



Management Information System for Micro Climate logging data using REST API with Prototype Approach in a Coffee Plantation in UB Forest

Heru Nurwarsito

Department of Informatics Engineering, University of
Brawijaya
heru@ub.ac.id

Cahyo Prayogo

CEH Program, University of Brawijaya
c.prayogo@ub.ac.id

Yusuf Putra Al Azhar

Department of Informatics Engineering, University of
Brawijaya
yusufputra99@student.ub.ac.id

Simon Oakley

UK Centre for Ecology and Hydrology, Lancaster
Environment Centre
soak@ceh.ac.uk

ABSTRACT

Measured at 554 hectares, Brawijaya University's forest is located in Karangploso, Malang. The university intends to utilize the forest for education purposes and as part of its community service plan. The yield of the forest is Green Beans, which is one of the finest coffee bean variants from Indonesia. Meanwhile, as El-Nino and La-Nina phenomena greatly affect the global climate nowadays, the changes also bring impacts to the yield of coffee beans. Responding to this situation, UB Forest procures a tool to measure climate indicators, such as air temperature, soil temperature, and soil moisture. However, this tool does not have an integrated monitoring system (a standalone system). Consequently, the forest conservators have to inspect the forest in person to conduct climate monitoring, requiring them to go through mountainous terrains. By taking advantage of the rapidly developing technology, this research aimed to create a website-based microclimate monitoring system that can remotely control the sensors in coffee fields. Utilizing RESTful API, the system sends data through sensors and presents them in a website built with Laravel and ReactJS. The system was subjected to testing with 100% unit test results and a value of 80 for usability testing.

CCS CONCEPTS

• **Software and its engineering**; • **Software creation and management**; • **Software development techniques**; • **Software prototyping**;

KEYWORDS

Coffee, UB Forest, Microclimate, Sensor, REST API, Laravel, ReactJS

ACM Reference Format:

Heru Nurwarsito, Yusuf Putra Al Azhar, Cahyo Prayogo, and Simon Oakley. 2021. Management Information System for Micro Climate logging data using

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

SIET '21, September 13, 14, 2021, Malang, Indonesia

© 2021 Association for Computing Machinery.

ACM ISBN 978-1-4503-8407-0/21/09...\$15.00

<https://doi.org/10.1145/3479645.3479655>

REST API with Prototype Approach in a Coffee Plantation in UB Forest. In *6th International Conference on Sustainable Information Engineering and Technology 2021 (SIET '21), September 13, 14, 2021, Malang, Indonesia*. ACM, New York, NY, USA, 8 pages. <https://doi.org/10.1145/3479645.3479655>

1 INTRODUCTION

Indonesia ranks third in the world's biggest producers of coffee beans and is the home to 1.3 million coffee farmers. Coffee is the main source of livelihood for many global households and on average, women's participation as coffee laborers is 70% [1].

In 2018, coffee production in Indonesia grew 0.6% to 722,461 tons (preliminary figures) from the yield of the previous year. From this amount, 685,787 tonnes came from smallholder plantations, while 19,926 tonnes and 16,478 tonnes were the yields of large state plantations and large private plantations respectively [2].

UB Forest is an educational forest covering an area of 554 hectares on the slopes of Mount Arjuno, precisely in Sumbersari Hamlet, Tawang Argo Village, Karangploso, Malang Regency. The university declares that UB Education Forest serves as a part of the community service of the university's academics. UB Forest is expected to increase the productivity and production of the local community.

One of the commodities in UB Forest is various types of coffee beans, such as Green Beans, which is a superior coffee commodity, and coffee cherry. To ensure quality production, the coffee plantation management carefully controls the picking process, the planting location and neighboring plants (pine trees), and the climate monitoring.

The recent climate changes bring impact to all aspects of life, including the agriculture sector which relies heavily on the patterns of climate changes. As coffee is sensitive to climatic changes, irregular patterns of rain, drought, and air temperature result in the damage of coffee plants and the decrease of productivity. In Indonesia, El-Nino has caused prolonged dry months (rainfall below 60 mm per month) which decreased coffee production by 34.79%. Similarly, La-Nina also extended the period of wet months (rainfall above 100 mm per month) which resulted in coffee production decreased by 98.5%. Prolonged dry months (more than 3 months) cause the quality of coffee beans to decline. Every 1 degree Celsius temperature increase reduces coffee bean production by 30.04%. Furthermore, very low temperatures (3-5 degrees Celsius) can kill

coffee leaves [3]. To date, UB Forest does not possess a system that can monitor climate conditions in the area.

As the result of the progress made in science, technology inevitably develops and shapes the lives of modern humans. Various aspects of life now rely on technology, such as education, agriculture, environment, communication, society, health, infrastructure, and government. One of the advancements in technology that holds a crucial role is the ease of accessing information. In the modern era, the internet provides easy access to information utilizing websites. A large number of internet users indicates that the use of the internet in disseminating information is crucial. Research shows that the number of internet users in Indonesia has reached 171.17 million from 264.16 million of its population [4].

A cloud computing technology now allows a centralized system where the internet functions as the server for managing data and user applications. Cloud-based solutions are a cost-effective approach for most IT organizations that have budget constraints [5]. The application of cloud computing into a microclimate monitoring system through LoRa protocol can be the solution to improve the efficiency of the system data storage.

UB Forest has a microclimate monitoring system that is now under development. The cloud storage receives data from the microclimate monitoring system using the LoRa protocol. The approach simplifies the analysis of microclimate data sent from the sensors and also guarantees that the forest management and researchers can periodically access the microclimate data from UB Forest.

This research has designed an application for microclimate logging data management to monitor the distribution of microclimate conditions in UB Forest. This application connects to microclimate sensors that send data through LoRa protocol and therefore make it easy to monitor the conditions in the plantation to determine the exact planting period and estimate the yield. The system considers the fact that cloud computing attracts the attention of the industry and education world.

2 LITERATURE REVIEW

Karyati (2016) researched the fluctuations of the climate elements (air temperature, humidity, light intensity, and rainfall) in the education forest of the Faculty of Forestry, Mulawarman University, Samarinda, East Kalimantan. The research concluded that the existence of the forest created microclimate conditions that were different from the climate outside the forest. The average in-forest daily air temperature was lower than the outside, while the average outside-forest daily humidity was higher than the inside. Furthermore, the presence of vegetation affected the amount of light intensity that entered the forest floor. This research in microclimate fluctuations gives a foundation in the preliminary estimation of microclimate monitoring under certain conditions. The research objective is to provide inputs for the improvement of coffee yields in general.

A study by Merisa (2019) characterized the photodiode sensor and created a microclimate control and monitoring system using a DHT22 sensor, a photo-diode sensor, a YL-69 soil moisture sensor, an Arduino nano, and an NRF24L01 radio. The study tested and implemented the control and monitoring system in a miniature greenhouse with Cayenne chili. The results showed that the growth

of cayenne peppers in the controlled environment was improving than those without control. This research proves that coffee plants develop better in a well-controlled condition utilizing a data logging system.

2.1 Prototype

Clients often define a general-purpose set of software but do not identify detailed input, processing, or output requirements. On the other hand, the developers face the challenge of determining the algorithm efficacy, the adaptability of the operating system, and the form of the human-machine interaction. In this situation, and many others, the prototyping paradigm may offer the best approach [6].

A prototype is an unfinished form of a system based on which a developer makes a modification, alteration, and replacement. In this research, the prototype software gives the users an initial outlook of the program to be developed.

2.2 Blackbox Testing

Black-box testing, commonly referred to as validation testing, is a software testing method used to test software without knowing the internal structure of the code or the program. In this research, the subjects undergoing the test were aware of the program but did not have prior knowledge of how to use it.

2.3 Unit Testing

Unit testing is a test on a device software wherein each unit/component of the device software is tested. The goal is to validate that each unit within the device software performs as designed [7].

2.4 System Usability Scale (SUS)

The System Usability Scale (SUS) is one of the most popular usability testing tools. Developed by John Brooke in 1986, this tool evaluates various products and services, including device hardware, device software, device mobile, and website [8]. This SUS is a usability scale that is reliable, popular, effective, and cheap.

SUS consists of ten questions and five answer options. The answers range from 'very not agree' to 'agree.' SUS has a minimum score of 0 and a maximum of 100.

2.5 REST API

REST or Representational State Transfer is a software architecture model commonly applied by developers to website application programming interfaces (APIs) [15]. First introduced by Roy Fielding in 2000 [14], REST is a web-based communication architecture standard that is common in the development of web-based services.

3 METHODOLOGY

3.1 Requirements Engineering

3.1.1 General Description of the Software. Management of data logging for microclimate monitoring can help farmers record or monitor the microclimate conditions in the field. The information provides references for determining the best cultivation practices that benefit from the microclimate conditions at the time specified.

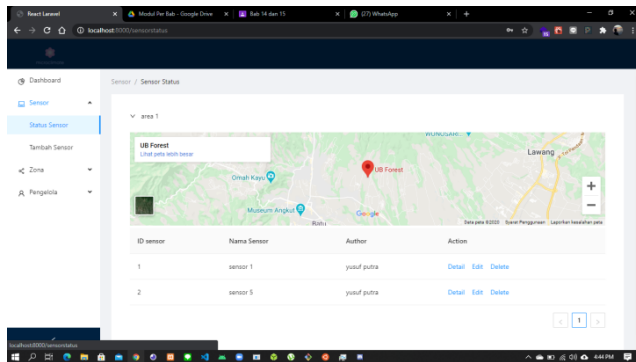


Figure 1: Prototype of Sensor List

The device software of the management for data logging for microclimate monitoring consists of a website using PHP language programming and a Laravel framework that acts as the back-end that regulates all data transfer. In addition, ReactJS, a Javascript library, serves as the front-end and the interface connecting the user and the system.

3.1.2 List of Requirements. Before defining a list of needs, the first step in the test is to define the involved subjects. In this study, the subjects are the user, the manager, and the farmer. The user acts as the individual with no access to the system, while the manager is the individual who has control of the system. In this situation, the manager can add a farmer and the sensors to be monitored and recorded. The farmer, on the other hand, can use the data from the microclimate monitoring periodically.

Second opinions from the university forest management regarding the management of data logging for microclimate monitoring are used to determine the list of needs. Each data is given a code so that each datum has a functional code SRS_DL_F_X. SRS is System Requirements Specification, DL stands for Data Logging, and F is Functional. Table 1 shows the drafted functional needs from the second opinion.

3.2 Prototyping

The prototyping stage begins with designing the interface and determining the program workflow from beginning to end. After the completion of the prototype, the next step is validating the users. When there is a review, the prototype will be revised and improved based on user input. Otherwise, the prototype moves to the next stage; design and implementation. During the prototyping stage in the research, it took three prototype iterations until the users were satisfied with the results.

3.2.1 Design. After reviewing the analysis of needs, the study found 16 features. Table 1 shows some of these features.

3.2.2 Prototyping. After the design is ready, the developer creates prototypes of a display interface for the application. In this research, the prototypes were built using the Figma application. Figure 1 displays some of the proposed prototypes.

Table 1: Lists the requirements of the third iteration

Requirement Name	Information
Login	Users can perform Login to enter into the system.
Add Sensor	Users can perform additional sensor
Delete Sensor	Users can perform sensor removal
Edit Sensor	Users can make sensor edits
List of Sensors	Users can see a list of sensors that are registered
Insert Zone	Users can perform additional zones
Delete Zone	Users can perform zone deletion
Zone Edit	Users can perform zone edits
Zone List	Users can view a list of zones
View Sensor Data	Users can view the sensor data is periodic, the data is displayed in the form of tables, charts bar, and line charts/lines.
Record Sensor Data	This requirement is used to record data sent by the gateway on the cloud server.

3.2.3 Customer Evaluation. Upon completing the prototyping stage, the users test prototypes. The users can immediately run the prototype to test whether the prototype can run properly.

From these prototypes, the appropriate prototype was selected and then developed into the system.

3.2.4 Review and Update. User review determines whether the prototypes need further improvement. If the review is positive, the prototype will move to the design stage. However, when users deem improvement is necessary, the prototype will undergo an update. In this research, after the third revision, the selected prototypes move to the design stage.

3.3 Needs Analysis

3.3.1 Use Case Diagram. Figure 2 above describes the interaction between users, managers, and farmers on the application. The users access the application by logging in, while managers and farmers constitute general users who can access the features on the diagram. Finally, there are individuals whose roles are to receive data sent by sensors.

3.3.2 User Case Scenario. The user case scenario explains the process of using the application. Furthermore, it helps developers get a broad description of the usage and record the experience when using the application from the beginning to the end. A case scenario gives an overview of what functions are running appropriately and whether the action and the system response are according to the actions of the subjects.

3.4 Design and Implementation

3.4.1 Design. After the entire diagram on the analysis is completed, the design, the data, interfaces, and algorithms are developed. Algorithms will facilitate the implementation of program code. The last step in the designing stage is designing the interface that assists in

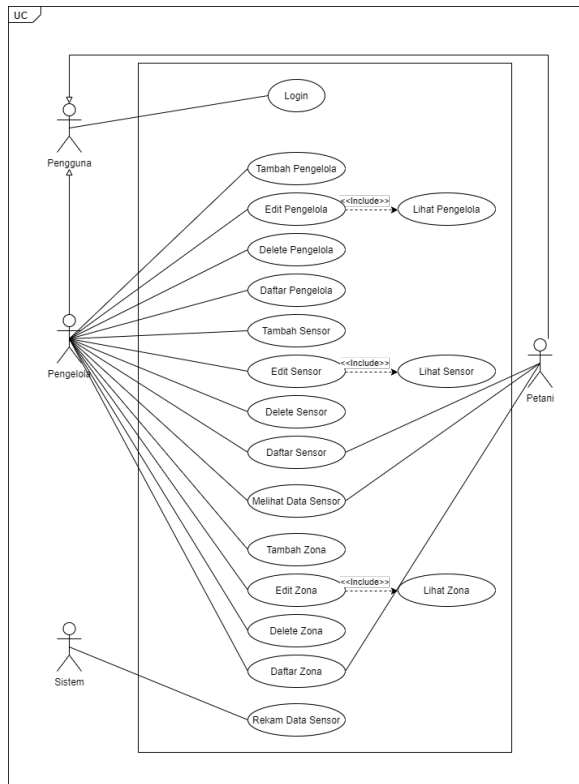


Figure 2: Use Case Diagram Application Data Logging Climate Micro



Figure 3: Design of Micro Climate Logging Data System

implementing the interface on the device software. The design of the prototype comes from the input from UB Forest management.

Overall, Figure 3 and Figure 4 . illustrate the system management data logging for microclimate monitoring in UB Forest.

Architectural Design. Application for the management of data logging for microclimate monitoring is developed in the platform's website by utilizing RESTful web service which in its communication is done with REST API. The system performs the request on the available APIs and the APIs that perform requests on the database per client request. For application, Restful web service

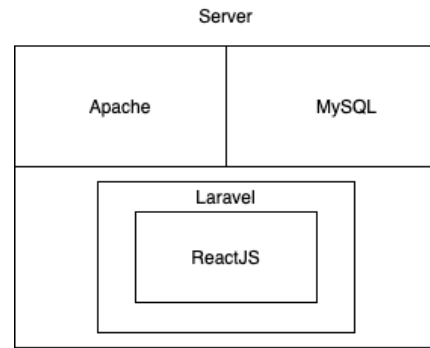


Figure 4: Architecture of the Application

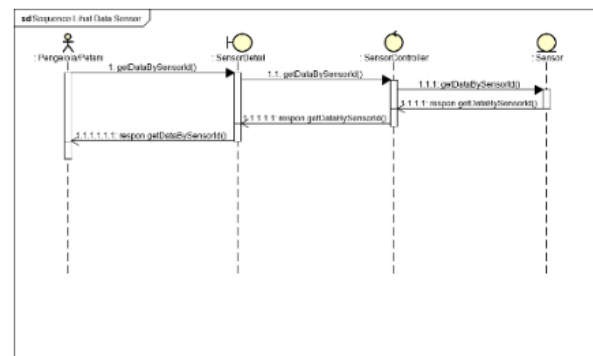


Figure 5: Sequence View Data Sensor

employs a PHP framework named Laravel that acts as a back-end service or the port for data transfer. In addition, a javascript library, ReactJS, serves as the front-end or the display interface.

Designing Sequence Diagram. The sequence diagram explains the steps that make the system respond to a function that generates an output. This sequence diagram is designed based on the use case scenario shown in Figure 5

Sequence View Data Sensor

Figure 5 above describes the sequence diagram of the needs of the sensor data view. The manager or farmer will access the sensor data in the sensorDetail view, and then the display will request sensor data from the sensorController and forward it to the sensor model to retrieve data from the database. Subsequently, the data will be returned to the sensorController and then to the sensorDetail view.

Class Diagram Design. Figure 6 is an illustration of the class diagram that illustrates the relationship of one class with another. In the picture above, there are four classes; UserController, zoneController, sensorController, and dataController, four of which extend or is plain called the inheritance of the class controller of Laravel. A class runs when the user is accessing the data in the database and when the data models support it.

Data Design. Figure 7 is an image designed to describe the data and their relationships that will be implemented later. The user table

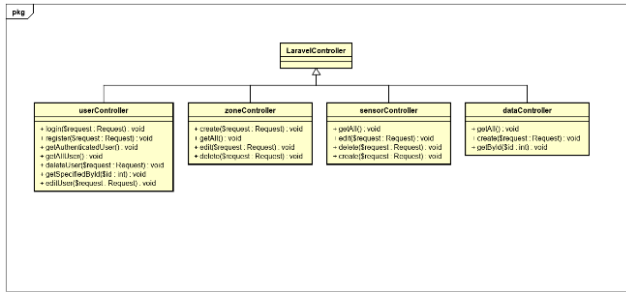


Figure 6: Class Diagram

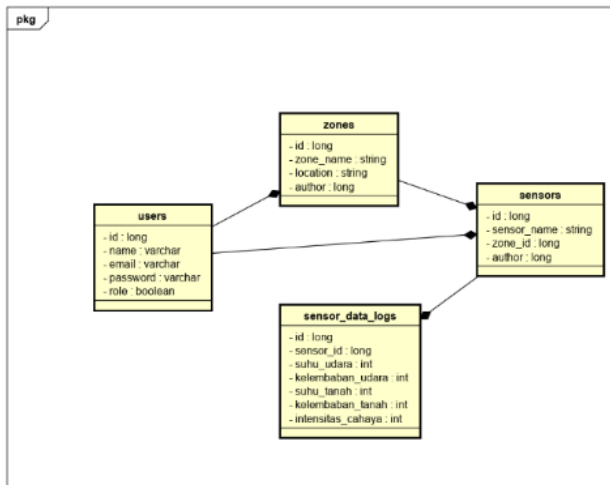


Figure 7: Data Design

becomes a part of some tables, such as the zone table and sensor table. The zone table is included in the composition of the sensor table, while the sensor table becomes a part of the sensor_data_logs table.

REST API Design. In designing a REST API, the developer firstly creates the design of the variable and the response to be applied to the REST API. The function explains the definition of each REST API.

3.4.2 Implementation. In the implementation stage, the foundation for the development of the device software comes from the design which previously has been made. The implementation is carried out based on the use case and sequence diagrams that have been previously determined. After the algorithm to the code of the program is completed, the PHP using a Laravel 7.0 framework will be implemented using Javascript programming language with ReactJS as the framework. Meanwhile, the database implementation employs MySQL, while REST API architecture sends the data to the database.

Component Implementation and REST API. Upon completing the analysis, prototyping, and designing stages, the subsequent phase

is implementing the results of the analysis, prototyping, and design into the developed application.

Sensor Data Record

Components of the sensor data record are the components that form an endpoint that became a point of communication between LoRa Gateway and the system so that the components do not have a display. In the next stage, the data recorded through this endpoint will be displayed on the preview of the components of the sensor data. The following is a snippet of program code from the sensor data record.

```

class DataController extends Controller
{
    public function create(Request $request)
    {
        $validator = Validator::make($request->all(), [
            "sensor_id" => 'required|digits_between:1,20|exists:sensors,id',
            "suhu_udara" => 'required|integer',
            "kelembaban_udara" => 'required|integer',
            "suhu_tanah" => 'required|integer',
            "kelembaban_tanah" => 'required|integer',
            "intensitas_cahaya" => 'required|integer'
        ]);
        if ($validator->fails()) {
            return response()->json($validator->errors()->toJson(), 400);
        }
        $data = SensorDataLog::create([
            "sensor_id" => $request->get('sensor_id'),
            "suhu_udara" => $request->get('suhu_udara'),
            "kelembaban_udara" => $request->get('kelembaban_udara'),
            "suhu_tanah" => $request->get('suhu_tanah'),
            "kelembaban_tanah" => $request->get('kelembaban_tanah'),
            "intensitas_cahaya" => $request->get('intensitas_cahaya')
        ]);
        return response()->json($data, 201);
    }
}
    
```

4 RESULT AND DISCUSSION

4.1 Testing

4.1.1 Testing Plan. The test plan is a test scenario according to which the tests are carried out. The results are presented in the test results section.

Unit Testing. Method create() in class dataController

a. Pseudocode

Algorithm 1 Iterative Algorithm

```

Begin
check request with the validator
if validator fails
return response error
else
insert data to sensor_data_logs table
return response data
end
b. Basis Path Testing
c. Flowgraph
    
```

Figure 8 above explains how the create method runs. This flow is called a flowgraph. From the flowgraph, we can determine cyclo-matic complexity.

i Cyclomatic Complexity
 $V(G) = 2$ Region

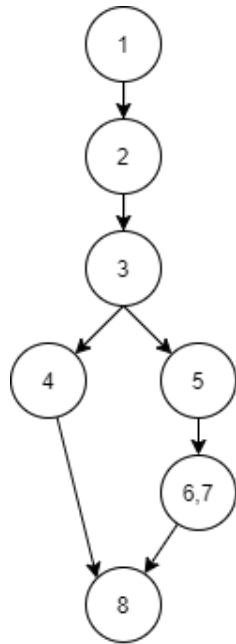


Figure 8: Flowgraph create()

ii Independent Path

Path 1 = 1,2,3,4,8

Path 2 = 1,2,3,5,6,7,8

4.1.2 Validation Testing. The previously created flowgraph design determines the validation testing. Subsequently, the expected result is the reference for whether the feature is valid when it is run for testing later.

4.1.3 SUS Testing. In the next test, the usability test was carried out using SUS (System Usability Scale). The entire research team, including the students and lecturers, took part in the testing. The objective of the test was to find out whether the system is feasible to run and record the data and whether the sensors for microclimate monitoring are working appropriately. In this test, users tested the system and then answered a questionnaire consisting of ten questions.

4.2 Test Result

4.2.1 Unit Testing. The testing was performed on each class based on the planning and the proposed flowgraph in the planning stage. The following are the results of the unit testing.

Method create() in class DataController

Table 2 above describes the results of the unit tests. The unit tests resulted in 100% validity, and the output of each unit was as per the expected result.

4.2.2 Validation Testing. Table 3 explains the results of validation testing. The results indicate 100% validity, while the output of each unit matched the expected results. It means that the system has fulfilled every requirement defined at the initial stage, and the system runs according to the expected result.

Table 2: Unit Test

Input Data	Expected Result	Result	Status
sensor_id : five air_temperature: two humidity_air: two ground_temperature: is not appropriate. two humidity_soil: two intensity_light : two	The system will show a JSON response form in which data alone is not appropriate.	System show JSON response form which data alone is not appropriate.	Valid
sensor_id : 1 air_temperature : 26 humidity_air : 88 Ground temperature : 30 humidity_soil : 99 intensity_light : 120	The system will display the data stored in the database along with the time and id.	The system displays the data that is stored in the database along with the time and id.	Valid

Name	Profession	Real Score									
		P1	P2	P3	P4	P5	P6	P7	P8	P9	P10
Adi Wahyu	Student	4	3	4	2	4	3	4	2	4	2
Isman	Student	5	1	5	1	5	4	5	1	5	1
Erdian Sahlan	Student	3	1	5	1	3	1	4	1	5	1
Awang Satya Kusuma	Student	4	2	4	2	4	2	5	2	4	4
Krishna Adam	Student	3	1	5	2	4	2	4	2	4	2

Figure 9: SUS Test Results

Real Score										Sum	Value
P1	P2	P3	P4	P5	P6	P7	P8	P9	P10		Sum x 2.5
4	3	4	2	4	3	4	2	4	2	28	70
5	1	5	1	5	4	5	1	5	1	37	92,5
3	1	5	1	3	1	4	1	5	1	35	87,5
4	2	4	2	4	2	5	2	4	4	29	72,5
3	1	5	2	4	2	4	2	4	2	31	77,5
Final Score											80

Figure 10: Final Results of SUS Calculations

4.2.3 SUS Testing. From testing the system and filling out the SUS questionnaire, Figure 9 presents the results of the questionnaire analysis. Furthermore, it shows the columns P1 to P10 that contain the answers to the ten questions in the questionnaire.

After the test, the next step was to calculate the score from the respondent data. Figure 10 illustrates the calculation of SUS that was performed based on the SUS calculation rules.

4.3 Analysis of Test Results

4.3.1 Unit Testing and Validation. In the unit testing stage, according to the case studies illustrated in the previous section, testing results indicate 100% validity, which means that all components within the system are appropriately running. Furthermore, in the

Table 3: Validation Testing Result

Test Name	Expected Result	Result	Status
Login Testing	The system will redirect to the dashboard page.	The system directs to the dashboard page.	Valid
	The system will display "please input your email" and "please input your password".	The system will display "please input your email" and "please input your password".	Valid
	The system will show the right "login failed".	The system displays "login failed".	Valid
Testing see sensor data	The system will show you the data in the form of graphs bar graph lines, and tables.	The system displays the data in the form of graphs bar graph lines and tables.	Valid
Testing added manager	The system will redirect to the manager list if the addition is successful.	The system redirects to a list of employees.	Valid
	An empty form will give a message that the form must be filled.	An empty form gives a message that the form must be filled.	Valid
Testing sensor data records	The data received will be stored in the database.	The data received is stored in a database.	Valid

validity testing employing Black-box, the score obtained was 100% valid. It means that the system has checked all the required needs.

With unit testing and validation testing done, the overall system can run according to the proposed flowgraph. Furthermore, all the requirements written in the requirements list have been embedded in the application so that users can test this application. In addition, unit testing and validation testing test whether the application can communicate with the sensors through the proposed REST API. Based on these tests, the sensors and the application can communicate and exchange data through the REST API according to the designed flowgraph.

4.3.2 SUS Testing. The five respondents who had filled the questionnaire produced a test score of an average of 80. The results fall

into a 'Good' category with a grade scale of B and acceptability ranges of 'Excellent'.

Successful usability testing means that the user can run the system and receive information smoothly according to the usability scheme. The results of this test indicate an average score of 80 and an 'Excellent' acceptability range, which means that this system is already above the average usability rating of 65 and that users have successfully used and approved the application with minimum difficulty. Additionally, the users also provided inputs for improvements to the application.

5 CONCLUSION AND SUGGESTION

5.1 Conclusion

After a series of development plans which consists of analysis, design, implementation, and testing procedures, the research concludes several points. Firstly, the designed API can successfully accommodate the reading of microclimate data through the internet gateway. The gateway then accesses the API by sending the designed body. Furthermore, the system verifies the data before being stored on the database. Processing of the data recorded is done by reading the data and storing the data on the database. The parameters monitor the data in the filters either every 15 minutes, every day, every month, or every year. The process takes place at the back-end service on the system. Upon processing, the front-end service then summons the data in the form of a bar graph, line graph, or table.

5.2 SUGGESTION

The research suggests some insights for future research. A mobile version of the application that is not native but a website in the form of an application might simplify the access for the users.

ACKNOWLEDGMENTS

The researchers would like to express gratitude to the UB Forest management and the Faculty of Agriculture, University of Brawijaya, for making it possible to conduct this research and the CEH Program for providing grant no: CRP 10-2012-03. SO was funded by the UK Natural Environment Research Council (SUNRISE project; NE/R000131/1).

REFERENCES

- [1] Y. Pusparsa, "Databoks," 2019. [Online]. Available: <https://databoks.katadata.co.id/datapublish/2019/12/10/sebaran-jumlah-lahan-produsen-kopi-di-dunia>. [Accessed 13 08 2020].
- [2] V. B. Kusnandar, "Databoks," 2019. [Online]. Available: <https://databoks.katadata.co.id/datapublish/2019/09/11/di-mana-lambung-kopi-indonesia>. [Accessed 13 08 2020].
- [3] H. Supriadi, "BUDIDAYA TANAMAN KOPI UNTUK ADAPTASI DAN MITIGASI PERUBAHAN IKLIM," vol. 13, p. 1, 2014.
- [4] APJII, "PENETRASI & PROFIL PERILAKU PENGGUNA INTERNET INDONESIA," Asosiasi Penyelenggara Jasa Internet Indonesia, Indonesia, 2018.
- [5] E. Kurniawan, "Penerapan Teknologi Cloud Computing di Universitas Kristen Duta Wacana," vol. 08, pp. 29-30, 2015.
- [6] P. Roger S. Pressman, Software Engineering: A PRACTITIONER'S APPROACH, 5 ed., Americas, New York: McGraw-Hill, 2001.
- [7] J. Suhartono, "Unit Pengujian," 2016. [Online]. Available: <https://sis.binus.ac.id/2016/12/16/unit-pengujian/#:~:text=Unit%20tes%20pada%20dasarnya%20ditulis,bagian%20individu%20bekerja%20dengan%20benar..> [Accessed 27 September 2020].

- [8] J. Broke, SUS-A quick and dirty usability scale (in "Usability Evaluation in Industry", PW Jordan, B Thomas, I McLelland, BA Weerdmeester (eds)), London: London: Taylor and Francis, 1996.
- [9] R. T. Fielding, "Architectural Style and the Design of Network-based Software Architectures (Dissertation)," 2000. [Online]. Available: <https://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm>. [Accessed 5 10 2020].
- [10] J. Heinonen, "Design and Implementation of Automated Visual Regression Testing in a Large Software Product," 2020.
- [11] Puppeteer, "Puppeteer v5.3.0," 2020. [Online]. Available: <https://pptr.dev/>. [Accessed 16 September 2020].
- [12] E. Susilo, "Cara Menggunakan System Usability Scale (SUS) Pada Evaluasi Usability," 7 March 2019. [Online]. Available: <https://www.edisusilo.com/cara-menggunakan-system-usability-scale/#:~:text=SUS%20ini%20merupakan%20salah%20satu,%2C%20populer%2C%20efektif%20dan%20murah.&text=SUS%20memiliki%2010%20pertanyaan%20dan%205%20pilihan%20jawaban>.
- [13] K. Lane, "API 101: What is a REST API? " 2020. [Online]. Available: <https://blog.postman.com/rest-api-definition/>. [Accessed 5 10 2020].
- [14] Fielding, R. T. (2000). Architectural Style and the Design of Network-based Software Architectures (Dissertation). Retrieved 10 5, 2020, from <https://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm>
- [15] Lane, K. (2020). API 101: What is a REST API? Retrieved 10 5, 2020, from <https://blog.postman.com/rest-api-definition/>