UNIVERSITY OF GLOUCESTERSHIRE
at Cheltenham and Gloucester

Designing and Developing an IOT Project: Home Environmental Monitoring System

Date of Completion: 22nd December 2024

Table of Contents

1. Introduction....2
1.1. Overview of the IOT Project....2
2. Design and Specifications....3
2.1. System Design & Data Representation Diagrams....3
2.2. Ethical Considerations....5
3. Development Process of the System....5
3.1. Hardware & Coding Implementations....5
3.2. IOT System Integration....11
4. Testing & Maintenance....11
4.1. Test Cases & Results....11
5. Conclusion....15
6. Appendix....15
7. Reference List....16

Introduction

The concept and implementation of Internet of Things (IoT) has led to innovative technologies involving network connectivity and computer processing linked with common objects. These objects use sensors, actuators and microprocessors to produce, transfer and receive data based on specific factors such as heart rate however the key benefit of this technology is to help minimize manual interaction on a daily. As much a spectacle it can be, several issues arise when operating IoT systems such as privacy concerns and higher chances of vulnerabilities for cyberattacks due to weak security compared to other computer systems along with minor malfunctions with corrupted data due to security and poor communication with users (Rose, Eldridge & Chapin, 2015). This document briefly covers the initial planning, implementation, testing and maintenance of a simple IoT project that mimics a Home Environmental Monitoring System (HEMS) for users to view and manage environmental conditions.

1.1. Overview of the IOT Project

A Home Environmental Monitoring System, or HEMS operates within households, farms or workspaces to assist with automatic monitoring on multiple environmental quantities such as temperature, air quality and humidity levels for growing and managing crops to seeking comfortability in your room. There are many computer hardware components, specialized IoT features and architectural designs which must be considered when constructing this system. Key hardware components include sensors to detect these background environmental quantities and display hardware like light-emitting diodes and buzzers to alert users when certain conditions are triggered based on the quantities. Additionally, important IoT principles relating to protocols and data analysis integrated with system architecture are recommended to successfully build and utilize the performance of available components for the system.

Due to the lack of a compatible Wi-Fi module, the HEMS IoT project cannot be defined as a complete IoT system, providing zero remote access however the concept is briefly mentioned to understand the theoretical usage and significance to the HEMS device.

Design and Specification Process

2.1. System Design & Data Representation Diagrams

It is vital to acknowledge, collect and document initial stakeholder requirements to generate a firm plan on the entities (stakeholders), the functional and non-functional requirements, system structure and the data flow of gathered information to correctly input and configure the relevant hardware and features (Silva, Gonçalves & da Rocha, 2019). Traditional and agile techniques, notably Scrum will be implemented into this project to gather and modify requirements. Figure 1 presents the production of a Product Backlog with requirements/tasks with user stories and a Scrum Backlog during 2-week sprints in the development phase to develop a usable prototype of the system.

The main stakeholder focused on are users living in common households therefore no monitoring on specific complex quantities is necessary, meaning basic values for temperature, air quality and humidity are collected. The following requirements:

Functional Requirements	***Non-Functional Requirements***
- Temperature is tracked	- Reading should be accurate for all quantities
- All quantity readings are visually shown in real-time on the OLED	- All readings should be logged and viewed using remote access
- Buzzer beeps when temperature crosses certain threshold	- A red LED should glow when temperature threshold is crossed
- Air quality is tracked (CO2)	- A yellow LED should glow when air quality critical value is passed
- Buzzer beeps when air quality below a specific value	- A green LED should glow when humidity threshold is crossed
- Humidity is tracked	- Use of remote access

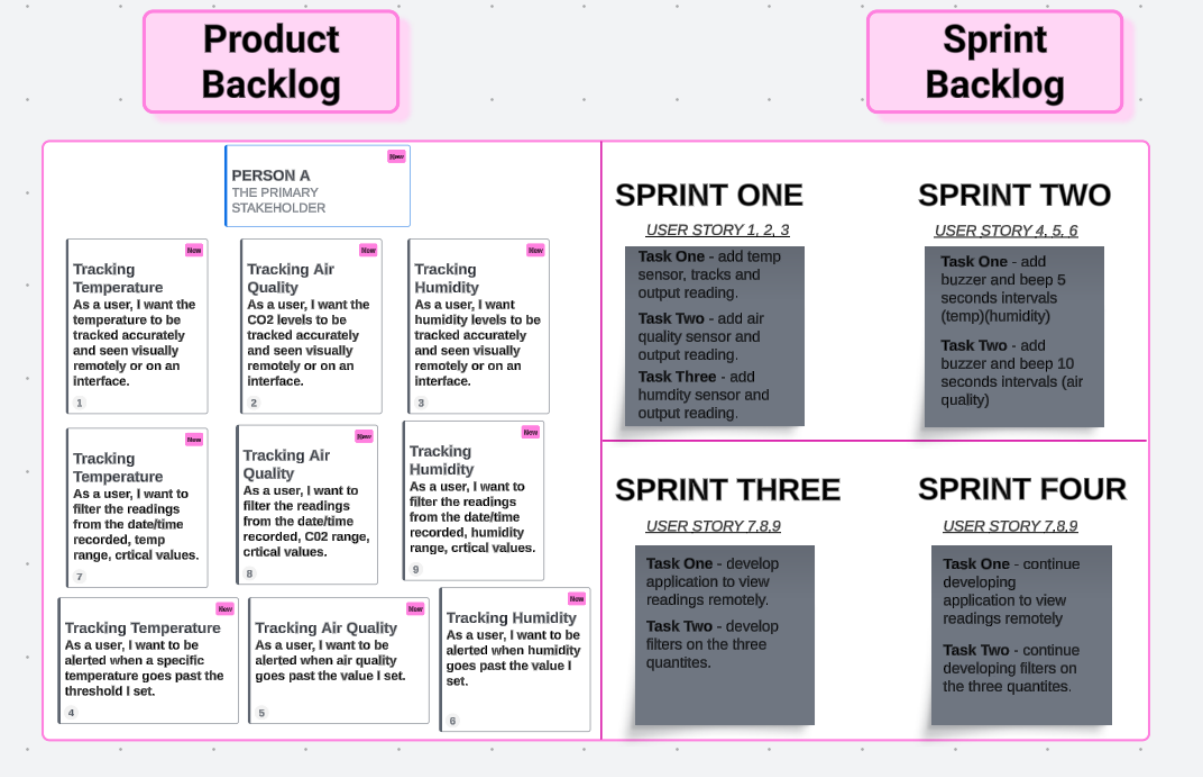

Figure 1 – custom made Product and Scrum backlog with user stories

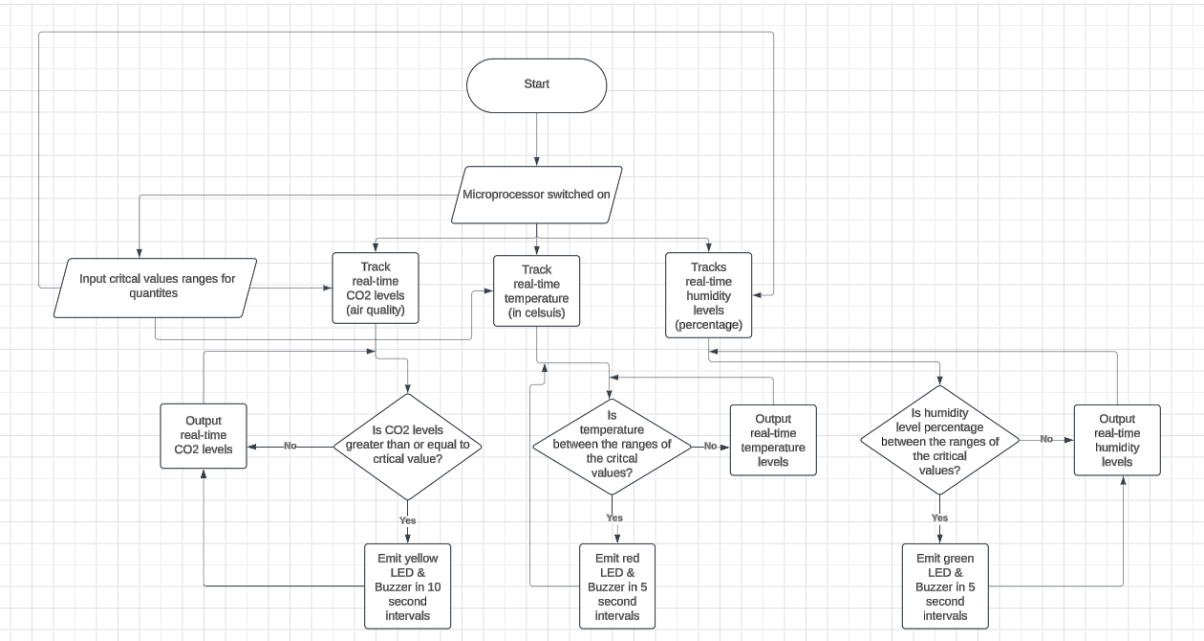

Figure 2 – flowchart representing the flow of data on the IoT device

IoT architecture usually consist of four layers essential for full automation – the sensing layer that is responsible for data collection using sensors to measure quantities, network layer which provides communication to devices, data processing layer for data analysis and application layer reveals the user output from the data (Li et al, 2015). HEMS is built on the backbone of these four layers from collecting the environmental quantity measurements using specialized sensors, communicating with these sensors, auditory and display devices, commit data processing using a microcontroller and release this data as useful output using an appropriate format for the user to understand. Figure 2 presents a flowchart which resembles the data flow that HEMS should replicate as a basis. The Arduino microprocessor is switched on, an input is executed for the user to add each quantities' critical value or range if applicable (temperature and humidity have upper-lower critical ranges) and the sensors are activated, tracking real-time readings in the background. When a reading executes True to the condition of a reading falling in the critical range/equal to the critical value, a specific LED light will blink and the buzzer will activate in a certain time interval.

2.2. Ethical Considerations

Rose, Eldridge & Chapin (2015) states that IoT technologies require and rely on the trustworthiness of sensitive information gathered and must ensure the protection of private data to avoid ethical concerns. The HEMS IoT device connects with the university "Eduroam" Wi-Fi therefore there is a risk of unusual connections with other devices in the network, prompting networking vulnerabilities for cyberattacks. This means authentication is key by applying specialized network protocols like MQTT, especially when implementing remote access (Tawalbeh et al, 2020). This project has considered and follows the University of Gloucestershire's rules, regulations and code of conduct to respect, modify and manipulate the collected data without causing vulnerabilities in security or result in severe criminal offences (University of Gloucestershire, 2021). Handling data in general requires high level of responsibility.

Development Process of the System

3.1. Hardware & Coding Implementations

For the HEMS project, several hardware components are required:

- **Temperature & Humidity Sensor** -> measures temperature & humidity levels.
- **Air Quality Sensor** -> measures the CO2 levels for air quality.
- **Arduino Microcontroller** -> to process data and control other components.

- **Breadboard** -> hold all the hardware components together.
- **LCD Display** -> show the real-time environmental quantities data.
- **Piezo Buzzer** -> auditory indicator when quantities cross critical ranges/values.
- **LED Lights** -> a visual indicator when quantities cross critical ranges/values.
- **Wi-Fi Module** -> for network connection of the Arduino.
- **Web Interface/Mobile App** -> visual access to the data remotely.
- **Resistors** -> to reduce current travelling through components on breadboard.
- **Jumper Cables** -> to connect components together.

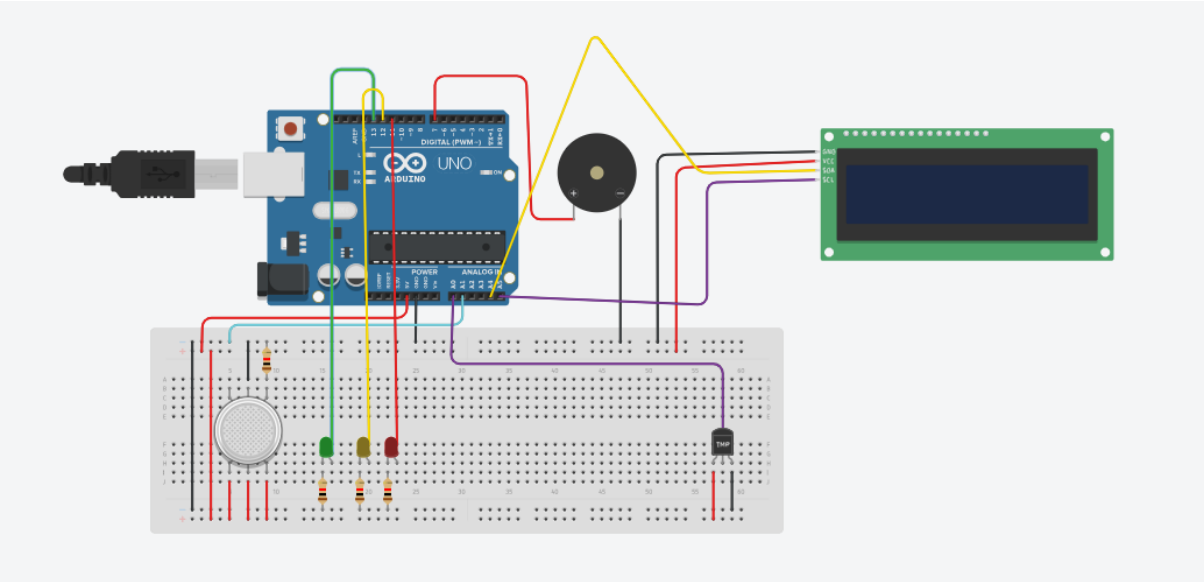

Figure 3 – initial HEMS IoT device model (Tinkercad.com)

The configuration of the HEMS device is similar to the model shown in Figure 3 except the TMP36 Temperature Sensor is replaced with the DHT11 or DHT22 Temperature & Humidity Sensor which measures both environmental quantities simultaneously.

SPRINT ONE

Due to financial issues, the Wi-Fi Module and the Air Quality Sensor were not configured therefore wireless remote access capabilities and the basic functionality to measure CO2 levels of the system were excluded. However, these setbacks did not cause significant issues with the project as the other two quantities were measured.

Figure 4 – hardware components

Before integrating hardware together on the breadboard, it was required to test them individually to verify working status with the Arduino Uno. All hardware passed the unit tests but the light crystal display exhibited some issues due to impractical configuration without the I2C module but backlight and display was visible. To begin, the sensor was added on the basic circuit of the breadboard to produce a flowing current, connecting to the 5V and Ground pins of the Arduino Uno as well with required pins for the sensor. Arduino offers an Integrated Development Interface (IDE) for coding to operate data control for hardware components. Figure 6 shows the specific library compatible to the sensor. It was downloaded and called in the IDE, presenting and executing an example of pre-written code to gather and output the humidity, heat index and temperature (in Celsius & Fahrenheit) from the sensor. After verifying the sensor worked with readings shown in the Serial Monitor (shown in Figure 5), a successful attempt occurred to output the readings on the LCD in real-time although random symbols randomly respawned due to faulty wiring or placement on the breadboard.

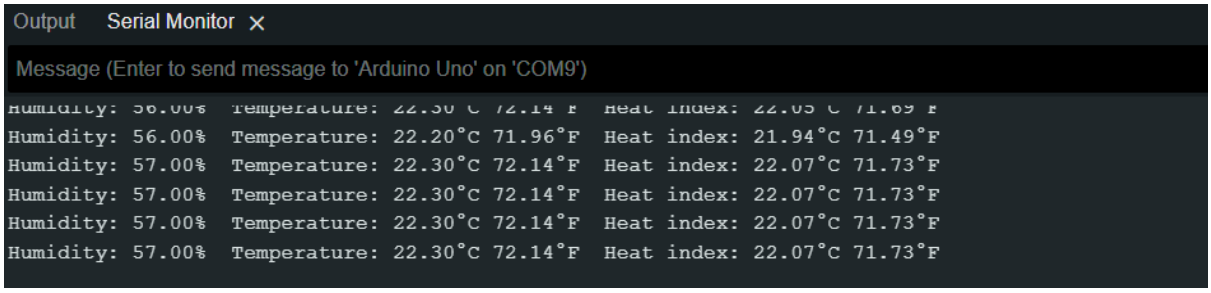

Figure 5 – humidity, temperature and heat index readings on Arduino IDE Serial Monitor

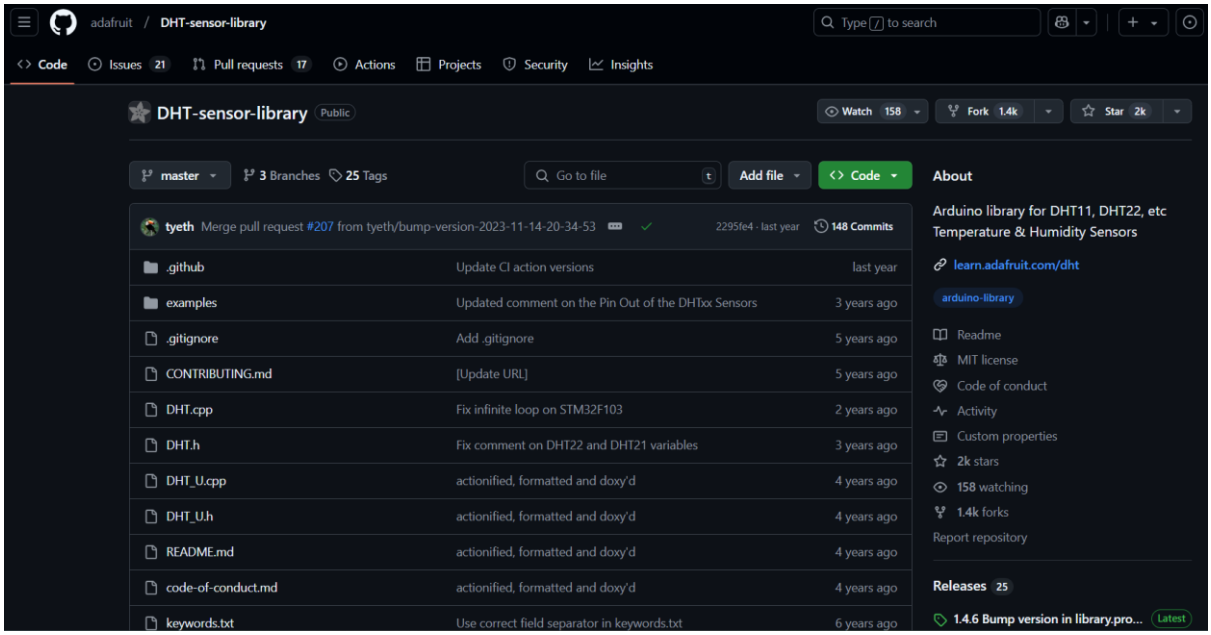

Figure 6 – Adafruit DHT Temperature/Humidity Sensor Library on GitHub

SPRINT TWO

Later on, a potentiometer was enabled to the HEMS to control the contrast of the LCD display, reducing the voltage applied to the contrast pin (V0 pin). It provided better readability for users and efficient for power usage. This sprint focused on applying the display devices as indicators to alert users when temperature and humidity levels crossed critical values/ranges. A buzzer and two LED lights were configured as a result which the red LED represented the temperature and the green LED represented humidity while the buzzer performed different frequences and beep intervals. The below figure reveals applying selection statements to dictate how the HEMS device responds when specific conditions are True. In this case, the temperature readings are less than 25 degrees Celsius therefore the LED light shines red and the buzzer emits at a frequency of 40Hz every 5 milliseconds until the executed statement becomes false.

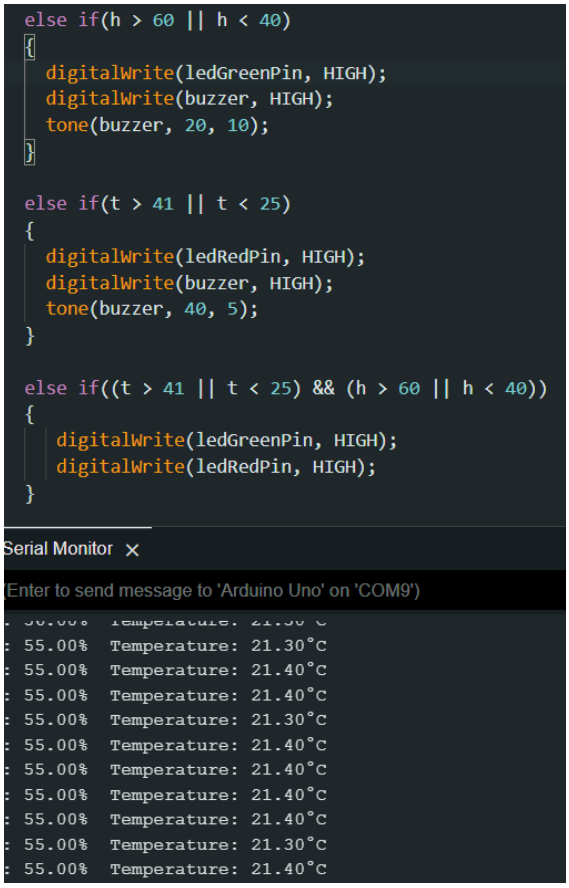

Figure 7 – selection statements on temperature/humidity readings

After the two sprints were completed, a simple web interface was planned to be developed to permit USB serial connection rather than applying Wi-Fi or Ethernet for remote access for users to view the live data. Potential features conjured were the possibility of including filters for the user to categorize the quantities (only viewing previous critical temperatures, time recorded, etc) however this was not a definite priority rather it is was defined as a non-functional feature to improve user experience.

SPRINT THREE

The main functionalities of the HEMS device were completed therefore a simplistic functional interface was configured with minimal widgets to output the real-time temperature and humidity levels, shown in Figure 8. Arduino offers an online cloud synchronisation service called Arduino Cloud with the purpose to connect Arduino microcontrollers to other systems to transmit data using WI-FI but the Arduino Uno SMD Edition was not compatible. An alternative simple IoT cloud service named RemoteXY was accepted due to the capability to allow USB Serial connections.

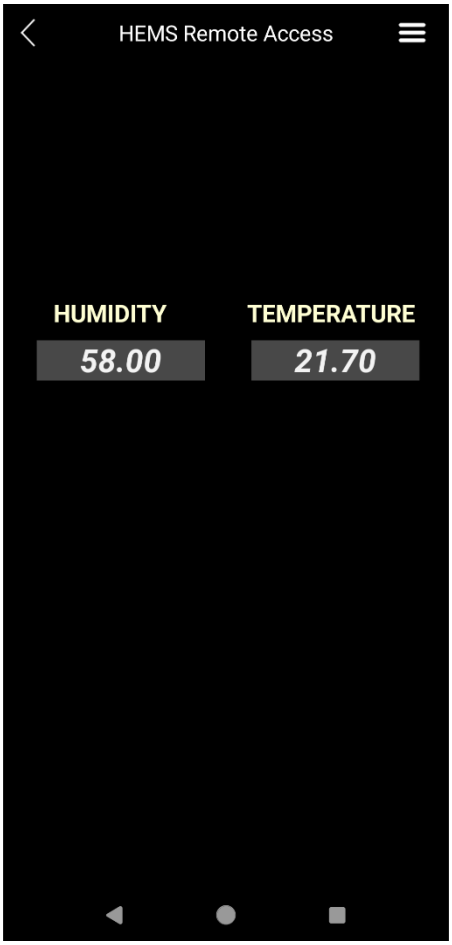

Figure 8 – RemoteXY simple mobile interface on the HEMS IoT device

SPRINT FOUR

The final sprint of this project prioritised on applying the non-functional features like filtering readings and LED indicator widgets on the simple mobile interface however this failed as it required premium access for RemoteXY. As a result, further trials of unit and system testing were executed on the existing configured system to ensure consistent operation. At the end, the integrated HEMS device went through multitudes of testing.

3.3. IOT System Integration

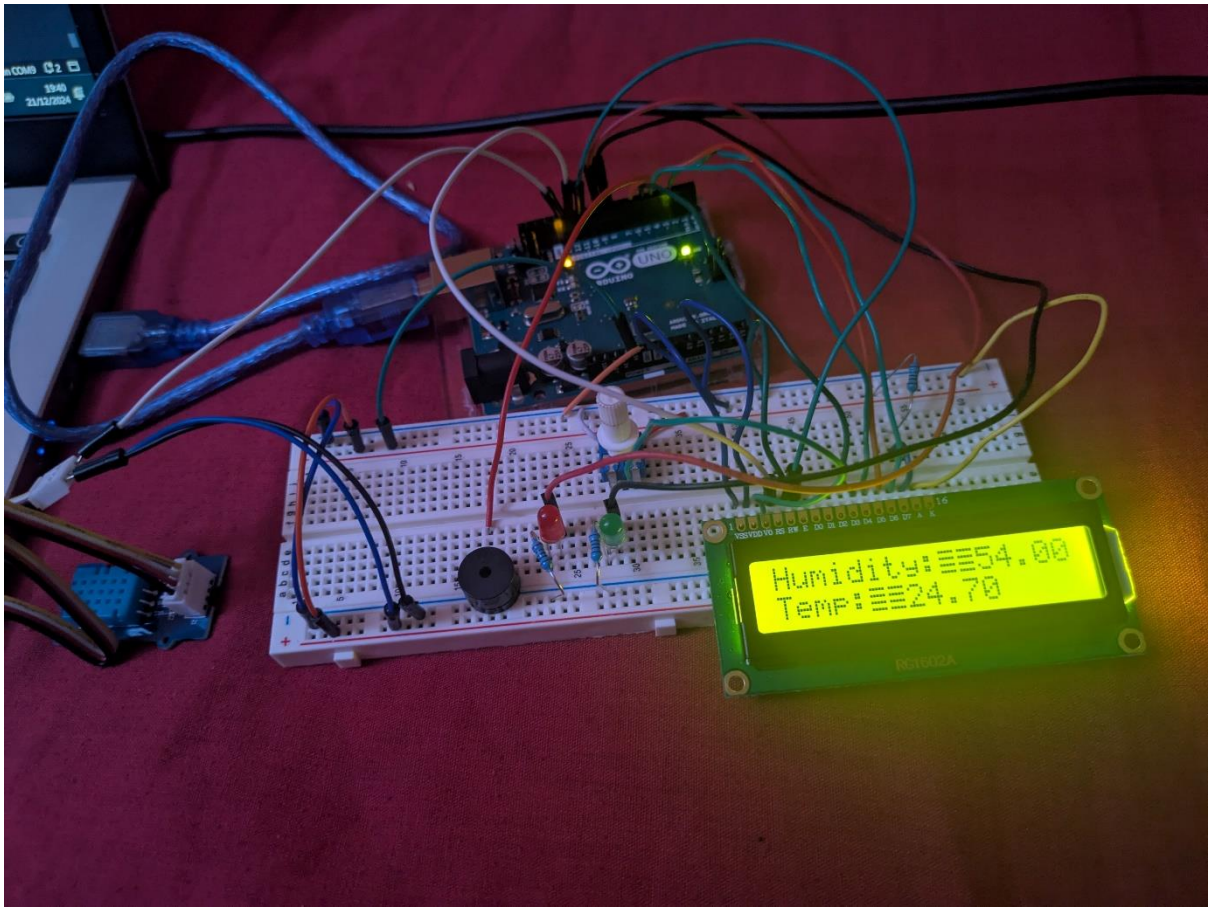

Figure 9 – integrated IoT HEMS device

Testing & Maintenance

4.1. Test Plan, Cases & Improvements

Several categories were formulated within the test plan with appropriate test suites and cases for organisation, targeting specific functionality and quality of performance for improvement. All software and hardware were tested individually and systematically during and after complete development of the IoT project, resulting in improvements.

The following test suites established:

- Primary Functionality (PF)
- Compatibility With Hardware & Software (CWHS)
- Change In Data (CID)

Before & After Complete IoT HEMS Integration

Test Case 2 & 6 -> LCD needed to be adjusted on the breadboard for register select, enable and data pins to connect properly. This resulted in a readable output.

Test Case 10 -> a Python script had to be ran to enable the laptop as a serial bridge for data to transfer between the Arduino Uno and the mobile app interface.

Test Table #1

ID	Action	Inputs	Desired Output	Actual Output	Type	Result
1	DHT11 Temp/Hum Sensor gathering readings	Environment	Temp and humidity levels output on Serial Monitor	Temp and humidity readings are printed on Serial Monitor	PF	Pass
2	LCD backlight & display activated	N/A	Yellow backlight on + “Hello World” displayed	Yellow backlight switched on + random weird symbols displayed	PF	Fail
3	Buzzer beeps	N/A	Buzzer beeps at a frequency of 1000Hz	Buzzer beeps at correct frequency	PF	Pass
4	LEDs glow	N/A	Green and Red LEDs glow	Green and Red LEDs glow	PF	Pass
5	Arduino Uno powers on	N/A	It is switched on	It is switched on	PF	Pass
6	LCD displays live temp/hum readings with sensor	N/A	Readings should be accurately displayed on the LCD display	Some words/numbers are displayed however the majority of the text are weird symbols + not live	CWHS	Fail

7	Green LED blinks and buzzer beeps at frequency of 1000 every 10 milliseconds	N/A	LED glows and buzzer beeps at frequency of 1000 every 10 milliseconds	LED glows and buzzer beeps at frequency of 1000 every 10 milliseconds	CWHS	Pass
8	Red LED blinks and buzzer beeps at frequency of 1000 every 5 milliseconds	N/A	LED glows and buzzer beeps at frequency of 1000 every 5 milliseconds	LED glows and buzzer beeps at frequency of 1000 every 5 milliseconds	CWHS	Pass
9	Potentiometer changes LCD display contrast	Slider	Adjusts LCD display contrast	Adjusts LCD display contrast	CWHS	Pass
10	Remote access enabled	N/A	RemoteXY mobile interface activated	RemoteXY mobile interface failed to load	PF	Fail
11	Mobile interface displays live temp/hum readings with sensor	N/A	Readings displayed on RemoteXY mobile interface	Readings displayed on RemoteXY mobile interface but it is delayed by 2 seconds	CWHS	Pass
12	Red and Green LED blink and buzzer beeps at frequency of 2000 every 5 milliseconds when	N/A	When temperature and humidity levels cross, both LEDs and buzzer alerted	When temperature and humidity levels cross, both LEDs and buzzer alerted	CID	Pass

	temp and humidity thresholds crossed					
13	Green LED blinks and buzzer beeps at frequency of 1000 every 10 milliseconds when humidity thresholds crossed	N/A	When humidity levels cross, green LED and buzzer alerted	When humidity levels cross, green LED and buzzer alerted.	CID	Pass
14	Red LED blinks and buzzer beeps at frequency of 1000 every 5 milliseconds when temp thresholds crossed	N/A	When humidity levels cross, red LED and buzzer alerted	When humidity levels cross, red LED and buzzer alerted	CID	Pass

Conclusion

This IoT project serves as a basic example on how complex yet simplified these technologies are and the several capabilities, they possess from tailored data processing for users to remote access using mobile and web graphic interfaces. As the future progresses, it is a high chance these devices will be the leading technology.

Appendix

Remote Access to the HEMS Device

1. Configure the HEMS setup. This includes:
- DHT11 Temperature & Humidity Sensor to the Arduino on Pin 13, GND and 5V
- LCD – VSS(GND), VDD(5V), V0(Potentiometer), RS(Pin 12), RW(GND), E(Pin 11), D4(Pin 5), D5(Pin 4), D6(Pin 3), D7(Pin 2), A(5V), K(GND)
- Buzzer on Pin 6
- Green LED on Pin 8, Red LED on Pin 9

2. Verify and upload the “hems.ino” source code into the Arduino.

3. Next, close the Arduino IDE application and open the “SERIAL_bridge” python script.

4. Connect mobile device to computer. On the RemoteXY Mobile App, add new device. Choose Ethernet. Input the computer local IP address. The port should automatically be generated.

5. If you choose Wi-Fi, it requires login to the network.

6. Modify the Python script to replace Arduino serial communication port connected, computer local IP address and the port for communication with the RemoteXY mobile app when connecting using Ethernet or WI-Fi.

7. The mobile interface should show like Figure 8 however the real-time temperature and humidity readings should appear.

Reference List

Rose, K., Eldridge, S. and Chapin, L., 2015. The internet of things: An overview. The internet society (ISOC), 80(15), pp.1-53. Accessed on: 26th September 2024

Silva, D., Gonçalves, T.G. and da Rocha, A.R.C., 2019, October. A Requirements Engineering Process for IoT Systems. In Proceedings of the XVIII Brazilian symposium on software quality (pp. 204-209). Accessed on: 26th September 2024

Li, S., Xu, L.D. and Zhao, S., 2015. The internet of things: a survey. *Information systems frontiers*, *17*, pp.243-259. Accessed on: 26th September 2024

Tawalbeh, L., Muheidat, F., Tawalbeh, M. and Quwaider, M. (2020). IoT Privacy and Security: Challenges and Solutions. *Applied Sciences*, [online] 10(12), p.4102. doi:https://doi.org/10.3390/app10124102. Accessed on: 16th November 2024

Knowledge Base. (2021). *Student Code of Conduct - Knowledge Base - University of Gloucestershire*. Available at: https://www.glos.ac.uk/information/knowledge-base/student-code-of-conduct/ . Accessed on: 18th November 2024

Arduino Documentation. (2024). Getting Started with Arduino Cloud. Available at: https://docs.arduino.cc/arduino-cloud/guides/overview/ . Accessed on: 26th November 2024