Applied Information Technology And Computer Science Vol. 3 No. 2 (2022) 990-1006
© Universiti Tun Hussein Onn Malaysia Publisher's Office

AITCS

Homepage: http://publisher.uthm.edu.my/periodicals/index.php/aitcs
e-ISSN :2773-5141

Development of Automatic Watering System Using Arduino and Android Base for Southern Sissoo

Muhammad Hazim Mohd Alim, Noraini Ibrahim [1]*

[1]Faculty of Computer Science and Information Technology,
Universiti Tun Hussein Onn Malaysia, Parit Raja, Batu Pahat, 86400, MALAYSIA

DOI: https://doi.org/10.30880/aitcs.2022.03.02.060
Received 9 August 2022; Accepted 28 October 2022; Available online 30 November 2022

Abstract: Automatic Watering System is a mobile application watering schedule management system specifically develop to assist the watering session of crop for Southern Sissoo. Currently at the company, they lack manpower and hiring new worker is not an option as it will not benefit the company in the matter of long-term cost-effective. The main goal of this system development is to improve the quality of work at Southern Sissoo business particularly because they are still using the traditional physical watering method that has many flaws compared to today's technique. Using Arduino and Android application, this system will increase in efficiency compared to the current physical method. With the proposed system, user can create custom watering session remotely from the farm and check the data of current and past temperature, humidity, light intensity and etc. The system will make the process of watering session and technique at the company improved and resource efficient.

Keywords: Automatic Watering System, Arduino, Android application

1. Introduction

Agriculture is the practice of cultivating plants and livestock. The growth of agriculture means economic development [1]. In Malaysia, smallholder, which is a small farm operating under a small-scale agriculture model is common and widely known. Plant, like human, has basic need and one of them is water. In any smallholding or big industrial agriculture company, taking care of the plant needs is crucial in maintaining the crops survivability and quality. One of the ways to assist the production of crops is using irrigation system, which is the artificial process of applying controlled amounts of water to land to assist in the production of crops [2].

Southern Sissoo is a smallholding company that was established in Pasir Gudang, Johor. It is a new company that focused on producing Sissoo Spinach that are also well known "Bayam Brazil" as their crops. Like any smallholding company, they are still using manpower to water their crops. Each worker has their own scheduled task, and one of it is to water the crop every day. It is also to be noted that the worker are consist of people with disability and single mom.

*Corresponding author: noraini@uthm.edu.my
2020 UTHM Publisher. All rights reserved.

Currently in the company, there exist a few problems with the current system. Because there is a rising demand for Sissoo Spinach in the market, the number of crops has to be increase and worker need to take care more crops per capita. This led to lack of manpower to water the crop and take more time to complete the task. As we know, workers are not exclusively to water the crops, they also need to fertilize the crop and do other activity to maintain the quality of it. Increasing number of workers currently is not an option as it will cost ineffective for the company. Furthermore, there are no possible way to water the crop remotely. Many systems that used for garden irrigation are located on the fare ways of cities and difficult to monitor therefore, manual operation will cause many problems [3]. When there is no worker available at the farm particularly during holiday season or worker taking leave, the crop is not watered and thus reduced the quality of the sissoo. Next, because Southern Sissoo is currently in tight budget, they really need a watering system to be developed rather than hiring new worker because hiring new worker is not cost effective for the long-term benefit for the company.

To overcome all the problems that had been mentioned, the development of Automatic Watering System for Sissoo Spinach has been proposed so that the watering process for this company can become better and efficient. Automation and wireless technology have become a key technology in the twenty-first century. It helps communication between one point to another without the use of cables, and this makes the system to be more secure [4]. The system will be developed as an android based so administrator can control the watering of their crop at the tip of their finger miles away from the farm. Arduino will also be implemented to the system to create the automatic watering model. The system will also allow user to create a scheduled watering session for the crop and record every watering session so that repeatable watering session will not happen. To end, it is justified why the development of this system is necessary to increase the productivity and efficiency of the company.

2. Related Work

2.1 Internet of Thing (IoT)

IoT means the ability to make everything around us starting from (i.e. machine, devices, mobile phone and cars) even (cities and roads) are expected to be connected to the Internet with an intelligent behavior and taking into account the existence of the kind of autonomy and privacy [5]. A person with a heart monitor implant, a farm animal with a biochip transponder, an automobile with built-in sensors to alert the driver when tire pressure is low, or any other natural or man-made object that can be assigned an Internet Protocol (IP) address and can transfer data over a network are all examples of things in the internet of things. An IoT ecosystem is made up of web-enabled smart devices that gather, send, and act on data from their surroundings using embedded systems such as CPUs, sensors, and communication hardware. By connecting to an IoT gateway or other edge device, IoT devices can share sensor data that is either routed to the cloud for analysis or examined locally. These gadgets may occasionally communicate with one another and act on the information they receive. Although individuals can engage with the devices to set them up, give them instructions, or retrieve data, the gadgets do the majority of the work without human participation.

2.2 Arduino Technology

The Arduino project began in 2005 as a tool for students at the Interaction Design Institute Ivrea, Italy [6]. Arduino is an open-source electronics platform based on easy-to-use hardware and software [7]. Arduino is made up of a hardware programmable circuit board (also known as a microcontroller) and software, known as an IDE (Integrated Development Environment), that runs on your computer and is used to create and upload computer code to the physical board. The Arduino platform has grown in popularity among those who are just getting started with electronics, and for good cause. Unlike most prior programmable circuit boards, the Arduino does not require a separate piece of hardware (known as a programmer) to load new code into the board; instead, a USB cable is all that is required. Furthermore, the Arduino IDE makes programming easier by using a simplified form of C++. Finally,

Arduino offers a standard form factor that separates the microcontroller's tasks into a more manageable packaging.

2.3 Android Application

An Android application is a piece of software that runs on the Android operating system. A typical Android app is intended for a smartphone or tablet PC running on the Android OS because the Android platform is built for mobile devices. Although developers can make Android apps available through their websites, most of Android apps are uploaded and published on the Android Market, a specialized online marketplace for these applications. Both free and paid apps are available in the Android Market. Android apps employ Java core libraries and are created in the Java programming language. They're initially compiled into Dalvik executables, which operate on the Dalvik virtual machine, which is a mobile-specific virtual machine. The Android software development kit (SDK) is available for download from the Android website. For developing Android apps, the SDK offers tools, sample code, and essential documents. The App Inventor is a useful tool for inexperienced developers that want to experiment with Android development. As though fitting together puzzle pieces, a user can build an Android app using this online programmed.

2.4 Comparison with Existing System

Table 1 summarizes the comparison of three existing systems with a proposed system. Three current systems were investigated in order to gather more data for the proposed system development. The comparisons are between the proposed system's modules and characteristics.

Table 1: Comparison of 3 Existing System with Proposed System

Systems	Automatic Shrimp Feeder System	Automated Water Irrigation System and Crop Suggestion	Automatic Plant Watering System Using Android.	Automatic Watering System Using Arduino and Android Base for Southern Sissoo
Homepage Module	Available	Available	Available	Available
Recorded Data from Sensor Module	Available	Available	Available	Available
Custom Scheduled Watering Module	Unavailable	Unavailable	Unavailable	Available
Manual toggle button to switch on or off feeder/watering system	Available	Available	Available	Available
Crop Suggestion Module (Redundant)	Unavailable	Available	Unavailable	Unavailable

In summary, the proposed system will have all the functionality of the three compared system except Crop Suggestion Module because it is redundant and will not be required for any of the stakeholder.

3. Methodology/Framework

The system that will be develop will be using the Prototype Model. Prototyping Model is a software development model in which prototype is built, tested, and reworked until an acceptable prototype is achieved [8]. It encourages developers to create a sample of the resolution in order to evaluate its functional essence to users and iterate before generating the final result. This could also mean that the system is developed, tested, and tweaked until an acceptable result is achieved, leading to the ideal product that the client envisioned. Table 2 shows all the software development activities as well as task during each development phases.

Table 2: Software development activities and their task

Phase	Task	Output
Planning	Identify problems, scope and objectives	- Gantt Chart - Proposal
Analysis	- Interview and observe client - Gather all the information	- Functional requirement - Non-functional requirement - Activity diagram - Use Case diagram - Class diagram - Requirement Definition
Design	- Design wireframe for the system before begin developing - Design the interface of the system based on the requirement - Use Fluid UI to design system wireframe design - Use C#/C++ to develop Arduino code - Use Google Cloud Firebase as database	- Interface of the system - Scheme Table - Data dictionary
Implementation	- Conduct testing of the system - Repair the fault of the system - Develop Android application using Dart language on Visual Studio Code	- System - Test case - Test report
Prototype	- Detect any error and improvement that can be implemented into the system - Fix all the error that had been identified and proceed to Prototype 2 - If any error exist, development cycle will repeat back to analysis phase	- Prototype 1 and 2, Finish System - Make sure system met with user requirement

3.1 System Requirement Analysis

System requirements are the configuration that a system must have in order for a hardware or software application to run smoothly and efficiently [9]. Analyzing system requirement is a process where the requirement of the system that will be developed need to be fulfil were identified. Both functional requirement and non-functional requirement are included in system requirement. Table 3 shows system functional module.

Table 3: System functional module

No.	Module	Function	User
1.	Session History Module	Every watering session will be recorded and user can view past history of watering session.	Administrator
2.	Watering Schedule Module	Where user can set up a custom watering schedule for watering session for the crop.	Administrator
3.	Device Information Module	Every watering device can configure from this module and device information can be edit here.	Administrator

3.2 Functional Requirement and Non-Functional Requirements Analysis

In software engineering and systems engineering, a functional requirement defines a function of a system or its component, where a function is described as a specification of behavior between inputs and outputs [10]. Table 4 shows functional requirements of the system.

Table 4: Functional requirements

No	Module	Description
1.	Session History Module	• System should be able to show every recorded watering session • System should allow user to select any recorded watering session • System should be able to show every information for selected session when selected
2.	Watering Schedule Module	• System should allow user to make custom scheduled watering system with their own preference. • System should allow user to edit or delete created watering scheduled. • System should record every watering session and update it to the database.
3.	Device Information Module	• System should allow user to switch on or off the watering device. • System should allow user to edit device information System should let user to set the strength of water pump from the module.

Non-functional requirements (NFRs) define constraints which affect how the system should do it [11]. It is used to judge the operation of a system, rather that the specific behavior or function of the system. Table 5 shows non-functional requirement for the developed system.

Table 5: Non-functional requirements of the developed system

No	Requirements	Description
1.	Performance	The system should be able to be use all the time without any problem
2.	Operation	Time requires to upload or retrieve any data from the database should be less than 1 minute.
3.	Credibility	Every module in the system should work perfectly fine.
4.	Usability	The system should be easy to learn and user friendly.

3.3 User Requirement Analysis

User requirements define the expectation of user from the functionality of the system. Table 6 shows the user requirements of the developed system.

Table 6: User requirements of the developed system

No.	User Requirements
1.	User should be able to see all the module in the navigation bar
2.	User should be able to navigate to other module via navigation bar
3.	User should be able to see connected device at the dashboard
4.	User should be able to see every recorded watering session
5.	User should be able to select any recorded watering session
6.	User should be able review recorded information for selected watering session
7.	User should be able to make a custom scheduled watering session
8.	User should be able to edit or delete created watering scheduled
9.	User should be able to turn on or off watering device
10.	User should be able to edit device information
11.	User should be able to set the strength of the water pump

3.4 Use Case Diagram

The use case specification is a detailed description that describes and explains the details of the Automatic Watering System using Arduino for Southern Sissoo Use Case Diagrams. A use case specification comprises contextual information of the use case, its change history, the complete graph of possible pathways, attached requirements and open issues [12]. It explains how a user interacts with a system as well as how the system responds to the user's actions. The next part provides an overview of each use case. Figure 1 shows the use case diagram of the proposed system.

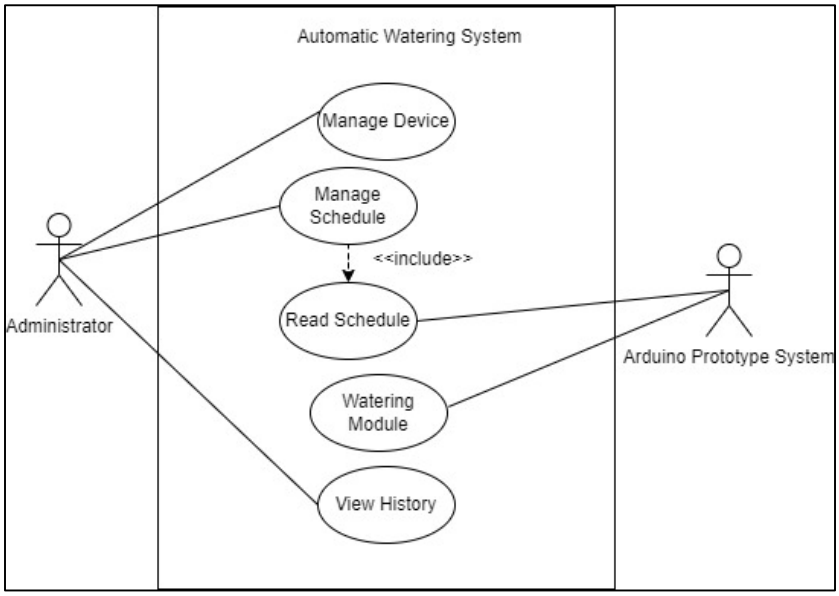

Figure 1: Use Case Diagram

3.5 Class Diagram

Class diagrams is a part of a unified modeling language (UML) that displays a software's static structure by showing the classes within a piece of software and the logical relationships between those classes [13]. It is used for both general conceptual modelling of the application's structure and detailed modelling, which involves turning the models into computer code. Data modelling can also be done with class diagrams. It usually depicts the classes in a system, their attributes and activities, as well as the relationships between them. The suggested system's class diagram is shown in Figure 4.2.

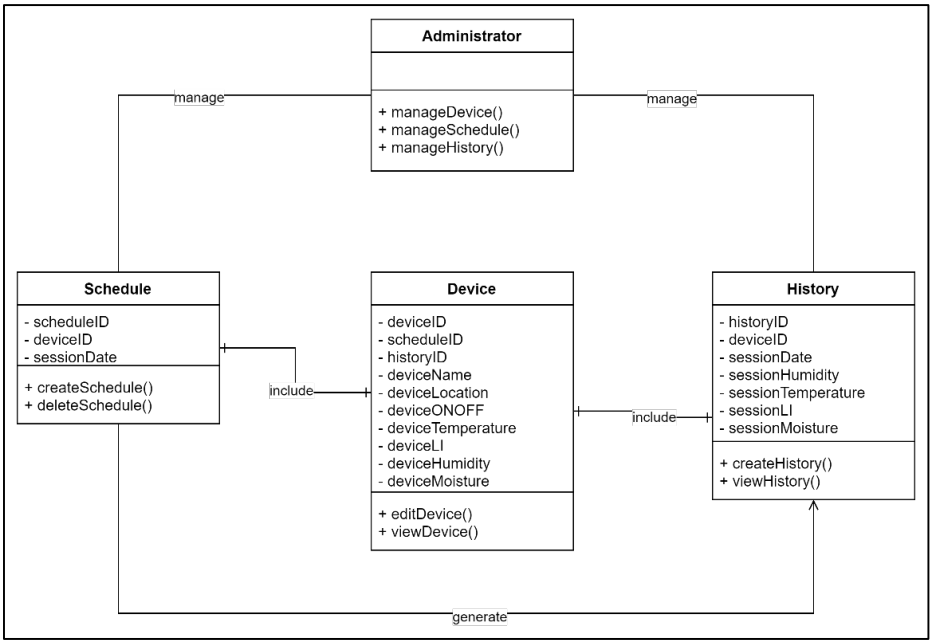

Figure 2: Class Diagram

3.6 Swimlane Diagram

The swim lane describes the overall process of the system from registration until logging out the system. Figure 3 describes the system flowchart.

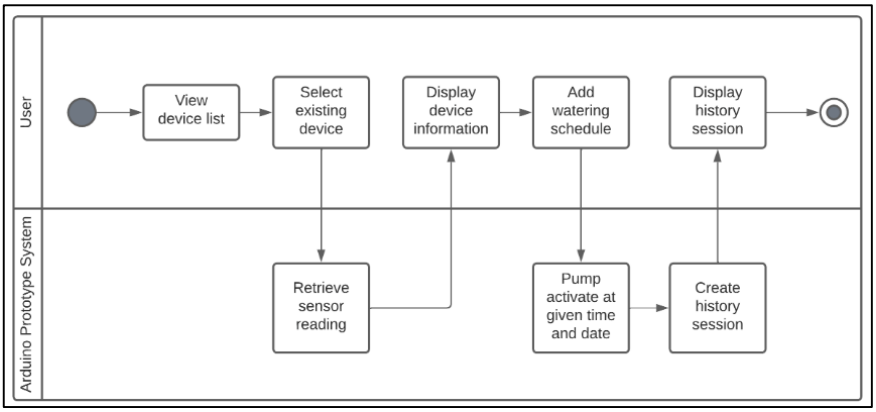

Figure 3: Registration and login flow chart

3.7 System Design

The process of defining a system's architecture, product design, modules, interfaces, and data to meet defined requirements is known as systems design. The system architecture describes the main components, their relationships (structures), and how they interact. The design architecture of the proposed system's users is shown in Figure 5.

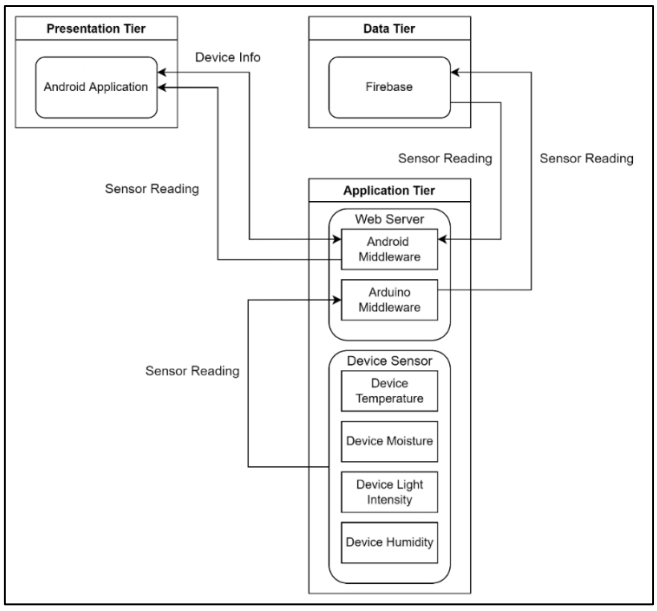

Figure 4: System Design Architecture

3.8 Database Design

A data dictionary, or metadata repository, as defined in the IBM Dictionary of Computing, is a "centralized repository of information about data such as meaning, relationships to other data, origin, usage, and format"[14]. In another word, data dictionary is a list of names, definitions, and properties for data components in a database or information system. Attributes, data type, size, and data explanation are all included in this system's data. Every module in this system has its own set of data dictionaries. A database schema is a representation of how system data is stored in a database. It describes how data is organized and how tables in a database are related.

The database scheme is listed as follows:

i. `Schedule (scheduleID, deviceID, time, sessionDate)`

ii. Device (deviceID, scheduleID, historyID, deviceName, deviceLocation, deviceTemperature, deviceHumidity, deviceLI, deviceMoisture, deviceONOFF)

iii. History (historyID, deviceID, sessionDate, sessionHumidity, sessionTemperature, sessionLI, sessionMoisture)

4. Result and Discussion

The result of the system will be discussed in this section. All the analysis and data presentation of Automatic Watering System for Southern Sissoo will be available here. The objective of the project is to develop an automatic watering system using Arduino and Android based and to test the developed system. In addition, the scope of the project specifically stated that the system will be develop using Dart programming language (Flutter framework) and the system will only has one user which was the system administrator which include device information module, watering schedule module and session history module. Aside from using the Flutter framework to develop the application, the Arduino prototype system was built using Arduino IDE which was used to encode the microcontroller using C++ and C# language. The Arduino prototype model are also included with variety of Arduino hardware such as sensors. Moreover, Google Cloud Firebase act as a cloud database both for the Android application and Arduino prototype model. Overall, the results of test cases for this system shows that the system has been successfully developed.

4.1 System Implementation

A few prototypes of the system had been developed to acquire high user satisfaction from the user of the system. The final prototype of the system will be the final product of this project. Dart is the main programming language that has been used to develop the Android application while Google Cloud Firebase, a NoSQL cloud database is used to store live sensor reading from the Arduino model as well as to store various related data. The system contains several modules such as device information modules, session history modules and watering schedule modules.

The flow of the system (Android application) starts when user launch the application. The system will only have one user, the farm owner. Therefore, only he can access the system and no login require to use the system. By default, when user launch the system, user will directly be navigated to the dashboard of the system which included in the device information module. Figure 5 shows the device dashboad, user will be able to add new device by tapping the '+' floating button and delete existing device via holding the card of the existing device

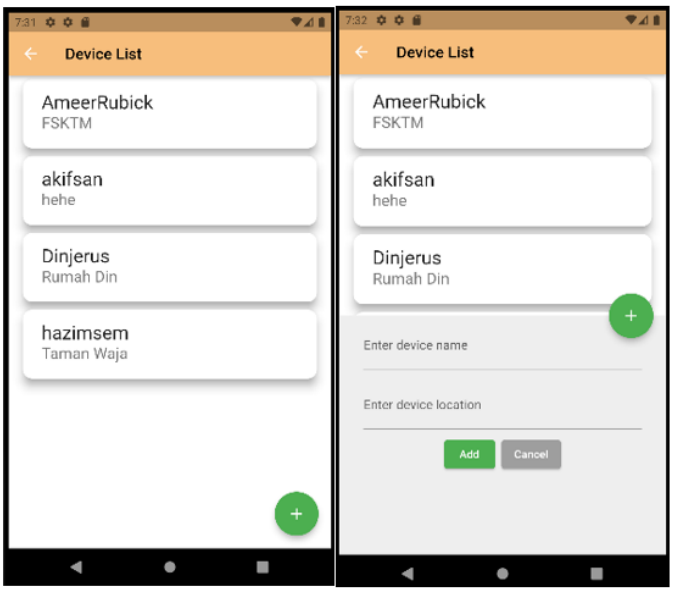

Figure 5: Device Dashboard

When user click on any of the device, user will be navigated to that device information page which was shown in Figure 6. In the device information page, all information such as device location and name as well as the live reading of the sensor can be read here. User can also edit the name and location of device and update it to Firebase via the save button. The sensor live reading will be updating every few seconds.

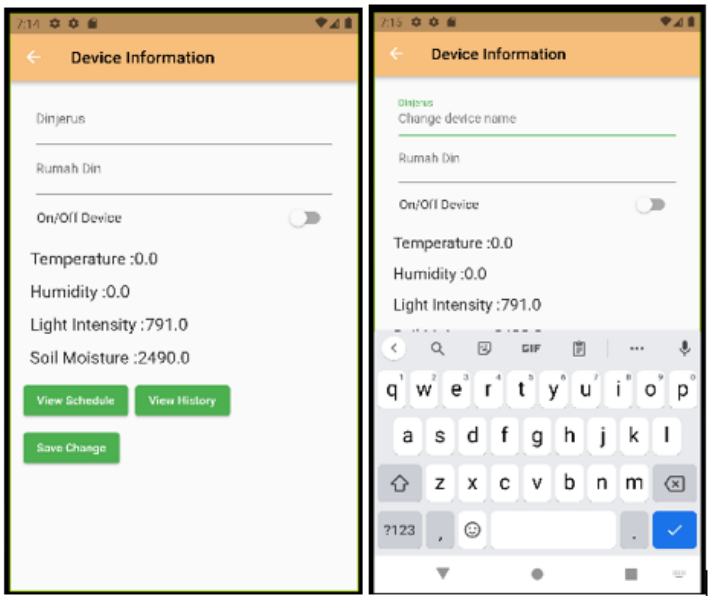

Figure 6: Device Information Interface

When user press 'schedule' button from device information page, user will be landed on watering schedule dashboard which was shown in Figure 7. From here, all schedule list for that device will be displayed. Every device will have unique schedule and will not conflicted with one another.

Figure 7: Schedule Dashboard

Figure 8 shows how to user can add new schedule. User can just simply tap '+' floating button and will be asked to choose date and time for the watering schedule. User can also delete existing schedule via tapping on any of the schedule on the schedule dashboard.

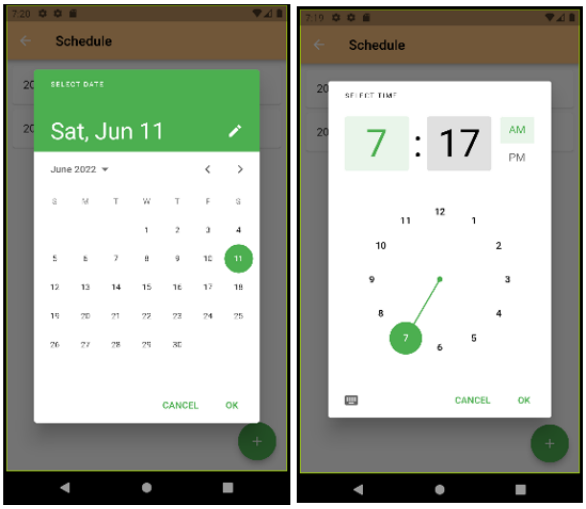

Figure 8: Add New Schedule Interface

Figure 9 shows the History dashboard. When user press 'history' button from device information page, user will be landed on Session History dashboard. From here, all watering session that had been take place for that device will be displayed. Every device will have unique history and will not conflicted with one another. To add, every session history will have their own unique id as well as information. If user tap on any of the history card, user will be landed on history information page.

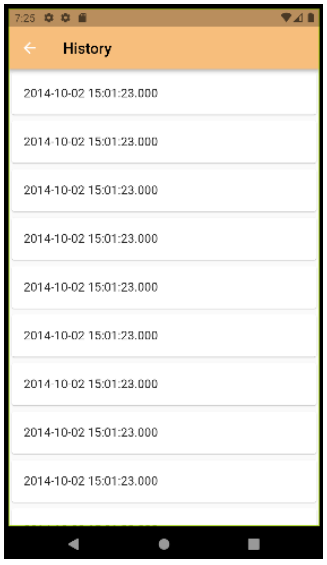

Figure 9: History Dashboard

Figure 10 shows the history information page. All information regarding the watering session such as the date and the time the session take place as well as the sensor reading during the watering session will be display here.

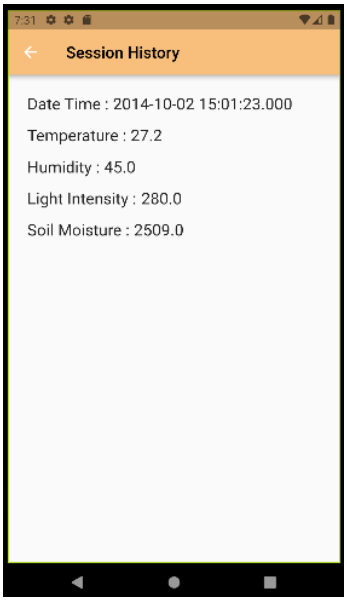

Figure 10: History Information Interface

For this project, an Arduino Model was built from scratch and throughout the project, many prototypes version had been made with more and more hardware being added. Below is the Arduino hardware that had been used to achieve the objective of this project. Figure 11 shows the actual Arduino Prototype System.

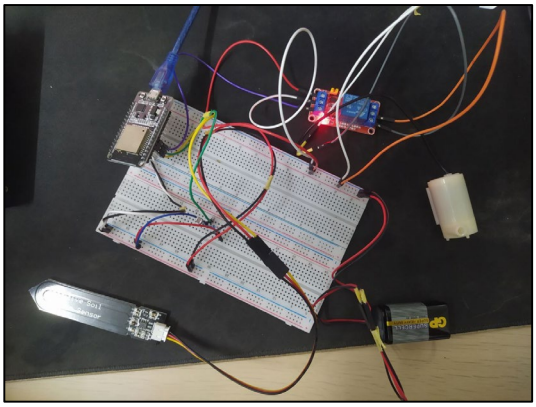

Figure 11: Arduino Prototype System

Every sensor that was used to acquire data for user will be save and store in Google Firebase. The Android application was used to retrieve and display all the data to the user. Sensor that had been used are temperature sensor, soil moisture sensor, humidity sensor and light intensity sensor. Figure 12 shows the temperature and humidity sensor (BME280).

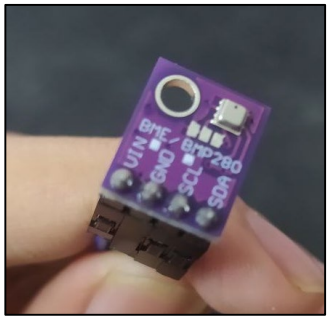

Figure 12: BME280

Figure 13 shows the soil moisture sensor that was used during the project development.

Figure 13: Soil Moisture

Figure 14 shows the light dependent resistor which was used as to detect the intensity of the light (Light Intensity Sensor).

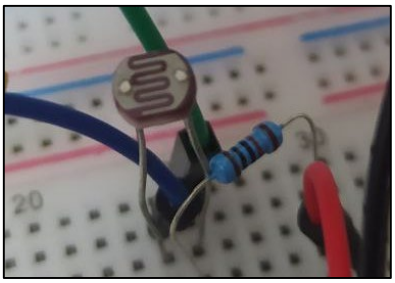

Figure 14: Light Dependent Resistor

Figure 15 shows the schematic diagram of the Arduino Prototype System.

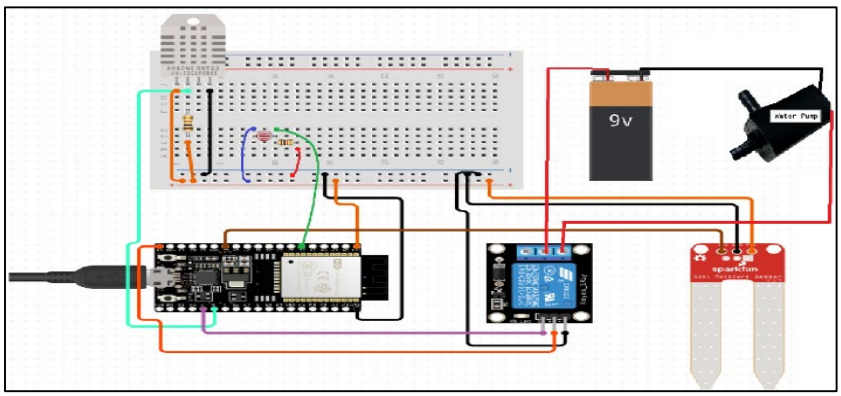

Figure 15: Schematic Diagram

4.2 System Testing

A few prototypes of the system had been developed to acquire high user satisfaction from the user of the system. The final prototype of the system will be the final product of this project. Dart is the main programming language that has been used to develop the Android application while Google Cloud Firebase, a NoSQL cloud database is used to store live sensor reading from the Arduino model as well as to store various related data. Table 7 shows the result for every test case of the system during system testing

Table 7: Test Cases and Result

No	Test Cases	Description	Result
TC_100 (MANAGE DEVICE)			
1.	TC_100_001	System displays all existing device from database.	Pass
2.	TC _100_002	System allow user to add new device into database.	Pass
3.	TC _100_003	System refresh device dashboard screen every time new device added.	Pass
4.	TC _100_004	Bottom sheet to add device will close down when 'Cancel' button pressed.	Pass
5.	TC _100_005	System allow user to delete existing device via holding on chosen device.	Pass
6.	TC _100_006	System will ask user confirmation to delete device.	Pass
7.	TC _100_007	System navigate user to unique device information page when select any device.	Pass
8.	TC _100_008	System displays all information for device in device information page.	Pass
9.	TC _100_009	System allow user to edit device name and location.	Pass
10.	TC _100_010	System allow user to turn device switch on and off.	Pass
11.	TC _100_011	System saves updated information to database when 'Save' button is pressed.	Pass
12.	TC _100_012	System refresh device information screen every time data in database changes.	Pass
TC_200 (MANAGE SCHEDULE)			
1.	TC_200_001	System displays all existing schedule from database	Pass
2.	TC_200_002	System displays only schedule for chosen device uniquely.	Pass
3.	TC_200_003	System allow user to add new schedule into database.	Pass
4.	TC_200_004	System will show Date and Time picker when user want to add new schedule.	Pass
5.	TC_200_005	System refresh screen every time new schedule added.	Pass
6.	TC_200_006	System allow user to delete schedule via tapping on chosen schedule.	Pass
7.	TC_200_007	System will ask user confirmation to delete schedule.	Pass
TC_300 (WATERING MODULE)			
1.	TC_300_001	Arduino model shall able to read schedule information.	Pass

Table 7: Test Cases and Result

No	Test Cases	Description	Result
TC_300 (WATERING MODULE)			
2.	TC_300_002	Arduino model shall able to initiate watering for scheduled watering session	Fail
3.	TC_300_003	Arduino model shall able to initiate watering for automatic (conditional) watering session	Pass
4.	TC_300_004	Arduino model shall able to create new history session after a watering session take place.	Pass
5.	TC_300_005	Arduino model shall able to send newly created history session to Firebase database	Pass
TC_400 (READ HISTORY)			
1.	TC_400_001	System displays all existing history from database	Pass
2.	TC_400_002	System displays only history for chosen device uniquely.	Pass
3.	TC_400_003	System will navigate user to history information screen via tapping any chosen history.	Pass
4.	TC_400_004	System will display history detailed information.	Pass

4.3 User Acceptance Testing

Result for every test case can be simplify in the bar graph as shown below. These results are crucial to make sure which function has high level of failure so that it can be overcome immediately.

Figure 16, Figure 17, Figure 18 and Figure 19 shows the bar graph of every test case result respectively.

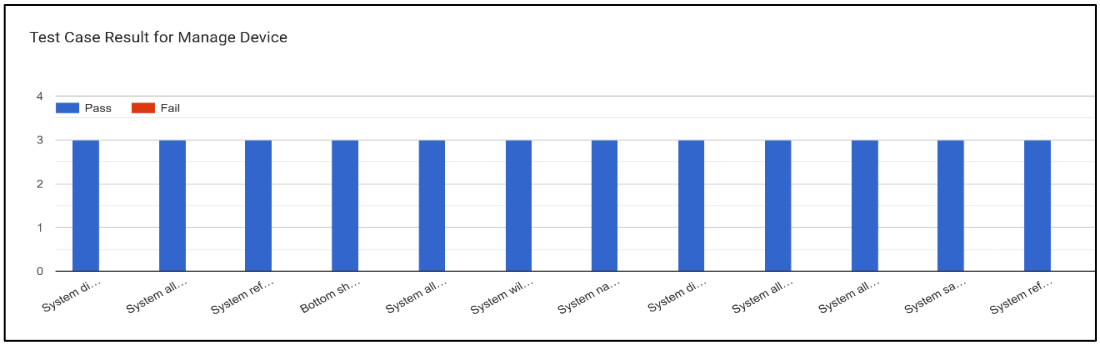

Figure 16: Bar Graph of Test Case Result for Manage Device Use Case

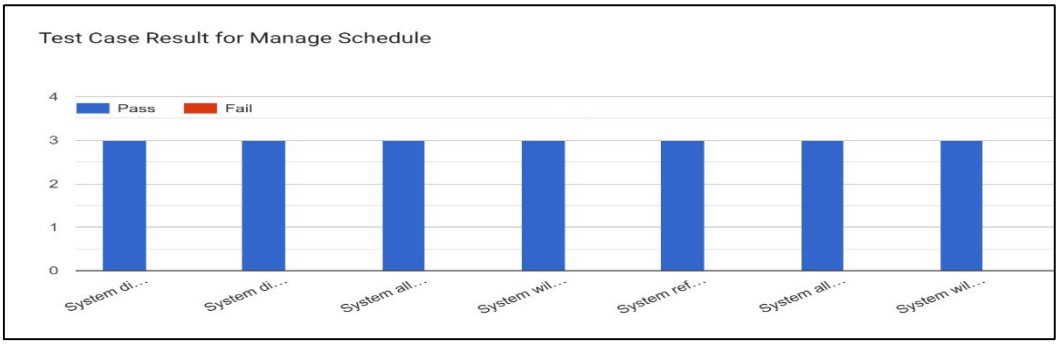

Figure 17: Bar Graph of Test Case Result for Manage Schedule Use Case

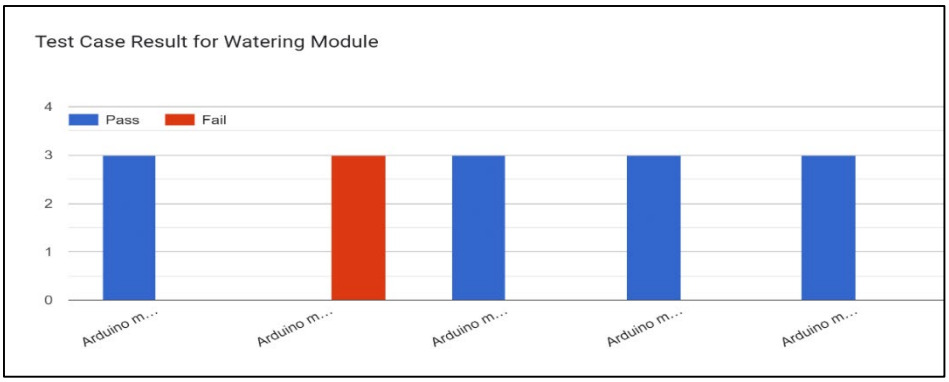

Figure 18: Bar Graph of Test Case Result for Watering Module Use Case

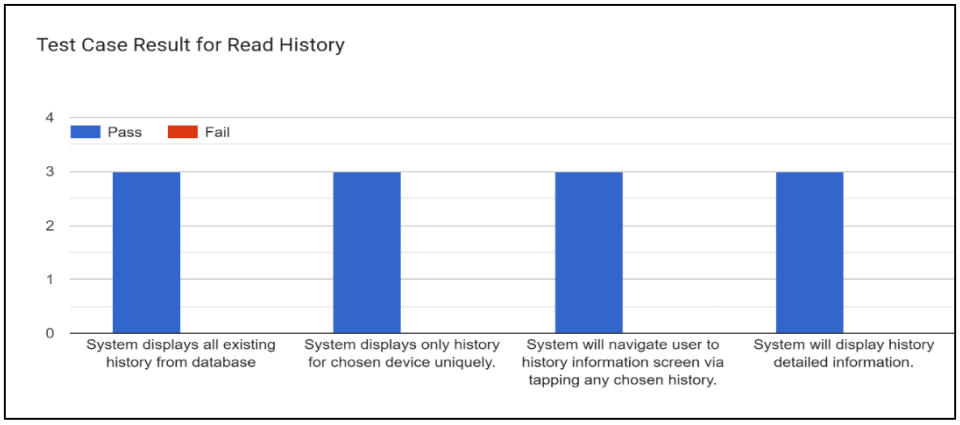

Figure 19: Bar Graph of Test Case Result for Read History Use Case

5. Conclusion

As a conclusion, the problem that Southern Sissoo is currently encounter can be overcome by developing the system. The objective of this project which is to develop the prototype of Automatic Watering System Using Arduino and Android Based for Southern Sissoo can improve the process of watering session and schedule to become smoother. Administration does not require to instruct worker to manually water the crop can be water remotely without having anyone available at the farm. Administrator can also view the current information at the farm and its history such as temperature, humidity, soil moisture and etc. The project succeeds in almost all of the test cases except when initiating scheduled watering session that user had created. For future improvement, the system should able to initiate watering system both from schedule session and automatic watering (user give condition when to automatically initiate watering).

Acknowledgment

The authors would like to thank the Faculty of Computer Science and Information Technology, Universiti Tun Hussein Onn Malaysia for its support

References

[1] K. K. Monisha, "Smart irrigation system using Arduino Uno," 2018.

[2] Dictionary. (2021). Retrieved December 27, 2021, from www.dictionary.com website: https://www.dictionary.com/browse/irrigation

[3] S. Panigrahi and A. Thakur, "Modeling and simulation of three phases cascaded H-bridge grid-tied PV inverter," Bulletin of Electrical Engineering and Informatics, vol 8, no. 1, pp 1-9, 2019.

[4] A.K. Kasim , A. Raheem (2017). Bluetooth based smart home automation system using Arduino UNO microcontroller, Al-Mansour J. 27 139 .

[5] H., Z., A., H., & M., M. (2015). Internet of Things (IoT): Definitions, Challenges and Recent Research Directions. International Journal of Computer Applications, 128(1), 37–47. https://doi.org/10.5120/ijca2015906430

[6] Kushner, D. (2011, October 26). The Making of Arduino. IEEE Spectrum; IEEE Spectrum. https://spectrum.ieee.org/the-making-of-arduino#toggle-gdpr

[7] Arduino - Introduction. (2021). Arduino.cc. https://www.arduino.cc/en/guide/introduction

[8] Martin, M. (2021, November). Prototyping Model in Software Engineering: Methodology, Process, Approach. Guru99. https://www.guru99.com/software-engineering-prototyping-model.html#:~:text=In%20Software%20Engineering%2C%20Prototype%20methodology,an%20acceptable%20prototype%20is%20achieved.&text=Regular%20meetings%20are%20essential%20to,costly%20delays%20in%20prototyping%20approach

[9] Techopedia. (2015, May 11). System Requirements. Techopedia.com; Techopedia. https://www.techopedia.com/definition/4371/system-requirements

[10] Fulton, R., Vandermolen, R., & Rtca (Firm. (2015). Airborne electronic hardware design assurance : a practitioner's guide to RTCA/DO-254. Crc Press, Taylor & Francis Group.

[11] Rome, P. (2020). What are Non Functional Requirements — With Examples | Perforce Software. Perforce Software. https://www.perforce.com/blog/alm/what-are-non-functional-requirements-examples

[12] Rolland, C., & Achour, C. B. (1998). Guiding the construction of textual use case specifications. Data & Knowledge Engineering, 25(1-2), 125–160. https://doi.org/10.1016/s0169-023x(97)86223-4

[13] Fauzan, R., Siahaan, D., Rochimah, S., & Triandini, E. (2021). Automated Class Diagram Assessment using Semantic and Structural Similarities. International Journal of Intelligent Engineering and Systems, 14(2), 52–66. https://doi.org/10.22266/ijies2021.0430.06

[14] Mcdaniel, G., & Business, I. (1994). IBM dictionary of computing. Mcgraw-Hill.