Future of Farming: Smart Farming
–
A Guide to Creating A Greenhouse Farming System.

Muhammad Mohtashim Sadiq,

M00904156

October 2023

A thesis submitted in partial fulfilment of the requirements and need for an MSc in Mechatronic Systems Engineering

Supervisor(s): PDE 4519 – MSc Thesis Prof Michael Heaney

Abstract:

This paper investigates a greenhouse farming system based on an ESP32 microcontroller. The greenhouse monitoring and control system employs an ESP32 microcontroller as its central node, interfaced with various sensors and actuators and integrated with a Raspberry Pi-based graphical user interface (GUI). The viable and functional system demonstrates the capabilities needed to create and upscale into a full-fledged system. A holistic and design-level thinking approach is applied to dissect the architecture and identify bottlenecks and pathways of creating a Greenhouse system. This enabled the implementation of targeted optimizations in the code, enhancing the system's overall performance without introducing new hardware or protocols and identifying things necessary for healthy plant growth. The paper thus serves as a comprehensive guide for implementing efficient and reliable greenhouse control systems that are scalable, robust, and suitable for modern agricultural needs.

Acknowledgements:

First and foremost, I would like to thank Professor Michael Heaney for his constant support and guidance throughout the project. It was only possible to write this thesis with his motivation and advice. I would also like to thank Sir Mehmemt Karamanoglu and Nick Weldin for their invaluable support and guidance during the project. I am grateful to my family, friends, and colleagues for their support, encouragement, and understanding.

Table of Contents

List of Figures

1. Introduction:

The advancements in IoT devices have ushered in a new wave of technology in various sectors, including agriculture, which has revolutionized traditional farming methods. Amid the climate change crisis and the increasing need to feed the world's ever-growing population, new and sustainable agriculture methods are needed. Greenhouse farming is a viable solution to increase food yield while minimizing environmental impacts. However, this approach requires an innovative, reliable, efficient control system. This paper dives into the implications of IoT-driven agriculture systems. The project, therefore, is a Greenhouse monitoring system built around an ESP32 microcontroller, integrated with a GUI running on the Home Assistant Operating System on Raspberry Pi.

Substantial work is needed in the agricultural sector of the United Kingdom. The cold temperature in the UK means no loT crops can be grown outside, especially plants in arid or semi-arid areas of the world. Since it has restricted access to imports from other European countries, there is also an increasing need for agricultural practices after Brexit. There has also been a decrease in the availability of labor after Brexit because people are more likely to choose to go to countries that are still part of the European countries since there is a more significant influx of people coming from different regions [1].

Therefore, there is a need for agriculture to be “smart” in the UK since smart agriculture and Greenhouse farming can grow a wide variety of plants regardless of what the requirements of the plants are. There is also an increase in food production efficiency because less labor is needed for crops and plants.

2. Objectives:

This paper investigates the aspects of an IoT-based Greenhouse Control and Monitoring system. The project's focus is to offer a comprehensive guide and evaluation of what software and hardware are needed to create a Greenhouse project and what the project flow should be.

Evaluation: To thoroughly analyze the existing greenhouse monitoring system and incorporate it into the project.

Optimization: Introduce optimization in the project and methodize a systematic way to build a project.

User Experience: Introduce a way to make the project easily accessible to the general masses by increasing ways of interacting with the project.

Scalability: To assess the potential for scaling the existing system to accommodate more functionality and to scale the project into grander aspirations.

Sustainability: To evaluate the system's environmental impact and propose improvements to make it more energy-efficient and sustainable in the long term.

Documentation: To create documentation of the findings and to serve as a foundation for future research and implementation in IoT-based greenhouse farming systems.

By achieving these objectives, this paper aims to offer a subjective and well-rounded perspective on the subject matter and lay the groundwork for future endeavours in IoT-based agriculture.

3. Literature Review

The Internet of Things refers to how machines and computers talk, forming a mesh of different systems connected, over a specific network like Wifi or Zigbee. All these systems and devices have unique identifiers, allowing great modularity and adaptability. These machines or systems can be swapped out and changed, and the whole system can be adapted for an application, thus increasing operation ability and efficiency [2].

These IoT devices, sensors, and communication networks can offer diverse problem-solving solutions. For example, controlling and analyzing buildings' energy consumption can help reduce energy costs [3]. IoT can also be used in manufacturing jobs[4]. IoT is also widely applied in industrial applications that require communication between machines and plants, health and medical care, environmental control and monitoring systems, drone-based systems, and drone-based services[5][6][7].

The dynamic shifts in climate, exponential population growth, and diminishing availability of cultivable areas necessitate the adoption of novel methodologies to guarantee agriculture's sustainability and food provision in the following years. Greenhouse agriculture is widely regarded as a feasible and sustainable approach to address the impending food problem. By exerting control over the local environment, this method enables year-round cultivation of crops, even in adverse outdoor conditions. Nevertheless, greenhouse farms provide numerous operational efficiency and administration obstacles. The Internet of Things (IoT) technologies, including smart sensors, devices, network topologies, big data analytics, and intelligent decision-making, are a potential solution for addressing the primary challenges encountered in greenhouse farming. These challenges encompass Greenhouse climate control, crop growth monitoring, crop harvesting with machines, and other related aspects. This paper overviews greenhouse cultivation technologies and systems and examines the latest Internet of Things (IoT) technologies advancements for intelligent greenhouse and vertical farms. The study additionally emphasizes the primary obstacles that necessitate resolution [8].

The crops' yield dramatically depends upon parameter controls, such as temperature control, lighting control, water flow and irrigation, ventilation, level of CO_2 and other atmospheric gases, etc., inside the farm; therefore, controlling all these factors using IoT can significantly benefit farmers in increasing crop yields.

1.1 Smart Farming:

According to some surveys, by 2050 the world needs a 60% increase in food production due to population growth [9]. The advantages of using IoT are increasing crop yield while reducing the cost of resources used; for example, a study done by OnFarm resulted in findings that for an average farm the use of IoT increases profit by a considerable 1.75% amount while dropping energy cost by 32 dollars per hectare, while also decreasing the usage of water for irrigation by 8% [10]

Greenhouse farming also helps plants resist plant diseases. Open-field agriculture is susceptible to plant disease and pests. Whole crop fields can be affected by diseases and problems. To increase yield and protection against diseases and issues, closed-door farming, like Greenhouse farming, is very viable and has been proven in many cases. [11].

Recent technological advancements in the IoT sector have made it easy to adapt and incorporate IoT devices in farming [12].

IoT assisted advancements in machines have increased farm equipment's scale of usage, speed of work done, and productivity, allowing for more efficient and profitable cultivation. Irrigation and usage of fertilizers have also greatly improved, assisting farmers in increasing profit and crop yields.

Advancements in Smart Farming have increased the speed and productivity of farmers, allowing for more efficient cultivation [13]

Sensing technologies can also aid in increasing crop yield by identifying soil, crop status, and environmental conditions, allowing decreased need to use other resources like herbicides and pesticides [14]

1.2 Agriculture in the United Kingdom:

The agricultural terrain in the United Kingdom poses distinctive difficulties and prospects, predominantly shaped by its diverse climate patterns and recent political developments, notably the occurrence of Brexit.

Based on research from the United Kingdom government, Internet of Things (IoT) technology in the agricultural sector is still in its early phases; however, it displays encouraging patterns. The paper highlights that the utilization of Internet of Things (IoT) technology has the potential to enhance resource utilization efficiency, a matter of considerable importance in the United Kingdom, given its diverse climatic conditions.

There needs to be more going on in terms of Smart agriculture in the United Kingdom. In 2022, the government launched two new competitions with a funding of £20.5 million to develop innovative agriculture practices in the United Kingdom [15]. The government is also working towards creating Net-Zero Carbon farming practices and e ending farming-related emissions by 2050 [16].

Following the United Kingdom's exit from the European Union, there has been a noticable trend towards the adoption of autonomous systems as a means to address the workforce shortages experienced within the agricultural industry. According to a survey conducted by the Agriculture and Horticulture Development Board (AHDB), there has been a notable rise of 20% in the deployment of the Internet of Things (IoT) and autonomous systems after the occurrence of Brexit [17]

In addition, the UK's restricted growing seasons render greenhouse farming a rational option for achieving continuous food production throughout the year. The efficacy of greenhouse systems integrated with Internet of Things (IoT) sensors in adapting to the climatic conditions of the United Kingdom and enhancing crop productivity by as much as 30% was underscored in a scholarly article published in the Journal of Agricultural Science.

In summary, the adoption of innovative farming practices in the United Kingdom is experiencing an upward trend driven by technological developments and political shifts. Nevertheless, there is significant potential for expansion and enhancement, namely through implementing the Internet of Things (IoT) and greenhouse systems, to achieve more efficiency and sustainability in agriculture.

2 Project Approach:

Developing an intelligent agricultural project from scratch requires a multifaceted strategy, including hardware and software elements. In the early stages of the project, we concentrated on assembling prerequisites and defining goals, such as how the plants would be watered, how we would check the level of water delivered, temperature control to make sure plants have the optimal temperature to grow, what the light source will be, etc. Following this should be a thorough analysis of the parts required to build the said project and what software will be available to interface this hardware. We then moved with the system design after establishing the fundamental components. Here, hardware components like sensors, microcontrollers (like the ESP32), and actuators were chosen, and a reliable software architecture was created. Data security requires extra consideration, including secure transmission storage and encryption. The next step was prototyping, which entailed creating and testing each module to ensure it complies with the necessary functional specifications. Following extensive testing to confirm functioning and security features, integration—where hardware and software pieces are merged into a coherent system—was the next critical phase. A pilot system deployment with ongoing data logging and performance monitoring was only possible after the method had been validated.

3 Project Components:

3.1 ESP32

For this project, an Expressif ESP32 was chosen as the sensor hub. ESP32 offers a robust set of features that helps it be the project's main microcontroller. The most significant advantage of ESP32 over other microcontrollers is that it has built-in Wi-Fi that can be seamlessly incorporated into various projects. This Wi-Fi enables ESP32 to connect to a plethora of devices both on local networks and on other networks. The ESP32 also features a robust set of I/O features that help connect sensors; these I/O ports can handle Analogue to Digital conversions and allow ESP32 to communicate via SPI / SDIO or I2C / UART protocols. These communication abilities of ESP32 enables it to function as a standalone system or as a slave device to a host Microcontroller Unit, thereby reducing the communication stack overhead of the central application processor. [18].

The ESP32 is connected to 4 Capacitive Soil Sensors, a BME280 sensor that provides Temperature, Humidity, and Pressure readings, a fan, a heater, Plant Light, and a water pump. This combination of sensors and actuators mimics the systems used in a greenhouse system.

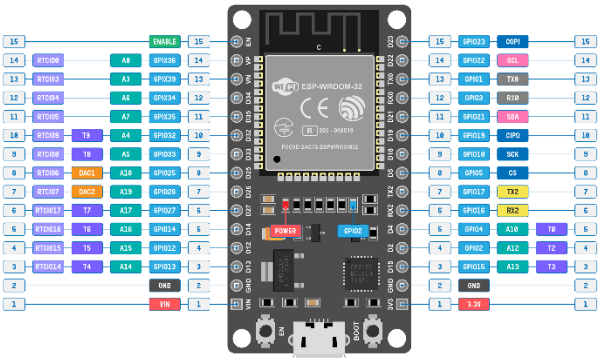

Figure 1: ESP32 Development Board [19]

3.2 Raspberry Pi

The Raspberry Pi is widely recognized for its compact dimensions and cost-effectiveness, making it a remarkably adaptable microcomputer. Home automation is frequently utilized to control many aspects of a household, encompassing lighting, thermostats, and appliances. Moreover, it is advantageous to establish a cost-efficient media center by using software such as Kodi or Plex. Individuals who engage in gaming frequently use it as a retro gaming console, employing emulation techniques to play old video games from platforms such as the Nintendo Entertainment System (NES). Furthermore, it is essential to acknowledge this platform's educational significance as it is a robust tool for instructing coding languages such as Python and Scratch. In this instance, the content is condensed into a single paragraph, expressed with a solid emotional emphasis. [20].

In this project, we use Raspberry Pi to host an operating system called Home Assistant OS. Home Assistant OS is a GUI to control and display our program's functionality. The Raspberry Pi is also the system's central hub, receives updates from ESP32 depending on the readings, and displays them in a Human readable format.

Figure 2: Raspberry Pi 4 [21]

3.3 Home Assistant OS:

Home Assistant OS is a dedicated operating system specifically developed to facilitate the execution of the Home Assistant software, which serves as the central hub for managing various smart home functionalities. The operating system in question is specifically designed to be optimized for the Raspberry Pi and other platforms. It provides a streamlined installation and operation process, eliminating the need for guesswork when setting up a home automation system. The system's architecture is precisely engineered to prioritize dependability, ensuring the seamless operation of your smart home setup without any disruptions. Moreover, it effortlessly incorporates a wide range of intelligent home devices, enabling users to conveniently manage various aspects of their household, including lighting and temperature settings. The primary objective of these efforts is to provide an exceptional user experience. Therefore, Home Assistant OS serves as an organizing tool for the whole project, displaying and controlling the entire project through visualization.

One of the biggest reasons for choosing Home Assistant over Node-Red was the ease of use and a much faster-performing GUI. Linktap Api also natively supports Home Automation, meaning setting up LinkTap on Home Assistant was quicker.

Some other benefits that Home Assistant has over Node-Red are.

1. **Home Ecosystem Integration**: Home Assistant has native support for many smart home protocols and devices. Home Assistant often provides more out-of-the-box compatibility if you're trying to create a comprehensive smart home ecosystem.
2. **State Management**: Home Assistant excels at maintaining the state of various devices and entities, allowing for complex automation based on state changes or events. While Node-RED can also manage states, Home Assistant offers a more comprehensive approach, making it easier to build conditions based on the history and state of multiple devices.

3. **Community Support**: While both platforms have strong communities, Home Assistant's more extensive user base often means more frequent updates, more available integrations, and better community support for troubleshooting specific smart home issues.
4. **Security Features**: Home Assistant has been designed with privacy and local control in mind. All your data can be kept locally, and advanced security features like two-factor authentication are built into the system.
5. **YAML Configuration**: For those who prefer to write configurations in text form, as opposed to the flow-based graphical approach of Node-RED, Home Assistant's YAML-based configuration might be more appealing.
6. **Built-in Automation**: Home Assistant has its own automation and scripting engine, allowing you to define complex logic without needing another service. Although Node-RED arguably offers a more visual and intuitive way to create automation, having built-in options can be convenient.
7. **Native Mobile App**: Home Assistant offers a native mobile app with geofencing capabilities, allowing for complex automation based on location without needing an additional service.

Home Assistant is the control hub of the project and is used to display graphs for different variables we use in our project. It also shows the buttons needed to control the system manually. Home Assistant uses MQTT to send and receive data from ESP32.

Figure 3: The main GUI for displaying and controlling variables from ESP32.

3.4 MQTT:

A lightweight publish-subscribe network protocol, MQTT (Message Queuing Telemetry Transport) enables devices to communicate with one another promptly and effectively. Emphasize that MQTT shines in low-bandwidth scenarios, such as intelligent agricultural installations with numerous sensors, actuators and other devices in the field. To relay sensor data (such as temperature, humidity, and soil moisture) and receive CSV-formatted setup parameters from a centralized server, our system relies on MQTT as the communication layer for ESP32. We used Eclipse Mosquitto which is an open source MQTT broker for our project.

The code is organized in a way that makes it possible to monitor and control many environmental variables in real time using data from various sensors and actuators. Modules within the program collect data from sensors, activate actuators depending on that data, and maintain constant communication over the MQTT protocol. Critical data, such as sensor thresholds, actuator statuses, and other operational characteristics, are stored in several global variables and arrays. The MQTT part of the code includes solid capabilities for managing connection, authentication, and data publication to a broker. Two primary features are responsible for dynamically controlling heaters, fans, and water pumps depending on pre-defined criteria; these features are named for handling temperature and moisture situations. These limits can be adjusted as needed by editing a CSV file that will be read by the software at runtime. This works together to form a robust yet adaptable, smart agriculture system that can regulate and monitor its surrounding environment in real time.

Alternative Communication Methods Between Raspberry Pi and ESP32:

1. **Serial Communication (UART)**:
    - Pros: Simple, low-latency, doesn't require additional hardware.
    - Cons: Limited to point-to-point and relatively short distances.
2. **SPI (Serial Peripheral Interface)**:
    - Pros: High-speed, full-duplex communication.
    - Cons: Requires multiple pins, short-distance communication.
3. **I2C (Inter-Integrated Circuit)**:
    - Pros: Requires only two wires to connect multiple devices.
    - Cons: Lower speed compared to SPI, complex for multiple devices.
4. **Bluetooth**:
    - Pros: Wireless, straightforward for simple data transfer.
    - Cons: Limited range, lower data rates compared to WiFi.
5. **WiFi Direct / HTTP / WebSockets**:

- Pros: Wireless, well-suited for larger data or bi-directional communication.
- Cons: More complex to set up, higher power consumption.

6. **Zigbee / LoRa**:
    - Pros: Long-range, low power consumption.
    - Cons: Requires additional transceiver modules, lower data rate.

Why MQTT Is Superior:

1. **Low Bandwidth**: MQTT messages can be much smaller than messages in other protocols, which is ideal for IoT networks, where bandwidth might be at a premium.
2. **Quality of Service Levels**: MQTT supports different levels of protocols for message delivery:
    - QoS 0: The message is delivered once, and delivery is not confirmed.
    - QoS 1: The message is delivered at least once, and delivery is confirmed.
    - QoS 2: A four-step handshake delivers. The message exactly once.
3. **last will**: A feature that allows a message to be sent if the client disconnects unexpectedly. This is useful for sending "offline" or "fault" states.
4. **Persistent Sessions**: Even if a client disconnects, the session can be stored and quickly resumed when the client reconnects.
5. **Ease of Scaling**: MQTT brokers can handle many concurrently connected devices, which is crucial for scalability in IoT.
6. **Real-Time Capability**: MQTT is well-suited for real-time monitoring and automation tasks due to its low latency and various QoS levels.
7. **Security**: SSL/TLS can encrypt the entire communication process for the publisher and subscriber.
8. **Broker-Based**: This architecture makes it easier to decouple the producer and consumer of the data, allowing you to change one without affecting the other.

Because of these features, MQTT is often considered superior for IoT applications, especially those that require real-time operation, low latency, and the ability to handle many clients, like your smart agriculture system. [22]

3.5 Capacitive Soil Sensor:

Resistive soil moisture sensors are susceptible to corrosion with a limited lifespan regardless of measures taken. These sensors feature exposed plating that corrodes over time, leading to errors in readings. Capacitive Soil Moisture sensors, however, use capacitive sensing to detect soil moisture resulting in much more accurate and robust sensor without corrosion. The sensor has two electrodes acting as a capacitor, and the surrounding soil is the charge-storing medium. The more water the soil has, the higher the readings of the Soil Moisture sensor will be [23]

The Soil Moisture sensor has a built-in voltage regulator that supports 3.3V and works with 3V controllers and 5V Controllers. The sensor produces readings in Analogue, which are then converted to the percentage of water in a part of the soil.

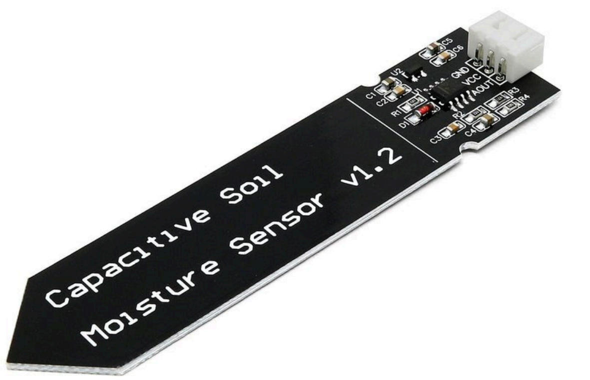

Figure 4: Soil Moisture Sensor [24]

3.6 BME280:

The BME280 is a miniature environmental sensor with temperature, barometric pressure, and humidity measuring capabilities. It’s an all-in-one sensor, meaning that all packages are in a small, single PCB board, making it great for hobby projects yet offering great accuracy. The sensor can be connected to ESP32 via SPI or I2C. It measures humidity with ±3% accuracy, barometric pressure with ±1 hPa absolute accuracy, and temperature with ±1.0°C accuracy, which is accurate enough for Green-House purposes [25]. Different plants require different temperatures to thrive; desert plants need a higher temperature than plants that live in cooler temperatures. Plants also use temperature to sense changes in seasons, so keeping plants in a certain range of temperatures is important. This is why BME280 is needed to check the Greenhouse temperatures and for the monitoring system to take action depending on the temperature. Humidity is also useful for Plants because high humidity stops plants from losing moisture through the leaves and drying out.

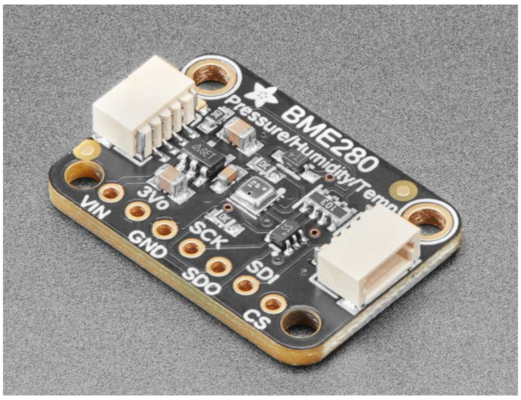

Figure 5: BME280 Sensor [26]

3.7 Fan and Heater:

To promote optimal plant growth and increase agricultural productivity, a Greenhouse needs optimal temperature. One of a greenhouse's main objectives is to offer a regulated environment that supports plant growth independent of the weather outside. Maintaining an ideal temperature range within the greenhouse is crucial for promoting healthy plant growth and increasing yield during cooler months and in cooler climates or hotter months and in hotter climates.

The fan serves the purpose of providing ventilation to the plants and regulating the temperature of the Greenhouse in cases of high temperature. Traditionally, Greenhouses have windows that allow exchanges of air to refresh oxygen and carbon dioxide levels and allow pollinating insects and animals to interact with plants and pollinate them. But for this project, the functionality of opening windows is switched with the working of a fan.

For a heating source we were going to use a Peltier device for providing heat during winters.

In order to promote plant growth and increase agricultural productivity, a greenhouse may need a heat source. One of a greenhouse's main objectives is to offer a regulated environment that supports plant growth independent of the weather outside. Maintaining an ideal temperature range within the greenhouse is crucial for promoting healthy plant growth and increasing yield during cooler months or in cooler climates. A source of heat can aid in sustaining this preferred temperature range, supporting temperature-sensitive biological processes like photosynthesis, germination, and flowering. Furthermore, a heat source can help protect delicate plants from frost damage, which is especially important when producing warm-weather crops in cold or temperate climates.

However, due to budget and time constraints we were unable to secure a Peltier device in time, but the functionality can be observed using LEDs in the project.

3.8 Water Pump:

Plants need water to survive as it provides sustenance and aids plants in Photosynthesis. Different plants require different amounts of water to thrive. Some plants, like almond and fig plants, require very little water and are drought-resistant [27] plants require high volumes of water to grow, like the watermelon plant [28]. I have used a 5v submersible water pump for this project to supply water to plants. This water pump can easily be scaled depending on the volume of water a greenhouse requires.

3.9 Light:

Light is necessary for plants as it is one of the building blocks of Photosynthesis. Plants require an appropriate amount of light for life, excess of which leads to plant damage. In extreme cases, high exposure to light can also lead to Plant death, a phenomenon easily recognized by people who put houseplants outside in direct sunlight [29]. So, plants must have a stable supply of Light, especially without sunlight. A constant supply of light is generated from LED lights. In a traditional greenhouse, the LED Lights are mounted on the roof to provide light to the plants; however, we chose a readily available plant lamp for this project. The Plant lamp includes 3000K warm light, 5000K white light, and 660nm red light LEDs, which are a full spectrum close to natural sunlight and have time cycles of 3, 9, and 12 hours. Plants also use light as an indication of what season it is. Plants recognize how long light is falling on the leaves to determine what season it is and bloom and wither accordingly. This process is called photoperiod [30]. So, plants must have a stable supply of either artificial or real.

4 Code Analysis:

4.1 Libraries:

The core part of the code is the use of libraries available for each functionality.

1. WiFi.h

- **Purpose**: Manages the WiFi functionalities of the ESP32.
- **Key Functions**:
    - **WiFi.begin()**: Connects to a WiFi network.
    - **WiFi.status()**: Checks the current status of the WiFi connection.
- **Importance**: Essential for establishing and maintaining the WiFi connection, which is crucial for MQTT communication.

2. PubSubClient.h

- **Purpose**: Implements the MQTT protocol.
- **Key Functions**:
    - **client.setServer()**: Sets the MQTT server to connect to.
    - **client.setCallback()**: Registers a callback function to handle incoming MQTT messages.
    - **client.publish()**: Publishes messages to MQTT topics.
    - **client.subscribe()**: Subscribes to MQTT topics.
- **Importance**: Enables real-time data publishing to and from the MQTT broker, facilitating remote monitoring and control.

3. Wire.h

- **Purpose**: Enables I2C communication between the ESP32 and other devices.
- **Key Functions**:
    - **Wire.begin()**: Initializes the I2C bus.
    - **Wire.requestFrom()**, **Wire.write()**, **Wire.read()**: Facilitate read and write operations.
- **Importance**: Vital for interfacing with sensors like the BME280, which relies on I2C communication.

4. Adafruit_Sensor.h and Adafruit_BME280.h

- **Purpose**: Specific libraries for interfacing with the BME280 sensor.

- **Key Functions**:
  - **bme.begin()**: Initializes the BME280 sensor.
  - **bme.readTemperature()**, **bme.readPressure()**, **bme.readHumidity()**: Retrieve sensor readings.
- **Importance**: Simplifies the data collection process from the BME280 sensor, providing ready-to-use methods for obtaining environmental data.

5. SPI.h

- **Purpose**: Provides SPI communication functionalities.
- **Key Functions**:
  - **SPI.begin()**: Initializes SPI communication.
  - **SPI.transfer()**: Transmits data over SPI.
- **Importance**: Useful for interfacing with devices that require SPI communication, though not explicitly utilized in the current codebase.

4.2 Initialization and Global Variables:

The program starts with the declaration of global variables and constants:

- **Network Parameters**: These include WiFi credentials (**ssid**, **password**) and the MQTT server address (**mqtt_server**), assisting in the secure establishment of network connections and authentication.
- **MQTT Topics**: Descriptive MQTT topics (**temperature_topic**, **humidity_topic**, etc.) are declared for different environmental factors to maintain structured data flow and for Home Assistant to understand what topic includes data.
- **Hardware Pins**: The hardware pins are declared as constants to explicitly define the PINs for the soil moisture sensor and BME280, establishing a clear hardware-software link.
- **Time Variables**: Timekeeping variables (**lastMsg**, **lastRecu**) are introduced to control the frequency of data publishing and ensure asynchronous operation.

4.3 Initialization Phase: The **setup()** Function

1. **WiFi Initialization**:

- The **WiFi.begin(ssid, password)** method attempts to establish a secure WiFi connection.
- A **while** loop ensures that the program remains in this stage until the connection is successful, indicated by **WiFi.status()** returning **WL_CONNECTED**. This means that the program continues only when the internet is connected.

2. **MQTT Configuration**:

- The MQTT client is instantiated with the server address specified in **mqtt_server**.
- The **client.setCallback(callback)** function links the MQTT client to a callback function, facilitating the handling of incoming MQTT messages and deciphering them.

3. **BME280 Sensor Setup**:

- The **Wire.begin()** initializes the I2C bus, allowing communication with the BME280 sensor instead of SPI.
- Sensor initialization and validation checks occur, confirming the presence of the BME280 sensor.

4.4 Main Program Loop: The **loop()** Function

1. **Connection Checks**:

- The code periodically verifies WiFi and MQTT client connections; if they are not connected, this part of the code ensures they reconnect.

2. **Event Maintenance**:

- The **client. Loop ()** function ensures that the MQTT client remains operational, listening for incoming messages and maintaining keep-alive messages with the broker.

3. **Sensor Data Collection and Transmission**:

- Every **publish_interval** milliseconds, the system takes sensor reading functions for the BME280 and the soil moisture sensor.
- These readings are then published to the MQTT broker under their respective topics.

4. **Remote Control Mechanism**:

- The **callback** function processes incoming MQTT messages to control actuators based on the received payload and topic.
- It reads data from the BME280 sensor and soil moisture sensors.

- Calls two functions, **heatloop()** and **moistureloop(),** to decide the control logic for heating, cooling, and irrigation.
- Constructs a JSON object containing all the sensor and actuator statuses.
- Publishes the JSON object to an MQTT topic if the client is connected.

MQTT Callback Function: **callback()**

- The function is invoked upon receiving an MQTT message.
- Parses the message topic and payload to identify which actuator should be controlled.
- Conditional control logic toggles the state of actuators like pumps, heaters, or fans based on the received MQTT payload.

Heat Loop

- The **heatloop()** function decides whether to turn the heater or the fan on/off. It uses the temperature (**temp**) and compares it to pre-defined temperature ranges (**floatVariablesArray[0]** and **floatVariablesArray[1]**) to make the decision.

Moisture Loop

- The **moisture-loop ()** function takes the average of four soil moisture readings and decides whether to turn the water pump on or off based on pre-defined moisture levels (**floatVariablesArray[4]** and **floatVariablesArray[5]**).

MQTT Loop

- A section of the main **loop()** ensures that if the MQTT client is disconnected, it will attempt to reconnect. Once connected, it serializes the JSON object and publishes it to an MQTT topic.
- Count Lines and Parse CSV
- Two utility functions, **countLines()** and **parseCSV()**, are employed to read and parse a CSV file. The CSV file is assumed to contain control variables. These parsed variables are then stored in global arrays for dynamic control of the system.

4.5 Code Flow:

1. **Start** the Beginning of the program flow.
2. **Initialization**:
   - Initialize variables.
   - Set up pins.
   - Connect to Wi-Fi and MQTT broker.
3. **Fetch CSV Data**:
   - Read variables from the CSV file.
   - Parse and populate into **floatVariablesArray.**
4. **Read Sensors**:
   - Read temperature (**tempF**).
   - Read moisture levels (**moist1**, **moist2**, **moist3**, **moist4**).
5. **Manual Override:**
   - If the Manual Override is Not Active, go to Step 6.
   - If Active Control Heater/Fan/Pump from GUI.
6. **Heat Loop Decision from CSV**:
   - If temperature <= set lower limit, turn on the heater.
   - If temperature >= set upper limit, turn on the fan.
   - Otherwise, turn off both.
7. **Moisture Loop Decision from CSV**:
   - If average moisture <= set lower limit, turn on the pump.
   - If average moisture >= set upper limit, turn off the pump.
8. **Generate MQTT Payload**:
   - Create a JSON payload with the current statuses of the heater, fan, and pump.
9. **MQTT Publish**:
   - Publish the JSON payload to the MQTT broker.
10. **Delay**: Wait for a certain time before the next loop.
11. **End Loop**: Go back to "Read Sensors."
12. **End**: Indicates the end of the program flow (though in an Arduino loop, it essentially goes on forever).

5 `Problems and Solutions:

5.1 LinkTap:

The LinkTap G1S Wireless Water Timer is designed for controlling water flow in irrigation systems. Irrigation systems can benefit greatly from using the LinkTap G1S Wireless Water Timer. It is typically used in automated farming systems that can be managed from afar with a smartphone or other Internet-connected device. There are several functions available on this sophisticated water timer that go beyond basic timing systems. It is tremendously useful for farmers and gardeners because users can remotely control water schedules through a dedicated mobile application. The G1S model enables automated watering schedules and includes a rain delay option to prevent overwatering in the event of precipitation. The user is kept informed about water consumption in real time, which helps with resource management. The gadget seamlessly connects with voice-activated systems like Google Assistant and Amazon Alexa, making operations more hands-free. Furthermore, its cloud support raises the prospect of sophisticated data logging and analytics. The efficiency and effectiveness of irrigation systems can be greatly increased using the LinkTap G1S's capabilities.

Based on the general principles outlined in the document on the manufacturer’s website, we can use MQTT to integrate the LinkTap Gateway with Home Assistant. Here's a step-by-step guide on how to achieve this:

Prerequisites:

1. A working Home Assistant setup.
2. A LinkTap Gateway and its associated watering devices.
3. Access to the MQTT broker that Home Assistant uses.

Steps for MQTT Integration:

LinkTap Gateway Configuration

1. **Access LinkTap Gateway Admin Panel**: Log in to your LinkTap Gateway admin panel.
2. **Configure MQTT Settings**: Go to the MQTT settings page and enter the following details:
    - **Broker Address**: The IP address of your MQTT broker (likely the same machine where Home Assistant runs).
    - **Broker Port**: Usually 1883 for non-SSL, 8883 for SSL.
    - **Client ID**: A unique ID for LinkTap.
    - **Username and Password**: If your MQTT broker is secured.

- **Keep Alive Interval**: Setting it in the range of 30~120 seconds is recommended.

3. **Configure MQTT Topics**: Define the topics for uplink and downlink.
    - **Uplink Topic**: Topic for LinkTap to publish status info.
    - **Downlink Topic**: Topic where LinkTap will receive commands.
4. **Save and Reboot**: Save these settings and reboot the LinkTap Gateway.

Home Assistant Configuration

1. **Install MQTT Integration**: If not already installed, install the MQTT integration in Home Assistant.
2. **Subscribe to Uplink Topic**: Create an automation or sensor in Home Assistant to subscribe to the "Uplink Topic" you defined in LinkTap Gateway.
3. **Publish to Downlink Topic**: To send commands to LinkTap, publish messages to the "Downlink Topic" through Home Assistant's services.

After being set up, Linktap can be controlled via Home Assistant. The different variables that can be controlled through Home Assistant and their descriptions are as follows.

Home Assistant Component	Entity name	Note
binary_sensor	X_is_flm_plugin	Flow meter's connection status. "on" means the flow meter is connected.
	X_is_rf_linked	Water timer's connection status. "on" means the water timer is connected to the Gateway.
	X_is_watering	Is the water timer currently watering?
	X_is_manual_mode	Is watering by pressing the "Manual On/Off" button (on Taplinker G1S/G2S)?
	X_child_lock	Is the child-lock of the "Manual On/Off" button (on Taplinker G1S/G2S) enabled?
sensor	X_total_duration	Total watering duration in seconds.
	X_remain_duration	The remaining watering duration is in seconds.
	X_speed	Current flow rate (LPM).
	X_volume	Accumulated volume (Litre).
	X_volume_limit	The target volume for the current watering cycle. Once the "volume" reaches the "volume_limit," the Taplinker will stop watering. When it is 0, the watering process retains until "remain_duration." runs out.
	X_signal	Percentage of signal strength of water timer
	X_battery	Percentage of remaining battery power of water timer.
	X_ failsafe_duration	Configurable failsafe duration in seconds. When the "water_switch" is turned on, the failsafe

		Duration will be used as "total_duration." When it is 0, "total_duration" uses the default value on the gateway.
switch	X_water_switch	Watering switch.
alarm_control_panel	X_fall_alarm	Alarm status: --- Disarmed. The “TapLinker fall alert” (In the LinkTap App) is disabled.

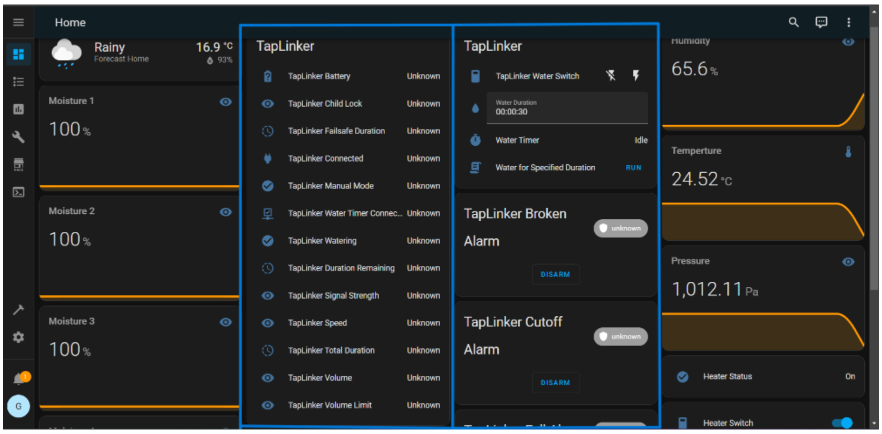

Figure 6: Home Assistant connected to LinkTap

However, during our testing, we found that controlling LinkTap through Home Assistant is problematic as it needs to be connected to the MQTT Broker and hangs all other systems connected to it, meaning that if LinkTap is connected to an MQTT broker, no other devices can connect to the same broker.

Figure 7: LinkTap Model G1S [31]

Figure 8: LinkTap Gateway [31]

Solution: We had to abandon Linktap as a controller in our system. The LinkTap's functionality can be easily created using a pump and a flow meter connected to a microcontroller.

5.2 Node-Red:

Node-RED is a flow-based development tool that combines devices, online services and various APIs. It's a browser-based editor making it easy to wire together flows using different nodes. Node-RED is well-suited for Internet of Things (IoT) applications.

Node-RED served as an intermediary between the ESP32 and the MQTT broker. It could perform additional data processing transformation and even execute conditional logic based on the incoming sensor data. For instance, using MQTT to publish data from the ESP32, Node-RED subscribed to MQTT topics and visualized the data using its dashboard nodes. We created gauge charts for temperature, moisture, and other variables.

Furthermore, Node-RED can integrate easily with databases like MySQL, allowing for more convenient and robust data storage solutions than the ESP32.

However, Node-Red running on Raspberry Pi was very slow and did not inherently support many features we wanted to add to the project, especially LinkTap. Due to the slow processing power of Node-Red, we had to abandon it and move towards Home Assistant, which natively supported LinkTap and was much faster in terms of GUI performance.

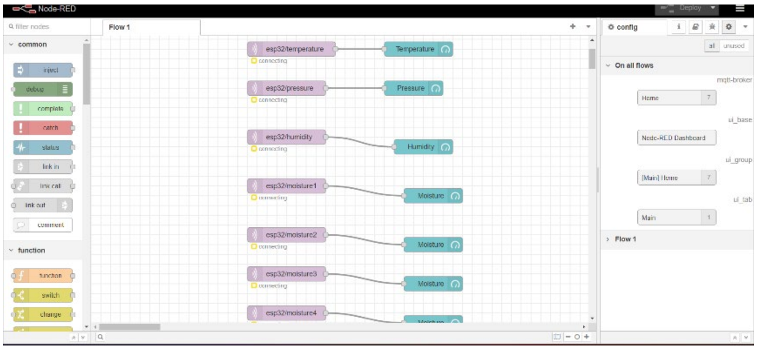

Figure 9: Node-Red Dashboard

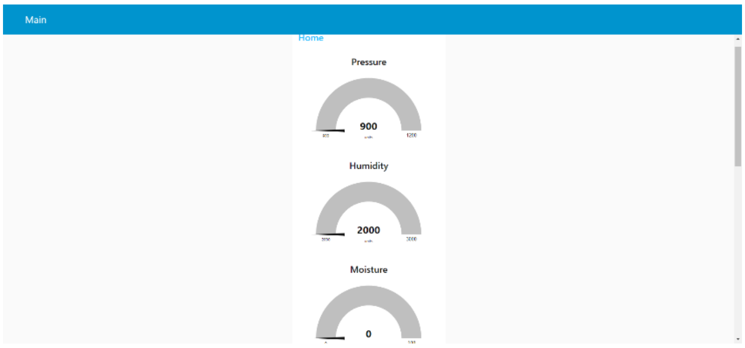

Figure 10: Node-Red GUI

Solution: We switched to Home Assistant as it offers a faster functioning GUI and extra features catered towards Automation.

5.3 ESP32

ESP32 has two channels of ADC pins; however, when connected to WiFi, the ADC2 set of pins which includes pins GPIO 0, GPIO 2, GPIO 15, GPIO 13, GPIO 12, GPIO 14, GPIO 27, GPIO 25, GPIO 26 cannot be used. This limits the functionality of ESP32 in projects that require numerous sensors.

Solution: Multiplexer chips like TCA9548A I2C can increase the number of sensors connected to ESP32. Also, some other microcontrollers can be used.

6 Testing and Results:

The aim of the project was to test out a hardware and software of the smart monitoring system and control system for Greenhouse application. Unlike typical hardware and software systems that can be tested and adjusted in short periods of time, the efficacy of the proposed farming solution can only be validated over an entire growing season. To address this, we conducted preliminary tests on individual components, focusing on their accuracy and reliability. The system was initially deployed with just the soil sensors to test the soil moisture, and other sensors were added step by step as the system matured. Current iterations and deployments of the system are successful, indicating that the system can have long term testing periods as well where the system remains active for the whole growing season.

7 Future Advancements

7.1 Scalability:

Scalability is an important part of any project that must be considered in developing IoT-based smart farming initiatives to achieve long-term effectiveness. The agricultural industry in the United Kingdom exhibits considerable potential for implementing technological innovations, which can enhance crop productivity, maximize resource utilization, and foster environmental sustainability. A project that can readily broaden its scope, accommodate varying scales of agricultural practices, and incorporate emerging technologies is said to be scalable. In addition, scalability guarantees the project's ability to accommodate different crops, weather conditions, and soil types without necessitating a comprehensive revamp.

1. **Multi-Node System**: The project right now is built upon a single ESP32 board. It can be scaled by implementing a multi-node system where multiple ESP32 devices work in unison. MQTT topic structures need to be updated to differentiate between different nodes.
2. **Distributed Computing**: Using edge computing solutions like AWS Greengrass or Azure IoT Edge to distribute computation and data storage.
3. **Containerization**: To manage services like MQTT broker, databases, and possibly a backend server, containerization technologies like Docker can be used.
4. **Sensor Diversity**: More sensors like wind speed, UV index, and light intensity are introduced to collect more comprehensive environmental data. This can help in creating advanced models for farm management.

5. **Cloud Integration**: A cloud-based management system can store large volumes of data, run advanced analytics, and provide access to multiple users simultaneously, allowing for easier managing of multiple farms.
6. **Edge Computing**: Implementing edge computing can reduce the latency in decision-making processes for the actuators, making the system more responsive.
7. **AI and Machine Learning**: Advanced algorithms can analyze the data to predict yields, detect diseases early, or suggest specific interventions.
8. **Automated Machinery**: Incorporate IoT-enabled farm equipment like tractors and drones for planting, harvesting, and monitoring. These can be controlled via the same central system.
9. **Blockchain**: For traceability of produce, implementing blockchain can bring about transparency in the supply chain, from farm to consumer.
10. **Multiple Crops**: Initially designed for a specific crop type, the system can be adapted for multiple crops, thereby increasing its utility.
11. **Livestock Monitoring**: Integration of the system with livestock monitoring solutions to get an all-encompassing smart farm.
12. **Multi-Location Management**: Scale the system from one farm to multiple farms in different parts of the country.
13. **Data Sharing and Cooperative Models**: Enable data sharing amongst farmers to foster a cooperative model where everyone can benefit from collective data.

7.2 Feature Additions

A few feature additions are suggested to make the smart farming system more comprehensive and versatile. Machine learning algorithms in predictive analytics can increase yield quality and optimize resource consumption. A more complete image of farm conditions will be provided by supporting various sensors, such as pH levels and gas detectors. Finally, the system's value and sophistication will be increased by real-time alerts and remote-control capabilities, enabling farmers to act immediately.

1. **Machine Learning Models**: Implement machine learning models to predict future soil conditions, temperature, etc., and adjust your actuator states accordingly.
2. **Real-Time Dashboard**: A real-time dashboard can be implemented using technologies like Grafana to visualize data and allow manual overrides.
3. **User Notifications**: Implement an SMS, Email, or push notifications notification system to alert the user when sensor readings exceed certain limits.
4. **Energy Saving**: Implement an energy-saving mode that intelligently turns off/on sensors or actuators when unnecessary.

5. **Weather API Integration**: The system could make more informed decisions by integrating with a weather forecasting API. For example, if rain is predicted, the watering system could be turned off in advance.
6. **Camera Integration**: Integrate cameras to monitor the condition of crops. Image processing could help in disease identification.
7. **Voice Control**: Adding voice command features through platforms like Alexa or Google Assistant for easy control and monitoring.
8. **Battery Backup and Health Monitoring**: Ensure the system has a battery backup. Regularly check the backup status and other critical components, then warn if anything is out of order.
9. **Advanced Control Algorithms**: PID controllers could be used for more efficient resource usage instead of simple threshold-based control.
10. **OTA Updates**: Over-the-air updates for the firmware can help easily upgrade the system without manual intervention.
11. **Multi-Language Support**: If the system has a user interface, it can be more accessible.

7.3 Impact on the UK's Farming Scene:

The development of smart farming technologies has the potential to transform British agriculture. Scalable IoT systems, predictive analytics, and real-time monitoring can help farmers increase agricultural yields, use resources more effectively, and have a smaller environmental impact. In addition to increasing the UK's agricultural production, this will improve the sustainability and competitiveness of its farming methods globally.

1. **Precision Agriculture**: The introduction of advanced sensing and automation can increase yield and reduce waste, which is crucial for the UK's agricultural competitiveness.
2. **Sustainability**: Efficient use of resources can lead to more sustainable farming practices, which aligns with the UK's environmental goals.
3. **Data-Driven Decisions**: The UK's farmers can make more informed decisions based on real-time data and analytics, leading to higher productivity.
4. **Labor Efficiency**: Automation can significantly reduce the need for manual labor, mitigating one of the key challenges in the UK's agricultural sector.
5. **Supply Chain Optimization**: Transparency and traceability can greatly enhance the agricultural supply chain, making UK produce more appealing both domestically and for export.
6. **Regulatory Compliance**: With stricter environmental regulations, a smart agriculture system can help farmers comply more easily by precisely monitoring and controlling farm conditions.

8 Conclusion:

The growing need for ensuring food availability, food security and implementing sustainable agricultural practices has created a requirement for technological intervention in the farming scene, which is effectively addressed by Smart Farming Systems. The present study has conducted an extensive examination of the incorporation of Internet of Things (IoT) devices, such as the ESP32 microcontroller, sensors, and actuators, in order to create an effective system for agricultural monitoring and control that operates in real-time.

Noteworthy achievements include the effective implementation of sensors for soil moisture, temperature, and humidity, enabling the monitoring of environmental variables. The data collected by these sensors is transmitted to cloud-based systems, enabling the utilization of real-time analytics and adaptive decision-making algorithms. Furthermore, we discussed the possibility of integrating the Weather API to predict irrigation requirements by considering future weather conditions. Additionally, we explored the utilization of sophisticated control algorithms such as PID controllers to enhance resource efficiency.

The potential advantages are not only technical aspects but are also ecological and economic dimensions. The development and use of such systems is in accordance with the United Kingdom's aspirations to enhance its agricultural industry. The use of smart farming technology by British farmers has the potential to enhance crop yields, optimize resource utilization, and mitigate environmental impacts. This, in turn, can make a significant contribution to the sustainability of the nation and its worldwide competitiveness.

In the context of future research, several potential routes were identified.

- The utilization of machine learning algorithms in the context of predictive analytics.
- The utilization of cameras and image processing techniques for the purpose of monitoring crop health has gained significant attention in recent years.
- The incorporation of IoT devices in Crop machines for seamless integration with monitoring and control systems.

In conclusion, the integration of Internet of Things (IoT) in the context of smart agriculture not only addresses the pressing requirement for enhanced agricultural output, but also achieves this objective in an environmentally sustainable manner, aligning with the stricter environmental standards in the United Kingdom. Based on the present situation and prospective developments, it is evident that smart agriculture holds significant potential as a fundamental cornerstone in the advancement of agricultural methods in the United Kingdom.

References

[1] T. Lang and M. McKee, "Brexit poses serious threats to the availability and affordability of food in the United Kingdom," *J Public Health (Oxf),* vol. 40, *(4),* pp. e608-e610, 2018. Available: https://doi.org/10.1093/pubmed/fdy073. DOI: 10.1093/pubmed/fdy073.

[2] Anonymous "Energies | Free Full-Text | An Energy-Efficient and Secure Routing Protocol for Intrusion Avoidance in IoT-Based WSN," *Mdpi,* 2019. Available: https://www.mdpi.com/1996-1073/12/21/4174. DOI: https://doi.org/10.3390/en12214174.

[3] A. Zouinkhi *et al*, "Auto-management of energy in IoT networks," *Int J Commun Syst,* vol. 33, *(1),* pp. e4168, 2020. Available: https://doi.org/10.1002/dac.4168. DOI: 10.1002/dac.4168.

[4] V. Tsiatsis, "Hö ller J, Mulligan C (2014) From Machine-to-Machine to the Internet of Things: Introduction to a New Age of Intelligence," .

[5] D. Evans, "The internet of things," *How the Next Evolution of the Internet is Changing Everything, Whitepaper, Cisco Internet Business Solutions Group (IBSG),* vol. 1, pp. 1-12, 2011.

[6] N. H. Motlagh, M. Bagaa and T. Taleb, "Energy and delay aware task assignment mechanism for UAV-based IoT platform," *IEEE Internet of Things Journal,* vol. 6, *(4),* pp. 6523-6536, 2019.

[7] A. Ramamurthy and P. Jain, "The internet of things in the power sector opportunities in Asia and the Pacific," 2017.

[8] R. Rayhana, G. Xiao and Z. Liu, "Internet of Things Empowered Smart Greenhouse Farming," *IEEE Journal of Radio Frequency Identification,* vol. 4, *(3),* pp. 195-211, 2020. . DOI: 10.1109/JRFID.2020.2984391.

[9] M. Mykleby, P. Doherty and J. Makower, *The New Grand Strategy: Restoring America's Prosperity, Security, and Sustainability in the 21st Century.* 2016.

[10] P. Gralla, "Precision agriculture yields higher profits, lower risks," *Hewlett Packard Enterprise,* 2018.

[11] B. Richard, A. Qi and B. D. L. Fitt, "Control of crop diseases through Integrated Crop Management to deliver climate-smart farming systems for low- and high-input crop production," *Plant Pathol.,* vol. 71, *(1),* pp. 187-206, 2022. Available: https://doi.org/10.1111/ppa.13493. DOI: 10.1111/ppa.13493.

[12] J. E. Ibarra-Esquer *et al*, "Tracking the evolution of the internet of things concept across different application domains," *Sensors,* vol. 17, *(6),* pp. 1379, 2017.

[13] M. Javaid *et al*, "Enhancing smart farming through the applications of Agriculture 4.0 technologies," *International Journal of Intelligent Networks,* vol. 3, pp. 150-164, 2022. Available: https://www.sciencedirect.com/science/article/pii/S2666603022000173. DOI: 10.1016/j.ijin.2022.09.004.

[14] Y. Zhang, "The role of precision agriculture," *Resource Magazine,* vol. 26, *(6),* pp. 9, 2019.

[15] Anonymous (). *Farming Innovation Programme: 2 new competitions launched - Farming*. Available: https://defrafarming.blog.gov.uk/2022/03/30/fip-info/.

[16] Anonymous (). *The Net Zero Growth Plan and our farming offer - Farming*. Available: https://defrafarming.blog.gov.uk/2023/04/06/the-net-zero-growth-plan-and-our-farming-offer/.

[17] D. Helm, "Agriculture after Brexit," *Oxf Rev Econ Policy,* vol. 33, pp. S124-S133, 2017. Available: https://doi.org/10.1093/oxrep/grx010. DOI: 10.1093/oxrep/grx010.

[18] A. Maier, A. Sharp and Y. Vagapov, "Comparative analysis and practical implementation of the ESP32 microcontroller module for the internet of things," in 2017, . DOI: 10.1109/ITECHA.2017.8101926.

[19] Anonymous "DOIT ESP32 DevKit V1 Wi-Fi Development Board - Pinout Diagram & Arduino Reference - CIRCUITSTATE Electronics," 2022. Available: https://www.circuitstate.com/pinouts/doit-esp32-devkit-v1-wifi-development-board-pinout-diagram-and-reference/.

[20] M. Richardson and S. Wallace, *Getting Started with Raspberry PI.* 2012.

[21] Anonymous "Raspberry Pi," 2023. Available: https://en.wikipedia.org/w/index.php?title=Raspberry_Pi&oldid=1180130522.

[22] D. Soni and A. Makwana, "A survey on mqtt: A protocol of internet of things (iot)," in *International Conference on Telecommunication, Power Analysis and Computing Techniques (ICTPACT-2017),* 2017, .

[23] S. Adla *et al*, "Laboratory Calibration and Performance Evaluation of Low-Cost Capacitive and Very Low-Cost Resistive Soil Moisture Sensors," *Sensors,* vol. 20, *(2),* 2020. . DOI: 10.3390/s20020363.

[24] Anonymous (). *Capacitive Soil Moisture Sensor*. Available: https://thepihut.com/products/capacitive-soil-moisture-sensor.

[25] B. Sensortec, "BME280 Combined humidity and pressure sensor," *Bosch Sensortec,* 2020.

[26] A. Industries. (). *Adafruit BME280 I2C or SPI Temperature Humidity Pressure Sensor*. Available: https://www.adafruit.com/product/2652.

[27] Y. Fang and L. Xiong, "General mechanisms of drought response and their application in drought resistance improvement in plants," *Cellular and Molecular Life Sciences,* vol. 72, *(4),* pp. 673-689, 2015. Available: https://doi.org/10.1007/s00018-014-1767-0. DOI: 10.1007/s00018-014-1767-0.

[28] T. C. Wehner, "Watermelon," in *Vegetables I: Asteraceae, Brassicaceae, Chenopodicaceae, and Cucurbitaceae*, J. Prohens and F. Nuez, Eds. 2008, Available: https://doi.org/10.1007/978-0-387-30443-4_12. DOI: 10.1007/978-0-387-30443-4_12.

[29] P. Müller, X. Li and K. K. Niyogi, "Non-Photochemical Quenching. A Response to Excess Light Energy1," *Plant Physiol.,* vol. 125, *(4),* pp. 1558-1566, 2001. Available: https://doi.org/10.1104/pp.125.4.1558. DOI: 10.1104/pp.125.4.1558.

[30] B. Thomas and D. Vince-Prue, *Photoperiodism in Plants.* 1996.

[31] Anonymous (). *Wireless Watering System - LinkTap*. Available: http://www.link-tap.com/.