

**Development of Arduino-based Smart Cloth Drying System**

By

Chia Shen Yang

A REPORT

SUBMITTED TO

Universiti Tunku Abdul Rahman

in partial fulfillment of the requirements

for the degree of

BACHELOR OF COMPUTER SCIENCE (HONOURS)

Faculty of Information and Communication Technology

(Kampar Campus)

JUNE 2024

## REPORT STATUS DECLARATION FORM

**Title:** Development of Arduino-based Smart Cloth Drying System

**Academic Session:** JUNE 2024

I CHIA SHEN YANG

declare that I allow this Final Year Project Report to be kept in  
Universiti Tunku Abdul Rahman Library subject to the regulations as follows:

1. The dissertation is a property of the Library.
2. The Library is allowed to make copies of this dissertation for academic purposes.



\_\_\_\_\_  
(Author's signature)

Verified by,



\_\_\_\_\_  
(Supervisor's signature)

**Address:**

No A1, Kampung Baru Ban Foo,  
81800 Ulu Tiram, Johor.

OH ZI XIN

\_\_\_\_\_  
Supervisor's name

**Date:** 13/09/2024

**Date:** 13/9/2024  
\_\_\_\_\_

<b>Universiti Tunku Abdul Rahman</b>			
Form Title: <b>Submission Sheet for FYP</b>			
Form Number: <b>FM-IAD-004</b>	Rev No.: <b>0</b>	Effective Date: <b>21 JUNE 2011</b>	Page No.: <b>1 of 1</b>

**FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY**

**UNIVERSITI TUNKU ABDUL RAHMAN**


Date: 13/9/2024

**SUBMISSION OF FINAL YEAR PROJECT THESIS**

It is hereby certified that *Chia Shen Yang* (ID No: 22ACB00552) has completed this final year project entitled “**Development of Arduino-based Smart Cloth Drying System**” under the supervision of Ms. Oh Zi Xin (Supervisor) from the Department of Computer and Communication Technology, Faculty of Information and Communication Technology.


I understand that University will upload softcopy of my final year project in pdf format into UTAR Institutional Repository, which may be made accessible to UTAR community and public.

Yours truly, Chia Shen Yang.

Signature :   
Name : Chia Shen Yang

## DECLARATION OF ORIGINALITY

I declare that this report entitled “**Development of Arduino-based Smart Cloth Drying System**” is my own work except as cited in the references. The report has not been accepted for any degree and is not being submitted concurrently in candidature for any degree or other award.

Signature : 

Name : Chia Shen Yang

Date : 13/09/2024

## **ACKNOWLEDGEMENTS**

I would like to express my sincere thanks and appreciation to my supervisors, Ms. Oh Zi Xin who has given me this bright opportunity to engage in a IoT-based project. It is my first step to establish a career in IT Career. A million thanks to you.

Finally, I must say thanks to my parents and my family for their love, support, and continuous encouragement throughout the course.

## **ABSTRACT**

In Malaysia, the climate often fluctuates between extreme heat and sudden heavy rainfall, causing inconvenience for residents who are away from home. To address this issue, this study aims to create a smart clothing drying system using Arduino technology. This system allows for semi-automatic or fully automatic clothes drying, adapting to changing weather conditions. The system's core is an Arduino, which interfaces with sensors measuring humidity, temperature, air pressure, light intensity, UV intensity, and rainfall to gather comprehensive environmental data. This data feeds into a model that evaluates if the current weather conditions are suitable for air-drying clothes. The system utilizes predefined strategies and thresholds to make autonomous decisions about whether to proceed with clothes drying. Arduino controls two DC motors to move the clothes rack to suitable locations, such as indoors during unfavorable weather or outdoors on sunny days. For remote control, Arduino is equipped with Wi-Fi connectivity, enabling a mobile app through the Blynk IoT platform for precise manual adjustments. This app not only offers remote control for retrieving or hanging clothes but also provides real-time sensor data and notifications about clothing rack movements and potential rain. The system incorporates advanced features such as manual remote control of wheel movements, path recording and playback, and reverse path execution. All settings and recorded paths are stored in EEPROM, ensuring persistence across power cycles. Additionally, the system considers sunrise and sunset times to determine its operational hours. This intelligent system provides users with convenient control over clothes drying, whether through automated decisions based on sensor data and predefined strategies or remote manual operations, thus simplifying daily life and improving energy efficiency.

# TABLE OF CONTENTS

<b>TITLE PAGE</b>	<b>i</b>
<b>REPORT STATUS DECLARATION FORM</b>	<b>ii</b>
<b>FYP THESIS SUBMISSION FORM</b>	<b>iii</b>
<b>DECLARATION OF ORIGINALITY</b>	<b>iv</b>
<b>ACKNOWLEDGEMENTS</b>	<b>v</b>
<b>ABSTRACT</b>	<b>vi</b>
<b>TABLE OF CONTENTS</b>	<b>vii</b>
<b>LIST OF FIGURES</b>	<b>viii</b>
<b>LIST OF TABLES</b>	<b>ix</b>
<b>LIST OF ABBREVIATIONS</b>	<b>x</b>
<b>CHAPTER 1 INTRODUCTION</b>	<b>1</b>
1.1 Problem Statement and Motivation	1
1.2 Research Objectives	2
1.3 Project Scope and Direction	3
1.4 Contributions	4
1.5 Report Organization	5
<b>CHAPTER 2 LITERATURE REVIEW</b>	<b>6</b>
2.1 Previous Works	6
2.1.1 Automatic Control Circuit Design for a Clothes	7
2.1.2 Designing an Internet of Things Based Automatic Clothesline	12
2.1.3 Design and Experimental Study on Automatic Cloth Retrieval and Drying System	15
2.1.4 Automatic Cloth Drying Line Solution Using IoT	18
2.1.5 Summary of Previous Works	20
2.2 Weather API Selection	22
2.2.1 AccuWeather APIs	22
2.2.2 Open Weather API	26
2.2.3 MET API	29
2.3 Arduino Hardware Selection	32
2.3.1 Spark Fun Arduino IoT Weather Station	32

<b>CHAPTER 3</b>	<b>SYSTEM METHODOLOGY/APPROACH</b>	
3.1	System Requirement	34
3.1.1	Hardware	34
3.1.2	Software	38
3.2	System Design Diagram	39
3.2.1	Smart Clotheshorse System Architecture	39
3.2.2	Automation Flowchart Diagram	40
3.2.3	Replay Sequence Diagram	41
3.3	Timeline (Gantt Chart)	43
<b>CHAPTER 4</b>	<b>SYSTEM IMPLEMENTATION</b>	<b>45</b>
4.1	Arduino Connection	45
4.2	Blynk Interface	46
4.3	Blynk Notification	48
4.4	Blynk Sunrise and Sunset Automation	49
4.5	Implementation Issues and Challenges	49
<b>CHAPTER 5</b>	<b>SYSTEM EVALUATION AND DISCUSSION</b>	<b>51</b>
5.1	Testing Setup and Result	51
5.2	Objectives Evaluation	52
<b>CHAPTER 6</b>	<b>CONCLUSION AND RECOMMENDATION</b>	<b>53</b>
6.1	Conclusion	53
6.2	Recommendation	54
<b>REFERENCES</b>		<b>55</b>
<b>APPENDIX</b>		
<b>WEEKLY LOG</b>		<b>A-1</b>
<b>POSTER</b>		<b>A-2</b>
<b>ARDUINO CODING</b>		<b>A-3</b>
<b>PLAGIARISM CHECK RESULT</b>		<b>A-4</b>
<b>FYP2 CHECKLIST</b>		<b>A-5</b>



## LIST OF FIGURES

<b>Figure Number</b>	<b>Title</b>	<b>Page</b>
Figure 2.1	Component	8
Figure 2.2	Automatic drying tool design	13
Figure 2.3	Weather Alarms API	22
Figure 2.4	Weather Alarm Thresholds	22
Figure 2.5	Packages and Pricing	23
Figure 2.6	Daily Forecast	24
Figure 2.7	Current Conditions	25
Figure 2.8	Hourly Forecast	25
Figure 2.9	Location Key	25
Figure 2.10	Countries	25
Figure 2.11	One Call API	26
Figure 2.12	Special products	27
Figure 2.13	Global Weather Alerts Push notifications	27
Figure 2.14	Current & Forecast weather data collection	28
Figure 2.15	API Token	29
Figure 2.16	Throttling	29
Figure 2.17	Available Data	30
Figure 2.18	Warnings	30
Figure 2.19	General Forecast	31
Figure 2.20	Locations	31
Figure 2.21	Arduino IoT Weather Station kit content	32
Figure 2.22	SparkFun Arduino IoT Weather Station	33
Figure 3.1	Arduino UNO R4 Wi-Fi	35
Figure 3.2	BH1750 Light Intensity Sensor	35
Figure 3.3	DHT22 Temperature and Humidity Sensor	36
Figure 3.4	GYML8511 UV Intensity Sensor	36
Figure 3.5	Rain Drop Sensor FC-37 or YL-83	37
Figure 3.6	L298N Motor Driver, Motor and Battery Holder	37
Figure 3.7	System Diagram	39

Figure 3.8	Automation Flowchart Diagram	40
Figure 3.9	Replay Sequence Diagram	41
Figure 3.10	Timeline Gantt Chart	43
Figure 4.1	Arduino Connection	45
Figure 4.2	Blynk Interface	46
Figure 4.3	Blynk Notification	48
Figure 4.4	Blynk Sunrise and Sunset Automation	49
Figure 5.1	Serial Monitor Output	51

## LIST OF TABLES

<b>Table Number</b>	<b>Title</b>	<b>Page</b>
Table 2.1	Specifications of Hardware	7
Table 2.2	Fuzzy Rules	10
Table 2.3	Specifications of Hardware	12
Table 2.4	Specifications of Hardware	15
Table 2.5	Literature Review	17
Table 2.6	Specifications of Hardware	18
Table 2.7	Specifications of Control Unit and Wi-Fi Module	20
Table 2.8	Specifications of Sensor	20
Table 3.1	Specifications of laptop	34
Table 3.2	Specifications of Arduino and Sensor	34

## LIST OF ABBREVIATIONS

<i>IoT</i>	Internet of Things
<i>API</i>	Application Programming Interface
<i>DC</i>	Direct Current
<i>UV</i>	Ultraviolet
<i>Wi-Fi</i>	Wireless Fidelity
<i>LDR</i>	Light Dependent Resistor
<i>AC</i>	Alternating Current
<i>GBP</i>	British Pound
<i>SSD</i>	Solid State Drive
<i>IDE</i>	Integrated Development Environment
<i>APP</i>	Application
<i>MET</i>	Meteorological

# CHAPTER 1

## INTRODUCTION

The project titled "Development of an Arduino-based Intelligent Clothing Drying System" aims to create an innovative solution for clothing drying by designing and implementing a smart system that utilizes Arduino technology. This system will autonomously detect rainwater and sunlight conditions to optimize the clothing drying process, enhancing user convenience and experience. By integrating sensors for temperature, humidity, light intensity, UV intensity, and rainfall, along with actuators, the Arduino-based system will effectively assess environmental conditions and control the clothing drying process.

### 1.1 Problem Statement and Motivation

Traditional clothes drying racks require constant attention to weather conditions when hanging clothes out in sunny areas. They need to be promptly retracted if there are signs of rain. If no one is at home, whether the clothes dry or not becomes a matter of luck. Good weather poses no issue, but if it rains, the clothes need to be laundered again. While drying machines are convenient, not everyone possesses one, and some individuals prefer naturally sun-dried clothes.

The challenge lies in creating an intelligent system that can adapt to ever-changing weather conditions, pausing drying during rain and resuming when ample sunlight is available, thereby achieving a semi-automatic or even fully automatic clothes drying process. For instance, imagine someone who is busy working or away from home while their laundry is hanging outside to dry. If an unexpected rain shower occurs, the clothes could get soaked, requiring them to be washed again. This situation can be inconvenient and lead to additional time and effort.

To address this issue, developing an intelligent clothing drying system enables users to remotely monitor weather conditions and control the drying status of their clothes. This way, users don't have to personally keep an eye on the weather near their homes, ensuring that their clothes are dried at the right time, thus enhancing convenience and comfort in their daily lives.

## **1.2 Research Objectives**

The objective of this project is to develop intelligent and autonomous clothes drying system based on Arduino technology. This system will seamlessly integrate hardware components like sensors and actuators with a user-friendly mobile application. By assessing various environmental data such as temperature, humidity, light intensity, UV intensity, and rainfall, the system will autonomously determine the optimal drying conditions and control the movement of the clothes rack accordingly.

The goal is to realize a fully automated drying process that operates without human intervention, providing a practical and convenient solution for the daily chore of clothes drying. To achieve this, a series of progressive steps will be undertaken, including the development of an Arduino-based controller, the integration of relevant sensors, the implementation of decision-making algorithms, and the creation of a mobile app for remote monitoring and control. Through this comprehensive approach, the project aims to enhance the efficiency and user experience of the clothes drying process, offering a smart and sustainable solution for modern households.

### **1.3 Project Scope and Direction**

The scope of the project encompasses the design and implementation of a smart clothes drying system centred around Arduino technology. Additionally, a mobile application will be developed to complement this system. The system's functionality stems from its ability to assess current weather conditions through data obtained from sensors interfaced with the Arduino, including humidity, temperature, light intensity, UV intensity, and rainfall sensors. It automatically adjusts the clothes drying process, aiming to enhance the convenience and efficiency of this household task.

The core of the system lies in the Arduino, which interfaces with sensors measuring humidity, temperature, light intensity, UV intensity, and rainfall to gather comprehensive environmental data. This data is then processed to evaluate if the current weather conditions are suitable for air-drying clothes. The system also incorporates predefined strategies and thresholds to autonomously decide whether to proceed with the drying process or not.

The Arduino controls two DC motors to move the clothes rack to suitable locations, such as indoors during unfavourable weather conditions or outdoors on sunny days. For remote control capabilities, the Arduino is equipped with Wi-Fi connectivity, enabling a mobile app through the Blynk IoT platform for precise manual adjustments.

The project also includes advanced features such as manual remote control of wheel movements, path recording and playback, and reverse path execution. All settings and recorded paths are stored in EEPROM, ensuring persistence across power cycles. Additionally, the system considers sunrise and sunset times to determine its operational hours.

## 1.4 Contributions

The contributions of this project are reflected in several aspects:

**Intelligent Clothing Drying Solution:** By integrating Arduino technology with sensors, an intelligent clothing drying system has been developed. It can adjust the drying process intelligently based on weather conditions, preventing clothes from being affected by rainwater.

**Enhanced User Experience:** By offering remote control and real-time weather feedback, the project significantly improves user convenience and satisfaction. Users no longer need to personally monitor the weather, as the system handles drying decisions automatically.

**Mobile App Development:** The creation of a mobile application enables users to conveniently control the drying system remotely, provide weather feedback, and receive notifications through the Blynk platform. This enhances the interactivity and usability of the system.

**Personalized Choices:** Implementation of both semi-automatic and fully automatic drying modes allows users to choose how their clothes are dried based on personal preferences and needs, catering to various user habits.

**Sustainability Considerations:** Through intelligent drying control, the project reduces the need for clothes to be rewashed due to rainwater exposure, thereby reducing water and energy wastage to some extent.

**Technological Innovation:** The integration of advanced features such as path recording, playback, and reverse execution, as well as storing settings and paths in EEPROM, demonstrates the potential for applying complex control systems in household appliances.

**Adaptive Design:** The feature considering sunrise and sunset times allows the system to better adapt to natural light cycles, optimizing energy use and improving efficiency.

These contributions collectively create a comprehensive, intelligent, and user-friendly clothes drying system that not only solves practical problems but also provides innovative ideas for the smart home sector.



## **1.5 Report Organization**

The details of this study will be introduced in the following chapters. The second chapter reviews some related backgrounds, including previous works on automatic cloth drying systems and a comparison of different weather APIs and Arduino hardware options. Then, in the third chapter, it introduces the system methodology and approach, detailing the hardware and software requirements, system design diagram and project timeline. The fourth chapter focuses on system implementation, describing the Blynk interface, Arduino connections, and challenges encountered during the development process. Finally, Chapter 5 presents the system evaluation and discussion, including testing setups, results and an assessment of how well the project objectives were met. The report concludes with Chapter 6, which provides a conclusion and recommendations for future improvements and extensions of the project.

## **CHAPTER 2**

### **LITERATURE REVIEWS**

#### **2.1 Previous Works**

In the following study will describe various systems with similar primary objectives and principles. Each of these systems utilizes a Control Unit to connect to sensors such as Rain Sensors, Temperature Sensors, or Light Sensors, and based on the data obtained from these sensors, they determine the weather conditions and control a motor to move clothes. While the main objectives are consistent across these systems, there are differences in the choice of Control Unit, Sensors, and the specific methods employed to move the clothes. The following content will elaborate on the characteristics of each system, summarize their differences, highlight limitations, and propose solutions.

### 2.1.1 Automatic Control Circuit Design for a Clothes Dryer [1]

#### Hardware Components

The table below outlines the hardware components used in the system along with their functionalities:

*Table 2.1 Specifications of Hardware*

<b>Component</b>	<b>Functionality</b>
ESP32 Microcontroller	Central control unit, supports Wi-Fi and Bluetooth connections, used for internet connectivity.
Light Dependent Resistor (LDR)	Detects the presence of sunlight, used to determine changes in light intensity.
Raindrop Sensor (MD0127)	Detects rainwater, generates output voltage when water droplets meet the sensor.
Temperature and Humidity Sensor (DHT22)	Detects temperature and humidity, ensuring a suitable environment for drying clothes.
Direct Current (DC) Motor	Controls the movement of clothes by toggling the polarity of the DC power source.
H-Bridge Motor Driver	Controls the rotation direction and speed of the DC motor.
Firebase Realtime Database	Stores and syncs sensor data, accessible from any platform.

### System Operating Principle

A Light Dependent Resistor (LDR) is used to detect the presence of sunlight. The resistance value of the LDR changes according to variations in light intensity, allowing the system to determine the presence of sunlight. A Raindrop Sensor (MD0127) is employed to detect rainwater. When water droplets meet the sensor, it generates a sufficient output voltage to determine the occurrence of rain. A Temperature and Humidity Sensor (DHT22) is utilized to monitor environmental temperature and humidity, ensuring suitable conditions for clothes drying. When the need to block sunlight or protect against rain is detected, the ESP32 controls a DC motor using an H-Bridge motor driver, moving the clothes to a shaded area. The system sends sensor data to the Firebase Realtime Database via Wi-Fi connectivity, making the data accessible from any device at any time.

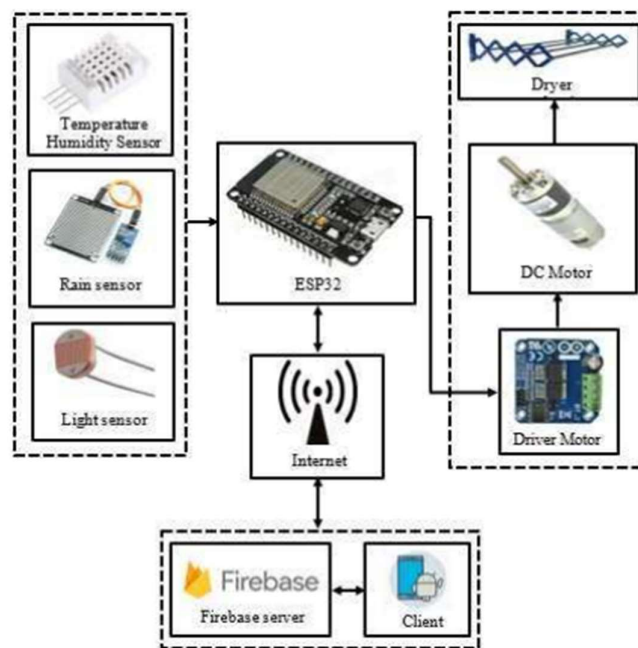


Figure 2.1 Components

### **Parameter Standards and Fuzzy Logic Control:**

The system employs fuzzy logic control to determine whether the weather is suitable for hanging clothes. The following are some parameter standards used for fuzzy logic:

1. Light Intensity Measurement: Mapping light intensity from ADC values to a percentage scale (0-100%), divided into Dark (0-50%), Bright (30-80%), and Very Bright (70-100%).
2. Humidity Measurement: Represented in percentage as relative humidity, divided into Dry (0-60%), Normal (50-90%), and Humid (80-100%).
3. Temperature Measurement: In degrees Celsius, divided into Cold (10-25°C), Warm (20-35°C), and Hot (32-50°C).

### **Steps of Fuzzy Logic Control:**

1. Map sensor measurements to their respective fuzzy sets.
2. Use fuzzy rules based on combinations of light intensity, humidity, and temperature for decision-making.
3. Employ Mamdani method's implication function (MIN) and aggregation function (MAX) to calculate whether the weather is suitable for clothes drying.
4. If deemed sunny, clothes are exposed to sunlight; otherwise, they are moved to a shaded area.

The table is structured as follows, with each row representing a combination of light intensity, humidity, and temperature conditions, along with the resultant weather conclusion:

*Table 2.2 Fuzzy Rules*

<b>No.</b>	<b>Light</b>	<b>Humidity</b>	<b>Temperature</b>	<b>Weather</b>
1	Dark	Dry	Cold	Dark
2	Dark	Dry	Warm	Bright
3	Dark	Dry	Hot	Bright
4	Dark	Normal	Cold	Dark
5	Dark	Normal	Warm	Dark
6	Dark	Normal	Hot	Bright
7	Dark	Humid	Cold	Dark
8	Dark	Humid	Warm	Dark
9	Dark	Humid	Hot	Dark
10	Bright	Dry	Cold	Bright
11	Bright	Dry	Warm	Bright
12	Bright	Dry	Hot	Bright
13	Bright	Normal	Cold	Dark
14	Bright	Normal	Warm	Bright
15	Bright	Normal	Hot	Bright
16	Bright	Humid	Cold	Dark
17	Bright	Humid	Warm	Dark
18	Bright	Humid	Hot	Bright
19	Very Bright	Dry	Cold	Bright
20	Very Bright	Dry	Warm	Bright
21	Very Bright	Dry	Hot	Bright
22	Very Bright	Normal	Cold	Bright
23	Very Bright	Normal	Warm	Bright
24	Very Bright	Normal	Hot	Bright
25	Very Bright	Humid	Cold	Dark
26	Very Bright	Humid	Warm	Bright
27	Very Bright	Humid	Hot	Bright

### **Method to Move Cloth**

Moving clothes uses components such as DC motor and H-Bridge motor driver, and the rotation direction of the motor is controlled by PWM signal, so that the clothes rope is rolled up or put down, and the clothes are taken to the position of shading or sunbathing.

The system contains many valuable aspects worth learning and referencing. For instance, it implements Parameter Standards and Fuzzy Logic Control. Since weather conditions can vary with elements like heat, cold, dryness, and humidity, and there may not be an absolute numerical threshold, the use of fuzzy logic and the Mamdani method is highly suitable.

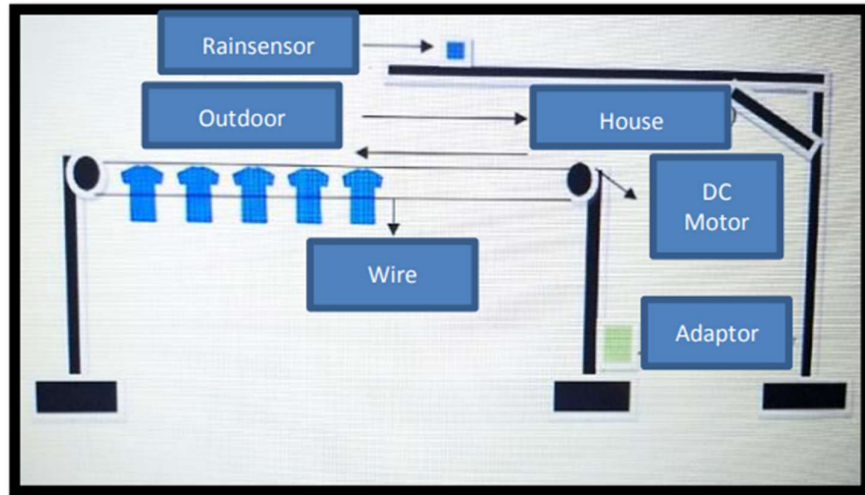
## 2.1.2 Designing an Internet of Things Based Automatic Clothesline [2]

### Hardware Components

*Table 2.3 Specifications of Hardware*

<b>Components</b>	<b>Functionality</b>
NodeMCU ESP8266	An IoT platform based on the ESP8266 microcontroller, used to control the clothes drying rack system via Wi-Fi.
Rainwater Sensor	Detects rainwater and triggers the retraction of the clothes drying rack.
DC Motor	Drives the extension and retraction motion of the clothes drying rack.
Relay Module	An electrical switch that controls the power supply to the DC motor based on input from the NodeMCU and the rainwater sensor.
Access Point	Establishes a Wi-Fi network for communication between the NodeMCU and smartphones, enabling remote control.
DC Power Adapter	Converts AC power into DC power for powering components like the NodeMCU and the motor.





*Figure 2.2 Automatic drying tool design*

### **System Operating Principle**

**NodeMCU ESP8266:** The NodeMCU is programmed to connect to a Wi-Fi network and respond to commands from smartphones or other devices. It communicates with a rainwater sensor to determine if it's raining and sends signals to control clothes drying rack system through a relay module and a DC motor.

**Rainwater Sensor:** The rainwater sensor detects raindrops and is activated when rainwater meets it. This results in a change in its output, which is read by the NodeMCU. When rain is detected, the NodeMCU sends commands to retract the drying rack, protecting wet clothes from getting soaked in the rain.

**DC Motor:** The DC motor drives the movement of the drying rack. When the NodeMCU receives a signal from the rainwater sensor indicating rain, it activates the motor through the relay module, causing the drying rack to retract and safeguarding the clothes from getting wet in the rain.

**Relay Module:** The relay module acts as a switch controlled by the NodeMCU. When the NodeMCU receives a signal from the rainwater sensor or a smartphone application, it activates the relay, allowing current to flow and power the DC motor, thereby controlling the movement of the drying rack.

**Access Point:** The access point creates a local Wi-Fi network, enabling communication between the NodeMCU and a smartphone. This allows users to remotely control the drying rack system through a smartphone application.

**DC Power Adapter:** The DC adapter converts the AC power supplied into DC power suitable for powering components like the NodeMCU and motor, ensuring the proper functioning of the system.

In summary, the automated clothes drying rack system utilizes the NodeMCU ESP8266 as the main controller, communicating with various components including the rainwater sensor, DC motor, and relay module. When rain is detected, the system retracts the drying rack using the motor to protect wet clothes. Users can interact remotely with the system through a smartphone application connected to the NodeMCU's access point.

### **Method to Move Cloth**

In this paper, a DC motor is used as the driver of the clothesline, which can rotate forward or backward according to the instruction of the microcontroller, thus moving the clothesline indoors or outdoors. The microcontroller controls the switch and direction of the DC motor through a relay module. When the microcontroller judges that it is raining, it will activate the relay module, reverse the DC motor, and take the clothesline back indoors. When the microcontroller judges that it is not raining, it will activate the relay module and make the DC motor rotate forward to push the clothesline out of the room.

### 2.1.3 Design and Experimental Study on Automatic Cloth Retrieval and Drying System

#### Hardware Components [3]

*Table 2.4 Specifications of Hardware*

<b>Components</b>	<b>Functionality</b>
Sheet Metal	Used to provide structural support and external protection.
Electric Motor	Converts electrical energy into mechanical energy.
Solar Panel	Converts solar energy into electricity for powering electronic devices.
Rain Detector	Used to detect the presence or intensity of rain.
Blower	Uses to blow out hot air.
12V DC geared motor	Used to provide precise mechanical motion, often combined with gears to achieve the desired output.
Temperature Sensor	Measures the ambient temperature of the surroundings.
Light Sensor	Measures the ambient light intensity.
Battery	Stores electrical energy for use when needed.
Control Unit (PIC 16F877)	Used to control and coordinate other components.

### **Method to Move Cloth**

This research mentions a DC motor, which can control the movement of clothes through a pulley and a rope. When the system judges that the weather is suitable for drying clothes, the motor will pull out the rope and extend the clothes out of the window; When the system judges that the weather is not suitable for drying clothes, the motor will take back the rope and the clothes in the window.

### **Factors for Material Selection**

The research mentions that there are a total of seven factors for material selection, which are physical properties, mechanical performance, manufacturability, manufacturing cost, quality requirements, material availability, and space considerations.

### **System Operating Principle**

The system is composed of a solar panel, a battery, a motor, and a control unit. The solar panel harnesses solar energy and charges the battery. The battery stores this energy and uses it to power the motor. The motor, in turn, operates a pulley and rope mechanism that controls the movement of the clothes rack. Users can hang their wet laundry on the rack and set a desired drying time using a rotary switch. Control unit is the brain of the system. It receives signals from various sensors (temperature, light-sensitive, and rain sensors) and determines the current weather conditions. This data is then displayed on an LCD screen. If the weather is sunny and warm, the control unit signals the motor to extend the clothes rack outside for drying. A timer for the drying process is also initiated. In case of unfavourable weather conditions like rain or cloudiness, the control unit instructs the motor to retract the clothes rack indoors. The drying timer is paused during this period. Once the set drying time is reached, the control unit prompts the motor to retract the clothes rack indoors if it isn't already. Users are then notified to collect their now dry laundry.

**Literature Review in This Research***Table 2.5 Literature Review*

Outdoor Scalable Clothes Drying Rack	This system employs German gas springs and Japanese rolling bearings to enable the drying rack to be stored near the ceiling when not in use. It can be lowered and extended for use, making it suitable for homes with limited space.
Rain Gauge RG-10	This device utilizes infrared beams to detect rainfall. It offers different operational modes and signal outputs through DIP switches. It is suitable for applications requiring dependable and sensitive rainwater detection, such as automatic awning retraction and ship/vehicle wiper control.
Temperature Control System	This system is designed for server rooms and comprises temperature sensors, a PIC microcontroller, an LCD, a drive circuit, an AC air heater, and an AC motor. It automatically regulates the air heater and motor based on room temperature to maintain an optimal temperature inside the server room.
Roller Blind Awning	This awning is constructed using heavy-duty American components with a robust aluminium alloy frame and weather-resistant coating. An electric motor allows for easy rolling up or down, making it suitable for locations requiring sun protection or protection from wind and rain.
Balance of Hanging Clothesline	This method applies principles from engineering mechanics, specifically the concept of a balance force system, to analyse the equilibrium of hanging clotheslines. The hanging clothesline is an example of a parallel force system where all forces acting on the steel pipe are vertical. Design considerations include the mass of the hanging object, whether it's wet or dry, and the position of the clothesline.
Research on Drying Laundry	This section presents research on laundry drying, covering the rationale and methods for indoor and outdoor drying. It also explores factors influencing the drying process, such as moisture evaporation, humidity changes, and energy conversion.

### 2.1.4 Automatic Cloth Drying Line Solution Using IoT [4]

#### Hardware Components

*Table 2.6 Specifications of Hardware*

<b>Components</b>	<b>Functionality</b>
Arduino UNO	A micro-controller board used to control and coordinate other components.
Rain Sensor Module	Used to detect rainfall and is a kind of switching device.
L293D Motor Driver	A dual-channel H-Bridge motor driver and used to control the direction of rotation of a bi-directional DC motor.
ESP8266 wi-fi module	A cost-effective Wi-Fi microchips.
Jumper Wires	Wires with connector pins at both their ends, which allow for setting up connections without soldering.
Breadboard	A simple board which helps in solderless circuit.
12V DC geared motor	Used to move heavy objects.
Drawer Slider	A device that allows for horizontal motion or the sliding motion.

### **Method to Move Cloth**

This system does not need to move clothes, but the roof. It uses drawer slide rails to realize the translation movement of the roof.

The movable roof is a transparent roof made of fibreboard, which can slide on the clothes rack to cover or expose clothes according to weather conditions. The sliding movement of the moving roof is realized by a 12V DC deceleration motor, and a pulley is installed on the motor. When the motor rotates, cotton thread will wind or loosen the pulley, thus driving the roof to slide back and forth along the slide rail. Drawer slide rails are devices that can move horizontally, and they are fixed at both ends of the clothes rack.

### **Literature in This Research**

The article reviews the types and characteristics of existing clothes drying rack systems, including traditional T-Poles, retractable Versa lines, and automatic systems that use fixed roofs and sensors to retract clothes.

The article points out the shortcomings of existing systems, such as space occupation, high cost, poor drying due to clothes contact, etc., and proposes the innovations of the SMART-DRY system, such as using a moving roof instead of moving clothes, using drawer rails instead of complex folding mechanisms, using T-Poles as the basic structure, etc.

### **System Operating Principle**

The system uses an Arduino UNO as a microcontroller and a raindrop sensor as the main functional unit. When the raindrop sensor detects rain, it sends a signal to the Arduino UNO, triggering a motor driver to start a 12V DC motor, moving the mobile roof to cover the clothes and protect them from the rain. When the rain stops, the raindrop sensor notifies the microcontroller via a signal, and the microcontroller triggers the motor driver again, causing the motor to start and move the mobile roof in the opposite direction, exposing the clothes to the air again. The system uses an ESP8266 wi-fi module for wireless communication, synchronizing the status of the system in real time to an application that users can use via the internet.

### 2.1.5 Summary of Previous Works

*Table 2.7 Specifications of Control Unit and Wi-Fi Module*

	<b>Control Unit</b>	<b>Wi-Fi Module</b>
<b>1</b>	ESP32	Built-in
<b>2</b>	NodeMCU	ESP8266
<b>3</b>	PIC 16F877	No
<b>4</b>	Arduino UNO	ESP8266

The above table are the control units used by four different systems, some of which have a built-in Wi-Fi module, while others require external connections. The third system does not intend to use Wi-Fi functionality. The control unit's primary function here is to establish connections and control various components. In addition, some of them have a Wi-Fi module that allows users to remotely control the entire system.

*Table 2.8 Specifications of Sensor*

	<b>Temperature</b>	<b>Humidity</b>	<b>Light</b>	<b>Rain</b>
<b>1</b>	DHT22	DHT22	LDR	MD0127
<b>2</b>	No	No	No	Yes
<b>3</b>	Yes	No	Yes	Yes
<b>4</b>	No	No	No	Yes

The above table are the sensors used by four different systems, which include Temperature, Humidity, Light, and Rain sensors. The first system uses the DHT22, which is a sensor that combines Temperature and Humidity measurements. The second and fourth systems only use Rain Sensors. When using only the rain sensor, it means that the system is triggered only when the rain sensor detects rain. Since this is a prototype, the rain sensor has a relatively small sensing area. In practical applications, to ensure that the rain sensor can detect rain as quickly as possible, the sensing area of the rain sensor needs to be increased.

Moreover, when the rain sensor detects rain, it often implies that clothes have also been exposed to some rainwater. This makes this solution more of a contingency plan for drying clothes after they have been rained on, rather than a system that can replace a



person to successfully dry freshly washed clothes. In comparison, the first and third systems are equipped with sensors such as temperature, humidity, and brightness sensors. These sensors allow the system to assess the weather conditions. If the weather assessment is accurate enough, it can predict rain with a high probability before it starts raining. In such cases, it can proactively bring the clothes indoors and move them outside to dry when the temperature and brightness are suitable.

However, weather conditions can be quite variable, and it may be advisable to consider adding more sensors to gather additional data for weather prediction, such as a wind direction sensor and anemometer.

In summary, there are three approaches to moving clothes, all of which involve the use of DC motors, pulleys, and ropes. Approach 4, however, takes a completely different approach by not moving the clothes but rather the roof itself. It utilizes a roof, a 12V DC reduction motor, pulleys, and a sliding rail system. In the former solutions, the ropes need to be divided into two segments: one in the exposed area and the other in the sheltered area. In practice, they can only dry a limited amount of clothes at a time.

The latter approach is relatively expensive and challenging to implement, making it less accessible. However, these two approaches are currently considered relatively suitable solutions. In the future, this study will select the appropriate implementation method based on specific customer requirements. Currently, the plan is to use the DC motor, pulleys, and ropes approach.

## 2.2 Weather API Selection

### 2.2.1

#### AccuWeather APIs [5]

Within the suite of APIs provided by AccuWeather, the Weather Alarms API is particularly well-suited for determining whether it might rain in a specific area to retrieve clothes left outdoors for drying. By utilizing AccuWeather's Weather Alarms API, valuable weather alarm information for a particular location can be obtained to ascertain the presence of rainfall or any other weather anomalies. This information holds significant value for making informed decisions regarding clothing management.

### Weather Alarms API

Get Weather Alarms for a specific location. AccuWeather Weather Alarms are determined using the daily forecasts for a location. An alarm exists for a location if the forecast weather meets or exceeds a specific threshold.

GET	<b>1 Day of Weather Alarms</b> http://dataservice.accuweather.com/alerts/v1/1day/{locationKey}	Returns 1 day of weather alarms for a specific location.
GET	<b>10 Days of Weather Alarms</b> http://dataservice.accuweather.com/alerts/v1/10day/{locationKey}	Returns 10 days of weather alarms for a specific location.
GET	<b>15 Days of Weather Alarms</b> http://dataservice.accuweather.com/alerts/v1/15day/{locationKey}	Returns 15 days of weather alarms for a specific location.
GET	<b>5 Days of Weather Alarms</b> http://dataservice.accuweather.com/alerts/v1/5day/{locationKey}	Returns 5 days of weather alarms for a specific location.

**Weather Alarm Thresholds**

*Figure 2.3 Weather Alarms API*

### Weather Alarm Thresholds

AccuWeather weather alarms are determined using the daily forecasts for a location. An alarm exists for a location if the forecast weather meets or exceeds the following thresholds:

Alarm Type	Imperial Threshold	Metric Threshold
Rain	0.5 inch	12.7 mm
Snow	1 inch	2.54 cm
Ice	0.1 inch	0.254 mm
Sustained Wind	30 mph	48 kph
Wind Gust	40 mph	64 kph
Thunderstorm Probability	75%	75%

*Figure 2.4 Weather Alarm Thresholds*

## Packages

To utilize the Weather Alarms API, a minimum requirement of the Standard package from AccuWeather is necessary, which is priced at \$25 per month. However, it's important to note that this may have a relatively low cost-effectiveness in achieving the specific objectives of this study.

### CORE WEATHER

Features	Free	Standard	Prime	Elite
Locations	✓	✓	✓	✓
Current Conditions	✓	✓	✓	✓
24 Hours Historical Current Conditions	✓	✓	✓	✓
Daily Forecast	5 Days	5 Days	10 Days	15 Days
Hourly Forecast	12 Hours	12 Hours	72 Hours	120 Hours
Indices	5 Days	5 Days	10 Days	15 Days
Alarms		5 Days	10 Days	15 Days
Translations		✓	✓	✓
Tropical			✓	✓
Alerts			✓	✓
Imagery			✓	✓
	<b>Free</b> 50 calls/day Limit 1 key/developer Free: <a href="#">Get started now!</a>	<b>\$25<sub>mo</sub></b> \$0.12 CPM over 225,000 calls per month <input type="checkbox"/>	<b>\$250<sub>mo</sub></b> \$0.15 CPM over 1,800,000 calls per month <input type="checkbox"/>	<b>\$500<sub>mo</sub></b> \$0.22 CPM over 2,400,000 calls per month <input type="checkbox"/>

Figure 2.5 Packages and Pricing

## Free Package API

In addition to the Standard package, AccuWeather's free package includes several APIs that are suitable for this study, such as the Daily and Hourly Forecasts APIs. The Daily Forecast API provides users with the weather forecast for the current day, aiding in decisions about whether to dry clothes on that day. On the other hand, the Hourly Forecasts API offers more detailed and up-to-date weather forecasts, facilitating decisions regarding when to halt clothes drying activities. The Current Conditions API allows retrieval of real-time weather data for a specific location, offering valuable information about the current weather conditions. Furthermore, the Location API, with its location key, can provide information about neighbouring cities or specific locations, enhancing specificity and precision at the regional level. This can greatly assist users or models in making informed decisions.

Daily		DESCRIPTION
METHOD		
GET	<b>1 Day of Daily Forecasts</b> <a href="http://dataservice.accuweather.com/forecasts/v1/daily/1day/{locationKey}">http://dataservice.accuweather.com/forecasts/v1/daily/1day/{locationKey}</a>	Returns daily forecast data for a specific location. Forecast searches require a location key. Please use the Locations API to obtain the location key for your desired location. By default, a truncated version of the hourly forecast data is returned. The full object can be obtained by passing "details=true" into the url string.
GET	<b>10 Days of Daily Forecasts</b> <a href="http://dataservice.accuweather.com/forecasts/v1/daily/10day/{locationKey}">http://dataservice.accuweather.com/forecasts/v1/daily/10day/{locationKey}</a>	Returns an array of daily forecasts for the next 10 days for a specific location. Forecast searches require a location key. Please use the Locations API to obtain the location key for your desired location. By default, a truncated version of the hourly forecast data is returned. The full object can be obtained by passing "details=true" into the url string.
GET	<b>15 Days of Daily Forecasts</b> <a href="http://dataservice.accuweather.com/forecasts/v1/daily/15day/{locationKey}">http://dataservice.accuweather.com/forecasts/v1/daily/15day/{locationKey}</a>	Returns an array of daily forecasts for the next 15 days for a specific location. Forecast searches require a location key. Please use the Locations API to obtain the location key for your desired location. By default, a truncated version of the hourly forecast data is returned. The full object can be obtained by passing "details=true" into the url string.
GET	<b>5 Days of Daily Forecasts</b> <a href="http://dataservice.accuweather.com/forecasts/v1/daily/5day/{locationKey}">http://dataservice.accuweather.com/forecasts/v1/daily/5day/{locationKey}</a>	Returns an array of daily forecasts for the next 5 days for a specific location. Forecast searches require a location key. Please use the Locations API to obtain the location key for your desired location. By default, a truncated version of the hourly forecast data is returned. The full object can be obtained by passing "details=true" into the url string.

*Figure 2.6 Daily Forecast*

## Current Conditions

METHOD		DESCRIPTION
GET	<b>Current Conditions</b> <a href="http://dataservice.accuweather.com/currentconditions/v1/{locationKey}">http://dataservice.accuweather.com/currentconditions/v1/{locationKey}</a>	Returns current conditions data for a specific location. Current conditions searches require a location key. Please use the Locations API to obtain the location key for your desired location. By default, a truncated version of the current conditions data is returned. The full object can be obtained by passing "details=true" into the url string.

*Figure 2.7 Current Conditions*

## Hourly

METHOD		DESCRIPTION
GET	<b>1 Hour of Hourly Forecasts</b> <a href="http://dataservice.accuweather.com/forecasts/v1/hourly/1hour/{locationKey}">http://dataservice.accuweather.com/forecasts/v1/hourly/1hour/{locationKey}</a>	Returns forecast data for the next hour for a specific location. Forecast searches require a location key. Please use the Locations API to obtain the location key for your desired location. By default, a truncated version of the hourly forecast data is returned. The full object can be obtained by passing "details=true" into the url string.

*Figure 2.8 Hourly Forecast*

## Location Key

METHOD		DESCRIPTION
GET	<b>City Neighbors by locationKey</b> <a href="http://dataservice.accuweather.com/locations/v1/cities/neighbors/{locationKey}">http://dataservice.accuweather.com/locations/v1/cities/neighbors/{locationKey}</a>	Returns information about neighboring cities, by location key. You must know the location key to perform this query.
GET	<b>Search by locationKey</b> <a href="http://dataservice.accuweather.com/locations/v1/{locationKey}">http://dataservice.accuweather.com/locations/v1/{locationKey}</a>	Returns information about a specific location, by location key. You must know the location key to perform this query.

*Figure 2.9 Location Key*

## Countries

In the "Countries" section of the AccuWeather API, it can be observed that it supports most countries, including Malaysia.

Malaysia	MY	Asia	ASI
----------	----	------	-----

*Figure 2.10 Countries*

## 2.2.2

### Open Weather API [6]

A variety of data can be accessed from the Open Weather API, including current weather conditions and forecasts, ranging from minute-by-minute forecasts for the next hour to daily forecasts covering 8 days, along with government weather alerts. Additionally, the API offers the capability to retrieve weather data for any specific timestamp from a historical archive spanning over 40 years, as well as obtain a 4-day ahead forecast. Moreover, it provides daily aggregated weather data from a 40+ year archive and extends its forecasts 1.5 years into the future.

### "One Call by Call" subscription plan

One Call API 3.0 is included to the "One call by call" subscription plan only, users pay for the actual use of the product. There are no limits on the number of API calls, but you can set API calls limit per day in the "Billing plans" tab in your Personal account.

Please note, that you do not need to subscribe to any other OpenWeather subscription plans to get access to the One Call API 3.0. Read more about this subscription plan in the [FAQ](#).

#### One Call API 3.0

Make an API call to receive access to the various data:

- **Current weather and forecasts:**
  - minute forecast for 1 hour
  - hourly forecast for 48 hours
  - daily forecast for 8 days
- **Weather data for any timestamp** for 40+ years historical archive and 4 days ahead forecast
- **Daily aggregation** of weather data for 40+ years archive and 1.5 years ahead forecast

Pay as you call

**1,000 API calls per day for free**  
**0.0012 GBP** per API call over the daily limit

Subscribe

*Figure 2.11 One Call API*

### Global Weather Alerts Push Notifications

Like AccuWeather, the Open Weather API also provides the "Global Weather Alerts Push Notifications" feature, which aligns with the requirements of this study. However, it comes with a price tag of 95 GBP (approximately RM554.80). Consideration for this feature may be warranted in the context of future commercialization and expansion to additional regions in software development. Currently, it may not align with the project's immediate requirements.

## Special products

<p><b>Global Weather Alerts Push notifications</b></p> <p>based on warnings from government meteorological agencies around the globe</p>	<p><b>95 GBP/month</b></p> <p><a href="#">Get access</a></p>
<p><b>Road Risk API (advanced configuration)</b></p> <p>This tool is specifically designed for logistics and delivery businesses to enable effective planning of the routes' schedule and fuel consumption. It is flexibly configured to provide detailed weather information along the route with minute granularity, alerts, push notifications and forecasts for the destination point.</p>	<p>By request</p> <p><a href="#">Get access</a></p>
<p><b>Global Precipitation Map - Forecast and historical data</b></p> <p>Current, forecast and historical precipitation maps for the globe with 10-minutes data update</p>	<p>By request</p> <p><a href="#">Get access</a></p>
<p><b>Weather Maps 2.0 with 1-hour step</b></p> <p>Forecast, historical and current weather maps for 14 different weather layers</p>	<p>By request</p> <p><a href="#">Get access</a></p>

*Figure 2.12 Special products*

### Global Weather Alerts Push notifications

[Doc](#) [Get access](#)

- Get all the **warnings from national weather agencies**
- Weather alerts are pushed to your endpoint as soon as they occur
- Data feed provides all active weather alerts from the entire world
- Each alert contents date, time, location, and detailed description
- **Monthly subscription.** Please **contact us** to get access.

*Figure 2.13 Global Weather Alerts Push notifications*

### Other Suitable Service

The Open Weather API offers additional services suitable for this study. For instance, the "Current Weather Data" service provides real-time weather data for any location. The "Hourly Forecast 4 days" service offers a continuous 4-day hourly forecast but necessitates the "Startup Package," priced at approximately 30 GBP per month (approximately RM175.20). The "Daily Forecast 16 days" service provides a continuous 16-day daily forecast and requires the "Developer Package," priced at approximately 140 GBP per month (approximately 817.60 MYR). Overall, the pricing of suitable services offered by the Open Weather API tends to be on the higher side. Therefore, it is advisable to consider them with a lower priority in the project planning of this study.

### Current & Forecast weather data collection

#### Current Weather Data

[API doc](#) [Subscribe](#)

- Access current weather data for any location
- We collect and process weather data from different sources such as global and local weather models, satellites, radars and a vast network of weather stations
- JSON, XML, and HTML formats
- Included in both free and paid subscriptions

#### Hourly Forecast 4 days

[API doc](#) [Subscribe](#)

- Hourly forecast is available for 4 days
- Forecast weather data for 96 timestamps
- JSON and XML formats
- Included in the Developer, Professional and Enterprise subscription plans

#### Daily Forecast 16 days

[API doc](#) [Subscribe](#)

- 16 days forecast is available for any location on the globe
- 1-day step for 16 days
- JSON and XML formats
- Included in all paid subscription plans

*Figure 2.14 Current & Forecast weather data collection*



### 2.2.3 MET API [7]

The Malaysian Meteorological Department Web Service API, commonly referred to as MET API, is a complimentary service provided by the Malaysian Meteorological Department to the public. This service operates over the HTTP protocol and offers endpoints that deliver responses in JSON format.

The API Token has been successfully obtained through registration with the Malaysian Meteorological Department Web Service API.

#### Your API Token



*Figure 2.15 API Token*

The following is the throttling of the website.

#### Throttling

Every authenticated requests are subjected to throttling.

Rate	Description
Burst rate	Maximum burst requests is 10 per minute.
Sustained rate	Maximum sustained requests is 2,000 per day.

*Figure 2.16 Throttling*

There are three types of information that can be obtained from the website: General (weather) Forecast, Marine Forecast, and Warnings. The relationship between Marine Forecast and this study is not significant, so the following will provide detailed explanations for the other two.

## Available Data

### Data Available

There are 3 types of data available:

- General (weather) Forecast
- Marine Forecast
- Warnings

By default, upon registration you only have access to general forecast, marine forecast & warnings.

*Figure 2.17 Available Data*

In contrast to the other two APIs, it's important to note that within the Malaysian Meteorological Department Web Service API, the term "warning" does not equate to their "Alarm" feature. This warning system is primarily utilized to issue alerts for extreme weather conditions or natural disasters and is unrelated to this study.

### Warnings

There five (5) types of warnings available namely earthquake/tsunami; strong wind & rough seas; thunderstorm; continuous heavy rain; and tropical cyclone. In order to query warnings, you need to request from endpoint: `/data`

Four (4) query parameters are required:

Query Parameter	Value
<code>datasetid</code>	WARNING
<code>datacategoryid</code>	QUAKETSUNAMI (earthquake/tsunami) WINDSEA (strong wind & rough sea) THUNDERSTORM (thunderstorm) RAIN (continuous heavy rain) CYCLONE (tropical cyclone)
<code>start_date</code>	Date in ISO8601 format (YYYY-MM-DD) e.g. 2023-09-11. The value must be today's date or greater.
<code>end_date</code>	Date in ISO8601 format (YYYY-MM-DD) e.g. 2023-09-11. The value must be equal or greater than <code>start_date</code>

*Figure 2.18 Warnings*

Like the previous two APIs, this API also provides access to General Forecast information.

## General Forecast

In order to query weather forecast data, you need to request from endpoint. Most of the time you would want a location that falls under the category TOWN or TOURISTDEST (tourist destination). See [Locations](#) for further information. `/data`

Five (5) query parameters are required. There is one (1) optional parameter to select language:

Query Parameter	Value
<code>datasetid</code>	FORECAST
<code>datacategoryid</code>	GENERAL
<code>locationid</code>	See <a href="#">Locations</a> . In order to fetch data for all locations put the value <code>ALL_LOCATIONS</code> .
<code>start_date</code>	Date in ISO8601 format (YYYY-MM-DD) e.g. 2023-09-11. The value must be today's date or greater.
<code>end_date</code>	Date in ISO8601 format (YYYY-MM-DD) e.g. 2023-09-11. The value must be equal or greater than <code>start_date</code> .
<code>lang</code>	<code>en</code> for English or for Bahasa Melayu. If this parameter is not specified, the default is English. <code>ms</code>

*Figure 2.19 General Forecast*

The General Forecast is divided into five location categories, namely STATE, DISTRICT, TOWN, TOURISTDEST, and WATERS. Among these, the first three (STATE, DISTRICT, and TOWN) will be particularly beneficial for this study.

## Locations

There are 5 location categories:

- STATE (for general forecast)
- DISTRICT (for general forecast)
- TOWN (for general forecast)
- TOURISTDEST (for general forecast)
- WATERS (for marine forecast)

*Figure 2.20 Locations*

## 2.3 Arduino Hardware Selection

### 2.3.1

#### SparkFun Arduino IoT Weather Station [8][9]

This suite of components is exceptionally comprehensive and suitable for building a weather station within the scope of this study. The package includes the fundamental ESP32 processor module, along with various sensors, such as the BME280, which integrates temperature, humidity, and pressure measurements into a single sensor. Furthermore, it features the AS3935 lightning detector, capable of detecting lightning storms within a radius of up to 40 km and issuing alerts. It can even estimate the storm's leading-edge distance as close as 1 km with 14 step values. The AS3935 can also detect both cloud-to-ground and cloud-to-cloud discharges.

While the VEML6075 ultraviolet (UV) sensor is not included in the package by default, it can be optionally added to measure UV index and UV intensity. The suite also includes components for measuring wind direction and wind speed, a rain gauge, and related accessories. Additionally, it incorporates a soil moisture sensor and various other accessories to create a comprehensive weather monitoring system.

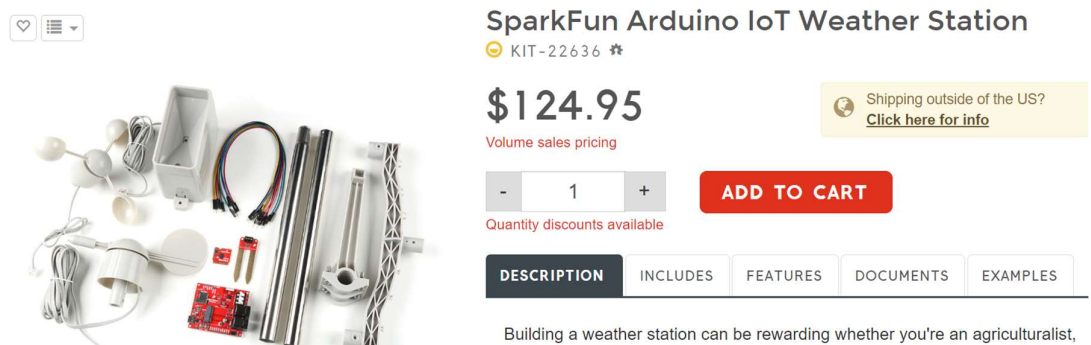
Arduino [IoT](#) Weather Station kit content:

- MicroMod ESP32 Processor module
- MicroMod Weather Carrier Board with a BME280 temperature, pressure, and humidity sensor and an AS3935 lightning detector only. The optional VEML6075 UV sensor is NOT included.
- The Weather Meter Kit with
  - Wind vane
  - Cup anemometer
  - Wind meter mounting bar
  - Tipping bucket rain gauge and mounting arm
  - Two-Part mounting Mast
  - 2x mounting clamps
  - 4x zip ties
- SparkFun Soil Moisture Sensor with screw terminals (Note: variable resistance soil moisture sensor)
- A pack of ten "premium" jumper wires

*Figure 2.21 Arduino IoT Weather Station kit content*

## Pricing

The kit is available for purchase on the Sparkfun website at a price of \$124.95 (approximately RM595). Overall, this kit offers a high level of comprehensiveness and provides a wide range of parameters for weather observation. It represents a hardware bundle that should be considered for acquisition and utilization in this study.



*Figure 2.22 SparkFun Arduino IoT Weather Station*

## CHAPTER 3

### SYSTEM METHODOLOGY/APPROACH

#### 3.1 System Requirement

##### 3.1.1 Hardware

*Table 3.1 Specifications of laptop*

<b>Description</b>	<b>Specifications</b>
Model	Victus 16 by HP Laptop
Processor	Intel Core i5-12500H
Operating System	Windows 10
Graphic	NVIDIA GeForce RTX 3050 4GB DDR5
Memory	24GB DDR5 RAM
Storage	500GB SATA SSD

*Table 3.2 Specifications of Arduino and Sensor*

<b>Description</b>	<b>Specifications</b>
Arduino Model	Arduino UNO R4 Wi-Fi
UV Intensity Sensor	GYML8511
Temperature, Humidity Sensor	DHT22
Light Intensity Sensor	BH1750
Raindrop Sensor	FC-37 or YL-83
Motor Driver	L298N



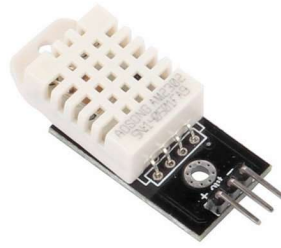
*Figure 3.1 Arduino UNO R4 Wi-Fi*

Figure 3.1 displays an Arduino UNO R4 WI-FI which, compared to the standard Arduino UNO R4, offers several advantages, the most significant being the integrated WI-FI and Bluetooth modules.



*Figure 3.2 BH1750 Light Intensity Sensor*

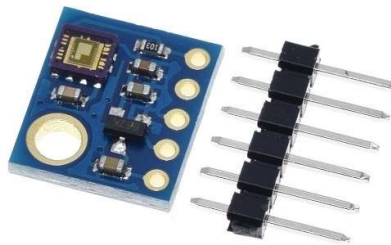
Figure 3.2 shows a BH1750, a Light Intensity Sensor, capable of outputting light intensity data in lux (lx). Connect the VCC pin of the sensor to the 5V pin on the Arduino, the GND pin to the GND on the Arduino, the SCL pin to A5, and the SDA pin to A4.



*Figure 3.3 DHT22 Temperature and Humidity Sensor*

Figure 3.3 depicts a DHT22, a temperature and humidity sensor. It provides humidity readings in percentage form and temperature readings in degrees Celsius.

Connect the "+" pin of the sensor to the 5V pin on the Arduino, the "OUT" pin to pin 7, and the "-" pin to the GND on the Arduino.



*Figure 3.4 GYML8511 UV Intensity Sensor*

Figure 3.4 presents a GYML8511, a UV intensity sensor, capable of outputting ultraviolet intensity data in milliwatts per square centimetre ( $\text{mW}/\text{cm}^2$ ). Connect the "VIN" pin of the sensor to the "VIN" pin on the Arduino, the "3V3" pin to the "3.3V" pin on the Arduino, the "GND" pin to the "GND" pin on the Arduino, and the "OUT" pin to the "A0" pin on the Arduino. For the "EN" pin, connect it to a breadboard row that is also connected to the "3V3" pin of the Arduino. In the same row on the breadboard, connect another wire leading to the "A1" pin on the Arduino.





*Figure 3.5 Rain Drop Sensor FC-37 or YL-83*

Figure 3.5 showcases the FC-37 or YL-83 rain drop sensor, a device capable of detecting water beyond the scope of traditional humidity sensors. This sensor comprises two key components: an electronic board and a collector board designed to gather water droplets. To integrate the sensor with an Arduino, connect the VCC pin to the Arduino's 5V, the GND pin to Arduino's ground, and the A0 (analog output) pin to an analog pin on the Arduino, typically A0. In this project, due to A0 being occupied by the UV sensor, A3 is used instead.



*Figure 3.6 L298N Motor Driver, Motor and Battery Holder*

From left to right, Figure 3.6 sequentially displays an L298N Motor Driver, used to control the operation and speed of a motor, the motor itself, and a battery holder. Connect the "+" and "-" wires of the first motor to OUT1 and OUT2, and the "+" and "-" wires of the second motor to OUT3 and OUT4. Connect the red wire of the battery holder to 9V, the black wire to GND. Finally, connect ENA of the Motor Driver to pin 0 on the Arduino, IN1 to pin 1, IN2 to pin 2, IN3 to pin 9, IN4 to pin 10, and ENB to pin 11.

### 3.1.2 Software

1. Arduino IDE 2.3.2
2. Blynk IoT Cloud

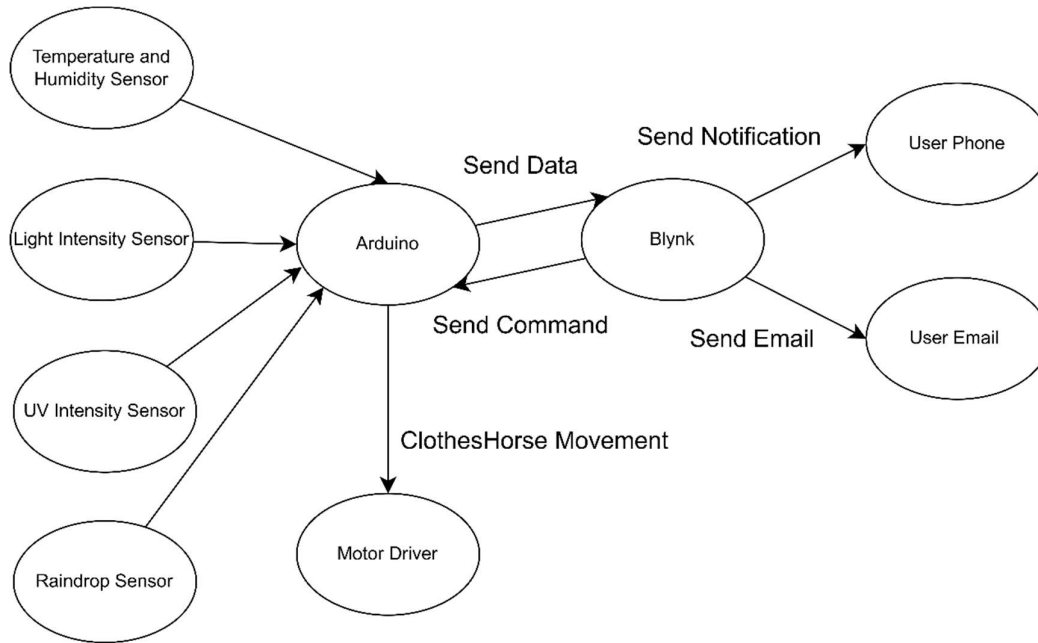
Develop on the Arduino Uno R4 Wi-Fi board by first downloading the Arduino IDE from the official website. After determining which interfaces of the sensors should be connected to which interfaces of the Arduino, connect each sensor to the Arduino one by one. Connect the Arduino to the computer, refer to the relevant sensor documentation, and write code for the Arduino to send the data received from the sensors to the serial monitor, checking if the output is normal. Test each sensor individually to ensure its code can run normally, then merge all sensor codes to allow the Arduino to simultaneously send data from all sensors to the serial monitor.

For the motor driver module, follow a similar process by correctly wiring it, referring to relevant documentation to write code, and testing if it can run normally. If successful, merge the code with the previous code and test again. It is also necessary to test if the Arduino can use Blynk to send virtual data to the Blynk IoT cloud and receive data from the cloud.

After confirming the ability to operate all sensors and the motor driver module using the Arduino IDE, log in to the Blynk website and create a new project, creating corresponding data streams for the sensors and motor driver module, and placing corresponding controls on the dashboard. Finally, test if sensor data can be viewed on the Blynk IoT cloud and if the motor driver module can be controlled via the cloud to simulate the extension and retraction of the clothes drying rack.

## 3.2 System Design Diagram

### 3.2.1 Smart Clotheshorse System Architecture



*Figure 3.7 System Diagram*

Diagram in Figure 3.7 explains how the system operates. First, the temperature and humidity sensor, light intensity sensor, UV intensity sensor and raindrop sensor transmit data to the Arduino. Then, the Arduino sends this data in real time to Blynk so it can be remotely viewed via Blynk's mobile, or web app. Users can manually control the movement of the Motor Driver through Blynk, moving it indoors or outdoors. Additionally, a function is set up in the Arduino to send signals to Blynk based on the sensor data, particularly in situations where there is a high probability of rain. Blynk then notifies the user's phone to prompt a response, automatically moves the Motor Driver, and sends a message through Blynk email function to user's email.

### 3.2.2 Automation Flowchart Diagram

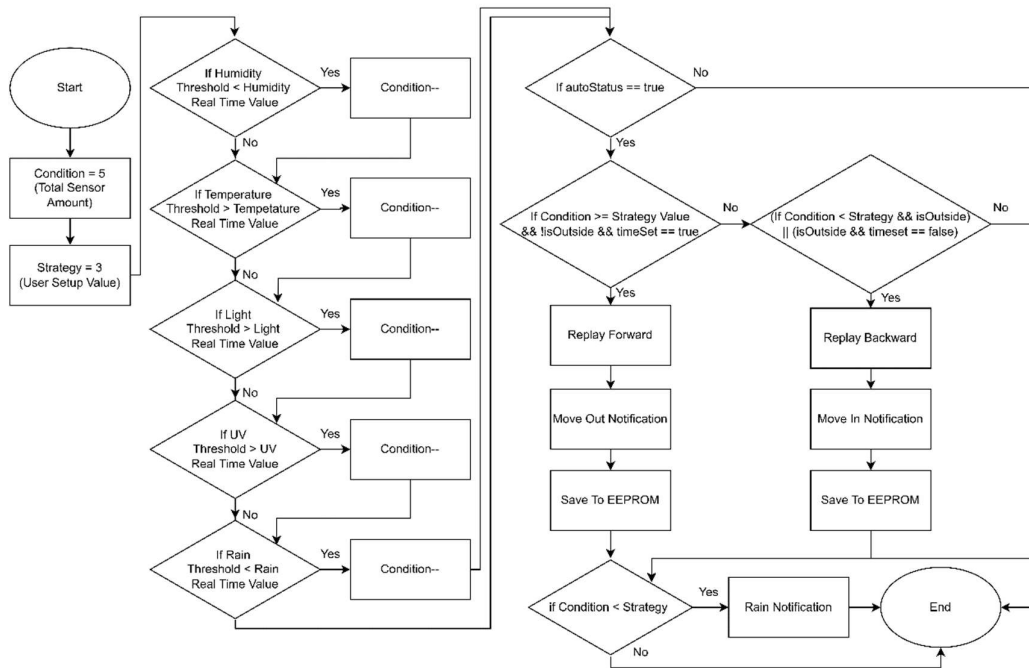


Figure 3.8 Automation Flowchart Diagram

Figure 3.8 illustrates the Automation algorithm for the clothes drying rack system. Initially, the Condition value (total sensor amount) and Strategy (User Setup Value) are established. For instance, with 5 sensors, the Condition value is set to 5, while the user-defined Strategy might be 3, indicating that suitable drying conditions are met when three or more sensors satisfy their criteria.

Sensors are categorized into two types: those directly proportional to favourable drying conditions (such as Temperature, Light Intensity, and UV Intensity) and those inversely proportional (like Humidity and Raindrop detection). The system evaluates each sensor against its threshold. For directly proportional sensors, if the real-time value falls below the threshold, the Condition value decreases by one. Conversely, for inversely proportional sensors, the Condition value decreases if the real-time value exceeds the threshold. For example, if the humidity threshold is 75% and the current humidity is 65%, this sensor meets the condition, and the Condition value remains unchanged.

When the automation feature is active and weather conditions are favourable for drying (as determined by the sensors), the system checks if the drying rack is indoors and if

the current time falls within the designated drying period. If these conditions are met, the Replay Forward function is activated to move the rack outdoors. Simultaneously, a Move Out Notification is triggered, alerting the user via email and mobile notification. The movement path, sensor thresholds, and other relevant data are then saved in the EEPROM.

Conversely, if weather conditions become unfavourable or the current time falls outside the drying period while the rack is outdoors, the system initiates the Replay Backward function, sends a Move in Notification, and saves the updated data to the EEPROM.

Lastly, if weather conditions deteriorate significantly, the system notifies users of potential rainfall, allowing for proactive measures to protect drying clothes.

This comprehensive automation system ensures optimal clothes drying management, adapting to real-time weather conditions and user-defined parameters while keeping users informed through timely notifications.

### 3.2.3 Replay Sequence Diagram

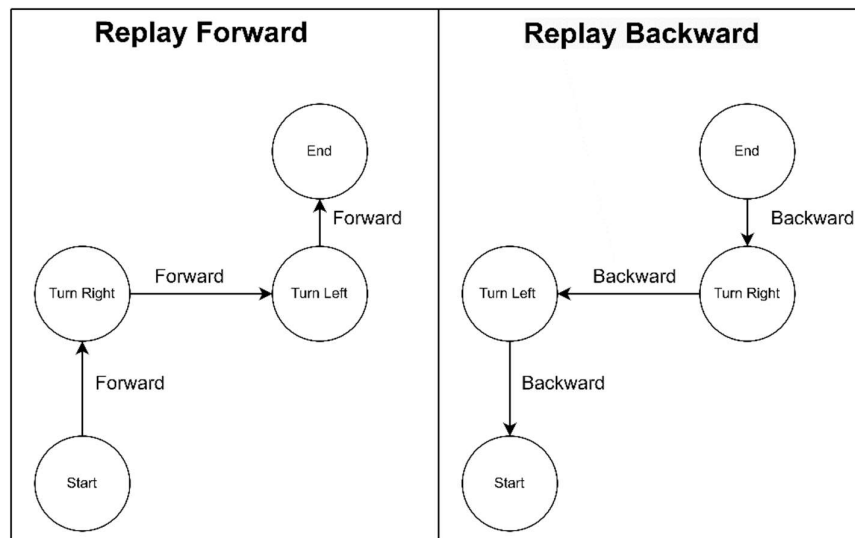


Figure 3.9 Replay Sequence Diagram

Figure 3.9 illustrates examples of Replay Forward and Replay Backward paths. When a user initiates recording by pressing the Recording button in the Blynk App, subsequent joystick operations are logged sequentially. In this instance, the sequence includes forward movement, a right turn, another forward movement, a left turn, and a final forward movement. The recording concludes when the user presses the Recording button again.

Each action and its corresponding duration are queued and recorded in order, allowing for accurate reproduction through the Replay Forward function. Simultaneously, the system records data for Replay Backward functionality. This reverse playback operates by inverting each action: forward movements become backward, right turns become left turns, and so forth. These inverted actions and their durations are stored in a stack structure, enabling the derivation of a reverse path that returns to the starting point.

For example, as shown in the image, the original sequence of forward, turn right, forward, turn left, and forward is transformed into backward, turn right, backward, turn left, and backward for the Replay Backward path. This inversion ensures that the clothes horse can retrace its steps and return to the origin point.

Through these functions, users can customize the movement path of the clothes drying rack. After a one-time setup, the drying rack can be controlled manually or automatically via the Blynk App, allowing for easy manipulation of its movement. If users are dissatisfied with the current path, they have the option to re-record it. The new recording will overwrite the previous path, ensuring that the drying rack's movement always aligns with the user's most recent preferences.

### 3.3 Timeline (Gantt Chart)

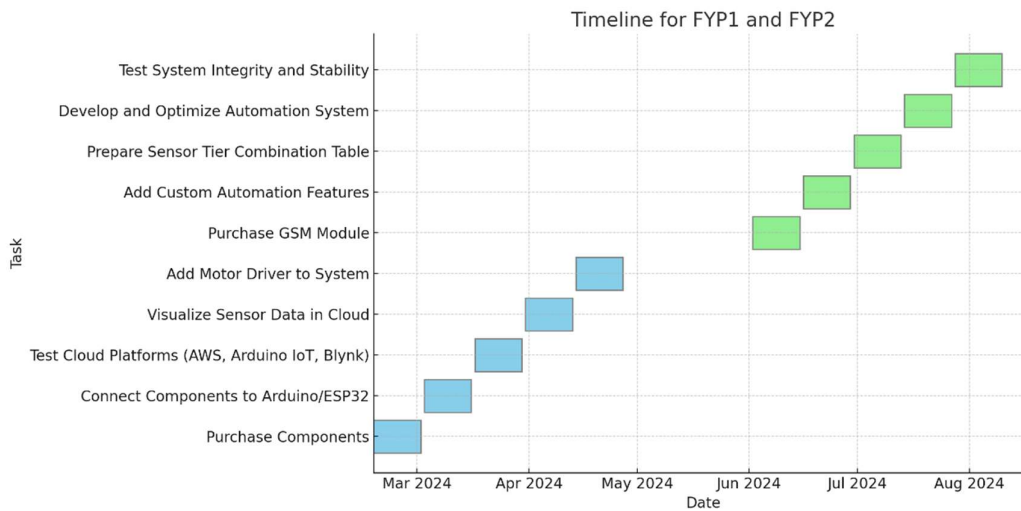


Figure 3.10 Timeline Gantt Chart

Gantt chart in Figure 3.10 illustrates the timeline for FYP1 and FYP2, divided into two major phases. Each phase is further subdivided into five smaller stages.

For FYP1, the first stage involves purchasing the required components, followed by the second stage of connecting these components to the Arduino or ESP32. The third stage entails testing different cloud platforms such as AWS, Arduino IoT, and Blynk. The fourth stage focuses on visualizing sensor data on the selected cloud platform, while the fifth stage involves integrating a motor driver into the current system.

In FYP2, the first stage involves purchasing a GSM module and incorporating it into the existing system. The second stage involves adding custom automation features. The third stage involves preparing sensor tier combination tables for balanced, aggressive, and conservative strategies. The fourth stage focuses on developing and optimizing the automation system based on these tables. Finally, the fifth stage tests the system's functional integrity and stability.

The estimated timeline for FYP1 starts in mid-February, with each subsequent stage occurring every two weeks. For FYP2, the timeline begins in June, following a similar biweekly progression for each stage.

However, during implementation, challenges arose with the GSM module's functionality. As a solution, the project opted to use Blynk's email notification feature instead. This adaptation led to further developments on the Blynk platform, including the addition of a joystick button for manual control of the clothes rack movement. The

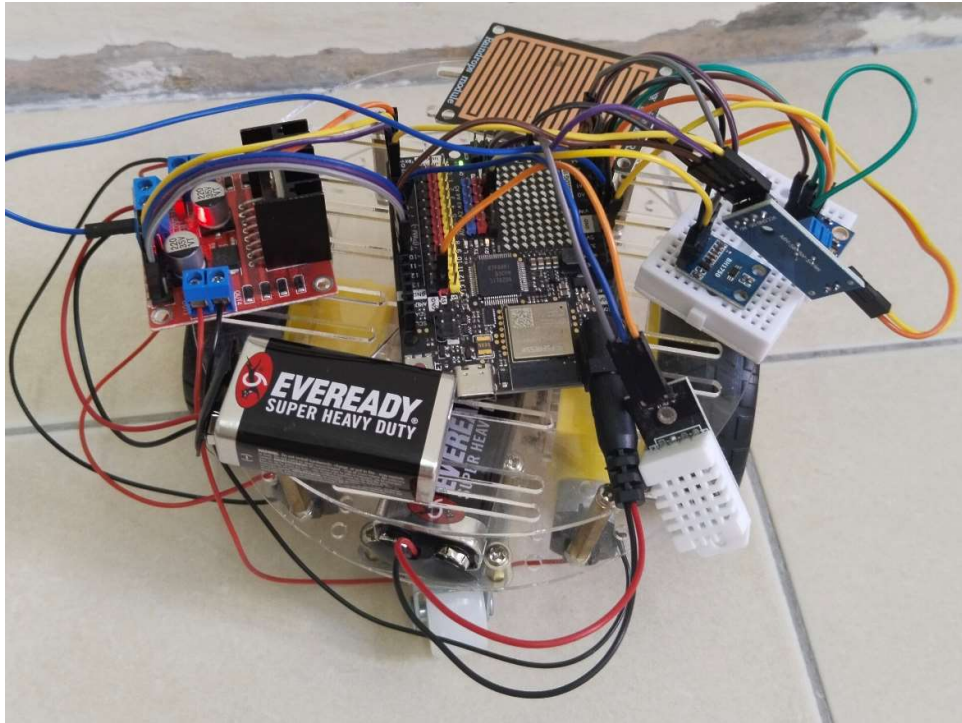
project also implemented features to record movement trajectories, replay them, and derive reverse trajectories to return the rack to its initial position.



## CHAPTER 4

### SYSTEM IMPLEMENTATION

#### 4.1 Arduino Connection



*Figure 4.1 Arduino Connection*

As shown in the Figure 4.1, this is the result of the project. In the upper left corner is the L298N motor driver, which is connected to the Arduino Uno R4 Wi-Fi, a 9V battery, and the motors and wheels below. The Arduino Uno R4 Wi-Fi is connected to all sensors, namely the DHT22 temperature and humidity sensor, the GYML8511 UV intensity sensor, the BH1750 light intensity sensor, and the HL-01 Rain Drop Sensor. It is also powered by a 9V battery, allowing for unrestricted movement without being constrained by data cables. This setup enables the system to operate autonomously and move freely while collecting environmental data.

## 4.2 Blynk Interface



Figure 4.2 Blynk Interface

As shown in Figure 4.2, the Blynk interface displays the device name "Arduino UNO R4" at the top. Below it, the "Automation" button toggles the automatic functionality, useful when users wish to manually adjust the clothes rack position.

To the right of the Automation button, the "Move The Cloth" button indicates the current position of the clothes rack (Inside or Outside). Clicking this button moves the rack in or out. When Automation is on and the algorithm deems the weather suitable for drying clothes, the rack automatically moves outside. If a user manually moves it inside during this time, it will return outside once the movement is complete. The Automation toggle allows users to override this behaviour for convenience.

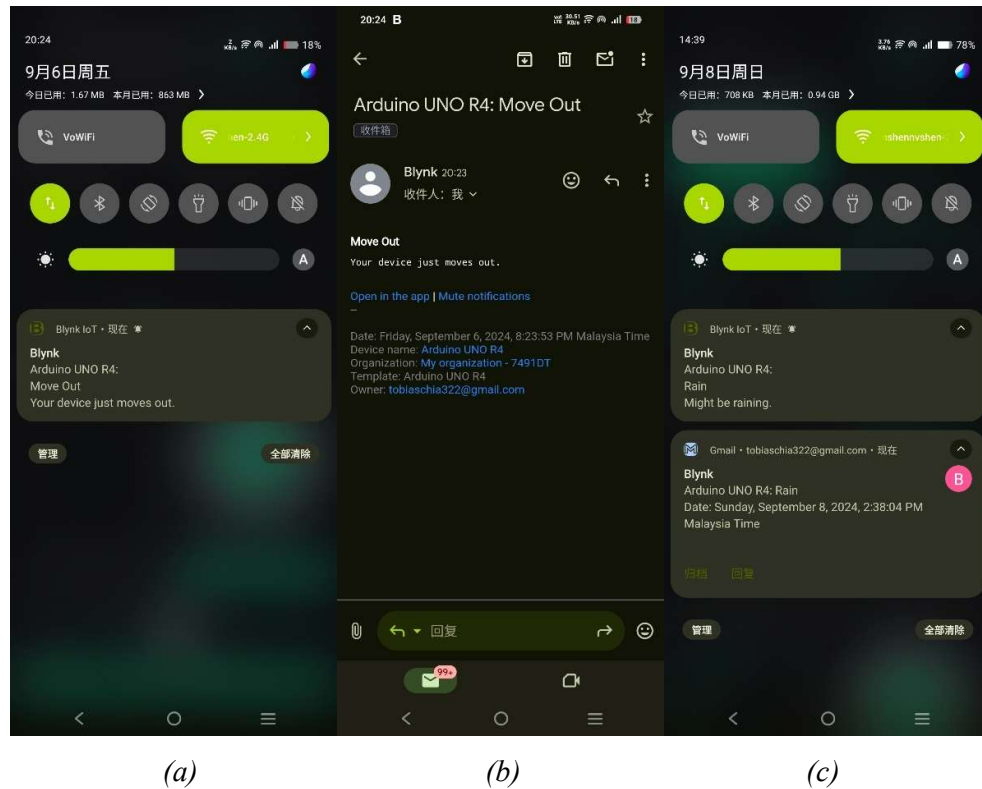
Below these controls are real-time gauges for various sensors: Humidity (0-100%, higher values less suitable for drying), Temperature ( $^{\circ}\text{C}$ , higher values more suitable), Light Intensity (0-65535 lx, higher values more suitable), UV Intensity (0-10  $\text{mW}/\text{cm}^2$ , higher values more suitable), and Rain Value (0-1000, higher values less suitable).

Each sensor has an adjustable input below its gauge. Users can modify these thresholds using +/- buttons or by entering a number directly. The system uses these thresholds to determine suitable drying conditions. For example, if the humidity threshold is set to 75% and the current humidity is 90%, the humidity condition is considered unsuitable for drying.

The joystick button to the left of the Rain Value allows manual control of the clothes rack movement. The Recording button below it, when activated, records movement sequences initiated by the joystick. These sequences can be replayed forward or backward using the respective Replay buttons. The "Move the Cloth" function utilizes these recorded sequences based on the current inside/outside status.

Finally, the Strategy Slider, with five positions corresponding to the number of parameters, determines how many conditions must be met for the system to move the rack. For instance, if set to 3, the rack will move outside (if inside) when three or more parameters indicate suitable drying conditions. Conversely, it will move inside (if outside) when fewer than three parameters are suitable.

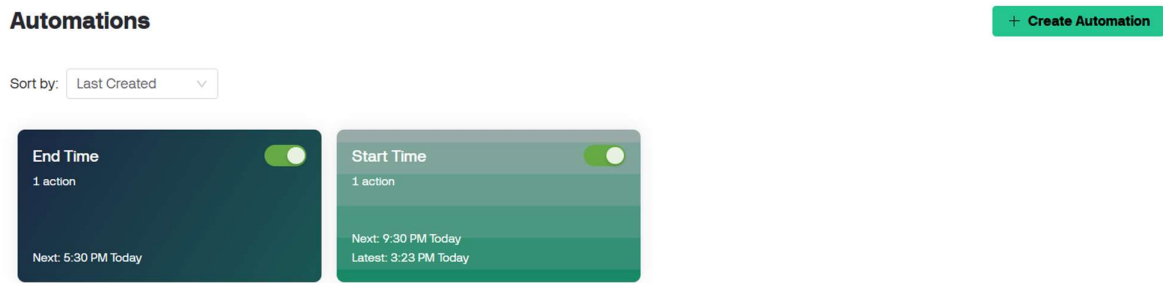
### 4.3 Blynk Notification



*Figure 4.3 Blynk Notification. (a) Move Out Phone Notification (b) Move Out Email Notification (c) Rain Phone and Email Notification*

As shown in the Figure 4.3, whenever the device performs a movement operation, or when real-time data falls below the set strategy values, users receive notifications on both their mobile devices and via email.

## 4.4 Blynk Sunrise and Sunset Automation



*Figure 4.4 Blynk Sunrise and Sunset Automation*

As illustrated in the Figure 4.4, users can set specific time periods for drying clothes throughout the day via the Blynk web interface. For example, a user might set the drying period from 9:30 AM to 5:30 PM. When the current time falls outside this designated range, the system automatically checks the position of the clothes rack. If the rack is detected to be outside, it will automatically retract to the inside position.

## 4.5 Implementation Issues and Challenges

Throughout the development process of this intelligent clothes drying system, numerous challenges were encountered, each systematically addressed and resolved. The initial hurdles involved determining the correct physical connections between various components and the Arduino, as well as configuring appropriate code for each sensor and actuator. Integrating individual component codes into a cohesive whole proved to be a significant challenge.

Establishing a stable connection between the Arduino and the Blynk IoT platform for data communication presented its own set of difficulties. Frequent errors during code uploading to the Arduino necessitated retries, reboots, or driver reinstallations. Testing cloud platforms revealed issues such as incomplete reference materials and libraries, particularly for newer devices.

The original plan included using a GSM Module for SMS notifications during adverse weather conditions. However, despite extensive efforts including trying different SIM cards, modifying code, and replacing the module, stable operation could not be achieved. This setback led to the discovery and implementation of Blynk platform's notification and email alert features as an effective alternative.

Sensor instability posed a significant challenge. The UV sensor, for instance, typically expected to produce readings between 0-15mW/cm<sup>2</sup>, generated anomalous data on the Arduino Uno R4 Wi-Fi. After ruling out issues with code, wiring, and the sensors themselves, the conclusion pointed to incomplete library support for this sensor in the newer model. Similarly, the BH1750 sensor frequently displayed a "Device is not configured" error, with intermittent functionality. These issues highlighted the importance of sensor stability and the potential problems to consider when selecting and using sensors.

Another challenge emerged when the Arduino lost power. Upon restarting, all settings for sensor values used in algorithms, movement paths, and other critical data were lost, preventing normal operation. To address this issue, the relevant data was stored in EEPROM. This solution ensured that even if the Arduino lost power, it could still retrieve the saved data upon restart, allowing for normal operation to resume.

## CHAPTER 5

### SYSTEM EVALUATION AND DISCUSSION

#### 5.1 Testing Setup and Result

```

15:06:32.505 -> Humidity Threshold = 75.00
15:06:32.505 -> Temperature Threshold = 32.00
15:06:32.505 -> Light Threshold = 3000
15:06:32.505 -> UV Threshold = 7.00
15:06:32.505 -> Rain Threshold = 400
15:06:32.505 -> Strategy Value = 2
15:06:32.505 -> Joystick X = 50
15:06:32.505 -> Joystick Y = 50
15:06:32.505 -> Is Recording = 0
15:06:32.505 -> Automation Status = 1
15:06:32.505 -> Time Setting = 0
15:06:32.505 ->
15:06:32.542 -> Humidity (%): 64.9
15:06:32.542 -> Temperature (C): 33.1
15:06:32.542 -> Light intensity (lx): 131
15:06:32.542 -> UV intensity (mW/cm^2): 12.5204
15:06:32.542 -> Rain analog value: 512
15:06:32.542 -> Rain is detected
15:06:32.542 ->

```

*Figure 5.1 Serial Monitor Output*

As illustrated in the Figure 5.1, when the Arduino Uno R4 Wi-Fi is connected to a computer, the output can be successfully observed in the Arduino IDE's Serial Monitor. The first five lines display the threshold values for each sensor, which can be configured in the Blynk App. These values are used to compare against real-time sensor data to determine whether the parameters are suitable for drying clothes.

The "Strategy Value" ranges from 1 to 5, indicating the number of conditions that need to be met for the algorithm to deem the environment suitable for drying clothes.

The subsequent two lines, "Joystick X" and "Y", represent data that determines the direction of each wheel controlled by the motor driver.

"Is Recording" is a Boolean value used to track the movement trajectory of the clothes rack.

"Automation Status" is another Boolean that enables or disables the automation feature.

"Time Setting" is also a Boolean; when true, it indicates that the current time falls within the predetermined clothes-drying time interval.

Finally, the last six lines of data represent the real-time readings from the five sensor parameters.

This comprehensive display provides a clear overview of the system's status, including user-defined thresholds, operational modes, and real-time environmental data, facilitating efficient monitoring and control of the automated clothes-drying system.

## **5.2 Objectives Evaluation**

This project has successfully developed an intelligent, autonomous clothes drying system based on Arduino technology, seamlessly integrating hardware components such as sensors and actuators with a user-friendly mobile application. By evaluating various environmental data including temperature, humidity, light intensity, UV radiation, and rainfall, the system automatically determines optimal drying conditions and controls the movement of the clothes rack accordingly. Beyond these core functionalities, the project has achieved several additional features that enhance user experience and system versatility. These include sending alerts to users' mobile devices and emails, allowing manual control of the clothes rack via a joystick interface, recording and replaying movement trajectories, calculating reverse paths to return the rack to its original position, and enabling users to define specific time periods for drying operations. The latter feature activates the automatic drying function only during the most suitable times, significantly reducing the risk of unexpected exposure to rain or adverse conditions. These achievements demonstrate the project's success in creating a comprehensive, intelligent clothes drying solution that not only automates the process based on environmental factors but also provides users with enhanced control, customization, and peace of mind. The system's ability to adapt to user preferences and environmental conditions while offering both automated and manual control options represents a significant advancement in home automation technology.



## CHAPTER 6

### CONCLUSION AND RECOMMENDATION

#### 6.1 Conclusion

The development of an Arduino-based Smart Cloth Drying System has successfully addressed the challenges of automated clothes drying in fluctuating weather conditions. This project has effectively integrated various environmental sensors, including those for humidity, temperature, light intensity, UV intensity, and rainfall, with an Arduino-controlled system. The implementation of a user-friendly mobile application through the Blynk IoT platform has enhanced the system's accessibility and control.

The system demonstrates significant advancements in home automation technology, offering both automated and manual control options. It successfully adapts to changing weather conditions, making intelligent decisions about clothes drying based on real-time environmental data. The inclusion of features such as path recording, playback, and reverse execution adds a layer of sophistication to the system's functionality. Moreover, the integration of sunrise and sunset timings for operational hours showcases the system's adaptability to natural light cycles.

This project has not only achieved its primary objective of creating intelligent clothes drying solution but has also contributed to energy efficiency and user convenience. The system's ability to prevent unnecessary rewashing due to unexpected rain exposure addresses a common household problem, thereby saving time, water, and energy. The combination of automated decision-making and remote manual control offers users flexibility in managing their laundry, regardless of their location or schedule.

## **6.2 Recommendation**

For the future development and enhancement of the Arduino-based Smart Cloth Drying System, several recommendations can be considered to improve its functionality and user experience. Firstly, incorporating additional sensors, such as wind speed and direction sensors, can further refine the drying process and increase the overall efficiency of the system. By adjusting the drying parameters according to real-time weather conditions, the system can optimize its performance and reduce drying time.

Secondly, implementing energy harvesting technologies, such as small-scale solar panels, would make the system more self-sustainable and decrease its reliance on external power sources. This approach not only promotes eco-friendliness but also provides a cost-effective solution in the long run, making the system more appealing to environmentally conscious users.

Another enhancement involves exploring IoT integration with other smart home devices to offer a more comprehensive home automation experience. By connecting with existing smart home ecosystems, users can enjoy seamless control and monitoring of their cloth drying process from anywhere, adding convenience and efficiency to their daily routines.

Lastly, investigating waterproof designs for outdoor components would enhance the durability and longevity of the system in various weather conditions. Protecting the components from water damage ensures that the system remains reliable and effective, even when exposed to rain or humidity.

These recommendations aim to build upon the current system's strengths by addressing potential areas for improvement and expansion. By implementing these enhancements, the smart cloth drying system can become even more efficient, user-friendly, and versatile, offering a superior solution for modern households.

## REFERENCES

- [1] M. Pinem, C. F. Nasution, and Suherman, "Automatic control circuit design for a clothes dryer," in 2021 5th International Conference on Electrical, Telecommunication and Computer Engineering (ELTICOM), 2021, pp. 84-87.
- [2] I.A. Salihi, S.A. Hulukati, and S. Humena, "Designing an Internet of Things Based Automatic Clothesline," J. Sustain. Eng.: Proc. Ser., vol. 1, no. 2, pp. 240-246, Sep. 2019.
- [3] B.S. Kumar Sathish et al., "Design and Experimental Study of Automatic Cloth Retrieval and Drying System," International Journal of Advanced Research, Ideas and Innovations in Technology, vol. 3, no. 2, pp. 49-57, 2017.
- [4] R. Nandan, R. G. P. and P. Honnavalli, "Automatic Cloth Drying Line Solution Using IoT," 2021 Second International Conference on Advanced Technologies in Intelligent Control, Environment, Computing and Communication Engineering (ICATIECE), 2021, pp. 1-5, doi: 10.1109/ICATIECE56365.2022.10046759.
- [5] AccuWeather. "AccuWeather APIs." AccuWeather. <https://developer.accuweather.com/> (accessed Sept. 5, 2023)
- [6] OpenWeatherMap. "Weather API." OpenWeatherMap. <https://openweathermap.org/api> (accessed Sept. 5, 2023)
- [7] Malaysian Meteorological Department. "Malaysian Meteorological Department Web Service API" Malaysian Meteorological Department. <https://api.met.gov.my/> (accessed Sept. 11, 2023)
- [8] SparkFun Electronics. "SparkFun Arduino IoT Weather Station." SparkFun Electronics. <https://www.sparkfun.com/products/22636> (accessed Sept. 5, 2023)
- [9] Jean-Luc Aufranc. "SparkFun launches ESP32-based 'Arduino IoT Weather Station' with Arduino IoT Cloud integration." CNX Software. <https://www.cnx-software.com/2023/09/03/sparkfun-esp32-arduino-iot-weather-station-with-arduino-iot-cloud/> (accessed Sept. 5, 2023)

**APPENDIX**  
**WEEKLY LOG**  
**FINAL YEAR PROJECT WEEKLY REPORT**  
*(Project II)*

<b>Trimester, Year: T2Y3</b>	<b>Study week no.: 2</b>
<b>Student Name &amp; ID: Chia Shen Yang &amp; 22ACB00552</b>	
<b>Supervisor: Ms. Oh Zi Xin</b>	
<b>Project Title: Development of Arduino-based Smart Cloth Drying System</b>	

**1. WORK DONE**

In FYP1, completed the connection of Sensors, Motor Driver and Arduino, as well as the connection and communication between Arduino and Blynk platform.

**2. WORK TO BE DONE**

Connect GSM Module.

**3. PROBLEMS ENCOUNTERED**

After changing different SIM cards, GSM Modules and coding, it still couldn't function properly.

**4. SELF EVALUATION OF THE PROGRESS**

Not running as smoothly as expected, may need alternative solutions.



\_\_\_\_\_  
Supervisor's signature



\_\_\_\_\_  
Student's signature

# FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

<b>Trimester, Year: T2Y3</b>	<b>Study week no.: 4</b>
<b>Student Name &amp; ID: Chia Shen Yang &amp; 22ACB00552</b>	
<b>Supervisor: Ms. Oh Zi Xin</b>	
<b>Project Title: Development of Arduino-based Smart Cloth Drying System</b>	

## 1. WORK DONE

Failed attempts to connect GSM Module.

## 2. WORK TO BE DONE

Attempted to use Blynk platform's Email and Notification features to replace GSM Module.

## 3. PROBLEMS ENCOUNTERED

GSM still not working properly, using Blynk's Email and Notification features as a replacement.

## 4. SELF EVALUATION OF THE PROGRESS

Successfully used Blynk's Email and Notification features to replace GSM Module, solving one problem.

\_\_\_\_\_  
Supervisor's signature

\_\_\_\_\_  
Student's signature

# FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

<b>Trimester, Year: T2Y3</b>	<b>Study week no.: 6</b>
<b>Student Name &amp; ID: Chia Shen Yang &amp; 22ACB00552</b>	
<b>Supervisor: Ms. Oh Zi Xin</b>	
<b>Project Title: Development of Arduino-based Smart Cloth Drying System</b>	

## 1. WORK DONE

Using Blynk platform's Email and Notification features to replace GSM Module.

## 2. WORK TO BE DONE

Designed various clothes rack movement functions.

## 3. PROBLEMS ENCOUNTERED

By replacing the Motor Drive battery with a 9v battery, solved the problem of two motors running at asynchronous speeds when the Motor Driver was running simultaneously.

## 4. SELF EVALUATION OF THE PROGRESS

Progressing smoothly as expected.



\_\_\_\_\_  
Supervisor's signature



\_\_\_\_\_  
Student's signature

# FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

<b>Trimester, Year: T2Y3</b>	<b>Study week no.: 8</b>
<b>Student Name &amp; ID: Chia Shen Yang &amp; 22ACB00552</b>	
<b>Supervisor: Ms. Oh Zi Xin</b>	
<b>Project Title: Development of Arduino-based Smart Cloth Drying System</b>	

## 1. WORK DONE

Implemented functions to control clothes rack movement using Joystick, record movement path, replay movement path, and replay reverse movement path to return to the original point.

## 2. WORK TO BE DONE

Develop customizable drying time function.

## 3. PROBLEMS ENCOUNTERED

Blynk platform only sends data to change Arduino's time set status at specific set times. If Arduino is not powered on at that time, related variables won't be updated. However, in real usage scenarios, Arduino will always be plugged in, so it's not a major issue.

## 4. SELF EVALUATION OF THE PROGRESS

Smooth progress without unexpected issues.



Supervisor's signature



Student's signature

# FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

<b>Trimester, Year: T2Y3</b>	<b>Study week no.: 10</b>
<b>Student Name &amp; ID: Chia Shen Yang &amp; 22ACB00552</b>	
<b>Supervisor: Ms. Oh Zi Xin</b>	
<b>Project Title: Development of Arduino-based Smart Cloth Drying System</b>	

## 1. WORK DONE

Developed customizable drying time function.

## 2. WORK TO BE DONE

Refactor and comment the code, test the functions.

## 3. PROBLEMS ENCOUNTERED

If power is cut off, Arduino won't save data such as movement paths, set values for various sensors, etc.

## 4. SELF EVALUATION OF THE PROGRESS

The problem is quite difficult, progress is slightly slow.



\_\_\_\_\_  
Supervisor's signature



\_\_\_\_\_  
Student's signature



# FINAL YEAR PROJECT WEEKLY REPORT

*(Project II)*

<b>Trimester, Year: T2Y3</b>	<b>Study week no.: 12</b>
<b>Student Name &amp; ID: Chia Shen Yang &amp; 22ACB00552</b>	
<b>Supervisor: Ms. Oh Zi Xin</b>	
<b>Project Title: Development of Arduino-based Smart Cloth Drying System</b>	

## 1. WORK DONE

Used EEPROM to save data such as movement paths, Sensor settings, etc.  
Refactored and commented the code, tested the functions.

## 2. WORK TO BE DONE

Start writing FYP2 report.

## 3. PROBLEMS ENCOUNTERED

Sometimes the status of Light Intensity and UV Intensity Sensors is unstable, with instances of not sending data.

## 4. SELF EVALUATION OF THE PROGRESS

Overall, everything is good.



\_\_\_\_\_  
Supervisor's signature



\_\_\_\_\_  
Student's signature

## POSTER

# DEVELOPMENT OF ARDUINO-BASED SMART CLOTH DRYING SYSTEM

FACULTY OF INFORMATION COMMUNICATION AND TECHNOLOGY

### INTRODUCTION

This Arduino-based Smart Cloth Drying System provides an intelligent solution for automatic and remote control of clothes drying based on real-time weather conditions.

### OBJECTIVE

To develop an autonomous system that assesses weather data, controls clothes rack movement, enables remote monitoring, and enhances user convenience for daily clothes drying.

### PROPOSED METHOD

- 1.Environmental Data Collection: Sensors measure temperature, humidity, light, UV, and rain
- 2.Cloud Integration: Blynk IoT platform for data visualization and remote access
- 3.Automated & Manual Control: System evaluates conditions and allows user override
- 4.Path Recording & Playback: Records movement sequences for future use
- 5.Customizable Drying Times: Users can set specific operational hours

### WHY IS THIS SYSTEM BETTER THAN EXISTING SOLUTIONS?

- 1.Intelligent Decision Making: Uses multiple sensor inputs for optimal drying
- 2.Enhanced User Experience: Remote monitoring and control via mobile app
- 3.Adaptable Automation: Offers various automation levels based on user preference
- 4.Energy Efficient: Reduces rewashing due to unexpected rain

### CONCLUSION

This system integrates advanced sensors, wireless connectivity, and user-friendly controls to optimize the clothes drying process, offering a practical and adaptable smart home solution.

## ARDUINO CODING

```
// Define Blynk settings
#define BLYNK_PRINT Serial
#define BLYNK_TEMPLATE_ID "TMPL6GXznUpS7"
#define BLYNK_TEMPLATE_NAME "Arduino UNO R4"
#define BLYNK_AUTH_TOKEN "iEnW934atwMSBCQ0pV7RpwonfKfPur02"

// Include necessary libraries
#include <SPI.h>
#include <WiFiS3.h>
#include <BlynkSimpleWifi.h>
#include <Wire.h>
#include <BH1750.h>
#include <Arduino.h>
#include <ML8511.h>
#include <DHT22.h>
#include <EEPROM.h>
#include <queue>
#include <stack>

// Define pin assignments for various sensors and motors
#define RAIN_POWER_PIN 3
#define RAIN_DO_PIN 8
#define RAIN_AO_PIN A2

#define MOTOR_ENA 0
#define MOTOR_IN1 1
#define MOTOR_IN2 2
#define MOTOR_IN3 9
#define MOTOR_IN4 10
#define MOTOR_ENB 11
#define MAX_ACTIONS 50

#define DHT_DATA_PIN 7
#define UV_ANALOG_PIN A0
#define UV_ENABLE_PIN A1

// Initialize sensor objects
ML8511 uvSensor(UV_ANALOG_PIN, UV_ENABLE_PIN);
DHT22 dht22(DHT_DATA_PIN);
```

## APPENDIX

```
// Define an enum for car movement actions
enum Action {
    STOP = 0,
    FORWARD = 1,
    BACKWARD = 2,
    LEFT = 3,
    RIGHT = 4
};

// WiFi credentials
char wifiSsid[] = "vivoY35";
char wifiPass[] = "12345678";

// Global variables for various functionalities
unsigned long startTime = 0;
const unsigned long runDuration = 50;
bool isOutside = false;
int ledPin = 13;
int conditionCount;
double humidityThreshold;
double temperatureThreshold;
int lightThreshold;
double uvThreshold;
int rainThreshold;
int strategyValue;
int joystickX = 50;
int joystickY = 50;
int motorSpeed = 255;
bool isRecording = false;
unsigned long lastActionTime = 0;
bool isFirstAction = true;
int replayForwardFlag;
int replayBackwardFlag;
bool autoStatus;
bool timeSet;

// Initialize Blynk timer and light sensor
BlynkTimer timer;
BH1750 lightMeter;
```

## APPENDIX

```
// Define a struct to represent motor actions
struct MotorAction {
    Action action;
    unsigned int duration;
    MotorAction() : action(STOP), duration(0) {}
    MotorAction(Action a, unsigned int d) : action(a), duration(d) {}
};

// Define a struct for EEPROM data layout
struct EEPROMLayout {
    double humidityThreshold;
    double temperatureThreshold;
    int lightThreshold;
    double uvThreshold;
    int rainThreshold;
    int strategyValue;
    bool isOutside;
    bool autoStatus;
    size_t actionSequenceSize;
    MotorAction actions[MAX_ACTIONS];
};

EEPROMLayout eepromData;

// Data structures to store recorded actions
std::queue<MotorAction> forwardQueue;
std::stack<MotorAction> backwardStack;
std::queue<MotorAction> tempForwardQueue;
std::stack<MotorAction> tempBackwardStack;
```

```
// Blynk virtual pin handlers
BLYNK_WRITE(V0) {
  // Handle indoor/outdoor status changes
  int newValue = param.asInt();
  if (newValue != isOutside) {
    if (newValue == 1 && !isOutside) {
      replayForward();
      isOutside = true;
      Blynk.logEvent("move_out");
    } else if (newValue == 0 && isOutside) {
      replayBackward();
      isOutside = false;
      Blynk.logEvent("move_in");
    }
    updateLEDStatus();
    saveToEEPROM();
  }
}

BLYNK_WRITE(V6) {
  // Update humidity threshold
  humidityThreshold = param.asDouble();
  saveToEEPROM();
}

BLYNK_WRITE(V7) {
  // Update temperature threshold
  temperatureThreshold = param.asDouble();
  saveToEEPROM();
}

BLYNK_WRITE(V8) {
  // Update light threshold
  lightThreshold = param.asInt();
  saveToEEPROM();
}

BLYNK_WRITE(V9) {
  // Update UV threshold
  uvThreshold = param.asDouble();
  saveToEEPROM();
}
```

```
BLYNK_WRITE(V10) {
  // Update rain threshold
  rainThreshold = param.asInt();
  saveToEEPROM();
}

BLYNK_WRITE(V11) {
  // Update strategy value
  strategyValue = param.asInt();
  saveToEEPROM();
}

BLYNK_WRITE(V12){
  // Update joystick X value
  joystickX = param[0].asInt();
}

BLYNK_WRITE(V13){
  // Update joystick Y value
  joystickY = param[0].asInt();
}

BLYNK_WRITE(V14) {
  // Handle recording state
  isRecording = param.asInt();
  if (isRecording) {
    while (!forwardQueue.empty()) forwardQueue.pop();
    while (!backwardStack.empty()) backwardStack.pop();
    lastActionTime = millis();
    isFirstAction = true;
  } else {
    saveToEEPROM();
  }
}

BLYNK_WRITE(V15){
  // Trigger forward replay
  replayForwardFlag = param.asInt();
}

BLYNK_WRITE(V16){
  // Trigger backward replay
  replayBackwardFlag = param.asInt();
}
```

## APPENDIX

```
BLYNK_WRITE(V17){
  // Update time set status
  timeSet = param.asInt();
}

BLYNK_WRITE(V19){
  // Update auto status
  autoStatus = param.asInt();
  saveToEEPROM();
}

void setup()
{
  // Initialize serial communication and sensors
  Wire.begin();
  Serial.begin(115200);
  Blynk.begin(BLYNK_AUTH_TOKEN, wifiSsid, wifiPass);
  lastActionTime = millis();

  // Set up motor control pins
  pinMode(MOTOR_ENA, OUTPUT);
  pinMode(MOTOR_IN1, OUTPUT);
  pinMode(MOTOR_IN2, OUTPUT);
  pinMode(MOTOR_IN3, OUTPUT);
  pinMode(MOTOR_IN4, OUTPUT);
  pinMode(MOTOR_ENB, OUTPUT);

  lightMeter.begin();

  // Set up rain sensor pins
  pinMode(RAIN_POWER_PIN, OUTPUT);
  pinMode(RAIN_DO_PIN, INPUT);
  pinMode(ledPin, OUTPUT);

  // Load saved data from EEPROM and update LED status
  loadFromEEPROM();
  updateLEDStatus();
}
```



```
void loop()
{
    conditionCount = 5;
    Blynk.run();
    timer.run();

    // Print current sensor thresholds and control values
    Serial.print("Humidity Threshold = ");
    Serial.println(humidityThreshold);
    Serial.print("Temperature Threshold = ");
    Serial.println(temperatureThreshold);
    Serial.print("Light Threshold = ");
    Serial.println(lightThreshold);
    Serial.print("UV Threshold = ");
    Serial.println(uvThreshold);
    Serial.print("Rain Threshold = ");
    Serial.println(rainThreshold);
    Serial.print("Strategy Value = ");
    Serial.println(strategyValue);
    Serial.print("Joystick X = ");
    Serial.println(joystickX);
    Serial.print("Joystick Y = ");
    Serial.println(joystickY);
    Serial.print("Is Recording = ");
    Serial.println(isRecording);
    Serial.print("Automation Status = ");
    Serial.println(autoStatus);
    Serial.print("Time Setting = ");
    Serial.println(timeSet);
    Serial.println("");

    // Read rain sensor
    digitalWrite(RAIN_POWER_PIN, HIGH);
    int rainState = digitalRead(RAIN_DO_PIN);
    int rainValue = analogRead(RAIN_AO_PIN);
    digitalWrite(RAIN_POWER_PIN, LOW);

    // Read light sensor
    uint16_t lightIntensity = lightMeter.readLightLevel();
```

```

// Read UV sensor
uvSensor.enable();
float uvIntensity = uvSensor.getUV();
uvSensor.disable();

// Read temperature and humidity
float temperature = dht22.getTemperature();
float humidity = dht22.getHumidity();

// Print sensor readings
Serial.print("Humidity (%): ");Serial.println(humidity, 1);
Serial.print("Temperature (C): ");Serial.println(temperature, 1);
Serial.print("Light intensity (lx): ");Serial.println(lightIntensity);
Serial.print("UV intensity (mW/cm^2): ");Serial.println(uvIntensity,
4);
Serial.print("Rain analog value: ");Serial.println(rainValue);

if (rainState == HIGH)
    Serial.println("Rain is NOT detected");
else
    Serial.println("Rain is detected");
Serial.println("");

// Send sensor data to Blynk app
Blynk.virtualWrite(V1, humidity);
Blynk.virtualWrite(V2, temperature);
Blynk.virtualWrite(V3, lightIntensity);
Blynk.virtualWrite(V4, uvIntensity);
Blynk.virtualWrite(V5, rainValue);

// Control smart car movement
smartCar();

// Check environmental conditions and update conditionCount
if(humidityThreshold < humidity) conditionCount--;
if(temperatureThreshold > temperature) conditionCount--;
if(lightThreshold > lightIntensity) conditionCount--;
if(uvThreshold > uvIntensity) conditionCount--;
if(rainThreshold < rainValue) conditionCount--;

```

```

// Automated control based on environmental conditions
if(autoStatus == true){
  if (conditionCount >= strategyValue && !isOutside && timeSet ==
true) {
    replayForward();
    isOutside = true;
    Blynk.logEvent("move_out");
    updateLEDStatus();
    saveToEEPROM();
  } else if ((conditionCount < strategyValue && isOutside) ||
(isOutside && timeSet == false)) {
    replayBackward();
    isOutside = false;
    Blynk.logEvent("move_in");
    updateLEDStatus();
    saveToEEPROM();
  }
  if (conditionCount < strategyValue) {
    Blynk.logEvent("rain");
  }
}

// Handle manual replay requests
if(replayForwardFlag == 1){
  replayForward();
  replayForwardFlag = 0;
} else if(replayBackwardFlag == 1){
  replayBackward();
  replayBackwardFlag = 0;
}

delay(10);
}

void updateLEDStatus() {
  // Update LED status based on indoor/outdoor position
  digitalWrite(ledPin, isOutside ? HIGH : LOW);
}

```

```

void smartCar() {
    // Control car movement based on joystick input
    unsigned long currentTime = millis();
    static Action lastAction = STOP;

    if (currentTime - lastActionTime >= 100) {
        Action currentAction;
        if (joystickY > 70) {
            currentAction = FORWARD;
        } else if (joystickY < 30) {
            currentAction = BACKWARD;
        } else if (joystickX < 30) {
            currentAction = LEFT;
        } else if (joystickX > 70) {
            currentAction = RIGHT;
        } else {
            currentAction = STOP;
        }

        if (currentAction != lastAction) {
            if (isRecording) {
                if (!isFirstAction) {
                    forwardQueue.push(MotorAction(lastAction, currentTime -
lastActionTime - 100));
                    backwardStack.push(MotorAction(getOppositeAction(lastAction),
currentTime - lastActionTime - 100));
                } else {
                    isFirstAction = false;
                }
            }
            performAction(currentAction);
            Serial.println(actionToString(currentAction));
            lastAction = currentAction;
            lastActionTime = currentTime;
        }
    }
}

```

```

Action getOppositeAction(Action action) {
    // Get the opposite action for reverse playback
    switch(action) {
        case FORWARD: return BACKWARD;
        case BACKWARD: return FORWARD;
        case LEFT: return RIGHT;
        case RIGHT: return LEFT;
        default: return STOP;
    }
}

void replayForward() {
    // Replay recorded forward actions
    tempForwardQueue = forwardQueue;
    while (!tempForwardQueue.empty()) {
        MotorAction action = tempForwardQueue.front();
        tempForwardQueue.pop();
        performAction(action.action);
        delay(action.duration);
        performAction(STOP);
    }
    isOutside = true;
    Blynk.virtualWrite(V0, isOutside);
    Blynk.logEvent("move_out");
    updateLEDStatus();
    saveToEEPROM();
}

void replayBackward() {
    // Replay recorded actions in reverse
    tempBackwardStack = backwardStack;
    while (!tempBackwardStack.empty()) {
        MotorAction action = tempBackwardStack.top();
        tempBackwardStack.pop();
        performAction(action.action);
        delay(action.duration);
        performAction(STOP);
    }
    isOutside = false;
    Blynk.virtualWrite(V0, isOutside);
    Blynk.logEvent("move_in");
    updateLEDStatus();
    saveToEEPROM(); }

```

```
void performAction(Action action) {
    // Perform the specified car movement action
    switch(action) {
        case FORWARD: carForward(); break;
        case BACKWARD: carBackward(); break;
        case LEFT: carLeft(); break;
        case RIGHT: carRight(); break;
        default: carStop(); break;
    }
}
```

```
String actionToString(Action action) {
    // Convert action enum to string for logging
    switch(action) {
        case FORWARD: return "forward";
        case BACKWARD: return "backward";
        case LEFT: return "left";
        case RIGHT: return "right";
        default: return "stop";
    }
}
```

```
void carForward() {
    // Move car forward
    analogWrite(MOTOR_ENA, motorSpeed);
    analogWrite(MOTOR_ENB, motorSpeed);
    digitalWrite(MOTOR_IN1, LOW);
    digitalWrite(MOTOR_IN2, HIGH);
    digitalWrite(MOTOR_IN3, HIGH);
    digitalWrite(MOTOR_IN4, LOW);
}
```

```
void carBackward() {
    // Move car backward
    analogWrite(MOTOR_ENA, motorSpeed);
    analogWrite(MOTOR_ENB, motorSpeed);
    digitalWrite(MOTOR_IN1, HIGH);
    digitalWrite(MOTOR_IN2, LOW);
    digitalWrite(MOTOR_IN3, LOW);
    digitalWrite(MOTOR_IN4, HIGH);
}
```

```
void carLeft() {
    // Turn car left
    analogWrite(MOTOR_ENA, motorSpeed);
    analogWrite(MOTOR_ENB, motorSpeed);
    digitalWrite(MOTOR_IN1, HIGH);
    digitalWrite(MOTOR_IN2, LOW);
    digitalWrite(MOTOR_IN3, HIGH);
    digitalWrite(MOTOR_IN4, LOW);
}

void carRight() {
    // Turn car right
    analogWrite(MOTOR_ENA, motorSpeed);
    analogWrite(MOTOR_ENB, motorSpeed);
    digitalWrite(MOTOR_IN1, LOW);
    digitalWrite(MOTOR_IN2, HIGH);
    digitalWrite(MOTOR_IN3, LOW);
    digitalWrite(MOTOR_IN4, HIGH);
}

void carStop() {
    // Stop the car
    digitalWrite(MOTOR_IN1, LOW);
    digitalWrite(MOTOR_IN2, LOW);
    digitalWrite(MOTOR_IN3, LOW);
    digitalWrite(MOTOR_IN4, LOW);
}

void saveToEEPROM() {
    // Save current settings and recorded actions to EEPROM
    eepromData.humidityThreshold = humidityThreshold;
    eepromData.temperatureThreshold = temperatureThreshold;
    eepromData.lightThreshold = lightThreshold;
    eepromData.uvThreshold = uvThreshold;
    eepromData.rainThreshold = rainThreshold;
    eepromData.strategyValue = strategyValue;
    eepromData.isOutside = isOutside;
    eepromData.autoStatus = autoStatus;

    // Save recorded actions
    eepromData.actionSequenceSize = forwardQueue.size();
    std::queue<MotorAction> tempQueue = forwardQueue;
```

```

    for (size_t i = 0; i < eepromData.actionSequenceSize && i <
MAX_ACTIONS; i++) {
        eepromData.actions[i] = tempQueue.front();
        tempQueue.pop();
    }

    // Write the entire struct to EEPROM
    EEPROM.put(0, eepromData);
}

void loadFromEEPROM() {
    // Load settings and recorded actions from EEPROM
    EEPROM.get(0, eepromData);
    humidityThreshold = eepromData.humidityThreshold;
    temperatureThreshold = eepromData.temperatureThreshold;
    lightThreshold = eepromData.lightThreshold;
    uvThreshold = eepromData.uvThreshold;
    rainThreshold = eepromData.rainThreshold;
    strategyValue = eepromData.strategyValue;
    isOutside = eepromData.isOutside;
    autoStatus = eepromData.autoStatus;

    // Clear existing queues and reconstruct them from saved data
    while (!forwardQueue.empty()) forwardQueue.pop();
    while (!backwardStack.empty()) backwardStack.pop();

    for (size_t i = 0; i < eepromData.actionSequenceSize && i <
MAX_ACTIONS; i++) {
        forwardQueue.push(eepromData.actions[i]);
        backwardStack.push(MotorAction(getOppositeAction(eepromData.actions[
i].action), eepromData.actions[i].duration));
    }
}

```



## PLAGIARISM CHECK RESULT

FYP2\_ChiaSY

### ORIGINALITY REPORT

<b>9</b> %	<b>5</b> %	<b>6</b> %	<b>3</b> %
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

### PRIMARY SOURCES

<b>1</b>	Maksum Pinem, Chairul Fahmi Nasution, Suherman. "Automatic Control Circuit Design for a Clothes Dryer", 2021 5th International Conference on Electrical, Telecommunication and Computer Engineering (ELTICOM), 2021 Publication	<b>1</b> %
<b>2</b>	docplayer.net Internet Source	<b>1</b> %
<b>3</b>	Rohan Nandan, Revathi GP, Prasad Honnavalli. "Automatic Cloth Drying Line Solution Using IoT", 2022 Second International Conference on Advanced Technologies in Intelligent Control, Environment, Computing & Communication Engineering (ICATIECE), 2022 Publication	<b>1</b> %
<b>4</b>	Erika T. Machtinger, Emma N.I. Weeks, Christopher J. Geden, Erica Lacher. "Pests and parasites of horses", Brill, 2022 Publication	<b>&lt;1</b> %

5	Submitted to Global College of Engineering and technology, Oman Student Paper	<1 %
6	Submitted to University of Dundee Student Paper	<1 %
7	jp.ic-on-line.cn Internet Source	<1 %
8	repository.sustech.edu Internet Source	<1 %
9	Submitted to Aligarh Muslim University, Aligarh Student Paper	<1 %
10	Submitted to Southampton Solent University Student Paper	<1 %
11	Wei He, M. Tariq Iqbal. "A Novel Design of a Low-Cost SCADA System for Monitoring Standalone Photovoltaic Systems", Journal of Electronics and Electrical Engineering, 2024 Publication	<1 %
12	community.blynk.cc Internet Source	<1 %
13	Submitted to Higher Education Commission Pakistan Student Paper	<1 %

14	Nampally Tejasri, Mongkol Ekapanyapong. "Material Recognition Using Deep Learning Techniques", IndiaRxiv, 2019 Publication	<1%
15	www.instructables.com Internet Source	<1%
16	Submitted to University of Salford Student Paper	<1%
17	Submitted to Universidad Tecnológica Centroamericana UNITEC Student Paper	<1%
18	www.coursehero.com Internet Source	<1%
19	Submitted to University of Birmingham Student Paper	<1%
20	Dan Hosken. "An Introduction to Music Technology", Routledge, 2019 Publication	<1%
21	Divyansh Arun, Karun Sahani, Ansh Gangwar, Samrat Borah, Gunjan Aggarwal. "Being Human: An AI Based Android Application for Improving, Easing and Enhancing Our Lifestyle", 2022 3rd International Conference for Emerging Technology (INCET), 2022 Publication	<1%

22	M. Y. Hew, A. M. Andrew, Y. Z. Q. Faith, Y. Y. Low, M. K. Y. Natasha. "Automated clothesline retrieval system using LDR and raindrop sensors", Engineering Technology International Conference (ETIC 2022), 2022 Publication	<1%
23	Melissa Raven. "Depression and antidepressants in Australia and beyond: A critical public health analysis", Thesis Commons, 2018 Publication	<1%
24	Wajahat Khalid, Mohsin Jamil, Ashraf Ali Khan, Qasim Awais. "Open-Source Internet of Things-Based Supervisory Control and Data Acquisition System for Photovoltaic Monitoring and Control Using HTTP and TCP/IP Protocols", Energies, 2024 Publication	<1%
25	<a href="https://create.arduino.cc">create.arduino.cc</a> Internet Source	<1%
26	Submitted to Universiti Malaysia Perlis Student Paper	<1%
27	Anes Mohamed, Gunaseelan Gunasegaran, Daminda Herath. "Cloud-based Weather Condition Monitoring System using ESP8266 and Amazon Web Services", 2023 8th	<1%

## International Conference on Information Technology Research (ICITR), 2023

Publication

28	Ton Duc Thang University Publication	<1 %
29	Submitted to University of Greenwich Student Paper	<1 %
30	Hanoi Pedagogical University 2 Publication	<1 %
31	git.sdf.org Internet Source	<1 %
32	mil.ufl.edu Internet Source	<1 %
33	Submitted to American University of Kuwait Student Paper	<1 %
34	Hean Ong, Hean Ong, Elok Tee, R. Sureswaran. "Scalability study of ad-hoc wireless mobile network routing protocol in sparse and dense networks", The 2nd International Conference on Distributed Frameworks for Multimedia Applications, 2006 Publication	<1 %
35	Lefrancois, Guy . "Psychology: The Human Puzzle, Fourth Edition", UAGC, 2023 Publication	<1 %

36	Submitted to University of Nottingham Student Paper	<1 %
37	gitlab.au.dk Internet Source	<1 %
38	www.allreadable.com Internet Source	<1 %
39	Aliye Saraç, Nesrin Özdener. "chapter 4 Adapting to the Industry 4.0 Era", IGI Global, 2024 Publication	<1 %
40	Submitted to De Montfort University Student Paper	<1 %
41	Submitted to Middlesex University Student Paper	<1 %
42	Mohit Kumar, Saifatullah, Simon Tongbram. "Development of IoT Based Home Automation System Using Smartphone", 2023 5th International Conference on Energy, Power and Environment: Towards Flexible Green Energy Technologies (ICEPE), 2023 Publication	<1 %
43	dalspace.library.dal.ca Internet Source	<1 %
44	programtalk.com Internet Source	<1 %

45 repository.pnj.ac.id <1%  
Internet Source

---

46 www.furniture-malaysian.com <1%  
Internet Source

---


47 www.researchsquare.com <1%  
Internet Source

---

Exclude quotes On

Exclude matches < 8 words

Exclude bibliography On

<b>Universiti Tunku Abdul Rahman</b>			
<b>Form Title : Supervisor's Comments on Originality Report Generated by Turnitin</b>			
	FM-IAD-005	Rev No.: 0	Effective Date: 01/10/2013   Page No.: 1 of 1

**FACULTY OF INFORMATION AND COMMUNICATION  
TECHNOLOGY**

<b>Full Name(s) of Candidate(s)</b>	Chia Shen Yang
<b>ID Number(s)</b>	22ACB00552
<b>Programme / Course</b>	BACHELOR OF COMPUTER SCIENCE (HONOURS) / UCCC3596 PROJECT II
<b>Title of Final Year Project</b>	Development of Arduino-based Smart Cloth Drying System

<b>Similarity</b>	<b>Supervisor's Comments</b> (Compulsory if parameters of originality exceeds the limits approved by UTAR)
<b>Overall similarity index: 9 %</b>  <b>Similarity by source</b> Internet Sources: 5_% Publications: 6_% Student Papers: 3_%	
<b>Number of individual sources listed of more than 3% similarity: -</b>	
<b>Parameters of originality required, and limits approved by UTAR are as Follows:</b> (i) Overall similarity index is 20% and below, and (ii) Matching of individual sources listed must be less than 3% each, and (iii) Matching texts in continuous block must not exceed 8 words  <i>Note: Parameters (i) – (ii) shall exclude quotes, bibliography and text matches which are less than 8 words.</i>	

Note Supervisor/Candidate(s) is/are required to provide softcopy of full set of the originality report to Faculty/Institute

***Based on the above results, I hereby declare that I am satisfied with the originality of the Final Year Project Report submitted by my student(s) as named above.***





\_\_\_\_\_  
Signature of Supervisor

Name:

OH ZI XIN \_\_\_\_\_

Date:

13/9/2024 \_\_\_\_\_

\_\_\_\_\_  
Signature of Co-Supervisor

Name: \_\_\_\_\_

Date: \_\_\_\_\_



## UNIVERSITI TUNKU ABDUL RAHMAN

### FACULTY OF INFORMATION & COMMUNICATION

#### TECHNOLOGY (KAMPAR CAMPUS)

#### CHECKLIST FOR FYP2 THESIS SUBMISSION

Student Id	22ACB00552
Student Name	Chia Shen Yang
Supervisor Name	Ms. Oh Zi Xin

TICK (√)	DOCUMENT ITEMS
	Your report must include all the items below. Put a tick on the left column after you have checked your report with respect to the corresponding item.
√	Title Page
√	Signed Report Status Declaration Form
√	Signed FYP Thesis Submission Form
√	Signed form of the Declaration of Originality
√	Acknowledgement
√	Abstract
√	Table of Contents
√	List of Figures (if applicable)
√	List of Tables (if applicable)
√	List of Symbols (if applicable)
√	List of Abbreviations (if applicable)
√	Chapters / Content
√	Bibliography (or References)
√	All references in bibliography are cited in the thesis, especially in the chapter of literature review
√	Appendices (if applicable)
√	Weekly Log
√	Poster
√	Signed Turnitin Report (Plagiarism Check Result - Form Number: FM-IAD-005)

APPENDIX

√	I agree 5 marks will be deducted due to incorrect format, declare wrongly the ticked of these items, and/or any dispute happening for these items in this report.
---	---

\*Include this form (checklist) in the thesis (Bind together as the last page)

I, the author, have checked and confirmed all the items listed in the table are included in my report.



\_\_\_\_\_  
(Signature of Student)

Date: 13/09/2024