The country has a diverse agricultural terrain and a competitive advantage in exporting vegetables, fruits, wine, olive oil, and tubers (United Nations, 2022). Agriculture now provides just 5% of the country's GDP and employs 8% of the working force, making it one of the most important economic contributors. Farming is a key source of employment, and it is especially important in the country's poorest regions, such as Akkar, Dinnyeh, Northern Bekaa, and the South, where agriculture may account for up to 80% of local GDP [1].

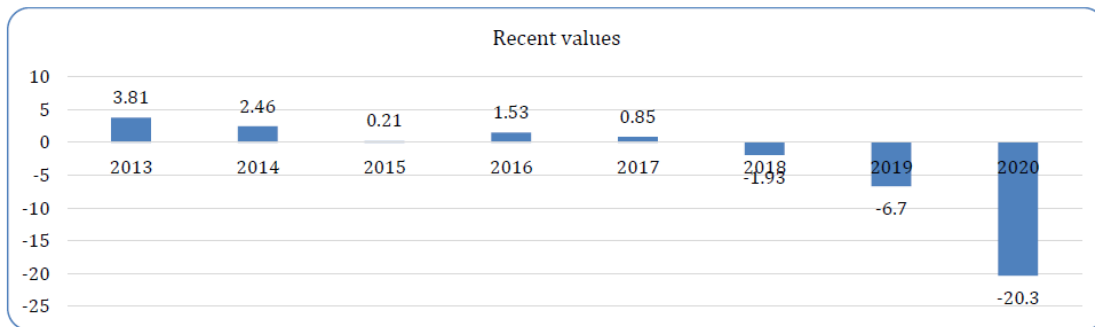


Figure 1: Agriculture's contribution to Lebanon's National GDP

1.2.2 Challenges in Traditional Agriculture

Traditional farming has many problems. Nowadays, relying too much on special crops, fake fertilizers, and chemicals is degrading the environment. This destroys the land, makes soil quality worse, and hinders nutrient movement in the soil. Also, the scarcity of water makes it hard for farms to stay productive. Moreover, the injudicious use of chemical fertilizers has made the soil more acidic and eroded it. To make sure farming remains sustainable for the future, it's

important to switch to methods of farming that mix what we know from tradition with newer technologically advanced methods [2].

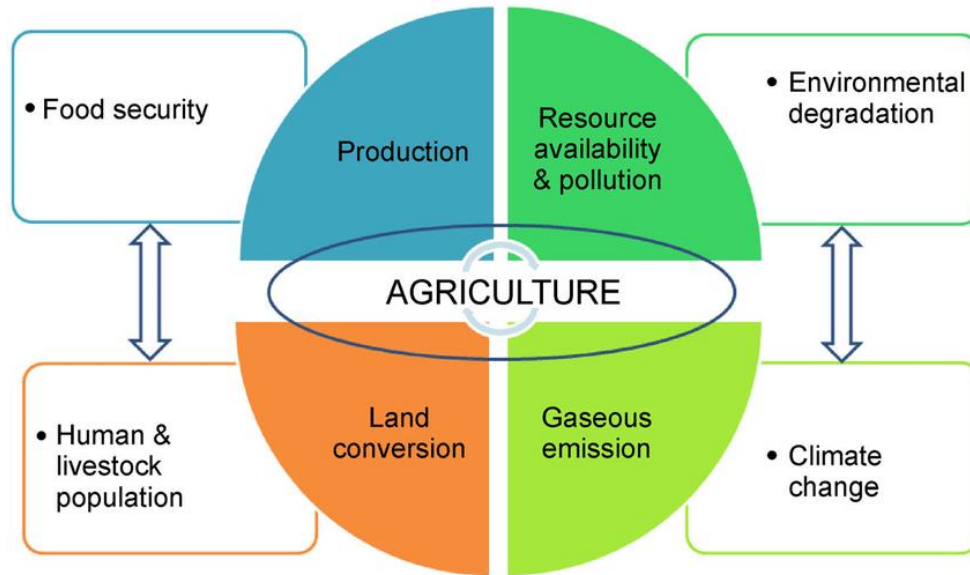


Figure 2: An overview of integrated challenges majorly faced by the agriculture sector.

Traditional agriculture also faces challenges in meeting rising demand due to population growth, urbanization, and income increases. The natural resource base is being exhausted and yield growth has slowed. Issues include the negative impact of chemical inputs, accessibility constraints to available land, and water scarcity. Climate change intensifies challenges, with adverse impacts expected to increase post-2030. Sustainable solutions require global efforts to improve resource efficiency, conserve resources, and adopt climate-smart agricultural practices [3].

1.3 Sustainability

1.3.1 Why do we need sustainability?

Amidst increasing climate change and global resource scarcity, sustainable engineering is essential for addressing the multifaceted challenges our world faces. As we navigate the complexities of an ever-evolving environmental landscape, the integration of sustainable practices into the design criteria of industrial products has become an important aspect of consideration. This shift towards sustainable engineering not only aligns with ethical responsibilities but also underscores the urgent need to foster a harmonious coexistence between human activities and the natural environment. With that in mind, while working on the project, it is within our obligation to adhere towards the 3 main principles of sustainable development: environmental, economic, and social sustainability [4].



Figure 3: The 3 pillars of sustainable development

1.3.2 Sustainable Development

The emphasis on sustainable engineering aligns with global initiatives such as the United Nations' Sustainable Development Goals (SDGs). The Sustainable Development Goals (SDGs) are a set of 17 global goals adopted by all United Nations Member States. They address various challenges, including poverty, inequality, climate change, environmental degradation, peace, and justice, with the aim of creating a more sustainable and equitable world by 2030. The goals cover diverse areas such as health, education, clean water, gender equality, and responsible consumption [5].



Figure 4: The 17 SDGs [6]

Our project emphasizes the following goals [7]:

- Goal 2: Zero Hunger: Greenhouses ensure food security and promote sustainable agriculture.
- Goal 7: Affordable and Clean Energy: Smart greenhouses incorporate energy-efficient technologies and renewable energy sources for sustainable energy use.

- Goal 9: Industry, Innovation, and Infrastructure: The development and implementation of smart greenhouse technologies involve innovation and contribute to building resilient infrastructure.
- Goal 11: Sustainable Cities and Communities: Smart greenhouses can be part of urban agriculture initiatives, promoting sustainable and resilient cities.
- Goal 12: Responsible Consumption and Production: Smart greenhouses can optimize resource use, reduce waste, and promote sustainable consumption and production practices.
- Goal 13: Climate Action: Greenhouses mitigate climate through optimizing energy use and reducing the environmental impact of traditional agriculture.
- Goal 15: Life on Land: Greenhouses can help combat land degradation and biodiversity loss.
- Goal 17: Partnerships for the Goals: Collaborative efforts and partnerships are often necessary for the development and implementation of smart greenhouse technologies.

1.4 Proposal

Our team proposes a smart greenhouse control system with AI integration to enhance crop yield, plant quality, and resource efficiency. Automation reduces labor costs, enables year-round cultivation, and creates income opportunities in rural areas. These structures provide resilience against weather disruptions and have the potential to boost the agricultural sector's GDP which in turn benefits the agricultural sphere and the national economy of Lebanon. In addition, the use of AI will be integrated into the system to allow for real-time analysis of the health of crops which would assist in indicating possible diseases.

1.5 Design Criteria

The goal of this project is to develop an automated smart greenhouse. In order to achieve this, the design should harbor the following specifications:

- It should consider the cost of implementation, maintenance, and operation to ensure long-term sustainability and financial viability.
- The project should be scalable to accommodate different greenhouse sizes and crop types.
- The project should align with overall sustainability goals, including reducing environmental impact, minimizing resource consumption, and promoting sustainable agricultural practices.

In addition, after studying the current literature on the topic and running iterative simulations to see what work rights, the greenhouse should operate as follows:

- The greenhouse should be automated and maintain optimal environmental conditions for plant growth. The system will efficiently manage various greenhouse components, including vents, oxygen levels, humidifier/dehumidifier, and adjustable shaders that regulate the amount of sunlight entering the greenhouse.
- A user-friendly mobile application or web interface will be developed to allow remote monitoring and control of the greenhouse. Users will also have the capability to visualize sensor data, providing them with insights into greenhouse conditions and historical performance.
- An AI-based system, integrated with a camera, will be implemented to monitor, and process images of the crops. This feature aims to detect signs of disease or stress in plants. Early detection will enable timely intervention, minimizing crop loss.

- To minimize the environmental impact, a solar irrigation system will be integrated to power the greenhouse and water pumps. This sustainable energy source aligns with the project's focus on resource conservation and environmental responsibility.

1.6 Anticipated Outcomes

The initiative aims to culminate in an advanced Automated Greenhouse Control System, seamlessly integrating cutting-edge embedded systems and sophisticated AI algorithms, ensuring optimal functionality. It will favor the following outcomes:

- **Improved Crop Yields:** Implementation of the system is expected to demonstrate improved crop yields compared to traditional greenhouse management.
- **Resource Conservation:** The project will showcase the efficient use of resources like water and energy.
- **Data Utilization:** The data collected by the sensors will not only be used for real-time control but will also serve as a valuable resource for further studies. These studies will focus on enhancing crop yields by analyzing historical data and optimizing growing conditions.

In order for these goals to be achieved, the design process should take into consideration various general, technical and realistic constraints.

1.7 Scope of the project

In this section, we will delegate the scope of the project by examining its limitations and constraints. Understanding these criteria is vital for maintaining the ethical standards of the system and establishing clear objectives for its successful completion. The project's development will be governed by three overarching categories: general constraints, technical constraints, and realistic constraints.

1.7.1 General Constraints

The smart greenhouse system must adhere to the following general constraints:

- Cost:
 - Smart greenhouse technologies, especially those involving AI, can be expensive to implement.
 - High initial investment costs may be a barrier for smaller or resource-constrained agricultural operations.
- Regulations:
 - Complying with the national regulations on greenhouses may hinder the creative process of the design

1.7.2 Technical Constraints

The technical constraints of the smart greenhouse project encompass:

- Detection Process Limitation:

- Due to cost constraints, the AI-based detection process should focus on areas near the crops, necessitating strategic camera placement.
- The camera must remain shielded to ensure cleanliness and accurate image categorization.
- Compatibility Issues:
 - Ensuring compatibility between different hardware and software components can be a technical challenge.
 - Devices and systems from different manufacturers may not always work seamlessly together.
- Scalability:
 - Scaling up smart greenhouse systems from small-scale prototypes to larger agricultural operations may present technical difficulties.
 - Ensuring that the technology is adaptable to different greenhouse sizes and crops is important.

1.7.3 Realistic Constraints

1.7.3.1 Time Constraints

- Project Completion Deadline:
 - The project should be completed and functional by the end of the 2023-2024 spring semester.

1.7.3.2 Environmental Constraints

- Sustainability:

- While the project aims to be environmentally friendly, the sourcing and disposal of materials must be considered to minimize the overall environmental impact.

The design and development of the smart greenhouse system will consider these general, technical, ethical, and realistic constraints to ensure a holistic and effective solution for revolutionizing agriculture in Lebanon.

1.8 Conclusion

To summarize this chapter, we outlined the need for a smart greenhouse in Lebanon's agriculture sector, addressing challenges through technology. The project aims for year-round cultivation, improved yields, and sustainable practices. Our proposal aligns with sustainability goals and emphasizes AI for crop health analysis. Despite constraints, we aim to complete the project by the 2023-2024 spring semester. We move on to the next chapter which discusses the literature surrounding smart greenhouses.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

A greenhouse is an enclosed system designed to modify and manage environmental factors, allowing plants to thrive in climates unsuitable for their natural growth. Originally made of brick or timber, modern greenhouses are typically glass or plastic structures crucial in agriculture and horticulture. They optimize growing conditions, protect crops from weather and pests, and facilitate effective crop management. Cover materials like polyethylene or fiberglass, along with climate-control systems, are continually evaluated for efficiency. Greenhouses are categorized based on nighttime temperature requirements, ranging from cool (7–10°C) to warm (10–13°C) and tropical (16–21°C). Greenhouse farming involves various sheltered structures, each differing in shape, lifespan, cover material, size, and technological management, impacting responses to sunlight, ventilation, heating, and costs [7].

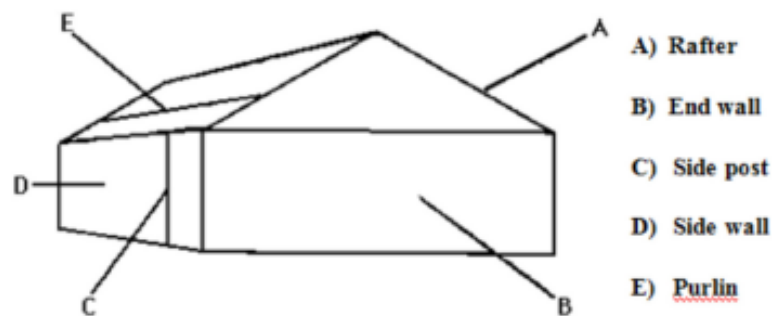


Figure 5: Primary components of a typical greenhouse

Research in greenhouse farming spans various aspects, including water use for GH irrigation, optimal GH structure design, soil conservation, overall energy consumption, climate control, and pest management. Current trends focus on utilizing photovoltaic greenhouses (PVG) for increased crop production, leveraging microclimates for optimal environmental control, strategic GH placement based on factors like orientation, light, and ventilation. Other areas of interest include site selection, integration of greenhouse management and environmental information systems for effective control and developing reliable models to predict and manage greenhouse microclimates. Additionally, there is a growing emphasis on assessing greenhouse system sustainability through energy evaluation [8].

2.2 Overview of Smart Technologies in Agriculture

Smart Agriculture employs innovative technologies, including data management, analysis systems, and remote-control technologies, to address global nutritional needs. Utilizing Fourth Industrial Revolution technologies like AI (Artificial Intelligence), robotics, and IoT (Internet of Things), it enhances productivity, profitability, and environmental sustainability while minimizing land resource usage. This approach involves advanced technology for clean food production, optimizing natural resource usage, especially water and fertilizers. Key features include reliance on management and information analysis systems for cost-effective production decisions and automation of processes such as irrigation, pest control, soil monitoring, and crop monitoring. Smart farms hold significant promise for achieving sustainable and resource-efficient agricultural production [9].

2.3 Overview of Artificial Intelligence in Smart Agriculture

AI technologies for smart greenhouses and precision agriculture encompass diverse options, from bio-inspired algorithms like Adaptive Neuro-Fuzzy Inference System–Particle Swarm Optimization (ANFIS-PSO), ANFIS-GA (ANFIS with Genetic Algorithm), and ANFIS-ACO (ANFIS with Ant Colony Optimization), to swarm intelligence algorithms (e.g., Artificial Bee Colony, Flower Pollination algorithm, Firefly algorithm, Krill Herd algorithm, and genetic optimization algorithms). Additionally, AI applications extend to UAV (drones) for pesticide application and robotic systems for tasks like harvesting, irrigation, soil treatment, seeding, and fertilizer application. The integration of robotics, UAVs, and bio-inspired algorithms in agriculture is driven by the need to enhance yields, operational efficiency, and cost reduction. Notably, the iGrow intelligent greenhouse system demonstrated a 10% improvement in tomato yields and a 92% increase in profits. Conversely, AI adoption in smart agriculture is also influenced by natural events such as pesticide resistance and forest fires compromising agricultural production. [10].

2.4 Internet of Things (IoT)

IoT, a key technology in smart farming, involves connecting various devices over the Internet, enabling remote operation and control. In smart agriculture, IoT applications use information and communication technology, sensors, and autonomous machines to optimize farm management and crop control. This aims to enhance agricultural production with lower costs and higher quality by obtaining accurate data, such as detailed terrain maps and climate predictions [11].

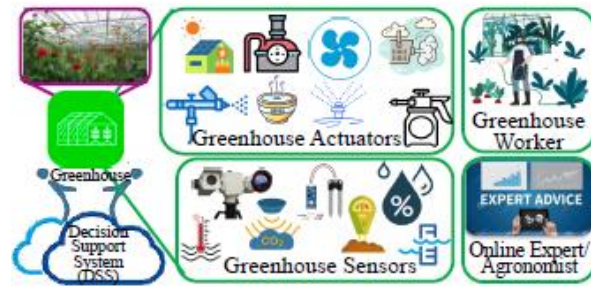


Figure 6: IoT-Enabled Greenhouse

IoT-based agriculture facilitates improved decision-making in production, monitoring pests, and optimizing pesticide use. The concept extends beyond agriculture, encompassing various items with processing units and internet connections. In a combined IoT system, the main components include physical devices, connection, data processing, statistical processes, and interface. The IoT process involves sensors collecting data from the environment, transmitting it across the network, processing the data, and providing results to end-users through alerts. The IoT's future impact is anticipated to streamline daily operations, increase efficiency, and reduce factors like downtime, energy consumption, and commuting efforts [9].

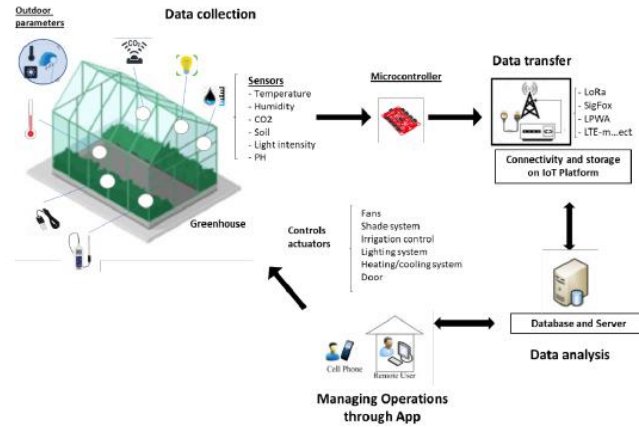


Figure 7: IoT General architecture for greenhouse applications

2.5 Sensor Technologies in Greenhouses

2.5.1 Types of Sensors

There are two primary categories of sensor technologies: bio/chemical sensors and physical parameter sensors [12]. Biosensors encompass microbial sensors, adaptive sensors and enzyme sensors; all of which are applied to detect residues of insecticides, antibiotics and toxic elements. On the other hand, physical sensors gather data pertaining to different environmental variables including temperature, humidity, soil moisture, light intensity, CO₂ levels and many more. To enhance efficiency, micro electro-mechanical systems (MEMS) are employed in creating sensors that are low-power, compact, cost-effective, and reliable [12].

Environmental Parameters	Sensor Technologies
Soil humidity and temperature	DHT11, SHT21, EC sensor (EC250), HA2001, Pogo portable, SHT71, SHT75

Soil moisture	SKY: SEN0114, FC-28, ECH2O, Hydra probe II, MP406, YL69, HA2001, Pogo portable
Air humidity	DHT22, CM-100, HMP45C(Vaisala's HUMICAP RH-chip), SHT17, Met Station One (MSO), CL-340
Air quality index	MQ-135
Temperature	SKY: DFR0026, 107-L (BetaTerm 100K6A1B), SHT17, SFAM-115KPASR
Light Intensity	BH170, S1133 (Photo diode), S2506 (infrared diode), LM-393, YL-369
CO2 level	Figaro's CDM4161A, TPS-2, PTM-48A, CL-340

Table 1: Different types of physical sensors that are used in smart greenhouse farms [12]

Table 1 notes some physical sensors that are commonly used in smart greenhouse farms. The choice of a sensor leads back to factors such as climate conditions, crop status, system compatibility, and market availability/cost. Choosing the right sensor is important in order to

guarantee that it remains operational and ideal under the extreme climate conditions it might endure.

2.5.2 Sensor Monitoring

There are 4 main aspect of greenhouses that need to be monitored in order to ensure a successful harvest: climate, soil conditions, plant growth, and plant diseases [12]. Figure 8 denotes a diagram of a typical monitoring system working hand-in-hand with other components of a smart greenhouse.

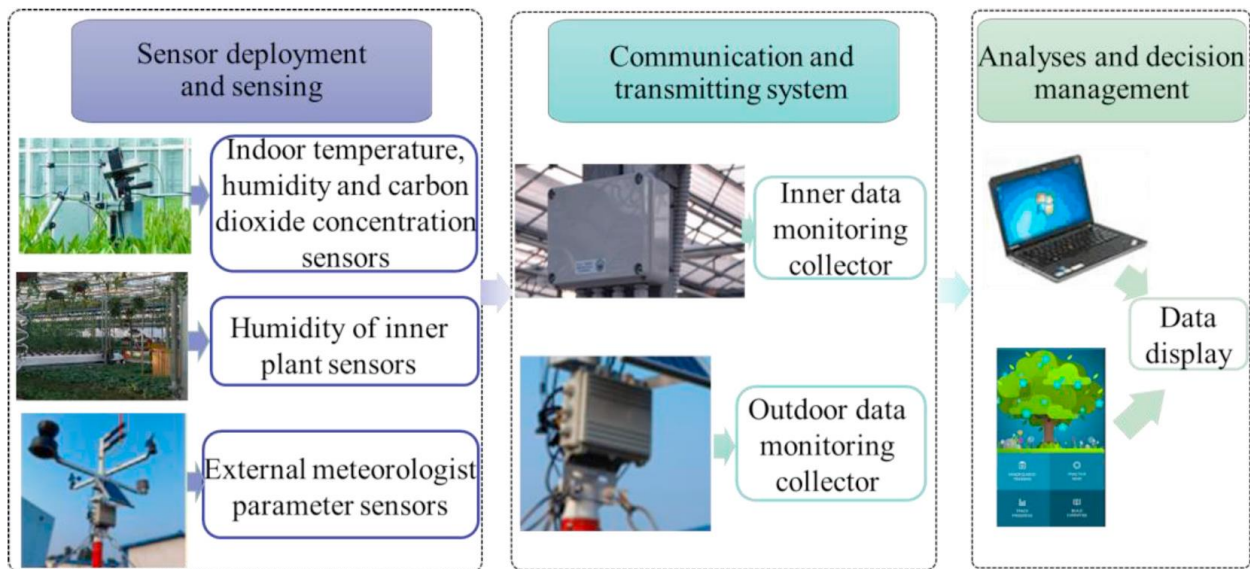


Figure 8: : Diagram of a monitoring system

2.6 Data Transmission and Processing

Data gathered from sensors will need to be transmitted and processed in a timely manner in order to maintain the greenhouse in real-time. Nowadays, wireless sensor networks (WSN) have replaced the used of wired transmission technologies. The sensor nodes used in Wireless Sensor network systems are integrated with the onboard controllers. The complete circuitry manages the operation and monitors it mainly. Everything is connected with the base station known as the Gateway (The Gateway acts as a bridge between the Wireless Sensor Network and cloud.), where high end processing of data collected from distributed sensors is done. Figure 9 demonstrates an overview of the sensor node networks transmitting data.

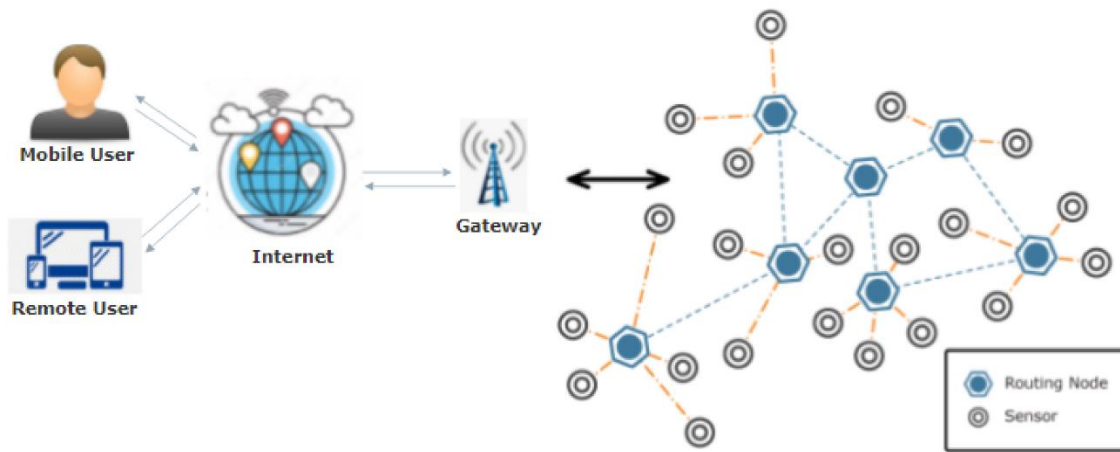


Figure 9: Sensor node networks

The following section will give detailed information about the latest data transmission technologies used in farming.

2.6.1 Types of WSN Protocols

WSN protocols	Topologies	Data transmission rate (bps)	Operating range	Power Consumption
IEEE 802.15.4	Star and Peer-to-Peer	20-250 kbps	10 m	Low
Zigbee	Star, Tree, and Mesh	250 kbps	10-100 m	Low
BLE	Star and Tree	1-2 Mbps	30 m	Low
6LowPAN	Star and Mesh	0.3-50 kbps	10-100 m	Low
Cellular	2G/3G frequency bands	10 Mbps	Several miles	High
LoRa WAN	Star and Peer-to-Peer	Less than 50 kbps	5-50 km	Low
RFID	Peer to peer	50 tags/s	10-20 cm	Ultra-low
Wi-Fi	Star	0.1-54 Mbps	Less than 92 m	Low

Table 2: WSN communication protocols typically used in smart greenhouses [12]

As noted in Table 2, there are a plethora of different WSN protocols to choose from. Effective data transfer in smart greenhouse farms relies on diverse communication protocols [12]:

- **IEEE 802.15.4:**
 - Uses LR-WPAN for cost-effective, low-power communication.
 - Integrated with modules like TelosB, ATmega256RFR2, and OpenMote-CC2538.
- **ZigBee:**

- Utilizes LR-WPAN for reliable global-standard data transmission.
- Devices include coordinator, router/modem, supporting topologies like Star, Mesh, and Cluster Tree.
- **BLE (Bluetooth Low Energy):**
 - Operates in short range with low bandwidth, suitable for IoT devices.
 - Offers low power consumption and quick initialization, but limited connectivity and security.
- **6LoWPAN:**
 - IPv6 protocol supporting star and mesh topologies with IEEE 802.15.4.
 - Used for monitoring soil and climatic conditions.
 - Sends IPV6 packets.
- **Cellular:**
 - For devices needing higher data rates, using 3G, 4G, and GSM modules.
 - Suitable for underground wireless networks, with higher battery and operational costs.
- **LoRa WAN:**
 - Based on IEEE 802.11 ah, employs LoRa network for long-distance, low-power communication.
 - No gateways in Lebanon, no LoRaWAN networks
- **RFID (Radio Frequency Identification):**
 - Uses radio waves for affirming and recording sensor information.
 - Includes active, passive, and battery-assisted passive tags.
- **Wi-Fi:**

- Commonly used WLAN protocol (IEEE 802.11) for monitoring plants, climatic conditions, and irrigation control.

2.6.2 *Computing*

There are two methods of computing data: edge, and cloud computing [12]:

- **Edge Computing:**

- Originating in 1990 with Akamai's CDN proposal, Edge Computing extends to support cloud systems using cloudlets and fog nodes.
- Other proposals include a three-tier platform addressing compatibility issues and an intelligent IoT-based water monitoring system.

- **Cloud Computing:**

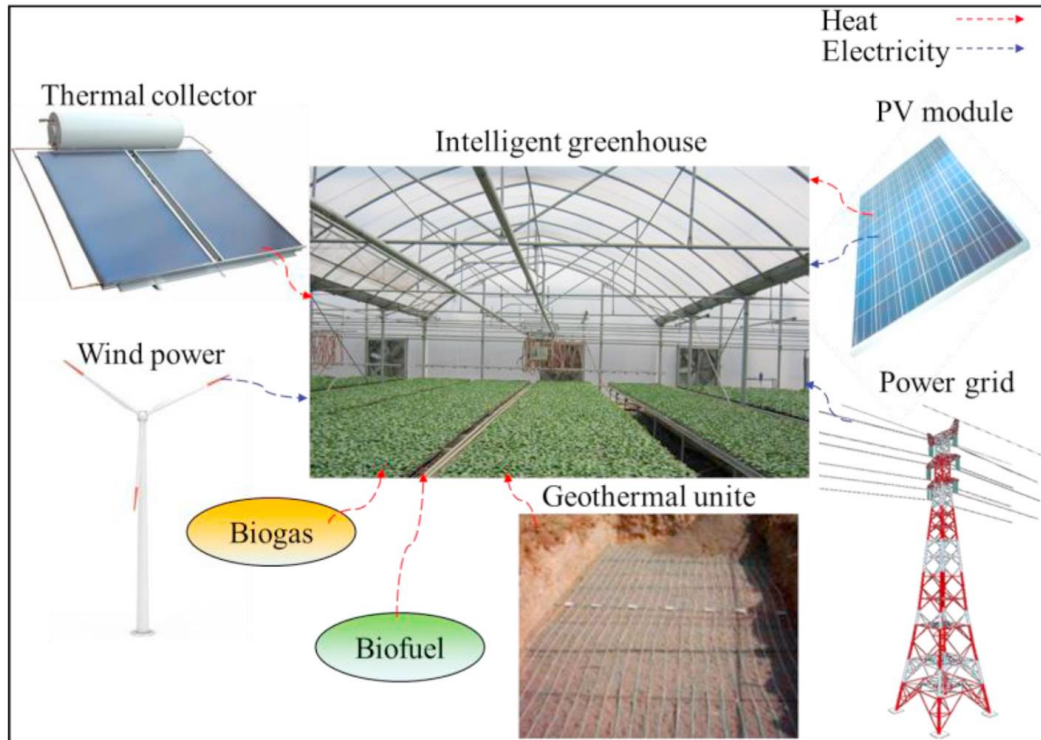
- As an internet cloud reservoir, cloud computing uses centralized or distributed computing technology.
- It offers infrastructure, hardware, and software services to IoT devices, allowing dynamic monitoring and cost-effective data storage.
- Applications range from disease identification to real-time monitoring platforms for enhanced crop yield and energy efficiency.
- There are challenges in data and system security, particularly with RFID tags and WSN, along with the need for affordable data processing for small-scale farmers.

Machine learning and artificial intelligence (AI) are employed to process sensor data, enabling intelligent decision-making. Examples include planning irrigation systems, identifying plant

diseases, and automating farm monitoring. Challenges include the early-stage development of expert systems for smart greenhouse IoT architecture. Factors hindering progress include the lack of IoT technology for detailed crop monitoring, insufficient local environment data, and the absence of correlations between crop growth and environmental data [12].

2.7 Energy conservation

The agricultural industry heavily relies on fossil fuels, leading to high greenhouse gas emissions. Rising production costs (40–50%) are attributed to increased energy consumption and labor costs. Various studies propose solutions, such as dynamic energy models, modified HVAC control strategies, and integration with building systems. Strategies for effective greenhouse management include energy-efficient designs, high-performance materials, advanced lighting and ventilation, and renewable technologies. Both active and passive technologies are employed to enhance energy efficiency, including solar, biogas, and ground energy for heating. Combining solar air collectors with phase change materials has proven to be effective in heating greenhouses. Overall, the focus is on adopting energy-efficient practices and renewable technologies to ensure the sustainability of greenhouse agriculture [13]. Figure 10 shows an example of multiple sources of renewable energy connecting to a greenhouse.



2.8 *Figure 10: Greenhouse combined with several renewable energy sources*

Case studies

In the following section, we will look at the current literature surrounding the smart greenhouses in the industry.

2.8.1 Case study: Automation and monitoring of greenhouse [14]

The author of the project designed and implemented a Smart Greenhouse Control System. The goal of the project was to optimize environmental conditions within a greenhouse to enhance plant growth and increase yield in the minimum possible time. The system utilized sensors and actuators to maintain optimal conditions such as temperature, light intensity, and soil moisture. The key components of the project (as noted in Figure 9) included an Arduino Mega 2560 R3

microcontroller, Arduino Ethernet Shield R2, various sensors (light-dependent resistor, DHT11 for temperature and humidity, moisture sensors), and actuators (fans, shower, artificial light).

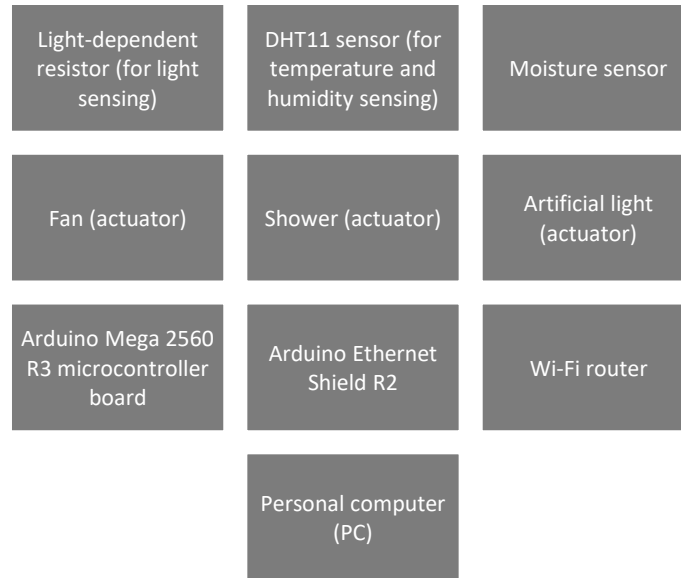


Figure 11: Overview of components used

The system comprised two main stations: the Sensors/Actuators Station and the Remote Monitoring Station. The Sensors/Actuators Station regulated the greenhouse environment by acquiring data from sensors, processing it with the Arduino board, and activating actuators to maintain optimal conditions. The Remote Monitoring Station allowed users to monitor the greenhouse system through either wired (serial port) or wireless (Wi-Fi) communication. A web-based application or an Android app connected to the local area network (LAN) provided access to the monitoring system. Figure 12 depicts a picture of the smart greenhouse.



Figure 12: Picture of the Smart Greenhouse

The project demonstrated control mechanisms for ventilation, light intensity, and soil moisture. For example, the system used DC fans to control temperature based on data from the DHT11 sensor. In addition, the system activated artificial light based on the light-dependent resistor readings. Furthermore, it controlled soil moisture using three moisture sensors and a pump.

In conclusion, the Smart Greenhouse Control System successfully achieved its goal of optimizing environmental conditions for plant growth. The system offered automation, efficiency, and remote monitoring. The author suggested potential applications for small-scale farming, herb cultivation, and preservation of endangered plant species.

2.8.2 Case study: Smart Agriculture with an Automated IoT-Based Greenhouse System for Local Communities

The author's design aims to create an IoT-based hydroponic greenhouse with the objective of serving as a comprehensive solution for local communities' agricultural needs. The design focuses on three main aspects:

1. Market Demand: The IoT system utilizes machine learning algorithms to analyze historical data on crop prices. It predicts the quantity and price demanded for specific types of plants in the local market. The system then provides optimal farming conditions and recommends specific plants based on the predicted demand.
2. Precision in Operation: The greenhouse employs automation using various techniques, including automated heating/cooling, CO₂ emissions control, LED lights, and nutrient supplementation. These components work together to create a unique growing recipe for each plant which enhances precision in the cultivation process.
3. Supervision: The IoT system continuously monitors the greenhouse using a variety of sensors. If any unusual changes occur in the plants or devices, the system immediately sends alerts to users via a mobile application. Users can access detailed reports in case of emergencies.

The architecture of the greenhouse system (depicted in Figure 13) uses various sensors and actuators connected through ZigBee for data collection. The data is sent to a local host where it is updated to the cloud for analysis using linear regression. Users can access the cloud to check plant status and receive cognitive recommendations.

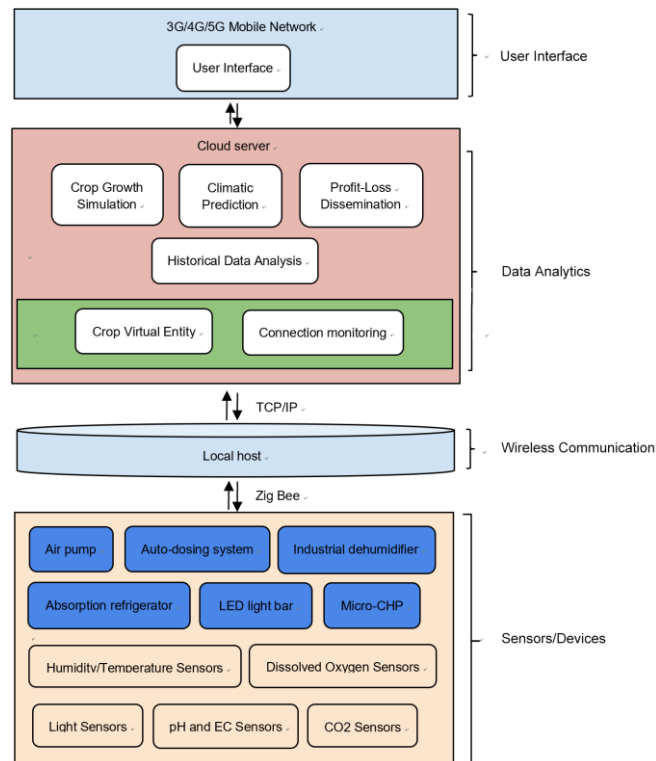


Figure 13: Flowchart of case study

The system also includes an analytics layer that uses machine learning algorithms (such as artificial neural networks) to formulate forecast models predicting plant growth and crop yield. The data collected from the greenhouse contribute to improving these forecast models.

Finally a mobile application (as seen in Figure 14) allows users to access information about the greenhouse. This information includes initial settings, real-time supervision, market exploration, trading opportunities, alerts, and user account management.

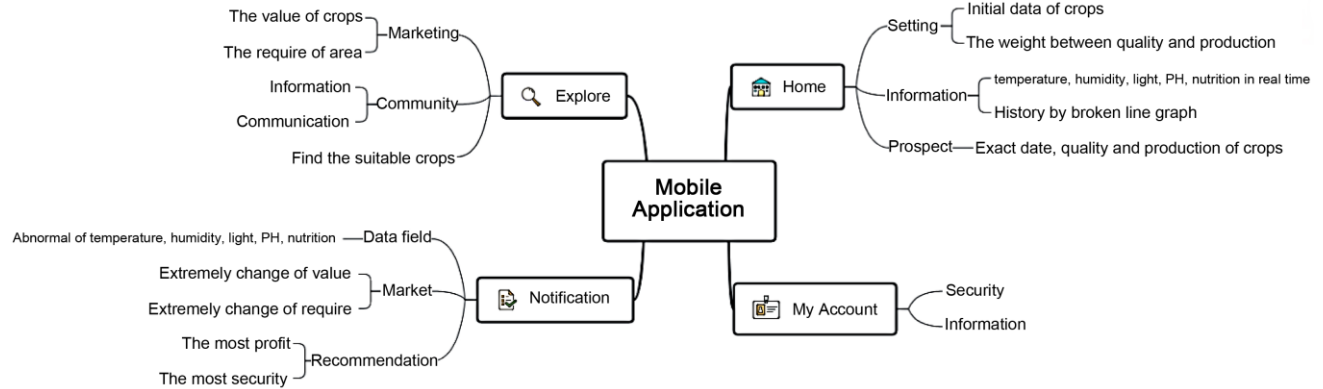


Figure 14: The framework of the application

In conclusion, the design aims to automate and optimize greenhouse operations. The proposed system combines data analytics, automation, and user-friendly interfaces to create an advanced IoT-based hydroponic greenhouse.

2.8.3 Case study: IoT Smart Agriculture (University of Balamand)

Around the time we started working on the project, it came to our attention that there were a couple of students implementing a project that is similar to ours. This project does the following:

1. It consists of an Arduino board that is connected to DevKitv1 (ESP32). They used five sensors: temperature and humidity, light, soil moisture, water turbidity and flame sensors. The data gathered from the five sensors is uploaded to the cloud.
2. For the cloud, they opened a channel in ThingSpeak. The channel contains read and write APIs that are used in the code of the Arduino board.
3. The data in the channel is analyzed, and the condition of the environment variables is then displayed to the user.

It's crucial to highlight the distinctions between our project ideas. In their project, users are only limited to visualizing gathered data (up to 7 fields). In contrast, our project not only allows data visualization but also enables actionable responses through the integration of actuators.

Furthermore, our project incorporates a camera equipped with AI capabilities to detect diseases on crops. Last but not least, we also have a dedicated web application for the user to control and monitor the greenhouse.

2.9 Conclusion

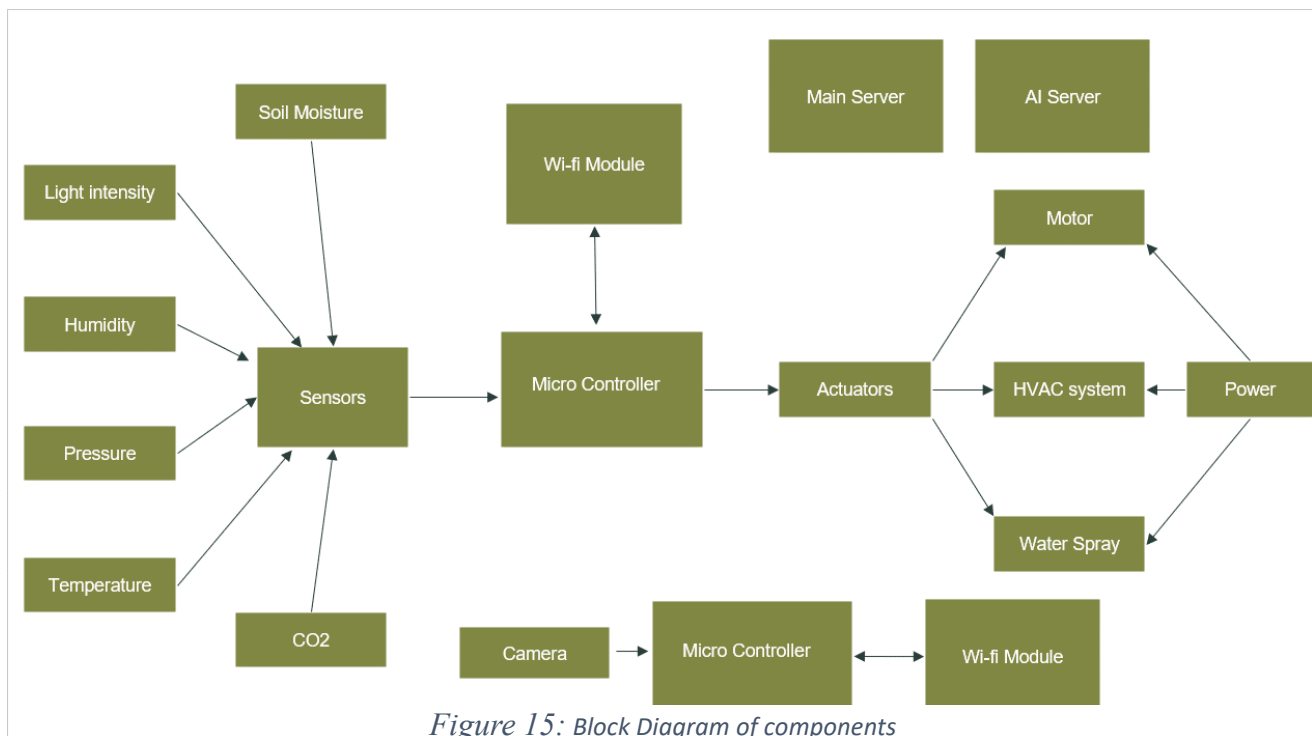
In this chapter, we covered the current literature surrounding smart greenhouses and the different prototypes available in the industry. Considering all the topics covered, which include sensor monitoring, computing, and current industrial applications, we move to the next chapter where we delve into the implementation of our project idea.

CHAPTER 3

3.1 Introduction

This chapter will cover the design's specification. A generic block diagram is included in the first section to list the project's numerous components in a general way. The ideal component combination for the design is then determined by comparing the various components' costs, availability, and specifications. Following the selection of the component, the basic block diagram is made more specific by adding the type of communication between each component and generating a block diagram with the exact name of each component. Finally, we list the various standards and protocols that were employed in the design.

3.2 Block Diagram



The Block Diagram in Figure 15 gives us an idea how the system will be connected as one. In the middle, we have the microcontroller which will be gathering the data from the sensors (to the left), controlling the various actuators (to the right), and finally transfer data using the wi-fi module (above). In addition, we have another microcontroller (below) which will be monitoring the crops with the camera (below) and transfer data using the wi-fi module (below).

Moreover, we have 2 servers, the Main and AI server. The AI server will be receiving images over Wi-Fi from the microcontroller monitoring the plant's leaf, running the AI for disease detection, and reporting back with the results to the Main server. The Main server will process all the data retrieved from sensors, let the user control the actuators, sampling frequency of sensors, and view retrieved data, and report if any diseases are detected. Along that, the Main server also hosts our website and stores the databases.

For the sensors, we have 6 types of sensors all of which will be connected to the microcontroller. We have 6 types of sensors temperature, humidity, carbon dioxide, pressure, and soil moisture.

For the actuators, the microcontroller will provide signals that control them. We have 3 main actuators the HVAC system, the motor for the curtains, and the water sprays. These actuators will be powered by a green power source.

The Wi-Fi module will receive and transmit data to the microcontroller. When receiving this means it will upload the data to a webserver. When transmitting data, this means a command or update from the user is being serviced.

3.3 Components and Sensors

3.3.1 Processor

Feature	Arduino Nano 33 BLE Sense [16]	ESP32 [17]	Raspberry Pi 4 Model B [18]
Microcontroller	Nordic nRF52840	Tensilica Xtensa LX6 dual-core	Broadcom BCM2711
Architecture	ARM Cortex-M4	32-bit RISC	ARM Cortex-A72
Clock Speed	64 MHz	Up to 240 MHz	1.5 GHz
Flash Memory	1 MB	4 MB	MicroSD (up to 2 TB)
RAM	256 KB	520 KB	2 GB
Wireless Connectivity	Bluetooth LE, IMU sensors	Wi-Fi 4, Bluetooth LE	Wi-Fi, Bluetooth
GPIO Pins	21	34	40
Analog Input Pins	8	12	None (but can be added)
USB Ports	1	1	2
Operating Voltage	3.3V	3.3V	5V
Board Size	Small	Depending on the manufacturer	Compact
Development Environment	Arduino IDE	Arduino IDE	Raspberry Pi OS
Cost	\$40.5	\$2+	\$35+
Communication Protocols	I2C, SPI, UART, BLE	I2C, SPI, UART, Ethernet, USB	I2C, SPI, UART, USB, Ethernet

Table 3: Comparison of microprocessors

The Arduino Nano 33 BLE Sense, ESP32, and Raspberry Pi 4 Model B each offer distinct advantages based on their features and capabilities. The ESP32 is notable for its advanced Tensilica Xtensa LX6 dual-core microcontroller, providing more processing power and integrated Wi-Fi 4 and Bluetooth Low Energy (BLE) connectivity. This makes it a suitable choice for compact projects requiring sensor data and wireless capabilities. The Arduino Nano 33 BLE sense, on the other hand, is a simpler, less powerful ARM Cortex M4 microcontroller, and more costly microcontroller board with integrated Bluetooth Low Energy (BLE), built-in sensors, and high programming capabilities due to big storage, suitable for less demanding applications. The Raspberry Pi 4 Model B stands out as a full-fledged single-board computer with a powerful ARM Cortex-A72 processor, multiple USB ports, and extensive RAM options. It excels in applications where a more robust computing platform is required, but it operates at a higher cost. On the other hand, our system requires Wi-Fi connectivity. Upon this request, we can directly eliminate the Arduino Nano 33 BLE sense microcontroller since it does not have a Wi-Fi module. Whereas both the Raspberry Pi 4 Model B and ESP32 have Wi-Fi connectivity. For our project, we will choose the ESP32 as it fits in well within the constraints, we have placed on ourselves. In addition, the processor also can support machine learning models designed with TensorFlow Lite.

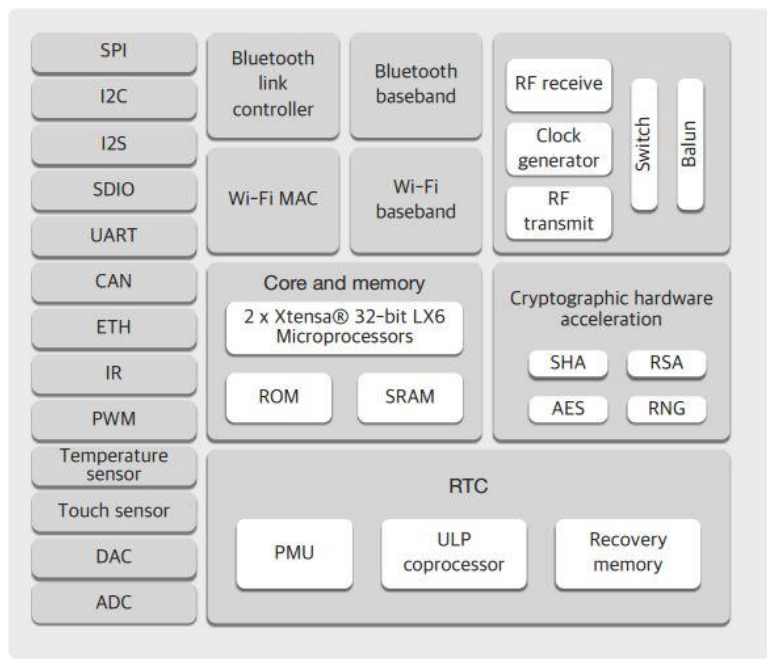


Figure 16: ESP32 architecture.

ESP32 DEVKIT V1 – DOIT version with 30 GPIOs

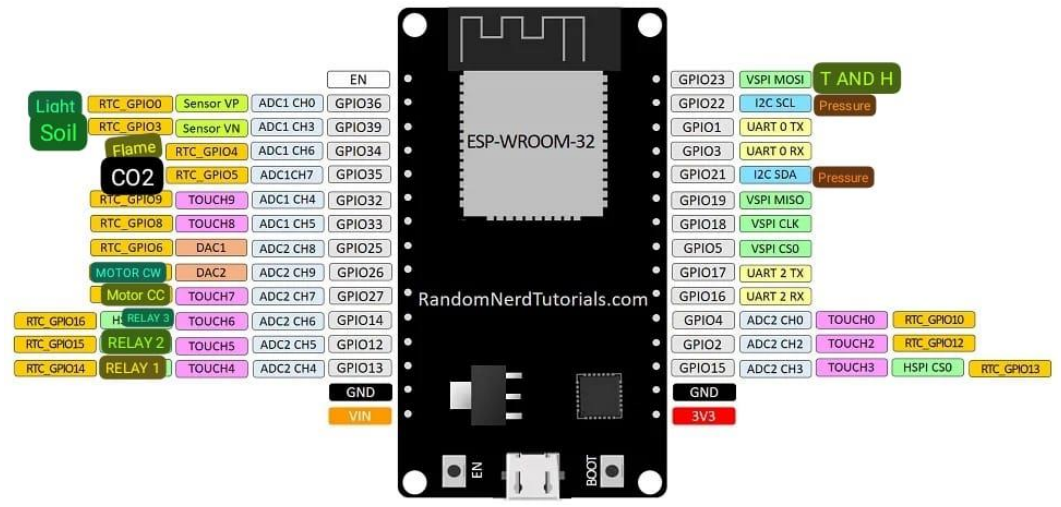


Figure 17: ESP32 development board.

3.3.2 Temperature sensor

Feature	DHT22 [19]	DS18B20 [20]	BMP280 [21]
Type	Temperature and Humidity	Temperature	Temperature and Pressure
Sensing Range	-40 to 80°C, 0 to 100% RH	-55 to 125°C	-40 to 85°C, 300 to 1100 hPa
Accuracy	±0.5°C, ±2% RH	±0.5°C	±1°C, ±1 hPa
Interface	Digital	Digital	I2C, SPI
Power Consumption	Low	Low	Low
Voltage	3.3V to 5V	3V to 5.5V	1.8V to 3.6V
Cost	\$9.95	\$3.36	\$4.00

Table 4: Comparison of temperature sensors

The DHT22, DS18B20, and BMP280 are three distinct sensors with unique capabilities. The DHT22 stands out for its dual functionality, measuring both temperature and humidity, making it ideal for applications requiring environmental monitoring. Its digital interface facilitates easy integration into various projects, and its cost-effectiveness makes it a budget-friendly option. The DS18B20, specializing in temperature sensing, might be preferred for applications where precision is crucial. Meanwhile, the BMP280, capable of measuring temperature and pressure, is suitable for projects requiring atmospheric pressure monitoring. Therefore, we will choose the DHT22 to measure the temperature.

3.3.3 Humidity sensor

Feature	DHT22 [22]	HCH-1000 [23]	SHS-A4L [24]
Type	Temperature and Humidity	Humidity	Humidity
Sensing Range	-40 to 80°C, 0 to 100% RH	0 to 100 %	10 to 95%
Accuracy	±0.5°C, ±2% RH	±2%RH	±5%RH
Interface	Digital	CCP/ADC	ADC
Power Consumption	Low	Low	Low
Voltage	3.3V to 5V	1.0V to 5.5V	2.4V to 5.5V
Cost	\$9.95	\$3.30	\$6.00

Table 5: Comparison of temperature sensors

The DHT22 stands out for its dual functionality, measuring both temperature and humidity, making it ideal for applications requiring environmental monitoring. Its digital interface facilitates easy integration into various projects, and its cost-effectiveness makes it a budget-friendly option. The HCH-1000, specializing in humidity sensing, might be preferred for applications where precision is crucial and harsh environmental conditions are present (Like HVAC systems). However, the tool is not very user-friendly due to its complex measuring process (Capacitance). Meanwhile, the SHS-A4L, capable of measuring only humidity, has a small range when compared with the other 2. Moreover, it is not very accurate but easy to

measure since humidity is based on its resistance. Hence, we will also stick with the DHT22 for its convenience.

3.3.4 Atmospheric pressure sensor

Feature	BMP180 [25]	MPL3150A2 [26]	MS5611 [27]
Type	Barometric pressure	Barometric pressure	Barometric pressure
Sensing Range	300 to 1100 hPa	0 to 1100 hPa	10 to 1100 hpa
Accuracy	± 2.0 hPa	± 1.0 hPa	± 0.1 hPa
Interface	I2C, SPI	I2C, SPI	I2C, SPI
Power Consumption	Low	Low	Low
Voltage	1.8V to 3.6V	2.2V to 3.6V	2.5V to 5.5V
Cost	\$3.50	\$9.95	\$10.83

Table 6: Comparison of barometric sensors

The BMP180, specializing in atmospheric pressure sensing, might be preferred for applications where reasonable precision is affected by harsh environmental conditions (Like in HVAC systems). However, the tool is very user and budget friendly. Subsequently, the MPL3150A2 stands out for its sensing range together with accuracy making it ideal for applications requiring environmental monitoring. Its digital interface facilitates easy integration into various projects, since it supports both I2C and SPI communication protocols, and its cost-effectiveness (compared to others) makes it a budget-friendly option. Meanwhile, the Ms5611, has a similar

range when compared with the other 2. Whereas, it has exceptional accuracy but when it comes to cost it is the most expensive. In the end, we have chosen the MPL3150A2.

3.3.5 Carbon dioxide (CO₂) sensor

Feature	MG-811 [28]	MQ-135 [29]	SCD-30 [30]
Type	CO ₂ sensor	CO ₂ sensor	CO ₂ sensor
Sensing Range	400 to 10,000 ppm	100 to 10,000 ppm	0 to 5,000 ppm
Accuracy	±100 ppm	±500 ppm	±20 ppm
Interface	ADC	ADC	Digital
Power Consumption	Low	Low	Low
Voltage	5.0V to 6.0V	5.0V	5.0V
Cost	\$5.00	\$4.00	\$48

Table 7: Comparison of CO₂ sensors

The MG-811 stands out for its sensing range and high-accuracy. Its analog interface facilitates easy integration into various projects, due to the presence of an Analog-to-Digital converter, and its cost-effectiveness makes it a budget-friendly option. The MQ-135 might be preferred for hobbyists and experimenters due to its low-accuracy and good sensing range. However, the tool is very user and budget friendly. Meanwhile, the SCD-30 has a small range when compared with

the other 2. Moreover, it has exceptional accuracy but when it comes to cost it is very expensive (due to the presence of an IR sensor). Therefore, we chose the MG-811.

3.3.6 Light intensity sensor

Feature	TEMT6000 [31]	VL53L0X [32]	TSL2561 [33]
Type	Light intensity	Light intensity and distance	Light intensity
Sensing Range	0 to 1000 lux	40 to 30,000 lux 0 to 2 meters	0 to 65,535 lux
Accuracy	$\pm 2\%$	$\pm 10\%$ to $\pm 20\%$ ± 2 meters	$\pm 3\%$
Interface	ADC	I2C	I2C
Power Consumption	Low	Low	Low
Voltage	3.3V to 5.0V	2.8V to 3.3V	3.3V to 5.0V
Cost	\$1.46	\$9.00	\$9.00

Table 8: Comparison of light intensity sensors

The TEMT6000, specializing in light intensity sensing, might be preferred for applications where precision and cost are crucial but features a small sensing range. Meanwhile, the VL53L0x, has dual functionality, measuring both distance and light intensity, has an acceptable sensing range when compared with the other 2. However, when it comes to light intensity it has a very low accuracy which makes it highly unreliable, and it has a high cost. The TSL2561 stands out for its reasonable accuracy and wide sensing range, making it ideal for applications requiring

environmental monitoring but it is costly. The TEMT6000 analog interface facilitates easy integration into various projects and its cost-effectiveness makes it a budget-friendly option.

3.3.7 Soil Moisture sensor

Feature	Grove – Capacitive Moisture Sensor [34]	FC-28 Soil Moisture [35]	DFRobot Soil Moisture [36]
Type	Moisture (Capacitive)	Moisture (Resistive)	Moisture (Resistive)
Sensing Range	0 to 1023	0 to 100%	0 to 100%
Accuracy	±5%	±5%	±5%
Interface	ADC	ADC	ADC
Power Consumption	Low	Low	Low
Voltage	5.0V	5.0V	5.0V
Cost	\$5.3	\$3.5	\$2.7

Table 9: Comparison of moisture sensors

The grove, FC-28, and DFRobot soil moisture sensors have different features. The grove is good for tough environments because it resists corrosion. It works by measuring capacitance, which changes with water presence. On the other hand, the FC-28 and DFRobot sensors are the same in terms of criteria, and they both measure moisture using resistance. All three sensors are affordable, but only the grove and DFRobot resist corrosion. Considering the price, the DFRobot sensor is the better choice.

3.3.8 Cameras

Feature	OV7670 [37]	OV2640 [38]
Type	CMOS	CMOS
Resolution	640x480 VGA	1600x1200 UXGA
Frame Rate	Up to 30fps	Up to 30fps
Interface	I2C	I2C
Power Consumption	60mW/15fps VGAYU V	120mW/30fps UXGA
Voltage	2.5V to 3.0V	2.5V to 3.0V
Cost	\$6.00	\$13

Table 10: Comparison of cameras

We were looking for a development board with a built-in camera and Bluetooth BLE or Wi-Fi to transmit the images. So, we had 2 choices either an ESP32 custom board which has the OV2640 camera built in or the machine learning kit for the Arduino Nano 33 BLE sense which has a custom PCB to integrate the OV7670 camera. The OV7670 and OV2640 are two different cameras with special features.

The OV7670 and OV2640 are similar, but the OV2640 has better resolution with higher power usage and cost. The OV2640 can also be used for object detection and classification due to having double frame per second when taking a live video. For our project, resolution is key to

detect the symptoms on the plant's leaves. Moreover The ESP32-CAM AI Thinker (13\$) is cheaper than the machine learning kit (140\$) and more powerful than the Arduino Nano 33 BLE sense. Both cameras are good, but the OV2640 and its development board are more convenient.

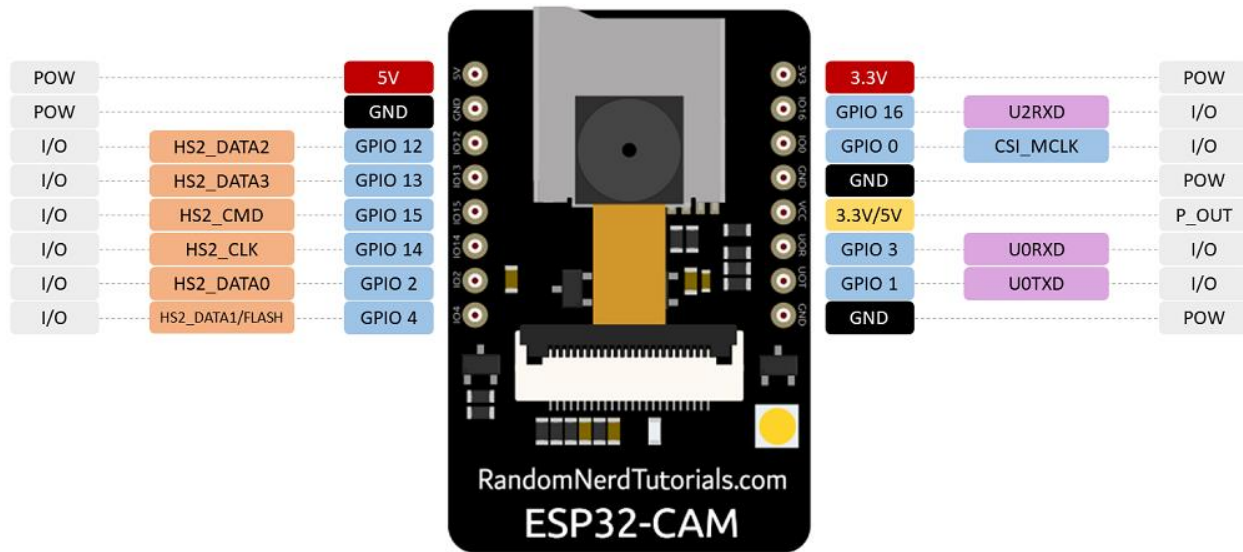


Figure 18: ESP32-CAM AI Thinker development board.

3.3.9 Wi-fi modules

Feature	ESP32 [39]	ESP8285 [40]	ESP8266 [41]
Processor	Dual-core	Single-core	Single-core
RAM	520 KB	512 KB	256 KB
Flash	4 MB	4 MB	1 MB
Interface	Bluetooth LE, SPI, I2C, UART	SPI, I2C, UART	SPI, I2C. UART
Power Consumption	200 mA	80 mA	80 mA

Bandwidth	802.11 b/g/n up to 150 Mbps	802.11 b/g/n up to 54 Mbps	802.11 b/g/n up to 150 Mbps
Cost	\$1+	\$5.5	\$1.2

Table 11: Comparison of wi-fi modules

The ESP32 and ESP8285 are practically identical when it comes to the processor, RAM, interface. However, the ESP32 consumes more power and supports an extra interface which is Bluetooth LE. Meanwhile, the ESP8266 and ESP8285 both are from the same family. The ESP8266 is a weaker module overall but superior in bandwidth. For our application we need a very powerful wi-fi module for scalability. This makes the ESP32 a good choice for it to be used as a Wi-Fi module. It is also an excellent choice since we are also using it as our main board.

3.3.9 Relay

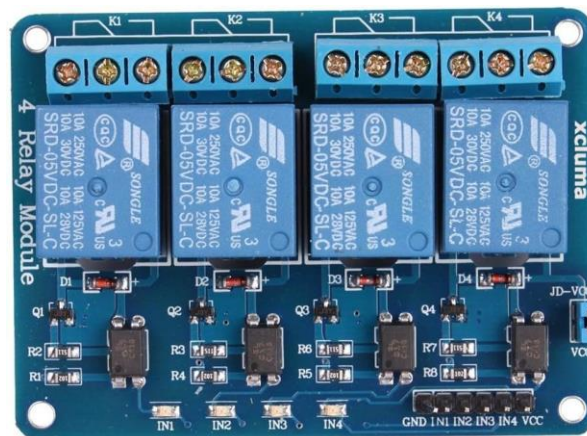


Figure 19: 4 5V relays.

In order to control the actuators, we will use a module that contains 4 5V relays. This module will be controlled by the ESP32. The SONGLE relays have a good reputation for being stable, long lasting, and hot-swappable. The price of such module is 4\$ which is budget friendly. This module will control the fan, motor, and water pump.

3.3.10 Motors

Feature	Stepper Motor [42] 4-Phase	DC Motor DIY [43]
Type	Bipolar stepper motor	Brushless DC motor
Speed	Up to 60 RPM	Up to 200 RPM
Torque	> 34.3 mN.m (at 120Hz)	Not specified
Interface	4 wires (requires stepper driver)	2 wires
Current	0.6 W	0.45W – 1.2W
Consumption	120 mA	150mA – 200 mA
Voltage	5.0V	3.0V to 6.0V
Cost	\$2.5	\$1.5

Table 20: Table of comparison motors

The stepper and DC motors are practically different. The stepper motor is less powerful, consumes less power but requires a stepper driver to control. On the other hand, the DC motor is more powerful, consumes more power and requires only 2 wires to control. The DC motor will

be controlling the curtains, so power is highly required. Along that, the DC motor price makes it more favorable which as a whole makes it the perfect pick for our project.

3.3.11 Fans

Feature	Raspberry Pi Cooling Fan [44]	DC Cooling Fan [45]
Type	Axial Fan	Axial Fan
Speed	Up to 3000 RPM	Up to 3500 rpm
Interface	2-Pin	2-Pin XH2.54 connector
Power and Current Consumption	Low Up to 1 W	Low Max of 2 W
Voltage	3.0V to 5.8V	5.0V
Cost	\$3.00	\$6.00

Figure 21: Table of comparison fans

Both fans are practically identical. The raspberry Pi cooling fan has a decent speed, good power consumption and low cost. On the other hand, the DC cooling fan has higher speed, more power consumption, and a higher cost. In our project, high amount of airflow is needed to keep the greenhouse temperature up to the owner's specifications and for this reason we will pick the DC cooling fan for having a better speed and a budget friendly cost.

3.3.12 Water pump



Figure 22: Water pump

We have chosen a water submersible motor pump that can pump from 80 to 120 L/H at only 0.4-1.5W power consumption. Moreover, the water pump price is budget friendly and comes at a cost of \$4.75. What is special about this pump is that it is submersible, has a low power consumption, and budget friendly price.

3.4 Standards and Protocols

3.4.1 Protocols

The ESP32 board features many important protocols. The protocols that we will use for connecting the components to the board and communicate with them are I2C, SPI and UART:

I2C (Inter-integrated circuit): It is a synchronous protocol which works over 2 lines in a half-duplex communication. It's a bidirectional protocol used for short distance communication. These 2 lines are SCL and SDA. The first is used for clock synchronization while the other is used for data transfer [46][47].

SPI (Serial Peripheral Interface bus): It is a synchronous protocol which works over 4 or more lines in a full-duplex communication. It's a bidirectional protocol used for short distance communication. These 4 lines are SCL, used for clock synchronization, MOSI, carries data from the master to the slave device, MISO, carries data from slave device to master device, and finally the CS, this wire is used by the master device to select the slave device it wants to communicate with. Each slave requires a unique CS line [47][48].

UART (Universal asynchronous Receiver-Transmitter): It is an asynchronous protocol which can work over simplex, half-duplex, and full duplex transmissions. It's a bidirectional protocol used for short distance communication. It consists of 2 data lines TX used for transmitting, and RX used for receiving data [49].

Feature	I2C [46][47]	SPI [47][48]	UART [49]
Data Rate	Low (up to 400 kHz)	Medium (up to 8 MHz)	Medium (up to 115,200 bps)
Range	Short (up to 10 meters)	Short (up to 10 meters)	Short (up to 10 meters)

Number of wires	2	4	2
Power Consumption	Low	Low	Low
Complexity	Simple	Simple	Simple
Channel type	Half-duplex	Full-duplex	Half-duplex

Table 12: Comparison of protocols

3.4.2 Standards

We will conform with the following standards during our implementation of the system:

IEEE 754: representing the IEEE floating point standard. This standard is used for offline training of the edge impulse neural network [50].

IEEE Wi-Fi 802.11: defines the rules for wireless networking devices to communicate with each other. This standard is made up of many protocols such as Physical layer, Media access control, Logical link control [51].

RFC 768: This standard will define the rules and methods to use the User Datagram Protocol [52].

RFC 9293: This standard will define the rules and methods to use the Transmission Control Protocol [53].

RFC 791: This standard will define the rules and methods to use the Internet protocol [54].

ISO/IEC FDIS 10918-5: representing the JPEG standard used in the disease detection system to take pictures of plants in JPEG format. This format is used since it allows compression of the image while preserving its quality. This feature is essential in image processing [55].

3.5 Conclusion

In conclusion, we outlined the design specifications, featuring a block diagram illustrating component connections. The central microcontroller manages various sensors, actuators, cameras, and Wi-Fi modules, facilitating artificial intelligence functions. The selection of communication protocols (I2C, SPI, UART) and adherence to standards like IEEE 754 and IEEE Wi-Fi 802.11 are discussed, ensuring compatibility and reliable operation.

CHAPTER 4

4.1 Introduction

In this chapter, we'll discuss the software innerworkings and network architecture of the smart greenhouse. More specifically, we will delve into the distributed nature of the system which include two servers (one main server hosting the database and the website and the AI server which contains trained models that detect diseases). The process behind the development of AI disease detection will be explored. Further emphasis will be placed on the website and its many features tailored towards a very user-friendly interface and robust security.

4.2 AI for Disease Detection

4.2.1 Overview of the AI Disease Detector

In this section, we will explore the development behind the AI disease detector we have implemented. In essence, images of crops captured from the ESP-32 CAM device module will have to be checked for diseases. Therefore, we have trained several models alluding to a different type of crop to diagnose the plant with a variety of diseases if suspected. The AI utilizes a convolutional neural network (CNN) architecture, leveraging transfer learning for efficient model training. Initially, we collected a diverse dataset consisting of images of healthy crops as well as crops affected by various diseases. The training process involved fine-tuning a pre-trained CNN model called InceptionV3, on our dataset. We employed techniques like data augmentation to increase the diversity of the training data and mitigate overfitting. Additionally, we partitioned the dataset into training and testing sets to evaluate the model's performance accurately and obtained confusion matrices for each crop we trained.

4.2.2 Dataset

Since crops come in different shapes and sizes, we were limited to having to train separate models for each crop we chose. This also meant we had to find datasets for each crop of our choosing since it's hard to have a single dataset covering every type of plant there is. Since we are based in Lebanon, we took into consideration the commonly grown crops and have chosen the following 5 crops to train:

- Cucumber
- Tomato
- Bell Pepper
- Eggplant
- Strawberry.

Since the project is focused on an automated greenhouse, we decided to not spend our budget on purchasing high quality datasets and instead opted for open-sourced datasets from a website called Kaggle. The criteria for choosing the datasets were to look at each dataset's traffic analytics and engagement on Kaggle to determine if they were suitable/trustworthy to work with or not. In addition, we utilized the InceptionV3 model (illustrated in Figure 24). Before the model can be used to recognize images, it must be trained using a large set of labeled images. ImageNet is a common dataset to use. ImageNet has over ten million URLs of labeled images. One million of the images also have bounding boxes specifying a more precise location for the labeled objects. This dataset spans 1000 object classes. For this model, the ImageNet dataset is

composed of 1,331,167 images which are split into training and evaluation datasets containing 1,281,167 and 50,000 images, respectively.

4.2.3 Training

We used Python as the framework to write the script to train these models. The sections below will feature a flowchart and breakdown of the script that detail the steps taken to train and save a model.

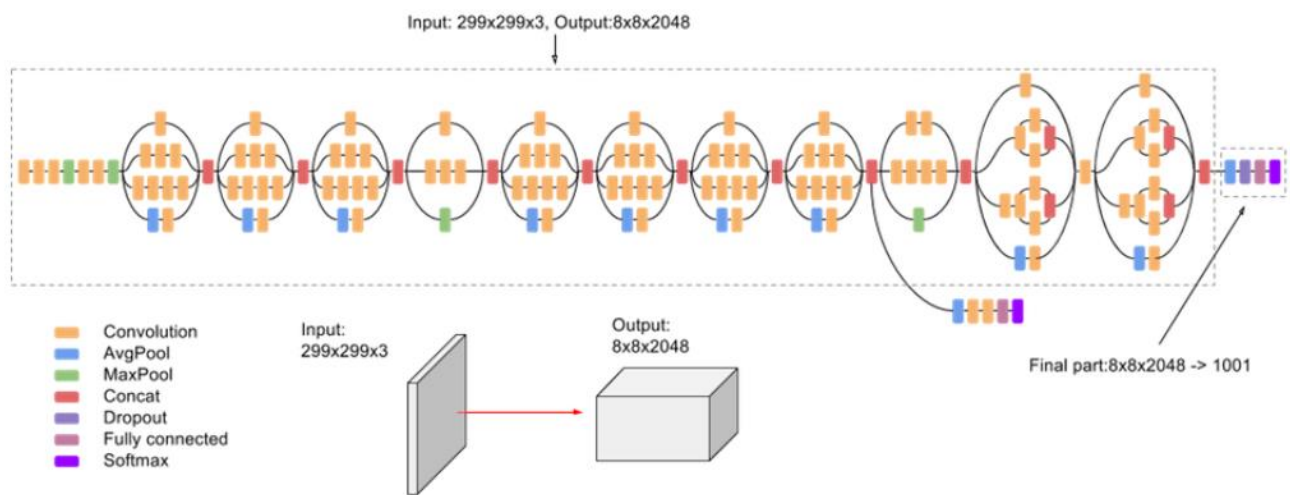


Figure 23: InceptionV3 model

4.2.3.1 Data Preparation

As noted in Figure 24, we first configure the data generators for the training and testing sets.

Then, the data generator for the training set applies various data augmentation techniques, such as rotation, shifts, shear, zoom, and horizontal flipping, to increase the diversity of the training data. The data generator for the testing set only applies normalization without any data

augmentation. The training and testing data are then loaded from the specified directories using the configured data generators.

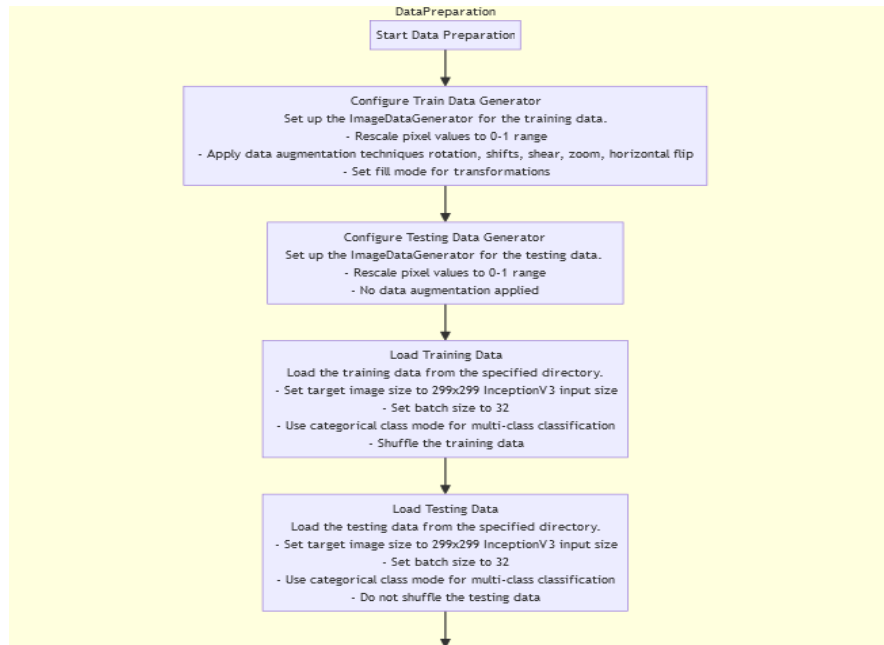


Figure 24: Logic of data preparation

4.2.3.2 Model Definition

Next, as seen in Figure 25, the pre-trained InceptionV3 model is loaded, with the top (fully connected) layers excluded. The base model is set as non-trainable to prevent the pre-trained weights from being updated during training. Additional custom layers are added on top of the base InceptionV3 model, including a global average pooling layer, dropout layers, a fully connected layer, and an output layer with the appropriate number of classes. The model is then compiled with the Adam optimizer, categorical cross-entropy loss, and accuracy metric.

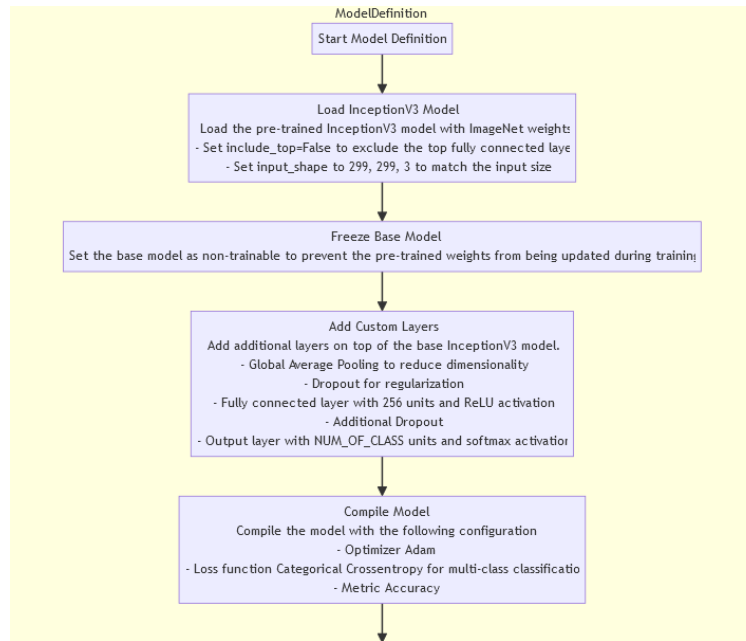


Figure 25: Logic of model definition

4.2.3.3 Model Training

As shown in Figure 26, the compiled model is trained using the prepared data generators for the training and the testing sets. The training is performed for a set amount of epochs, which was adjusted accordingly to the model. After training, the model summary is printed to provide an overview of the model architecture.

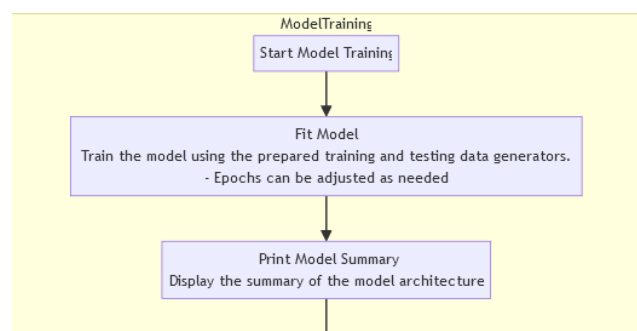


Figure 26: Logic of model training

4.2.3.4 Model Evaluation

Finally, as denoted by Figure 27, the trained model is used to make predictions on the validation data. The predictions are converted from probability distributions to class indices, as the dataset uses categorical labels. The true class labels from the validation data are retrieved. A confusion matrix is calculated to evaluate the model's performance. The confusion matrix is plotted as a heatmap using Seaborn and Matplotlib, with the class labels annotated.

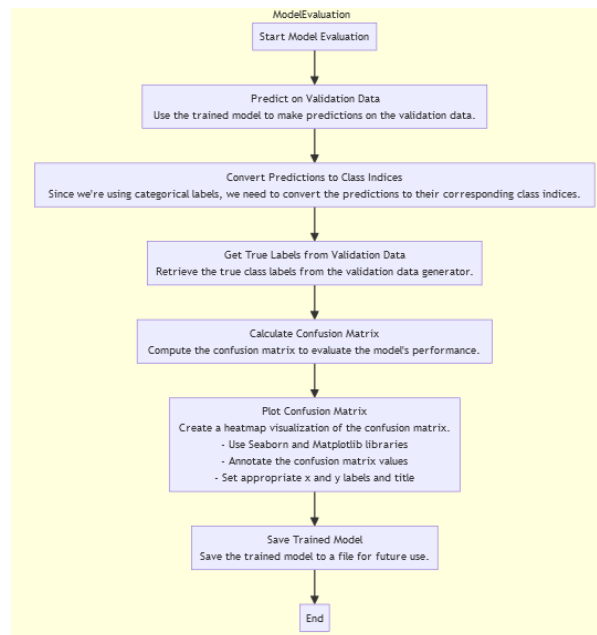


Figure 27: Logic of model evaluation

4.2.3 Results

Figures 28 to 32 showcase the confusion matrices for each crop we trained visualized using Seaborn's heatmap. Upon evaluating these matrices, on average the results across the board are deemed satisfactory for the scale of this project. Each model showed a commendable ability to correctly classify the majority of instances across different diseases. However, several factors contributed to instances where accuracy fell short, specifically the cucumber, eggplant and

tomato. Understandably, being that these datasets were free and open-sourced, limitations in the size and diversity of them may have hindered each model's ability to learn robust representations of each class. Despite efforts to augment the data and balance class distributions, certain classes may have been underrepresented compared to the more frequently occurring ones. In the end, given the circumstances we faced with the accessibility we had to datasets, we have moved forward with these models and integrated them into our design.

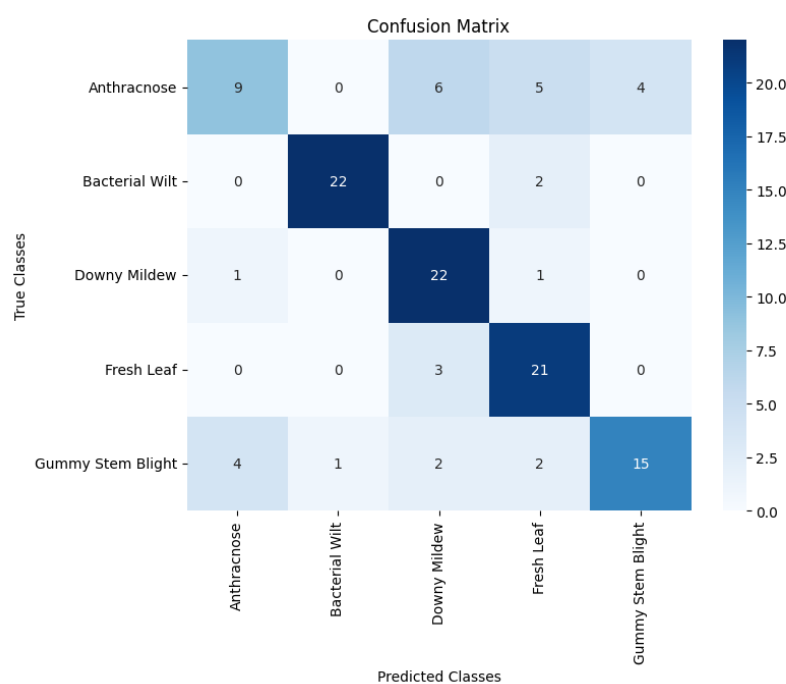


Figure 28: Cucumber confusion matrix

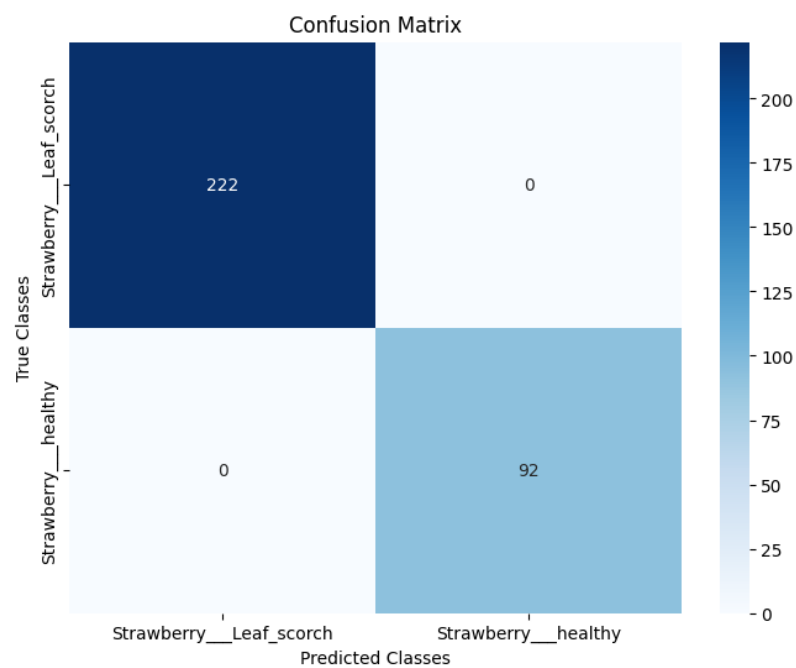


Figure 29: Strawberry confusion matrix

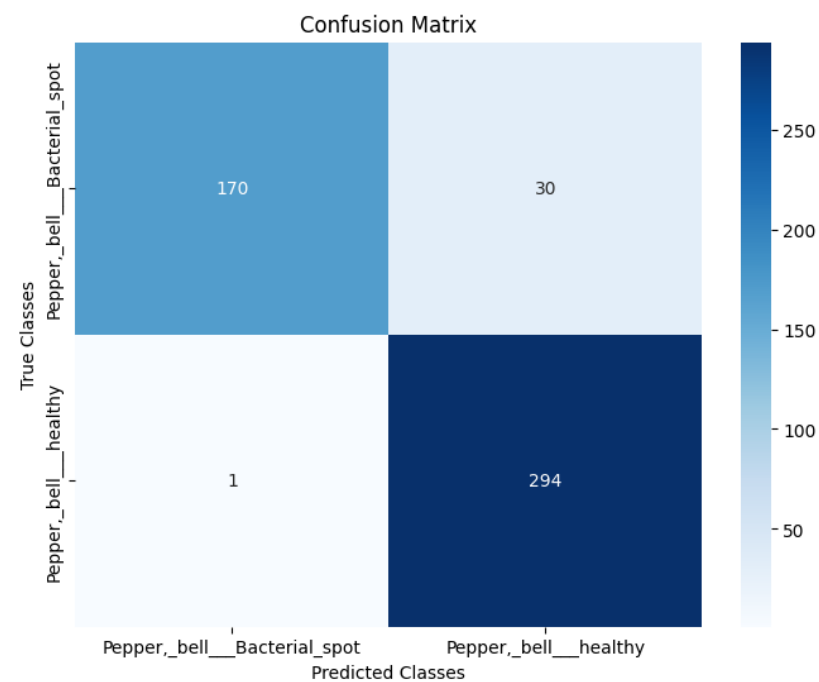


Figure 30: Bell Pepper confusion matrix

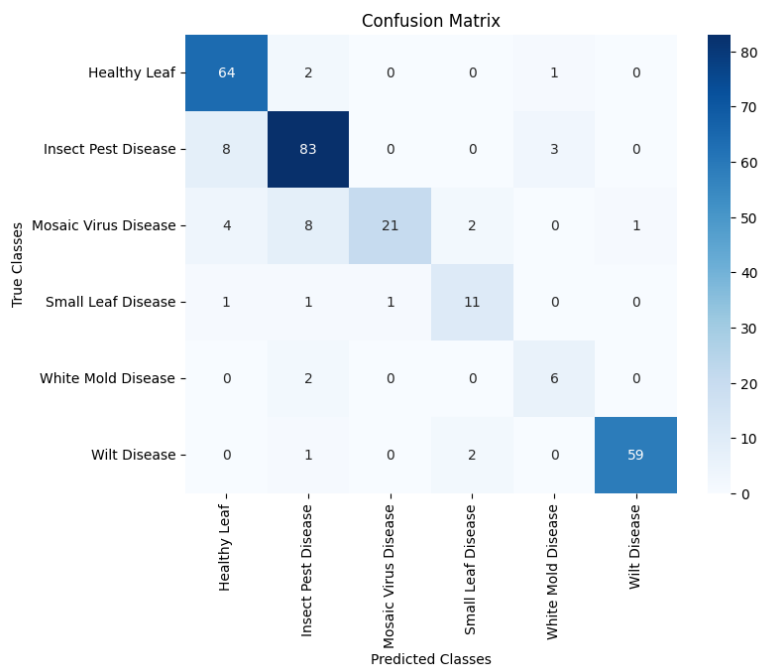


Figure 31: Eggplant confusion matrix

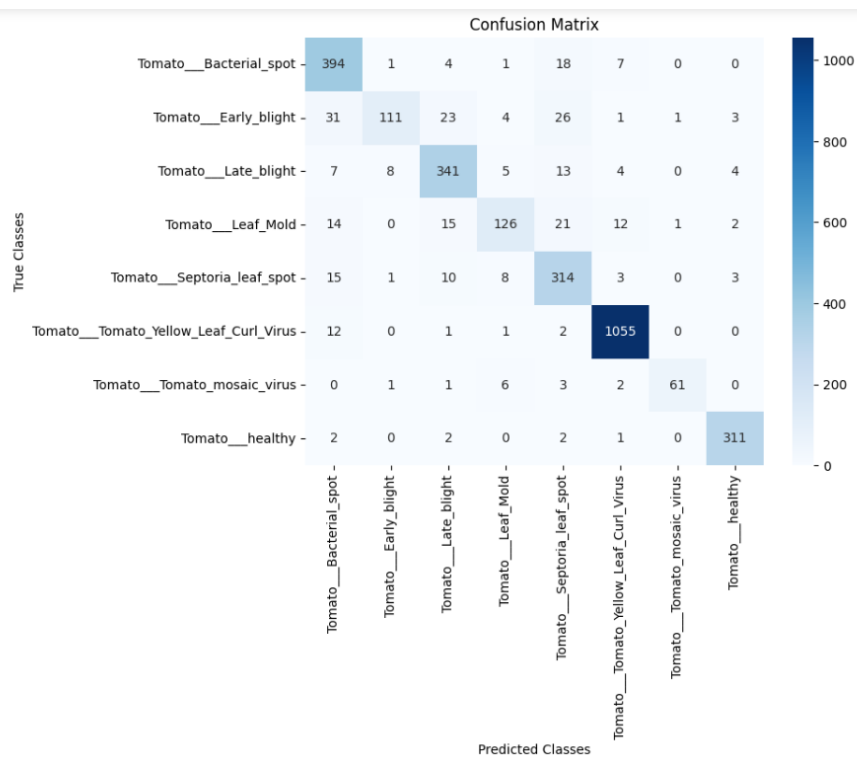


Figure 32: Tomato confusion matrix

4.3 AI server

4.3.1 Overview of the AI server

In this section, we'll be showcasing the AI server that the ESP32-CAM node connects to. The AI server was conjured up using Python as its framework. An HTTP server is setup that receives POST requests containing images from an ESP32-CAM device. Then, the server analyzes these images using pre-trained TensorFlow models to determine if they are disease-ridden plants or if they are healthy. Each model is specific to a different type of crop which are cucumber, strawberry, eggplant, tomato and bell pepper.

After completing the analysis of these images, the results (along with the associated key that identifies which node it received the image from) are then forwarded to the main server for storage.

4.3.2 Flowchart of the code

In this section, we have created visualizations to gain a better understanding of the logic behind the AI server. The functions mentioned here will be explained in the section involving the breakdown of the code.

4.3.2.1 Server initializer

Figure 33 denotes It first starts by getting the local IP address of the server machine using the 'get_local_ip()' function which then initializes the HTTP server with that retrieved address and the configured port. Finally it starts the server and begins serving requests. Next, we will see how the requests are being handled.

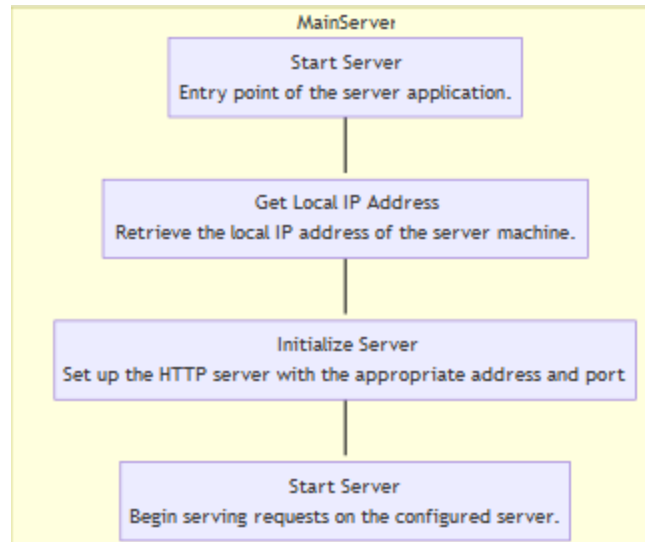


Figure 33: Logic for initialization of the server

4.3.2.2 Request Handler

Figure 34 represents the logic behind the handling of incoming HTTP POST requests. It begins with the request path being identified to select the appropriate model for processing, and then calling model-specific handling methods such as `HandleCucumber` or `HandleStrawberry`. In the event of an unknown route, it returns a 404 HTTP status. For image preparation and prediction, model-specific handling functions delegate responsibilities to the `ProcessImage` function. Finally, the `SendResponse` method sends the predicted results to the client. We will now see the logic behind the processing of the images.

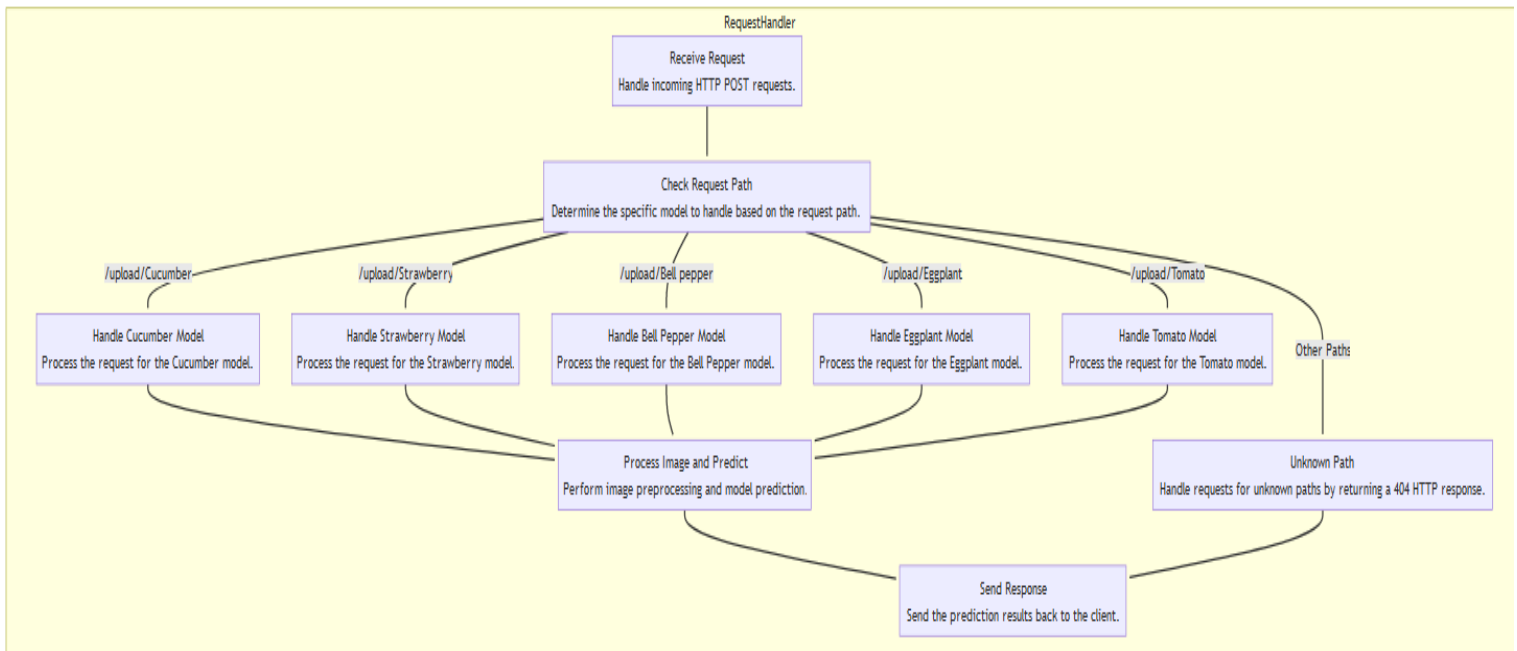


Figure 34: Logic for handling HTTP POST requests

4.3.2.3 Image Processor

Figure 35 represents the subgraph which shows the logic behind handling image preprocessing and prediction with a pre-trained model. Initially, it normalizes pixel values and modifies image size as required. It then loads the appropriate pre-trained model for the given crop. The MakePrediction function uses this model to anticipate the illness class and its associated confidence levels. Finally, the SendPrediction function sends these prediction results back to the main server. Now, we will see the breakdown of the code and its functions.

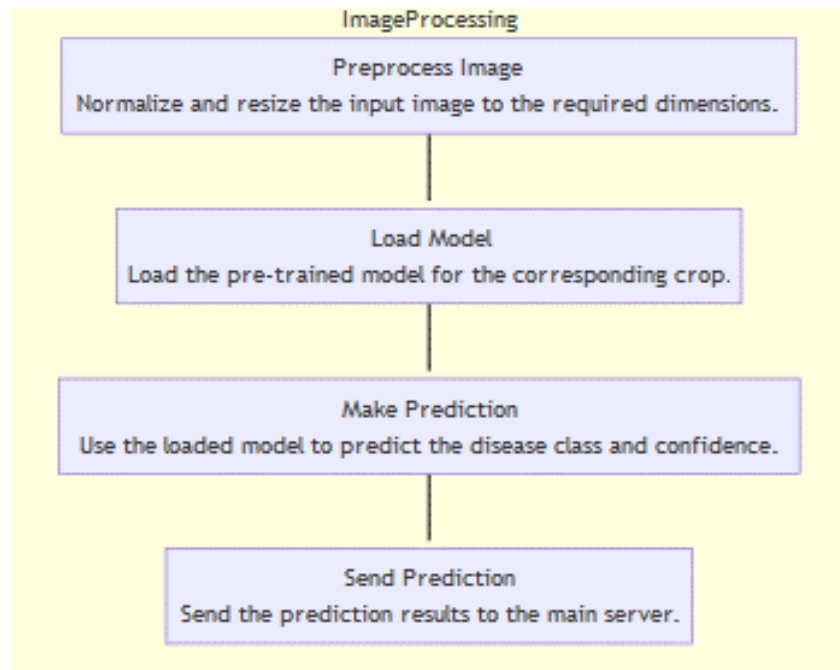


Figure 35: Logic for image processing

4.3.3 Breakdown of the Code

The following is a breakdown of each function and part of the Python code:

- **get_local_ip():** This function retrieves the local IP address of the server using a UDP socket. It creates a socket, connects to a known external IP address, retrieves its own socket name, which includes the local IP, and then closes the socket. Finally, it returns the local IP address.
- **Server class:** This server handles everything HTTP related. This class inherits from `BaseHTTPRequestHandler` and overrides the `do_POST()` method. This method is called when a POST request is received by the server. Inside `do_POST()`, it checks if the path contains '/upload'. If it does, it determines the model number based on the path ('/upload/Cucumber', '/upload/Strawberry', etc.), reads the image data from the

request, saves the image to a file, and then calls `process_image_and_predict()` method.

- **`process_image_and_predict(modelNumber, image)`:** This method takes the model number and the image as input. It preprocesses the image (normalizes pixel values and resizes it to 299x299), loads the corresponding model based on the model number, makes predictions on the image using the model, extracts the predicted class and confidence level from the predictions, sends a POST request to another server with the prediction results, and prints out the results and status of the request.
- **`run(server_class=HTTPServer, handler_class=Server, port=8000)`:** This function initializes and runs the HTTP server. It gets the local IP address, defines the server address using the IP address and a specified port, creates an instance of `HTTPServer` with the specified server address and `Server` class as the request handler, and starts serving HTTP requests indefinitely.

4.4 Main Server

4.4.1 Overview of the Main Server

In this section, we will delve into the details of our main server. The main server serves as the backbone behind the greenhouse. It hosts a database which contains all the data surrounding the device nodes in addition to hosting a website in which users can access and monitor the conditions of the greenhouse.

4.4.2 Database

The website utilizes MongoDB as its primary database management system, leveraging its NoSQL capabilities for storing user data, device information, session data, and other persistent data. MongoDB's flexibility makes it well-suited for managing the data requirements of device nodes and user accounts.

The database comprises multiple collections, each representing a distinct entity or data type within the application domain.

For example, the User collection stores user account information, including usernames, hashed passwords, salts, and associated devices.

The Device collection maintains device-specific data, such as device IDs, secret keys, linked users, sensor settings, and sensor data readings.

Additionally, the SensorData collection stores time-series sensor data captured by the device nodes, organized by device ID and sensor type for efficient querying and analysis.

Schema definitions are defined using Mongoose, an Object Data Modeling (ODM) library for MongoDB, providing a schema-based solution for modeling application data and enforcing data integrity. Each schema defines the structure of documents within a collection, specifying field names, data types, validation rules, and default values.

For example, the UserSchema defines fields for username, hash, salt, and devices, ensuring consistency and coherence in user account data.

Similarly, the DeviceSchema and SensorDataSchema define fields for device metadata, sensor settings, and sensor data readings, enabling standardized data representation and efficient data retrieval.

Mongoose schema validation features are employed to enforce data validation rules and ensure data consistency and integrity.

Validation rules are defined for each schema field, specifying constraints such as required fields, data types, minimum and maximum values, and pattern matching criteria.

Data sanitization techniques, such as input validation and sanitization middleware, are applied to prevent common security vulnerabilities, such as injection attacks and data manipulation, safeguarding the integrity and security of the database.

Access control mechanisms, such as role-based access control (RBAC) and user authentication, are implemented to restrict access to sensitive database resources and enforce data privacy and confidentiality.

User authentication tokens and session management techniques are employed to authenticate and authorize user access to database resources, preventing unauthorized data access and unauthorized operations.

Encryption techniques, such as TLS/SSL encryption for data transmission and encryption-at-rest, are utilized to protect sensitive data stored in the database from unauthorized access and data breaches, ensuring data security and compliance with privacy regulations.

4.4.3 Website

4.4.3.1 Overview of the Website

The website serves as a comprehensive platform for managing the device nodes, offering users the ability to register, log in securely, and efficiently manage their connected devices. It provides a centralized hub for users to monitor device data, control device settings, and communicate with their devices in real-time. The platform aims to streamline the process of managing IoT devices, enhancing user convenience and enabling seamless interaction with connected devices. Due to immense work and effort put into the website, no amount of words can do it justice. Hence, we have linked the GitHub repository in the appendix. Refer to it for more information.

4.4.3.2 Functionality

Users can register for an account with unique usernames and secure passwords, leveraging password hashing and salting for enhanced security. In addition, secure login functionality allows users to authenticate their identities and maintain persistent login sessions, enhancing user convenience and privacy. User management features enable users to add, remove, and modify their devices allowing for efficient management of their greenhouse system.

The platform also offers comprehensive device management functionalities, allowing users to view detailed information about their connected devices, such as device IDs and sensor configurations. Users can control various settings of their devices, such as sensor thresholds, actuator configurations, and data reporting intervals, providing flexibility and customization options. Moreover, a socket is established to allow for real-time communication between users and the device nodes. Users can send commands to their devices and receive real-time updates on device status, sensor readings, and environmental conditions.

Furthermore, the platform provides data visualization tools to display device data in intuitive charts, graphs, and dashboards, enabling users to analyze trends, identify patterns, and gain insights into their ecosystem. Users can monitor various parameters such as temperature, humidity, light levels, and air quality, as well as editing the sampling time dictating how often the device samples data. Users can also control the actuators to adjust these variables as needed via toggling them on conditionally according to the range of the environmental variables set for them.

4.4.3.3 User Experience (UX)

The website features a user-friendly interface with intuitive navigation, allowing users to easily access different sections of the platform, such as the dashboard, device settings, and data visualization pages. Throughout these pages, there are clear and concise menus, buttons, and navigation elements that enhance user experience and allow for seamless interaction with the platform. In addition, interactive feedback mechanisms, such as flash messages and form validation, provide users with immediate feedback on their actions, guiding them through the user journey and reducing errors and confusion. Last but not least, clear error messages and prompts help users understand and resolve issues quickly. Figure 37 illustrates the home page of the website on the PC and mobile. For more information, please refer to the GitHub repository in the appendix.

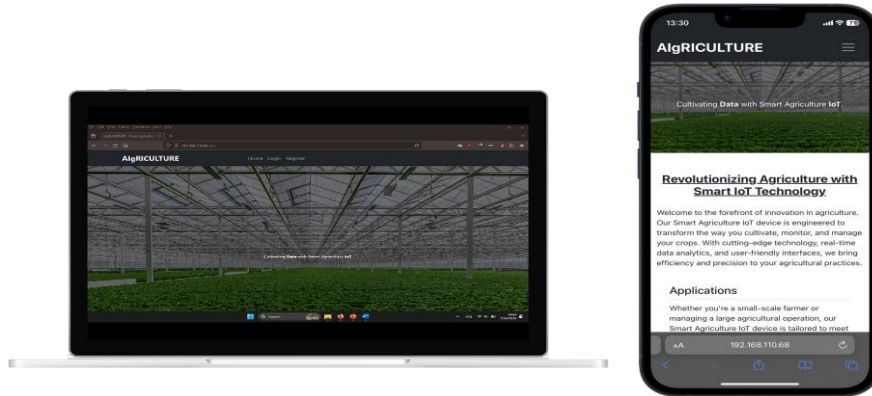


Figure 36: Home page of the website displayed on different devices

4.4.3.4 Technologies Used

Backend Technologies:

Express.js: The backend server framework for handling HTTP requests, routing, middleware management, and server-side logic implementation.

MongoDB: The NoSQL database used for storing user data, device information, session data, and other persistent data, providing scalability, flexibility, and performance.

Authentication and Security:

Passport.js: The authentication middleware for implementing user authentication strategies, such as local authentication with username and password, ensuring secure user access to the platform.

Cryptographic Libraries: Crypto modules such as crypto and bcrypt for password hashing, salting, encryption, and decryption, ensuring the security of sensitive data and communications.

Real-time Communication:

WebSocket: The protocol used for establishing persistent, bidirectional communication channels between the server and connected IoT devices, enabling real-time data exchange and event-driven interactions.

Frontend Technologies:

EJS (Embedded JavaScript): The templating engine for generating dynamic HTML content and rendering server-side views, facilitating the creation of interactive and data-driven web pages.

4.4.3.5 Security

Secure authentication and authorization mechanisms, implemented through Passport.js, ensure that only authenticated users have access to exclusive platform functionalities. In addition, there are strong password hashing and salting techniques that protect user passwords from unauthorized access and data breaches. Moreover, sensitive data, such as user passwords, device credentials, and session information, are encrypted using cryptographic algorithms to prevent unauthorized access. Lastly, input validation and sanitization mechanisms are put in place to prevent common security vulnerabilities, such as SQL injection and cross-site scripting (XSS).

4.5 Conclusion

We've discussed the distributed setup, AI disease detection, and the pivotal roles of the AI and main servers. Despite dataset limitations, our AI models perform satisfactorily. The AI server handles real-time image analysis, while the main server manages data storage, user authentication, and device management. Security measures, such as password hashing and encryption, are in place to protect user data. Overall, our system emphasizes innovation, sustainability, and user convenience in advancing agriculture and environmental stewardship.

CHAPTER 5

5.1 Introduction

This chapter will cover the hardware components of the project, specifically the node containing an ESP32 with sensors and actuators and the node containing the ESP32-CAM, and their interactions with their designated servers. Also, we will discuss and interpret the results that we have obtained after their implementation. Moreover, we will talk about what future work lays ahead of this project to improve it. Last but not least we will conclude the report and our findings on the matter.

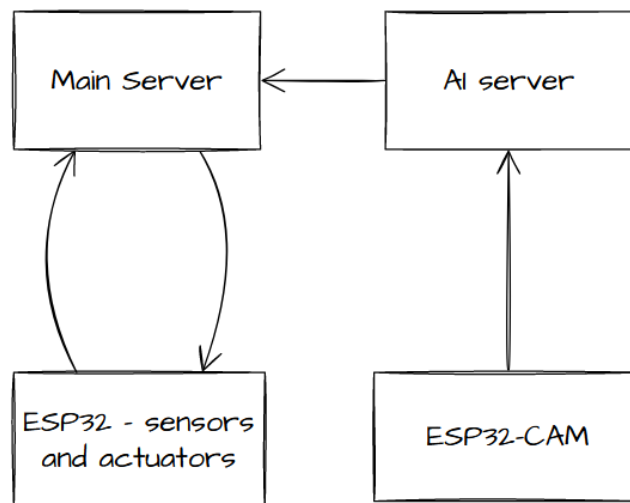


Figure 37: Flowchart of the Greenhouse and its communication pathways

As seen in Figure 38, the decentralized structure of the system emphasizes efficiency and practicality through its intuitive approach towards the real-time monitoring of the greenhouse.

The ESP32 with sensors and actuators periodically sends data to the server and this sample time can be adjusted in the website. In addition, the ESP32-CAM also periodically takes pictures and sends them to be evaluated.

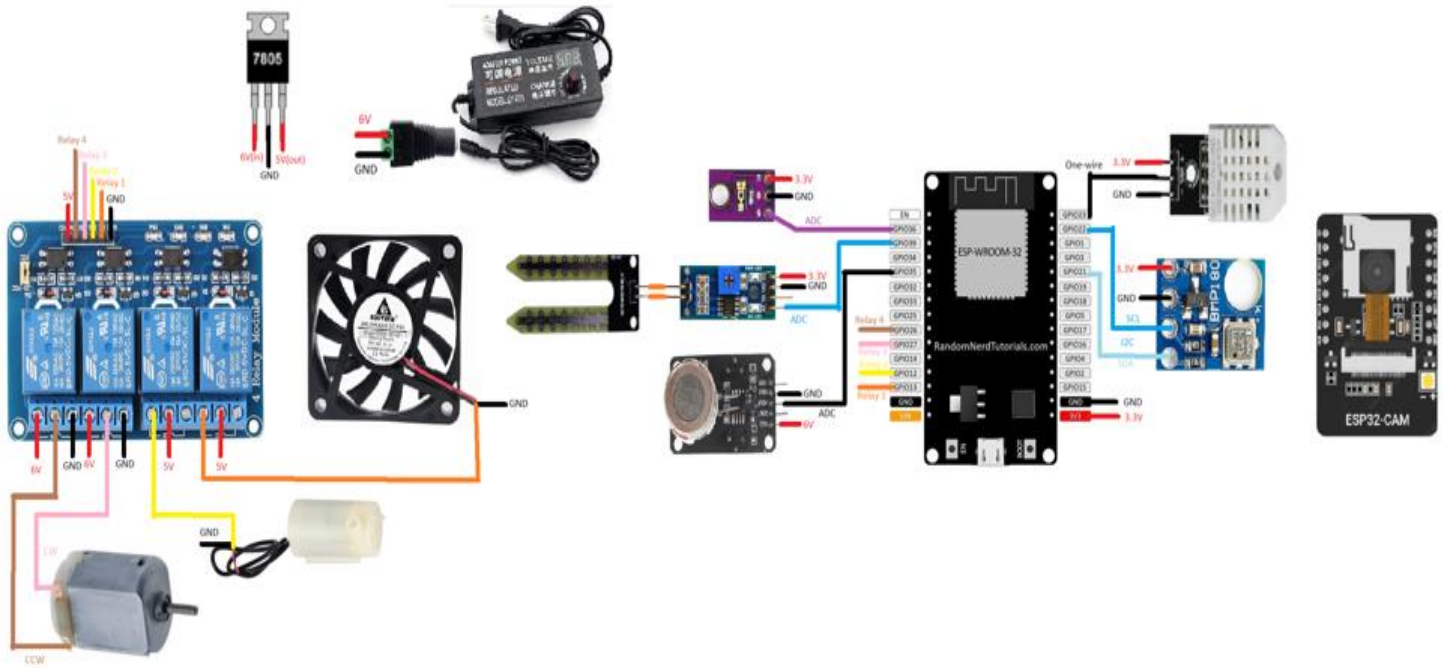


Figure 38: Hardware part of the system

5.2 ESP32 connected to sensors and actuators

5.2.1 Hardware connection of sensors to ESP32

Figures 40 to 44 depict the hardware connections of each sensor to the ESP32.

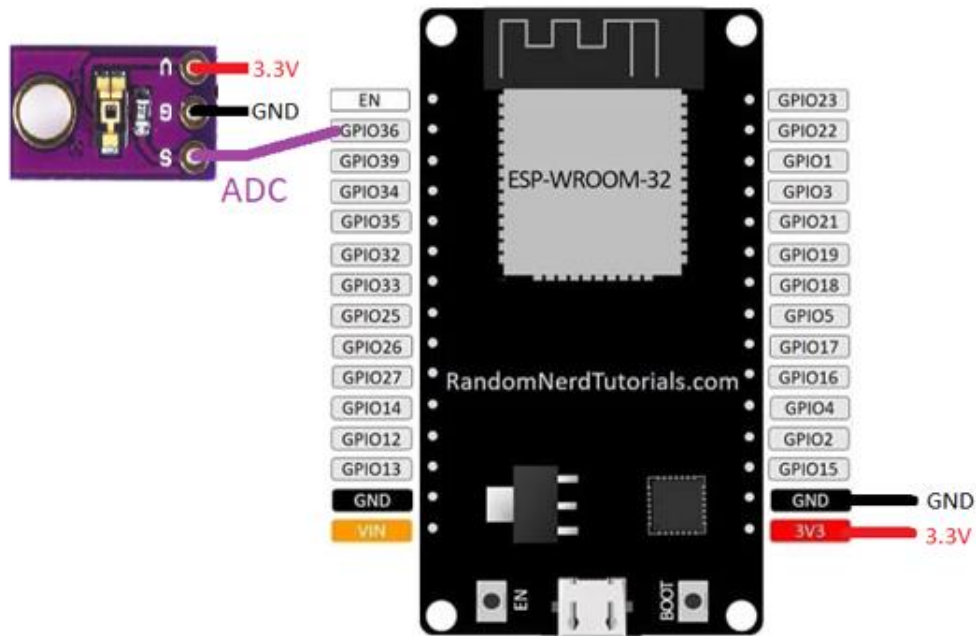


Figure 39: Hardware connection of the TEMT6000 sensor

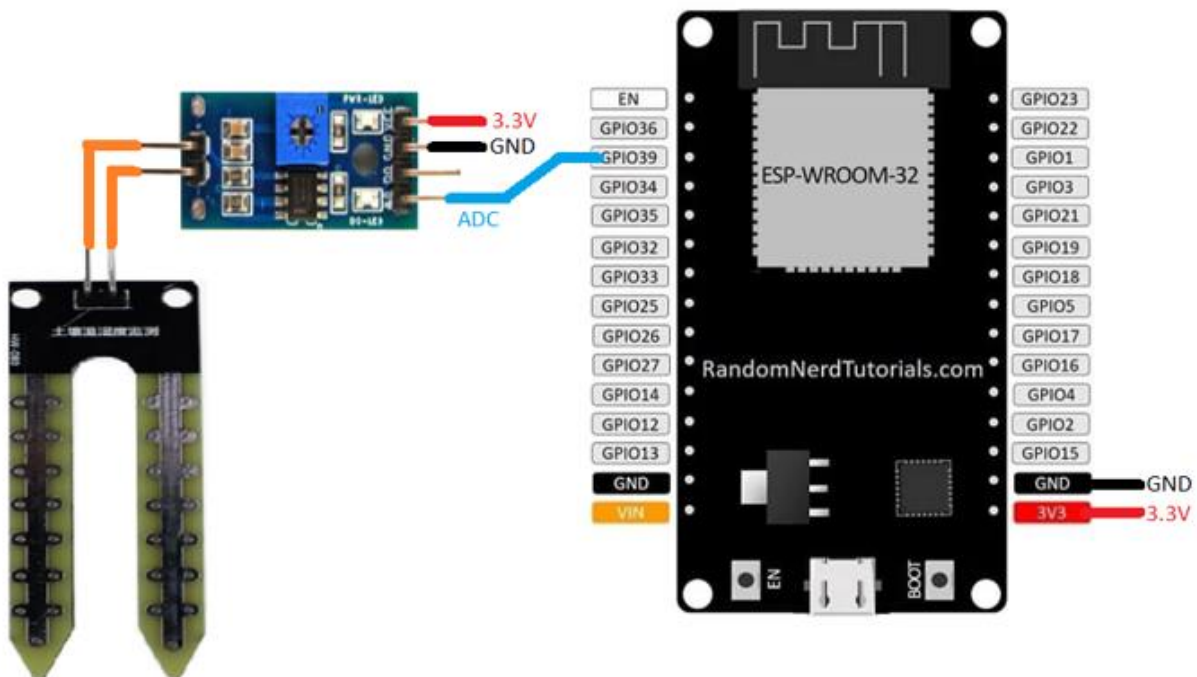


Figure 40: Hardware connection of the Hygrometer soil moisture sensor

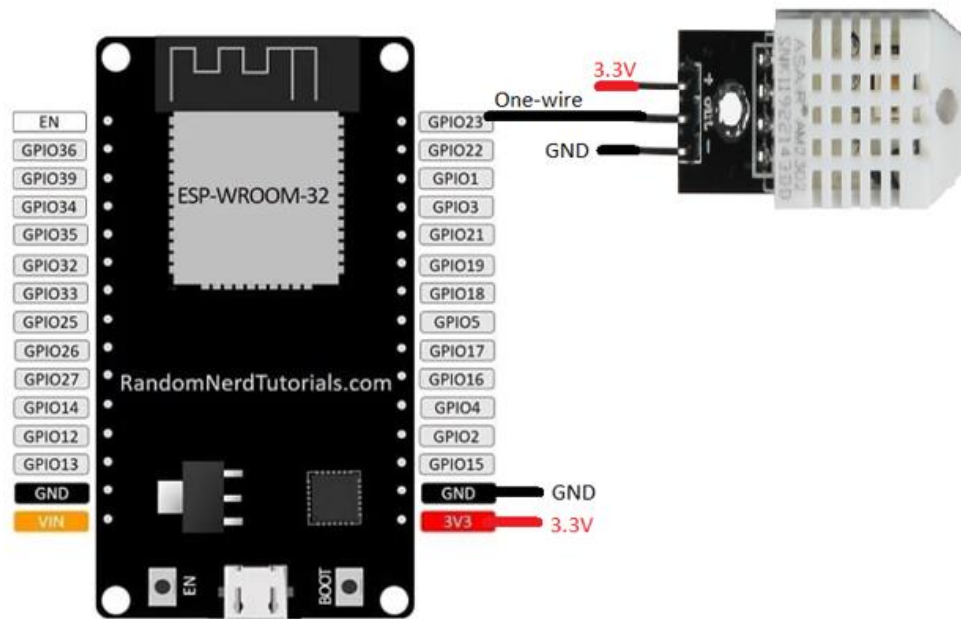


Figure 41: Hardware connection of the DHT22 sensor

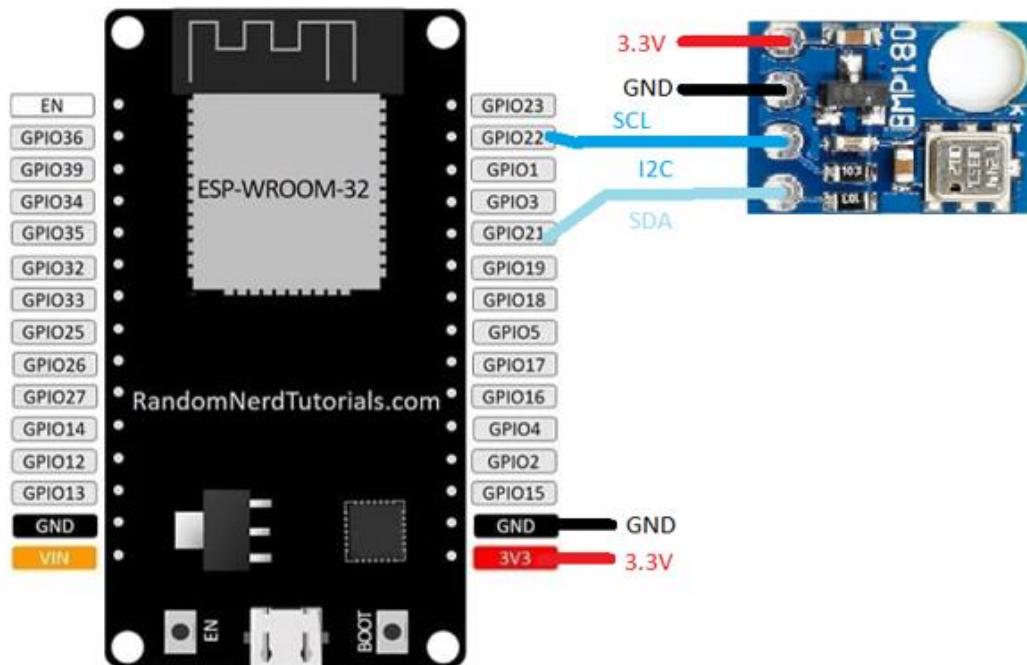


Figure 42: Hardware connection of the BMP180 sensor

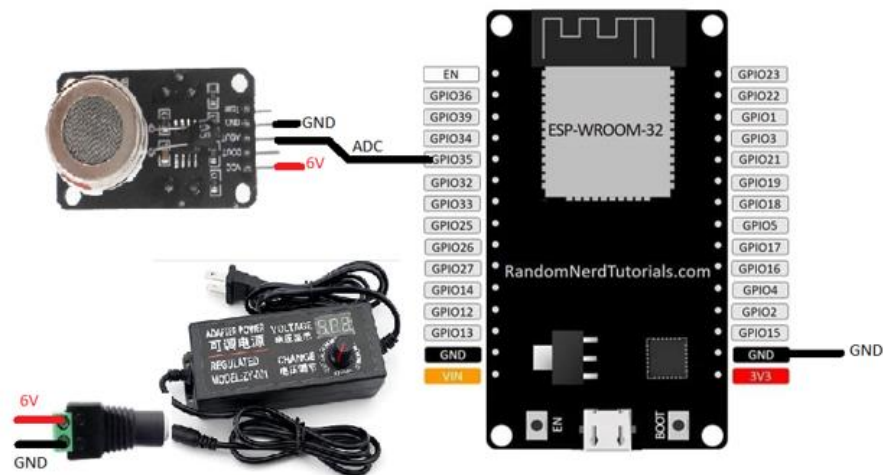


Figure 43: Hardware connection of the CO2 sensor

5.2.2 Hardware connection of actuators to ESP32

Figure 45 to 47 depict the hardware connections of each actuator to the ESP32.

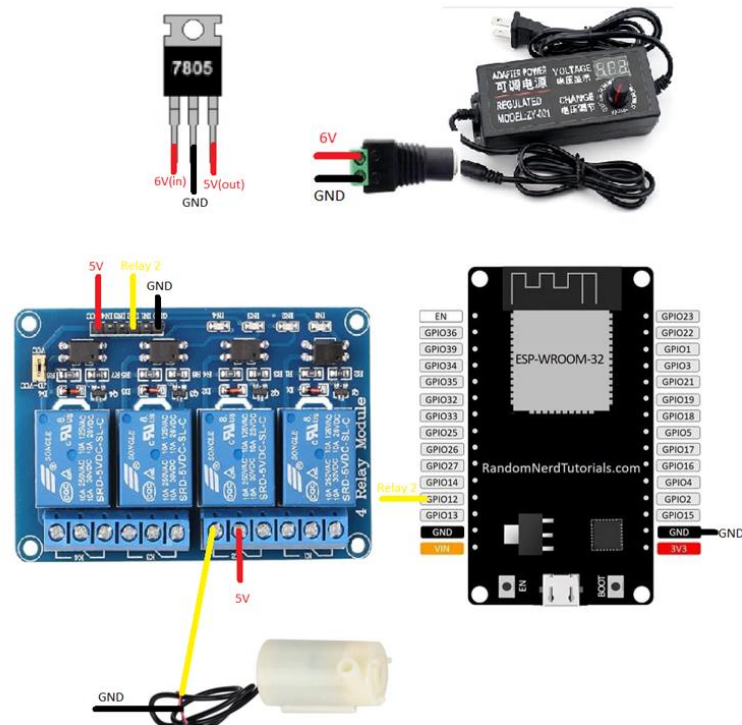


Figure 44: Hardware connection of the water pump

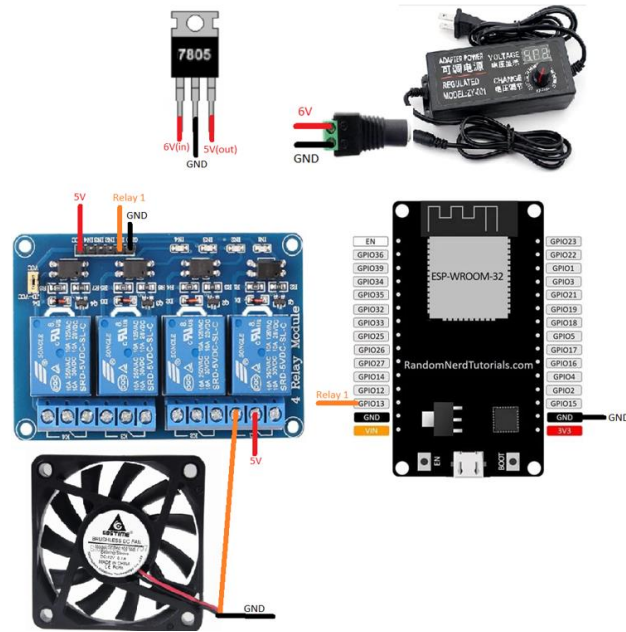


Figure 45: Hardware connection of the fan

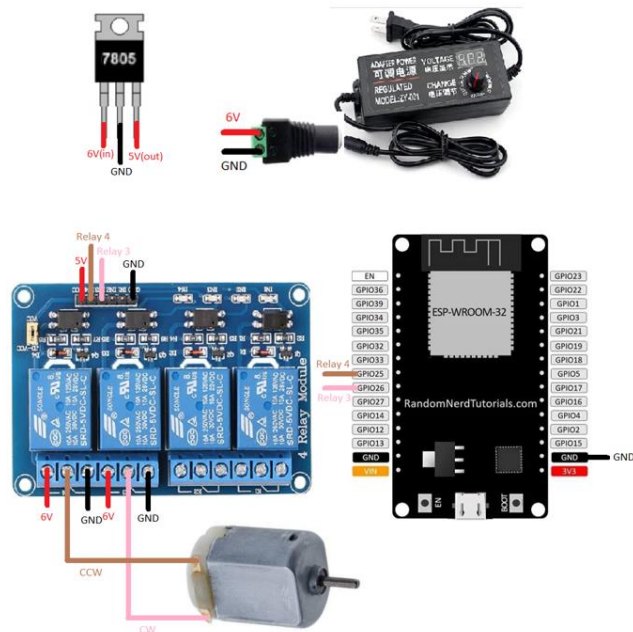


Figure 46: Hardware connection of DC motor

5.2.3 Mechanism

Figure 48 depicts in detail the mechanism of the Arduino code of the ESP32 with sensors and actuators. The full code will be listed in the appendix.

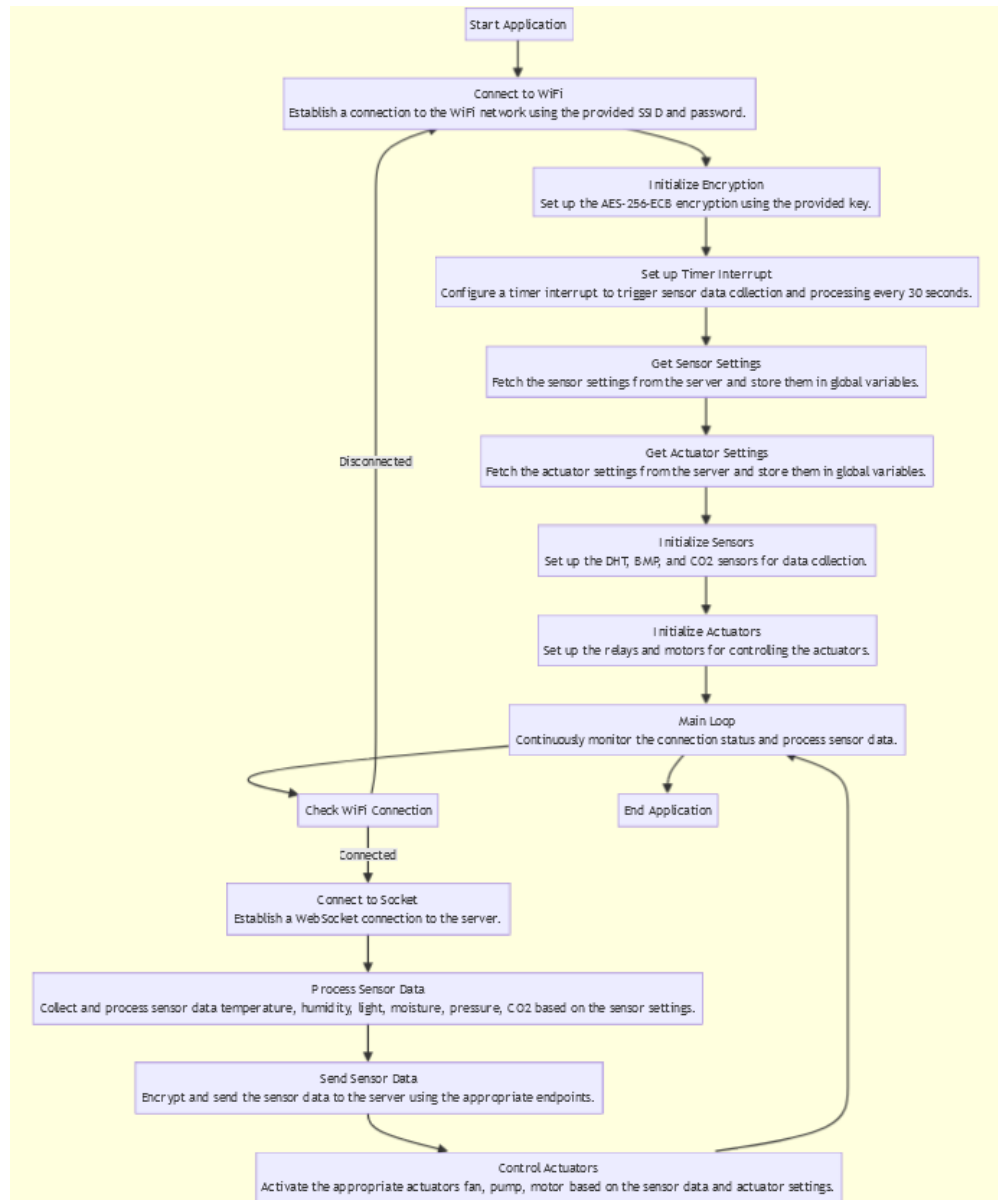


Figure 47: Flowchart of the code for ESP32 with sensors and actuators

5.3 ESP32-CAM

There are no external hardware connections to the ESP32-CAM as. To describe how to works, Figure 49 depicts in detail the mechanism of the Arduino code of the ESP32-CAM. The full code will be listed in the appendix.

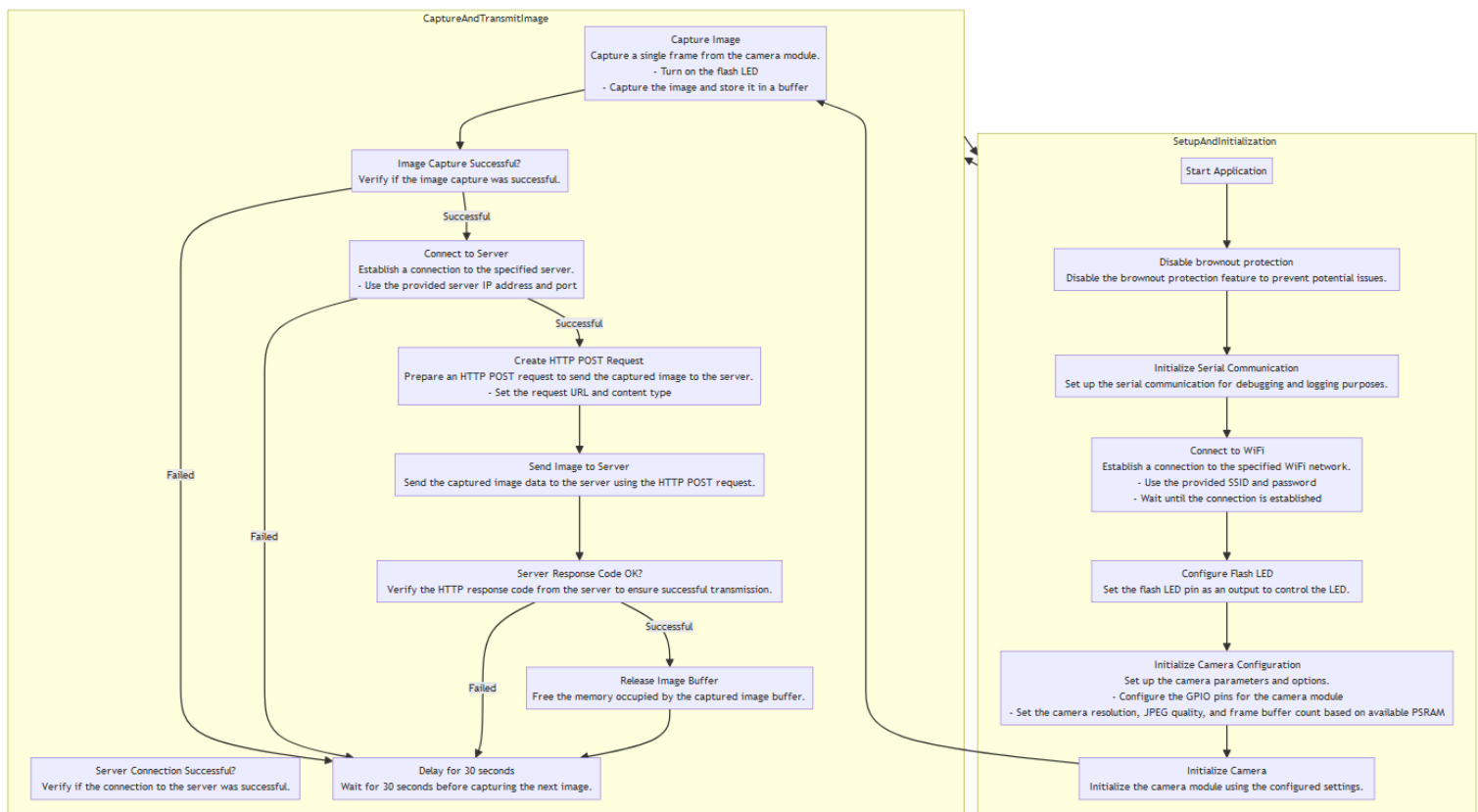


Figure 48: Flowchart of the code for ESP32-CAM

5.4 Results

5.4.1 ESP32 with sensors and actuators – Main server

Through the website, we were able to see the data coming out of the sensors and the actuators responding accordingly to the parameters we set. Figure 50 illustrates data for the light variable represented by a chart.

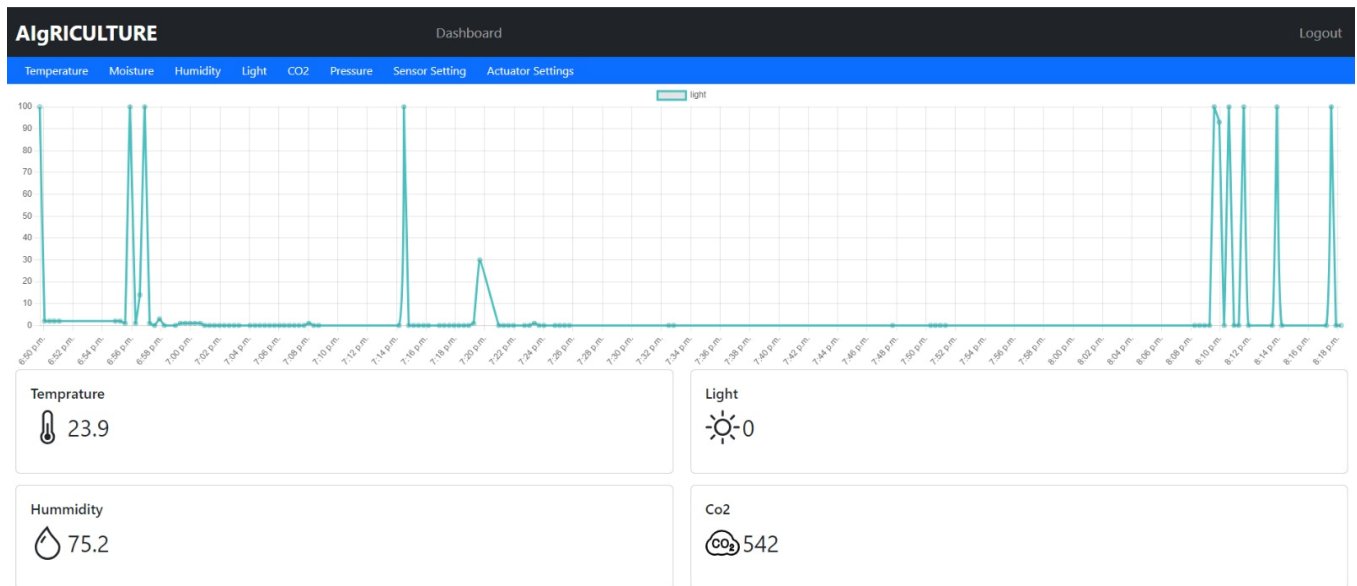


Figure 49: Chart representing the Light environmental variable

5.3.2 ESP32-CAM – AI server

To put the ESP32-CAM and the AI models to test, we've obtained permission to harvest leaves crop leaves from a farmhouse in Tripoli. We were unable to take the pictures directly from the plant itself since we were not able to set up a support for the ESP32-CAM and as a result the pictures came out unstable. Hence, we decided to harvest instead. After taking the pictures, the

ESP32-CAM sent the pictures to the server and received a prediction on each picture. Figures 51 to 58 depict the predicted class outcomes for each corresponding sample.



Figure 50: Strawberry leaf 1 picture taken from ESP32-CAM



```
-----  
Enterd  
Image received and saved as 'received image.jpg'  
192.168.154.98 - - [14/May/2024 18:43:41] "POST /upload/Strawberry HTTP/1.1" 200 -  
(1200, 1600, 3)  
1/1 [=====] - 1s 1s/step  
Healthy 91.40346646308899
```

Figure 51: Strawberry leaf 1 predicted class and confidence



Figure 52: Strawberry leaf 2 picture taken from ESP32-CAM



```

-----
Enterd

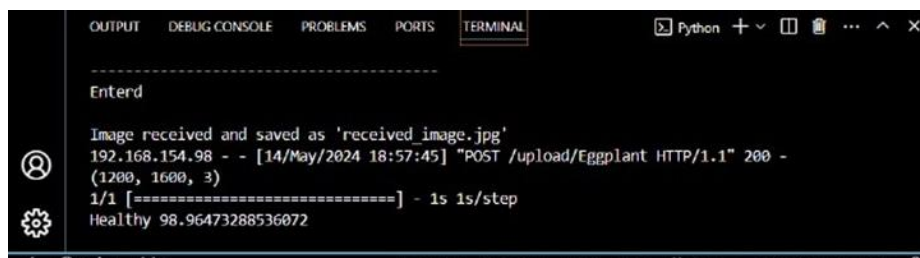
Image received and saved as 'received image.jpg'
192.168.154.98 - - [14/May/2024 18:49:38] "POST /upload/Strawberry HTTP/1.1" 200 -
(1200, 1600, 3)
1/1 [=====] - 1s 1s/step
Healthy 92.44965314865112

```

Figure 53: Strawberry leaf 2 predicted class and confidence



Figure 54: Eggplant leaf 1 picture taken from ESP32-CAM



```

-----
Enterd

Image received and saved as 'received image.jpg'
192.168.154.98 - - [14/May/2024 18:57:45] "POST /upload/Eggplant HTTP/1.1" 200 -
(1200, 1600, 3)
1/1 [=====] - 1s 1s/step
Healthy 98.96473288536072

```

Figure 55: Eggplant leaf 1 predicted class and confidence



Figure 56: Eggplant leaf 2 picture taken from ESP32-CAM

```

Enterd
Image received and saved as 'received_image.jpg'
192.168.154.98 - - [14/May/2024 18:57:18] "POST /upload/Eggplant HTTP/1.1" 200 -
(1200, 1600, 3)
1/1 [=====] - 1s 1s/step
Healthy 78.1797707080841

```

Figure 57: Eggplant leaf 2 predicted class and confidence

Overall, the predictions made on the crops by the models appear to be intune with reality. The confidence level of the predicted class of Eggplant leaf 2 (Figure 53) seems to be lower than the others. This can be attributed to the quality of the dataset it was trained on and the angle of which the camera took the picture from. Unfortunately, we could not obtain more crop leaves to test our other models due to their unavailability. However, the outcome satisfies this proof of concept.

5.5 Future work

Moving forward, the project entails several key areas of focus for optimization. Firstly, we need to enhance the accuracy of AI algorithms to ensure they deliver more precise results. Secondly, we aim to expand and install more sensors and actuators to gain more insight about the

environmental variables and greater control on them. Moreover, there's still a lot of work including the optimization and improvement of the website as there are still many additional features yet to be implemented but were not due to time constraints. Additionally, we aim to further diversify the plant types for disease detection, broadening the platform's applicability and impact within agricultural contexts. All the aforementioned areas collectively represent the next steps in advancing the project towards its full potential.

5.6 Conclusion

To conclude, we have designed and implemented a full automated greenhouse system that is able to collect and data and respond to them via actuator control. We have also developed a website from scratch using various technologies to act as an intuitive form of managing, monitoring, and controlling the automated greenhouse system. In addition, we have implemented an AI able to detect diseases with satisfactory results. Future work includes improvement to AI enhancement, diversification of components, optimization of the website and more. Overall, this project establishes a strong foundation for advancing automated greenhouse management, showcasing the effective combination of multiple technologies in modern farming.

REFERENCES

- [1] Ushakov, D., & Shatila, K. (2022). The impact of agricultural sector on boosting the Lebanese economy throughout the economic crisis. *International Journal of Agricultural Extension*, 10(3), 377-385. doi:<https://doi.org/10.33687/ijae.10.03.4021>

- [2] Singh, R., Kumari, T., Verma, P., Singh, B. P., & Singh Raghubanshi, A. (2021). Compatible package-based agriculture systems: an urgent need for agro-ecological balance and climate change adaptation. *Soil Ecology Letters*. <https://doi.org/10.1007/s42832-021-0087-1>

- [3] The future of food and agriculture | FAO. (n.d.). [Www.fao.org](http://www.fao.org).
<https://www.fao.org/family-farming/detail/en/c/854650/>

- [4] D. T. Mollenkamp, "What is Sustainability?," Investopedia, 23 June 2022. [Online]. Available: <https://www.investopedia.com/terms/s/sustainability.asp>

- [5] U. N. D. Programme, "Sustainable Development Goals," Sustainable Development Goals, 2015. [Online]. Available: <https://www.undp.org/sustainable-development-goals>

- [6] UNESCO, "UNESCO and Sustainable Development Goals," UNESCO, 23 September 2015. [Online]. Available: <https://en.unesco.org/sustainabledevelopmentgoals>

- [7] U. Nations, "Sustainable Development Goals: 17 Goals to Transform our World," United Nations, [Online]. Available: <https://www.un.org/en/exhibits/page/sdgs-17-goals-transform-world>.

- [8] Ahuchaogu, Ifechukwude & Ehiomogue, Precious & Udoumoh, Unwana. (2022). Trends in greenhouse farming technology: A review. Journal of Food Agriculture and Environment. 19. 69-74. 10.1234/4.2022.5613.

- [9] Tawfeek, M. A., Alanazi, S., & El-Aziz, A. A. A. (2022). Smart Greenhouse Based on ANN and IOT. Processes, 10(11), 2402. <https://doi.org/10.3390/pr10112402>

- [10] Maraveas, C. (2022). Incorporating Artificial Intelligence Technology in Smart Greenhouses: Current State of the Art. Applied Sciences, 13(1), 14. <https://doi.org/10.3390/app13010014>

- [11] Tripathy, P. K., Tripathy, A. K., Agarwal, A., & Mohanty, S. P. (2021). MyGreen: An IoT-Enabled Smart Greenhouse for Sustainable Agriculture. IEEE Consumer Electronics Magazine, 1–1. <https://doi.org/10.1109/mce.2021.3055930>

- [12] Rayhana, R., Xiao, G., & Liu, Z. (2020). Internet of Things Empowered Smart Greenhouse Farming. *IEEE Journal of Radio Frequency Identification*, 4(3), 195–211.
<https://doi.org/10.1109/JRFID.2020.2984391>

- [13] Zhang, M., Yan, T., Wang, W., Jia, X., Wang, J., & Klemeš, J. J. (2022). Energy-saving design and control strategy towards modern sustainable greenhouse: A review. *Renewable and Sustainable Energy Reviews*, 164, 112602. <https://doi.org/10.1016/j.rser.2022.112602>

- [14] Siddiqui, Muhammad & Khan, Asim-Ur-Rehman & Kanwal, Neel & Mehdi, Haider & Noor, Aqib & Khan, M.. (2017). Automation and monitoring of greenhouse. 197-201.
10.1109/ICICT.2017.8320190

- [15] Li, N.P., Xiao, Y.M., Shen, L., Xu, Z.Y., Li, B.T. and Yin, C.X. (2019) Smart Agriculture with an Automated IoT-Based Greenhouse System for Local Communities. *Advances in Internet of Things* , 9, 15-31. <https://doi.org/10.4236/ait.2019.92002>

- [16] Arduino Nano 33 BLE Sense. (n.d.). Arduino Online Shop. <https://store-usa.arduino.cc/products/arduino-nano-33-ble-sense>

- [17] ESP32. ESP32 Wi-Fi & Bluetooth SoC | Espressif Systems. (n.d.).
<https://www.espressif.com/en/products/socs/esp32>
- [18] Ltd, R. P. (Trading). (n.d.). Buy a Raspberry Pi 4 Model B. Raspberry Pi.
<https://www.raspberrypi.com/products/raspberry-pi-4-model-b/>
- [19] Adafruit Industries. (2019). DHT22 temperature-humidity sensor + extras. Adafruit.com.
<https://www.adafruit.com/product/385>
- [20] Addicore DS18B20 Digital Temperature Probe Waterproof. (n.d.). Addicore. Retrieved December 16, 2023, from <https://www.addicore.com/products/ds18b20-digital-temperature-probe-waterproof>
- [21] High Precision Atmospheric Pressure Sensor Module GY-BMP280. (n.d.). Electroslab. Retrieved December 16, 2023, from <https://electroslab.com/products/high-precision-atmospheric-pressure-sensor-module-gy-bmp280>
- [22] Adafruit Industries. (2019). DHT22 temperature-humidity sensor + extras. Adafruit.com.
<https://www.adafruit.com/product/385>

- [23] Humidity Sensor, 2 %, 5 VAC, 0% to 100% Relative Humidity, SIP, 2 Pins, 15 s. (2021). Farnell.com. <https://uk.farnell.com/honeywell/hch-1000-001/sensor-humidity-capacitive/dp/1566264>
- [24] B+B Thermo-Technik | Resistive humidity sensor SHS-A4L | Measurement Technology. (n.d.). <https://shop.bb-sensors.com/>. <https://shop.bb-sensors.com/en/Measurement-by-branches/Building-automation/Resistive-humidity-sensor-SHS-A4L.html>
- [25] GY-68 BMP180 Digital Barometric Pressure Sensor. (n.d.). Electroslab. Retrieved December 16, 2023, from <https://electroslab.com/products/gy-68-bmp180-digital-barometric-pressure-sensor>
- [26] Industries, A. (n.d.). MPL3115A2 - I2C Barometric Pressure/Altitude/Temperature Sensor. [www.adafruit.com](https://www.adafruit.com/product/1893). <https://www.adafruit.com/product/1893>
- [27] Gy 63 Ms5611 01ba03 High Precision Pressure Sensor Height Sensor Module. (n.d.). Indiamart.com. Retrieved December 16, 2023, from <https://www.indiamart.com/proddetail/gy-63-ms5611-01ba03-high-precision-pressure-sensor-height-sensor-module-25402337791.html>

- [28] Gravity: Analog CO₂ Gas Sensor (MG-811 Sensor). (n.d.). Arduino Online Shop.
<https://store-usa.arduino.cc/products/gravity-analog-co2-gas-sensor-mg-811-sensor?selectedStore=us>
- [29] MQ-135 Air Quality Sensor Detection Module. (n.d.). Electroslab. Retrieved December 16, 2023, from <https://electroslab.com/products/mq-135-air-quality-sensor-detection-module>
- [30] AG, S. (n.d.). *SCD30-co₂ accuracy of $\pm(30 \text{ ppm} + 3\% \text{ MV}) @400-10000 \text{ ppm}$* . CO.
<https://sensirion.com/products/catalog/SCD30>
- [31] AG, S. (n.d.). *SCD30-co₂ accuracy of $\pm(30 \text{ ppm} + 3\% \text{ MV}) @400-10000 \text{ ppm}$* . CO.
<https://sensirion.com/products/catalog/SCD30>
- [32] VL53L0X Time of Flight Sensor (2m Precision Distance Measurements). (n.d.). Future Electronics Egypt. Retrieved December 16, 2023, from <https://store.futureelectronics.com/products/vl53l0x-time-of-flight-sensor-precision-distance-measurements#>
- [33] TSL2561 Digital Luminosity/Lux/Light Sensor Buy 9,00 USD | Sumozade.com. (n.d.).
 Www.sumozade.com. Retrieved December 16, 2023, from
<https://www.sumozade.com/product/tsl2561-digital-luminosity-lux-light-sensor>

- [34] Grove - Moisture Sensor. (n.d.). Solarbotics Ltd. Retrieved December 16, 2023, from <https://www.solarbotics.com/product/29064/>
- [35] FC-28 Soil Moisture Sensor - Analog and Digital Outputs. (n.d.). Maker Store PTY LTD. Retrieved December 16, 2023, from <https://www.makerstore.com.au/product/elec-sensor-soilhumidity/>
- [36] Gravity: Analog Soil Moisture Sensor for Arduino. (n.d.). Wwww.dfrobot.com. <https://www.dfrobot.com/product-599.html>
- [37] Arducam 0.3MP OV7675 20-pin DVP Camera Module for Arduino GIGA R1 WiFi. (n.d.). Arduino Online Shop. <https://store-usa.arduino.cc/products/arducam-camera-module>
- [38] OV2640 Camera Board Camera module, 2 Megapixel. (n.d.). Wwww.waveshare.com. Retrieved December 16, 2023, from <https://www.waveshare.com/ov2640-camera-board.htm>
- [39] EKT: ARDUINO WIFI + BT ESP32-WROOM. (n.d.). Wwww.katranji.com. <https://www.katranji.com/Item/457125>

- [40] Industries, A. (n.d.). ESP8285 SMT Module - ESP8266 with 1MB FLASH. Wwww.adafruit.com. Retrieved December 16, 2023, from <https://www.adafruit.com/product/4065>
- [41] Lorida Esp8266-12e Esp-12e Esp8266 Esp 8266 Wireless Transceiver Receiver Wifi Module Nodemcu Esp8266 Esp12e Esp 12e - Buy Esp 12e,Development Board Nodemcu Esp8266 Wifi Module Board Esp-12f Esp8266-12e Ch340 Esp-01,Esp8266 Esp-01 Esp-01s Esp-01m Esp-01f Esp-07 Esp-07s Esp-12s Esp-12e Esp-12f Esp Serial Wifi Wireless Module Adapter Board Product on Alibaba.com. (n.d.). Wwww.alibaba.com. Retrieved December 16, 2023, from https://www.alibaba.com/product-detail/Lorida-ESP8266-12E-ESP-12E-esp8266_1600879148397.html?spm=a2700.7724857.0.0.717c658aE8rymd
- [42] *5V 4-phase stepper motor*. Electroslab. (n.d.-a). https://electroslab.com/products/5v-4-phase-stepper-motor?_pos=1&_sid=5577f88a0&_ss=r
- [43] *3v DC Motor DIY*. Electroslab. (n.d.-a). https://electroslab.com/products/dc-motor?_pos=1&_sid=884e744c7&_ss=r
- [44] *Raspberry pi cooling fan*. Electroslab. (n.d.). https://electroslab.com/products/raspberry-pi-cooling-fan?_pos=32&_sid=6518ac640&_ss=r

- [45] 5V 3500 rpm DC cooling fan. Electroslab. (n.d.-a). https://electroslab.com/products/5v-3500-rpm-dc-cooling-fan?_pos=3&_sid=6518ac640&_ss=r

- [46] Hu, Z. (2010, October 1). I2C Protocol Design for Reusability. IEEE Xplore. <https://doi.org/10.1109/ISIP.2010.51>

- [47] An introduction to I2C and SPI protocols | IEEE Journals & Magazine | IEEE Xplore. (n.d.). Ieeexplore.ieee.org. <https://ieeexplore.ieee.org/document/4762946>

- [48] Dhaker, P. (2018, September). Introduction to SPI Interface | Analog Devices. Wwww.analog.com. <https://www.analog.com/en/analog-dialogue/articles/introduction-to-spi-interface.html>

- [49] Nanda, U., & Pattnaik, S. K. (2016, January 1). Universal Asynchronous Receiver and Transmitter (UART). IEEE Xplore. <https://doi.org/10.1109/ICACCS.2016.7586376>

- [50] IEEE SA - IEEE 754-2019. (n.d.). SA Main Site. <https://standards.ieee.org/ieee/754/6210/>

- [51] Crow, B. P., Widjaja, I., Kim, J. G., & Sakai, P. T. (1997). IEEE 802.11 Wireless Local Area Networks. IEEE Communications Magazine, 35(9), 116–126.
<https://doi.org/10.1109/35.620533>
- [52] User Datagram Protocol. (1980, August 1). IETF. <https://www.rfc-editor.org/info/rfc768>
- [53] RFC 9293 - Transmission Control Protocol (TCP). (n.d.). Datatracker.ietf.org.
<https://datatracker.ietf.org/doc/html/rfc9293>
- [54] J. Postel. (2019). Internet Protocol. Rfc-Editor.org. <https://www.rfc-editor.org/info/rfc791>
- [55] Anon, "JPEG - The JPEG Standard," www.liquisearch.com, [Online]. Available:
https://www.liquisearch.com/jpeg/the_jpeg_standard