



Nile Higher Institute for Engineering & Technology
Communication & Electronics Department
2021 - 2022



Green Hydroponic Vertical Farm System

Using RaspberryPi and IoT Technology

Under Supervision

Asstist. Prof. Hegazi Ibrahim

Eng. Nesma Abdulmawla

2021-2022

Contents

- Team Structure
- Acknowledgements
- List of Publications
- List of Figures
- List of Tables
- List of Abbreviations
- Abstract
- Problem Statement

- **Chapter 1: Introduction**
 - 1.1. Project Motivations
 - 1.2. Project Aims
 - 1.3. Project Contributions & Relation with the Environment

- **Chapter 2: Related Work**

- **Chapter 3: Proposed System**
 - 3.1. System Analysis and Design

- **Chapter 4: System Implementation**
 - 4.1. System Layouts
 - 4.2. System 3D Model
 - 4.3. System Components
 - 4.4. System Circuits Design
 - 4.5. System Prototype

- **Chapter 5: Conclusions and Future work**
 - 5.1. Conclusions
 - 5.2. Challenges & Future work
- **Appendix:**
 - I. Software codes.
 - II. Engineering Standard (IEEE)
 - III. Economically Feasibility Study

Team Structure

01

Hardware Team

- Mohamed Al-Basuny.
- Mostafa Ahmed.
- Ahmed Khaled.

02

Middleware Team

- Abdelhai Abdelhakeem.
- Ebraheem Sabry.
- Rana Nasser.

03

Software Team

- Merna Sheha.
- Sohaila Anter.
- Amr Sakaria.
- Abdelrahman Ahmed.

Acknowledgements

First and foremost, we feel always indebted to ALLAH, the most kind, and the most merciful, for his great love and care all through my life.

It is a great honor to express my deep gratitude to Assist. Prof. Hegazi Ibrahim for his remarkable supervision and continuous encouragement. With his advice and care during this work.

Also, We are very grateful and truly indebted to Assist. Lect. Nesma Abd El-Mawla who's not only served as my supervisor but also for her encouragement, valuable guidance, and indispensable help. Her words of advice, his trust, and his patience and understanding helped us to fulfill this work.

**Green.com Team
2022**

List of Publications

**Hegazi Ibrahim, Nesma Abd El-Mawla,
Green Team, Proposed Model for
Sustainable and Scalable Vertical Farm, Nile
Journal of Communication & Computer
Science, Feb. 2022.**

List of Figures

Fig.	Figure Caption	Page
Fig. 1.	Share of Global Food Loss and Waste.	13
Fig. 2.	Agriculture using IOT, [Source: Javapoint].	16
Fig. 3.	Economic Viability of Vertical Farming.	21
Fig.4.	Vertical Farming Development Releases	32
Fig. 5.	Vertical Farm in Singapore.	36
Fig. 6.	System Use Case Diagram.	36
Fig. 7.	System Sequence Diagram.	37
Fig. 8.	System Data-Flow Diagram.	38
Fig. 9.	System Flowchart.	38
Fig. 10.	Proposed System Circuits Design.	44
Fig. 11.	Proposed Model Scenario.	46
Fig. 12.	Mobile Application Layouts, First page & Signup Layouts.	47
Fig. 13.	login & check password Layouts.	48
Fig. 14.	Home Page (Control) Layout	49
Fig. 15.	Reports Layout	50
Fig. 16.	Green page & help Layouts.	51
Fig. 17.	The article & list Layouts.	52
Fig. 18.	Email Page & contact us Layouts.	53
Fig. 19.	Server Layouts.	54
Fig. 20.	Server CODE Layouts.	55

Fig. 21.	Sign up Layouts.	56
Fig. 22.	login page Layouts.	57
Fig. 23.	Home Page Layouts.	60
Fig. 24.	Reset Password Layouts.	60
Fig. 25.	Google Button Layouts.	61
Fig. 26.	Fig.26. MySQL First page.	62
Fig. 27.	Fig.27. Search page Layouts	78
Fig. 28.	SQL Report Layouts.	79
Fig. 29.	DC Motor Control Circuit.	80
Fig. 30.	Sensors connected circuit.	83
Fig. 31.	Full System Circuit.	84
Fig. 32.	Hardware platform.	85
Fig. 33.	power box of the project.	86
Fig. 34.	Water Float Switch.	87
Fig. 35.	Water and Manual Water Float Switch	88
Fig. 36.	Out view	89

List of Tables

Table	Table Caption	Page
Table 1	Pros and Cons of Vertical Farming.	14
Table 2	Project Components and Cost.	46
Table 3	Project Development Releases (Economically Feasibility Study)	145

List of Abbreviations

Abbrev.	Full Caption
UN	United Nations
Wi-Fi	Wireless Fidelity
IoT	Internet of Things
HTTP	Hypertext Transfer Protocol
LED	Light-Emitting Diode
NAS	Network-Attached Storage
IP	Internet Protocol
DHCP	Dynamic Host Configuration Protocol
TCP/IP	Transmission Control Protocol/Internet Protocol

Abstract

This project summarizes the implementation of vertical farming that uses a Wi-Fi network (**IEEE 802.11**) to communicate with sensors (**IEEE 802.15.4**) and actuators from multiple nodes. It addresses the issue of ordinary vertical farms, which require the user to monitor it occasionally to provide fertilizer and water. The system can be easily configured to automatically control the supply of nutrients, water, and light requirements for various plant types through a mobile application enabled Interface (**IEEE 802.15.2**). The mobile application (**IEEE P 2301**) dashboard can further provide a complex analysis of the whole system by collecting values from different sensors. The designed vertical farm system is power efficient, self-sustained, and can be set up easily by the user as each vertical rack acts as a single node or module according to the pre-designed circuit simulation (**IEEE 1074**) and implementation phase. The user only needs to plant the seeds and fill up the tanks. Due to the modular approach, the system is also scalable without the requirement of more complicated materials or wiring.

Keywords: *Wi-Fi, Raspberry Pi, Scalable, Sustainable, Vertical Farming.*

Problem Statement

- **Solution to increase food production to cater to the growing population:**
 - **Expansion and intensification**

1. Expansion:

- Land Degradation.
- Destruction of Natural Habitats.
- Deforestation.

2. Intensification:

- Genetically Modified Crops.
- Increasing Irrigation.
- Increasing Fertilization.



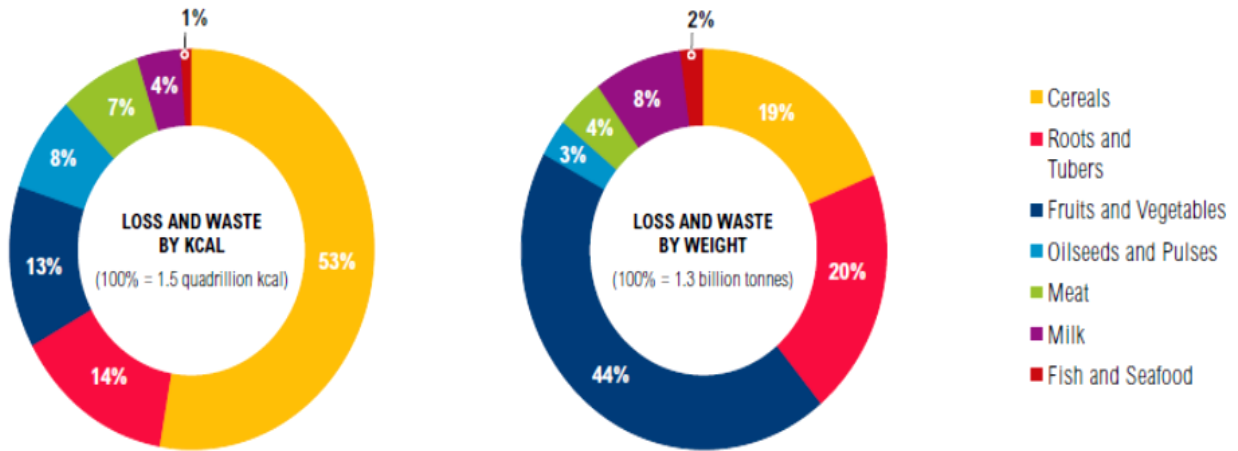
CHAPTER 1

Introduction

Chapter 1: Introduction

Vertical farming can be viewed as a way to combat the world's ever-increasing metropolitan areas, which are reducing the availability of agricultural land [1-3]. Figure 1 presents the share of global food loss and waste. Vertical farming opens the door to individuals being able to produce herbs and vegetables indoors with the assurance that the yield is 100% organic and hence considerably healthier. In addition, vertical farms are better for the environment than traditional farming because they have less carbon imprint (UN, 2019) [4]. Table 1 briefs the pros and cons of vertical farming.

Share of global food loss and waste by commodity, 2009



Data source: Reducing food loss and waste, [World Resources Institute](#), 2013.

Fig.1. Share of Global Food Loss and Waste.

Ordinary vertical farms, on the other hand, will require users to take time out of their busy schedules to give water and fertilizer, as well as a source of light, which can be natural or artificial. If sunlight is preferred, the user may need to locate a location with the best availability. Artificial lights, on the other hand, must be efficient in two ways: they must supply the correct color for the specific plant species [5-6], and they must be energy efficient to avoid high electricity bills.

Table 1. Pros and Cons of Vertical Farming [7-10].

Pros	Cons
<ul style="list-style-type: none">• It offers a plan to handle future food demands.	<ul style="list-style-type: none">• It could be very costly to build and economic feasibility studies haven't yet been completed.
<ul style="list-style-type: none">• It allows crops to grow year-round.	<ul style="list-style-type: none">• Pollination would be very difficult and costly.
<ul style="list-style-type: none">• It uses significantly less water.	<ul style="list-style-type: none">• It would involve higher labor costs.
<ul style="list-style-type: none">• Weather doesn't affect the crops.	<ul style="list-style-type: none">• It relies too much on technology and one day of power loss would be devastating.
<ul style="list-style-type: none">• More organic crops can be grown.	
<ul style="list-style-type: none">• There is less exposure to chemicals and diseases.	

This research proposes a solution in the form of an IoT-enabled vertical farm to address these issues (**IEEE P2301**). IoT is a network of items with a variety of sensors, software/applications, network connectivity, and computing capabilities that can connect, alter, and share data via the internet to enable smart solutions. IoT is already providing new potential for solving a variety of environmental challenges such as clean water, landfill waste, deforestation, and air

pollution, and will eventually aid in reducing the environmental effects of human activities. According to a report, the market for IoT will grow significantly around the world because all major corporations believe in the importance of IoT for both human life and the environment [11-12].

In this study, we offer a one-of-a-kind, totally self-contained, and energy-efficient vertical farm that can be placed anywhere indoors at the user's convenience. The user only needs to put up the shelf, rack the soil containers, place the seeds inside, and fill the water and fertilizer tanks. The user can then access the one-of-a-kind web-enabled dashboard that is offered. We call our unique ecosystem software the “Digital Farmer”, which as the name suggests, takes management of the farm on behalf of the user generating promising harvests. The user can also configure the farm as well as get important real-time data.

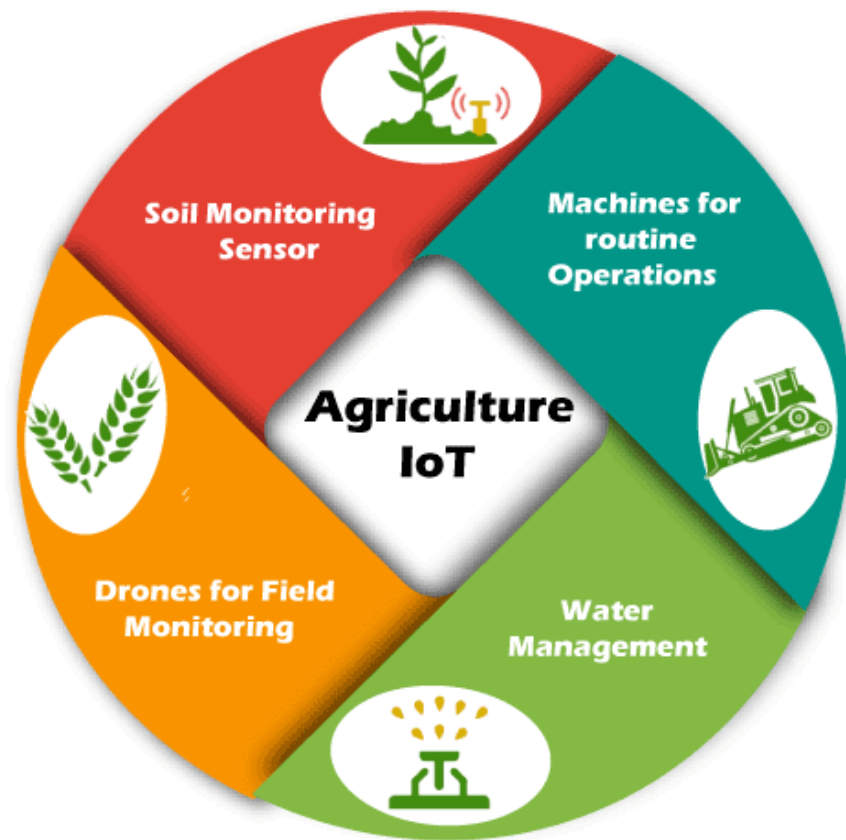


Fig.2. Agriculture using IoT, [Source: Javapoint].

The Digital Farmer kit includes extensive software features that allow for various configurations for cultivating a variety of herbs and vegetables, as well as a watering and fertilizing system that is automated. On-board Wi-Fi modules were used to build a communication link between the shelves and the microcontroller-based server. Through this, the REST API is used to seamlessly convey control signals to the server. The farm's data and status

information, on the other hand, is sent via HTTP. In the Proposed Solution section, there is further information on the system.

The necessity for a new agricultural system is unavoidable, given the predicted growth in the global population and the current climate change epidemic [1], [13]. Vertical farming appears to be the ideal option. The farm is just several layers of a typical farm piled on top of one other, each with its light, temperature control, water, and fertilizers. Despite extensive research, there has yet to be a solution capable of providing a fully self-sustaining, scalable, cost-effective, power-efficient, and water-efficient farm available to the user as a plug-and-play product.

1.1. Project Motivations

The motivation for this project in the smart agriculture area is that the required space to grow herbs, vegetables and flowers for achieving worldwide food demand becomes smaller and smaller with the crazy increase in human population. Although the existence of many agriculture solutions for resources consumption,

it is still an open area. Today agriculture accounts (on average) for 70% of all freshwater with drawls globally. There is also excessive use of chemical fertilizer that effect badly on the environment and the soil. Due to resource limitations in such as (i) Land; (ii) Cost; (iii) Water; (iv) Crops; so it is essential to implement and develop smart efficient agricultural system over IoT to:

- (i) avoid resource consumption.
- (ii) overcome traditional agriculture issues.
- (iii) improve crops quality.
- (iv) avoid using chemical fertilizers.

1.2. Project Aims

This project aims to:

- (i) spots the light on vertical farming and eco-friendly agriculture.
- (ii) introduce a new eco-friendly system for solving the problem of resources limitation.

(iii) increase the production of agricultural crops for achieving worldwide food demand.

(iv) preserves the soil where the use of chemical fertilizer.

(v) decrease resources consumption by recycling the water required for growing the plants.

(vi) bioremediation and reduce pollutions.

1.3. Project Contributions

This Project is a smart solution over IoT using vertical hydroponics (a growing system based on water, not soil) growers can farm a variety of crops indoors and with minimal square footage. The vertical farm is completely climate controlled, uses LED lights to promote rapid growth and connects to the app with Wi-Fi, making it easy for farmers to manage their crops remotely. The project performs the following enhancements:

- (i) Guarantee the secure connection between sensor nodes and server.

(ii) Overcome temperature issues: temperature and humidity is isolated and it cannot be affected by ambient temperature.

(iii) Overcome economic issues: the system is relatively small and cheaper than other system.

(iv) Enhancement in overall system control: This system can be controlled by a server and mobile application and this is a leading point in comparison with other systems. Also, the app provides full control over the care of the plants, allowing for adjustments to water cycles, humidity, temperature and lighting.

(v) Decrease water consumption: overall water quantity is saved up to a percent greater 95% than other traditional agriculture systems.

1.4 Economic Viability of Vertical Farming: Overcoming financial obstacles to a greener future of farming



Fig.3. Economic Viability of Vertical Farming.

Photo: Indoor and vertical farming may be part of the solution to rising demands for food and limited natural resources.

Economic Viability of Vertical Farming: Overcoming financial obstacles to a greener future of farming With world population projected to reach 9 billion by 2050, the United Nations Food and Agricultural Organization portends a truly alarming statistic: the

amount of available arable land will decrease to only one-third of the amount that was available in 1970. As climate change is expected to increase drastically over the next few decades, we will see more instances of intensifying heat stress, droughts, and damage to ecosystems that will only further hinder our ability to grow crops to feed the billions of people that exist on our planet. A new, more innovative method of food production is desperately needed. Vertical farming is one promising alternative that consists of plants stacked vertically in tall built environments, usually in urban hubs. This method of farming uses less than one percent of the land conventional agriculture does and consumes one percent of the amount of water. Vertical farming has the potential to significantly increase food production while reducing the environmental footprint of the agricultural sector by reducing land, water, chemical, and fertilizer use and increasing overall efficiency. While the environmental benefits are well documented, the economic feasibility of vertical farming is the key barrier. However, I argue that while there are large upfront costs associated with vertical

farming, the economic benefits associated with increased efficiency and decreased resource use tied with its increased sustainability clearly outweigh these costs.

The main barrier to vertical farming implementation are the large upfront costs. High capital expenditures are a result of both the higher real estate values per square meter in urban centers as well as the infrastructure needed to regulate plant growth. For example, in Australia, the average cost of land per square meter in the city center of Melbourne is \$3,491, whereas the same amount of land costs only \$0.40 in rural areas where conventional farms exist. Furthermore, because plants are grown indoors and have limited access to sunlight, steady access to LED lights are a necessity. Opponents of vertical farming argue that the amount of energy required to produce enough light for photosynthesis of crops in just one 37-story vertical farm facility requires a net total of 3.5 GWh of electricity at about \$6 million a year. Overall, just one vertical farm facility may require

hundreds of millions of dollars in upfront infrastructural costs and equipment.

Many elements of vertical farming help outweigh the upfront costs once a facility is up-and-running. For one, while real estate prices may be higher in urban areas, the localized production of crops near high density population centers will greatly reduce transportation costs. In turn, this will eliminate fluctuations in wholesale produce prices due to the variability of fuel prices at any given time.

The most important factor that makes vertical farming economically viable, however, is the controlled conditions under which it functions. Unlike traditional agricultural production, the external environmental conditions that impose further costs on farmers have a very limited effect on vertical farms. For example, the need for fertilizers and pesticides (and the ensuing costs thereof) would be almost entirely eliminated because the crops would not be open to the elements and would thus be inaccessible to pests. Because the effects of seasonality are also diminished due to internal regulation

of temperature, humidity, and access to light and water, conditions can be fine-tuned to optimal levels. This results in a large increase in production rate, sometimes producing crops at yields 530 times greater than would be produced on conventional farmlands of the same size. Additionally, control of nutrient levels and ambient temperatures will optimize the rate of plant growth and increase its nutritional value. For example, at vertical farm Vegata Farm in Tokyo, lettuce that would have taken at least 60 days to grow in the field only takes about 40. The combination of higher yields and more efficient production rates help balance and even counteract high capital expenditures.

Currently, however, private investments support the bulk of vertical farm implementation. The National Science Foundation recently granted researcher Neil Mattson at Cornell University \$2.4 million to investigate the workforce development and economic viability of crops grown in a controlled environment in order to optimize horticultural practices. SoftBank Vision Fund invested \$200 million

in Plenty, a startup company that is using vertical farming to produce millions of pounds of greens and is planning on expanding to urban areas in China. As companies begin to get their feet on the ground and adapt internal measures and improve LED lighting efficiency, productivity can be maximized and quality of produce will be consistent year-round. Streamlining of vertical farming and further research has the potential to reduce the cost in the future. Overall, increased investment and funding must be used towards tackling upfront costs of vertical farming. Fig.4 discusses the development phases for the vertical green farming techniques compared with the benchmark version one.



Fig.4. Vertical Farming Development Releases



CHAPTER 2

Related Work

Chapter 2: Related Work

Many researchers around the world have developed different vertical farming systems. Tsai and Liang [14] suggested a vertical farm monitoring system. The system, on the other hand, is merely a proof of concept until it is put to the test on a farm. The Arduino Yun is the sole microcontroller used, and only a single temperature sensor is tested. Sensory data is collected and written to a file, from which it is uploaded to a database and shown on the dashboard.

While drip irrigation is one of the most efficient watering systems, hydroponics is a method of growing plants or vegetables without the use of soil [13].

The application of hydroponics in small vertical farms is investigated by Ruengittinun et al. [15]. The authors' system is similar to the one described in this paper, except it is geared toward hydroponics. Sensor data is collected and stored in a database via a lightweight messaging protocol called MQTT broker once again in

the system design. The system dashboard is represented by a mobile application with manual and automatic modes. An Arduino microcontroller is used, similar to Tsai and Liang, however this time the model is the cheaper Uno. It's connected to an Arduino WiFi shield, which is in charge of data transfer.

The decision-making mechanism in Ruengittinun et al. control system does not incorporate any type of hysteresis, which could result in many motor on-off cycles. Furthermore, the proposed dashboard does not automatically monitor the growing process by taking into consideration the plant kind. For example, the user must enter a setup that they may not be familiar with, necessitating some agricultural expertise. Key attributes to the farm's success, such as power consumption, whether the autonomous mode can grow plants or veggies, and whether it is scalable, are all lacking from the results.

Ismail et al. [16] used a traditional vertical farming approach. An Arduino Mega with an Ethernet shield, a DC motor-driven water supply, LEDs, and soil moisture and water level sensors were all

part of their IoT-enabled vertical farm. The system works by requiring the user to open a web browser, navigate to the dashboard's webpage, and view the farm's current status. From there, the user must provide a lot of information because it does not take any activities on its own and instead relies on the user to turn things on and off manually. The idea is that when there is less water, the user will be warned and must take action based on their judgment. This is inconvenient for the user because it takes time.

Kalantari et al. [17] conducted a comprehensive survey of vertical farming's prospects and problems. The research looks at a variety of vertical farms, the majority of which are large-scale. The characteristics of each farm are also described. LED lighting, sensors, efficient watering, and automation are all common features of farms. The report discusses the different environmental, economic, and social effects of vertical farming, the majority of which indicate that this area has a bright future. In line with our previously proposed design, the goal of our proposed solution is to

be able to deliver high-end features in a practical and economical packaging by Liwal et al. [18].

Belista et al. [19] have presented a scalable vertical farm that uses aeroponics and promises a household-oriented product. A water level sensor, a pH level sensor, and temperature and humidity sensors all monitor the farm. The data from the sensors is saved on network-attached storage (NAS) device. The Raspberry Pi Zero W is the microcontroller that handles the storage and monitoring. There is no analog-to-digital converter included with this device, and no alternative is provided in the article.

Furthermore, while the framework supplied appears to be excellent for scalability, there are no explanations on how this might be accomplished. LEDs have been adjusted for individual plants, delivering the proper light spectrum for growth; however, there is no information on how this will be implemented. Because no implementation appears to have taken place, no information on power efficiency or cost-effectiveness is available. Overall, the

design appears to have all advanced features, but it is still incomplete, raising the question of its viability.



Fig.5. Vertical Farm in Singapore.



CHAPTER 3

Proposed System

Chapter 3: Proposed System

This Project is a smart solution over IoT using vertical hydroponics (a growing system based on water, not soil), growers can farm a variety of crops indoors and with minimal square footage. The vertical farm is completely climate controlled, uses LED lights to promote rapid growth, and connects to the app with Wi-Fi, making it easy for farmers to manage their crops remotely. Vertical hydroponics lends itself very well to leafy green vegetables [20]. We are growing successfully lettuces (romaine, boston bib, spring mix), cabbages (red and green), herbs (basil, cilantro, mint, dill, chives), spinach, kale (dwarf curly variety), broccoli, and petunias.

3.1. Project Aims

This project aims to: (i) spot the light on vertical farming and eco-friendly agriculture; (ii) introduce a new eco-friendly system for solving the problem of resources limitation. (iii) increase the

production of crops for achieving worldwide food demand; (iv) preserve the soil where the use of chemical fertilizer; (v) decrease resources consumption by recycling the water, required for growing the plants; (vi) bioremediation and reduce pollutions.

3.2. Methodology

An IoT-based system is employed for the suggested solution, with a focus on scalability and sustainability. An overall outline of the proposed vertical farm is shown in Figure 4.

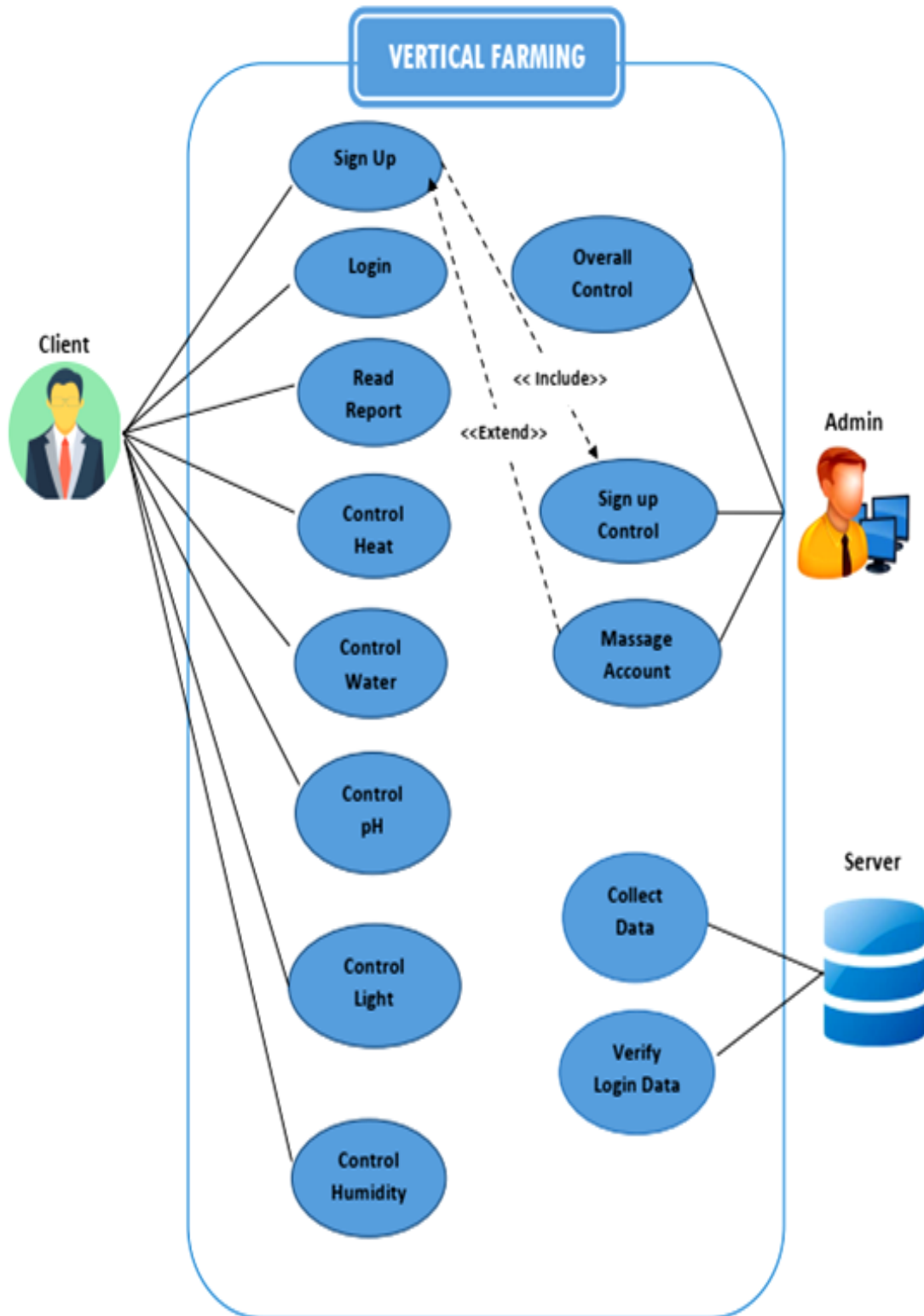


Fig.6. System Use Case Diagram.

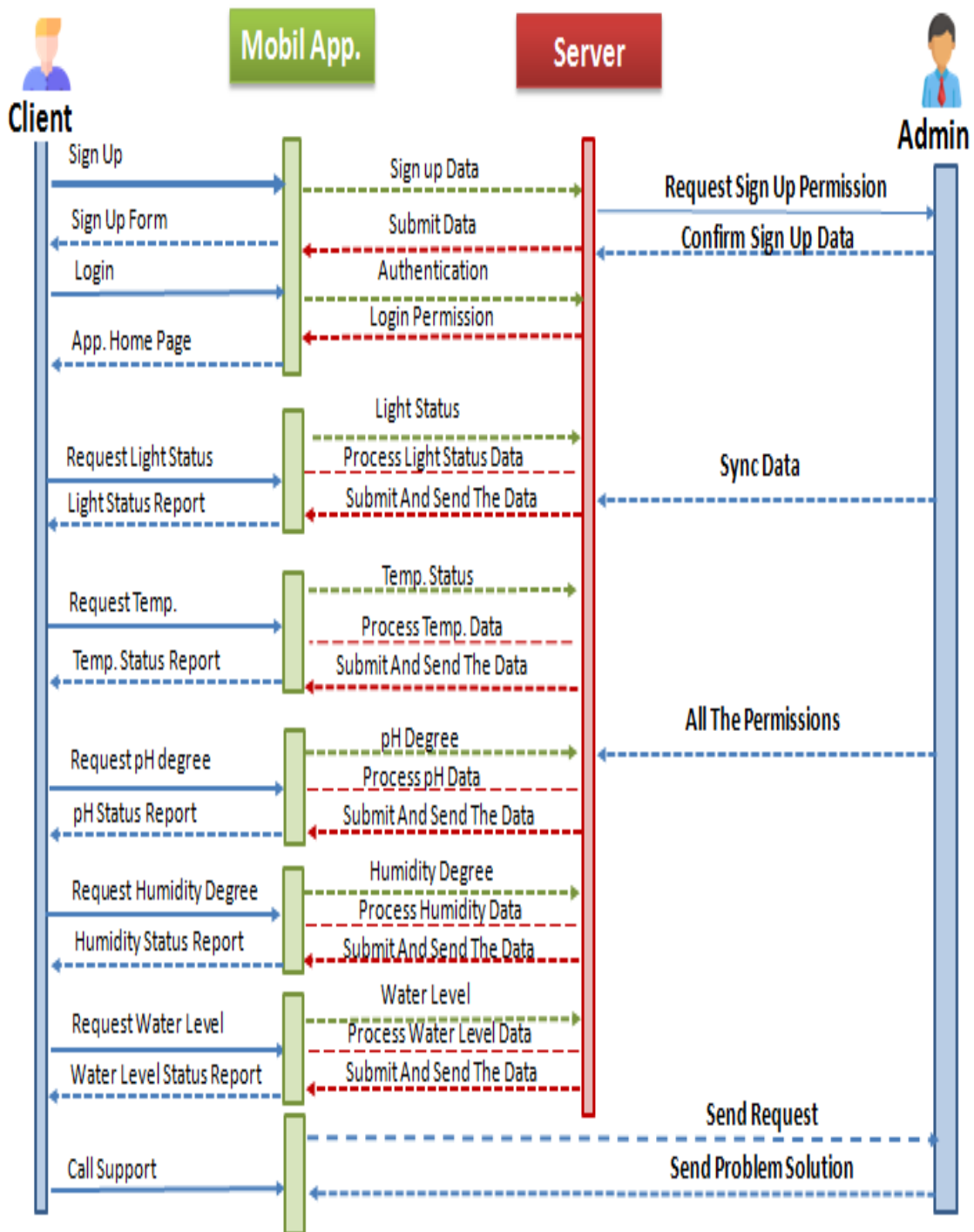


Fig.7. System Sequence Diagram.

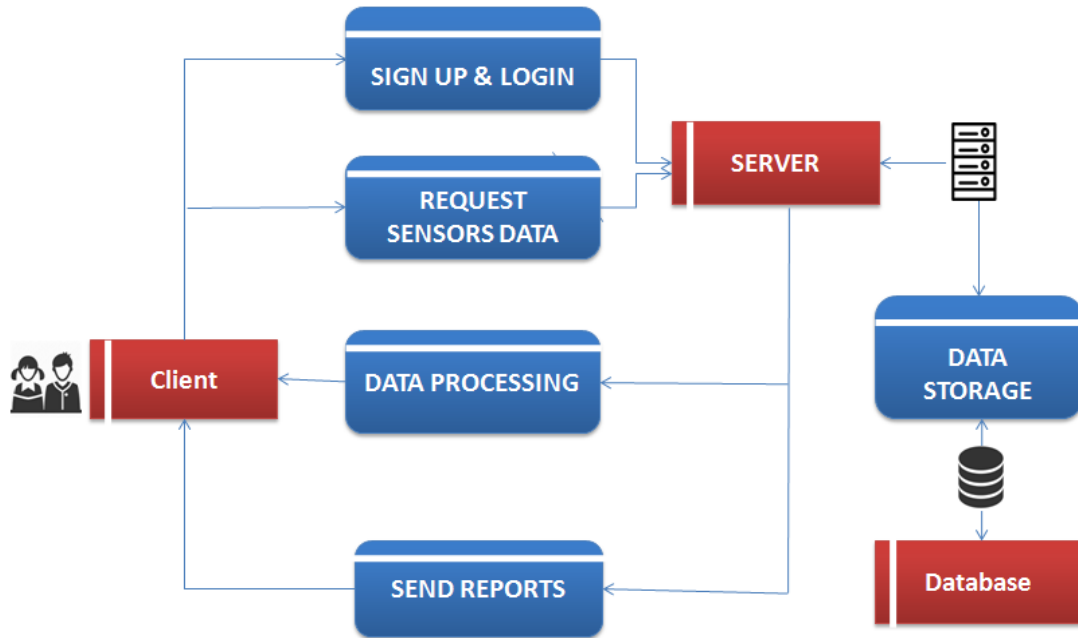


Fig.8. System Data-Flow Diagram.

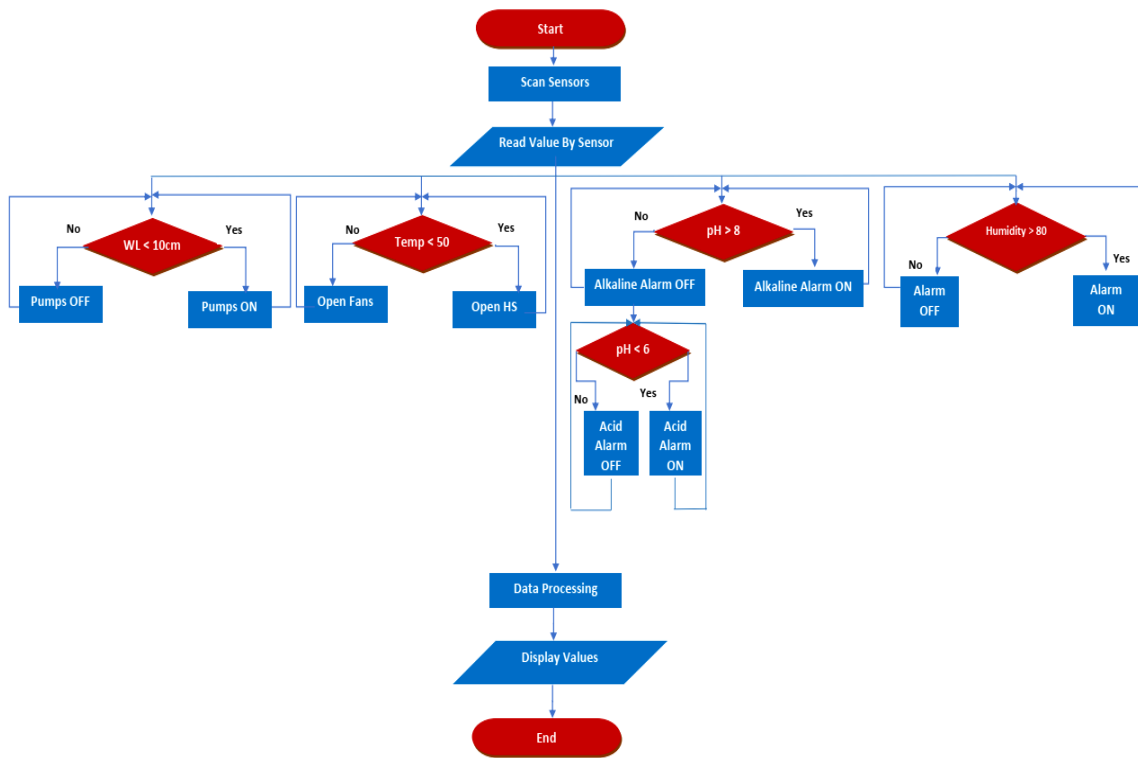


Fig.9. System Flowchart.

The IoT server acts as the master node for the entire operation, controlling N racks or slave nodes at a single location. Each slave node has its sensor, light, water supply, and fertilizer supply motors, all of which are interconnected to better care for the plant in that rack. A rack will now be referred to as the slave node, while the IoT server will be referred to as the master node. The water supply is loaded with the appropriate nutrients to assist the plants in obtaining the nutrition they require to support their growth. The nutrient levels are carefully analyzed and managed to help the plants grow to their maximum potential. Water is re-circulated, allowing for high efficiency, typically exceeding 90% on water consumption. Our vertical design allows us to pack more plants into a smaller space. We're producing too many plants in a space that's smaller than 5x 5.

The IoT server performs a variety of functions and is the primary point of access to the entire system. It provides a local web-based interface for controlling each node individually. It also needs an internet connection for security upgrades and, if necessary, remote

access. The master node also provides a Wi-Fi hotspot, allowing any slave node to connect to the master node and immediately get an IP address through the Dynamic Host Configuration Protocol (DHCP) [21] for easy integration of additional nodes. By delivering the required commands via HyperText Transfer Protocol, the master node arranges fixed water and fertilizer supply for each node (HTTP) [22].

Second, the nodes themselves are made up of hardware that connects to a variety of sensors, such as temperature, humidity, and an analog soil moisture sensor. The modules utilized in a single slave node are shown in Table 2. Raspberry pi is a low-cost, low-power Wi-Fi microchip with full TCP/IP stack and microcontroller capabilities [23].

Table 3. Project Components and Cost.

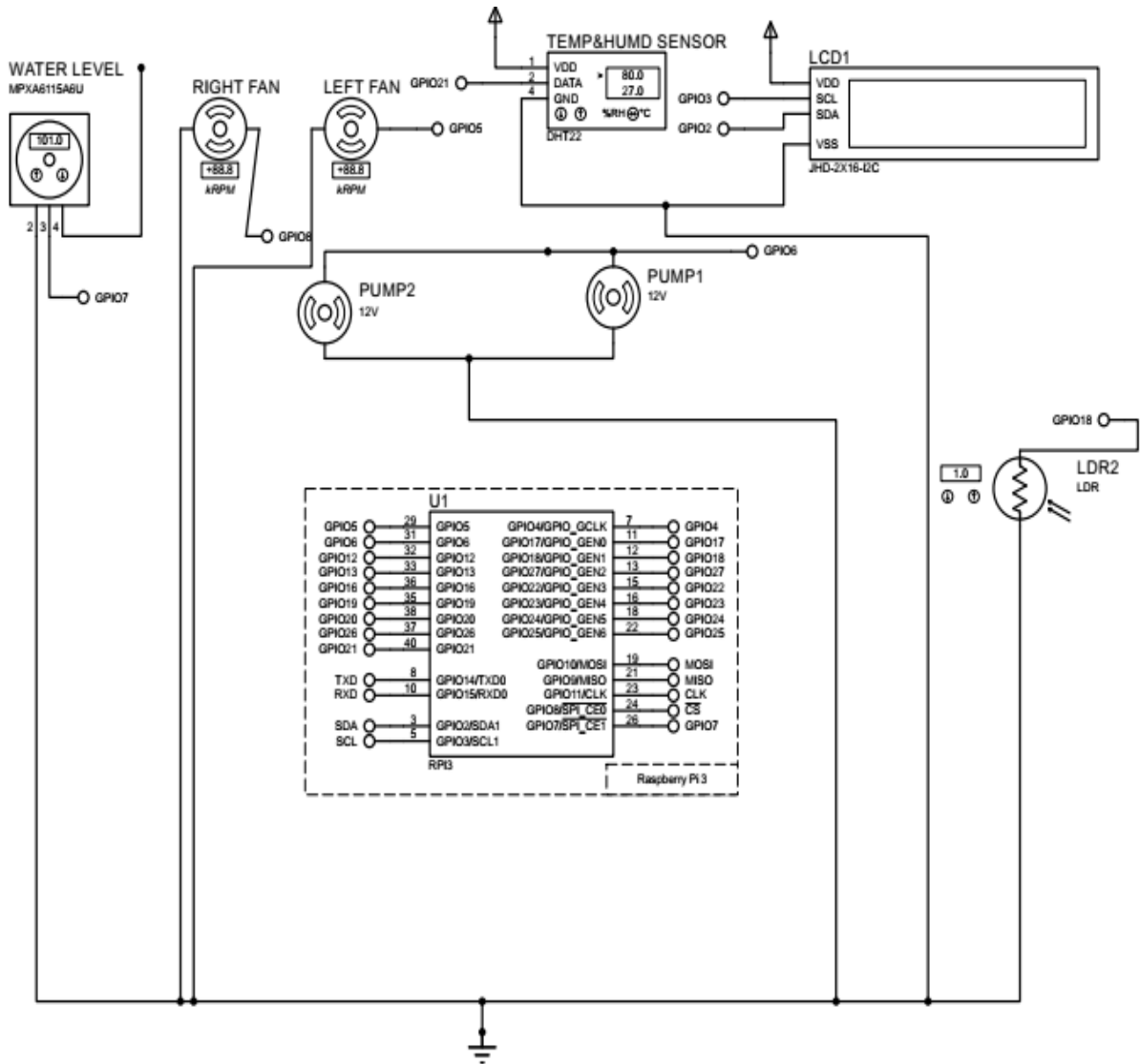
Index	Components	Price in L.E.	URL
1	Raspberry Pi 4B 4GB	2000	https://ram-e-shop.com/product/raspberry-pi-4-4gb-made-in-uk/
2	Enclosure For Raspberry Pi 4B	175	https://ram-e-shop.com/product/enclosure-for-raspberry-pi-4b-9-layers-acrylic-cooling-fan/
3	Raspberry Pi 4B official 15.3W USB-C Power Supply	350	https://ram-e-shop.com/product/raspberry-pi-4-official-15-3w-usb-c-power-supply-5v-3a/
4	Heat Sink Set for Raspberry Pi 4B Board	35	https://ram-e-shop.com/product/heat-sink-set-for-raspberry-pi-4-board-gold-aluminum-copper/
5	MicroSD Card - 128GB	400	https://www.amazon.de/-/en/PNY-PRO-Elite-MicroSD-Card/dp/B091JDB3FV/ref=sr_1_305?crid=1QG5A94A4JRW0&dchild=1&keywords=memory+128gb&qid=1635106621&prefix=memory+128gb%20Caps%20C403&sr=8-305
6	7" LCD Capacitive Touch Screen HDMI For Raspberry Pi	1350	https://ram-e-shop.com/product/lcd-hdmi-7-hi-1024/
7	7 Inch LCD Acrylic Screen Case Holder	200	https://ram-e-shop.com/product/7-inch-lcd-acrylic-screen-case-holder-bracket-for-raspberry-pi-3/

	Bracket For Raspberry Pi 4		
8	Raspberry Pi Camera Module	850	https://ram-e-shop.com/product/raspberry-pi-camera-module-v2-official-8-megapixel-hd/
9	50CM Raspberry Pi Camera Cable	35	https://ram-e-shop.com/product/50cm-raspberry-pi-camera-cable-ribbon-ffc-15pin-0-5mm-pitch/
10	Analog pH Sensor	850	https://ram-e-shop.com/product/kit-ph-meter/
11	Water Level Sensor	175	https://ram-e-shop.com/product/xkc-y25-npn-intelligent-non-contact-liquid-level-sensor-water-level-sensor/
12	Water Flow Sensor FS300A G 3/4"	175	https://ram-e-shop.com/product/sen-fs300a-water-flow/
13	2* Liquid Pump – 2000GPH (12vdc)	1100	https://ram-e-shop.com/product/liquid-pump-2000gph-12vdc/
14	6* Fan	360	https://ram-e-shop.com/product/220vac-fan-size-12x12x3-8-cm2/
15	Hair dryer max 709 - 5000 watt	216	https://www.amazon.de/-/en/Hair-dryer-max-709-black/dp/B09988T9PR/ref=pd_sbs_1/260-5789939-5372936?pd_rd_w=7KnSs&pf_rd_p=35010005-9e6c-41f1-bac1-9982e8d385be&pf_rd_r=MQGV81TKNFT4G322QZN9&pd_rd_r=582d328c-6dc3-4879-9470-a83aad5c6433&pd_rd_wg=KIw5x&pd_rd_i=B09988T9PR&psc=1

16	WS2812B 5050 RGB LED Strip	900	https://www.amazon.de/-/en/WS2812B-Strip-Individual-Addressable-waterproof/dp/B09FFHDT7Z/ref=sr_1_68?dchild=1&keywords=led+strip&qid=1635106332&sr=8-68
17	Color Sensor	195	https://ram-e-shop.com/product/color-sensor-tcs3200-with-focusable-lens/
18	SMPS	250	https://ram-e-shop.com/product/kit-smps-12v-8-5a/
19	Body	4500	N.A.
20	ML & PVC Tubes	950	N.A.
21	Reflective Thermal Felt	261	https://greenhouse-egypt.com/product/thermo/
22	Hydro Power	55	https://greenhouse-egypt.com/product/a-d9%87%d9%8a%d8%af%d8%b1%d9%88-d8%a8%d8%a7%d9%88%d8%b1-1-d9%84%d9%8a%d8%aa%d8%b1/
23	5cm Hydroponic Cups	95	https://greenhouse-egypt.com/product/net5100/
Total Cost		15477	

3.3. System Circuits Design

by this circuit We can control the machine through a temperature sensor, water level, humidity..etc





CHAPTER 4

System Implementation

Chapter 4: System Implementation

A Raspberry Pi is used to keep the system running. It also controls the watering cycles, pH levels, fertilizer levels, lighting cycles, and ventilation fans. All system parameters must be continuously monitored and updated. The Raspberry Pi is also used to keep track of all of the system's operational data and make it accessible through a series of web services. Figure 11 summarizes the proposed model scenario.

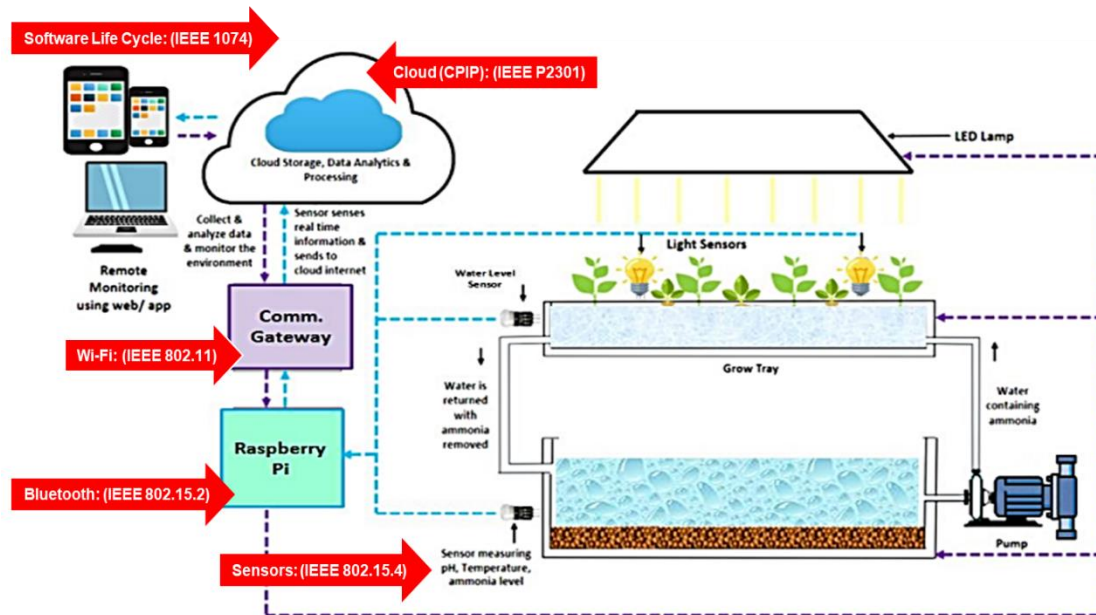


Fig.11. Proposed Model Scenario.

4.1 Mobile Application

4.1.1. Phase-1: First page

The start page is the first page that the user sees when opening the program, from which he can register and enter directly or go to the new registration.

4.1.2 Phase-2: Signup Layout

Registration page the user adds his own data such as a name, password, e-mail, and phone number so that the program registers this customer data in the established database.

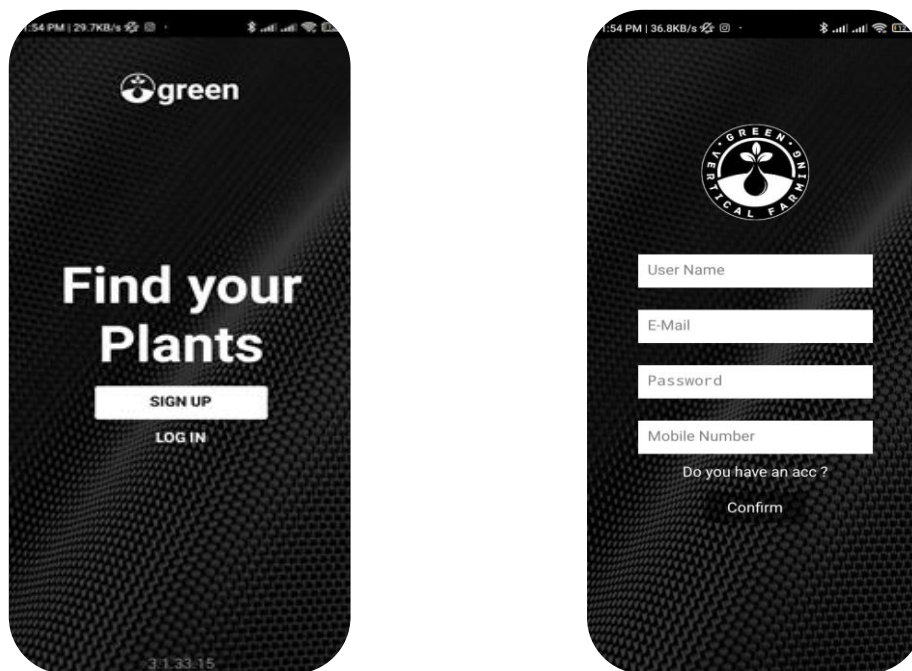


Fig.12. First page & Signup Layouts.

4.1.3 Phase-3: Login Layout

The login page and here the customer registers his data in the event of the first registration process and is registered on the database of the site. If an error occurs in the password, it is transferred to the password verification page.

4.1.4 Phase-4: Check Password Layout

Password confirmation page if there is any error in your registration, you can write your email to confirm your password in several ways.

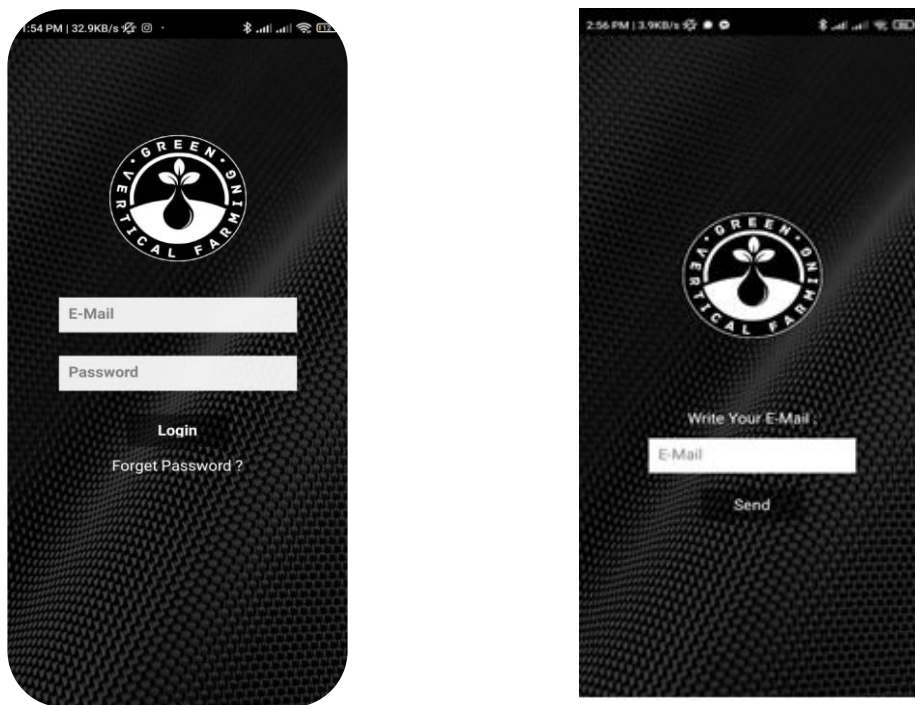


Fig.13. login & check password Layouts.

4.1.5 Phase-5: Home Page Control Page

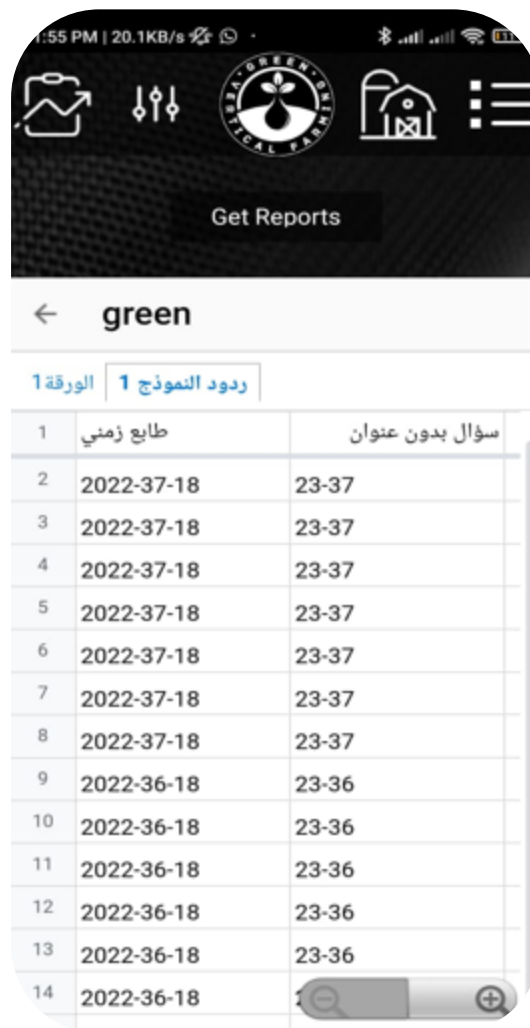
The control page here, the user can control all the details of the device from all sides, and there are also sensors that display data on temperature, humidity, and the degree of water rise moment by moment (lighting, water pressure, ventilation, knowing the temperature, ventilation, turning on and off the fans, turning on and off the lights).



Fig.14. Home Page (Control) Layout.

4.1.6 Phase-6: Reports Layout

The reports page displays data to provide the user with all developments such as temperature, humidity and water temperature. Also, it contains back and logout buttons.



The screenshot shows a mobile application interface. At the top, there is a navigation bar with several icons: a folder, a vertical slider, a circular logo with a plant, a house, and a menu icon. Below the navigation bar is a button labeled 'Get Reports'. Below that is a header with a back arrow and the text 'green'. Underneath the header, there are two tabs: 'الورقة 1' and 'ردود النموذج 1'. The main content is a table with 14 rows. The first row has a header 'سؤال بدون عنوان' and 'طابع زمني'. The subsequent rows contain dates and question numbers.

ردود النموذج 1 الورقة 1	سؤال بدون عنوان
1	طابع زمني
2	2022-37-18 23-37
3	2022-37-18 23-37
4	2022-37-18 23-37
5	2022-37-18 23-37
6	2022-37-18 23-37
7	2022-37-18 23-37
8	2022-37-18 23-37
9	2022-36-18 23-36
10	2022-36-18 23-36
11	2022-36-18 23-36
12	2022-36-18 23-36
13	2022-36-18 23-36
14	2022-36-18

Fig.15. Reports Layouts.

4.1.7 Phase-7: Green page Layout

Green page talks about the entire project and the members of the team implementing the idea.

4.1.8 Phase-8: Help Layout

The help page shows the user the proportions and data that the user needs for each plant.

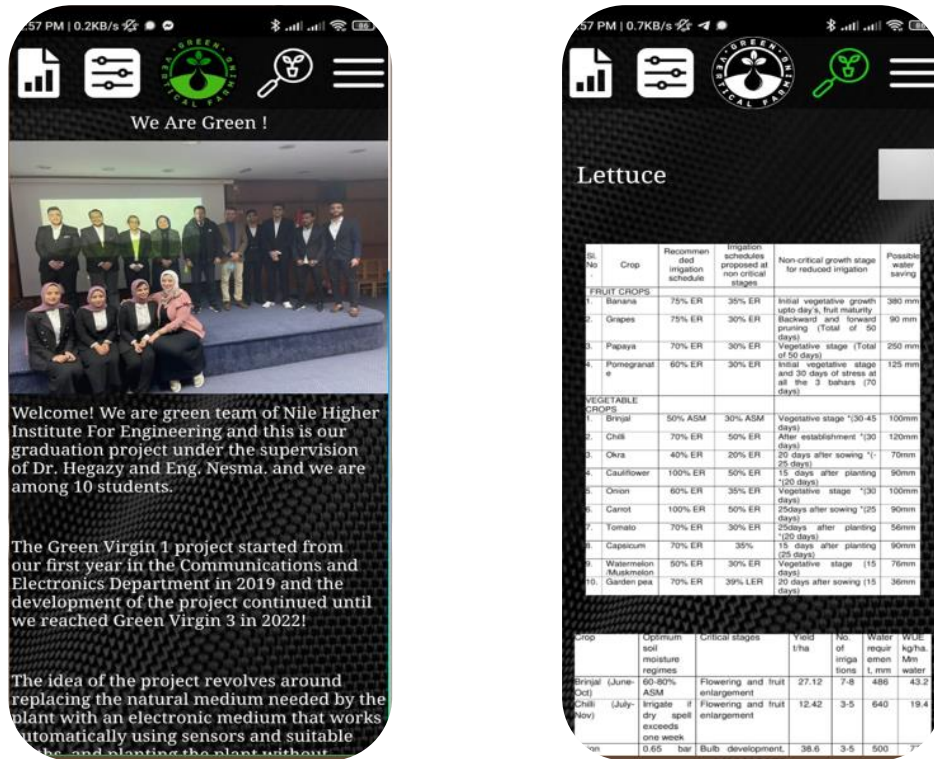


Fig.16. Green page & help Layouts.

4.1.9 Phase-9: The article of how-to vertical farming Layout

The detailed article page of how to vertical farming in boring detail that helps the user to work with the device.

4.1.10 Phase-10: List Layout

The menu page enables the user to navigate within the program pages (e-mail, control, reports, contact us, introduce us, log out).



Fig.17. The article & list Layouts.

4.1.11 Phase-11: Email page Layout

The e-mail page displays the user's data and changes his password.

4.1.12 Phase-12: Contact Us Layout

The contact us page through which the user can write down what he wants from problems or modifications, and his message will be sent to the program control officials to solve his problem, and there are the buttons for sending it to the site.

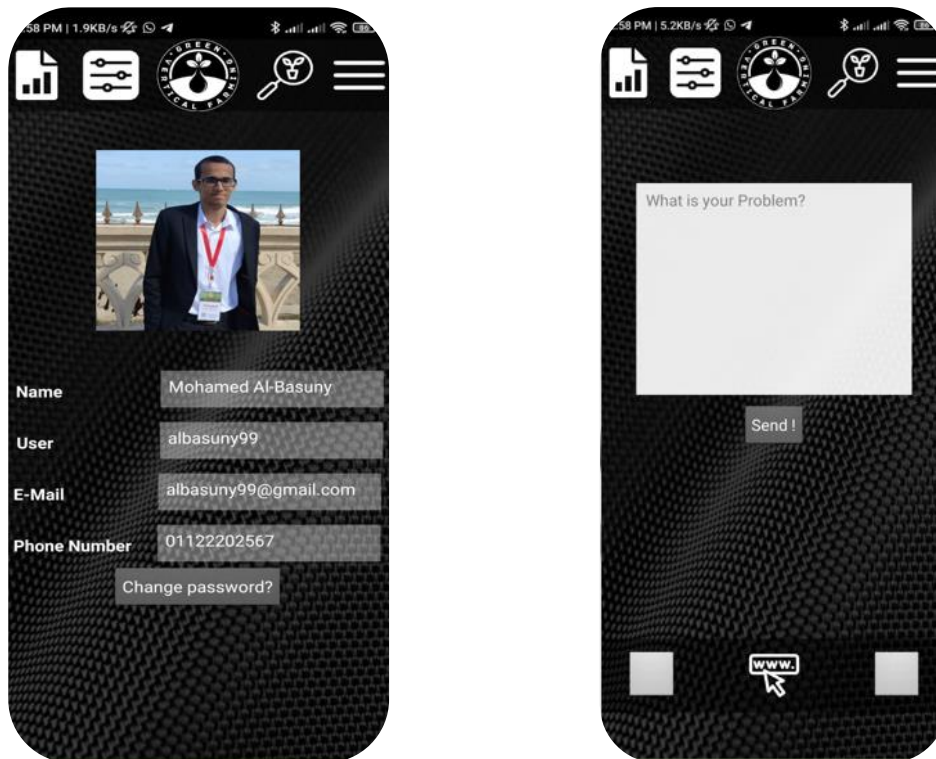


Fig.18. Email Page & contact us Layouts.

4.2 Server

A database is an organized collection of structured information, or data, typically stored electronically in a computer system. A database is usually controlled by a database management system (DBMS). Together, the data and the DBMS, along with the applications that are associated with them, are referred to as a database system, often shortened to just database.

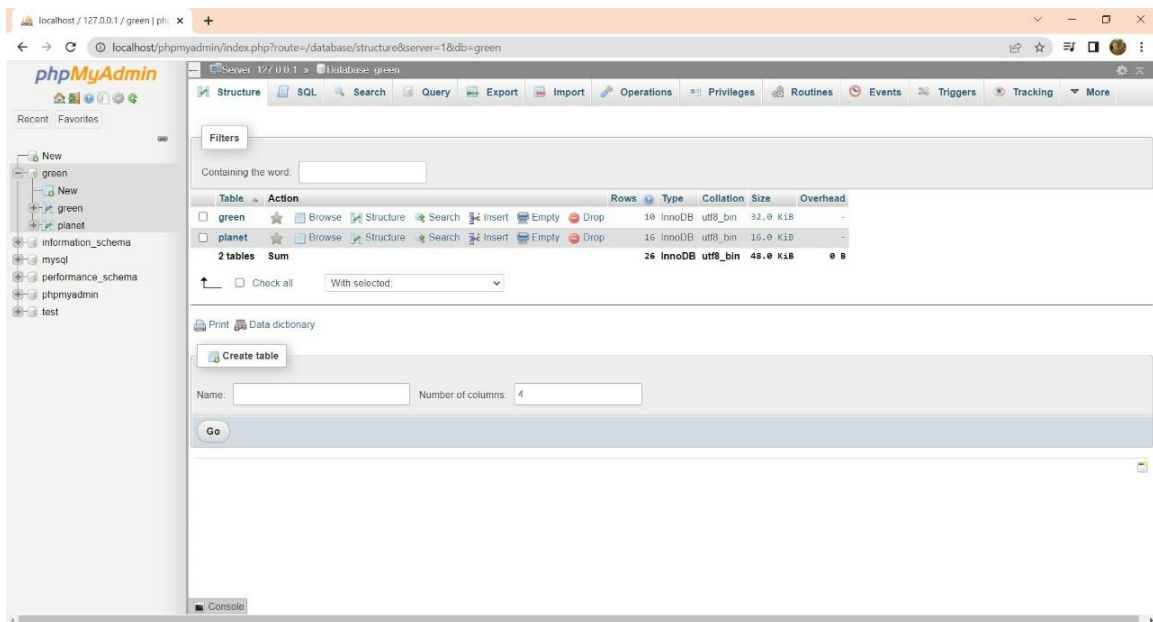


Fig.19. Server Layouts.

Data within the most common types of databases in operation today is typically modeled in rows and columns in a series of tables to make processing and data querying efficient. The data can then be easily accessed, managed, modified, updated, controlled, and organized. Most databases use structured query language (SQL) for writing and querying data.

4.2.1 Phase-1: Server CODE



The figure consists of two side-by-side screenshots of a code editor. The left screenshot shows the source code for a Node.js application. The code uses the `fs` module to create a stream for writing data to a file named `data.csv`. It defines a `data` stream and a `bigdata` stream. The `data` stream is used to write temperature data, and the `bigdata` stream is used to write humidity data. The code also includes a `listen` function that logs the server's listening address. The right screenshot shows the output of the application. It displays the server's listening address and the data being written to the `data.csv` file. The data is organized into rows and columns, with each row representing a data point and each column representing a different attribute (Date, Time, Temperature, Humidity).

```
app.js - node js - Visual Studio Code (Administrator)
app.js > app.listen() callback > on('end') callback
44
45
46 fs.createReadStream('data.csv')
47 .pipe(csv())
48 .on('data', (row) => {
49   temperature.push(row['Temperature']);
50   humidity.push(row['Humidity']);
51   bigdata.push(row);
52 })
53 }
54 .on('end', () => {
55   console.log(temperature);
56   console.log(humidity);
57   console.log(bigdata);
58 });
59 console.log("Example app listening at http://localhost:${port}")
60 });
61
62
```

```
19 app.get('/temperature', (req, res) => {
20   res.send(temperature[temperature.length - 1]);
21 })
22
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
{ Date: '1', Time: '5', Temperature: '9', Humidity: '3' },
{ Date: '2', Time: '6', Temperature: '10', Humidity: '14' },
{ Date: '3', Time: '7', Temperature: '11', Humidity: '15' },
{ Date: '4', Time: '8', Temperature: '12', Humidity: '16' }
]
PS C:\Users\Lenovo\Desktop\node js\node js> node app.js
Example app listening at http://localhost:3100
[ '9', '10', '11', '12' ]
[ '3', '14', '15', '16' ]
[
  { Date: '1', Time: '5', Temperature: '9', Humidity: '3' },
  { Date: '2', Time: '6', Temperature: '10', Humidity: '14' },
  { Date: '3', Time: '7', Temperature: '11', Humidity: '15' },
  { Date: '4', Time: '8', Temperature: '12', Humidity: '16' }
]
```

Fig.20. Server CODE Layouts.

4.2.2 Phase-2: Sign Up

❖ User Name

A username is a name that uniquely identifies someone on a computer system. For example, a computer may be setup with multiple accounts, with different usernames for each account. Many websites allow users to choose a username so that they can customize their settings or set up an online account. For example, your bank may allow you to choose a username for accessing your banking information. You may need to choose a username in order to post messages to a certain message board on the Web. E-mail services, such as Hotmail require users to choose a username in order to use the service.

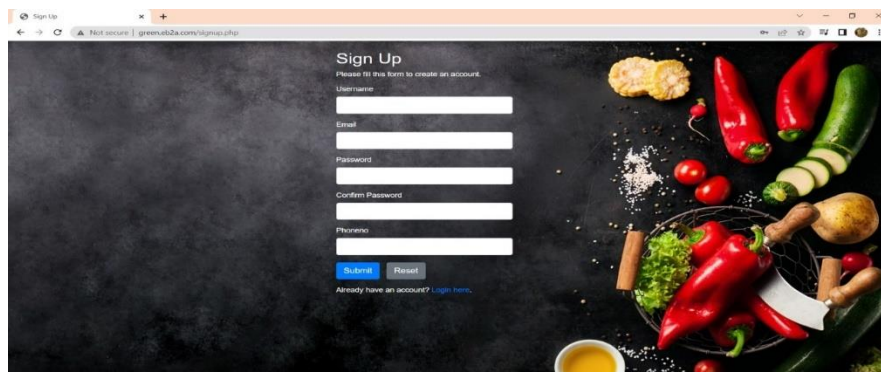


Fig.21. Sign up Layouts.

❖ Email

A method of exchanging messages instantly from one system to another with the help of the internet is called an Email. Initially, Email usage was limited to users of the same computer, and it asked for the users to be online to receive the messages. Time changed, and now we know how the mailbox looks. The mail can be sent to more than one recipient, and the recipient's name can be hidden from others by adding their names in Bcc.

The process starts with an email client's help by connecting it through a server called Simple Mail Transfer Protocol through the internet. A dedicated port is assigned to the server to help the client transfer the messages through the mail. It is necessary to keep the header information intact so that the recipient's email address should be correct. SMTP converts the information to transfer the mail content across the ports. The @ sign acts as a divider between the name and mail server, and hence SMTP looks for the mail server after @ sign.

There are different clients for emails such as Outlook, Gmail, thunderbird etc. and mails can be sent and received from different clients. When the domains are different, SMTP looks for the domains in Domain Name System, and with the help of an IP address, a signal is sent to the recipient's server saying about the email. Now the gateways are opened, and the mail is sent to the new server, and the communication happens.

❖ **password**

A password is a string of characters used to verify the identity of a user during the authentication process. Passwords are typically used in tandem with a username; they are designed to be known only to the user and allow that user to gain access to a device, application or website. Passwords can vary in length and can contain letters, numbers and special characters.

A password is sometimes called a passphrase, when the password uses more than one word, or a passcode or passkey, when the

password uses only numbers, such as a personal identification number (PIN).

❖ **Mobile Phone Number**

Accounts are accounts created using a mobile phone number instead of an email address. After creating your account, you must use your mobile phone number to sign in on all platform types. If you'd like the option of signing in with an email address, you can add an email address by going to Your Account and selecting Login & security.

If you don't have an email address linked to your account, you'll receive notifications and emails through the Message Center.

❖ **Login Page**

The login page, from it you enter your password and your user name, or if you do not want to enter the sign-up page

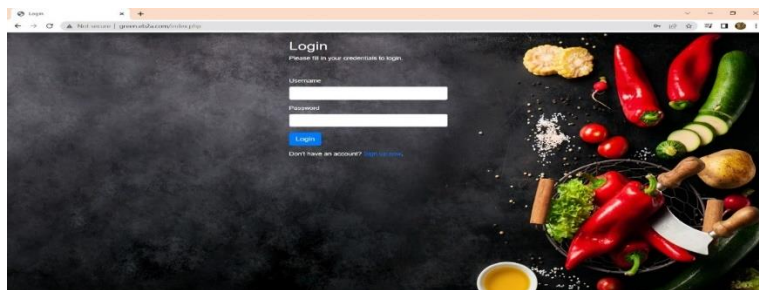


Fig.22. login page Layouts.

❖ Home Page

Now you are in the home screen, there are buttons in the north and above them you can try the password or sign out or open the report page on the right is the Planet Data and Google page.

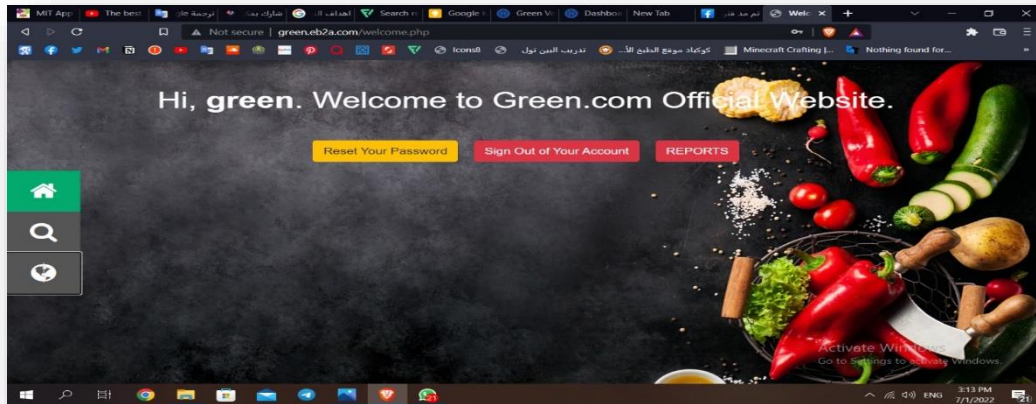


Fig.23. Home Page Layouts.

❖ Reset Password

You can change the password and it will be spoken in the server and you will receive confirmation.

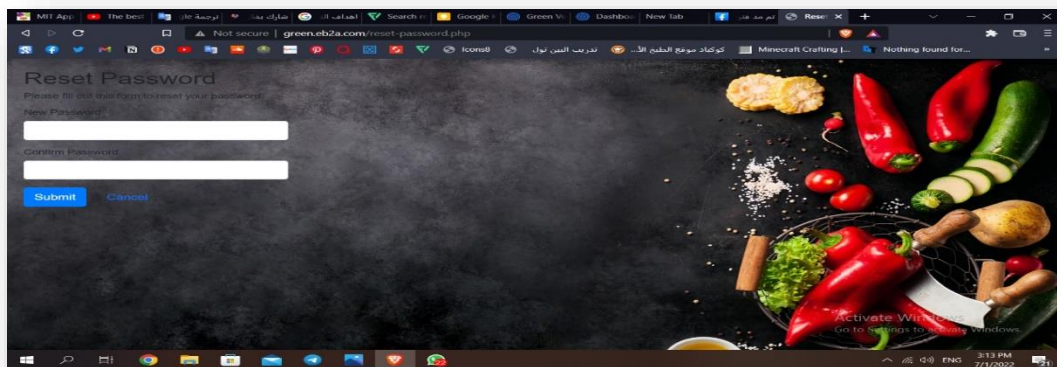


Fig.24. Reset Password Layouts.

❖ Google Button

The Google Button is your bio on Google, so that if you are in the site and would like to search for more data about plants.

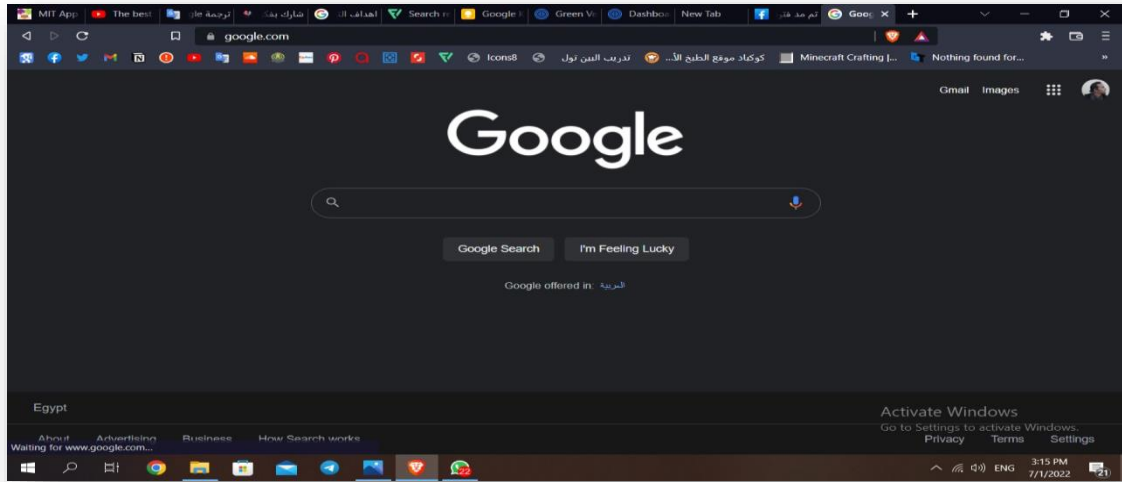


Fig.25. Google Button Layouts.

❖ First page

This page is displayed when the user sign in



Fig.26. MySQL First page Layouts.

MySQL, the most popular Open Source SQL database management system, is developed, distributed, and supported by Oracle Corporation.

MySQL is a database management system.

A database is a structured collection of data. It may be anything from a simple shopping list to a picture gallery or the vast amounts of information in a corporate network. To add, access, and process data stored in a computer database, you need a database management system such as MySQL Server. Since computers are very good at handling large amounts of data, database management systems play a central role in computing, as standalone utilities, or as parts of other applications.

MySQL databases are relational.

A relational database stores data in separate tables rather than putting all the data in one big storeroom. The database structures are organized into physical files optimized for speed. The logical model, with objects such as databases, tables, views, rows, and columns,

offers a flexible programming environment. You set up rules governing the relationships between different data fields, such as one-to-one, one-to-many, unique, required or optional, and “pointers” between different tables. The database enforces these rules, so that with a well-designed database, your application never sees inconsistent, duplicate, orphan, out-of-date, or missing data.

The SQL part of “MySQL” stands for “Structured Query Language”. SQL is the most common standardized language used to access databases. Depending on your programming environment, you might enter SQL directly (for example, to generate reports), embed SQL statements into code written in another language, or use a language-specific API that hides the SQL syntax.

SQL is defined by the ANSI/ISO SQL Standard. The SQL standard has been evolving since 1986 and several versions exist. In this manual, “SQL-92” refers to the standard released in 1992, “SQL:1999” refers to the standard released in 1999, and “SQL:2003” refers to the current version of the standard. We use the phrase “the SQL standard” to mean the current version of the SQL Standard at any time.

MySQL software is Open Source.

Open Source means that it is possible for anyone to use and modify the software. Anybody can download the MySQL software from the Internet and use it without paying anything. If you wish, you may study the source code and change it to suit your needs. The MySQL software uses the GPL (GNU General Public License), <http://www.fsf.org/licenses/>, to define what you may and may not do with the software in different situations. If you feel uncomfortable with the GPL or need to embed MySQL code into a commercial application, you can buy a commercially licensed version from us. See the MySQL Licensing Overview for more information (<http://www.mysql.com/company/legal/licensing/>).

The MySQL Database Server is very fast, reliable, scalable, and easy to use.

If that is what you are looking for, you should give it a try. MySQL Server can run comfortably on a desktop or laptop, alongside your

other applications, web servers, and so on, requiring little or no attention. If you dedicate an entire machine to MySQL, you can adjust the settings to take advantage of all the memory, CPU power, and I/O capacity available. MySQL can also scale up to clusters of machines, networked together.

MySQL Server was originally developed to handle large databases much faster than existing solutions and has been successfully used in highly demanding production environments for several years. Although under constant development, MySQL Server today offers a rich and useful set of functions. Its connectivity, speed, and security make MySQL Server highly suited for accessing databases on the Internet.

MySQL Server works in client/server or embedded systems.

The MySQL Database Software is a client/server system that consists of a multithreaded SQL server that supports different back ends, several different client programs and libraries, administrative tools, and a wide range of application programming interfaces (APIs).

We also provide MySQL Server as an embedded multithreaded library that you can link into your application to get a smaller, faster, easier-to-manage standalone product.

A large amount of contributed MySQL software is available.

MySQL Server has a practical set of features developed in close cooperation with our users. It is very likely that your favorite application or language supports the MySQL Database Server.

The official way to pronounce “MySQL” is “My Ess Que Ell” (not “my sequel”), but we do not mind if you pronounce it as “my sequel” or in some other localized way.

1.2.2 The Main Features of MySQL

This section describes some of the important characteristics of the MySQL Database Software. In most respects, the roadmap applies to all versions of MySQL. For information about features as they are introduced into MySQL on a series-specific basis, see the “In a Nutshell” section of the appropriate Manual:

MySQL 8.0: Section 1.3, “What Is New in MySQL 8.0”

MySQL 5.7: What Is New in MySQL 5.7

MySQL 5.6: What Is New in MySQL 5.6

Internals and Portability

Written in C and C++.

Tested with a broad range of different compilers.

Works on many different platforms.

See <https://www.mysql.com/support/supportedplatforms/database.html>.

For portability, configured using CMake.

Tested with Purify (a commercial memory leakage detector) as well as with Valgrind, a GPL tool (<http://developer.kde.org/~sewardj/>).

Uses multi-layered server design with independent modules.

Designed to be fully multithreaded using kernel threads, to easily use multiple CPUs if they are available. Provides transactional and no transactional storage engines. Uses very fast B-tree disk tables (MyISAM) with index compression.

Designed to make it relatively easy to add other storage engines. This is useful if you want to provide an SQL interface for an in-house database. Uses a very fast thread-based memory allocation system. Executes very fast joins using an optimized nested-loop join. Implements in-memory hash tables, which are used as temporary tables. Implements SQL functions using a highly optimized class library that should be as fast as possible. Usually there is no memory allocation at all after query initialization.

Provides the server as a separate program for use in a client/server networked environment.

Data Types

Many data types: signed/unsigned integers 1, 2, 3, 4, and 8 bytes long, FLOAT, DOUBLE, CHAR, VARCHAR, BINARY, VARBINARY, TEXT, BLOB, DATE, TIME, DATETIME, TIMESTAMP, YEAR, SET, ENUM, and OpenGIS spatial types. See Chapter 11, Data Types.

Fixed-length and variable-length string types.

Statements and Functions

Full operator and function support in the SELECT list and WHERE clause of queries. For example:

```
mysql> SELECT CONCAT(first_name, ' ', last_name)
```

```
-> FROM citizen
```

```
-> WHERE income/dependents > 10000 AND age > 30;
```

Full support for SQL GROUP BY and ORDER BY clauses. Support for

	group	functions			
(COUNT(),	AVG(),	STD(),	SUM(),	MAX(),	MIN(),
and GROUP_CONCAT()).					

Support for LEFT OUTER JOIN and RIGHT OUTER JOIN with both standard SQL and ODBC syntax.

Support for aliases on tables and columns as required by standard SQL.

Support for DELETE, INSERT, REPLACE, and UPDATE to return the number of rows that were changed (affected), or to return the number of rows matched instead by setting a flag when connecting to the server.

Support for MySQL-specific SHOW statements that retrieve information about databases, storage engines, tables, and indexes. Support for the INFORMATION_SCHEMA database, implemented according to standard SQL.

An EXPLAIN statement to show how the optimizer resolves a query.

Independence of function names from table or column names. For example, ABS is a valid column name. The only restriction is that for a function call, no spaces are permitted between the function name and the "(" that follows it. See Section 9.3, "Keywords and Reserved Words".

You can refer to tables from different databases in the same statement.

Security

A privilege and password system that is very flexible and secure, and that enables host-based verification.

Password security by encryption of all password traffic when you connect to a server.

Scalability and Limits

Support for large databases. We use MySQL Server with databases that contain 50 million records. We also know of users who use MySQL Server with 200,000 tables and about 5,000,000,000 rows. Support for up to 64 indexes per table. Each index may consist of 1 to 16 columns or parts of columns. The maximum index width for InnoDB tables is either 767 bytes or 3072 bytes. See Section 15.22, “InnoDB Limits”. The maximum index width for MyISAM tables is 1000 bytes. See Section 16.2, “The MyISAM Storage Engine”. An index may use a prefix of a column for CHAR, VARCHAR, BLOB, or TEXT column types.

Connectivity

Clients can connect to MySQL Server using several protocols:

Clients can connect using TCP/IP sockets on any platform.

On Windows systems, clients can connect using named pipes if the server is started with the `named_pipe` system variable enabled. Windows servers also support shared-memory connections if started with the `shared_memory` system variable enabled. Clients can connect through shared memory by using the `--protocol=memory` option.

On Unix systems, clients can connect using Unix domain socket files.

MySQL client programs can be written in many languages. A client library written in C is available for clients written in C or C++, or for any language that provides C bindings.

APIs for C, C++, Eiffel, Java, Perl, PHP, Python, Ruby, and Tcl are available, enabling MySQL clients to be written in many languages. See Chapter 29, Connectors and APIs.

The Connector/ODBC (MyODBC) interface provides MySQL support for client programs that use ODBC (Open Database Connectivity) connections. For example, you can use MS Access to connect to your MySQL server. Clients can be run on Windows or Unix. Connector/ODBC source is available. All ODBC 2.5 functions are supported, as are many others. See MySQL Connector/ODBC Developer Guide.

The Connector/J interface provides MySQL support for Java client programs that use JDBC connections. Clients can be run on Windows or Unix. Connector/J source is available. See MySQL Connector/J 5.1 Developer Guide.

MySQL Connector/NET enables developers to easily create .NET applications that require secure, high-performance data connectivity with MySQL. It implements the required ADO.NET interfaces and integrates into ADO.NET aware tools. Developers can build applications using their choice of .NET languages. MySQL Connector/NET is a fully managed ADO.NET driver written in 100% pure C#. See MySQL Connector/NET Developer Guide.

Localization

The server can provide error messages to clients in many languages. See Section 10.12, “Setting the Error Message Language”.

Full support for several different character sets, including latin1 (cp1252), german, big5, ujis, several Unicode character sets, and more. For example, the Scandinavian characters “å”, “ä” and “ö” are permitted in table and column names. All data is saved in the chosen character set.

Sorting and comparisons are done according to the default character set and collation. It is possible to change this when the MySQL server is started (see Section 10.3.2, “Server Character Set and Collation”). To see an example of very advanced sorting, look at the Czech sorting code. MySQL Server supports many different character sets that can be specified at compile time and runtime.

The server time zone can be changed dynamically, and individual clients can specify their own time zone. See Section 5.1.15, “MySQL Server Time Zone Support”.

Clients and Tools

MySQL includes several client and utility programs. These include both command-line programs such as mysqldump and mysqladmin, and graphical programs such as MySQL Workbench.

MySQL Server has built-in support for SQL statements to check, optimize, and repair tables. These statements are available from the command line through the mysqlcheck client. MySQL also

includes `myisamchk`, a very fast command-line utility for performing these operations on MyISAM tables. See Chapter 4, MySQL Programs.

MySQL programs can be invoked with the `--help` or `-?` option to obtain online assistance.

1.2.3 History of MySQL

We started out with the intention of using the mSQL database system to connect to our tables using our own fast low-level (ISAM) routines. However, after some testing, we came to the conclusion that mSQL was not fast enough or flexible enough for our needs. This resulted in a new SQL interface to our database but with almost the same API interface as mSQL. This API was designed to enable third-party code that was written for use with mSQL to be ported easily for use with MySQL.

MySQL is named after co-founder Monty Widenius's daughter, My. The name of the MySQL Dolphin (our logo) is “Sakila,” which was chosen from a huge list of names suggested by users in our “Name the Dolphin” contest. The winning name was submitted by Ambrose Twebaze, an Open Source software developer from Eswatini (formerly Swaziland), Africa. According to Ambrose, the feminine name Sakila has its roots in SiSwati, the local language of Eswatini. Sakila is also the name of a town in Arusha, Tanzania, near Ambrose's country of origin, Uganda.

1.3 What Is New in MySQL 8.0

This section summarizes what has been added to, deprecated in, and removed from MySQL 8.0. A companion section lists MySQL server options and variables that have been added, deprecated, or removed in MySQL 8.0; see Section 1.4, “Server and Status Variables and Options Added, Deprecated, or Removed in MySQL 8.0”.

Features Added in MySQL 8.0

Features Deprecated in MySQL 8.0

Features Removed in MySQL 8.0

Features Added in MySQL 8.0

The following features have been added to MySQL 8.0:

Data dictionary. MySQL now incorporates a transactional data dictionary that stores information about database objects. In previous MySQL releases, dictionary data was stored in metadata files and nontransactional tables. For more information, see Chapter 14, MySQL Data Dictionary.

Atomic data definition statements (Atomic DDL). An atomic DDL statement combines the data dictionary updates, storage engine operations, and binary log writes associated with a DDL operation into a single, atomic transaction. For more information, see Section 13.1.1, “Atomic Data Definition Statement Support”.

Upgrade procedure. Previously, after installation of a new version of MySQL, the MySQL server automatically upgrades the data dictionary tables at the next startup, after which the DBA is expected to invoke `mysql_upgrade` manually to upgrade the system tables in the `mysql` schema, as well as objects in other schemas such as the `sys` schema and user schemas.

As of MySQL 8.0.16, the server performs the tasks previously handled by `mysql_upgrade`. After installation of a new MySQL version, the server now automatically performs all necessary upgrade tasks at the next startup and is not dependent on the DBA invoking `mysql_upgrade`. In addition, the server updates the contents of the help tables (something `mysql_upgrade` did not do). A new `--upgrade` server option provides control over how the server performs automatic data dictionary and server upgrade operations. For more information, see Section 2.11.3, “What the MySQL Upgrade Process Upgrades”.

Session Reuse. MySQL Server now supports SSL session reuse by default with a timeout setting to control how long the server maintains a session cache that establishes the period during which a client is permitted to request session reuse for new connections. All MySQL client programs support session reuse. For server-side and client-side configuration information, see Section 6.3.5, “Reusing SSL Sessions”.

In addition, C applications now can use the C API capabilities to enable session reuse for encrypted connections (see [SSL Session Reuse](#)).

Security and account management. These enhancements were added to improve security and enable greater DBA flexibility in account management:

The grant tables in the `mysql` system database are now InnoDB (transactional) tables. Previously, these were MyISAM (nontransactional) tables. The change of grant table storage engine underlies an accompanying change to the behavior of account-management statements. Previously, an account-management statement (such as `CREATE USER` or `DROP USER`) that named multiple users could succeed for some users and fail for others. Now, each statement is transactional and either succeeds for all named users or rolls back and has no effect if any error occurs. The statement is written to the binary log if it succeeds, but not if it fails; in that case, rollback occurs and no changes are made. For more information, see [Section 13.1.1](#), “Atomic Data Definition Statement Support”.

A new `caching_sha2_password` authentication plugin is available. Like

the `sha256_password` plugin, `caching_sha2_password` implements SHA-256 password hashing, but uses caching to address latency issues at connect time. It also supports more transport protocols and does not require linking against OpenSSL for RSA key pair-based password-exchange capabilities. See [Section 6.4.1.2](#), “Caching SHA-2 Pluggable Authentication”.

The `caching_sha2_password` and `sha256_password` authentication plugins provide more secure password encryption than the `mysql_native_password` plugin, and `caching_sha2_password` provides better performance than `sha256_password`. Due to these superior security and performance characteristics of `caching_sha2_password`, it is now the preferred authentication plugin, and is also the default authentication plugin rather than `mysql_native_password`. For

information about the implications of this change of default plugin for server operation and compatibility of the server with clients and connectors, see `caching_sha2_password` as the Preferred Authentication Plugin.

The MySQL Enterprise Edition SASL LDAP authentication plugin now supports GSSAPI/Kerberos as an authentication method for MySQL clients and servers on Linux. This is useful in Linux environments where applications access LDAP using Microsoft Active Directory, which has Kerberos enabled by default. See LDAP Authentication Methods.

MySQL Enterprise Edition now supports an authentication method that enables users to authenticate to MySQL Server using Kerberos, provided that appropriate Kerberos tickets are available or can be obtained. For details, see Section 6.4.1.8, “Kerberos Pluggable Authentication”.

MySQL now supports roles, which are named collections of privileges. Roles can be created and dropped. Roles can have privileges granted to and revoked from them. Roles can be granted to and revoked from user accounts. The active applicable roles for an account can be selected from among those granted to the account, and can be changed during sessions for that account. For more information, see Section 6.2.10, “Using Roles”.

MySQL now incorporates the concept of user account categories, with system and regular users distinguished according to whether they have the `SYSTEM_USER` privilege. See Section 6.2.11, “Account Categories”.

Previously, it was not possible to grant privileges that apply globally except for certain schemas. This is now possible if the `partial_revokes` system variable is enabled. See Section 6.2.12, “Privilege Restriction Using Partial Revokes”.

The `GRANT` statement has an `AS user [WITH ROLE]` clause that specifies additional information about the privilege context to use for statement execution. This syntax is visible at the SQL level, although its primary purpose is to enable uniform replication across all nodes of grantor privilege restrictions imposed by partial

revokes, by causing those restrictions to appear in the binary log. See Section 13.7.1.6, “GRANT Statement”.

MySQL now maintains information about password history, enabling restrictions on reuse of previous passwords. DBAs can require that new passwords not be selected from previous passwords for some number of password changes or period of time. It is possible to establish password-reuse policy globally as well as on a per-account basis.

It is now possible to require that attempts to change account passwords be verified by specifying the current password to be replaced. This enables DBAs to prevent users from changing password without proving that they know the current password. It is possible to establish password-verification policy globally as well as on a per-account basis.

Accounts are now permitted to have dual passwords, which enables phased password changes to be performed seamlessly in complex multiple-server systems, without downtime.

MySQL now enables administrators to configure user accounts such that too many consecutive login failures due to incorrect passwords cause temporary account locking. The required number of failures and the lock time are configurable per account.

These new capabilities provide DBAs more complete control over password management. For more information, see Section 6.2.15, “Password Management”.

MySQL now supports FIPS mode, if compiled using OpenSSL, and an OpenSSL library and FIPS Object Module are available at runtime. FIPS mode imposes conditions on cryptographic operations such as restrictions on acceptable encryption algorithms or requirements for longer key lengths. See Section 6.8, “FIPS Support”.

The TLS context the server uses for new connections now is reconfigurable at runtime. This capability may be useful, for example, to avoid restarting a MySQL server that has been running so long that its SSL certificate has expired. See *Server-Side Runtime Configuration and Monitoring for Encrypted Connections*.

OpenSSL 1.1.1 supports the TLS v1.3 protocol for encrypted connections, and MySQL 8.0.16 and higher supports TLS v1.3 as well, if both the server and client are compiled using OpenSSL 1.1.1 or higher. See Section 6.3.2, “Encrypted Connection TLS Protocols and Ciphers”.

MySQL now sets the access control granted to clients on the named pipe to the minimum necessary for successful communication on Windows. Newer MySQL client software can open named pipe connections without any additional configuration. If older client software cannot be upgraded immediately, the new `named_pipe_full_access_group` system variable can be used to give a Windows group the necessary permissions to open a named pipe connection. Membership in the full-access group should be restricted and temporary.

Previously, MySQL user accounts authenticated to the server using a single authentication method. As of MySQL 8.0.27, MySQL supports multifactor authentication (MFA), which makes it possible to create accounts that have up to three authentication methods. MFA support entails these changes:

`CREATE USER` and `ALTER USER` syntax has been extended to permit specification of multiple authentication methods.

The `authentication_policy` system variable enables MFA policy to be established by controlling how many factors can be used and the types of authentication permitted for each factor. This places constraints on how the authentication-related clauses of `CREATE USER` and `ALTER USER` statements may be used.

Client programs have new `--password1`, `--password2`, and `--password3` command-line options for specifying multiple passwords. For applications that use the C API, the new `MYSQL_OPT_USER_PASSWORD` option for the `mysql_options4()` C API function enables the same capability.

In addition, MySQL Enterprise Edition now supports authentication to MySQL Server using devices such as smart cards, security keys, and biometric readers. This authentication method is based on the Fast Identity Online (FIDO) standard, and uses a pair of

plugins, `authentication_fido` on the server side and `authentication_fido_client` on the client side. The server-side FIDO authentication plugin is included only in MySQL Enterprise Edition distributions. It is not included in MySQL community distributions. However, the client-side plugin is included in all distributions, including community distributions. This enables clients from any distribution to connect to a server that has the server-side plugin loaded.

Multifactor authentication can use existing MySQL authentication methods, the new FIDO authentication method, or a combination of both. For more information, see Section 6.2.18, “Multifactor Authentication”, and Section 6.4.1.11, “FIDO Pluggable Authentication”.

Resource management. MySQL now supports creation and management of resource groups, and permits assigning threads running within the server to particular groups so that threads execute according to the resources available to the group. Group attributes enable control over its resources, to enable or restrict resource consumption by threads in the group. DBAs can modify these attributes as appropriate for different workloads. Currently, CPU time is a manageable resource, represented by the concept of “virtual CPU” as a term that includes CPU cores, hyperthreads, hardware threads, and so forth. The server determines at startup how many virtual CPUs are available, and database administrators with appropriate privileges can associate these CPUs with resource groups and assign threads to groups. For more information, see Section 5.1.16, “Resource Groups”.

Table encryption management. Table encryption can now be managed globally by defining and enforcing encryption defaults. The `default_table_encryption` variable defines an encryption default for newly created schemas and general tablespace. The encryption default for a schema can also be defined using the `DEFAULT ENCRYPTION` clause when creating a schema. By default, a table inherits the encryption of the schema or general tablespace it is created in. Encryption defaults are enforced by enabling

the `table_encryption_privilege_check` variable. The privilege check occurs when creating or altering a schema or general tablespace with an encryption setting that differs from the `default_table_encryption` setting, or when creating or altering a table with an encryption setting that differs from the default schema encryption. The `TABLE_ENCRYPTION_ADMIN` privilege permits overriding default encryption settings when `table_encryption_privilege_check` is enabled. For more information, see [Defining an Encryption Default for Schemas and General Tablespaces](#).

InnoDB enhancements. These InnoDB enhancements were added: The current maximum auto-increment counter value is written to the redo log each time the value changes, and saved to an engine-private system table on each checkpoint. These changes make the current maximum auto-increment counter value persistent across server restarts. Additionally:

A server restart no longer cancels the effect of the `AUTO_INCREMENT = N` table option. If you initialize the auto-increment counter to a specific value, or if you alter the auto-increment counter value to a larger value, the new value is persisted across server restarts.

A server restart immediately following a `ROLLBACK` operation no longer results in the reuse of auto-increment values that were allocated to the rolled-back transaction.

If you modify an `AUTO_INCREMENT` column value to a value larger than the current maximum auto-increment value (in an `UPDATE` operation, for example), the new value is persisted, and subsequent `INSERT` operations allocate auto-increment values starting from the new, larger value.

For more information, see [Section 15.6.1.6, “AUTO_INCREMENT Handling in InnoDB”](#), and [InnoDB AUTO_INCREMENT Counter Initialization](#).

When encountering index tree corruption, InnoDB writes a corruption flag to the redo log, which makes the corruption flag crash safe. InnoDB also writes in-memory corruption flag data to an

engine-private system table on each checkpoint. During recovery, InnoDB reads corruption flags from both locations and merges results before marking in-memory table and index objects as corrupt.

The InnoDB memcached plugin supports multiple get operations (fetching multiple key-value pairs in a single memcached query) and range queries. See Section 15.20.4, “InnoDB memcached Multiple get and Range Query Support”.

A new dynamic variable, `innodb_deadlock_detect`, may be used to disable deadlock detection. On high concurrency systems, deadlock detection can cause a slowdown when numerous threads wait for the same lock. At times, it may be more efficient to disable deadlock detection and rely on the `innodb_lock_wait_timeout` setting for transaction rollback when a deadlock occurs.

The

new `INFORMATION_SCHEMA.INNODB_CACHED_INDEXES` table reports the number of index pages cached in the InnoDB buffer pool for each index.

InnoDB temporary tables are now created in the shared temporary tablespace, `ibtmp1`.

The InnoDB tablespace encryption feature supports encryption of redo log and undo log data. See Redo Log Encryption, and Undo Log Encryption.

InnoDB supports `NOWAIT` and `SKIP LOCKED` options with `SELECT ... FOR SHARE` and `SELECT ... FOR UPDATE` locking read statements. `NOWAIT` causes the statement to return immediately if a requested row is locked by another transaction. `SKIP LOCKED` removes locked rows from the result set.

Here is the report table, in which the data that is sent via raspberry Pi is stored, it is stored with date and time , it is displayed on the website and the application, and it is change it every 8 hours.

The image shows two screenshots of a database management interface. The top screenshot displays the 'green' table with columns: id, username, email, password, and phoneno. The bottom screenshot displays the 'report' table with columns: id, time, w1, temp, hum, and w2.

Table: green

id	username	email	password	phoneno
79	green	green2022@gmail.com	123456	01022558899
80	ab	ab22@gmail.com	1234567	123456
81	greena	ab22@gmail.com	1234567	123456
82	greend	ab22@gmail.com	1234567	123456
86	nesma	nesma.abdelmawla@gma	123456	01287038787
87	ibra	ibra28032@gmail.com	123456	01027561511
166				
167	hamooo			
168	hamoomohf			0101010
169	Rananaser22	Rananaser52@gmail.co	rana1234567	01170575219
170	Rana23	Rananaser25@gmail.co	rana1234567	01250727019
171	ebrahem	ib	123456	01027561511

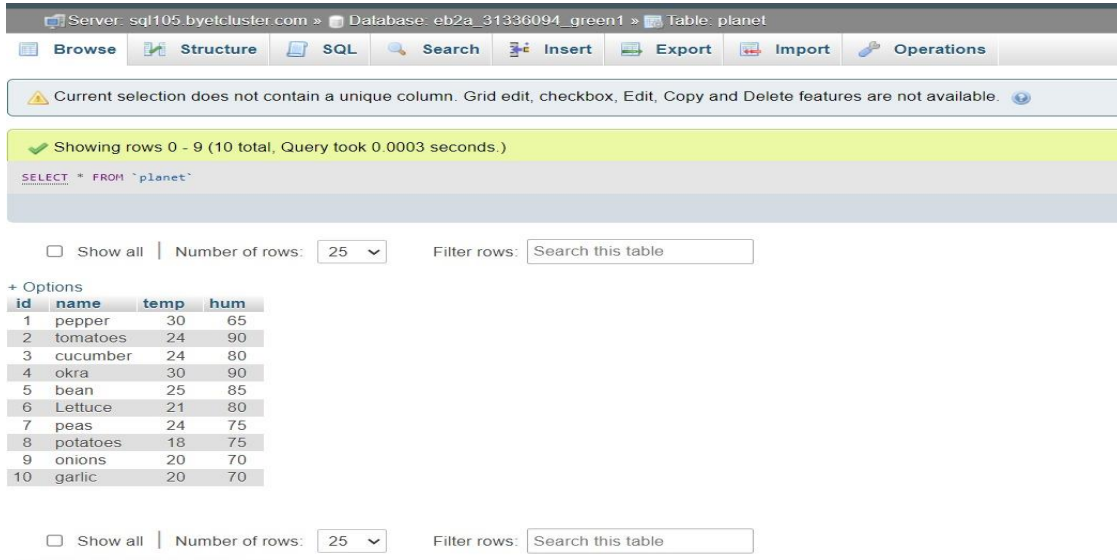
Table: report

id	time	w1	temp	hum	w2
11	2022-06-29 07:31:32	0	0	0	0
12	2022-06-29 07:31:32	0	0	0	0
13	2022-06-29 07:31:32	0	0	0	0
14	2022-06-29 07:32:39	0	0	0	0
15	2022-06-29 07:32:42	0	0	0	0
16	2022-06-29 07:34:59	0	0	0	0
17	2022-06-29 07:37:26	1	31	68	1
18	2022-06-29 07:38:59	1	31	68	1
19	2022-06-29 07:41:27	0	0	0	0
20	2022-06-29 07:43:27	1	31	0	0
21	2022-06-29 07:44:39	1	31	0	0
22	2022-06-29 07:45:34	1	31	0	0
23	2022-06-29 07:45:37	1	31	0	0
24	2022-06-29 07:45:54	1	31	0	0
25	2022-06-29 07:45:55	1	31	0	0

Fig.27. Report page Layouts

Search page:

Search page helps the customer the know the temperature and humidity of each elements , and he can spray specific element to facilitate time and effort.



Server: sql105.byetcluster.com » Database: eb2a_31336094_green1 » Table: planet

Browser Structure SQL Search Insert Export Import Operations

⚠ Current selection does not contain a unique column. Grid edit, checkbox, Edit, Copy and Delete features are not available.

✔ Showing rows 0 - 9 (10 total, Query took 0.0003 seconds.)

```
SELECT * FROM `planet`
```

Show all | Number of rows: 25 | Filter rows: Search this table

+ Options

id	name	temp	hum
1	pepper	30	65
2	tomatoes	24	90
3	cucumber	24	80
4	okra	30	90
5	bean	25	85
6	Lettuce	21	80
7	peas	24	75
8	potatoes	18	75
9	onions	20	70
10	garlic	20	70

Show all | Number of rows: 25 | Filter rows: Search this table

Fig.28. Search Layouts

System 3D Model

INITIAL IMAGINARY FROM OF VERTICAL FARMING

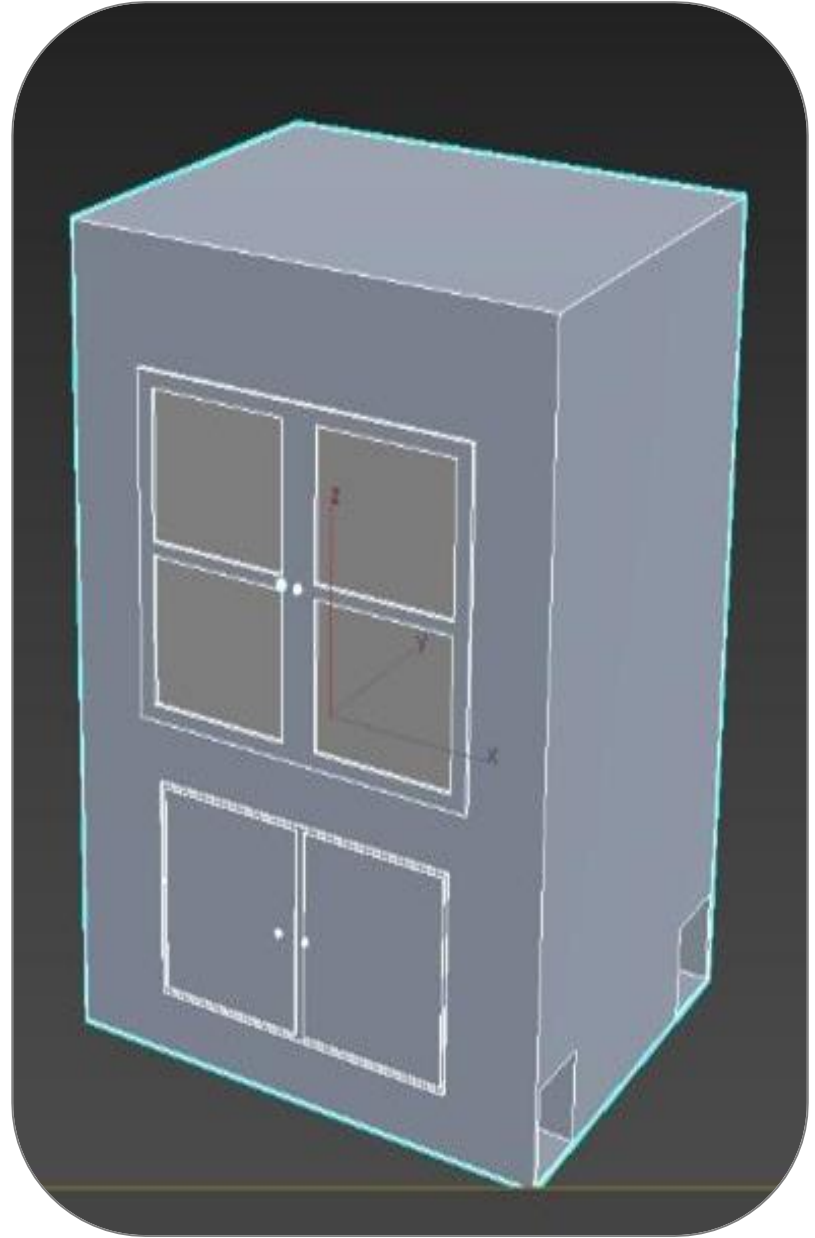
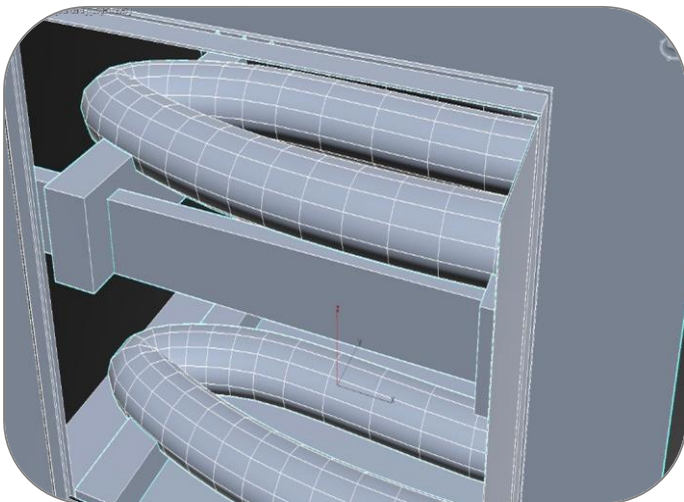
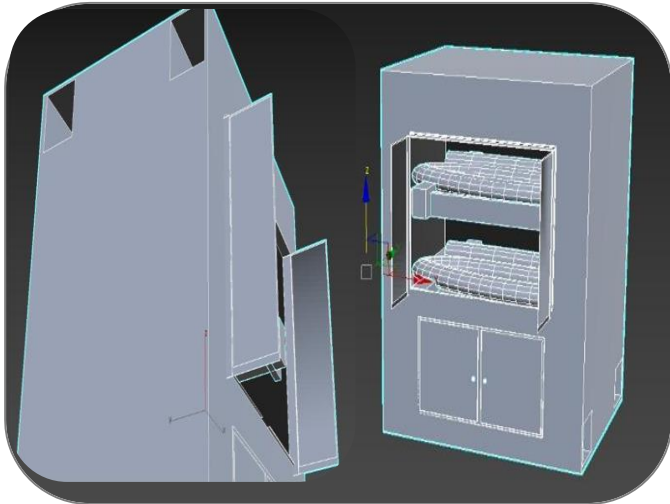
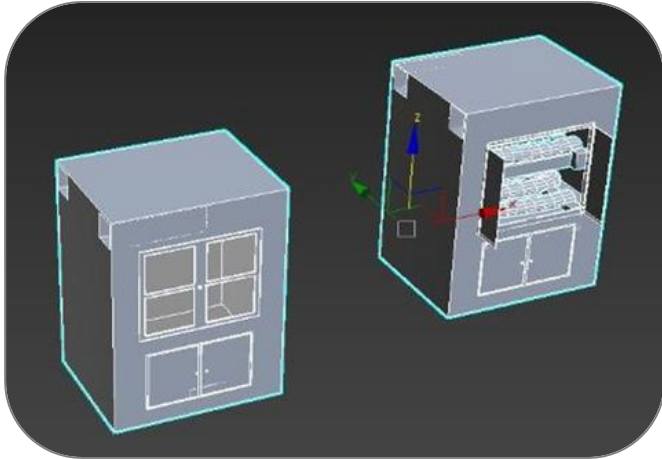


Fig.29. System 3D Model.

System Prototype

The idea of this version is that it is a closed system that controls the temperature, humidity and degree of soil moisture, and controls the intensity of the lighting using Bluetooth and mobile application.

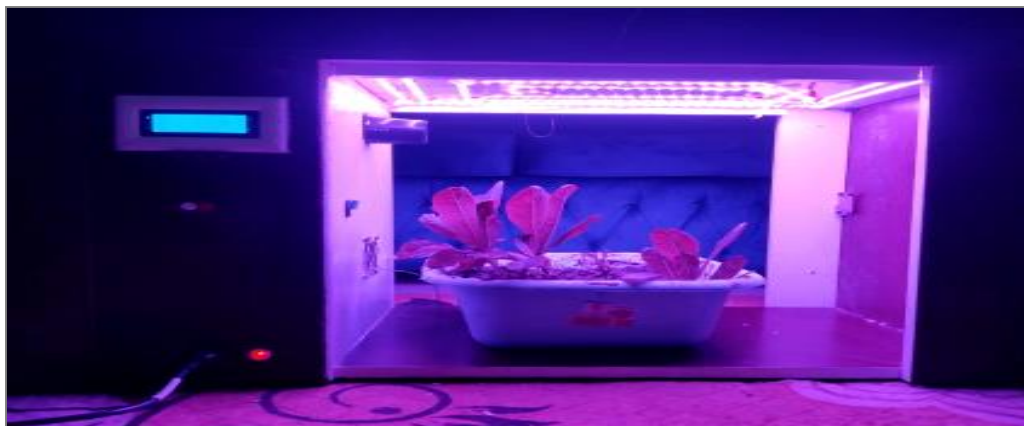
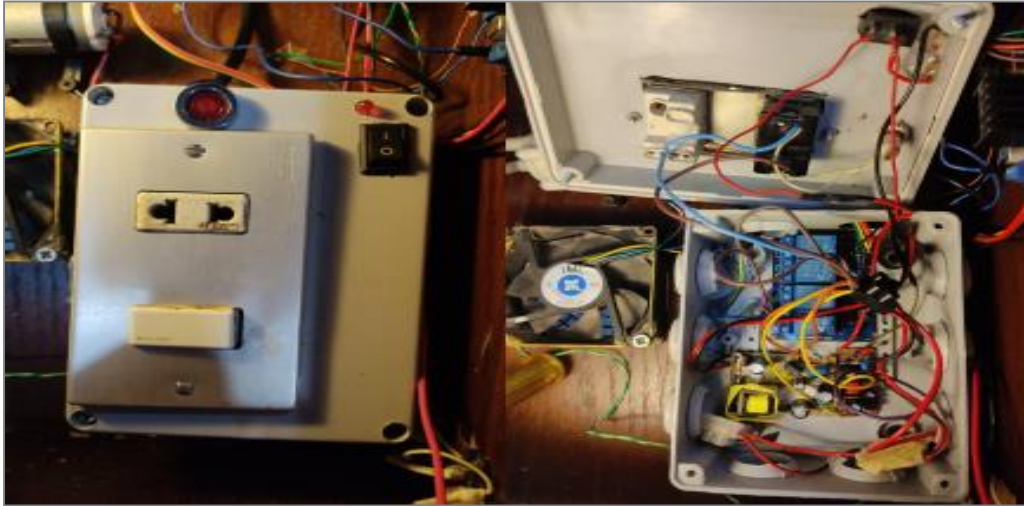
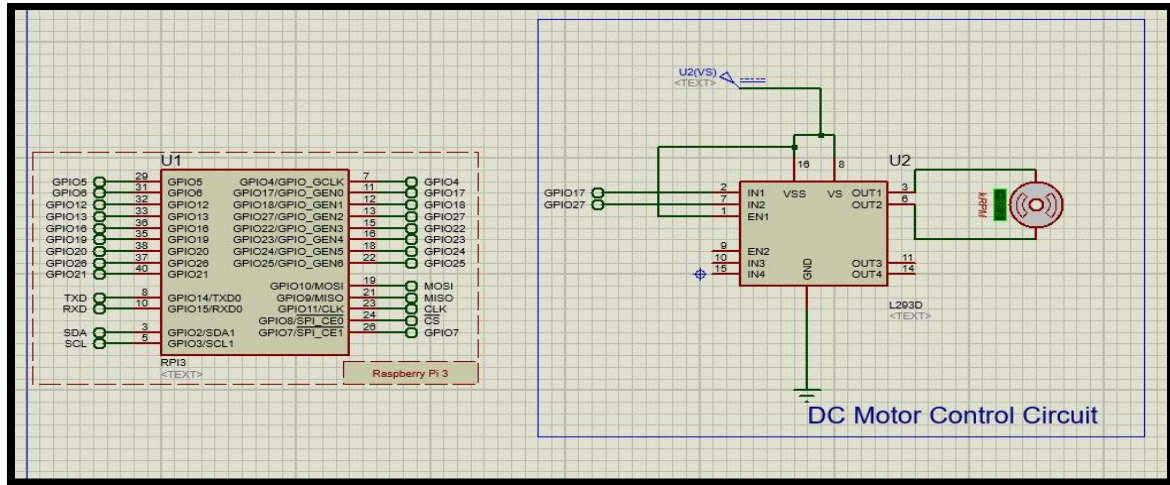


Fig.30. System Prototype.

Fans control circuits

1. Fan 1

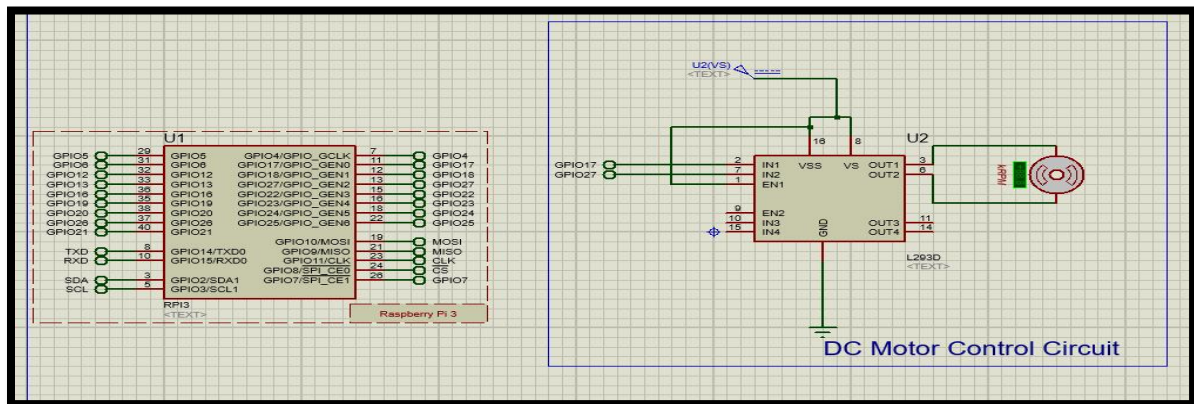


(a)

This circuit contains one fan that is working as a cooler for the system.

This fan is connected with the Dht22 sensor (temperature ,humidity) To work automatically when the temperature is rising.

2. Fan 2



(b)

This circuit contains one fan that is working as a humidity disposal for the system.

This fan is connected with the Dht22 sensor (temperature ,humidity) To work automatically when the humidity is rising.

Sensors connected circuit

1. Ultrasonic sensor

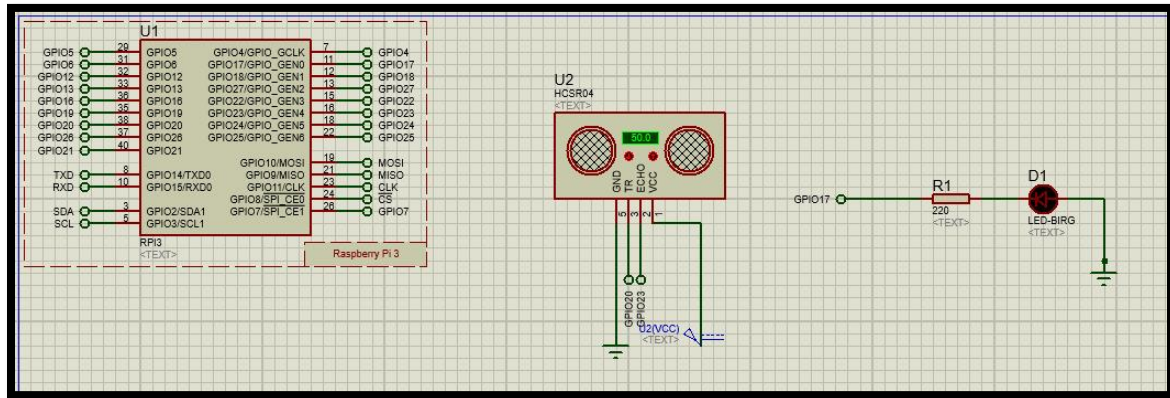
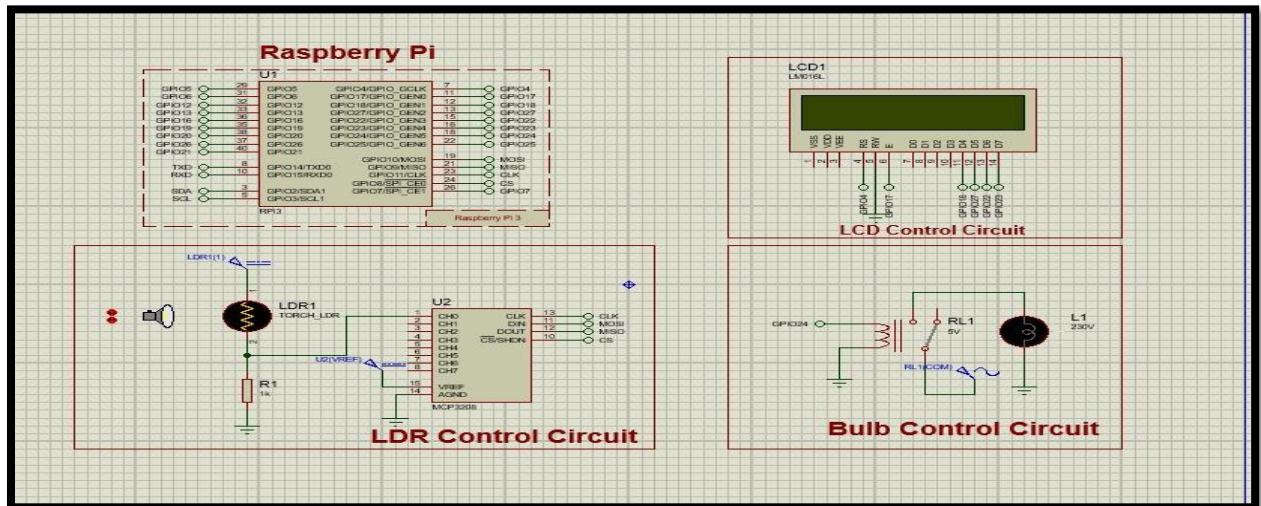


Fig.30. Sensors connected circuit.

(a)

This circuit contain ultrasonic sensor connected to the raspberry pi. It is responsible for calculating the water level and based on that it sets the functionality of the pump.

2. LDR sensor



This circuit contain ldr sensor connected to raspberry pi and lcd and led.

Ldr sensor is responsible for detecting the light and based on the intensity of the light the led is will light up .

System Circuit design

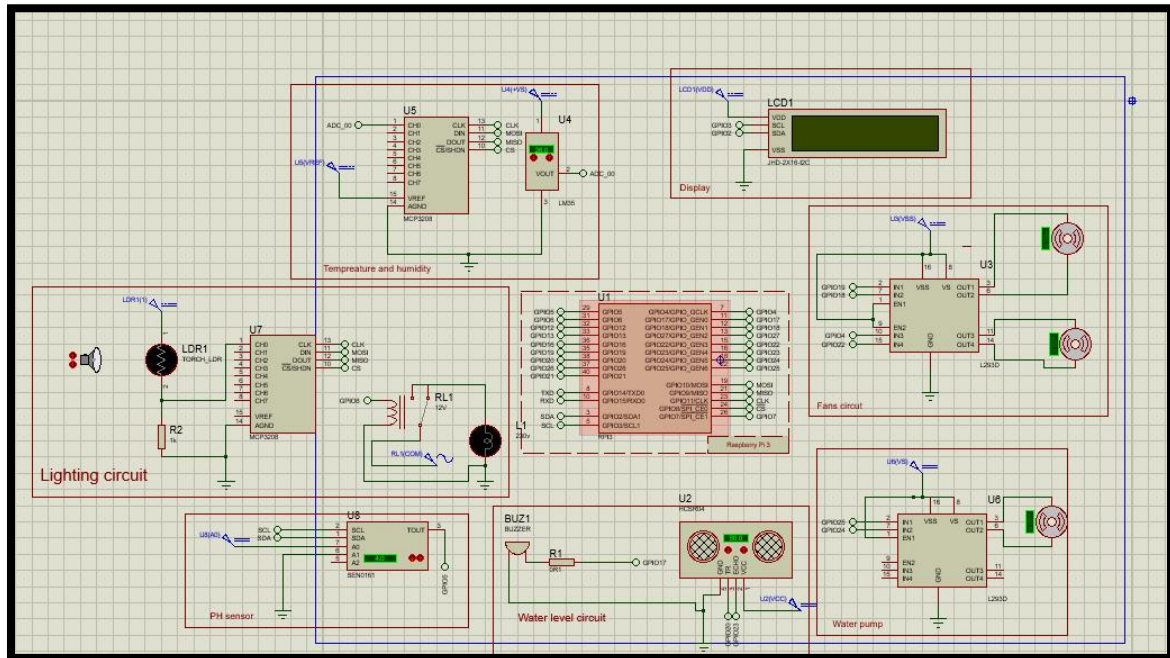
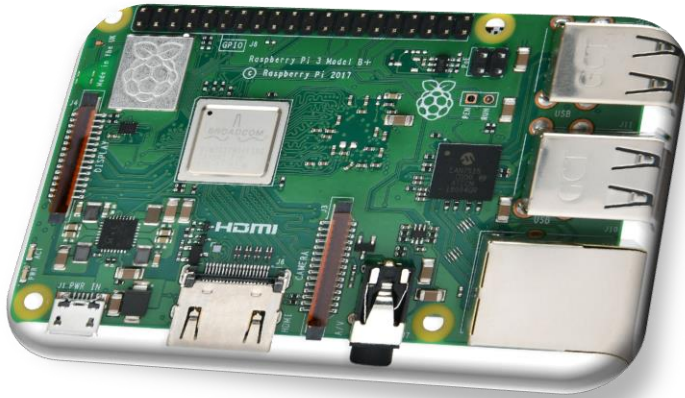


Fig.31. Full System Circuit.

Hardware platform

Raspberry pi model 3 B+



- Elements:- DHT11
- LDR
- Res 10k
- Led
- Input power 5v, .5A

The function of this circuit is that the temperature, humidity and light resistance sensor is installed in it so that it is fixed at the plant, so it measures the temperature around the plant and the ratio of lighting to its connection.

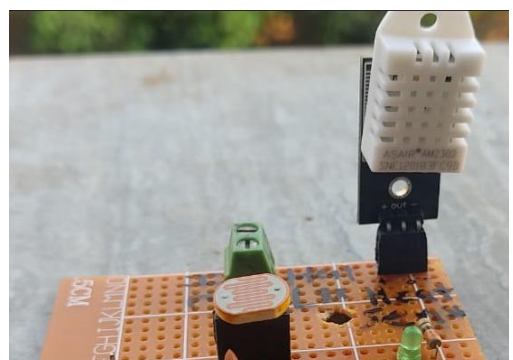
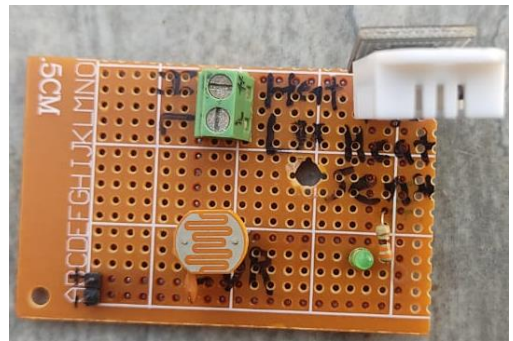
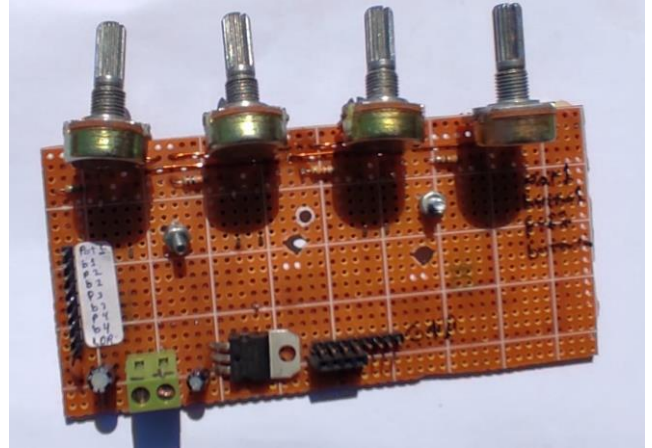


Fig.32. Hardware platform.

- Pot
- Pushbutton
- Capacitors 10,100micro
- Reg Lm7805
- Input power 12v

The function of the circuit is that it is the controller of the project, so that the user can control the



lighting, the fan and the hood so that he can lock it and open it through the push button or change the degree of lighting or the fan through the variable resistance And in the Regulator, it reduces the voltage from 12 to 5 so that it feeds the rotors around it.

- Computer power supply
- Fuses
- Led
- Reg LM7805
- Fan
- Relay medule 4 Ch
- Switch
- Input power 220v

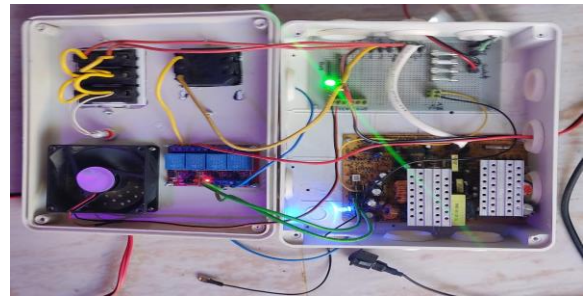


Fig.33. power box of the project.

The power box of the project, it enters 220 volts, it enters the circuit mentioned in the project, it is distributed to 320 volts, like the heater and power play, and from the power supply, of course, we feed it all And there is a Relay Medial up to 220 volts like the pump and the heater that we control, and there is a lamp showing them if it is in power connected to the box or not, and a plug and a fan to cool the box.

Water Float Switch

First, we switch between automatic and manual via a button in the mobile application or through the control panel on the interface of the device. In automatic mode, the raspberry takes decisions on its own, since it sees the values of the sensors that it has, and on the basis of which it operates the heater, fans, hoods, or lights. Of course, the temperature controls the heater or fans. Humidity controls the hood. And the LDR controls the lighting, and there is another need. Most plants need 12 hours of illumination per day, according to their intensity, so it will be in a variable resistance.



Fig.34. Water Float Switch.

Manual Water Float Switch

The manual situation remains simple, it is the user if the system looks at the screen of the device and sees the values of the sensors and evaluates by itself and turns on what and locks what Or through the mobile application, he sees the reboot and knows what happened in the past hours, sees the live evaluation, and then shuts down and turns on what.

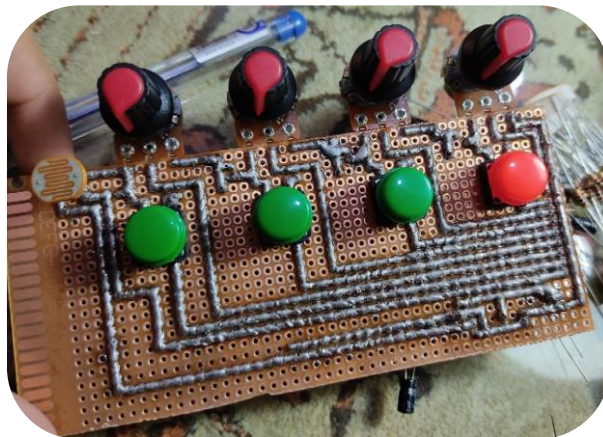


Fig.35. Water and Manual Water Float Switch.

And in ventilation fans, hoods and heaters, if the temperature is low and the light is suitable for the plant.



❖ The basement is where the electrical connections and devices used to control the device.

❖ The out Veiw Of Plants

Fig.36. Out view



CHAPTER 5

Conclusions & Future work

Chapter 5: Conclusions & Future work

5.1. Conclusions

Vertical farming is still a relatively new technology. Companies have yet to succeed in mass-producing crops cost-effectively to fulfill rising food demand. How crucial a role vertical farming will play in the future to meet the challenge of rising food demand will be determined by the performance of farms like AeroFarms. However, technology created for vertical farms is being adopted by other areas of the indoor farming sector, such as greenhouses, which can utilize natural sunlight but require much more space and longer routes to market. The suggested farm regulates the environmental conditions needed to produce plants in indoor vertical farms autonomously. It excels in scalability and convenience, since extra racks or slave nodes may be quickly added without requiring the end-user to perform any complicated wiring or setups. The dashboard that comes with it may easily be expanded to undertake a complicated analysis of the many variables received by the sensors.

5.2. Challenges & Future work

Vertical farming still has many limitations [24-27] as follows: (i) No Established Economics: The financial feasibility of this new farming method remains uncertain. The financial situation is changing, however, as the industry matures and technologies improve. (ii) Difficulties with Pollination: Vertical farming takes place in a controlled environment without the presence of insects. As such, the pollination process needs to be done manually, which will be labor-intensive and costly. (iii) Labor Costs: As high as energy costs are in vertical farming, labor costs can be even higher due to their concentration in urban centers where wages are higher, as well as the need for more skilled labor.

Automation in vertical farms, however, may lead to the need for fewer workers. Manual pollination may become one of the more labor-intensive functions in vertical farms. (iv) Too Much Dependency on Technology: The development of better

technologies can always increase efficiency and lessen costs. But the entire vertical farming is extremely dependent on various technologies for lighting, maintaining temperature, and humidity. Losing power for just a single day can prove very costly for a vertical farm. Many believe the technologies in use today are not ready for mass adoption.

Otherwise, many open issues need researchers efforts, these challenges could be summarized as [28-30]: (i) Lack of experienced farmers familiar with this technology; (ii) Economic viability of providing enough water and energy to a large scale farm; (iii) “Food from Chemicals”; (iv) Requires a collective effort of indoor agronomists, businessmen, engineers, architects, politicians, etc.

References

- [1] Al-Chalabi, M. 2015. Vertical farming: Skyscraper sustainability?. *Sustainable Cities and Society*, 18: 74-77. doi: 10.1016/j. scs.2015.06.003.
- [2] Teo, Y. L., & Go, Y. I. (2021). Techno-economic-environmental analysis of solar/hybrid/storage for vertical farming system: A case study, Malaysia. *Renewable Energy Focus*, 37, 50-67.
- [3] Despommier, D. 2009. The rise of vertical farms. *Scientific America*, 301(5): 80-87. doi: 10.1038/scientificamerican1109-80.
- [4] United Nations. "World Urbanization Prospects: The 2018 Revision." Accessed Dec. 16, 2020.
- [5] Olle, M. and A. Viršile, 2013. The effects of light-emitting diode lighting on greenhouse plant growth and quality. *Agricultural And Food Science*, 22(2): 223-234. doi: 10.23986/afsci.7897.
- [6] Columbia University Earth Institute. "Vertical Farms: From Vision to Reality." Accessed Dec. 16, 2020.
- [7] Kheir Al-Kodmany. "The Vertical Farm: A Review of Developments and Implications for the Vertical City," Page 4. Download PDF. Accessed Dec. 16, 2020.
- [8] Columbia University Earth Institute. "How Sustainable Is Vertical Farming? Students Try to Answer the Question." Accessed Dec. 16, 2020.
- [9] Research and Markets. "Vertical Farming Market in US - Industry Outlook and Forecast 2019-2024." Accessed Dec. 16, 2020.

- [10] Grand View Research. "U.S. Fruit & Vegetables Market Size, Share & Trends Analysis Report, by Product Type (Fresh, Dried, and Frozen), and Distribution Channel Type, Competitive Landscape, and Segment Forecasts, 2018 - 2025." Accessed Dec. 16, 2020.
- [11] Abd El-Mawla, N., Badawy, M., & Arafat, H. (2019). IoT for the failure of climate-change mitigation and adaptation and IIoT as a future solution. *World J. Environ. Eng.*, 6(1), 7-16.
- [12] Abd El-Mawla, N., & Nagy, A. (2020). IoT and Civil Engineering Based Solutions for Global And Environmental Risks of 2019. *Development*, 20, 24.
- [13] Despommier, D. 2013. Farming up the city: the rise of urban vertical farms. *Trends In Biotechnology*, 31(7): 388-389. doi: 10.1016/j.tibtech.2013.03.008.
- [14] Tsai, C. and T. Liang, 2018. Application of IoT technology in the simple micro-farming environmental monitoring. *IEEE International Conference on Advanced Manufacturing (ICAM)*, Yunlin, 2018, p. 170-172.
- [15] Ruengittinun, S., S. Phongsamsuan and P. Sureeratanakorn, 2017. Applied internet of thing for smart hydroponic farming ecosystem (HFE). *10th International Conference on Ubi-media Computing and Workshops (Ubi-Media)*, Pattaya, 2017, p. 1-4.
- [16] Ismail, M.I.H. and N.M. Thamrin, 2017. IoT implementation for indoor vertical farming watering system. *2017 International Conference on Electrical, Electronics and System Engineering (ICEESE)*, Kanazawa, 2017, p. 89-94.

- [17] Kalantari, F., O. Tahir, R. Joni and E. Fatemi, 2018. Opportunities and Challenges in Sustainability of Vertical Farming: A Review. *Journal Of Landscape Ecology*, 11(1): 35-60. doi: 10.1515/jlecol-2017-0016.
- [18] Liwal, K.K., C. Copiaco, M. Vohra, H. Rahman, N. Abdulaziz and
- [19] Belista, F.C.L., M.P.C. Go, L.L. Luceñara, C.J.G. Policarpio, X.J.M. Tan and R.G. Baldovino, 2018. A smart aeroponic tailored for IoT vertical agriculture using network connected modular environmental chambers. 2018 IEEE 10th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment and Management (HNICEM), Baguio City, 2018, p. 1-4.
- [20] Boopathy, S., Anand, K. G., & Rajalakshmi, N. R. (2020). Smart Irrigation System for Mint Cultivation through Hydroponics Using IOT. *TEST Engineering and Managemnet*, 83.
- [21] Droms, R. (1997). RFC2131: Dynamic Host Configuration Protocol.
- [22] Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., & Berners-Lee, T. (1999). Hypertext transfer protocol–HTTP/1.1.
- [23] Tresanchez, M., Pujol, A., Pallejà, T., Martínez, D., Clotet, E., & Palacín, J. (2019). An inexpensive wireless smart camera system for IoT applications based on an ARM Cortex-M7 microcontroller. *J. Ubiquitous Syst. Pervasive Netw*, 11, 01-08.
- [24] Möller Voss, P. (2013). Vertical farming: An agricultural revolution on the rise.
- [25] Benke, K., & Tomkins, B. (2017). Future food-production systems: vertical farming and controlled-environment agriculture. *Sustainability: Science, Practice and Policy*, 13(1), 13-26.

- [26] Miller, A. (2011). Scaling up or selling out? A critical appraisal of current developments in vertical farming (Doctoral dissertation, Carleton University).
- [27] Banerjee, C., & Adenaueer, L. (2014). Up, up and away! The economics of vertical farming. *Journal of Agricultural Studies*, 2(1), 40-60.
- [28] Jaeger, S. R., Chheang, S. L., & Ares, G. (2022). Text highlighting as a new way of measuring consumers' attitudes: A case study on vertical farming. *Food Quality and Preference*, 95, 104356.
- [29] Saxena, N. N. (2021). The Review on Techniques of Vertical Farming. *International Journal of Modern Agriculture*, 10(1), 732-738.
- [30] Naskali, A. T., Pinarer, O., & Tolga, A. C. (2022). Vertical Farming: Under Climate Change Effect. In *Environment and Climate-smart Food Production* (pp. 259-284). Springer, Cham.
- [31] <https://beyondstandards.ieee.org> › IEEE-addresses-standards- for-the-cloud (Last Accessed: 01/07/2022).
- [32] https://www.gristprojectmanagement.us/software-development/ieee-process-map-for-software-life-cycle_process.html (Last Accessed: 01/07/2022).
- [33] N. Barroca, F. Velez, and P. Chatzimisios, "Performance enhancement of IEEE 802.15.4 by employing RTS/CTS and frame concatenation", *IET Wireless Sensor Systems*, Vol.10 Iss.6, pp.308-319 <https://doi.org/10.1049/iet-wss.2019.0003>, 2020.
- [34] https://en.wikipedia.org/wiki/IEEE_802.15.4 (Last Accessed: 02/07/2022).
- [35] https://en.wikipedia.org/wiki/IEEE_802.15#IEEE_802.15.2:_Coexistence (Last Accessed: 02/07/2022).
- [36] https://en.wikipedia.org/wiki/IEEE_802.11 (Last Accessed: 02/07/2022).

Appendix

I. Software codes

```
import RPi.GPIO as GPIO
import Adafruit_DHT
import time
import os
import I2C_LCD_driver
GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)
global lcd
global humidity
global temperature
Fan = 17
Heat = 27
suction= 5
DHT_PIN = 4
waterlevel = 22
buzzer = 10
ldr = 9
led = 11
pump = 26
```

```

delayt=1
value=0
door=0
b3=18
b3s=False
GPIO.setup (19, GPIO .IN, pull_up_down = GPIO.PUD_UP)
GPIO.setup(23, GPIO.OUT)#red
GPIO.setup(24, GPIO.OUT)#blue
GPIO.setup(25, GPIO.OUT)#green
GPIO.setup(Heat, GPIO.OUT)
GPIO.setup(door, GPIO.OUT)
GPIO.setup(Fan, GPIO.OUT)
GPIO.setup(suction, GPIO.OUT)
DHT_SENSOR = Adafruit_DHT.DHT22
#humidity,          temperature          =
Adafruit_DHT.read_retry(DHT_SENSOR, DHT_PIN)
GPIO.setup(waterlevel, GPIO.IN, GPIO.PUD_UP)
GPIO.setup(buzzer, GPIO.OUT)
GPIO.setup(led, GPIO.OUT)
GPIO.output(led, False)
GPIO.setup(pump, GPIO.OUT)
GPIO.output(pump,0)
lcd = I2C_LCD_driver.lcd()

```

```

def rc_time (ldr):
    count = 0
    #Output on the pin for
    GPIO.setup(ldr, GPIO.OUT)
    GPIO.output(ldr, False)
    time.sleep(delayt)
    #Change the pin back to input
    GPIO.setup(ldr, GPIO.IN)
    #Count until the pin goes high
    while (GPIO.input(ldr) == 0):
        count += 1
    return count

while True:
    if GPIO.input(b3) == 0:
        print("Button 3 Was Pressed:")
        if b3s == False:
            os.system('ak.py')
            # Auto
            b3s = True
            time.sleep(.5)
        else:
            print("dont do anything")
            # Mobile

```

```

    b3s = False
    time.sleep(.5)
while True: #DHT22 with Fans and Heater and suction
    humidity,          temperature          =
Adafruit_DHT.read_retry(DHT_SENSOR, DHT_PIN)
    lcd lcd_display_string("Temp=%0.1fC    Humi=%0.1f"    %
(temperature, humidity), 1)
    #lcd lcd_display_string("humid= %0.1f " % humidity, 2)
    ti=4
    w11=333
    w12=555
import webbrowser
import os
import time
browserExe = "chromeium"

url="http://green.eb2a.com/ins.php?time=%d&w11=%d&temp=%0
.1f&hum=%0.1f&w12=%d" % (ti,w11,temperature,humidity,w12)
    print(url)
    webbrowser.open(url)
    time.sleep(10)
    os.system("pkill " + browserExe)
    break

```

```

if temperature < 26:
    #print("Temperature is Less than 26 degree.")
    lcd lcd_display_string("Fan is OFF." , 2)
    lcd lcd_display_string("Heater is ON.  " , 3)
    lcd lcd_display_string("suction is OFF." , 4)
    #print("Fan is OFF.")
    #print("Heater is ON.")
    #print("suction is OFF.")
    GPIO.output(Fan,0)
    GPIO.output(Heat,1)
    GPIO.output(suction,0)
elif temperature > 26:
    lcd lcd_display_string("Fan is ON." , 2)
    lcd lcd_display_string("Heater is OFF.  " , 3)
    lcd lcd_display_string("suction is ON." , 4)
    #print("Temperature is Greater than 26 degree.")
    #print("Fan is ON.")
    #print("Heater is OFF.")
    #print("suction is ON.")
    GPIO.output(Heat,0)
    GPIO.output(Fan,1)
    GPIO.output(suction,1)
elif temperature == 26:

```

```

lcd lcd_display_string("Fan is OFF." , 2)
lcd lcd_display_string("Heater is OFF.  " , 3)
lcd lcd_display_string("suction is OFF." , 4)
#print("Temperature at an acceptable degree.")
#print("Heater is OFF.")
#print("Fan is OFF.")
#print("suction is OFF.")
GPIO.output(Heat,0)
GPIO.output(Fan,0)
GPIO.output(suction,0)
else :
    lcd lcd_display_string("Fan is OFF." , 2)
    lcd lcd_display_string("Heater is OFF.  " , 3)
    lcd lcd_display_string("suction is OFF." , 4)
    #print("Heater is OFF.")
    #print("Fan is OFF.")
    #print("suction is OFF.")
    GPIO.output(Heat,0)
    GPIO.output(Fan,0)
    GPIO.output(suction,0)
break
#time.sleep(0.5)
#lcd.clear()

```

```

while True: # water flout switch with buzzer
    water_state = GPIO.input(waterlevel)
    if water_state == GPIO.HIGH:
        time.sleep(3)
        lcd.lcd_display_string("          ", 4)
        lcd.lcd_display_string("          ", 2)
        lcd.lcd_display_string("water level is high" , 3)
        print ("HIGH")
        GPIO.output(buzzer,1)
    else:
        time.sleep(3)
        lcd.lcd_display_string("          ", 4)
        lcd.lcd_display_string("          ", 2)
        lcd.lcd_display_string("water level is low" , 3)
        print ("LOW")
        GPIO.output(buzzer,0)
        time.sleep(0.5)
    break
#time.sleep(0.5)
while True:
    print("Ldr Value:")
    value = rc_time(ldr)
    print(value)

```



```

if ( value > 100 ):
    lcd lcd_display_string("          ", 2)
    lcd lcd_display_string("          ", 4)
    lcd lcd_display_string("Light is ON   ", 3)
    #print("Lights are ON")
    GPIO.output(led, True)

if (value < 100):
    lcd lcd_display_string("          ", 2)
    lcd lcd_display_string("          ", 4)
    lcd lcd_display_string("Light is Off  ", 3)
    print("Lights are OFF")
    GPIO.output(led, False)

break

while True:
    input = GPIO.input (19)
    print (input)
    time.sleep(5)
    if input == 0:
        GPIO.output(23,True)
        GPIO.output(24,True)
        GPIO.output(25,True)
    else:
        GPIO.output(23,True)

```

```

        GPIO.output(24,True)
        GPIO.output(25,False)
    break

import RPi.GPIO as GPIO
from http.server import BaseHTTPRequestHandler, HTTPServer
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)
p=GPIO.PWM(17,1000)
GPIO.setup(17, GPIO.OUT)
GPIO.setup(7, GPIO.OUT)
GPIO.setup(27, GPIO.OUT)
GPIO.setup(10, GPIO.OUT)
GPIO.setup(23, GPIO.OUT)#red
GPIO.setup(24, GPIO.OUT)#blue
GPIO.setup(25, GPIO.OUT)#green
GPIO.setup(1, GPIO.OUT)
request = None
temp1=None
temp2=None
class RequestHandler_httptd(BaseHTTPRequestHandler):
    def do_GET(self):
        global request
        global temp1

```

```

global temp2
messagetosend = bytes('hello',"utf")
self.send_response(200)
self.send_header('Content-Type', 'text/plain')
self.send_header('Content-Length', len(messagetosend))
self.end_headers()
self.wfile.write(messagetosend)
request = self.requestline
request = request[5 : int(len(request)-9)]
print(request)
if request == 'LedOn':
    GPIO.output(23,True)
    GPIO.output(24,True)
    GPIO.output(25,False)
if request == 'LedOff':
    GPIO.output(23,False)
    GPIO.output(24,False)
    GPIO.output(25,False)

if request == 'WhiteLedOn':
    GPIO.output(23,True)
    GPIO.output(24,True)
    GPIO.output(25,True)

```

```
if request == 'WhiteLedOff':
```

```
    GPIO.output(23,False)
```

```
    GPIO.output(24,False)
```

```
    GPIO.output(25,False)
```

```
if request == 'FanOn':
```

```
    GPIO.output(17,True)
```

```
if request == 'FanOff':
```

```
    GPIO.output(17,False)
```

```
if request == 'HeatOn':
```

```
    GPIO.output(7,False)
```

```
if request == 'HeatOff':
```

```
    GPIO.output(7,True)
```

```
if request == 'HoodOn':
```

```
    GPIO.output(27,True)
```

```
if request == 'HoodOff':
```

```
    GPIO.output(27,False)
```

```
if request == 'UnMute':
```

```
    GPIO.output(10,True)
```

```
if request == 'Mute':
```

```
    GPIO.output(10,False)
```

```
if request == 'PumpOn':
```

```
    GPIO.output(1,1)
```

```

if request == 'PumpOff':
    GPIO.output(1,0)
if request == 'auto':
    print("OKAY")
    #automation system code
if request == 'notauto':
    print("notOKAY")
    #automation system code
if request == 'xSlider0':
    GPIO.output(17, False)
    p.ChangeDutyCycle(0)
if request == 'xSlider25':
    GPIO.output(17, True)
    p.ChangeDutyCycle(25)
if request == 'xSlider50':
    GPIO.output(17, True)
    p.ChangeDutyCycle(50)
if request == 'xSlider100':
    GPIO.output(17, True)
    p.ChangeDutyCycle(75)
return

server_address_httpd = ('192.168.1.2',8080)
httpd = HTTPServer(server_address_httpd, RequestHandler_httpd)

```

```
print('starting')
httpd.serve_forever()
GPIO.cleanup()
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:tools="http://schemas.android.com/tools"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:background="@drawable/bgb"
  tools:context=".MainActivity">
  <LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_alignParentBottom="true"
    android:gravity="center">
  </LinearLayout>
  <TextView
    android:id="@+id/heat"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginStart="5dp"
    android:layout_marginTop="140dp"
```

```
android:layout_marginEnd="340dp"  
android:layout_marginBottom="130dp"  
android:gravity="left"  
android:text="HEAT:"  
android:textColor="@color/white"  
android:textSize="20dp"  
android:textStyle="bold" />
```

```
<TextView
```

```
    android:id="@+id/humidity"  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:layout_marginStart="250dp"  
    android:layout_marginTop="140dp"  
    android:layout_marginEnd="10dp"  
    android:layout_marginBottom="130dp"  
    android:gravity="left"  
    android:text="Humidity:"  
    android:textColor="@color/white"  
    android:textSize="20dp"  
    android:textStyle="bold" />
```

```
<TextView
```

```
    android:id="@+id/externallighting"  
    android:layout_width="match_parent"
```

```
android:layout_height="wrap_content"  
android:layout_marginStart="5dp"  
android:layout_marginTop="250dp"  
android:layout_marginEnd="240dp"  
android:layout_marginBottom="130dp"  
android:gravity="left"  
android:text="External lighting:"  
android:textColor="@color/white"  
android:textSize="20dp"  
android:textStyle="bold" />
```

```
<TextView
```

```
android:id="@+id/interiorlighting"  
android:layout_width="match_parent"  
android:layout_height="wrap_content"  
android:layout_marginStart="250dp"  
android:layout_marginTop="250dp"  
android:layout_marginEnd="10dp"  
android:layout_marginBottom="130dp"  
android:gravity="left"  
android:text="Interior lighting:"  
android:textColor="@color/white"  
android:textSize="20dp"  
android:textStyle="bold" />
```


<EditText

```
    android:id="@+id/heatc"  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:layout_marginStart="5dp"  
    android:layout_marginTop="180dp"  
    android:layout_marginEnd="250dp"  
    android:layout_marginBottom="10dp"  
    android:background="#30FFFFFF"  
    android:drawablePadding="20dp"  
    android:padding="10dp" />
```

<EditText

```
    android:id="@+id/exlighting"  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:layout_marginStart="5dp"  
    android:layout_marginTop="290dp"  
    android:layout_marginEnd="250dp"  
    android:layout_marginBottom="10dp"  
    android:background="#30FFFFFF"  
    android:drawablePadding="20dp"  
    android:padding="10dp" />
```

<EditText

```
android:id="@+id/humidityy"  
android:layout_width="match_parent"  
android:layout_height="wrap_content"  
android:layout_marginStart="250dp"  
android:layout_marginTop="180dp"  
android:layout_marginEnd="10dp"  
android:layout_marginBottom="10dp"  
android:background="#30FFFFFF"  
android:drawablePadding="20dp"  
android:padding="10dp" />
```

<EditText

```
android:id="@+id/inlighting"  
android:layout_width="match_parent"  
android:layout_height="wrap_content"  
android:layout_marginStart="250dp"  
android:layout_marginTop="290dp"  
android:layout_marginEnd="10dp"  
android:layout_marginBottom="10dp"  
android:background="#30FFFFFF"  
android:drawablePadding="20dp"  
android:padding="10dp" />
```

<TextView

```
android:id="@+id/upperwater"
```

```
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_marginStart="5dp"
android:layout_marginTop="350dp"
android:layout_marginEnd="290dp"
android:layout_marginBottom="130dp"
android:gravity="left"
android:text="Upper water:"
android:textColor="@color/white"
android:textSize="20dp"
android:textStyle="bold" />
```

<TextView

```
android:id="@+id/lowerwater"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_marginStart="280dp"
android:layout_marginTop="350dp"
android:layout_marginEnd="0dp"
android:layout_marginBottom="130dp"
android:gravity="left"
android:text="Lower water:"
android:textColor="@color/white"
android:textSize="20dp"
```

```
android:textStyle="bold" />
```

```
<TextView
```

```
android:id="@+id/pipewater"
```

```
android:layout_width="match_parent"
```

```
android:layout_height="wrap_content"
```

```
android:layout_marginStart="150dp"
```

```
android:layout_marginTop="350dp"
```

```
android:layout_marginEnd="140dp"
```

```
android:layout_marginBottom="130dp"
```

```
android:gravity="left"
```

```
android:text="Pipe water:"
```

```
android:textColor="@color/white"
```

```
android:textSize="20dp"
```

```
android:textStyle="bold" />
```

```
<TextView
```

```
android:id="@+id/doorswitch"
```

```
android:layout_width="match_parent"
```

```
android:layout_height="wrap_content"
```

```
android:layout_marginStart="5dp"
```

```
android:layout_marginTop="450dp"
```

```
android:layout_marginEnd="290dp"
```

```
android:layout_marginBottom="130dp"
```

```
android:gravity="left"
```

```
android:text="Door switch:"  
android:textColor="@color/white"  
android:textSize="20dp"  
android:textStyle="bold" />
```

```
<TextView
```

```
    android:id="@+id/heatswitch"  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:layout_marginStart="150dp"  
    android:layout_marginTop="450dp"  
    android:layout_marginEnd="140dp"  
    android:layout_marginBottom="130dp"  
    android:gravity="left"  
    android:text="Heat switch:"  
    android:textColor="@color/white"  
    android:textSize="20dp"  
    android:textStyle="bold" />
```

```
<TextView
```

```
    android:id="@+id/light"  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:layout_marginStart="280dp"  
    android:layout_marginTop="450dp"
```

```
android:layout_marginEnd="0dp"  
android:layout_marginBottom="130dp"  
android:gravity="left"  
android:text="Light switch:"  
android:textColor="@color/white"  
android:textSize="20dp"  
android:textStyle="bold" />
```

```
<Switch
```

```
android:layout_width="wrap_content"  
android:layout_height="wrap_content"  
android:layout_centerInParent="false"  
android:layout_marginStart="20dp"  
android:layout_marginTop="400dp"  
android:layout_marginEnd="290dp"  
android:layout_marginBottom="130dp"  
android:thumb="@drawable/thumb"  
android:track="@drawable/track" />
```

```
<Switch
```

```
android:layout_width="wrap_content"  
android:layout_height="wrap_content"  
android:layout_centerInParent="false"  
android:layout_marginStart="20dp"  
android:layout_marginTop="500dp"
```

```
android:layout_marginEnd="290dp"  
android:layout_marginBottom="130dp"  
android:thumb="@drawable/thumb"  
android:track="@drawable/track" />
```

<Switch

```
android:layout_width="wrap_content"  
android:layout_height="wrap_content"  
android:layout_centerInParent="false"  
android:layout_marginStart="170dp"  
android:layout_marginTop="400dp"  
android:layout_marginEnd="290dp"  
android:layout_marginBottom="130dp"  
android:thumb="@drawable/thumb"  
android:track="@drawable/track"
```

/>

<Switch

```
android:layout_width="wrap_content"  
android:layout_height="wrap_content"  
android:layout_centerInParent="false"  
android:layout_marginStart="170dp"  
android:layout_marginTop="500dp"  
android:layout_marginEnd="290dp"  
android:layout_marginBottom="130dp"
```

```
    android:thumb="@drawable/thumb"  
    android:track="@drawable/track"  
/>  
<Switch  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:layout_centerInParent="false"  
    android:layout_marginStart="300dp"  
    android:layout_marginTop="400dp"  
    android:layout_marginEnd="290dp"  
    android:layout_marginBottom="130dp"  
    android:thumb="@drawable/thumb"  
    android:track="@drawable/track"  
/>
```

```
<Switch  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:layout_centerInParent="false"  
    android:layout_marginStart="300dp"  
    android:layout_marginTop="500dp"  
    android:layout_marginEnd="290dp"  
    android:layout_marginBottom="130dp"  
    android:thumb="@drawable/thumb"
```



```
android:track="@drawable/track"/>
```

```
<TextView
```

```
android:id="@+id/pump"
```

```
android:layout_width="match_parent"
```

```
android:layout_height="wrap_content"
```

```
android:layout_marginStart="5dp"
```

```
android:layout_marginTop="550dp"
```

```
android:layout_marginEnd="340dp"
```

```
android:layout_marginBottom="130dp"
```

```
android:gravity="left"
```

```
android:text="Pump:"
```

```
android:textColor="@color/white"
```

```
android:textSize="20dp"
```

```
android:textStyle="bold" />
```

```
<TextView
```

```
android:id="@+id/fan1"
```

```
android:layout_width="match_parent"
```

```
android:layout_height="wrap_content"
```

```
android:layout_marginStart="5dp"
```

```
android:layout_marginTop="630dp"
```

```
android:layout_marginEnd="340dp"
```

```
android:layout_marginBottom="130dp"
```

```
android:gravity="left"
```

```
android:text="Fan1:"  
android:textColor="@color/white"  
android:textSize="20dp"  
android:textStyle="bold" />
```

```
<TextView
```

```
    android:id="@+id/lightintensity"  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:layout_marginStart="250dp"  
    android:layout_marginTop="550dp"  
    android:layout_marginEnd="10dp"  
    android:layout_marginBottom="130dp"  
    android:gravity="left"  
    android:text="Light intensity:"  
    android:textColor="@color/white"  
    android:textSize="20dp"  
    android:textStyle="bold" />
```

```
<TextView
```

```
    android:id="@+id/fan2"  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:layout_marginStart="250dp"  
    android:layout_marginTop="630dp"
```

```
android:layout_marginEnd="10dp"  
android:layout_marginBottom="130dp"  
android:gravity="left"  
android:text="Fan2:"  
android:textColor="@color/white"  
android:textSize="20dp"  
android:textStyle="bold" />
```

```
<SeekBar
```

```
android:id="@+id/seekBarpump"  
android:layout_width="180dp"  
android:layout_height="wrap_content"  
android:layout_marginTop="590dp"  
android:progressTint="#000000" />
```

```
<SeekBar
```

```
android:id="@+id/seekBarfan1 "  
android:layout_width="180dp"  
android:layout_height="wrap_content"  
android:layout_marginTop="670dp"  
android:progressTint="#000000" />
```

```
<SeekBar
```

```
android:id="@+id/seekBarlight"  
android:layout_width="180dp"  
android:layout_marginStart="250dp"
```

```
android:layout_height="wrap_content"  
android:layout_marginTop="590dp"  
android:progressTint="#000000" />
```

```
<SeekBar
```

```
android:id="@+id/seekBarfan2"  
android:layout_width="180dp"  
android:layout_marginStart="250dp"  
android:layout_height="wrap_content"  
android:layout_marginTop="670dp"  
android:progressTint="#000000" />
```

```
<Button
```

```
android:id="@+id/button2"  
android:layout_width="90dp"  
android:layout_height="90dp"  
android:layout_alignParentEnd="false"  
android:layout_centerHorizontal="true"  
android:layout_marginStart="280dp"  
android:layout_marginTop="40dp"  
android:layout_marginEnd="20dp"  
android:layout_marginBottom="20dp"  
android:background="@drawable/custom_button"  
android:text="Stop buzzer!"  
android:textColor="@color/white"
```

```

        android:textSize="19dp" />
<com.google.android.material.button.MaterialButton
    android:id="@+id/signupbtn"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_centerHorizontal="true"
    android:layout_marginStart="280dp"
    android:layout_marginTop="750dp"
    android:layout_marginEnd="20dp"
    android:layout_marginBottom="20dp"
    android:backgroundTint="#30FFFFFF"
    android:text="REPORT"
    android:textColor="@color/white"
    android:textSize="15dp" />
</RelativeLayout>
package com.example.greenverticalfarming;
import androidx.appcompat.app.AppCompatActivity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
public class controll extends AppCompatActivity {
    private Button reportbttn;

```

```

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_controll);
    reportbtnn = (Button) findViewById(R.id.reportbtnn);
    reportbtnn.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            Intent intent = new Intent(controll.this, reportt.class);
            startActivity(intent);
        }
    });
}
}

```

```

<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@drawable/bgb"
    tools:context=".MainActivity">
    <TextView

```

```
android:id="@+id/report"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_centerHorizontal="true"
android:layout_marginStart="25dp"
android:layout_marginTop="25dp"
android:layout_marginEnd="25dp"
android:layout_marginBottom="25dp"
android:gravity="center"
android:text="REPORTS"
android:textColor="@color/white"
android:textSize="35dp"
android:textStyle="bold" />
```

```
<TableLayout
```

```
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_marginStart="10dp"
android:layout_marginTop="150dp"
android:layout_marginEnd="10dp"
android:layout_marginBottom="10dp"
android:background="#30FFFFFF">
```

```
<!-- Table Heading -->
```

```
<TableRow
```

```
android:background="#144A16">
```

```
<TextView
```

```
    android:layout_width="0dp"
```

```
    android:layout_height="wrap_content"
```

```
    android:layout_gravity="center_horizontal"
```

```
    android:layout_weight="4"
```

```
    android:padding="10sp"
```

```
    android:text="@string/col1"
```

```
    android:textColor="@color/white"
```

```
    android:textSize="17sp"
```

```
    android:textStyle="bold" />
```

```
<TextView
```

```
    android:layout_width="0dp"
```

```
    android:layout_height="wrap_content"
```

```
    android:layout_gravity="center_horizontal"
```

```
    android:layout_weight="4"
```

```
    android:padding="10sp"
```

```
    android:text="@string/col2"
```

```
    android:textColor="@color/white"
```

```
    android:textSize="17sp"
```

```
    android:textStyle="bold" />
```

```
<TextView
```

```
    android:layout_width="0dp"
```



```
android:layout_height="wrap_content"  
android:layout_gravity="center_horizontal"  
android:layout_weight="4"  
android:padding="10sp"  
android:text="@string/col3"  
android:textColor="@color/white"  
android:textSize="17sp"  
android:textStyle="bold" />
```

```
<TextView
```

```
    android:layout_width="0dp"  
    android:layout_height="wrap_content"  
    android:layout_gravity="center_horizontal"  
    android:layout_weight="4"  
    android:padding="10sp"  
    android:text="@string/col4"  
    android:textColor="@color/white"  
    android:textSize="17sp"  
    android:textStyle="bold" />
```

```
<TextView
```

```
    android:layout_width="0dp"  
    android:layout_height="wrap_content"  
    android:layout_gravity="center_horizontal"  
    android:layout_weight="4"
```

```

        android:padding="10sp"
        android:text="@string/col5"
        android:textColor="@color/white"
        android:textSize="17sp"
        android:textStyle="bold" />
</TableRow>
android:>
<!-- Table Data-->
<TableRow>
    <TextView
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_gravity="center_horizontal"
        android:layout_weight="1"
        android:padding="10sp"
        android:text="@string/row1"
        android:textColor="@color/white"
        android:textSize="20sp"
        android:textStyle="bold" />
</TableRow>
<TableRow>
    <TextView
        android:layout_width="0dp"

```

```
    android:layout_height="wrap_content"
    android:layout_gravity="center_horizontal"
    android:layout_weight="4"
    android:padding="10sp"
    android:text="@string/row2"
    android:textColor="@color/white"
    android:textSize="20sp"
    android:textStyle="bold" />
```

```
</TableRow>
```

```
<TableRow>
```

```
    <TextView
```

```
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_gravity="center_horizontal"
        android:layout_weight="4"
        android:padding="10sp"
        android:text="@string/row3"
        android:textColor="@color/white"
        android:textSize="20sp"
        android:textStyle="bold" />
```

```
</TableRow>
```

```
<TableRow>
```

```
    <TextView
```

```
android:layout_width="0dp"  
android:layout_height="wrap_content"  
android:layout_gravity="center_horizontal"  
android:layout_weight="4"  
android:padding="10sp"  
android:text="@string/row4"  
android:textColor="@color/white"  
android:textSize="20sp"  
android:textStyle="bold" />
```

```
</TableRow>
```

```
<TableRow>
```

```
<TextView
```

```
android:layout_width="0dp"  
android:layout_height="wrap_content"  
android:layout_gravity="center_horizontal"  
android:layout_weight="4"  
android:padding="10sp"  
android:text="@string/row5"  
android:textColor="@color/white"  
android:textSize="20sp"  
android:textStyle="bold" />
```

```
</TableRow>
```

```
<TableRow>
```

```
<TextView
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_gravity="center_horizontal"
    android:layout_weight="4"
    android:padding="10sp"
    android:text="@string/row6"
    android:textColor="@color/white"
    android:textSize="20sp"
    android:textStyle="bold" />
```

```
</TableRow>
```

```
<TableRow>
```

```
<TextView
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_gravity="center_horizontal"
    android:layout_weight="4"
    android:padding="10sp"
    android:text="@string/row7"
    android:textColor="@color/white"
    android:textSize="20sp"
    android:textStyle="bold" />
```

```
</TableRow>
```

```
</TableLayout>
```

```
<Button
```

```
    android:id="@+id/backbtnn"  
    android:layout_width="165dp"  
    android:layout_height="wrap_content"  
    android:layout_centerHorizontal="false"  
    android:layout_marginStart="10dp"  
    android:layout_marginTop="550dp"  
    android:layout_marginEnd="20dp"  
    android:layout_marginBottom="20dp"  
    android:backgroundTint="#30FFFFFF"  
    android:text="BACK"  
    android:textColor="@color/white"  
    android:textSize="25dp">
```

```
</Button>
```

```
<Button
```

```
    android:id="@+id/logoutbtnn"  
    android:layout_width="165dp"  
    android:layout_height="wrap_content"  
    android:layout_centerHorizontal="false"  
    android:layout_marginStart="240dp"  
    android:layout_marginTop="550dp"  
    android:layout_marginEnd="200dp"
```

```
    android:layout_marginBottom="20dp"
    android:backgroundTint="#30FFFFFF"
    android:text="LOG OUT"
    android:textColor="@color/white"
    android:textSize="25dp">
</Button>
```

```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_alignParentBottom="true"
    android:gravity="center">
</LinearLayout>
```

```
</RelativeLayout>
```

```
package com.example.greenverticalfarming;
import androidx.appcompat.app.AppCompatActivity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
public class reportt extends AppCompatActivity {
    private Button backbtnn;
    private Button logoutbtnn;
```

```

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_reportt);
    backbtnn = (Button) findViewById(R.id.backbtnn);
    logoutbtnn = (Button) findViewById(R.id.logoutbtnn);
    backbtnn.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            Intent intent = new Intent(reportt.this, controll.class);
            startActivity(intent);
        }
    });
    logoutbtnn.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            Intent intent = new Intent(reportt.this, MainActivity.class)

```


II. Engineering Standard (IEEE)

The Institute of Electrical and Electronics Engineers Standards Association (IEEE SA) is an operating unit within IEEE that develops global standards in a broad range of industries, including: power and energy, artificial intelligence systems, internet of things, consumer technology and consumer electronics, biomedical and health care, learning technology, information technology and robotics, telecommunication and home automation, automotive, transportation, home automation, nanotechnology, information assurance, emerging technologies, and many more.

IEEE SA has developed standards for over a century, through a program that offers balance, openness, fair procedures, and consensus. Technical experts from all over the world participate in the development of IEEE standards.^[1]

This project follow the criteria of many sub-standards based on IEEE standardization policy such as: (i) **Cloud Computing (IEEE P2301)**, (ii) **Software Life Cycle (IEEE 1074)**, (iii) **Sensors (IEEE 802.15.4)**, (iv) **Raspberry based Bluetooth (IEEE 802.15.2)**, and (v) **Common Gateway based Wi-Fi (IEEE 802.11)**.

I. Cloud Computing (IEEE P2301)

IEEE P2301 defines essential topology, protocols, functionality, and governance required for reliable cloud-to-cloud interoperability and federation. The standard will help build an economy of scale among cloud product and service providers that remains transparent to users and applications.

With a dynamic infrastructure that supports evolving cloud business models, IEEE P2302 is an ideal platform for fostering growth and improving competitiveness. It will also address fundamental, transparent interoperability and federation much in the way SS7/IN did for the global telephony system, and naming and routing protocols did for the Internet [31].

II. **Software Life Cycle (IEEE 1074)**

IEEE 1074 provides a process for creating a software life cycle process (SLCP). The SLCP is defined as the project-specific description of the process that is based on a project's software life cycle (SLC) and the integral and project management processes used by the organization. These integral processes include configuration management, metrics, quality assurance, risk reduction, and the acts of estimating, planning, and training. It is primarily the responsibility of the project manager and the process architect for a given software project to create the SLCP [32].

This methodology begins with the selection of an appropriate software life cycle model (SLCM) for use on the specific project. It continues through the creation of the SLC, using the selected SLCM and the activities described in Table 3-1. The methodology concludes with the augmentation of the SLC with an organization's support processes to create the SLCP. The activities described in the 1074 mapping table cover the entire life cycle of a software project,

from concept exploration through the eventual retirement of the software system. 1074 does not address no software activities, such as contracting, purchasing, or hardware development. It also does not mandate the use of a specific SLCM. 1074 presumes that the project manager and process architect are already familiar with a variety of SLCMs, with the criteria for choosing among them, and with the criteria for determining the attributes and constraints of the desired end system and the development environment that affects this selection [32].

III. Sensors (IEEE 802.15.4)

IEEE standard 802.15.4 intends to offer the fundamental lower network layers of a type of wireless personal area network (WPAN) which focuses on low-cost, low-speed ubiquitous communication between devices. It can be contrasted with other approaches, such as Wi-Fi, which offer more bandwidth and requires more power. The emphasis is on very low cost communication of nearby devices with little to no underlying infrastructure, intending to exploit this to

lower power consumption even more. The basic framework conceives a 10-meter communications range with a transfer rate of 250Kbps. Tradeoffs are possible to favor more radically embedded devices with even lower power requirements, through the definition of not one, but several physical layers. Lower transfer rates of 20 and 40Kbps were initially defined, with the 100Kbps rate being added in the current revision.

IEEE 802.15.4 has been widely accepted as the de facto standard for wireless sensor networks (WSNs). However, as in their current solutions for medium access control (MAC) sub-layer protocols, channel efficiency has a margin for improvement [33].

Key 802.15.4 features include: (i) real-time suitability by reservation of Guaranteed Time Slots (GTS), (ii) collision avoidance through CSMA/CA, (iii) integrated support for secure communications, (iv) power management functions such as link speed/quality and energy detection, (v) Support for time and data rate sensitive applications because of its ability to operate either as

CSMA/CA or TDMA access modes. The TDMA mode of operation is supported via the GTS feature of the standard, and (vi) IEEE 802.15.4-conformant devices may use one of three possible frequency bands for operation (868/915/2450 MHz) [34].

IV. Raspberry based Bluetooth (IEEE 802.15.2)

Task group two addresses the coexistence of wireless personal area networks (WPAN) with other wireless devices operating in unlicensed frequency bands such as wireless local area networks (WLAN). The IEEE 802.15.2-2003 standard was published in 2003[3] and task group two went into "hibernation" [35].

V. Common Gateway based Wi-Fi (IEEE 802.11).

IEEE 802.11 is part of the IEEE 802 set of local area network (LAN) technical standards, and specifies the set of media access control (MAC) and physical layer (PHY) protocols for implementing wireless local area network (WLAN) computer communication. The standard and amendments provide the basis for wireless network products using the Wi-Fi brand and are the world's most widely used

wireless computer networking standards. IEEE 802.11 is used in most home and office networks to allow laptops, printers, smartphones, and other devices to communicate with each other and access the Internet without connecting wires. [36]

IEEE 802.11 uses various frequencies including, but not limited to, **2.4 GHz, 5 GHz, 6 GHz, and 60 GHz** frequency bands. Although IEEE 802.11 specifications list channels that might be used, the radio frequency spectrum availability allowed varies significantly by regulatory domain. The protocols are typically used in conjunction with IEEE 802.2, and are designed to interwork seamlessly with Ethernet, and are very often used to carry Internet Protocol traffic [36].

III. Economically Feasibility Study

The feasibility study relates to the study of a new project, and strategic planning is to study the best areas in which the organization can operate in the coming years and how it can compete in these areas. Strategic planning is more general than a feasibility study because strategic planning illustrates the best areas in which the institution can operate in the coming years, and the feasibility study is about studying a specific project or projects.

This project had been planned and optimized according to the common relation between cost and performance analysis to realize the best performance with lower available cost, however there are three versions for the overall vertical farming criteria that analyzed in common study as discussed in **Table 4**.

Table 4: Project Development Releases (Economically Feasibility Study)

Release	Green V.01	Green V.02	Green V.03	Benchmark
Technology	ARDUINO UNO	ARDUINO MEGA + BLUETOOTH Mobile App.	Raspberry + Web Server + Mobile App.	Raspberry + Web Server + Mobile App.
Technique	Soil Agriculture	Soil Agriculture	Hydroponic Agriculture	Hydroponic Agriculture
Planned Cost	120 \$	370 \$	1000 \$	2500 \$
Real Cost	100 \$	300 \$	700 \$	1500 \$
Reference Cost	135 \$	400 \$	870 \$	2000 \$

According to the above mentioned study, this project had been implemented based on the standards and rules for IOT technologies (i.e. web servers and mobile applications) with Raspberry as a control system (Green V.03). The total cost is 700\$ is less than the 50% of the benchmark cost that's mean the performance of this technology was verified economically and environmentally.