COMM018 - Internet of Things

Coursework 2 – Mini project

IoT Based Phytotron environmental monitoring and control system.

Rivindi Kalasing Ekanayake

24012043

MSc Advanced Computer Science

Dr Hamzah AlZubi

Submission on 30th May 2025

Abstract

The objective of this project is to advance precision agriculture and controlled plant growth settings by designing and developing an Internet of Things (IoT)-based phytotron environmental monitoring and control system. Real-time monitoring and automated control of critical growth parameters are made possible by the system's integration of affordable microcontroller platforms like the NodeMCU ESP8266 with environmental sensors like the DHT22, capacitive soil moisture sensors, and LDRs. A web-based program supports administrative and research functions by enabling remote access, user management, graphical data visualisation, and environment customisation options. For data processing, storage, and sensor-actuator coordination, cloud systems like Firebase and ThingSpeak were used. Data security is likewise covered by the implementation, guaranteeing confidentiality and integrity throughout the storage and communication levels. Although the prototype successfully illustrates the essential features, development was constrained by the availability of hardware, the capacity for temporary hosting, and the limitations of living and working on campus. Notwithstanding these drawbacks, the system provides a solid basis for expanding uses in commercial greenhouses, urban agriculture, and research labs.

Table of Content

List of figures and Tables

Chapter 01 – Introduction

1.1.Background of the Study

Botany researchers face challenges in using greenhouses, plant houses, or glass houses due to pesticide effects and lack of environmental variables. A phytotron, a controlled environment, has emerged as a solution. Automated control systems improve data collection and specimen observation, but COVID-19 has limited access. Miniature growing systems are often used, but researchers lack sufficient technical equipment or systems for these activities.

1.2.Motivation of the study

A miniature system is a laboratory setup designed to cultivate plants, providing necessary conditions for their growth. These systems are primarily used for agricultural research and test situations, typically built using metal or plastic pipes. They allow multiple plants to be cultivated simultaneously and can be cultivated using seeds. The primary objective of miniaturized systems is to provide controlled settings for plants that are challenging to replicate in open or ground areas. The growth of plants is influenced by soil moisture, temperature, humidity, light intensity, soil nutrients, and atmospheric gases. Miniature systems produce artificial environments for plants, allowing for manual adjustments to soil moisture, temperature, humidity, and light intensity. Institutions, particularly Indian companies, are investing in designing and building phytotrons for research purposes.

1.3. Project Description

1.3.1. Problem statement

Miniature cabinets, used for agricultural research, lack an automatic control system for monitoring plant growth under varying environmental conditions. Existing cabinets require physical proximity for remote operation, making them unsuitable for maintaining certain environmental conditions. Miniature systems, typically built using metal or plastic pipes, provide controlled conditions for plants, allowing multiple plants to be cultivated simultaneously. Seeds can be used to some extent within the system, but these mechanisms do not affect crop production. The primary objective of miniaturized systems is to provide controlled settings for plants that are difficult to replicate in open or ground areas. Phytotrons, specialized facilities used in plant science research, simulate various environmental conditions, such as temperature, humidity, light intensity, and CO2 concentration, in an enclosed growth chamber.

1.3.2. Proposed Solution

This project introduces an **Internet of Things (IoT)-based Phytotron Environmental Monitoring and Control System** that offers a fully working, remotely accessible smart micro environment to assist researchers and academics. The system integrates multiple sensors, including a capacitive soil moisture sensor, a DHT22 for temperature and humidity, an LDR for light intensity, and a DS18B20 for water temperature, to monitor critical environmental parameters. The NodeMCU ESP8266 microcontroller is used in a reliable and user-friendly research platform for plant growth studies in phytotrons. Its affordability and Wi-Fi compatibility make it an ideal choice for optimal growing conditions, ensuring real-time environmental data collection and management.

1.3.3. Objectives of the Project

The project aims to address challenges faced by research organizations and universities during the COVID-19 pandemic, including physical restrictions and institutional closures. It aims to improve remote-control systems, address issues like insufficient maintenance, and introduce modern technologies like cloud-based systems and microcontroller technology for remote data collection and control in phytotrons. The project also aims to meet the needs of researchers working overseas, providing an online-operating system for controlling CO2 levels and light. The project will build hardware systems for phytotrons, create web-based systems for easy navigation, automate data control, and make cloud-based interligent systems accessible.

1.3.4. Potential applications of the project

The IoT phytotron system has a lot of promise for a range of practical and scientific uses. It makes it possible to precisely simulate climatic conditions in agricultural and climate resilience research, which helps to design crop types that are resilient to climate stressors including heat, humidity, and drought. Phytotrons, for example, have been used to evaluate how climate change affects vegetable crop illnesses and offer insights into disease control methods in different temperature and CO_2 environments. The technology makes it easier to test various growth parameters, fertilisers, and irrigation techniques in controlled crop growth experiments by providing stable and adaptable conditions. Such study, which focusses on agricultural issues in the Southeast United

States, is supported by facilities such as the NC State Phytotron. In botany, the technique is also useful for genetic and physiological research, enabling scientists to examine plant characteristics in controlled environments to gain a better understanding of genetic expression and adaptability. Additionally, the technology facilitates vertical farming and urban agriculture, providing scalable, automated, and remotely controlled solutions that are perfect for producing food sustainably in urban environments with limited space. Vertical farming innovations, like those in Germany, show how phytotron technology may be integrated to maximise plant cultivation within specific environmental constraints.

1.3.5. Project prototype features

To provide a smooth and intelligent plant monitoring and control system, the suggested prototype combines hardware-level IoT devices, a centralised cloud database, and a responsive web-based application for interaction. Real-time alarms, data visualisation, remote experimentation, and safe phytotron operation management are all made possible by this technology.

IoT Device

The prototype's IoT hardware layer is essential for detecting, managing, and conveying important environmental elements. It has the following features:

- Transfer data
- Manage environmental conditions manually
- Monitor temperature, humidity, light, soil moisture
- Control LED lighting
- Operate fans/vents via stepper motors

Web Application

To monitor, regulate, and analyse phytotron settings, researchers, educators, and farm managers can use the system's fully integrated web-based platform as their primary interface.

- Track environmental conditions
- Real-time sensor data dashboard
- View reports and data trends
- User authentication and access management

– Responsive design for all devices

<u>Database Integration</u>

Data permanence, accessibility, and integrity are guaranteed across all modules via a strong cloud-based database layer, which mainly uses Firebase Realtime Database.

– Centralized data storage
– Backup and recovery
– Store experiment metadata
– Enable data visualization

1.3.6. Summary of the project description

The system is designed to provide real-time monitoring of environmental conditions within the miniature phytotron by displaying sensor values such as temperature, humidity, and soil moisture through a web application. Users can manually control actuators, including fans and water pumps, directly via the web interface. Additionally, the system supports automated control, enabling actuators to operate independently when sensor readings deviate from predefined thresholds. All collected data is securely stored in a cloud service, ensuring accessibility and enabling long-term analysis and research. Hardware component of this project aims to provide a smart, remotely accessible phytotron system that supports researchers and academics by integrating Internet of Things (IoT) technology. Furthermore, project system enables real-time environmental monitoring and control using a cloud-based platform developed with PHP and MySQL, and microcontroller systems programmed in Arduino. The goal is to offer a user-friendly, reliable, and efficient research environment.

Chapter 02 – Literature Review

2.1. Introduction

Most of the research and tests on plants in the agriculture research area involve miniature growing systems. The most important problem is that researchers do not have sufficient technical equipment or a system for conducting their research activities involving miniature growing cabinets. All functions and data-gathering processes manually and manual data gathering process is requiring more humans' resources and also the cost. In some situations, physical access to that kind of miniature system is not possible for all researchers and human errors are occurring when measuring the values since the observations are done by humans. There is no automated system yet developed to automatically control and monitor the system parameters within the miniature cabinets. As a result, researchers are encountering significant challenges in monitoring the growth of plants under varying environmental conditions. This chapter is with in-depth study regarding the IoT in smart agriculture and controlled environments, phytotron and sensor technology, component study used for IoT integration in phytotron and finally market trends and similar projects.

Further, this chapter will point out the issues that have found when studying similar research. Also, how researchers could be able proposed to accomplish the task by adapting and improvising using the existing technologies that they have used for their research activities.

2.2. IoT in smart agriculture and controlled environments

The Internet of Things (IoT) has transformed agriculture by enabling automated systems that monitor and control farming parameters using a network of smart sensors and actuators. By providing real-time monitoring and precise management over a variety of farming characteristics, the Internet of Things' (IoT) integration into agriculture, commonly known as "Smart Farming," has completely transformed conventional agricultural techniques. IoT technologies improve crop management decision-making by making it easier to gather and analyse data about temperature, humidity, soil moisture, and other environmental variables. In order to address the issues brought on by resource scarcity, climate change, and the rising need for food worldwide, this technical breakthrough is essential.

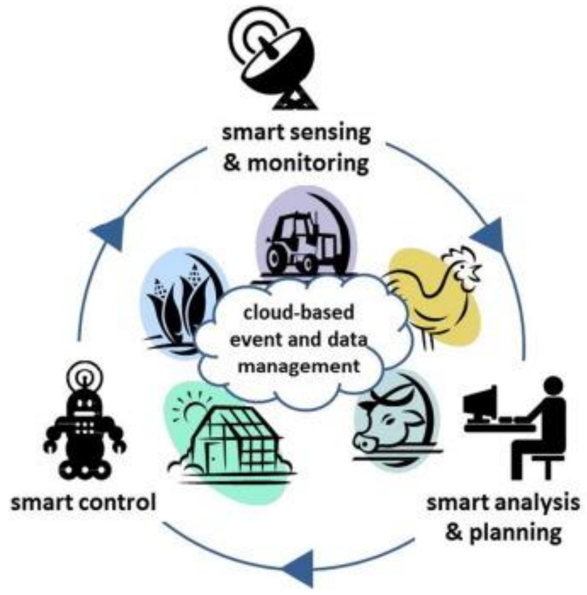

smart sensing & monitoring
cloud-based event and data management
smart control
smart analysis & planning

Figure 1- Concept diagram of Smart Agriculture- (Sjaak Wolfert, Lan Ge,Cor Verdouw, Marc-Jeroen Bogaardt, 2017)

The potential of IoT to revolutionise agricultural operations through increased sustainability and efficiency is highlighted in a thorough assessment by Wolfert on Big Data in Smart Farming (Sjaak Wolfert, Lan Ge,Cor Verdouw, Marc-Jeroen Bogaardt, 2017). The study highlights how smart sensors and gadgets generate enormous volumes of data, enabling hitherto unheard-of decision-making powers. The authors also talk about the ramifications of big data in agriculture, pointing out that it can cause major changes in the roles and power dynamics of both traditional and non-traditional participants in the industry. The creation of appropriate economic models and governance concerns, such as data ownership, privacy, and security, are noted as crucial topics in need of additional study.

IoT applications have shown considerable advantages in controlled environments like greenhouses and phytotrons. Increased agricultural yields and resource efficiency result from the preservation of ideal growth conditions made possible by the deployment of sensor networks and automated control systems. The efficiency of wireless sensor networks in monitoring and managing greenhouse settings is demonstrated by several studies. An automated irrigation system designed to maximise crop water utilisation is described in their paper (Joaquin Gutierrez, Juan Francisco Villa Medina ,Alejandra Nieto-Garibay,Miguel Angel Porta-Gándara, 2014). The system consists of a dispersed wireless network of temperature, moisture, and soil sensors positioned in the plant roots. A gateway unit sends data to a web application, manages sensor data, and activates actuators. To regulate the amount of water, a microcontroller-based gateway was designed with an algorithm that included threshold values for soil moisture and temperature. When compared to conventional watering methods, the photovoltaic panel-powered system showed water savings of up to 90% (Joaquin Gutierrez, Juan Francisco Villa Medina ,Alejandra Nieto-Garibay,Miguel Angel Porta-Gándara, 2014).

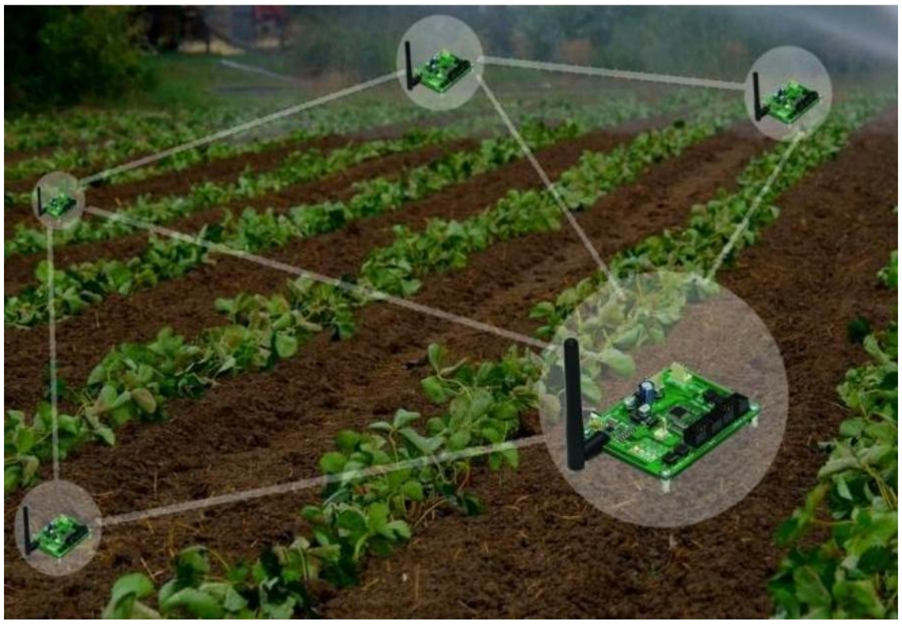


Figure 2- Irrigation system use wireless Technology - (Joaquin Gutierrez, Juan Francisco Villa Medina ,Alejandra Nieto-Garibay,Miguel Angel Porta-Gándara, 2014)

When considering the area of smart agriculture, it highly discussed smart farming. Either in smart farming or smart agriculture, IoT can be taken as an important point. "New inventive Internet of Things systems are inclining to issues of Agriculture production and escalating the worth, supportability, and cost-effectiveness of rural agricultural." (Odoi-Lartey, B., & Danso, E., 2018). In the smart farming or smart agricultural field, there are two main parts as sensors and the controlling system. (Anushree,2018) These controlling systems mainly consist of actuators. A research paper by Aditi Mehta and Sanjay Patel Department of Information Technology, Gandhinagar, India mentioned that wherever automation had been implemented and human beings had been replaced by automatic machinery, the yield has been improved in agriculture. Hence there is a need to implement modern science and technology in the agriculture sector for increasing the yield (Vineela MT,2018).

Despite these advancements, there are still barriers preventing the widespread use of IoT in agriculture. Issues include high implementation costs, a lack of standardised protocols, data privacy concerns, and the need for technical know-how, particularly in developing countries, limit the scalability of IoT solutions. Stakeholders, including legislators, researchers, technology developers, and the farming community, must work together to address these issues.

In conclusion, there is revolutionary potential for improving productivity, sustainability, and resource efficiency through the incorporation of IoT into controlled settings and agriculture. Harnessing the full benefits of IoT in the agriculture sector requires ongoing research and development, as well as legislation that support it and capacity-building programs.

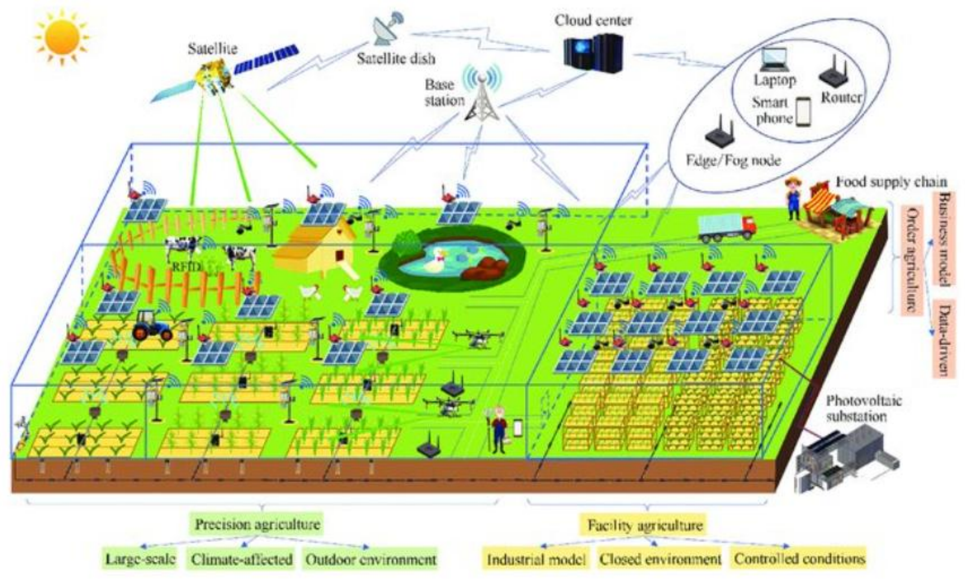


Figure 3-Smart Agriculture- (Xing Yang,Lei Shu, Jianing Chen, Mohamed Amine Ferrag, 2021)

2.3. Phytotron Technology

2.3.1. Miniature systems

A research paper of Masato Futagawa Toyohashi University, Japan revealed that miniature systems can build and set up in a laboratory for growing plants with all of the conditions that want for the plant. (ineela MT,2018). In the agricultural research field, miniature systems use as technical equipment for their test cases. Small-scale, task-specific subsystems or modules that combine to create an integrated IoT solution are referred to as miniature systems in the context of the Internet of Things. These systems create ideal settings for plant study and growth by fine-tuning and simulating various environmental conditions in a phytotron. Every little system is in charge of a certain operational or environmental factor. These comprise subsystems for air quality, light intensity, soil moisture, temperature, and humidity monitoring and control. For example, soil moisture sensors use solenoid valves to initiate automatic watering, while temperature and humidity sensors, such as the DHT22, enable feedback-based environmental control. Likewise, LDRs and programmable LEDs improve the accuracy of experimental plant growth stages by simulating natural lighting conditions according to photoperiodic requirements. Thermoregulation in the phytotron environment is supported by stepper motor-driven ventilation systems, which further control airflow and heat based on sensor input.

2.3.2. Phytotron Systems

An advanced controlled environment facility called a phytotron system which is used to research plant growth and development in meticulously regulated settings. These consist of atmospheric composition, photoperiod, light intensity, temperature, and humidity. Researchers can model and examine the effects of various environmental factors on plant physiology, morphology, and productivity thanks to these capabilities. Phytotrons have historically been essential to scholarly research, especially in the fields of plant biology, genetics, and crop breeding initiatives. Conventional phytotrons, however, usually require a lot of resources. The United States Department of Agriculture (USDA) claims that traditional phytotron setups frequently involve a great deal of manual labour, with staff members constantly monitoring and adjusting the environment to maintain the desired parameters based on the needs of the researcher (USDA, 2022). This raises operating expenses and raises the possibility of human error-related discrepancies.

Figure 4- NC State University Phytotron- (Phytotron, n.d.)

Researchers and developers have started creating intelligent phytotron systems in order to get over these obstacles. These contemporary systems combine automation technology and Internet of Things (IoT) devices to provide remote control and real-time environmental monitoring. In order to collect real-time data on temperature, soil moisture, humidity, and light, research article of Jaiswal (Jaiswal, A., Niwas, R., & Kumar, A. , 2020) presented a smart phytotron architecture that makes use of Internet of Things-based sensors. Actuators are then used to transfer this data to a cloud platform for analysis and environmental control. By using these technologies, less manual management is required, and precise modifications may be made to ensure ideal plant development conditions.

Energy efficiency is perhaps another noteworthy benefit of smart phytotron systems. Researchers can optimise the use of irrigation schedules, HVAC (heating, ventilation, and air conditioning) systems, and lights by combining IoT and AI. According to one study, a smart phytotron with temperature and adaptive lighting controls might save up to 40% on energy use without sacrificing plant health or research quality. at the long run, this makes these systems more economical and sustainable, especially at major research institutions (Singh, A., Soni, R., & Tiwari, A. , 2022).

In conclusion, a major advancement in the field of controlled environment agriculture has been made with the incorporation of IoT and AI into phytotron systems. These intelligent solutions facilitate data-driven decision-making, lower labor and energy expenses, and offer improved control over environmental conditions. It is anticipated that as technology advances, smart phytotrons' scalability, affordability, and accessibility will increase, opening the door to more effective and sustainable plant research.

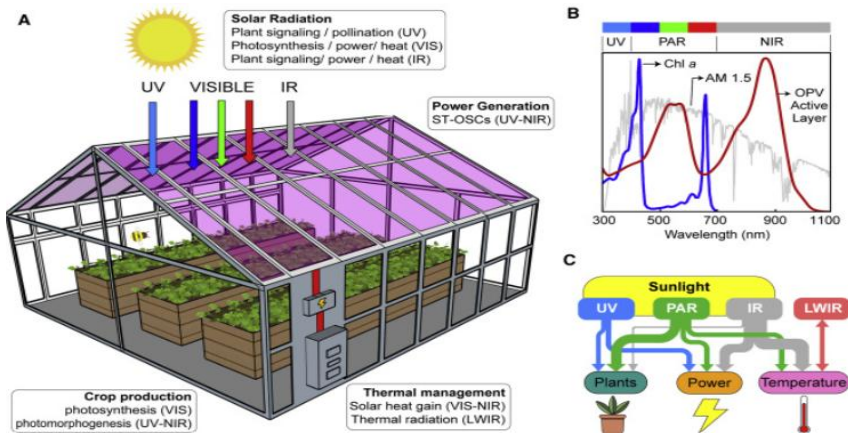


Figure 5-NC State University Phytotron- (Phytotron, n.d.)

2.3.3. Components and Technologies Used for IoT Integration in existing Miniature Systems

2.3.3.1. Soil moisture sensors

These sensors are used to calculate the content of the soil water level in the volume unit. Due to the plant uptake and evaporation which leads to the loss of moisture, this sensor is capable of measuring its value. Also, it can analyse the desired soil moisture contents for various species of plants. It can enhance the soil moisture content by monitoring the irrigation in the miniature system.

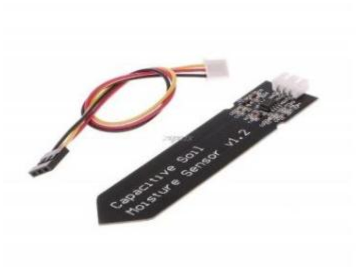


Figure 6- Capacitive soil moisture sensor

There are two different types of soil moistures. Resistive soil moistures and capacitive soil moistures. Rather than resistive moisture **sensing**, capacitive soil moisture **sensors can be sensed the soil moisture** levels more accurately. This sensor is made of corrosion-resistant material. Because of that, it can prevent correction and long service life.

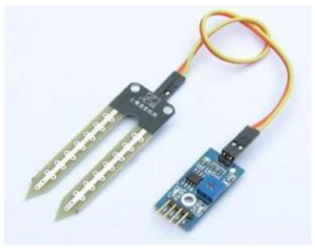


Figure 7-Resistive soil moisture sensor

2.3.3.2. Humidity and Temperature Sensor

The amount of water vapor includes in the air referred to as Humidity. Humidity sensors can be detected the relative humidity in the surrounding environment. To find out the amount of moisture present in the air, sensors have been used as a capacitive method (Sarkar,2018). The measurement of the temperature in the miniature system is done using the principle of a thermistor. These sensors provide long-term stability and high reliability (Mahir,2015). In the existing miniature systems, the temperature has been gathered manually by thermometers or low-level temperature meters.

2.3.3.3. Actuators – Fan

To control the temperature and the humidity levels, Direct Current (DC) fans have been used in many systems. All these systems had focused on off the fan and not considered the speed of the fan.

2.3.3.4. Light Emitting Diode (LED)

LED is a semiconductor gadget that produces light when an electric flow is gone through it. Light is created when the particles that convey the current combination inside the semiconductor material. The project Plant Talk has used LED lights to save energy. In this project, they developed a system to switch light sources between LED and sunlight based on the intensity of sunlight radiation to investigate the plant growth in a hydroponic system (LD Van,2029). A project by Kent Kobayashi, Teresita Amore, and Michelle Lazaro of the Tropical Plant & Soil Sciences Department in the United State of America presented that Light-emitting diodes (LEDs) offer the advantages of a narrow light spectrum, low power consumption, and little heat production for the plants in a miniature system (K. Kobayashi,2013).

2.3.3.5. Solar power for automation

When using Alternating Current (AC) or Direct Current (DC) for most of the systems, the least amount of system had focused on solar power as the main power resource for the whole system. As a result of a research work Simon Siregar, Marlindia Ike Sari, and Rakhmi Jauhari had presented a prototype of an automated system hydroponic using a smart solar power plant that can monitor and control pH, temperature, water level, and the intensity of light (Siregar,2016).

2.3.3.6. Raspberry Pi

Raspberry Pi is a credit card size computer that capable of all the functionalities of a desktop computer. Some of the automated hydroponic and miniature systems had used Raspberry Pi instead of using Arduino Uno. T.Vineela, assistant professor in VVIT, India proposed to monitor crop-field using sensors for soil moisture, humidity, and temperature based on the Raspberry Pi system. By monitoring these parameters, the irrigation system can be automated if soil moisture is low(Vineela,2018).

Benjamin Odoi-Lartey and Edward Danso Ansong designed a system to improve agriculture production by using Raspberry Pi model 3 that serves as the base station for the sensor network and connects with an Arduino Uno serving as an intermediary between various sensors and the Raspberry Pi. Among the sensors integrated with the system are the soil sensor, humidity and temperature sensor, a Passive Infra-Red (PIR) sensor, and the Raspberry Pi Camera Module (Odio,2018).

2.3.3.7. Zigbee networking technology

Zigbee is created by the Zigbee alliance. It is based on the guidelines of IEEE 802.15.4. Zigbee is a notable remote communication protocol with reliable for wireless personal area networks and very low power consumption. A Zigbee network may be comprised of a coordinator node and multiple router and end devices. Through X- CTU software, the configuration of Zigbee modules can be done. Zigbee is widely used in home building control, automation, security, consumer electronics, personal computer peripherals medical monitoring, and toys. Zigbee has the following advantages in these applications such as, it offers long battery life, reliability, automatic or semiautomatic installation, the ability to easily add or remove network nodes, signals that can pass through walls and ceilings, and a low system cost.

Aqsa Mahir Tanmay and Banavalikar developed an automated soil moisture system to remotely controlled and monitored systems based on existing technologies and a ZigBee based IoT network to allow efficient routing in the network. Aqsa Mahir Tanmay and Banavalikar designed a device that monitors the temperature, the humidity of the field atmosphere, soil moisture and, it transmits the information where the farmhouse or outside the field by the remote receiver which a laptop connected to the Zigbee transceiver (Mahir,2015).

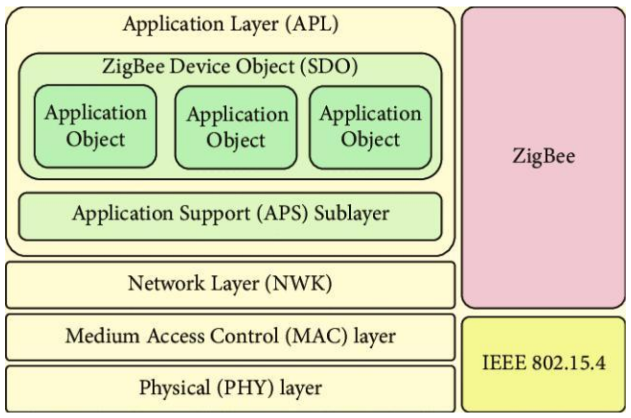


Figure 8- Zigbee protocol

2.3.3.8. Wireless Sensor Network (WSN)

Wireless Sensor Networks refers to a group of spatially dispersed and dedicated sensors for monitoring and recording the physical conditions of the environment and organizing the collected data at a central location. The data is forwarded through multiple nodes, and with a gateway, the data is connected to other networks like wireless Ethernet.

A project of a smart decision system for digital farming used a wireless sensor network system for gathered field parameters, the index vegetation (estimated using aerial images), and the irrigation events, such as flow level, pressure level, and wind speed. Then these data are processed in a decision-making system based on learning prediction rules in conjunction with the Drools rule engine (Cambra,2019). The research work of Aditi Mehta and Sanjay Patel had revealed the use of wireless sensor networks in IoT smart agriculture research and the significance of using the wireless sensor network. Besides, the authors referred to the use of wireless sensor networks in the IoT platform for improving the agricultural sector (J. Pitakphongmetha,2017).

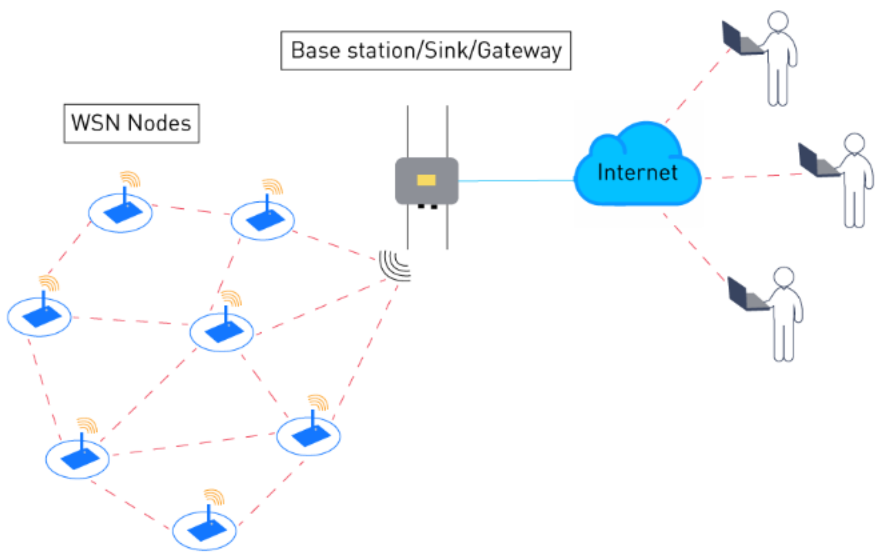


Figure 9- wireless sensor network- (Anon., 2025)

2.3.4.Market Research and gaps for the Miniature systems

The growing demand for food, the effects of climate change, and the quick development of agricultural technology have all contributed to the recent notable expansion of the worldwide smart greenhouse industry. The market was estimated to be worth USD 1.6 billion in 2021 and is expected to rise at a compound annual growth rate (CAGR) of about 12% to reach USD 3.2 billion by 2027. This increase highlights the growing use of smart greenhouse systems that combine automation, artificial intelligence (AI), and Internet of Things (IoT) technology to improve agricultural production sustainability and efficiency (Anon., 2024).

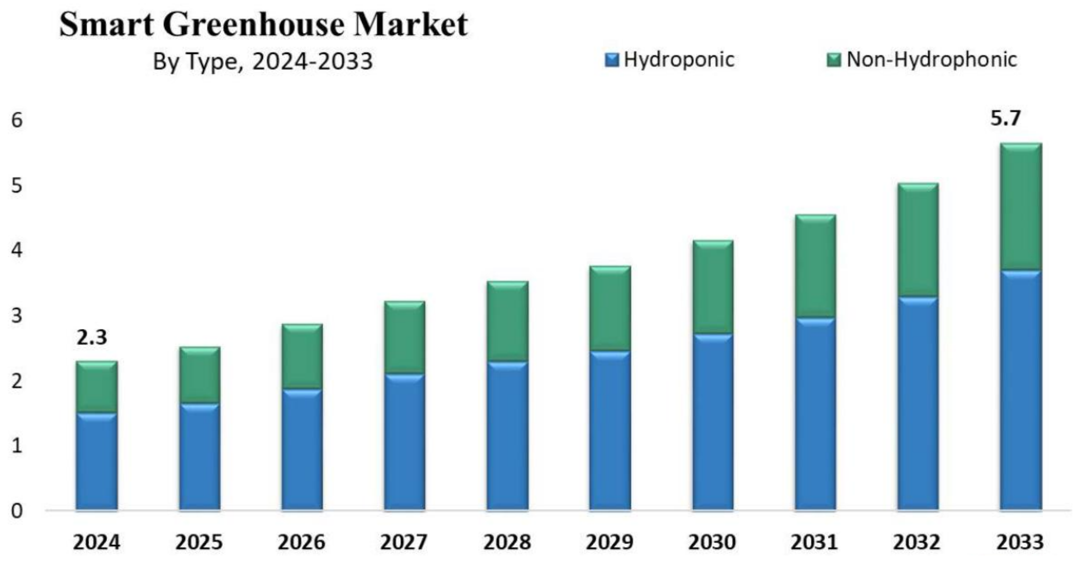


Figure 10- Smart greenhouse market prediction- (Anon., 2025)

The market is expanding due to a number of variables. The primary factor is the expanding world population, which raises the need for food production. The United Nations estimates that by 2050, there will be 9.6 billion people on the planet, which would require a significant expansion of the food supply. Due to limits including a lack of arable land, water shortage, and climate change susceptibility, traditional farming methods are unable to meet this demand. By providing controlled environment agriculture (CEA), which enables year-round cultivation, maximises resource utilisation, and lessens reliance on favourable weather conditions, smart greenhouses provide a workable alternative (Anon., 2025).

Innovations of technology concepts and components are essential to the growth of the smart greenhouse market. Operations in greenhouses have been completely transformed by developments in IoT devices, machine learning, and data analytics. For example, IoT and machine learning are used by intelligent agricultural greenhouse control systems to automatically monitor and modify environmental parameters, increasing crop growth efficiency and decreasing resource waste. By using less water, fertiliser, and pesticide, these technologies not only increase productivity but also support sustainable farming methods (Cangqing Wang, Jiangchuan Gong, 2025).

2.3.5. Conclusion

The changing landscape of IoT integration in agriculture has been covered in this chapter, with special attention paid to controlled environments, smart farming techniques, and phytotron systems. The analysis shows how many of the conventional problems in agricultural research have been effectively solved by technology developments, such as sensor networks, actuators, and automation platforms, especially with regard to the monitoring and upkeep of ideal plant growth conditions. Despite these advancements, it has been noted that a large number of tiny growing systems used for research still employ manual techniques, which leads to operational inefficiencies, higher costs, and a greater chance of human error.

Systems with IoT capabilities can increase accuracy, save labour costs, and provide long-term solutions for agriculture in regulated environments. Energy-efficient LED lighting systems and capacitive soil moisture sensors are examples of phytotron technologies that are essential for environmental control and real-time data collection. There are still gaps in practical deployment, particularly in the areas of standardisation, price, and accessibility. To create affordable, modular, and solar-powered solutions for scholarly and experimental applications, more research is required.

In conclusion, by offering dependable, precise, and remotely controlled conditions, the incorporation of IoT into phytotron-based agricultural systems shows significant promise to revolutionise plant research. The knowledge gathered from this thorough review lays the groundwork for the creation of increasingly complex, automated miniature phytotron systems that overcome the drawbacks of the manual methods currently in use while also being in line with the newest trends and industry best practices in smart agricultural technology.

Chapter 03 – Desing and Implementation

3.1. Design System Architecture

3.1.1. Functional features of the web system

- The system should be able to view the real-time sensor values in the miniature system such as temperature, humidity, soil moisture through the web application.
- The system should be able to control the actuators through the web application by the user.
- The system should be able to control actuators automatically when the sensor readings do not match with the pre-set values.
- The system should be able to store gathered data in the cloud service.

3.1.2. Functional features of the IoT based phytotron system

- Transfer sensor data to cloud server.
- Manage environmental conditions manually via control interface.
- Monitor temperature using DHT22 sensor.
- Monitor humidity levels using DHT22 sensor.
- Measure soil moisture using soil moisture sensor.
- Detect light intensity using LDR.
- Control artificial lighting with pixel LEDs.
- Operate ventilation system using stepper motors.
- Trigger fans or vents based on environmental thresholds.
- Control water flow or nutrient delivery using solenoid valve.
- Detect object or growth level using laser sensor.

3.1.3. Hardware and Software requirements

3.1.3.1. Hardware Requirements

The proposed system has to fulfil necessary hardware requirements in order to perform effectively. In terms of system implementation, a web server with minimum 25GB storage capability will be required to host the system as a web-based application was developed in this project.

This one is based on cloud server .user can log in to using PC, Laptops, Mobile devices.

Hardware Requirements for Phytotron

Microcontroller – NodeMCU

After the research, NodeMCU was selected as the best microcontroller among Raspberry Pi3 and Arduino UNO to implement the smart miniature system. Node MCU is a very simple device that is specially built for IoT devices. Node MCU Development Board depends on a generally esp8266 System on Chip from Expressive. ESP8266 is a very low-cost Wi-Fi chip and it has enough signal strength to give Wi-Fi for a home network. Its consolidated elements of the WIFI access point and station + microcontroller and utilizes straightforward LUA based programming.

ESP8266 NodeMCU offers Arduino-like equipment IO, this feature in the NodeMCU useful when building it as a web server to make the IoT project. It is an Event-driven API for system applications. 10 GPIOs D0-D10, PWM usefulness, IIC, and SPI communication, 1-Wire, and ADC A0, and so forth across the board. Wi-Fi systems administration (can be used as getting to the point and additionally station, have a webserver), interface with web to get or transfer information.

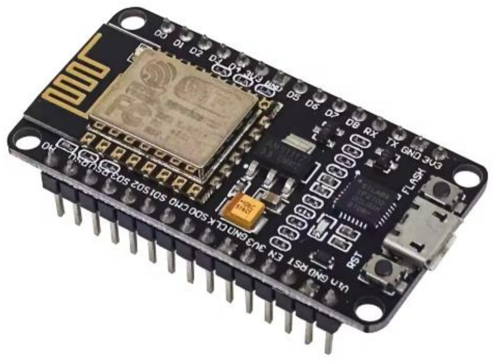


Figure 11- ESP8266 NodeMCU

Relay

Relays will be used to trigger the actuators in the system. 2-relay board used to control the fan and the water pump. VCC pin requires 5V which can be supplied from the VV pin in the NodeMCU. IN1 and IN2 pins are connected to the fan and the water pump respectively. Each relay has a red LED, and it will light up when the relay is on. Physical devices that need more than 5V (which cannot be supplied by the NodeMCU) can be controlled with the use of relays.

Figure 13- 5V relay for 2 channels

Figure 12-Relay with NodeMCU

Sensors

Temperature sensors and humidity sensors - DHT22 device

By the experiments throughout the problem domain to gather the temperature and humidity data selected DHT22 device. The DHT22 is a basic, digital temperature and humidity sensor uses a capacitive humidity sensor and a thermistor to measure the surrounding air in the miniature It spits out a digital signal on the data pin. 3V to 5V power and 2.5mA current can be supplied from the NodeMCU board as its I/O pins can give 3V and 5V. DHT22 ideal for 20-80% humidity readings with 5% accuracy. This sensor has 5% accuracy for humidity, and it is more than enough for a miniature system. Also, good enough for 0-50°C temperature readings with ±2°C accuracy. The accuracy of the temperature readings is enough to implement the miniature system.

Figure 14-DHT22 temperature and Humidity sensor

The capacitive soil moisture sensor

Moisture of the soil also another measurable factor of the plant growth and to measure the moisture need to use soil moisture sensors. However rather than other soil moisture sensing capacitive soil moisture sensor measures the soil moisture levels by the capacitive sensing. This sensor is made of corrosion-resistant material. Because of that, it can prevent correction and long service life. Soil moisture sensors measure the volumetric water content in the soil. It measures the soil moisture by using the properties of the soil such as electrical resistance, dielectric constant. The moisture sensor value becomes lesser with the presence of water in the soil (e.g. sensor shows value 0 when the nodes dipped in water).

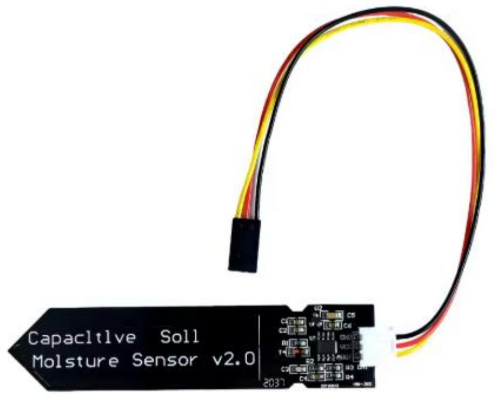


Figure 15- Capacitive soil moisture sensor V2.0

Light sensors - LDR Sensors

Data of the light conditions get from the LDR Sensors connected with the NodeMCU.

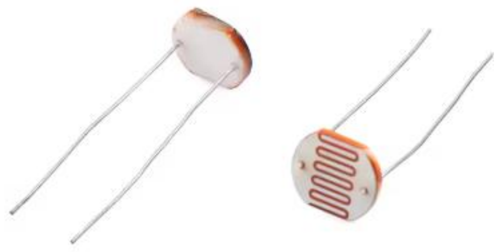


Figure 16- LDR sensors

Actuators

DC Fan

In the smart miniature growing system, DC fans have used to decrease the temperature inside the cabinet. Generally, the temperature in the miniature system is always increasing as they are designed to increase the temperature inside the cabinet. A medium-sized fan will control the inside temperature by passing inside warm air out. The fan will get signals through a relay which operates by NodeMCU main circuit.

Figure 17- DC fan

LED Light with controller

The purpose of an LED light controller is to mimic natural sunshine and control photoperiods, which are essential for photosynthesis, plant growth, and blooming. With controlled LED illumination, scientists may modify the duration, spectrum, and intensity of light to suit the requirements of various plant species or experimental setups.

Figure 18- LED Light with controller

3.1.3.2. Software requirements

Web application development

The selection of appropriate technologies and coding practices is crucial for successful software system development. Programmers use versatile tools to deliver software solutions for miniature system users. Web application technology is used for multiple end-users and remote logging. The smart miniature system must meet end-user and non-functional requirements, ensuring successful implementation and meeting the needs of multiple users.

- IDE – Visual Studio Code
- Languages – HTML5, PHP, BootStrap5, JavaScript
- Database – MySQL
- Server – Azure
- Operating System – Windows 10

Arduino

The Arduino Integrated Development Environment (IDE) is an application that supports C and C++ languages as a cross-platform application for Windows, macOS, and Linux. Arduino IDE has been used to write and upload programs with the help of third-party cores to Arduino compatible boards like Arduino UNO, Raspberry Pi, NodeMCU, and other vendor development boards. The Arduino IDE has been employed in the program avrdude to convert the executable code into a text file in hexadecimal. It encoded and loaded into the Arduino board by a loader program in the board's firmware, used as the uploading tool to flash which the user code onto official Arduino boards by default. For the program of the NodeMCU and the Arduino UNO board in the Smart, Minatare System has been used as the Arduino IDE.

3.1.4. Phytotron Design

3.1.4.1. Device sketch

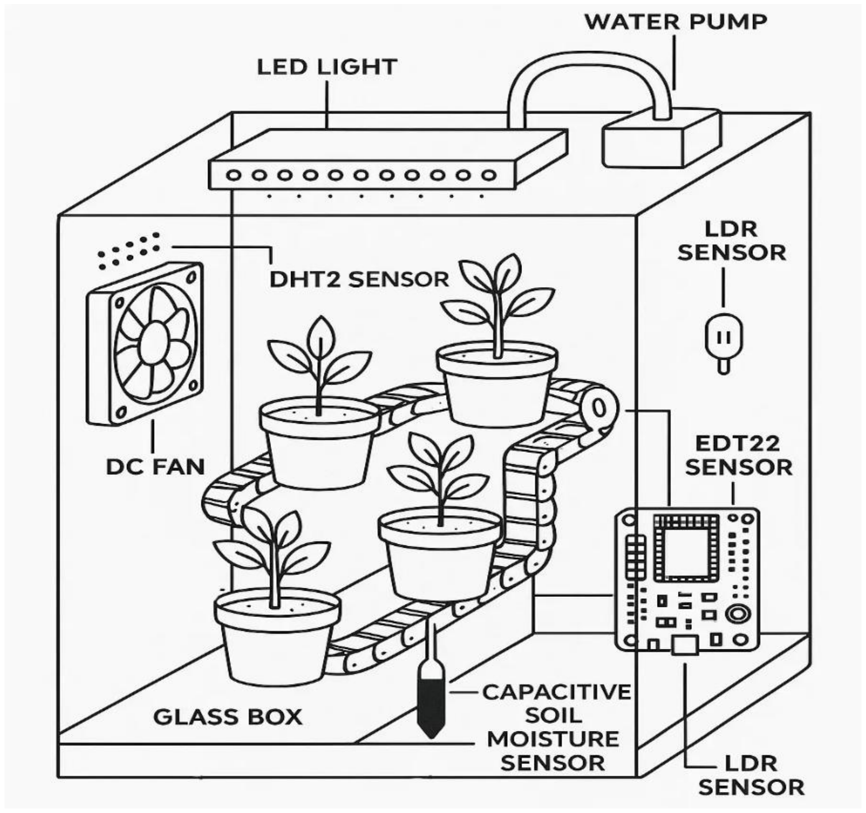


Figure 19- Phytotron sketch (source – Author)

The phytotron prototype sketch illustrates a clever environmental monitoring and control system for plant development that combines multiple hardware elements.

3.1.4.2. Process flow diagram of the Internet of Things (IoT)-based Phytotron Environmental Monitoring and Control System

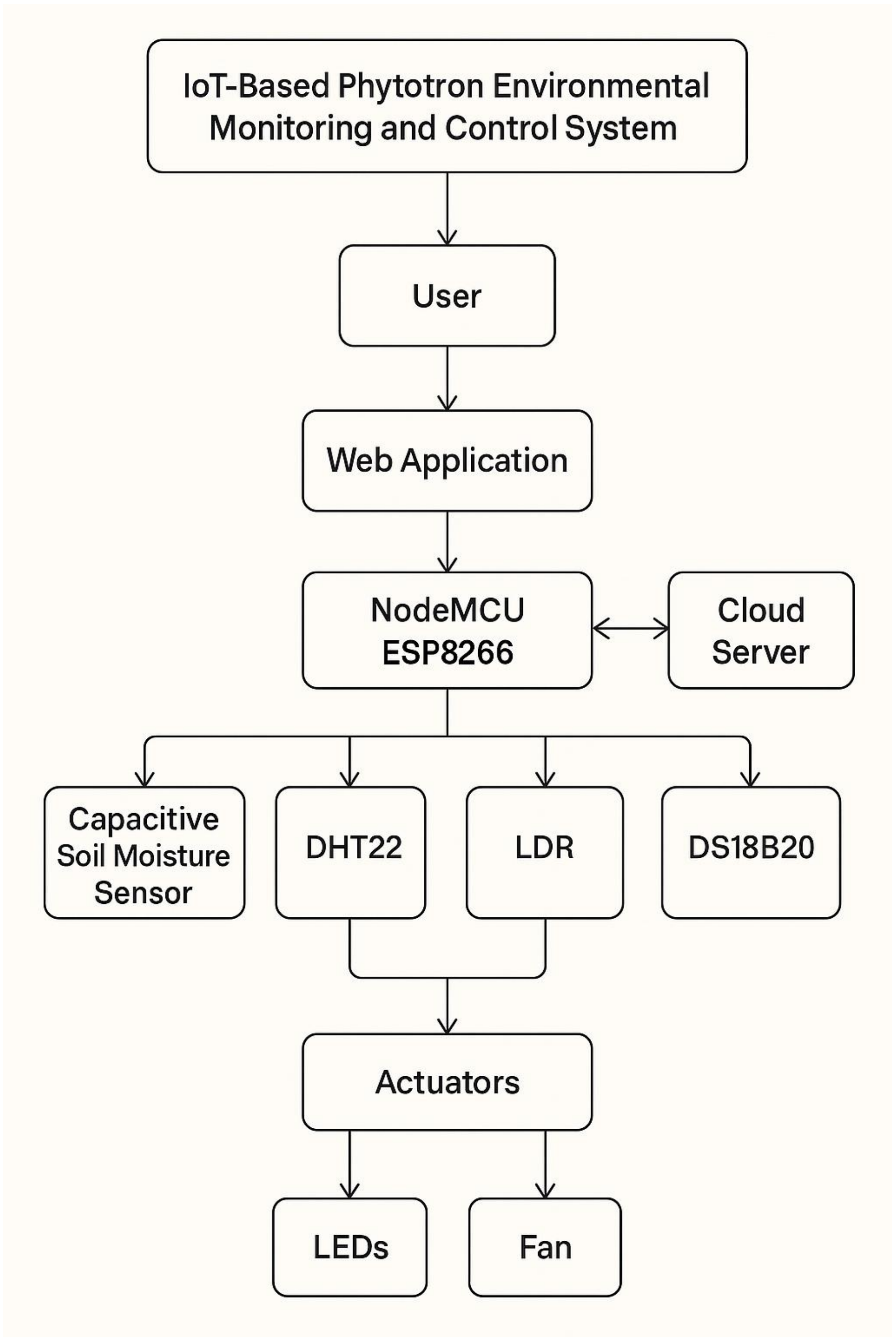

IoT-Based Phytotron Environmental Monitoring and Control System
User
Web Application
NodeMCU ESP8266
Cloud Server
Capacitive Soil Moisture Sensor
DHT22
LDR
DS18B20
Actuators
LEDs
Fan

Figure 20- system flow diagram

3.1.4.3. System communication sketch

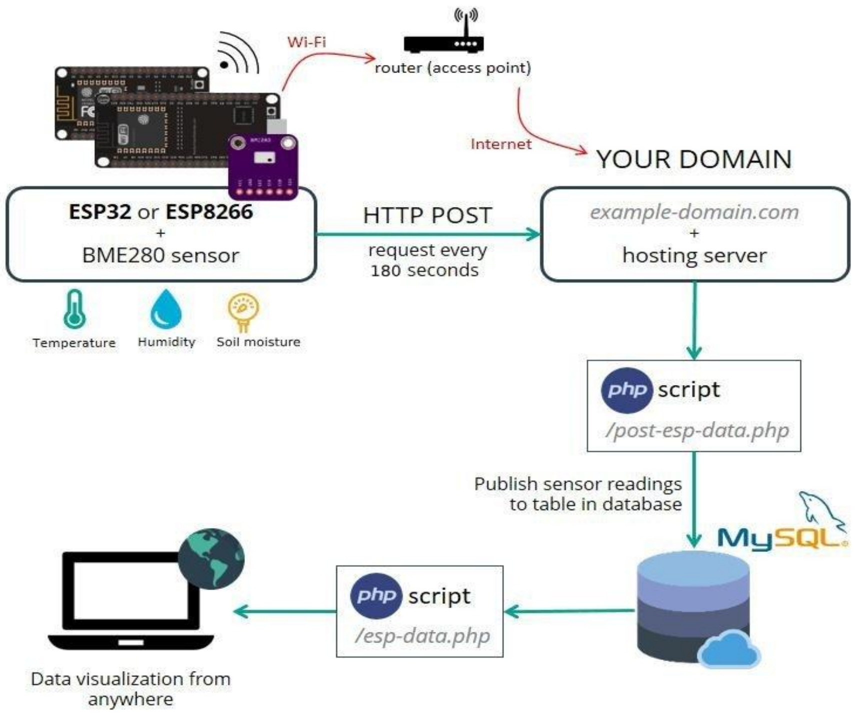


Figure 21- Phytotron communication sketch

The phytotron system's communication process creates a seamless interaction between hardware and software components. The NodeMCU ESP8266 microcontroller serves as the central hub, collecting data from multiple sensors (DHT22 temperature/humidity, capacitive soil moisture, and LDR light sensors) and transmitting this information to the Firebase cloud database via WiFi connectivity. In the meantime, the web application—which was developed on Azure servers using HTML5, PHP, Bootstrap5, and JavaScript—gets this saved data from Firebase and displays it in an easy-to-use visualisation interface that can be accessed from anywhere. Control commands are transmitted back to the NodeMCU over the cloud when users interact with the web application to modify growth conditions. The NodeMCU then uses the relay board to send the proper responses from the attached actuators (DC fan, water pump, pixel LEDs, and stepper motors). When sensor readings diverge from pre-established thresholds, the system autonomously activates actuators to create a responsive environment that maintains ideal growing conditions with the least amount of human intervention. This bidirectional communication pathway also supports automated control functions.

3.1.4.4. Database of web application design

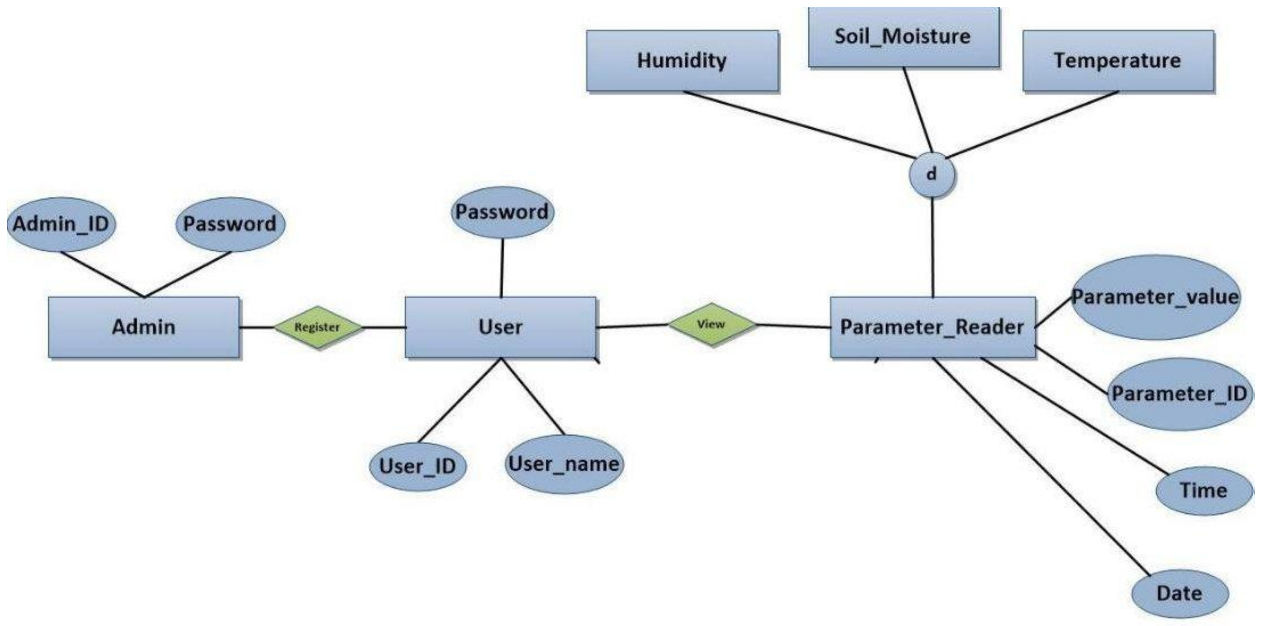


Figure 22-ER diagram

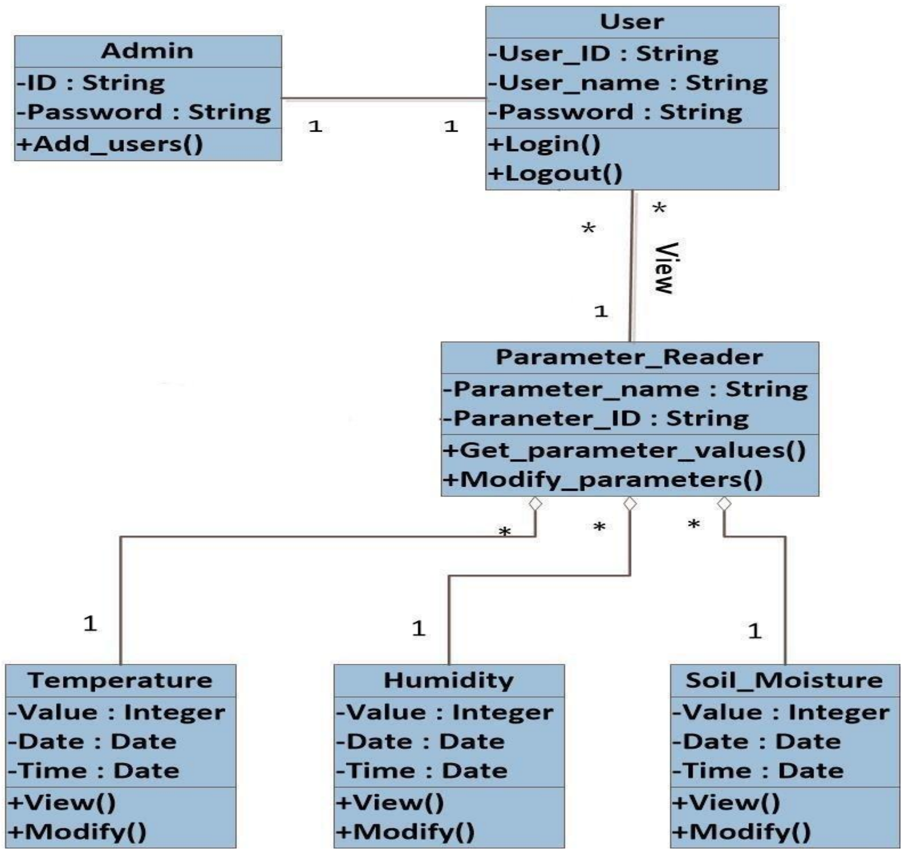


Figure 23-Calss diagram for the web application

3.2. Implementation of the project

3.2.1. IoT based Phytotron development

The core component of the Internet of Things-based phytotron system is the NodeMCU ESP8266 microcontroller. It is in charge of collecting information from multiple sensors and managing the actuators. An inexpensive, small gadget that combines Wi-Fi with the microcontroller, the NodeMCU is a great option for Internet of Things applications. Through its GPIO ports, it connects to sensors, processes the data locally, and then transmits it to the cloud for additional control and monitoring.

The phytotron's temperature and humidity are monitored using the DHT22 sensor. Accurate and dependable readings are provided by the thermistor and capacitive humidity sensor in this sensor. One of the NodeMCU's digital GPIO pins is directly connected to the DHT22. It is powered by the 3.3V output of the NodeMCU and transmits a digital signal that indicates the humidity and temperature. The sensor's data is essential for controlling the phytotron's environment and guaranteeing that plants grow in the best possible conditions.

The system uses a capacitive soil moisture sensor to measure soil moisture. This capacitive sensor employs the soil's dielectric characteristics to measure moisture levels, in contrast to other kinds of sensors that may deteriorate over time. Its lifespan is increased by the corrosion-resistant materials used in its construction. The NodeMCU reads the moisture level from the sensor's output, which is an analogue signal, using its analogue input pin. Additionally, the system incorporates LDR (Light Dependent Resistor) sensors to track the phytotron's light levels. By detecting the resistance changes in response to light exposure, these sensors are able to determine the intensity of light. The system can automatically change the lighting according to the detected light intensity because the LDR sensors are connected to the NodeMCU via an analogue pin.

Actuators such as a water pump and DC fans are incorporated into the system to regulate the environment. The phytotron's internal temperature is controlled by the DC fan. By removing warm air, the fan helps to cool the system's surroundings, which can occasionally get excessively warm. A relay module controls the fan, enabling the NodeMCU to turn it on or off in response to temperature readings. In a similar manner, the relay regulates the irrigation system by controlling the water pump. To keep the plants from drying out, the pump is turned on when the soil moisture falls below a predetermined level.

A 5V regulated power source powers the entire system, giving the NodeMCU and the connected devices enough current. By distributing the electricity across the parts, smooth operation is ensured. Because it guarantees that every component operates as intended and without interruption, the power supply is essential to preserving the system's stability and dependability.

In conclusion, a well-coordinated combination of sensors, actuators, and the central NodeMCU microcontroller is required for the actual realisation of the Internet of Things-based phytotron system. The architecture of the system enables automatic control and ongoing environmental monitoring, offering a practical and efficient way to manage the growing conditions of plants in a small phytotron.

Pin configuration details

Component	NodeMCU Pin	GPIO
Stepper Motor (via A4988)	D9	GPIO3
	D8	GPIO15
	GND	GND
Relay (Water Pump/Fan)	D12	GPIO10
LDR Sensor	A0	A0
Software Serial Device	TX → D5	GPIO14
	RX → D6	GPIO12

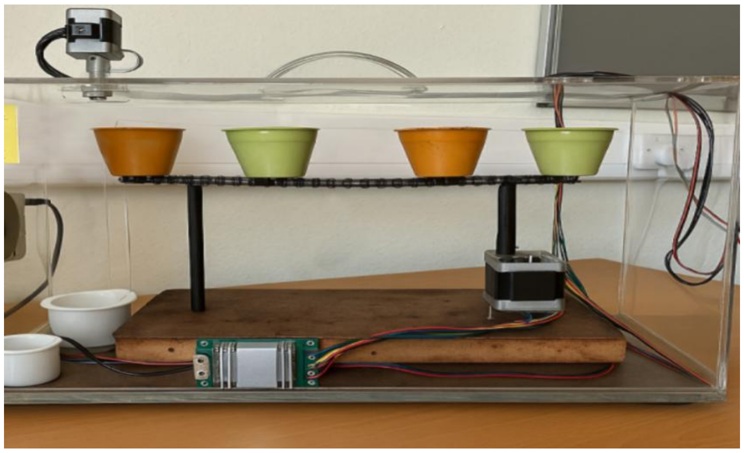


Figure 24-side view of the phytotron

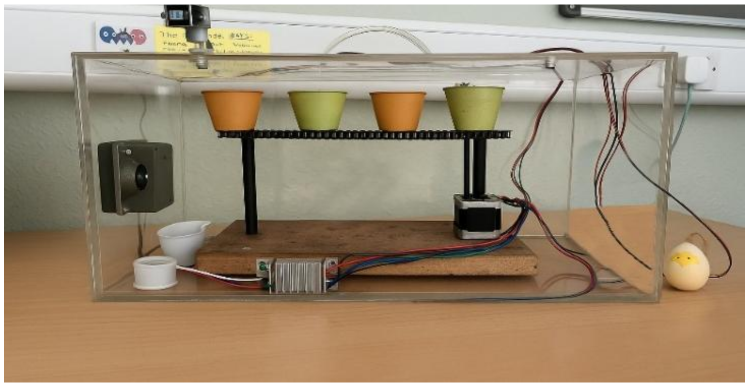
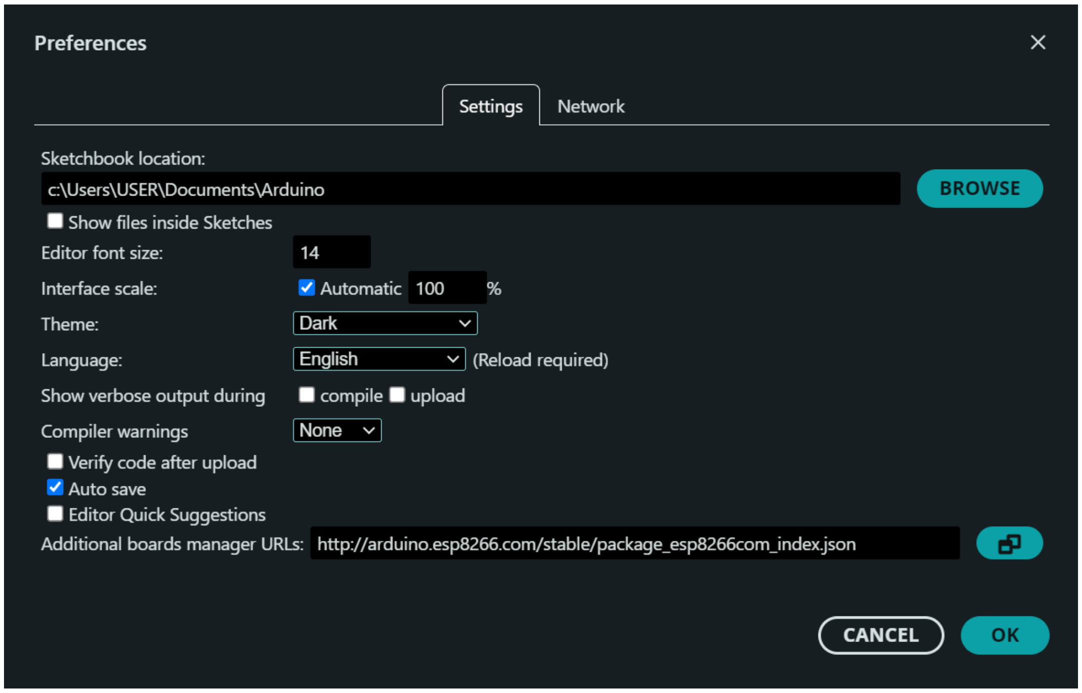


Figure 25- IoT based Phytotron

3.2.2. IOT Device Code explanation

Motor Controller code

IoT device code implementation completed using the Arduino IDE and since the targeted board if ESP8266 first board has added to the IDE through the additional boards URL option and needed some essential libraries for this phytotron development.



Sketchbook location: c:\Users\USER\Documents\Arduino

☐ Show files inside Sketches

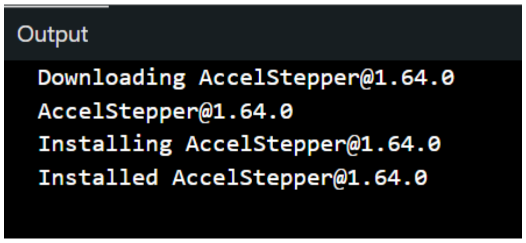
Editor font size:	14
Interface scale:	✓ Automatic 100 %
Theme:	Dark
Language:	English (Reload required)
Show verbose output during	☐ compile ☐ upload
Compiler warnings	None

☐ Verify code after upload
✓ Auto save
☐ Editor Quick Suggestions

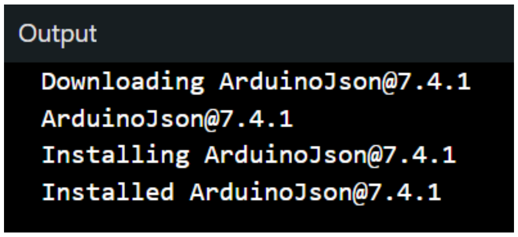
Additional boards manager URLs: http://arduino.esp8266.com/stable/package_esp8266com_index.json

Figure 26- board defining

This Arduino code integrates **sensor reading**, **stepper motor control**, **serial communication**, and **JSON data parsing** to create an automated pump controlling features to the phytotron system. According to the sensor inputs, when factors changing this automation is take place.

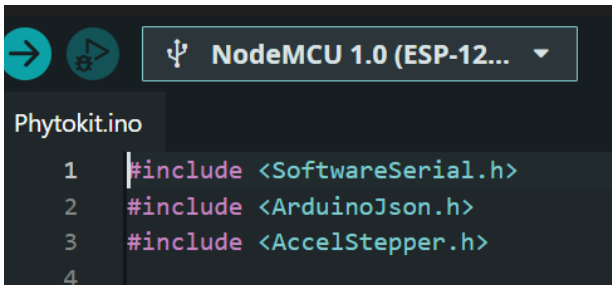


```
Downloading AccelStepper@1.64.0
AccelStepper@1.64.0
Installing AccelStepper@1.64.0
Installed AccelStepper@1.64.0
```



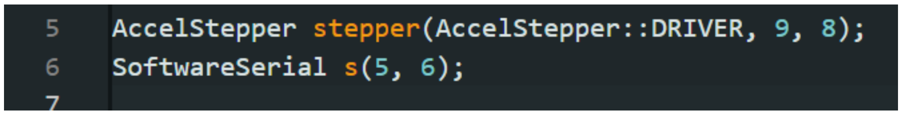
```
Downloading ArduinoJson@7.4.1
ArduinoJson@7.4.1
Installing ArduinoJson@7.4.1
Installed ArduinoJson@7.4.1
```

Figure 27-Library Inclusions and Definitions 2



```
#include <SoftwareSerial.h>
#include <ArduinoJson.h>
#include <AccelStepper.h>
```

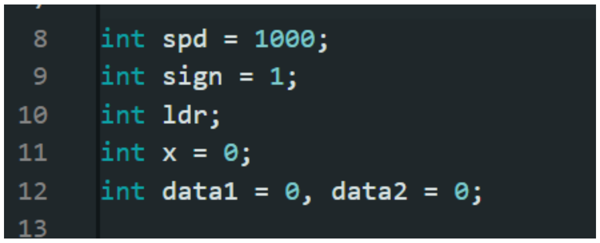
Figure 28- Library management



```
AccelStepper stepper(AccelStepper::DRIVER, 9, 8);
SoftwareSerial s(5, 6);
```

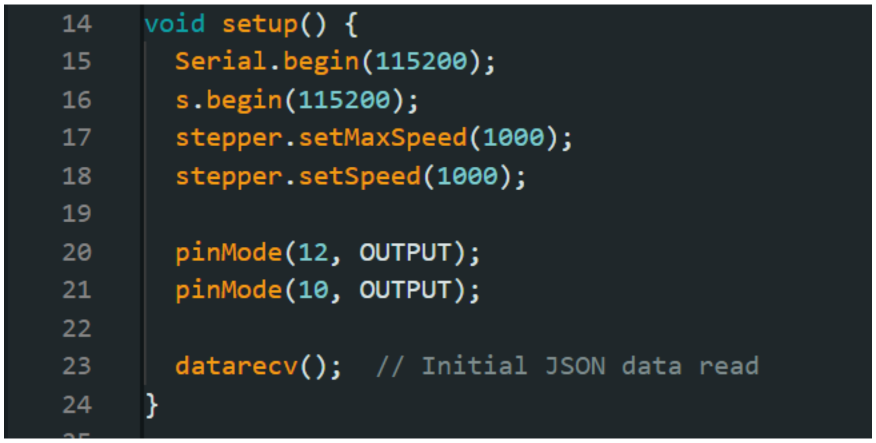
Figure 29- global variables for stepper motor

Declare the global variable to use a driver attached to pins 8 (direction) and 9 (step) to create a stepper motor object. Pins 5 (RX) and 6 (TX) are configured for software serial connection.



```
int spd = 1000;
int sign = 1;
int ldr;
int x = 0;
int data1 = 0, data2 = 0;
```

Figure 30- Object initiation



```
void setup() {
  Serial.begin(115200);
  s.begin(115200);
  stepper.setMaxSpeed(1000);
  stepper.setSpeed(1000);

  pinMode(12, OUTPUT);
  pinMode(10, OUTPUT);

  datarecv();  // Initial JSON data read
}
```

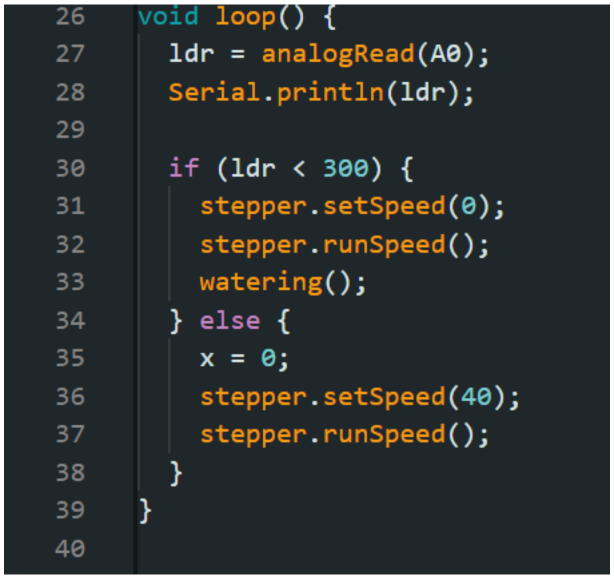
Figure 31-set up function

SoftwareSerial and Serial monitors are initialised.

Determines the motor's starting and maximum speed.

As outputs, pins 12 and 10 are utilised, for example, to control a motor driver or solenoid valve.

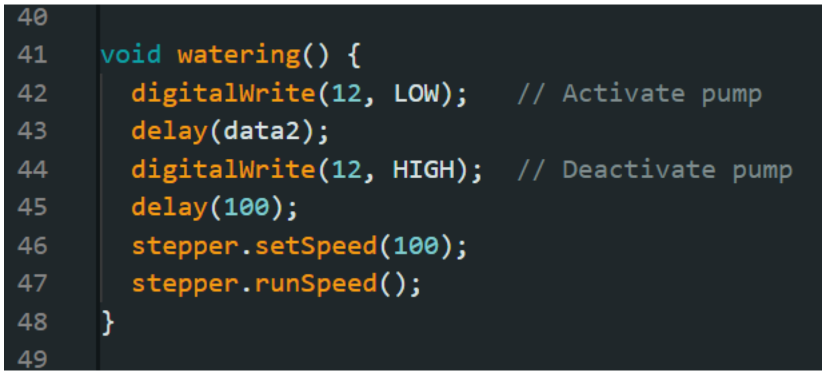
Retrieves initial configuration data (such as watering time) by calling datarecv().



```
void loop() {
  ldr = analogRead(A0);
  Serial.println(ldr);

  if (ldr < 300) {
    stepper.setSpeed(0);
    stepper.runSpeed();
    watering();
  } else {
    x = 0;
    stepper.setSpeed(40);
    stepper.runSpeed();
  }
}

```

Figure 32- Loop function

Reads the value of the LDR sensor from analogue pin A0. Watering() is activated and the motor stops if the light level is low (ldr < 300). If not, the stepper motor runs at a modest speed (for example, to change the ventilation or illumination).

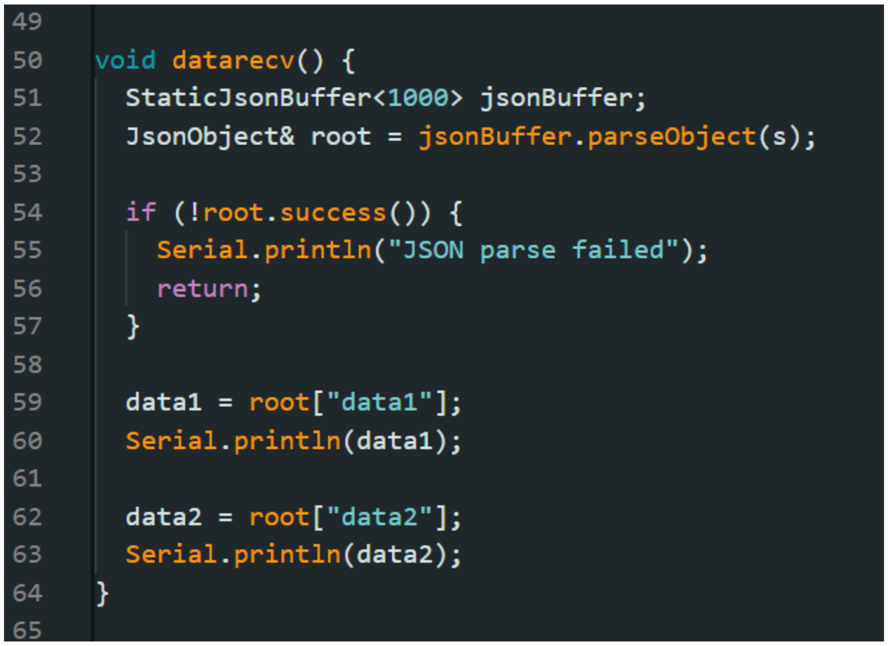


```
void watering() {
  digitalWrite(12, LOW);    // Activate pump
  delay(data2);
  digitalWrite(12, HIGH);   // Deactivate pump
  delay(100);
  stepper.setSpeed(100);
  stepper.runSpeed();
}
```

Figure 33-watering function

Simulates the watering via relay device and timing is controlled by the data2 which pass by the JSON inputs relates to the sensor. interprets JSON information obtained from software serial numbers. Two parameters, data1 and data2, are extracted, potentially:

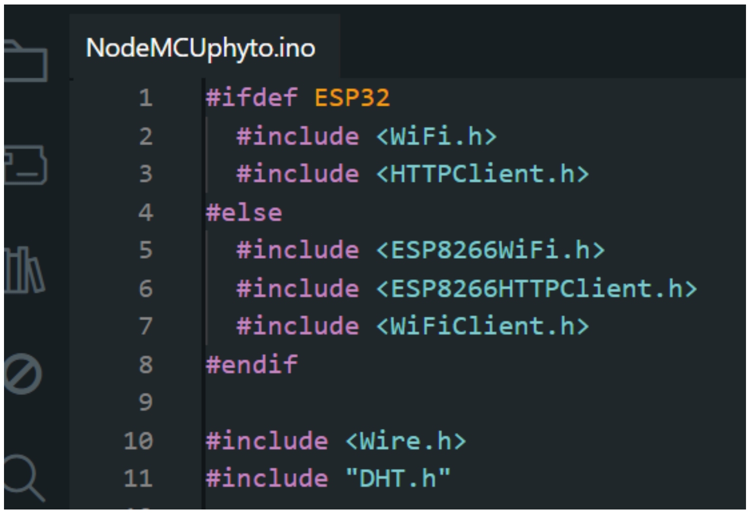
- data1: Although not used here, it may represent motor speed or threshold.
- data2: Watering delay duration (used in watering()).



```
void datarecv() {
  StaticJsonBuffer<1000> jsonBuffer;
  JsonObject& root = jsonBuffer.parseObject(s);

  if (!root.success()) {
    Serial.println("JSON parse failed");
    return;
  }

  data1 = root["data1"];
  Serial.println(data1);

  data2 = root["data2"];
  Serial.println(data2);
}
```

Figure 34- JSON passing

NodeMCU code with cloud integration



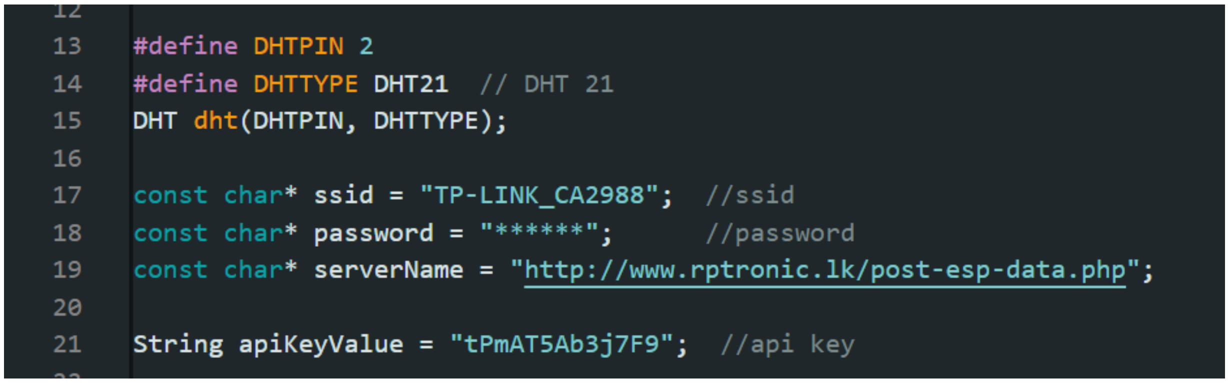
NodeMCUphyto.ino

```
#ifdef ESP32
  #include <WiFi.h>
  #include <HTTPClient.h>
#else
  #include <ESP8266WiFi.h>
  #include <ESP8266HTTPClient.h>
  #include <WiFiClient.h>
#endif

#include <Wire.h>
#include "DHT.h"
```

Figure 35- Platform dependency management

Depending on whether the board is an ESP32 or ESP8266, includes HTTP and Wi-Fi libraries. guarantees that the codebase is compatible with both microcontrollers. Incorporates the DHT sensor library and configures digital pin D2 to use DHT21. sets up the DHT sensor object so that it can subsequently measure humidity and temperature.

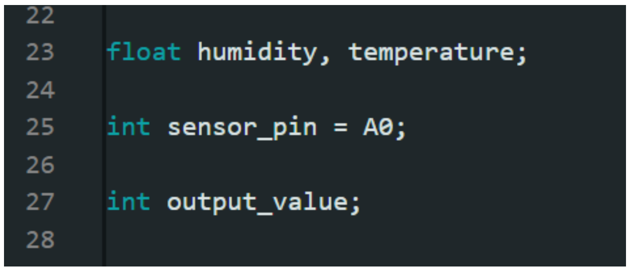
- Wire.h: Handles I2C communication (though not used directly here).
- DHT.h: Library to interface with DHT temperature & humidity sensors.
- DHTPIN 2: The DHT sensor is connected to GPIO pin 2.
- DHTTYPE DHT21: Specifies you're using a DHT21 sensor model.
- dht(...): Creates a DHT object to read sensor data



```
#define DHTPIN 2
#define DHTTYPE DHT21   // DHT 21
DHT dht(DHTPIN, DHTTYPE);

const char* ssid = "TP-LINK_CA2988";  //ssid
const char* password = "******";       //password
const char* serverName = "http://www.rptronic.lk/post-esp-data.php";

String apiKeyValue = "tPmAT5Ab3j7F9";  //api key
```

Figure 36-Server credentials

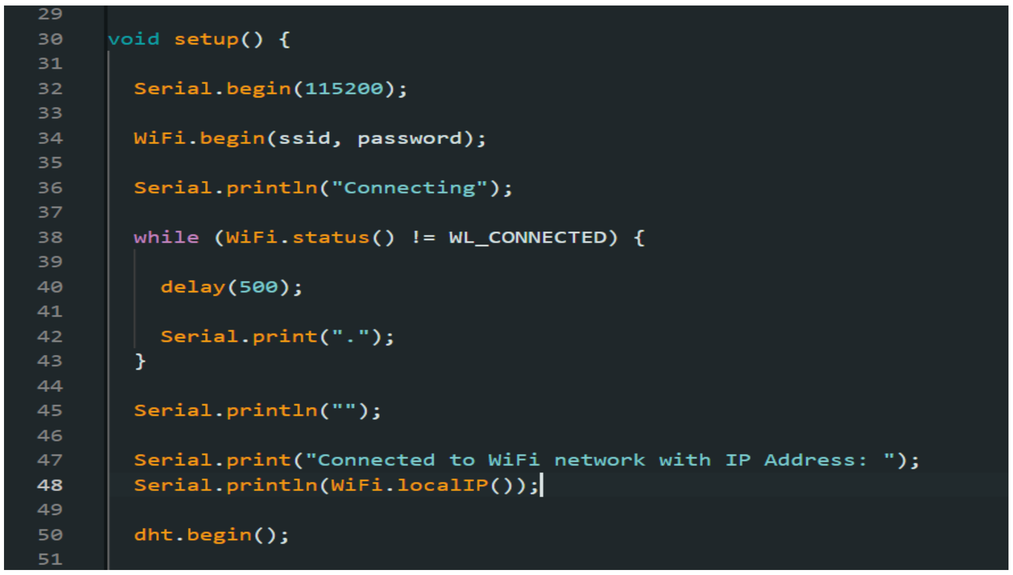
Saves the server URL to which the data is transferred, together with the Wi-Fi SSID and password.



```
float humidity, temperature;

int sensor_pin = A0;

int output_value;

```

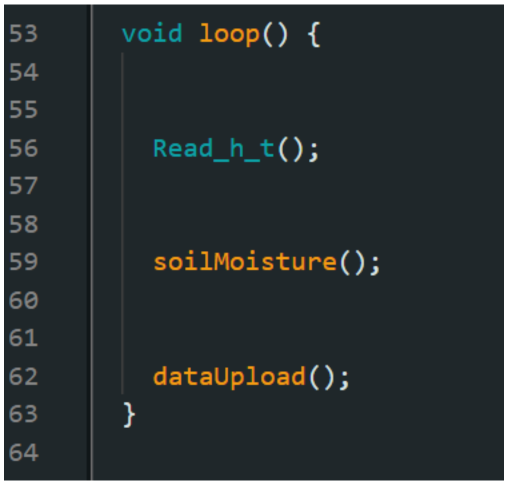
Figure 37-variables for the parameters

Declares variables to store the values from analogue soil moisture sensors, temperature, and humidity. Connects to the Wi-Fi network and initiates serial communication. When connected, it prints the IP address and shows the status of the connection. sets up the DHT sensor. initiates debugging serial transmission at 115200 baud. initiates and waits for a Wi-Fi connection. Once connected, the IP address is printed. uses dht.begin() to initialise the DHT sensor.



```

void setup() {

  Serial.begin(115200);

  WiFi.begin(ssid, password);

  Serial.println("Connecting");

  while (WiFi.status() != WL_CONNECTED) {

    delay(500);

    Serial.print(".");
  }

  Serial.println("");

  Serial.print("Connected to WiFi network with IP Address: ");
  Serial.println(WiFi.localIP());

  dht.begin();

```

Figure 38-function setup

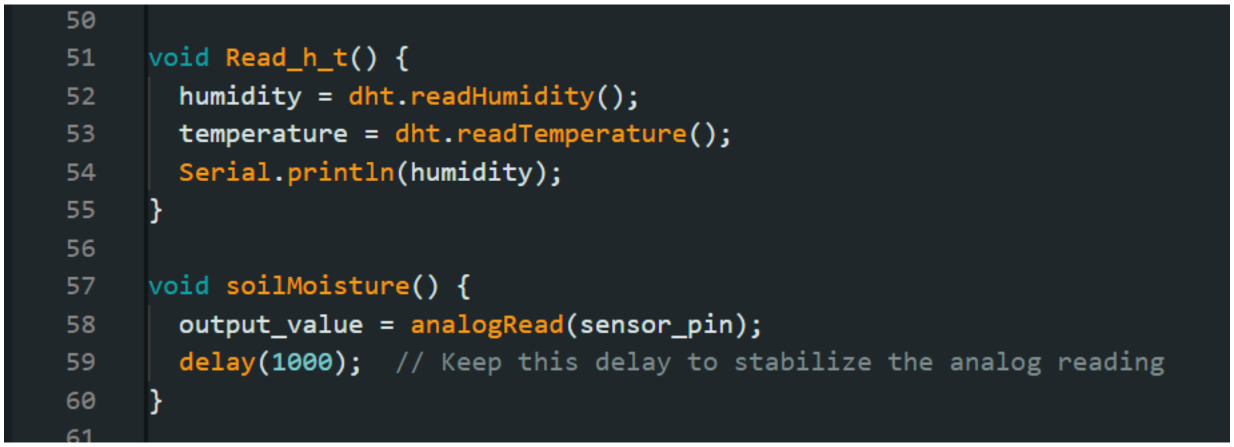


```
void loop() {


  Read_h_t();


  soilMoisture();


  dataUpload();
}

```

Figure 39-Loop the function

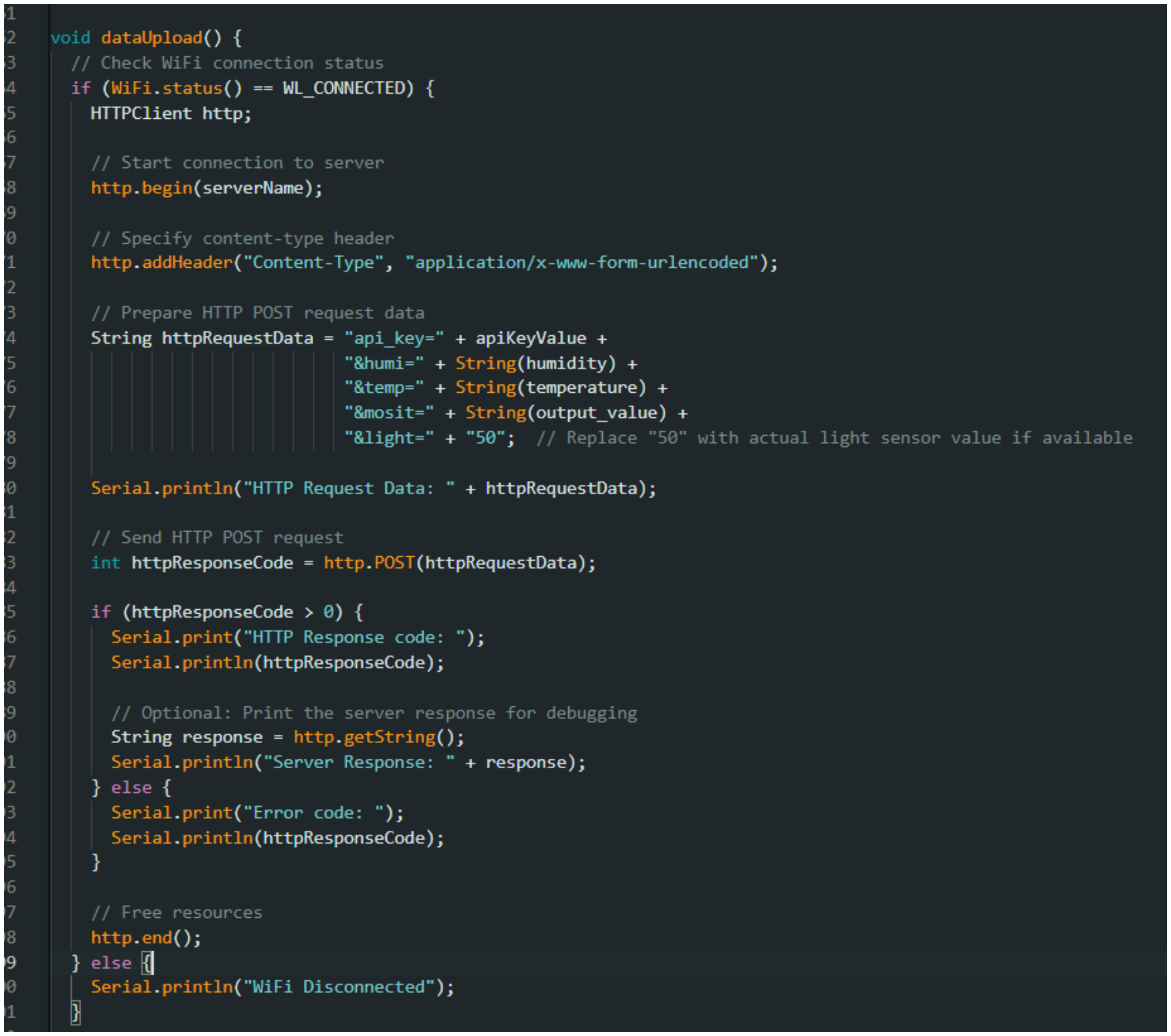
Makes the function runs repeatedly and reads sensors and uploads data to the server every 180 seconds (as set in dataUpload()).

Reads temperature and humidity readings from the DHT21 sensor. Uses global variables to store values. For simple debugging, it prints the humidity to the serial monitor. Analogue value is read from the soil moisture sensor that is attached to A0. The numbers returned by analogRead() range from 0 (wet) to 1023 (dry). Delay of 1000 includes a one-second pause to give the reading time to settle.



```

void Read_h_t() {
  humidity = dht.readHumidity();
  temperature = dht.readTemperature();
  Serial.println(humidity);
}

void soilMoisture() {
  output_value = analogRead(sensor_pin);
  delay(1000);  // Keep this delay to stabilize the analog reading
}

```

Figure 40-read sensor data

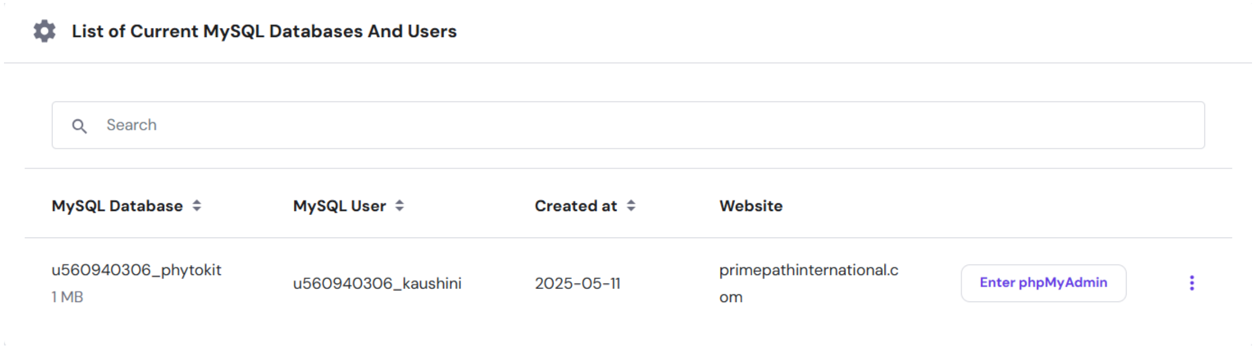


```
void dataUpload() {
  // Check WiFi connection status
  if (WiFi.status() == WL_CONNECTED) {
    HTTPClient http;

    // Start connection to server
    http.begin(serverName);

    // Specify content-type header
    http.addHeader("Content-Type", "application/x-www-form-urlencoded");

    // Prepare HTTP POST request data
    String httpRequestData = "api_key=" + apiKeyValue +
                             "&humi=" + String(humidity) +
                             "&temp=" + String(temperature) +
                             "&mosit=" + String(output_value) +
                             "&light=" + "50";  // Replace "50" with actual light sensor value if available

    Serial.println("HTTP Request Data: " + httpRequestData);

    // Send HTTP POST request
    int httpResponseCode = http.POST(httpRequestData);

    if (httpResponseCode > 0) {
      Serial.print("HTTP Response code: ");
      Serial.println(httpResponseCode);

      // Optional: Print the server response for debugging
      String response = http.getString();
      Serial.println("Server Response: " + response);
    } else {
      Serial.print("Error code: ");
      Serial.println(httpResponseCode);
    }

    // Free resources
    http.end();
  } else {
    Serial.println("WiFi Disconnected");
  }
```

Figure 41-Upload data to server

The dataUpload() function is in charge of using an HTTP POST request to upload sensor data to a distant server. After determining whether the device is Wi-Fi connected, it builds an HTTP client and connects to the designated server. The function creates a URL-encoded string that it delivers as the body of the POST request, including the API key, temperature, humidity, soil moisture, and a hardcoded light value. Following transmission, the HTTP response code and, if desired, the server's answer are logged for debugging purposes. To manage the frequency of data uploads, the function ends by releasing the HTTP resources and adding a 180-second delay.

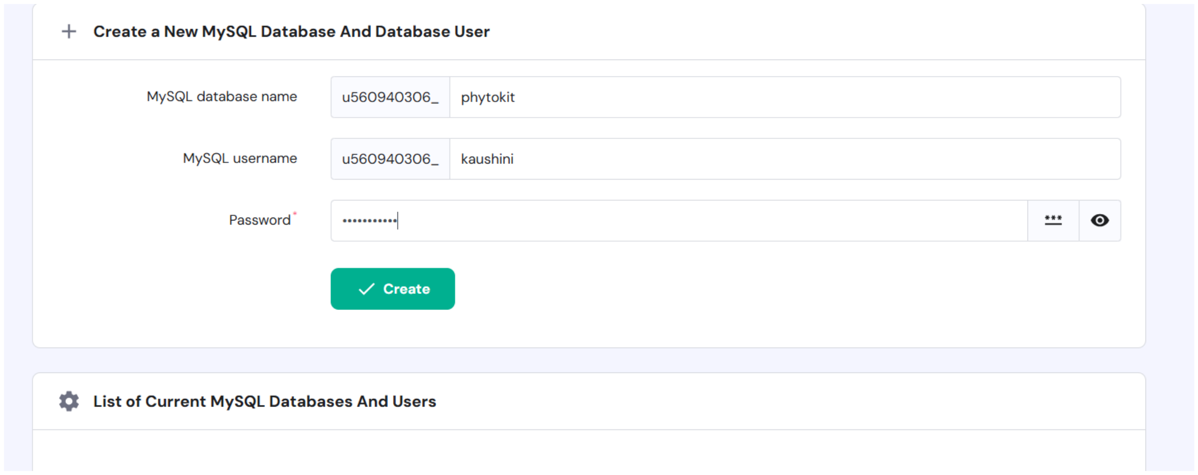
Database Management



List of Current MySQL Databases And Users

MySQL Database	MySQL User	Created at	Website
u560940306_phytokit 1 MB	u560940306_kaushini	2025-05-11	primepathinternational.com

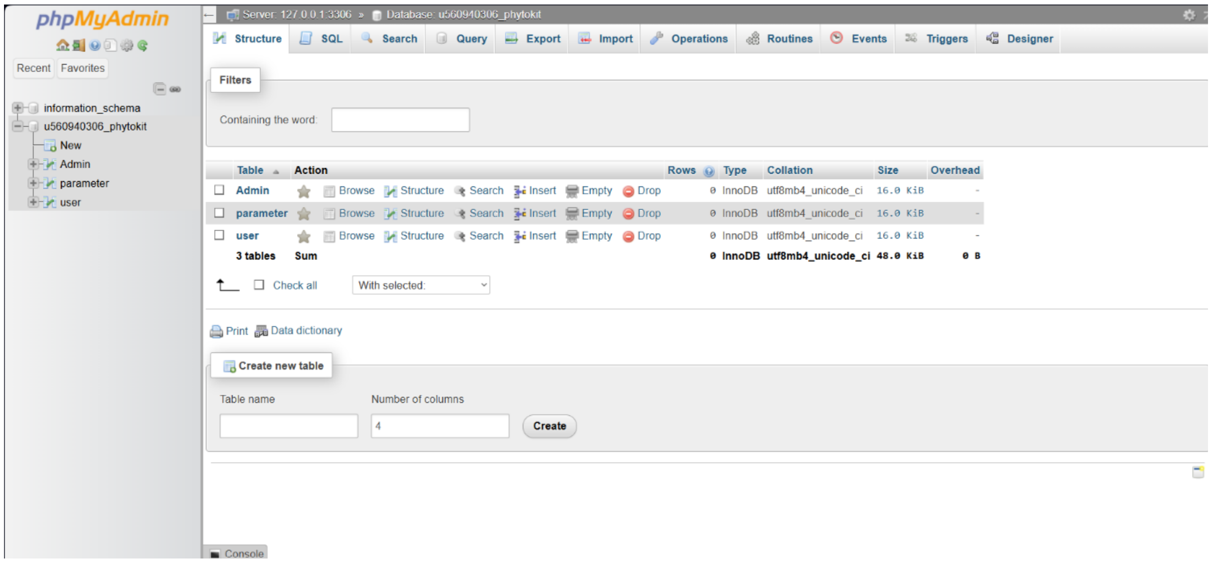
Figure 42-database connection on host could



Create a New MySQL Database And Database User

MySQL database name	u560940306_ phytokit
MySQL username	u560940306_ kaushini
Password*	•••••••••••

Figure 43-database and credentials



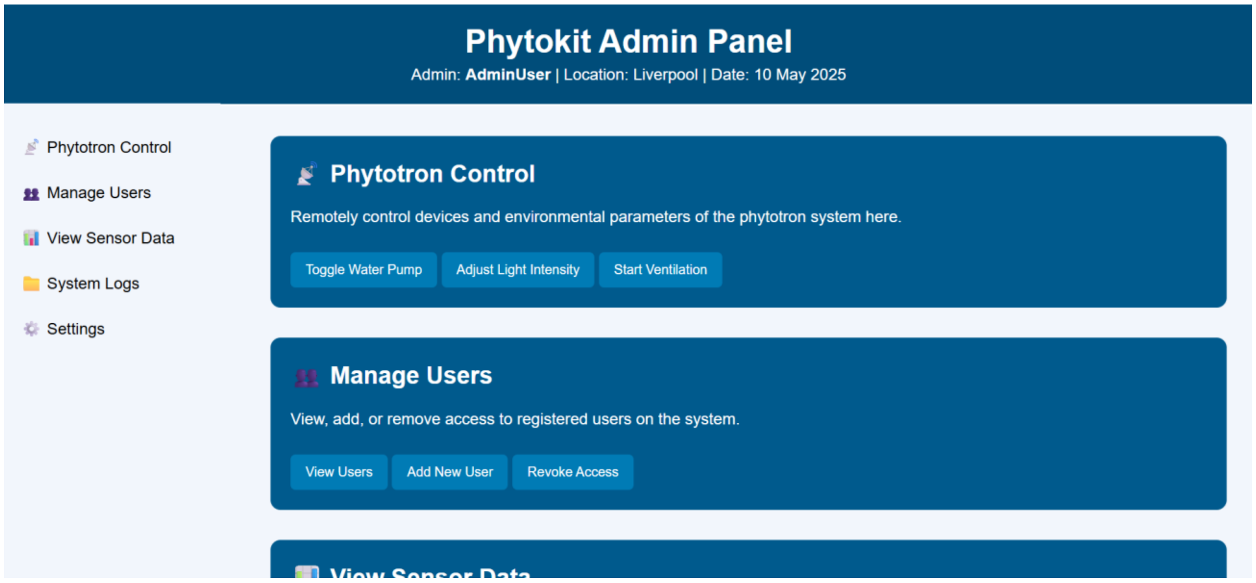
Server: 127.0.0.1:3306 » Database: u560940306_phytokit

Table	Action	Rows	Type	Collation	Size	Overhead
Admin	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci	16.0 KiB	-
parameter	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci	16.0 KiB	-
user	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci	16.0 KiB	-
3 tables	Sum	0	InnoDB	utf8mb4_unicode_ci	48.0 KiB	0 B

Figure 44-phytokit database for web application

3.2.3. Phytokit Web Application

Admin Pannel



Phytokit Admin Panel

Admin: **AdminUser** | Location: Liverpool | Date: 10 May 2025

Phytotron Control

Remotely control devices and environmental parameters of the phytotron system here.

Manage Users

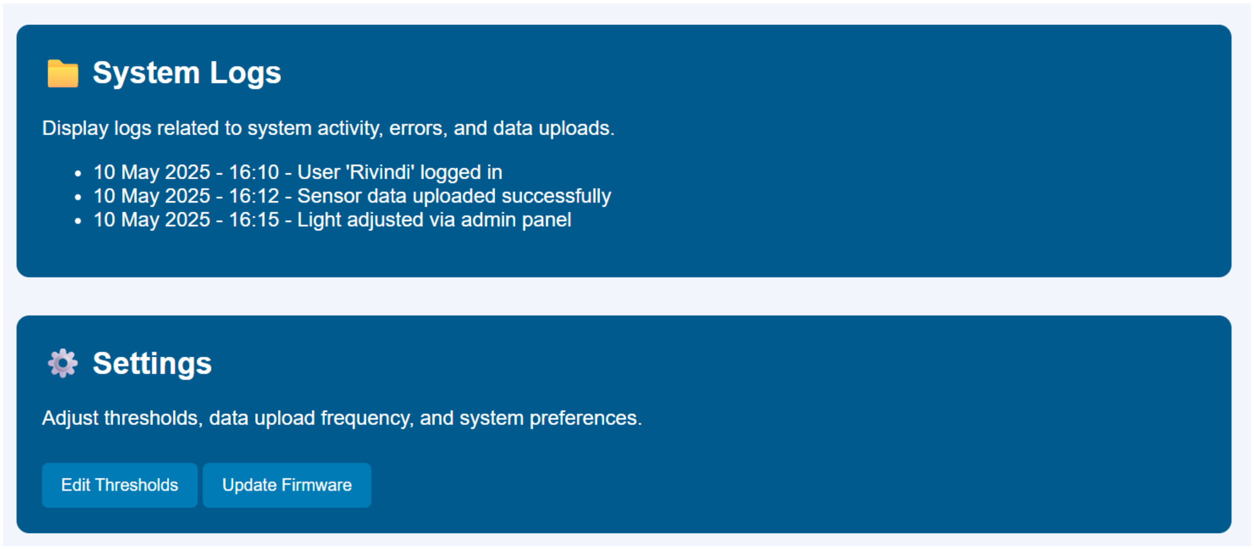
View, add, or remove access to registered users on the system.

Figure 45-admin panel I

Control Panel for Phytotron: Administrators can engage with the phytotron's hardware layer through the control panel, which enables them to manually or automatically activate actuators such as fans, irrigation pumps, and grow lights. Through GPIO-based relay modules, these controls are directly connected to the Internet of Things microcontroller (to the NodeMCU ESP8266). The microcontroller receives real-time commands from the admin interface and converts them into HTTP / MQTT requests, which cause the appropriate hardware components to be activated. One of the main benefits of IoT-enabled agricultural systems is this degree of automation and control.

Management of Users: Role-based access control, profile updates, authentication, and registration are among the user management features that administrators can access. This guarantees safe access and simplified system user roles.

Logs and Monitoring of Sensor Data: All historical and real-time sensor data, including temperature, humidity, soil moisture, and light intensity, are viewable by administrators and are time-stamped. System logs, including actuator trigger events and fault statuses, are also available.


System Logs

Display logs related to system activity, errors, and data uploads.

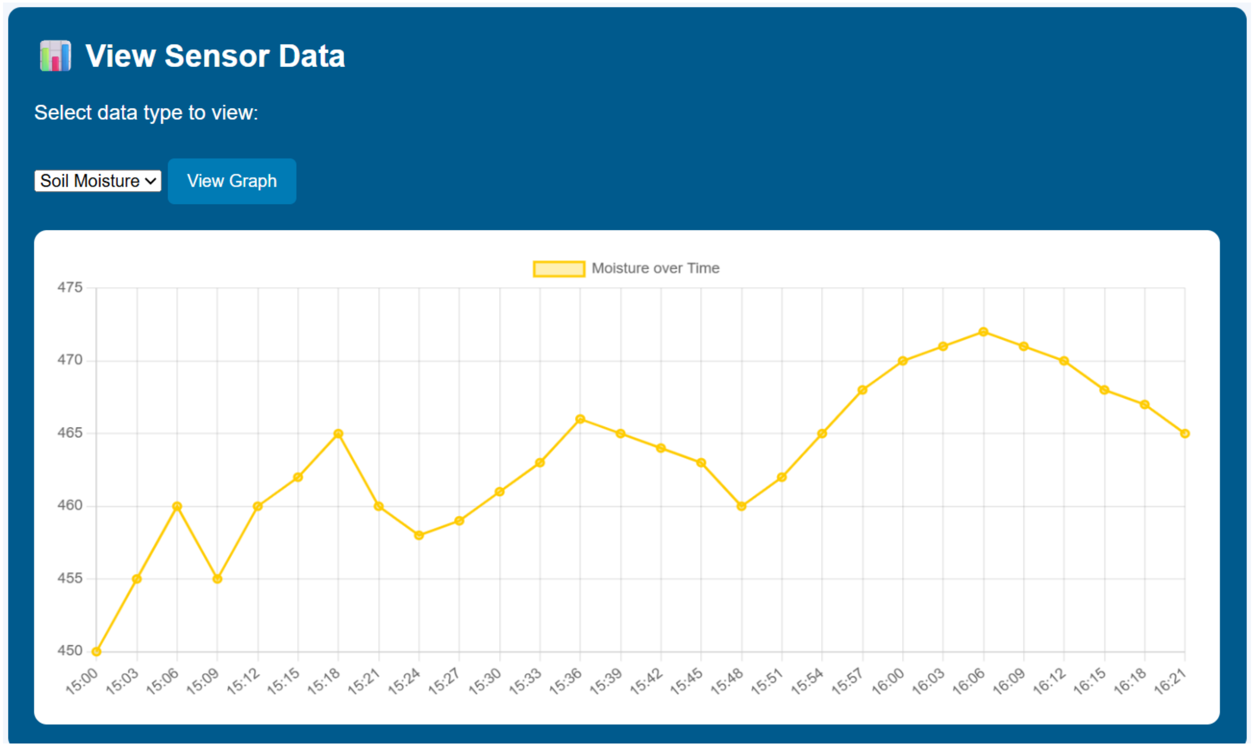
- 10 May 2025 - 16:10 - User 'Rivindi' logged in
- 10 May 2025 - 16:12 - Sensor data uploaded successfully
- 10 May 2025 - 16:15 - Light adjusted via admin panel

Settings

Adjust thresholds, data upload frequency, and system preferences.

Figure 46-admin panel II

Admin panel has feature to visualized the last hour data set in graphically by selecting the parameter (temperature, soil moisture, humidity) required. It helps admin to notify the changes related to the triggered alarms and also the provide ability to monitor the system conditions manually if there is any special scenario.


View Sensor Data

Select data type to view:

Figure 47- soil moisture inputs in graph

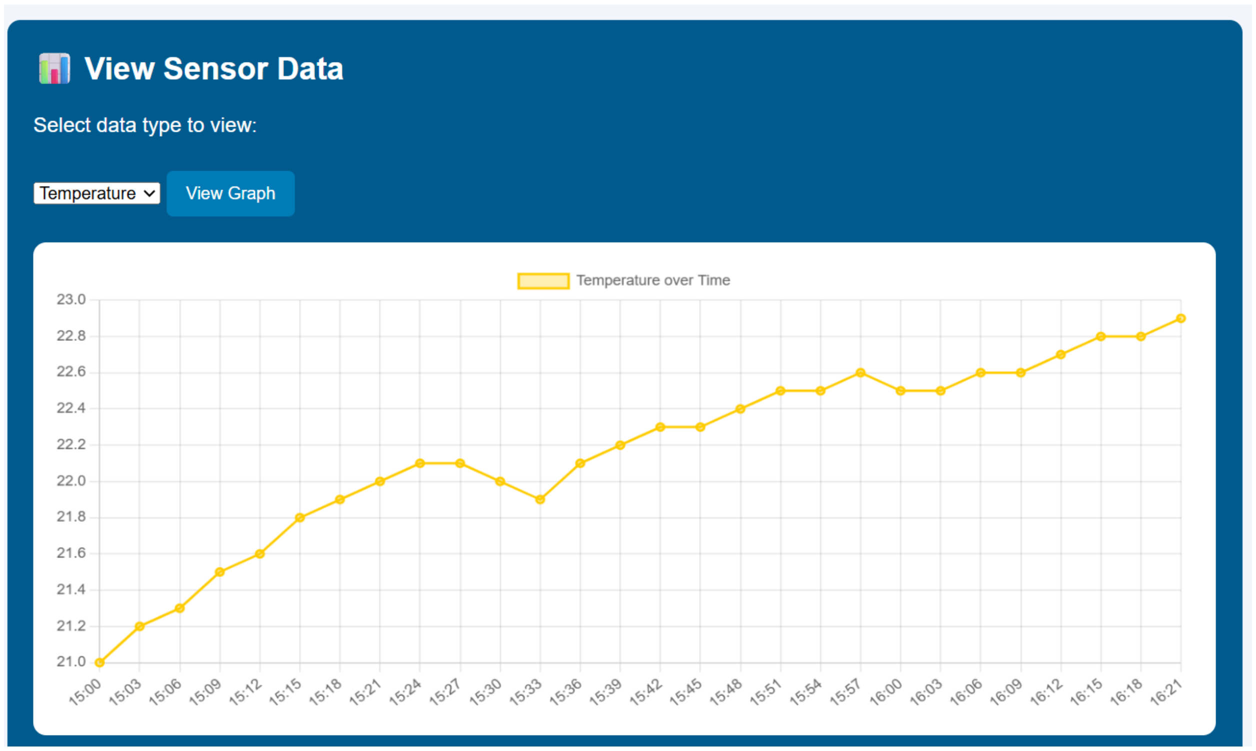
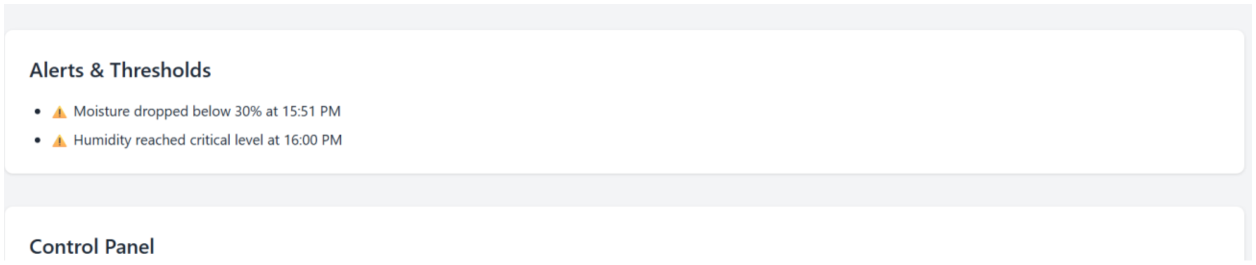


Figure 48-temperature measuring view by admin end



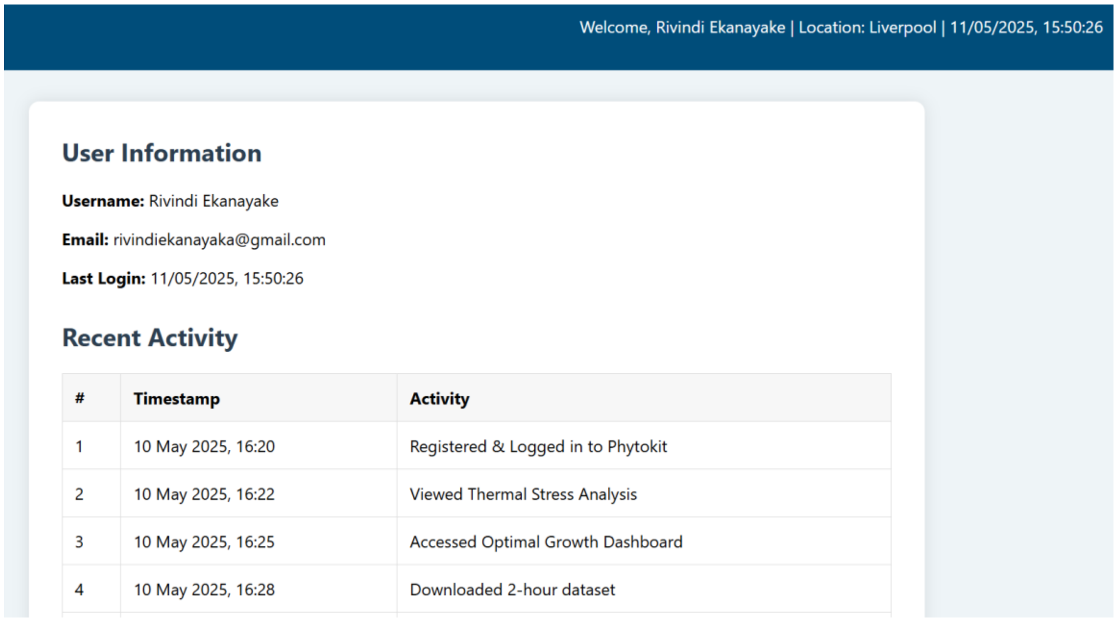
Alerts & Thresholds

- ⚠ Moisture dropped below 30% at 15:51 PM
- ⚠ Humidity reached critical level at 16:00 PM

Control Panel

Figure 49- latest alerts view

Shared Features (Admin and User)


Welcome, Rivindi Ekanayake | Location: Liverpool | 11/05/2025, 15:50:26

User Information

Username: Rivindi Ekanayake

Email: rivindiekanayaka@gmail.com

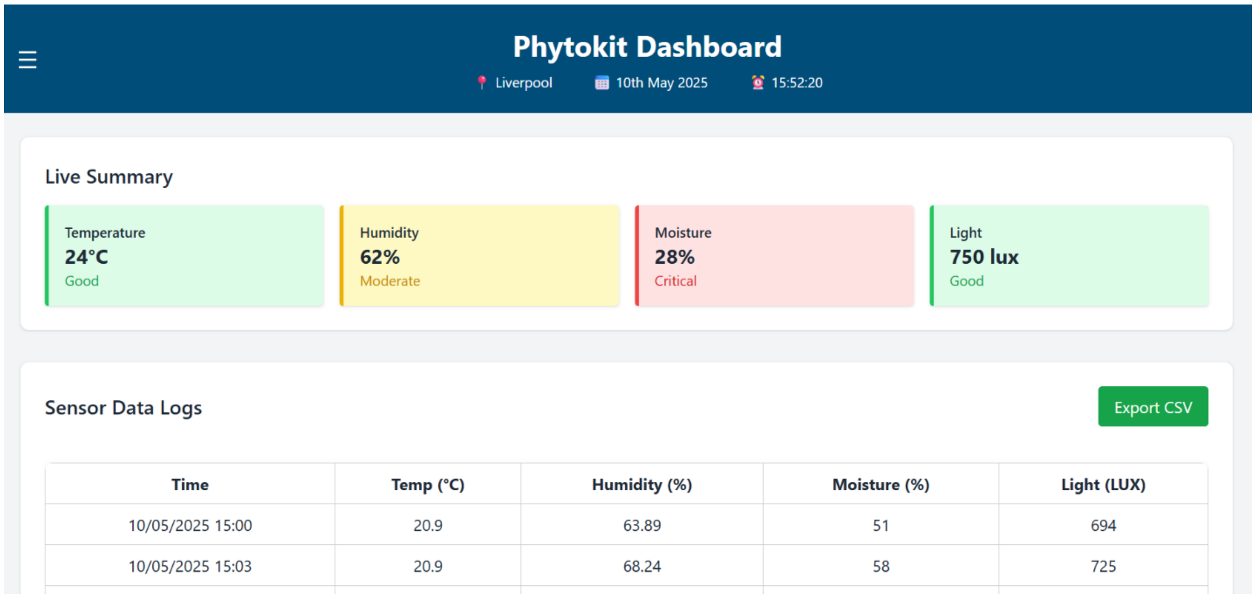
Last Login: 11/05/2025, 15:50:26

Recent Activity

#	Timestamp	Activity
1	10 May 2025, 16:20	Registered & Logged in to Phytokit
2	10 May 2025, 16:22	Viewed Thermal Stress Analysis
3	10 May 2025, 16:25	Accessed Optimal Growth Dashboard
4	10 May 2025, 16:28	Downloaded 2-hour dataset

Figure 50-user account n activity manager view- Realtime updates

Visualisation of General Sensor Input Using Graph-Based Analysis: Interactive charts allow users to view both historical trends and real-time sensor information. Pattern recognition and anomaly detection are made possible by the formats in which this data is shown, such as line and bar graphs. The combination of real-time data analytics with responsive web technologies is best demonstrated by the integration of data visualisation tools (Chart.js) with real-time IoT data sources. This gives consumers the ability to use sensor outputs to inform data-driven decisions.



Phytokit Dashboard

Liverpool 10th May 2025 15:52:20

Live Summary

Temperature	Humidity	Moisture	Light
24°C	62%	28%	750 lux
Good	Moderate	Critical	Good

Sensor Data Logs

Export CSV

Time	Temp (°C)	Humidity (%)	Moisture (%)	Light (LUX)
10/05/2025 15:00	20.9	63.89	51	694
10/05/2025 15:03	20.9	68.24	58	725

Figure 51-dashboard general view

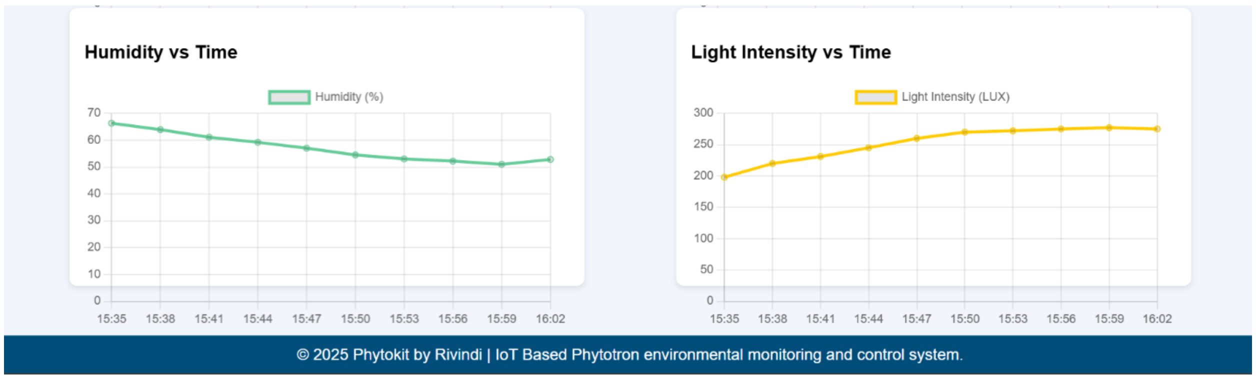
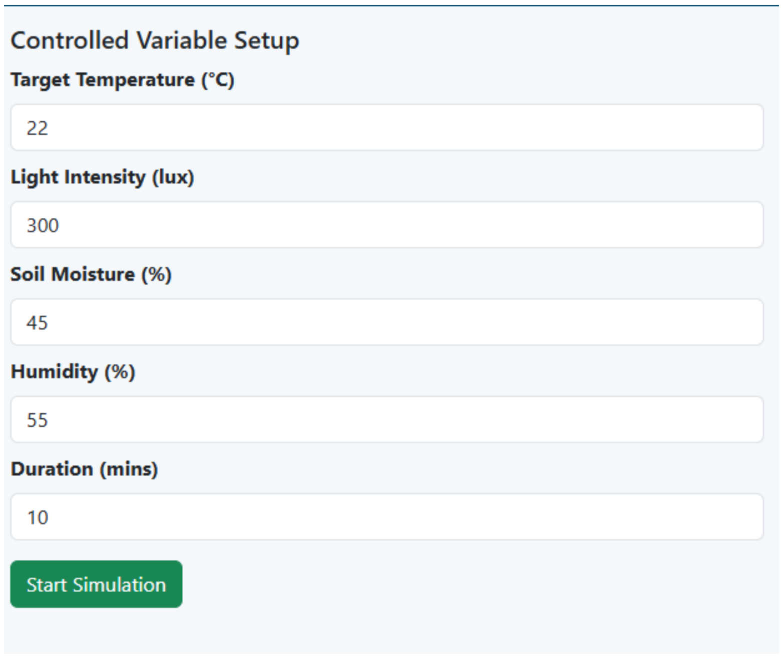


Figure 52-graphical visualization of the sensor inputs

Personalised Parameter Configurations: For environmental factors like temperature, humidity, soil moisture, and light intensity, users can establish unique thresholds by making request from the user end. This enables the phytotron to dynamically adjust to certain agricultural or experimental requirements upon the administrator review. A database contains these threshold values, which are then synchronised with the microcontroller by push notifications or scheduled polling. Predefined actions (such starting irrigation) are automatically carried out when actual sensor readings diverge from these criteria. In this situation of the project it is under the integration level and not working 100% as expected.


Controlled Variable Setup

Target Temperature (°C)

22

Light Intensity (lux)

300

Soil Moisture (%)

45

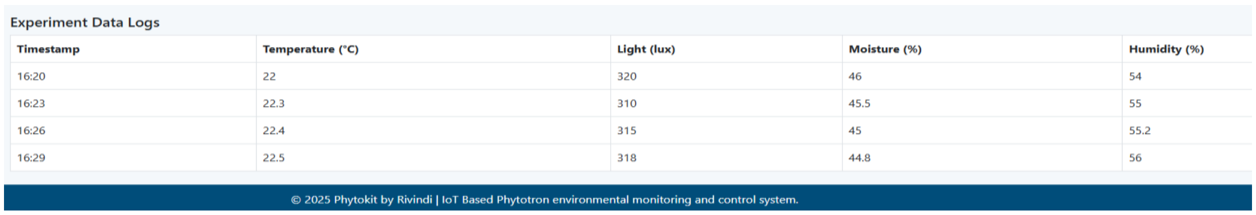
Humidity (%)

55

Duration (mins)

10

Figure 53-set the parameter values and the duration


Experiment Data Logs

Timestamp	Temperature (°C)	Light (lux)	Moisture (%)	Humidity (%)
16:20	22	320	46	54
16:23	22.3	310	45.5	55
16:26	22.4	315	45	55.2
16:29	22.5	318	44.8	56

© 2025 Phytokit by Rivindi | IoT Based Phytotron environmental monitoring and control system.

Figure 54-30 mins measured outputs

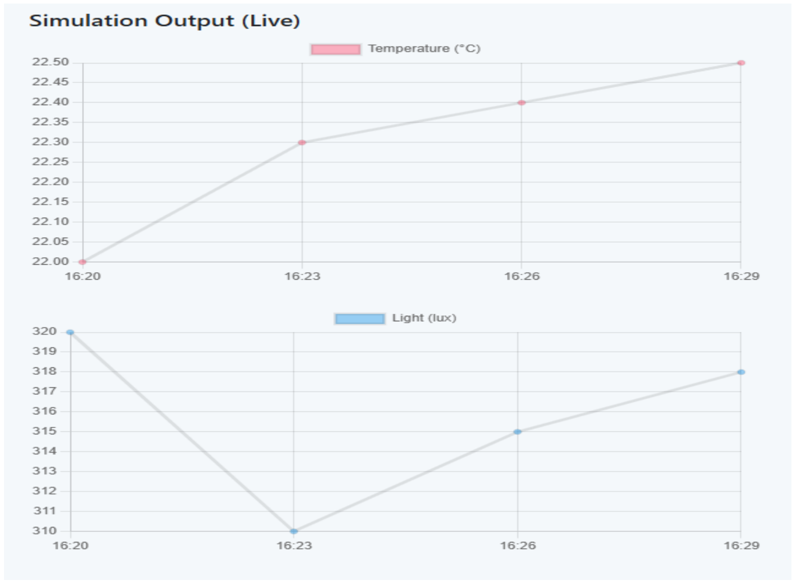


Figure 55-according to the values simulated graphical analysis output

Rather than the general view of the data this web application is targeting some special case studies related to the agricultural research activities.

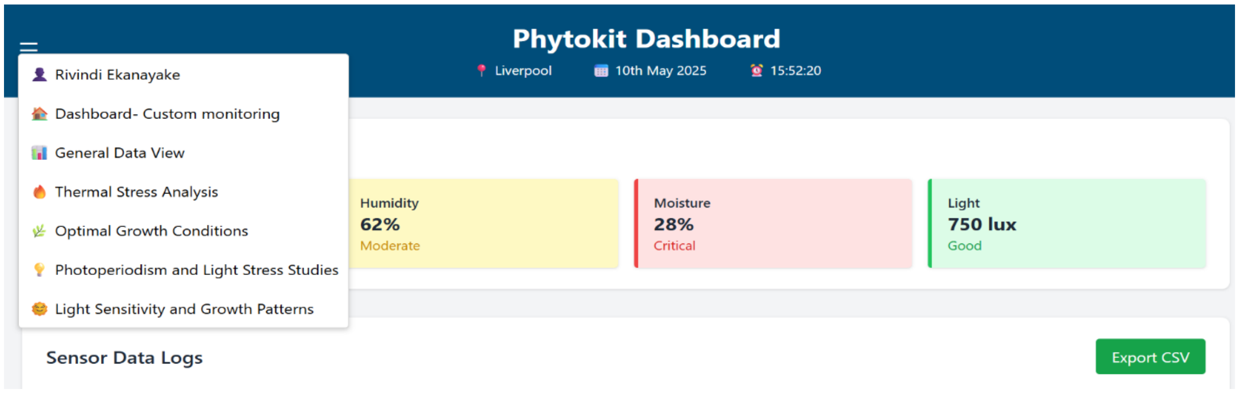
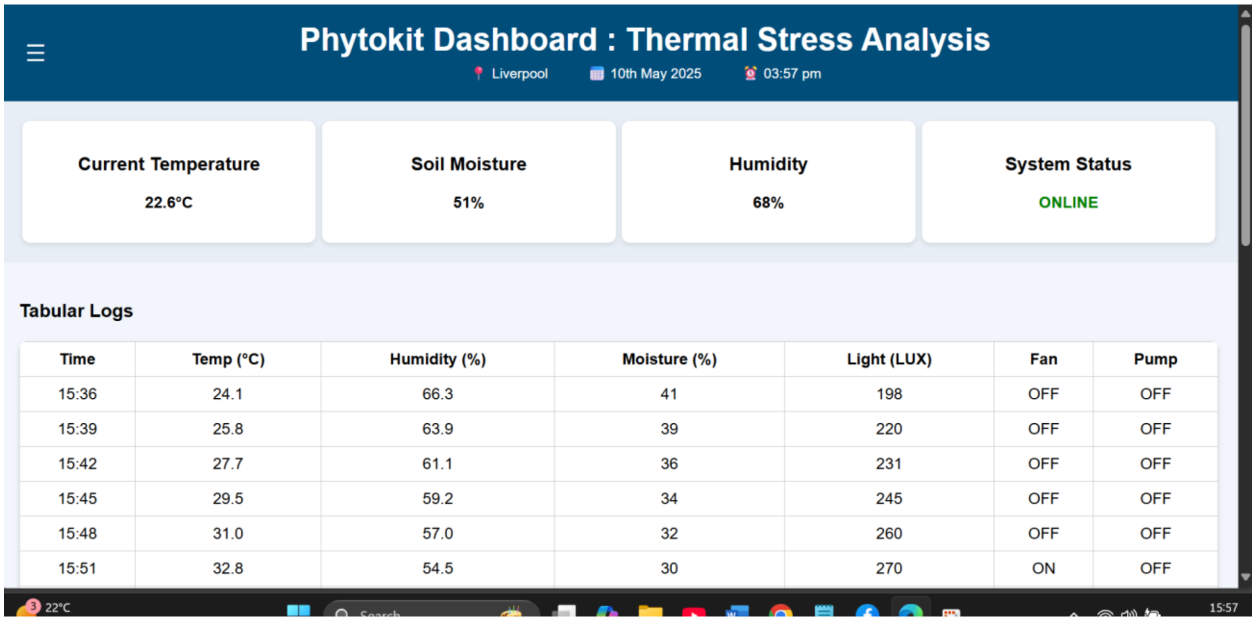


Figure 56-special case study menu

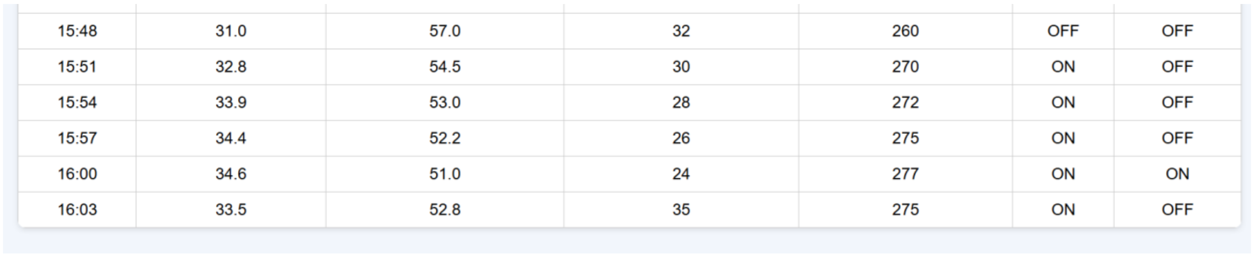
Unique Analytical Features

Analysis of Thermal Stress: The system detects and records instances in which heat stress affects plant development circumstances. This includes readings of consistently high temperatures and associated reductions in humidity. The sophisticated application of computational intelligence in precision agriculture is demonstrated by the combination of sensor data with thermal stress models (derived from plant physiological research). The server or edge device's implementation of rule-based algorithms and ongoing data collection enable this predictive capability.



Time	Temp (°C)	Humidity (%)	Moisture (%)	Light (LUX)	Fan	Pump
15:36	24.1	66.3	41	198	OFF	OFF
15:39	25.8	63.9	39	220	OFF	OFF
15:42	27.7	61.1	36	231	OFF	OFF
15:45	29.5	59.2	34	245	OFF	OFF
15:48	31.0	57.0	32	260	OFF	OFF
15:51	32.8	54.5	30	270	ON	OFF

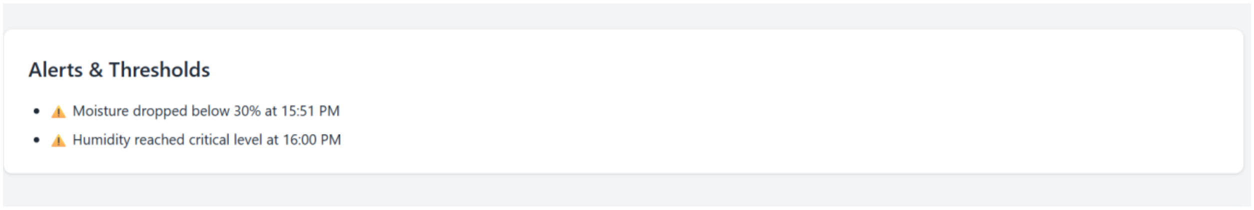
Figure 57-Realtime sensor data visualisation I



15:48	31.0	57.0	32	260	OFF	OFF
15:51	32.8	54.5	30	270	ON	OFF
15:54	33.9	53.0	28	272	ON	OFF
15:57	34.4	52.2	26	275	ON	OFF
16:00	34.6	51.0	24	277	ON	ON
16:03	33.5	52.8	35	275	ON	OFF

Figure 58-Realtime sensor data visualisation II

According to the sensor inputs system is able to automatically trigger the fan and water pump. Above image illustrates the on states of the Fan and also system generated alert to the system users when the system need attention.



Alerts & Thresholds

- ⚠️ Moisture dropped below 30% at 15:51 PM
- ⚠️ Humidity reached critical level at 16:00 PM

Figure 59-Moisture drop alert

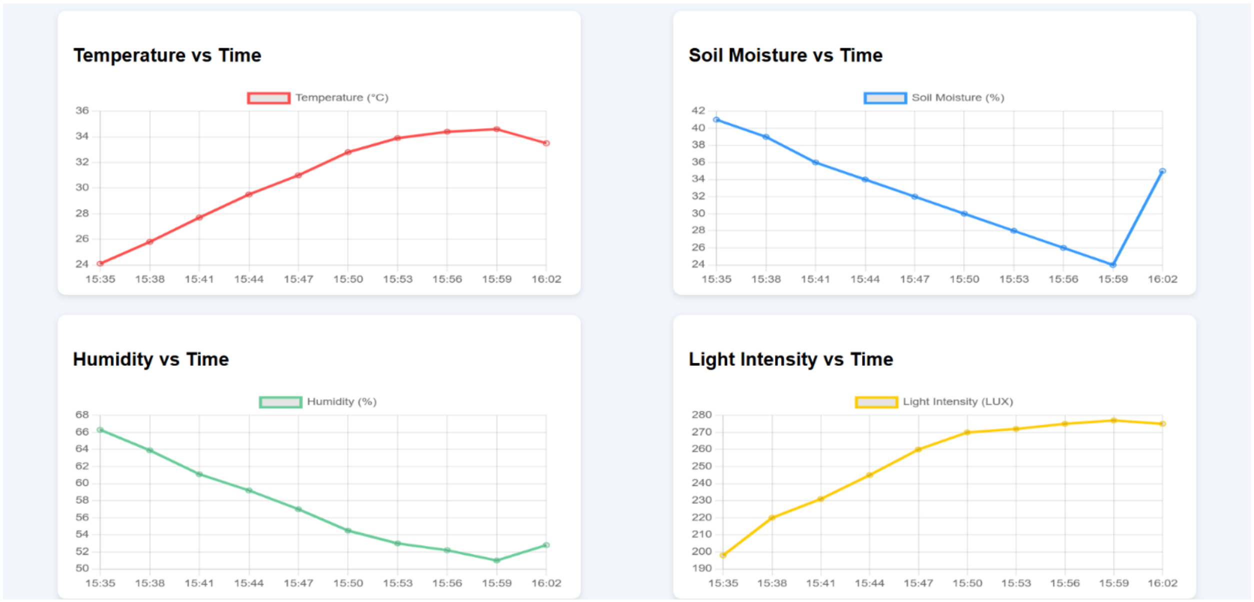


Figure 60-thermal stress analysis graphs generated by the application

Chapter 04 – Reflection and conclusion

This project is completed within limited time period with limited resources and because of that all the objectives and the all features that planned in the project design state are not performing 100%. But the accuracy and the other main features related to the IoT integration are completed.

4.1. Overview of significant of the Developed Application

With its integrated admin-user dashboard, this IoT-enabled phytotron system exemplifies the real-world use of intelligent environmental monitoring and precision agriculture. A combination of cloud computing, web-based interfaces, and microcontroller-based sensors represents advances in a number of technology domains:

- Using the NodeMCU for real-time sensing and actuation in embedded and Internet of Things systems.
- Cloud platforms: Making use of Firebase or comparable services for access control, synchronisation, and data storage.
- Web technologies: Creating safe and responsive web dashboards by combining HTML, CSS, JavaScript, and frameworks.
- Data-driven decision-making and autonomous control based on environmental feedback loops are made possible by automation and analytics.

Scalability, dependability, and real-time responsiveness are features of this technology. Because it reduces manual labour and the possibility of human error while facilitating remote observation, intervention, and analysis of historical data, it is particularly useful in controlled agricultural research.

4.2. Interface for Users Examining the Phytokit web Application

The Phytokit web application is a user-friendly, dynamic, and well-organized interface designed for an Internet of Things-based environmental monitoring system. It uses contemporary web technologies like HTML5, Tailwind CSS, and responsive layout concepts to ensure usability across platforms and devices. The dashboard divides key functionalities into visually different parts, including real-time temperature, humidity, moisture, and light monitoring, historical data presentation, and control panel for authorised users. The Research Tools section allows for custom notes and experimental tags. The user experience is facilitated by color-coded alerts, interactive menus, and hover effects. The system enforces role-specific UI controls, providing administrators more authority and general users view-only access to sensor analytics and real-time data. The interface maintains simplicity, making it easier for both technical and non-technical users. Real-time clock and position markers strengthen context awareness for data analysis.

4.3. IoT-Based Phytotron Systems Data Security

When designing and implementing IoT-based environmental monitoring systems, data security is crucial, especially if the system is connected to cloud platforms and accessible through web-based interfaces. Sensitive real-time environmental data, including temperature, humidity, soil moisture, and light intensity, are handled by the phytotron system created for this project. Both users and administrators can remotely access and log these data. The main security techniques used to safeguard data at the device level, during transmission, in cloud storage, and through the web application layer are described in this section.

4.3.1. Physical Device and Sensor Level Security

The physical phytotron system consists of several sensors and actuators that are linked to a NodeMCU (ESP8266) microprocessor. To reduce the danger of unauthorised physical entry or tampering:

- Secure Enclosures: All hardware components are enclosed in tamper-resistant enclosures to avoid physical manipulation or damage.
- Firmware hardening involves flashing the microcontroller firmware with digitally signed code, which prevents unauthorised reprogramming or malware injection.
- Reset/Boot Protections: Resistor-based designs safeguard GPIO boot control and reset pins against external force resets or unauthorised debugging.
- Device Authentication: Each NodeMCU is provided a unique identification (UID) and an API token to ensure secure communication with the cloud server.

4.3.2. Secure Data Transmission to the Cloud.

Sensor data from the NodeMCU is sent to a cloud platform (such as Firebase Realtime Database or AWS IoT Core) for remote storage and visualisation. The following methods are used to safeguard this data flow:

- TLS/SSL Encryption: All communication between the NodeMCU and the cloud is encrypted with HTTPS (TLS v1.2), which ensures data confidentiality and integrity throughout transmission.
- Token-Based Access Control: To communicate with cloud storage, API keys and authentication tokens are securely kept in the microcontroller's flash memory.
- Data Validation: On the cloud backend, input sanitisation and range validation processes are in place to reject any malformed or questionable data that may be injected during transmission.

4.3.3. Cloud Storage and Access Security

Temperature, humidity, and soil moisture are all recorded in real time to a cloud database for later retrieval and analysis. To ensure the security of this stored information:

- Role-Based Access Control (RBAC): Database access is controlled by stringent user roles (Admin/User), allowing users to only examine their own data while administrators can handle all data points.
- Read/Write Rules: Firebase Realtime Database security rules specify who can read and write to specified database nodes.
- Database Backup and Recovery: Scheduled cloud backups protect against data loss and ensure continuity in the case of a system failure or breach.

4.3.4. Web Application Security

The web-based dashboard used to retrieve and visualise sensor data has many levels of security safeguards to prevent unauthorised access or data leakage:

- User Authentication: The system uses email/password authentication for both administrators and users. Future enhancements could include OAuth integration (e.g., Google Sign-In) for more robust identity verification.

- Session Management: To decrease the possibility of session hijacking, user sessions are handled using secure cookies with short-lived tokens and automatic logout upon inactivity.
- HTTPS Deployment: The web application is hosted over HTTPS to encrypt data transmitted between the browser and the server.

4.3.5. Secure architecture's significance.

To ensure accurate research data, preserve system trust, and stop operational manipulation (such as unapproved control of fans, pumps, or LEDs), it is essential to secure the data lifecycle, which includes physical sensing, transmission, cloud storage, and web access. Data privacy and security compliance (e.g., GDPR, ISO/IEC 27001) is legally necessary in research or commercial deployments where IoT-based phytotrons may be used for sensitive investigations or high-value crops.Through the use of contemporary encryption, role-based access, and secure communication protocols, this solution ensures data authenticity, availability, and secrecy—all necessary for any trustworthy IoT application.

4.4. Testing

This section reflected in the testing and implementation scenarios and process done within the Smart Miniature Growing System web application. In first, the methods that have been used for testing the system before implementation will be taken into consideration with the testing of all the modules and their functionalities. Then, the implementation process will be taken into discussed.

4.4.1. System application testing

System testing has been used to identify the full functionality of the system. Also, it is used to identify if the system is processing well without any error or downtime. System testing may help the software product to fulfill all the functioning against business requirements. Proper testing will provide insurance to the system that it will perform as necessary without any inconvenience to the user and it can result in an error-free well-functioned system.

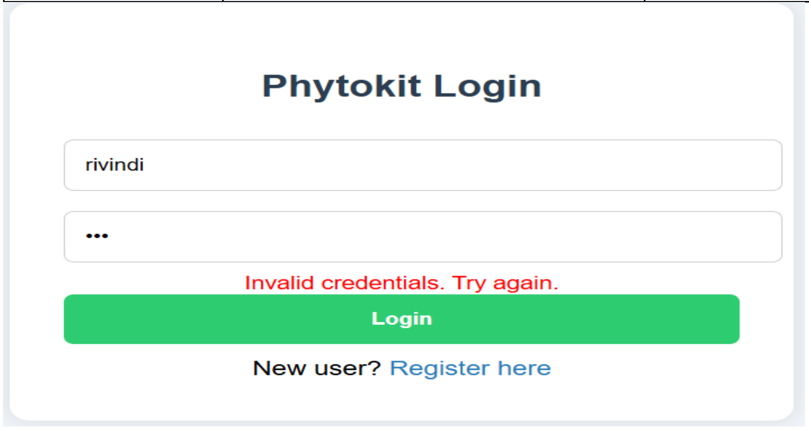
Unit Testing

Unit testing is a software development process where the distinct unit or cluster of connected units or smallest testable parts of an application. It can be done manually and must be frequently done by the programmer to check that the unit has given its output according to the given input and has been developed according to the specifications. Unit testing has reduced the number of errors and bugs in the system and has discovered usability problems in earlier phases. To get the help of unit testing it must be combined with other testing techniques. When the units in a program are being worked in an error-free manner, possible larger components of the program can be evaluated utilizing integration testing.

- Sensor Modules: Accurate output was verified with calibrated measuring devices.
- Actuator Controls: Trigger logic was tested in a range of simulated scenarios.
- Elements of the User Interface: Verified that buttons, forms, and charts all reacted appropriately to input.

Table 1- Test case I

Test Case	Description	Result
Login and Authentication	This module allows registered and authorized users to log in to the system. It is integrated with the cloud database to verify credentials and ensure secure access.	Login was successful when correct username and password were entered. Login failed when incorrect credentials were entered, and an error message appeared.

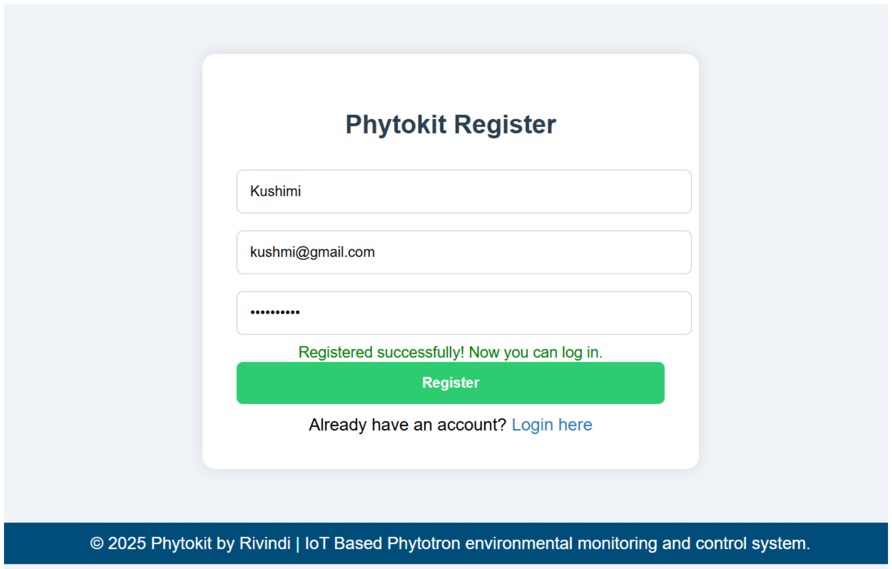

Phytokit Login

rivindi

•••

Invalid credentials. Try again.

Figure 61- logging credential validation


Phytokit Register

Kushimi

kushmi@gmail.com

••••••••••

Registered successfully! Now you can log in.

Register

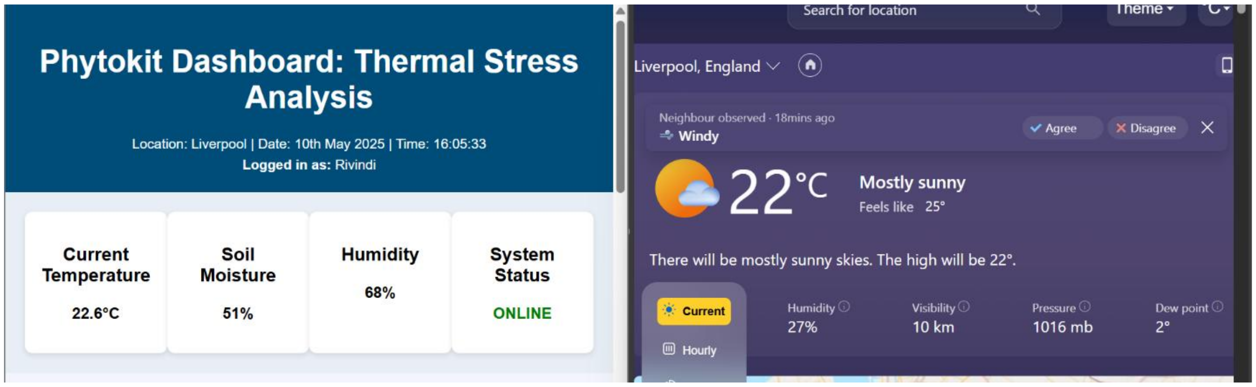
Already have an account? Login here

© 2025 Phytokit by Rivindi | IoT Based Phytotron environmental monitoring and control system.

Figure 62- new user registration

Table 2-Test case II

Test Case	Description	Result
Sensor Data Reading	This module enables users to view real-time sensor data via the web application interface. Sensor values are compared with manual readings under the same conditions.	Most sensor readings matched manual readings. Occasionally, some sensor readings deviated from the manual ones.


Phytokit Dashboard: Thermal Stress Analysis

Location: Liverpool | Date: 10th May 2025 | Time: 16:05:33

Logged in as: Rivindi

Current Temperature 22.6°C

Soil Moisture 51%

Humidity 68%

System Status ONLINE

Liverpool, England

Neighbour observed · 18mins ago

Windy

22°C Mostly sunny

Feels like 25°

There will be mostly sunny skies. The high will be 22°.

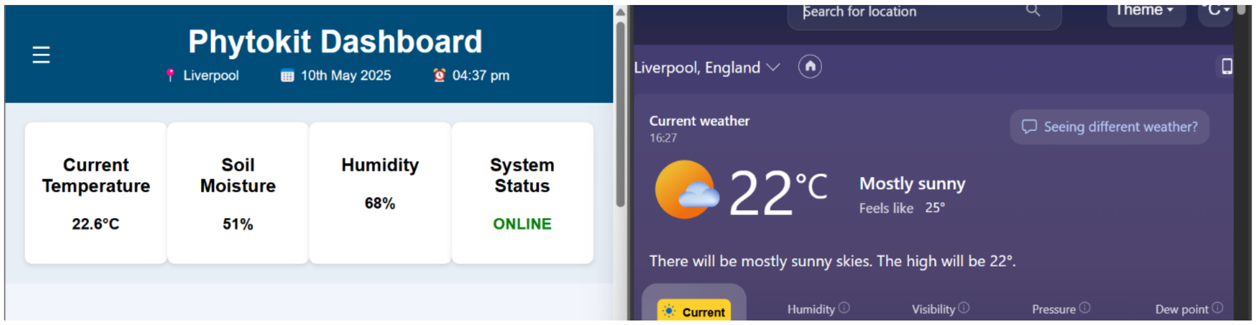
Humidity 27%

Visibility 10 km

Pressure 1016 mb

Dew point 2°

Figure 63- Sensor reading of the temperature



Phytokit Dashboard

Liverpool 10th May 2025 04:37 pm

Current Temperature	Soil Moisture	Humidity	System Status
22.6°C	51%	68%	ONLINE

Search for location Theme °C

Liverpool, England

Current weather
16:27

22°C Mostly sunny
Feels like 25°

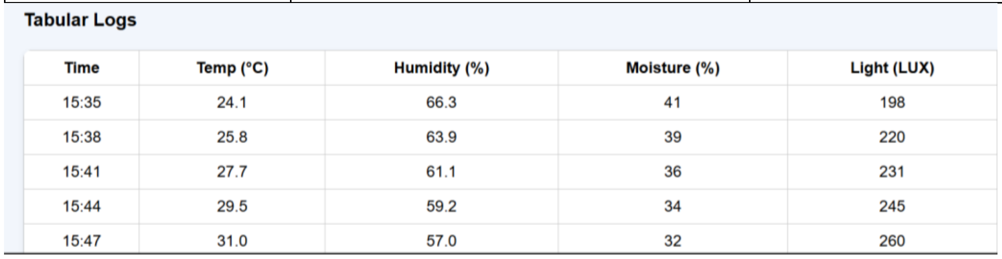
There will be mostly sunny skies. The high will be 22°.

Current Humidity Visibility Pressure Dew point

Figure 64- Sensor reading of the temperature II

Table 3-Test case III

Test Case	Description	Result
Statistical Information	This module generates and displays statistical graphs of sensor data. System-generated graphs were compared with manually created ones to verify accuracy.	Most of the time, system-generated graphs matched the manual graphs.Minor differences were noticed when sensor data didn't match manual input.



Tabular Logs

Time	Temp (°C)	Humidity (%)	Moisture (%)	Light (LUX)
15:35	24.1	66.3	41	198
15:38	25.8	63.9	39	220
15:41	27.7	61.1	36	231
15:44	29.5	59.2	34	245
15:47	31.0	57.0	32	260

Figure 65- Sensor inputs

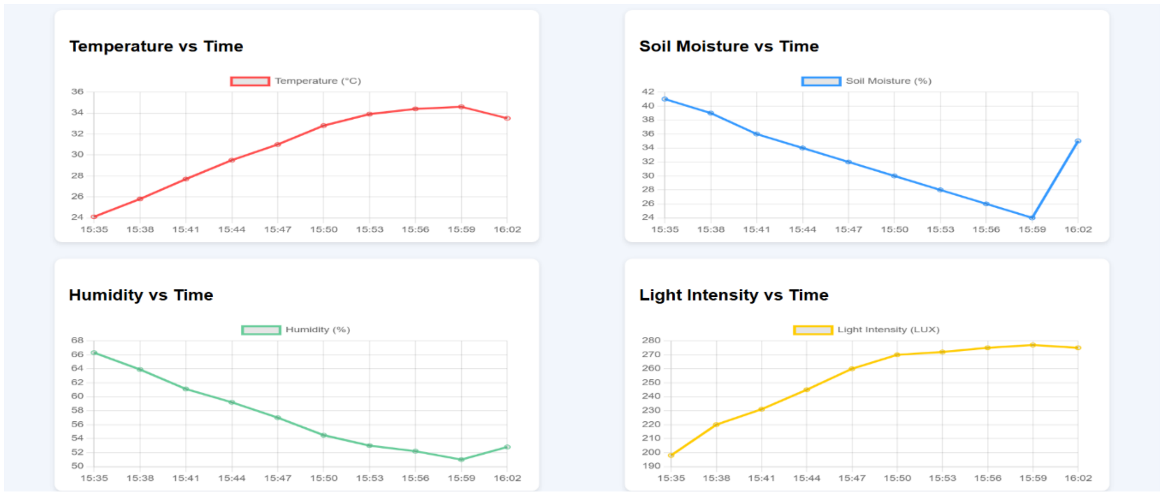


Figure 66-Generated graph

Integration Testing

Integration testing is a systematic software testing technique used in Extreme Programming to construct a program structure and tested individual units together. It aims to simplify error localization and detect interface errors. Integration testing involves building a system from its own components and testing for issues arising from component interactions. There are various approaches to integration testing, including the big bang approach, which includes top-down, bottom-up, incremental, and sandwich approaches.

Table 4-Integration testing test cases

Test Case	Description	Result
User Interface Testing	This test involved verifying navigation through each button and page in the web application using both top-down and bottom-up approaches.	- No errors were found in the linkages of the web interfaces. - All user interface elements, buttons, and process flows worked as required.
Use Scenario Testing	Testing involved going through every use case scenario defined in the system to ensure all user paths and workflows function as expected.	- All the test scenarios functioned properly.
Data Flow Testing	Tested the connectivity between modules and the application's database.	- Data flow within the system functioned properly and accurately. - No errors were encountered.
System Interface Testing	Checked the connectivity and interaction between all system components and modules to ensure proper integration and seamless operation.	- The entire system functioned correctly at the interface level.

4.5. Limitations and Challenges

The developed phytotron system was successful in general but it had a number of drawbacks:

- Limited Hardware Accessibility: Due to my status as an international student living in university housing, I had very limited access to cutting-edge IoT sensors and lab testing apparatus. This restriction limited hardware research and limited the variety of sensor deployment.
- Incapacity to Perform Manual Calibration: Sensors were used to monitor parameters including soil moisture and humidity, but it was challenging to confirm sensor accuracy because there were no reference devices or expert calibration tools available. For applications of a research calibre, this restricts the system's accuracy.
- Hosting Restrictions: The web application's cloud hosting and data retention features were limited because to budgetary limitations and the requirement to use free-tier or limited-time providers.
- Time-Bound Development Cycle: Because the project was created within the academic parameters of an MSc program, there were time restrictions on the implementation of advanced features, comprehensive testing, and iterations.
- Infrastructure and Logistics Restrictions: Living on campus presented additional challenges with regard to space, electricity supply, and sustaining the physical phytotron configuration for prolonged periods of time.

Understanding these limitations offers important insight into the real-world difficulties of implementing IoT-based phytotron systems in settings with limited resources. These issues can be resolved in future research by utilising affordable cloud options for sustainable hosting and growth, forming institutional alliances, and gaining access to improved testing environments.

4.6. Future Developments

Several enhancements are planned to broaden the system's capabilities and suitability for larger agricultural and research needs:

- Integrating carbon dioxide (CO_2) sensors (e.g., MG-811 or MH-Z19B) can provide essential data on air composition inside the phytotron. This feature will allow for more precise manipulation of photosynthetic efficiency and plant respiration cycles, particularly for high-yield or research-grade crops.

- Automatic Light Sensitivity Control: By integrating LDR data with configurable light intensity thresholds, the system may dynamically change artificial lighting (for example, LED grow lights) via PWM or relay-based dimmers. This simulates natural light cycles or creates unique photoperiods for different plant species.

The topic of IoT-based environmental automation has a solid foundation for future study and advancement thanks to this effort. Its open architecture, cost-effectiveness, and extensibility make it a prime contender for practical use in academic research, smart farming innovation, and agriculture.

4.7. Conclusion

The field of precision agriculture and controlled-environment plant research have advanced significantly with the creation of the Internet of Things-based phytotron environmental monitoring and control system. The NodeMCU ESP8266, DHT22, capacitive soil moisture sensors, LDRs, relay-controlled actuators, and other inexpensive hardware components are used in this system to provide real-time monitoring, automated environmental adjustments, and data visualisation via an integrated online interface. The design, which makes use of platforms like Firebase and ThingSpeak to enable data transmission, storage, and analytics, exemplifies the collaboration between cloud computing and IoT technology. The system gives researchers and agriculturists the ability to effectively and remotely optimise growing conditions with features including user authentication, experiment logging, graphical reporting, and dashboards that are available on mobile devices.

The project also ran into limitations with sensor accuracy, device scalability, and sophisticated analytics capabilities, despite its strong core functionality. Resolving these issues strengthens the project's potential as a modular and scalable solution while also creating chances for future improvement.

Chapter 06 – References

Anon., 2024. *Smart Greenhouse Market By Type, By Offerings, By Component (HVAC Systems, LED Grow Lights, Irrigation Systems, Valves & Pumps, Sensor & Control Systems, and Others), By End User - Global Industry Outlook, Key Companies (Cultivar, Kubo Greenhouse, Prospia.* [Online]
Available at: https://dimensionmarketresearch.com/report/smart-greenhouse-market/
[Accessed 06 05 2025].

Anon., 2025. *Smart Greenhouse Market by Type (Hydroponics and Non-Hydroponics), Covering Material Type (Polyethylene, Polycarbonate, and Others), Offering (Hardware and Software & Services), Component, Cultivation, End User, Region - Global Forecast to 2025.* [Online]
Available at: https://www.marketsandmarkets.com/Market-Reports/smart-greenhouse-market-63166169.html?utm_source=chatgpt.com
[Accessed 06 05 2025].

Anon., 2025. *Wireless Sensor Networks.* [Online]
Available at: https://www.monolithicpower.com/en/learning/mpscholar/sensors/advanced-topics-in-sensing/wireless-sensing-networks
[Accessed 06 05 2025].

Cangqing Wang, Jiangchuan Gong, 2025. *Intelligent Agricultural Greenhouse Control System Based on Internet of Things and Machine Learning.* [Online]
Available at: https://arxiv.org/abs/2402.09488?utm_source=chatgpt.com
[Accessed 06 05 2025].

Carole H. Saravitz, Robert J. Downs, Judith F. Thomas , 2009. *PHYTOTRON PROCEDURAL MANUAL - For Controlled-Environment Research at the Southeastern Plant Environment Laboratory ,* s.l.: North Carolina State University North Carolina Agricultural Research Service.

Jaiswal, A., Niwas, R., & Kumar, A. , 2020. Smart Phytotron: IoT-based system for monitoring and controlling plant growth environment.. *International Journal of Advanced Science and Technology,* Volume 29(7), p. 12778–12790.

Joaquin Gutierrez, Juan Francisco Villa Medina ,Alejandra Nieto-Garibay,Miguel Angel Porta-Gándara, 2014. Automated Irrigation System Using a Wireless Sensor Network and GPRS Module. *Research Gate.*

Odoi-Lartey, B., & Danso, E., 2018. Improving Agricultural Production using Internet of Things (IoT) and Open Source Technologies.. *International Journal of Computer Applications,* p. 36–42.

Phytotron, N. S. U., n.d. *Research and Other Articles – NC State University Phytotron.* [Online] Available at: https://phytotron.ncsu.edu/research/
[Accessed 06 05 2025].

Singh, A., Soni, R., & Tiwari, A. , 2022. Energy Optimization in Smart Phytotrons Using IoT and AI Technologies.. *Journal of Cleaner Production,* Volume 338.

Sjaak Wolfert, Lan Ge,Cor Verdouw, Marc-Jeroen Bogaardt, 2017. *Big-Data in Smart Farming– A review,* Wageningen: Science Direct.

USDA, 2022. *Controlled Environment Agriculture. U.S. Department of Agriculture..* [Online] Available at: https://www.usda.gov/media/blog/2022/01/13/controlled-environment-agriculture
[Accessed 04 05 2025].

Xing Yang,Lei Shu, Jianing Chen, Mohamed Amine Ferrag, 2021. A Survey on Smart Agriculture: Development Modes, Technologies, and Security and Privacy Challenges. *JOURNAL OF AUTOMATICA SINICA,* 08(http://ieeexplore.ieee.org), p. 30.

Xuelong Hua, Yang Liua, Zhengxi Zhaob, Jintao Liub, Xinting Yangb, Chuanheng Sunb, Shuhan Chena, Bin Lie, Chao Zhoub, 2021. Real-time detection of uneaten feed pellets in underwater images for aquaculture using an improved YOLO-V4 network. *Elsevier B.V,* 185(3April2021), p. 11.

Appendix

7.1 Node MCU code

```
#ifdef ESP32

  #include <WiFi.h>

  #include <HTTPClient.h>

#else

  #include <ESP8266WiFi.h>

  #include <ESP8266HTTPClient.h>

  #include <WiFiClient.h>

#endif

#include <Wire.h>

#include "DHT.h"

#define DHTPIN 2

#define DHTTYPE DHT21  // DHT 21

DHT dht(DHTPIN, DHTTYPE);

const char* ssid = "TP-LINK_CA2988";  //ssid

const char* password = "******";      //password

const char* serverName = "http://www.rptronic.lk/post-esp-data.php";


String apiKeyValue = "tPmAT5Ab3j7F9";  //api key

float humidity, temperature;

int sensor_pin = A0;
int output_value;
void setup() {
  Serial.begin(115200);
  WiFi.begin(ssid, password);
  Serial.println("Connecting");
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.print("Connected to WiFi network with IP Address: ");
  Serial.println(WiFi.localIP());
  dht.begin();
}
void loop() {
  Read_h_t();
  soilMoisture();
  dataUpload();
}
void Read_h_t() {
  humidity = dht.readHumidity();
  temperature = dht.readTemperature();
  Serial.println(humidity);
```

```
}
void soilMoisture() {
  output_value = analogRead(sensor_pin);
  delay(1000);  // Keep this delay to stabilize the analog reading
}
void dataUpload() {
  // Check WiFi connection status
  if (WiFi.status() == WL_CONNECTED) {
    HTTPClient http;
    // Start connection to server
    http.begin(serverName);
    // Specify content-type header
    http.addHeader("Content-Type", "application/x-www-form-urlencoded");
    // Prepare HTTP POST request data
    String httpRequestData = "api_key=" + apiKeyValue +
                             "&humi=" + String(humidity) +
                             "&temp=" + String(temperature) +
                             "&mosit=" + String(output_value) +
                            "&light=" + "50";  // Replace "50" with actual light sensor value if available
    Serial.println("HTTP Request Data: " + httpRequestData);
    // Send HTTP POST request
    int httpResponseCode = http.POST(httpRequestData);
    if (httpResponseCode > 0) {
      Serial.print("HTTP Response code: ");
      Serial.println(httpResponseCode);
      // Optional: Print the server response for debugging
      String response = http.getString();
      Serial.println("Server Response: " + response);
    } else {
      Serial.print("Error code: ");
      Serial.println(httpResponseCode);
    }
    // Free resources
    http.end();
  } else {
    Serial.println("WiFi Disconnected");
  }
  // Delay for 180 seconds (3 minutes)
  delay(180000);
}
```

7.2 Motor Controller code

```
#include <SoftwareSerial.h>
#include <ArduinoJson.h>
#include <AccelStepper.h>
AccelStepper stepper(AccelStepper::DRIVER, 9, 8);
SoftwareSerial s(5, 6);
```

```
int spd = 1000;
int sign = 1;
int ldr;
int x = 0;
int data1 = 0, data2 = 0;
void setup() {
  Serial.begin(115200);
  s.begin(115200);
  stepper.setMaxSpeed(1000);
  stepper.setSpeed(1000);
  pinMode(12, OUTPUT);
  pinMode(10, OUTPUT);
  datarecv();  // Initial JSON data read
}
void loop() {
  ldr = analogRead(A0);
  Serial.println(ldr);
  if (ldr < 300) {
    stepper.setSpeed(0);
    stepper.runSpeed();
    watering();
  } else {
    x = 0;
    stepper.setSpeed(40);
    stepper.runSpeed();
  }
}
void watering() {
  digitalWrite(12, LOW);  // Activate pump
  delay(data2);
  digitalWrite(12, HIGH);  // Deactivate pump
  delay(100);
  stepper.setSpeed(100);
  stepper.runSpeed();
}
void datarecv() {
  StaticJsonBuffer<1000> jsonBuffer;
  JsonObject& root = jsonBuffer.parseObject(s);
  if (!root.success()) {
    Serial.println("JSON parse failed");
    return;
  }
  data1 = root["data1"];
  Serial.println(data1);
  data2 = root["data2"];
  Serial.println(data2); }
```

7.3 Php script to insert data to the Database

```php
<?php

$servername = "123.45.67.89"; // IP address of external DB server



$dbname = "phytokit";

$username = "kaushini";

$password = "Kaushi@1234";

// API key for validating requests from NodeMCU/ESP devices

$api_key_value = "tPmAT5Ab3j7F9";

// Variables to store POST data

$api_key = $sensor = $location = $value1 = $value2 = $value3 = "";

// Validate if data is posted via HTTP POST method

if ($_SERVER["REQUEST_METHOD"] == "POST") {

    $api_key = test_input($_POST["api_key"]);

    // Check if the API key matches

    if ($api_key == $api_key_value) {

        // Collect and sanitize POST values

        $sensor   = test_input($_POST["sensor"]);

        $location = test_input($_POST["location"]);

        $value1   = test_input($_POST["value1"]); // e.g., humidity

        $value2   = test_input($_POST["value2"]); // e.g., temperature

        $value3   = test_input($_POST["value3"]); // e.g., soil moisture
        // Create DB connection
        $conn = new mysqli($servername, $username, $password, $dbname);
        // Check DB connection
        if ($conn->connect_error) {
           die("Connection failed: " . $conn->connect_error);
        }
        // SQL INSERT statement
        $sql = "INSERT INTO SensorData (sensor, location, value1, value2, value3)
              VALUES ('$sensor', '$location', '$value1', '$value2', '$value3')";
        if ($conn->query($sql) === TRUE) {
           echo "New record created successfully";
        } else {
           echo "Error: " . $sql . "<br>" . $conn->error;
        }
        // Close DB connection
        $conn->close();
     } else {
        echo "Wrong API Key provided.";
     }
  } else {
     echo "No data posted with HTTP POST.";
  }
```

```
// Sanitize function
function test_input($data) {
   $data = trim($data);
   $data = stripslashes($data);
   $data = htmlspecialchars($data);
   return $data;
}
?>
```

7.4 Php Script to review data in database

```
<!DOCTYPE html>
<html><body>
<?php
$servername = "123.45.67.89"; // IP address of external DB server
//  Database connection details
$dbname = "phytokit";
$username = "kaushini";
$password = "Kaushi@1234";
// Create connection
$conn = new mysqli($servername, $username, $password, $dbname); // Check connection
if ($conn->connect_error) {
die("Connection failed: " . $conn->connect_error);
}
```

```
$sql = "SELECT id, sensor, location, value1, value2, value3, reading_time FROM SensorData
ORDER BY id DESC";

echo '<table cellspacing="5" cellpadding="5"> <tr>

<td>ID</td>

<td>Sensor</td>

<td>Location</td>

<td>Value 1</td>

<td>Value 2</td>

<td>Value 3</td>

<td>Timestamp</td>

</tr>';

if ($result = $conn->query($sql)) {

while ($row = $result->fetch_assoc()) {

$row_id = $row["id"];

$row_sensor = $row["sensor"];

$row_location = $row["location"];

$row_value1 = $row["value1"];

$row_value2 = $row["value2"];

$row_value3 = $row["value3"];

$row_reading_time = $row["reading_time"];

echo '<tr>

<td>' . $row_id . '</td>

<td>' . $row_sensor . '</td>
<td>' . $row_location . '</td>

<td>' . $row_value1 . '</td>

<td>' . $row_value2 . '</td>

<td>' . $row_value3 . '</td>

<td>' . $row_reading_time . '</td>

</tr>';

}

$result->free();

}

$conn->close();

?>

</table>

</body>

</html>
```