International Journal of Academic Information Systems Research (IJAISR)
ISSN: 2643-9026
Vol. 6 Issue 6, June - 2022, Pages: 1-9

Android Application Design for Smart Greenhouse Controller

Aghus Sofwan[1], Sumardi[2], Rafi Fistra Ali[3]

Department of Electrical Engineering, Diponegoro University, Semarang, Indonesia
aghus.sofwan@gmail.com[1], sumardi@elektro.undip.ac.id[2], rafistraali@gmail.com[3]

Abstract: *Indonesia is an agricultural country that produces various kinds of agricultural products. With a land area of 190 million hectares, Indonesia has high opportunities in agriculture. One of the agricultural products that can take advantage of the vast potential of land in Indonesia is kale. Kangkung(Ipomoea aquatica) is a plant that is quite easy to cultivate. One way to cultivate kale is to use a greenhouse. Greenhouses are widely used in agricultural systems because the environmental conditions around plants can be adjusted, so plants can grow and develop ideally. To maintain optimal conditions for plant growth and development, an intelligent control system is needed in a smart greenhouse. In designing this intelligent control system based on the Internet of Things. Smart greenhouses can be observed and controlled remotely through devices with the Android operating system, which are flexible and have high mobility. This design has the result that, greenhouses can be observed and controlled remotely, thereby increasing user mobility in the agricultural world. The performance of the application created is influenced by the specifications of each device, such as memory and processor capacity.*

Keywords—Smart Greenhouse, Internet of Things, Android

1. INTRODUCTION

Today's information and communication technology has rapid development, intelligent systems are widely used in various fields. The Internet of Things is an example of evidence of the rapid development of information and communication technology. It can make a significant contribution in various sectors, one of which is agriculture. Smart farming is currently a trend in agricultural development. Intelligent systems in the Internet of Things that are applied in smart farming systems can provide convenience in getting better crop yields[1]. Smart greenhouse is one of the implementations of the previously mentioned smart farming system. Smart greenhouse is also a place to combine traditional farming models with information technology. This smart greenhouse is a very precise representation of a smart farming system. One of the important elements in a smart greenhouse is a flexible monitoring and control system, meaning that the system has mobility and is easy to use[2]. Applying Android with a smart greenhouse system is a solution to be able to observe and control the variables that affect plant growth in order to get optimal results and reduce farmer/labor costs[3].

The popularity of Android and the open source nature of Android allows it to be developed by anyone. The use of Android applications can help farmers to observe greenhouse conditions in real time and remotely so as to minimize crop failure due to uncontrolled environmental conditions [4]. The environmental conditions observed included temperature, humidity, soil moisture, light intensity, and soil pH.

To assist farmers in realizing effective agriculture and combining technology in greenhouses, so that in this project, a smart greenhouse remote monitoring and control system will be designed using an Android-based application. The data observed were air humidity, air temperature, soil moisture, and soil pH. In addition to observations, there is also a control system embedded in the Android application to control the actuator in the greenhouse. This system uses a Firebase database for data storage, so that the observation and control through mobile devices will be carried out in real time.

2. SYSTEM DESIGN

2.1 System Description

The design of the smart greenhouse application is composed of several parts, namely needs analysis, database design, and Android application design. The parameters of success in the design of this project include the system can display observation data in the Android application and can provide remote manual mode control to control the actuator. Fig. 1 shows a block diagram of the overall smart greenhouse system design.

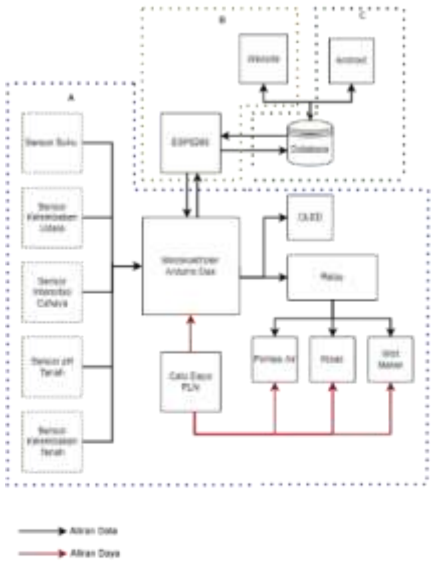

Fig. 1. *Smart Greenhouse System Planning Block Diagram*

This project system specification focuses on observing and controlling smart greenhouses through an Android application. This Android application displays various information regarding greenhouse conditions, remote actuator control, actuator status, and greenhouse data recording. In addition, this application can also display weather forecasts to ensure the condition of the greenhouse in the future. Android Studio was used for the creation of this application.

2.2 Functional Requirement

Functional requirements are a description of each function that can be performed by the system, the functional requirements of this sub system are as follows [5]: users can find out the condition of the greenhouse in real time, users can control the actuator greenhouse remotely, and users can view recorded data in the greenhouse.

2.3 Non-Functional Requirement

Non-functional requirements are system requirements such as performance, completeness of the operation of existing functions, and changes in the user environment [5]. Non-functional requirements contain many requirements that help functional requirements, some formulations of non-functional requirements are:

- Operational Requirement
  a. The application must be able to be installed on the Android device.
  b. This sub system can only be accessed through files with .apk format that have been installed on Android devices.
  c. The user interface is designed to be as comfortable and friendly as possible in its use.
  d. The minimum operating system in order to run the application is Android version 10.
  e. The application uses the Java language in its programming.
- System Performance

  The sub-system created is a mobile device application. Even though the mobile device is on the Android operating system, there are some limitations that need to be considered. Thus, many factors must be considered as a reference in system development, the appearance of the interface affects the waiting time required before the application is ready to be used, the more components used in the Android interface, the waiting time is also getting longer.

Several alternatives are proposed to support the performance of applications with limitations on Android, including: designing applications that make the best use of resources while maintaining application functionality and performance and creating a program with a simple user interface but attractive and easy to use.

2.4 Database Design

In this smart greenhouse system, the NoSQL Firebase database is used as data storage and is real time. The following is a data structure for storing greenhouse condition data captures sent from the microcontroller.

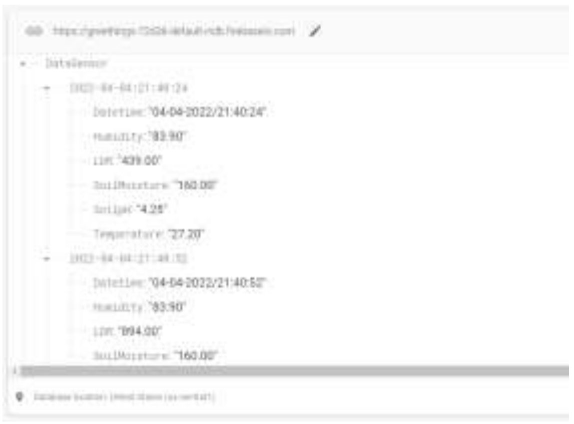
```
DataSensor
  2022-04-04|21:40:24
    Datetime: "04-04-2022/21:40:24"
    Humidity: "83.90"
    LDR: "439.00"
    SoilMoisture: "160.00"
    SoilpH: "4.28"
    Temperature: "27.20"
  2022-04-04|21:40:52
    Datetime: "04-04-2022/21:40:52"
    Humidity: "83.90"
    LDR: "894.00"
    SoilMoisture: "160.00"
```

Fig. 2. *Database Structure*

In Fig. 2 it can be seen that there are several data sent from the microcontroller, namely Datetime, Humidity, SoilMoisture, SoilpH, and Temperature. Key Datetime is used for time and calendar, Humidity is used to store humidity data, SoilMoisture is used to store soil moisture, SoilpH to store soil pH, and Temperature to store air temperature data in the greenhouse.

This real time database is not only used to store data on greenhouse conditions, it is also used to store the status of the actuator. The following is a real time database structure to store the actuator status.

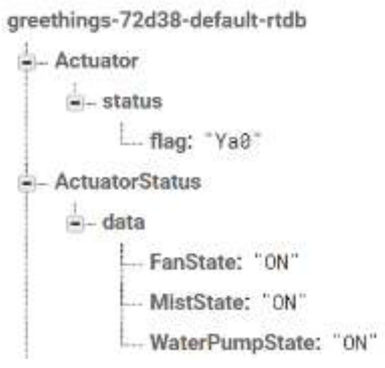
```
greethings-72d38-default-rtdb
  Actuator
    status
      flag: "Ya8"
  ActuatorStatus
    data
      FanState: "ON"
      MistState: "ON"
      WaterPumpState: "ON"
```

Fig. 3. *Various Conditions on Hardware*

In Fig. 3, there are two children for storing the actuator state. Child Actuator is used to store commands from Android and web applications.

Table 1: Command in Controlling the Actuator

Flag Command	Command

Ya0	Automatic mode
Ya110	Manual mode water pump state ON
Ya111	Manual mode water pump state OFF
Ya120	Manual mode mist maker state ON
Ya121	Manual mode mist maker state OFF
Ya130	Manual mode fan state ON
Ya131	Manual mode fan state OFF

Table 1 is the commands used in controlling the actuator. In Fig. 3, there is also a child ActuatorStatus which is used to determine the current state of the Actuator, which will later be displayed in the Android application and the web.

The smart greenhouse application system is equipped with an authentication feature by utilizing the Firebase Authentication feature. This feature is used as security in using the application where users are required to login first when using the application. In Firebase Authentication there is user table where the contents are a list of user accounts that will be used to log in to the application. The following is a view of the users table in Firebase Authentication.

Fig. 4. *Authentication Via E-Mail*

From Fig. 4 it can be seen that the users table consists of Identifier, providers, Created, Signed In, and User UID. Identifier is the e-mail from the user, provider is the login method used, created describes when the user was created, Sign In describes the last time users logged in, and User UID is the ID of each user. This table will be used by the system to verify the user's email address and password when logging in.

2.5 Application Design

- Use Case Diagram

In designing the application, it is necessary to know what activities can be carried out by the user when using the application. The use case diagram provides an explanation of this. This diagram also provides an explanation of the features in the application from the user's point of view[6].

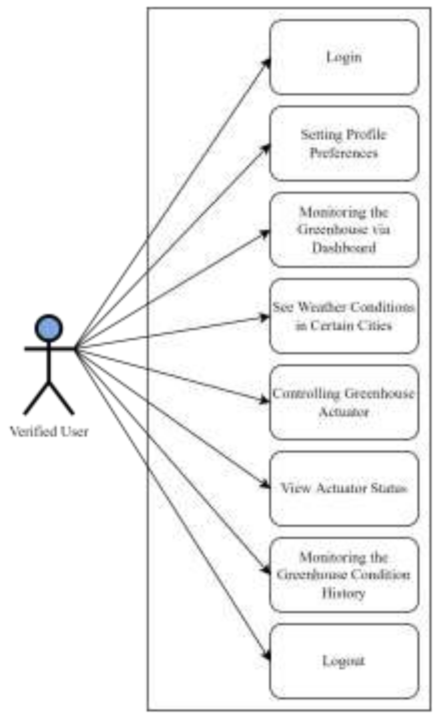
Login
Setting Profile Preferences
Monitoring the Greenhouse via Dashboard
See Weather Conditions in Certain Cities
Controlling Greenhouse Actuator
View Actuator Status
Monitoring the Greenhouse Condition History
Logout
Verified User

Fig. 5. *Use Case Diagram*

Fig. 5 shows the features accessible to verified users. Users can use the features to log in to an account, set profile preferences, observe greenhouse conditions through the dashboard, view weather conditions in certain cities, control greenhouse actuators, view greenhouse condition history, and log out.

- Activity Diagram

Activity diagrams are used to show how the application flows from one activity to the next [7]. The procedural logic and workflow of the system created are depicted in activity diagrams. Furthermore, activity diagrams can be used to illustrate parallel operations that occur in multiple executions. To separate the activity responsibilities of each item, the activity diagram can be divided into swimlane objects.

a. Main Activity Diagram

The first time the user uses the app is shown in this activity diagram. The splash screen and main menu will be visible to the user.

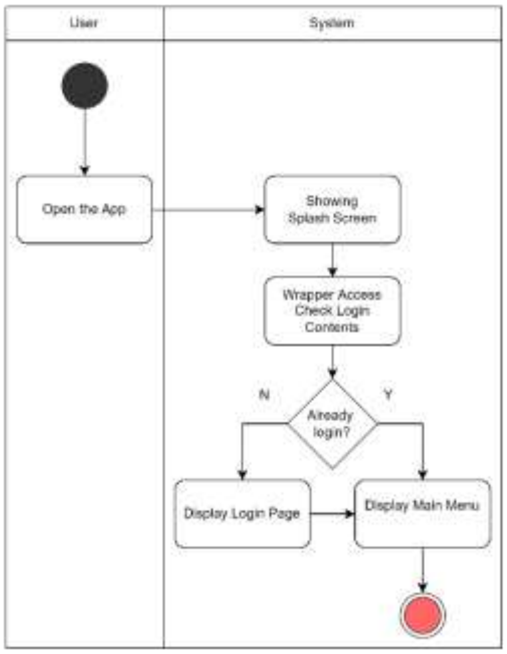

Fig. 6. *Main Activity Diagram*

Fig. 6 is a diagram of the main activity when the user starts using the application. The first time the system will display the activity splash screen. After that the system will check whether the user has logged in on the same device or not. If the user has not logged in, the system will direct the user to the login activity, otherwise if the user has previously logged in, the system will immediately direct the user to the main menu activity.

b. Login Menu Activity Diagram

Fig. 7 shows user activity when first starting the application and logging in. Users will see a splash screen and login menu.

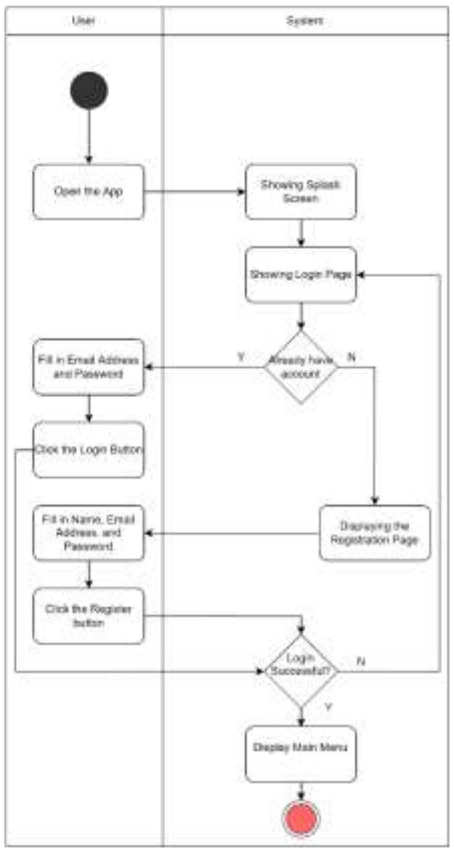

Fig. 7. *Login Menu Activity Diagram*

Fig. 7 is an activity diagram for login and registration. After the splash screen, the login activity will appear. If the user already has an account, the user can immediately fill in the email address and password, after the system authenticates, if it is wrong it will stay in the login activity, if it is true the system will redirect to the dashboard fragment on the main activity. If the user does not have an account yet, then the user must create an account first, by pressing the registration button. In activity registration, users must enter their full name, email, and password. After successful registration, the user will be directed to the fragment dashboard on the main activity.

c. Main Menu Activity Diagram

The main menu activity diagram shows the user's activity while on the main menu. Users can see weather conditions by selecting the city first, but if the city is not selected then the system will automatically retrieve weather condition data in the city where the device is located. After that, the user can see several greenhouse nodes, where each node briefly contains the current greenhouse condition. When the user selects a node, the user will be taken to the dashboard of the selected greenhouse node.

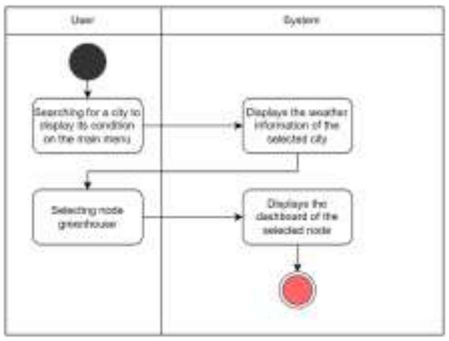
User
System
Searching for a city to display its condition on the main menu
Displays the weather information of the selected city
Selecting node greenhouse
Displays the dashboard of the selected node

Fig. 8. *Main Menu Activity Diagram*

d. User Activity Diagram

The user activity diagram shows when the user is on the user page. On this page, the user can change the user account password. When the user has changed the password, the user will be directed to the login page to log in again. On the user page, the user can also log out to end the session. Fig. 9 is an activity diagram on the user menu.

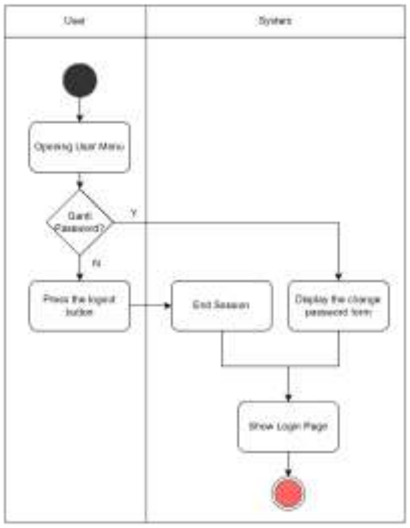

Fig. 9. *User Activity Diagram*

e. Main Activity Diagram

This activity diagram depicts the user's activity while in the Main Activity. Users will be faced with the dashboard fragment for the first time, then can move to other fragments, namely fragment control or fragment history. Fig. 10 is a diagram of the main activity.

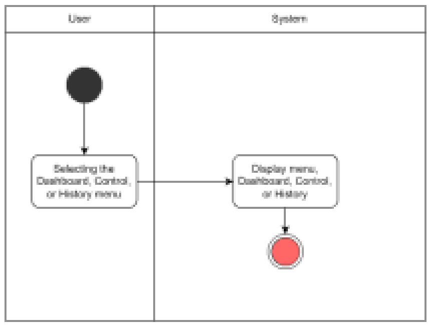
User
System
Selecting the Dashboard, Control, or History menu
Display menu, Dashboard, Control, or History

Fig. 10. *Main Activity Diagram*

f. Dashboard Fragment Activity Diagram

The activity diagram on the dashboard fragment can be seen in Fig. 11.

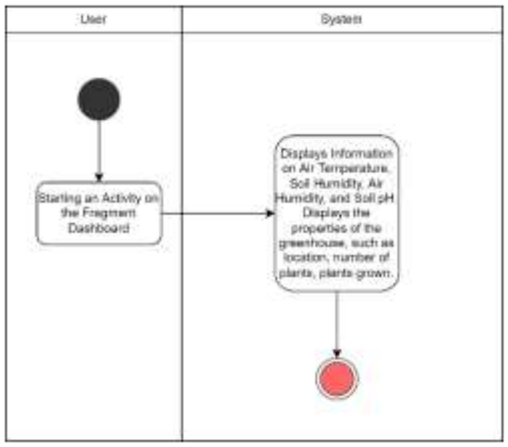
User
System
Starting an Activity on the Fragment Dashboard
Displays Information on Air Temperature, Soil Humidity, Air Humidity, and Soil pH Displays the properties of the greenhouse, such as location, number of plants, plants grown.

Fig. 11. *Dashboard Fragment Activity Diagram*

In this fragment, the user will display information in the form of air temperature, soil moisture, air humidity, and soil pH in the greenhouse. In addition to this information, users can also view other information such as the number of plants, the plants being planted, and the location of the greenhouse.

g. Control Fragment Activity Diagram

This activity diagram illustrates user activity while in fragment control to control the actuator. Users will see a display of several buttons used to control the actuator. The fragment control activity diagram is shown in Fig. 12.

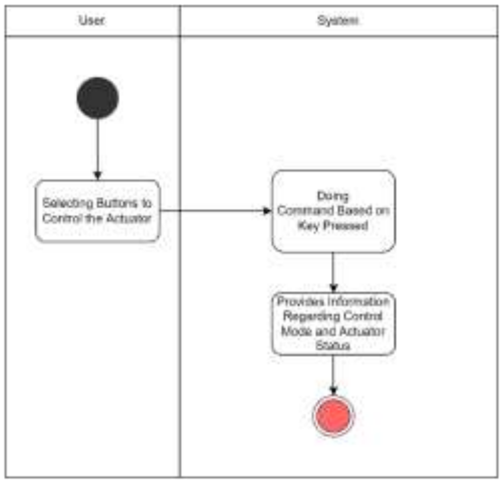
User
System
Selecting Buttons to Control the Actuator
Doing Command Based on Key Pressed
Provides Information Regarding Control Mode and Actuator Status

Fig. 12. *Control Fragment Activity Diagram*

In Fragment Control there are several buttons to control the actuator in the greenhouse. First, there are control mode buttons to select control automatically or manually, then there are ON/OFF buttons for each actuator when the control mode is selected in manual mode.

h. History Fragment Activity Diagram

This activity chart depicts user activity while in the fragment history. Fig. 13 is an activity diagram on the fragment history. The user can select the button to display the history of the data in the form of a table.

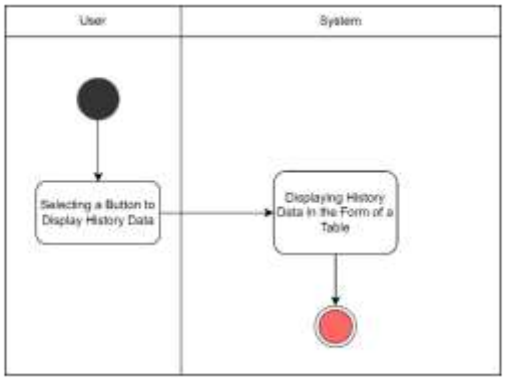
User
System
Selecting a Button to Display History Data
Displaying History Data in the Form of a Table

Fig. 13. *History Fragment Activity Diagram*

- User Interface Design

This activity design has a function to create the interface of the Android-based Smart Greenhouse application. The design is done in the Layout folder inside the Res folder. The user interface is created using XML, or you can use the design features included in Android Studio. The main function is then built in the Java sub directory. Here is the interface design for this application.

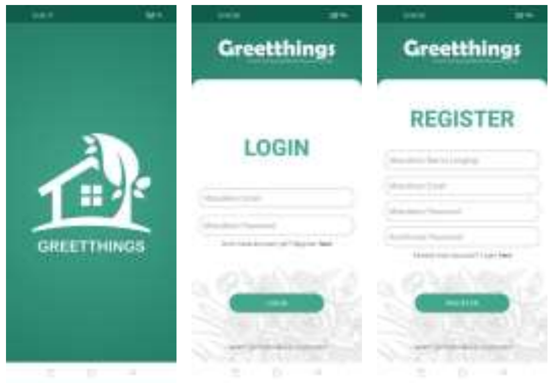

Fig. 14. *Splash Screen, Login Menu, and Register Menu Interface Design*

Fig. 14 shows the splash screen, login, and register display. The Splash Screen itself is just the application logo. The contents of the Login Menu display itself in the form of a form used to enter an email address and password. In addition, there is also a login button to process the Login Menu. To display the Registration Menu itself is a form used to create a new account by entering your full name, email, and password. In addition, there is also a register button to process the Registration Menu.

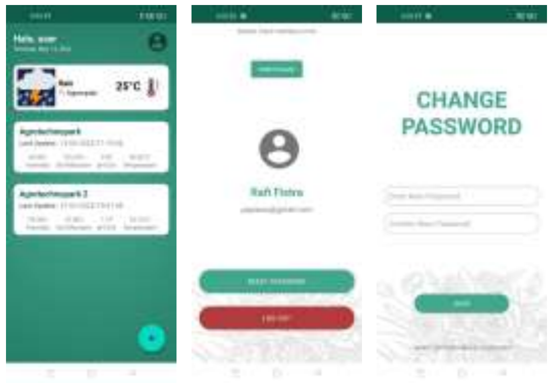

Fig. 15. *Main Menu, User Menu, and Change Password Interface Design*

Fig. 15 shows the main menu interface, user, and change password. In the main menu there are weather conditions in the city that can be determined by the user and a menu for selecting nodes for a greenhouse. In addition, there is also a profile button to direct the user to the activity profile/user. The current User Interface has a feature to change the password. The password button when pressed will direct the user to a page to change the password. In this interface there is user information, namely the user's name and email. In addition, in the user interface there is also a logout button, which is used by the user to exit out of the account from the application. The change password interface contains a form to update the user account password. There are two lines for entering a new password, namely entering a new password and confirming the password. Password confirmation is used to ensure that the entered password is correct.

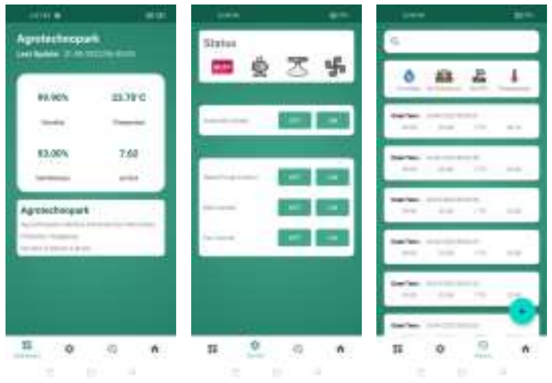

Fig. 16. *Dashboard, Control, and History Interface Design*

Fig. 16 is the dashboard, control, and history interface. In this dashboard design, it contains the conditions of the greenhouse that have been selected in the previous menu. Users can see the condition of the greenhouse and its attributes. The control design contains a display for controlling the smart greenhouse. Fig. 16 shows that this interface contains information on the status of the actuator, whether the actuator listed is on or off. In this interface there are also buttons to turn on and turn off the actuator remotely. The history interface itself displays the history of the data that has been sent to Firebase. There is a list of data in the form of the time/date the data was sent, the value of air humidity, soil moisture, soil pH, and temperature.

3. Results and Testing

3.1 System Functionality testing

In this Final section, application testing is done through alpha testing, which involves testing the functionality of the application. Alpha testing is a method for detecting and eliminating problems before they reach the end user. Two devices were used in the alpha testing of this app. This is done to test the performance of the application on various device specifications. The following devices were used to perform the test:

- Realme 7, OS Android version 11, Layar 6.5 inch 2400x1080 FHD+, Prosesor Helio G95 Gaming Processor, GPU Mali-G76 MC4.
- Oppo A74, OS Android version 11, Layar 6.4 inch 2400x1080 FHD, Prosesor Qualcomm® Snapdragon™ 662 2GHz, GPU Qualcomm® Adreno™ GPU 610 pada 950MHz.

Blackbox testing approach is used to perform alpha testing on this application. The two main types of testing are testing Android app components built using Android Studio and testing response times.

Table 2: Android Application Functionality Testing Method

Testing Form	Success Indicator
Installing apps on Android Device	App icon appears on App drawer.
Showing the splash screen	A splash screen appears.
Show Login Menu	A login menu will appear and user can fill out a form in the form of an email and password.
Displaying the Register Menu	The registration menu will appear and you can fill in the registration form in the form of full name, email, and password.
Login	Main Menu appears after successful login.
Registration	Successfully registered by logging in.
Showing Main Menu	The Main Menu appears, showing the greenhouse nodes, each node appears brief information on the latest greenhouse conditions
Tap Navigation Bar	Move fragments according to the navigation bar that is tapped
Displaying the Dashboard Menu	A Dashboard Menu appears with information on the latest greenhouse conditions and information about the greenhouse that has been selected.
Showing the Control Menu	A Dashboard Menu appears with buttons for remote control, the current status of the actuator appears
Tap Each Control Button	The actuator status changes according to the button that has been tapped.
Show History Menu	The entire history of the condition data from the greenhouse appears
Perform a historical data search	The desired data can be searched according to the date keyword entered.
Displaying User Menu	The name of the user who is currently in the session appears.
Doing password updates	User passwords can be updated via the User Menu.
Logout	Successfully logged out of the current session.

From Table 2, perform the test using 2 devices that have different specifications, then compare them with the indicators listed in the table.

Table 3 below is the results of testing Android applications that have been compared and adjusted to the success indicators.

Table 3: Android Application Functionality Test Results

Testing Form	**Result**	
	Realme 7	***Oppo A74***
Installing apps on Android Device	Success	Success
Showing the splash screen	Success	Success
Show Login Menu	Success	Success
Displaying the Register Menu	Success	Success
Login	Success	Success
Registration	Success	Success
Showing Main Menu	Success	Success
Tap Navigation Bar	Success	Success
Displaying the Dashboard Menu	Success	Success
Showing the Control Menu	Success	Success
Tap Each Control Button	Success	Success
Show History Menu	Success	Success
Perform a historical data search	Success	Success
Displaying User Menu	Success	Success
Doing password updates	Success	Success
Logout	Success	Success

In Table 3 it is known for testing each feature of the application successfully/successfully on both devices. This is because there are no errors from the application program that has been made and each device meets the conditions needed for the application to run properly, namely a good internet network and has met the required specifications.

3.2 Activity Response Time Testing

Activity response time testing is done by measuring the delay time between one activity and the next. Seven trials were performed for this test, with the results shown in Fig. 17.

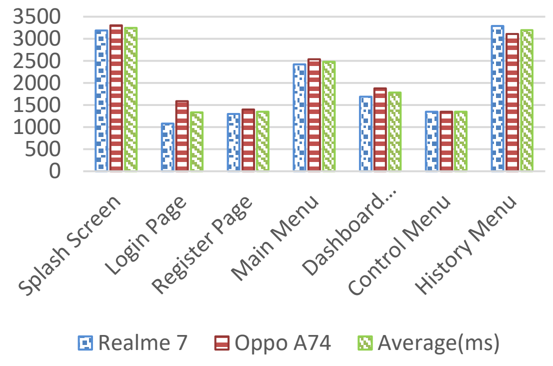

Fig. 17. *Activity Response Time Testing Results*

In Fig. 17 it can be seen that the longest response is the Splash Screen and the History Menu. For activities on the Splash Screen, it takes a long time because it has been determined how long it will take through the code that has been written, which is 3 seconds so that on any device it will take 3 seconds or more, depending on the specifications of the device. The History Menu also takes a long time because the activity on that page retrieves all the data in the database, so it depends on the quality of the internet network. The better the quality of the internet network, the faster the data will load.

Fig. 17 also shows that there is a difference in response time between the two devices, when performing various activities. The Realme 7 device has a faster response because it has higher specifications compared to other devices. The thing on the device that plays a role in making the difference is the CPU device.

4. CONCLUSION AND RECOMMENDATION

4.1 Conclusions

The conclusions obtained from the design, implementation, and testing of the system are as follows:

- In testing Android applications with the alpha testing method, 13 tests were carried out on 2 devices with different specifications. The test results show that 2 devices work well and are 100% successful in running each feature with predetermined success indicators.
- The activity response time test shows that the Splash Screen and History Menu are the features that have the highest response time compared to the others. The Splash Screen has a long response time because it has been determined and the time for the next activity is 3 seconds. While the History Menu has a long time to

respond because of the large amount of data loaded from the database and requires internet with good network quality to download data.

- The Login Page has the lowest response time, this is because the Login Page is not too heavy and is not timed to move to the next activity.
- Device specifications used in testing also have an influence on application performance. The higher the specifications and quality of the device, the faster the response time.
- In addition to the specifications of the device, the quality of the internet network also has an influence on the response between certain activities. Some activities require an internet connection, so the better the internet connection, the faster the response in the activity.

4.2 Recommendations

Further development of systems or similar research is recommended to improve:

- Added a notification feature under certain conditions, for example when the temperature is too hot, or other extreme conditions.
- Generate generic code for adding greenhouse nodes.

5. References

[1] [A. Sofwan, S. Sumardi, A. I. Ahmada, I. Ibrahim, K. Budiraharjo, and K. Karno, “Smart Greetthings: Smart Greenhouse Based on Internet of Things for Environmental Engineering,” in 2020 International Conference on Smart Technology and Applications (ICoSTA), Feb. 2020, pp. 1–5, doi: 10.1109/ICoSTA48221.2020.1570614124.

[2] N. Kitpo, Y. Kugai, M. Inoue, T. Yokemura, and S. Satomura, “Internet of Things for Greenhouse Monitoring System Using Deep Learning and Bot Notification Services,” in 2019 IEEE International Conference on Consumer Electronics (ICCE), Jan. 2019, pp. 1–4, doi: 10.1109/ICCE.2019.8661999.

[3] O. ALPAY and E. ERDEM, “Climate Control of an Smart Greenhouse based on Android,” in 2018 International Conference on Artificial Intelligence and Data Processing (IDAP), Sep. 2018, pp. 1–5, doi: 10.1109/IDAP.2018.8620803.

[4] Q. Xiaohui, D. Shangfeng, H. Yaofeng, and L. Meihui, “Development and design of mobile terminal APP for greenhouse environment control,” IFAC-PapersOnLine, vol. 51, no. 17, pp. 822–825, 2018, doi: 10.1016/j.ifacol.2018.08.093.

[5] L. Corbalán et al., “A Study of Non-functional Requirements in Apps for Mobile Devices,” 2019, pp. 125–136.

[6] E. Aquino, P. de Saqui-Sannes, and R. Vingerhoeds, “A Methodological Assistant for Use Case Diagrams,” in Proceedings of the 8th International Conference on Model-Driven Engineering and Software Development, 2020, pp. 227–236, doi: 10.5220/0008938002270236.

[7] T. Ahmad, J. Iqbal, A. Ashraf, D. Truscan, and I. Porres, “Model-based testing using UML activity diagrams: A systematic mapping study,” Comput. Sci. Rev., vol. 33, pp. 98–112, Aug. 2019, doi: 10.1016/j.cosrev.2019.07.001.