Brilliance
Research of Artificial Intelligence

E-ISSN : 2807-9035
Volume 4, Number 2, November 2024
https://doi.org/10.47709/brilliance.v4i2.5462

Design And Development Of A Backend System For Monitoring And Controlling Smart Agriculture

Maria Shusanti Febrianti[1*], Marzuki[2], Dian Resha Agustina[3], Aldi Prasetyo[4]
[1,2,3,4]Faculty of Computer Science, Bandar Lampung University, Indonesia
[1]maria.shusanti@ubl.ac.id, [2]marzuki@ubl.ac.id, [3]dian.resha@ubl.ac.id, [4]m.aldi.19421076@student.ubl.ac.id

open access

***Corresponding Author**

Article History:
Submitted: 03-12-2024
Accepted: 03-12-2024
Published: 31-12-2024

Keywords:
Backend; System Monitoring; System Controlling; Smart Agriculture

Brilliance: Research in Artificial Intelligenceis licensed under aCreative Commons Attribution-NonCommercial 4.0 International (CC BY-NC 4.0).

ABSTRACT

This study aims to build a system to manage temperature, pH, nutrient, and water availability sensor data for plants in smart agriculture. This is done so that the data sent by the sensor can be managed and stored properly. So that the data can be used for monitoring and controlling later. This study uses a waterfall model. Waterfall is often used to build a software system. The system that has been built needs to be tested. Based on the coding results, the system has two services. The first service handles user data, tools, plants, tool sets, and planting settings. The second service will hold the data that the tool sends to RabbitMQ. The results of the first service test run well, where every request sent by the client can be processed, and a response is returned as a request. In service, in two tests, some services ran fine. Where the data is successfully sent to the client side in real-time; however, when consuming and getting real-time data, it needs to be underlined; if getting real-time data is done when the JSON file is empty, it will return null. This may cause data delays to the client. The monitoring process will be a little late.

INTRODUCTION

The development of technology in the current era of modernization allows us to do work more easily, quickly, efficiently, and with high productivity.(Devira Ramady et al. 2019). One of the technologies that is currently developing rapidly is the Smart System. A Smart System is an intelligent system that connects various functions such as sensors, actuation, and control to analyze a situation.(Dwiyatno, Krisnaningsih, and Hidayat 2022). Then, a decision will be made based on the data obtained. The components have their respective functions—sensors to detect environmental conditions according to the sensors used. The controller unit is a decision-maker and provides instructions based on information/data obtained by the sensor(Lundin 2017). The actuator will perform the Actions commanded by the controller unit. This allows Smart System to address environmental, economic, and social issues such as climate change, resources, and globalization. Therefore, Smart Systems have been widely used in various sectors such as security, industry, housing, and agriculture, especially in agriculture itself, more familiarly called smart agriculture.

Smart agriculture changes land management patterns from conventional to more productive and efficient with the help of technology.(Ramady et al. 2020). Smart agriculture utilizes automation and monitoring systems by using the Internet of Things. The Internet of Things is a technology that allows electronic devices to connect via the Internet network (Siddhi, Rahmat, and Hertiana 2022). The Internet of Things works by utilizing a unique identity as a MAC address so that every device can connect anytime and anywhere as long as an Internet network is available (Dwiyatno et al. 2022). IoT has been widely used in many fields, especially agriculture and plantations. (Mardiana et al. 2024). Iot is an innovative solution, considering agriculture is the primary sector driving Indonesia's economy. In conventional agriculture, work is done manually. Such as in the process of watering, checking water, and so on. In smart agriculture, this can be done automatically by utilizing installed sensors, especially in greenhouses, where factors like this must be considered carefully.

A greenhouse is a building made to grow plants better by implementing the right ideas, such as temperature control, water pH, and automatic water control (Hafid Hilmi 2020). Greenhouses generally use plastic or transparent glass that can be penetrated by sunlight. This will certainly make the temperature in the Greenhouse even hotter.(Jeffry et al. 2021). The temperature in the Greenhouse must be continuously controlled to adjust to the needs of the plants. When the room temperature is not as needed, the microcontroller activates the actuator to readjust the room temperature to suit the needs of the plants.

The Greenhouse of Bandar Lampung University has two planting media, hydroponics and soil media. Hydroponics is a system of growing plants in a liquid solution.(Setiawan, Tanudjaja, and Octaviani 2019). The media often used are clay, coir, broken bricks, and wood fibers. There are 17 essential elements to be in the nutrient solution in hydroponics, such as carbon, hydrogen, oxygen, nitrogen, phosphorus, potassium, calcium, magnesium, sulfur, iron, copper, zinc, manganese, molten metal, boron, chlorine and nickel(Perwiratama, Setiadi, and Suyoto 2019). Hydroponics has been recognized as feasible for producing plants ranging from vegetables to ornamental plants.

This is an Creative Commons License This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

Brilliance
Research of Artificial Intelligence

E-ISSN : 2807-9035
Volume 4, Number 2, November 2024
https://doi.org/10.47709/brilliance.v4i2.5462

Hydroponics cannot be applied to all plants. In this study, researchers visited hydroponic cultivators, namely hydroponic friends. The visit was intended to obtain further information to support the research. Based on the results of interviews with employees, several things need to be considered in hydroponics, such as nutrient concentrations where each plant requires different concentrations. So, a tool such as a TDS meter is necessary to monitor the nutrient concentration in the water flowing through the hydroponics. The nutritional value must be stable. This will affect plant growth. If it is too excessive, it will cause plants to rot and dry easily, which will result in crop failure. The pH level in the water flowing through the hydroponics must also be considered. The pH of the water flowing through the hydroponics must be regarded because each plant requires a different water pH. If the water pH is not appropriate, it can affect plant growth. The temperature of the room around the plant must be controlled. If the temperature does not match the plant's needs, it can cause the plant to wilt. The temperature around the plant must be stable at around 27°-31°C. In the Greenhouse itself, the temperature of the plants is still not very stable depending on the time and weather. And the availability of water in hydroponics itself must always be considered. Because plants will absorb water in hydroponics, periodic filling is needed to ensure that water availability is always sufficient. Based on information from interviews with hydroponic friends, the water used on 1 of their hydroponic tables reaches 2 liters. Of course, this can vary based on the size of each table. Moreover, plants will absorb water periodically, which will cause the water to decrease continuously. So, the water in the reservoir must be filled periodically so that water availability is always sufficient. Meanwhile, the soil planting medium will be planted with vanilla plants.

The tools used in this study have been equipped with several sensors to monitor the environment around the plants. These sensors include TDS sensors to monitor nutrient levels, pH sensors to monitor the acidity level of water(Syafruddin et al. 2021), DHT11 sensors to monitor room temperature in the Greenhouse (Ramady et al. 2020), water flow sensors to measure the water discharge used to flow both planting media and water level sensor to regulate the filling of water in the tank automatically. In addition, a soil moisture sensor is used to monitor the humidity in the planting medium planted with vanilla and a ZMPT101B sensor is used to measure the electrical power used.

The main task of this research is to develop a system that can manage data sent by these sensors to RabbitMQ in real time. The backend will then process and store this data, providing an API for website administration needs. The microcontroller will send the data obtained from the sensors to RabbitMQ in real-time. Based on the experiment of sending data via a microcontroller, data sent at 10 seconds for 1 hour received 361 messages with a size of 13 KiB or 106496 bits. Of course, the size of each message will differ depending on the message sent itself. The new experiment was carried out with one microcontroller and 1 hour of experiment. Of course, in the IoT system that will be designed, data is not only from 1 sensor and lasts for 1 hour. Data needs to run continuously in real-time to monitor the greenhouse environment.(Dwiyatno et al. 2022). Therefore, a backend is needed that can manage and store the data well.

The backend is a program that runs on the server side.(Adam, Rachmat Anom Besari, and Bachtiar 2019). The backend is responsible for interacting directly with the database. One of the technologies often used in interacting with different client services is Application Programming Interfaces (API). API is a set of functions that other programs can run. API is designed in various ways. However, in web applications, it is necessary to go through the HTTP protocol, which is easy to implement, maintain, and standardize.(Afifah and Setiaji 2019). The API's purpose is to share data between different platforms.(Rahardja et al. 2018). Developers can create separate functions to speed up the application development process so that developers do not need to develop similar features again. One of the backend architectures is Representational State Transfer (REST). REST is a set of architectural principles that send data through an interface standardized by HTTP(Dog 2018). When using REST, the client will send commands or requests using HTTP requests, and the server will respond with HTTP that is made into text in various forms, although often in JSON or XML. (Haro et al. 2019).

In this study, researchers will design a Backend to manage data integrated with the sensors used to measure electrical power, starting from DHT-11, Water level, TDS, pH, Water flow, and ZMPT101B sensors. The data obtained by the sensors will be sent to RabbitMQ and then managed by the backend.

LITERATURE REVIEW

Research related to this research was conducted by BM Adam et al. on backend server system design based on rest API for cashless payment systems in the retail community that overcomes the problem of researching problems related to non-cash payment systems. Non-cash payment system requires special tools. One of them is electronic data capture (EDC). EDC is an electronic machine used on debit or credit cards to make payments without cash. However, EDC has requirements and provisions divided into individuals, business entities, and foundations. Some retail communities cannot always meet these requirements. Therefore, the author feels a system is needed to help consumers and retail communities make cashless payments. Therefore, in this study, a cashless payment system will be created as a backend server based on REST API. The non-cash transaction system is created to handle processes such as payments, top-up balances, and transaction logs. Then, these features will be packaged in one system that various platforms can access. The data communication method used is HTTP Request using REST API. This method is widely used for developing an application because many programming languages and platforms can use it. This study used a system

This is an Creative Commons License This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

Brilliance
Research of Artificial Intelligence

E-ISSN : 2807-9035
Volume 4, Number 2, November 2024
https://doi.org/10.47709/brilliance.v4i2.5462

design to define functional requirements and design preparation and implementation to explain how a system is formed. This aims to meet system needs and provide a clear and complete picture to developers and readers. In the design of the backend server system created, the server will manage data received from the client application and send the results to the client application. The system will receive input from the front end. The inputted data will be processed in the backend, such as account lists, account logins, payment transactions, and bank API access to request transfer codes with a database to store logs and data in the payment system. After the system was created, testing was carried out by running the program on a virtual private server or VPS with certain specifications. Several tests were carried out, such as token testing for system security, refill testing with Bank API, and resilience testing in data traffic. The test was carried out to assess the system's readiness for widespread use. After conducting the research above, a conclusion was obtained. Starting from the system designed in this study involves account management systems such as login, register, and logout. In addition, there is a balance system, admin refill information, log notifications, account transfers, and cash flow reports. The protocol uses REST architecture with HTTP requests: Get, Post, Delete, and Put. It involves a security system with JWT tokens.The system testing results of 100 API Requests were carried out in 1 second with a system success rate of 76.92% of the 13 existing features. The transaction feature has a success rate of 45% for the buyer's balance reduction process and a success rate of 46% for the process of 10 seller balance additions. The balance top-up feature reaches 72%, and the withdrawal feature has a success rate of 77%. In this study, several things must be added for further development. Such as improvements to the processor and memory so that the system can process all incoming requests. Additional permission is required from the MIDTRANS payment gateway to use real money transactions. Furthermore, research was conducted by Lundin et al., who wanted to build a Greenhouse that could grow plants throughout the season. Lundin thought that if it had to be imported from other regions, it would certainly be expensive. Therefore, Lundin aims to create an automatic greenhouse using hydroponics to grow fresh organic herbs and lettuce in urban environments to produce fresh vegetables and herbs, which can be produced organically using sensors and actuators to control water levels, pH, and nutrient levels. In the Lundin project, the system will be able to measure and control: 1) Height, 2) pH level, and 3) nutrient concentration. There will also be an artificial light source in the Greenhouse, but sensors cannot control it. Then, the plant data will be input into the system as an indicator to optimize plant growth and precise measurements of the nutritional value of the plants can be made. The selection of the hydroponic system makes it easier to measure nutrient levels and pH. Still, this system uses less water than conventional soil cultivation and naturally resists parasites and diseases. There are several types of hydroponic systems. In this project 11, an ebb and flow or drainage system is used. This system will receive the nutrient solution through the flow and be exposed to oxygen during the low tide period. A water pump will manage the flow. The water level sensor will ensure the water rises to the desired level. As explained above, the Greenhouse relies on several sensors for accurate measurements, such as water level, pH, and EC. With different actuators such as pH, nutrient dose, and water pumps. The pH value of the nutrients will be measured using a digital industrial-grade pH probe. The pH probe is made by DFRobot and is known as Gravity: Analog pH Sensor / Meter Pro Kit For Arduino. The measurements vary depending on the pH value being measured. The concentration of nutrient values in water is measured based on its electrical conductivity. Pure water will naturally have the ability to conduct electricity. Although very low, However, this capability can be improved by adding charged particles to it. Several influencing factors must be considered, including a. Water temperature, b. Concentration of particles in water that occurs naturally c. pH solution Based on the results of the pH test, the pH solution showed a clear correlation between increasing acidity and increasing the solution that had been added. However, the acidification appeared very nonlinear, and the exact correlation was not determined. The EC meter was calibrated, resulting in a 20% margin of error. Better results can be achieved through multiple measurement points. The theoretical value of 0.0058 EC units of resolution is likely still not achieved due to other factors, such as uneven distribution in the tank. 12 The drastic decrease in pH cannot be explained with certainty. A possible explanation is poor mixing of the pH-lowering solution, resulting in a high concentration of the solution because it was added at one time. When the water pump was turned off, the system also had poor water circulation. Therefore, mixing the pH solution into the tank took quite a long time. This is a possible explanation, indicating that the pH-lowering solution was added to the water tank before being mixed and measured. Based on the above, it can be concluded that the system can maintain stable pH and EC values and remain within a predetermined range. The interval should be adjusted upwards to avoid a large decrease and cannot be shown to maintain a smaller interval value. Since plants consume nutrients relatively slowly, it is easy to control EC. pH is somewhat sensitive but with better pH concentration. The system can also maintain pH levels within predetermined limits for longer. With further testing, it is believed to maintain a value of around 0.2. Then, the research related to the problem of the API gateway system, which is the only entrance to the backend service by M. Song et al. If the system is damaged, it can cause all services to be unavailable. Therefore, the high availability of API Gateway services must be guaranteed. Therefore, the author presents an API Gateway System that can effectively reduce the number of remote calls and enable changes to backend services that are invisible to the application. designed an automatic scaling system for API Gateway. This can dynamically adjust the number of API Gateway instances according to the workload, improving system resource utilization while ensuring the high availability and quality of API Gateway services. Several processes were carried out in this research. In the Gateway API design stage, the communication pattern consists of 2 parts: application and

This is an Creative Commons License This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

Brilliance
Research of Artificial Intelligence

E-ISSN : 2807-9035
Volume 4, Number 2, November 2024
https://doi.org/10.47709/brilliance.v4i2.5462

backend service.Backend services perform certain functions and display capabilities with service addresses. Applications are consumers that call backend services such as mobile and web applications. API Gateway has three subsystems, including the core subsystem, the monitor subsystem, and the admin subsystem. The core subsystem includes several components: user authentication, authorization, service discovery, service orchestration, routing, and result processing. The monitor subsystem includes monitor alerts and API statistics. The main task of this subsystem is to monitor the health and performance of the core subsystem. The admin subsystem is responsible for controlling and configuring the core subsystem. 16 Includes flow limits, visual orchestration, API lifecycle management, and privilege management. In a microservice architecture, the API Gateway system is the only gateway to the backend service. When the number of requests to the API Gateway service is simultaneous, it can cause the service itself to be blocked or even stopped. This can cause all services to stop working. To overcome this, Kubernetes and Prometheus were used to build an automatic scaling system. Kubernetes itself is an open-source cluster manager for docker containers. It provides cluster management based on the container as a service or CAAS level. While Prometheus is an open-source monitoring system by SoundCloud. The pull model and HTTP protocol are used to collect data indicators. As long as the system can provide the HTTP interface, it can access the monitoring system. Prometheus supports container monitoring and can be easily deployed to the Kubernetes cluster as a monitoring system. The auto-scaling system can dynamically adjust the number of application instances according to the workload that can be borne. This can ensure high application stability and improve system resource utilization. The maximum and minimum thresholds of the number of API Gateway instances themselves can be determined by system parameters. If the API Gateway workload is low, only the minimum number of API Gateway instances is deployed. With the increase in workload, which may even exceed the workload threshold. The scaling system will scale to allocate more instances to balance the load. When one of the API gateway instances goes out abnormally, the system will automatically allocate a new instance and move the workload to the new instance to ensure the continuity of API server availability. 17 Based on this research, it can be concluded that, after using the API Gateway mechanism, it can reduce the number of calls and simplify the complexity of backend services that call each other; applications and backend services only need to communicate with the API Gateway. Researchers can dynamically replace or modify backend services.While the Auto Scaling System can dynamically adjust the number of services according to the load of the API gateway itself. This can increase the utilization of system resources and produce higher availability and quality of service from the API Gateway. In the related research, a system has been developed that helps automate maintenance in hydroponic plant cultivation, which is the basis for conducting this research, namely the design of the backend monitoring and controlling system on Smart Agriculture.

METHOD

The research method used in this study is the SDCL method with a waterfall approach. SDLC is a methodology commonly used in designing, building, and maintaining information and industrial systems.(Dharmawan, Purwaningtias, and Risdiansyah 2018). The waterfall model is often used to create a system.(Mallisza, Hadi, and Aulia 2022). This model has sequential steps, starting from needs analysis, design, coding, and testing.(Syahputri, Ade Pratama, and Anggri Yulio Pernanda 2022).

Analysis is a data collection technique used to obtain and collect data. Various data are collected as supporting facts to support the research. The analysis used in this study includes literature analysis and interviews.

Literature analysis is done by searching for information related to the topic of discussion being studied. Information can be obtained from various references such as books, the internet, and journals related to research.

Meanwhile, interviews are a data collection technique that involves asking questions to sources. (Hansen 2020). Interviews were conducted to obtain data or information to support the research. In the interview we conducted with one of the hydroponic farms, namely "Sahabat Hidroponik," we collected some data about things that need to be considered in the hydroponic farming system, Such as nutrients, room temperature, pH, water discharge, and electricity costs. Nutrients will vary for each plant. In lettuce plants, nutrients range from a maximum of 1000 ppm; in mustard greens, a maximum of 1000 - 1300 ppm; and in spinach, it can be up to 1500 ppm. The pH of the water ranges from 5.5 - 7. At room temperature, the environment around the plant ranges from 27-31 °C. Every morning, they refill water up to 1000L.

The experiment was conducted to determine the amount of data sent within 1 hour. In this study, researchers used a Wemos D1 mini, an ESP32 microcontroller equipped with a Wi-Fi module. Wemos will send data continuously in 2 intervals of 5 and 10 seconds. Data transmission will be carried out for 1 hour.

The first experiment was conducted with a data-sending interval of 5 seconds. It lasted 1 hour and obtained 721 data with a size of 27 KiB or around 221184bit. The image of the incoming data in experiment 1 can be seen in table 1.

Table 1. Data obtained from 5-second intervals

Queue Matrix	Total	Ready
Message	721	721
Message Body Bytes	27 KiB	27 KiB
Process Memory	22 KiB	

This is an Creative Commons License This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

Brilliance
Research of Artificial Intelligence

E-ISSN : 2807-9035
Volume 4, **Number** 2, November 2024
https://doi.org/10.47709/brilliance.v4i2.5462

In the second experiment, conducted with a data sending interval of 10 seconds and a sending duration of 1 hour, 361 data were obtained with a size of 13 KiB or around 106496 bits. The image of the incoming data in experiment 1 can be seen in table 2.

Table 2. Data obtained from 10-second intervals

Queue Matrix	Total	Ready
Message	361	361
Message Body Bytes	13 KiB	13 KiB
Process Memory	22 KiB	

With the two experiments above, the data obtained is very much in just one hour. Of course, the system that will be built will not only run for one hour. It will run continuously so that the monitoring process can be carried out. Therefore, a worker who can handle the data sent will be needed to manage the data appropriately.

In the design stage, researchers create the system workflow design. This stage helps in the research process. The design aims to facilitate the creation of the system and help explain each part of it.

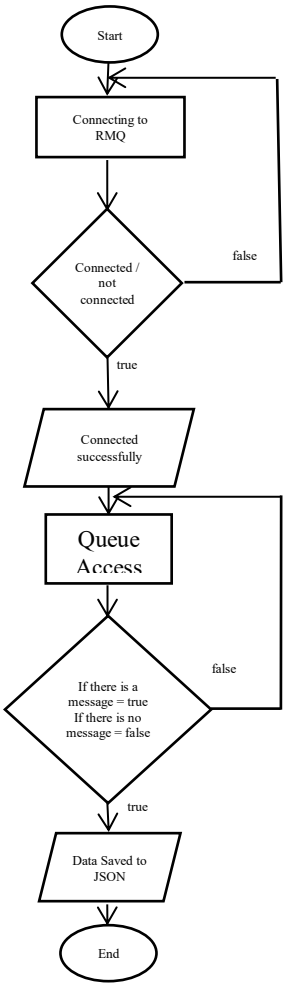


Figure 1. Data obtained from 10-second intervals

Flowchart diagram Description:

a. When the program is run, it will run the server and connect to the database.
b. If an error occurs, it will reconnect. If it is successfully connected, it will immediately connect to RabbitMQ.
c. If an error occurs, it will reconnect to RabbitMQ. If successful, it will output successfully connected.
d. After successfully connecting, it will access the queue and consume messages.
e. If msg is null, the program will re-consume it. If msg exists, the program will save it to the database.

At the coding stage, this project has two services with different tasks. The first handles requests for features, such as user data, plant data, tool data, data tool settings, and plant settings. This service will bridge the request and response flow sent by the client and server and interact with the MongoDB database. The other service will handle the data sent

This is an Creative Commons License This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

Brilliance
Research of Artificial Intelligence
E-ISSN : 2807-9035
Volume 4, Number 2, November 2024
https://doi.org/10.47709/brilliance.v4i2.5462

by the tool to RabbitMQ and save the data into a local file in JSON format.

Service1

The first service is divided into models, services, controllers, and routes, each of which is interconnected. The model manages fields and data types in the database, and the service plays a role in operations related to bridging the interaction between the database and the controller. The controller handles client requests, interacts with the service, and manages client responses. The route provides endpoints that clients can access and interact with the controller.

The first service will be assigned to handle client requests directly related to the database. Requests to be handled are related to features such as CRUD on users, plants, tools, tool configurations, and plant settings.

Service2

In service 2, the data received from the rmq queue will be handled and sent to the front end in real-time. The data obtained by the service will also be saved into the data.json file in *the local public folder*. Unlike service 1, the second service does not directly interact with the MongoDB database. Every time data is needed in the database, service 2 will send a request to Service 1 via the available endpoint.

RESULT

The service is built with nodejs in the development process because this framework is very suitable and flexible for creating services. This framework provides many libraries that can help the process of building services. The service that has been created will bridge the request and response process that occurs between the client side, RabbitMQ, and the database. The request process related to the database will be handled by Service 1, and requests related to sensor data will be handled by Service 2. The entire process will be as in Figure 2.

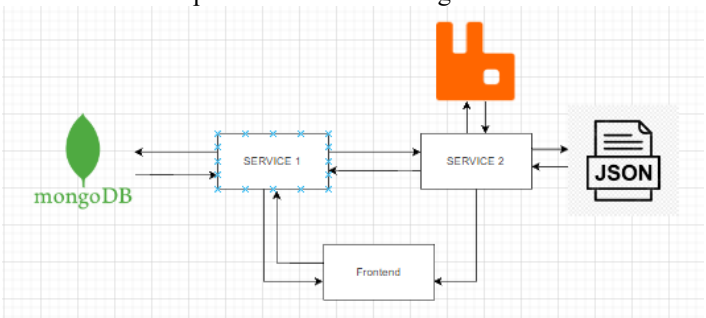


Figure 2. Service workflow that has been created

The workflow of the service that has been created is as follows:

1. If the front end makes a request related to a data request to the database, it will be handled by Service 1, and Service 1 will send a response according to what the client requested.
2. If the front end makes a request related to sensor data, service 2 will handle it and send a response as requested.
3. *Service* The second will handle the data stored by RabbitMQ. Service 2 will store the data in a JSON file and send it to the front end in real-time.

DISCUSSION

Based on the test results above, service 1, which provides services that will help manage data from the front end, runs well. The data sent from the client data request is stored in the database according to the declared model field. Also, the experiment with Postman successfully got data, *edited* data, deleted data properly, and returned the response as requested by the client (Postman). So it can be ensured that it can be applied to the front end properly.

Meanwhile, the consume data function ran well in testing service 2, which provides services related to data management sent by the device into the RabbitMQ queue. Service 2 will check the queue taken from the device name that the admin has registered. The name of this device will be used as the queue name that will be checked by consume. It will be saved to the JSON file if there is data. The data stored by the queue is like the image (). The service will take the data stored in this queue using the consume function. When the data is successfully consumed, the data will be reformatted into a data object by the function. It becomes like the object response variable Data in the image (). Then, the data will be saved into a JSON file by adding all the data in the JSON file, adding new data, and then keeping it again. This consumer service can run well if no problems exist with the RabbitMQ server that stores the data. If there is an error on the RabbitMQ server, service 2 will be disconnected so that the data cannot be consumed by service 2. The internet is needed in this process because connecting service 2 to RabbitMQ requires a stable internet network because the data must be stored continuously. So, the consume data function runs well if there is no problem on the RabbitMQ server itself or if there is no network problem that causes RabbitMQ to disconnect. It needs to be underlined: if service

This is an Creative Commons License This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

E-ISSN : 2807-9035
Volume 4, Number 2, November 2024
https://doi.org/10.47709/brilliance.v4i2.5462

2 and RabbitMQ are not connected, service 2 will try to connect to RabbitMQ, preventing other services from running. In service 2, there is also a service that allows clients to get setTanam data along with sensor data registered in setTanam in real-time. Based on testing, this service will run when the client sends IDUSERS periodically to service 2. When IDUSERS is received, the service will request service 1 to send the setTanam data to the MongoDB database.

When the data is received, service 2 will check whether the MAC_ADDRESS of the tool in setTanam is in the json file or not. Only the latest data will be taken and added to the setTanam object if there is. Then, the final result will be returned via socket io to the front end. However, there is something that needs to be underlined. If the sensor data retrieval in the JSON file is done simultaneously with the consume function, it will save. The sensor data may not be found. This is because the consume function will take data from the JSON file, which allows the JSON file to be empty for a while so that no sensor data is found.

Based on the discussion above, if more devices send data and more queues are required, it will cause more empty JSON files. This will increase the chances of retrieving data being an error. Of course, this can interfere with the monitoring process that the client will carry out.

CONCLUSION

Based on the research above, both services that have been created are running well. The first service can handle client requests well. Data inputted by the client will be saved to the database. The experiment with Postman succeeded in getting data, editing data, deleting data, and returning the response according to what was requested by the client (Postman). So, it can be ascertained that it can be applied to the frontend system well.

In addition, on the second service, the service that is built can run well, such as consume, get real-time data with socket io, and get log data. However, when consuming and getting real-time data, it should be underlined that if getting real-time data is done when the JSON file is empty, it will return null. This may cause data delays for the client so that the monitoring process will be a little late.

So, the first service can run well and is ready to be implemented. However, the second service needs to be adjusted to consume and get real-time data to minimize the clash of the two processes. You could also try saving the consuming data to the database only.

REFERENCES

Adam, Bob Maulana, Adnan Rachmat Anom Besari, and Mochamad Mobed Bachtiar. 2019. "Backend Server System Design Based on REST API for Cashless Payment System on Retail Community." *IES 2019 - International Electronics Symposium: The Role of Techno-Intelligence in Creating an Open Energy System Towards Energy Democracy, Proceedings* 208–13. doi: 10.1109/ELECSYM.2019.8901668.

Afifah, Khairina, and Hari Setiaji. 2019. "Pengembangan Rest Api Sebagai Teknologi Interoperabilitas Pada Aplikasi Uii Training Center." 1–8.

Devira Ramady, Givy, Rahmad Hidayat, Andrew Ghea Mahardika, Reza Rahman Hakim, and Sekolah Tinggi Teknologi Mandala. 2019. "Sistem Monitoring Data Pada Smart Agriculture System Menggunakan Wireless Multisensor Berbasis IoT." 4. doi: 10.22236/teknoka.v%vi%i.4173.

Dharmawan, Weiskhy Steven, Deasy Purwaningtias, and Deni Risdiansyah. 2018. "Penerapan Metode SDLC Waterfall Dalam Perancangan Sistem Informasi Administrasi Keuangan Berbasis Desktop." *Jurnal Khatulistiwa Informatika* 6(2):159–67. doi: 10.31294/khatulistiwa.v6i2.160.

Doglio, Fernando. 2018. *REST API Development with Node.Js: Manage and Understand the Full Capabilities of Successful REST Development*.

Dwiyatno, Saleh, Erni Krisnaningsih, and Dede Ryan Hidayat. 2022. "SMART AGRICULTURE MONITORING PENYIRAMAN TANAMAN BERBASIS INTERNET OF THINGS." 9(1).

Hafid Hilmi. 2020. "Sistem Monitoring Dan Otomatisasi Smart Greenhouse Untuk Tanaman Tembakau Kasturi Dan Besuki Menggunakan Wireless Sensor Network Berbasis IoT."

Hansen, Seng. 2020. "Investigasi Teknik Wawancara Dalam Penelitian Kualitatif Manajemen Konstruksi." *Jurnal Teknik Sipil* 27(3):283. doi: 10.5614/jts.2020.27.3.10.

Haro, Edward, Teresa Guarda, Alex Omar, Zambrano Peñaherrera, and Ninahualpa Quiña. 2019. "Desarrollo Backend Untuk Aplikasi Web , Layanan Web Restful : Node . Js vs Spring Boot." 309–22.

Jeffry, Luqman, Mei Yin Ong, Saifuddin Nomanbhay, M. Mofijur, Muhammad Mubashir, and Pau Loke Show. 2021. "Greenhouse Gases Utilization: A Review." *Fuel* 301(January):121017. doi: 10.1016/j.fuel.2021.121017.

Lundin, Karl. 2017. "Automated Hydroponics Greenhouse."

Mallisza, Danyl, Harry Setya Hadi, and Annisa Tri Aulia. 2022. "Implementasi Model Waterfall Dalam Perancangan Sistem Surat Perintah Perjalanan Dinas Berbasis Website Dengan Metode SDLC." *Jurnal Teknik, Komputer, Agroteknologi Dan Sains* 1(1):24–35. doi: 10.56248/marostek.v1i1.9.

Mardiana, Ardi, Ano Tarsono, Syiffa Safiera Wahono, and Harun Sujadi. 2024. "Alat Pencatat Cuaca Untuk Meningkatkan Kualitas Dan Efisiensi Di Majalengka." *Jurnal Pendidikan Teknologi Informasi* 4(2):649–59.

Perwiratama, Rangga, Yosef Kely Setiadi, and Suyoto. 2019. "Smart Hydroponic Farming with IoT-Based Climate and

This is an Creative Commons License This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

E-ISSN : 2807-9035
Volume 4, **Number** 2, November 2024
https://doi.org/10.47709/brilliance.v4i2.5462

Nutrient Manipulation System." *Proceeding - 2019 International Conference of Artificial Intelligence and Information Technology, ICAIIT 2019* 129–32. doi: 10.1109/ICAIIT.2019.8834533.

Rahardja, Untung, Qurotul Aini, Nuke Puji, and Lestari Santoso. 2018. "Integration of API-Based YII Framework on Attendance Assessment System." *Jurnal Ilmiah SISFOTENIKA* 8(2):140–52.

Ramady, Givy Devira, Andrew Ghea Mahardika, Ninik Sri Lestari, Hetty Fadriani, Herawati Yusuf, Sekolah Tinggi, Teknologi Mandala, and Cloud Computing. 2020. "PERANCANGAN MODEL SIMULASI SMART AGRICULTURE SYSTEM."

Setiawan, Yodi, Harlianto Tanudjaja, and Sandra Octaviani. 2019. "Penggunaan Internet of Things (IoT) Untuk Pemantauan Dan Pengendalian Sistem Hidroponik." *TESLA: Jurnal Teknik Elektro* 20(2):175. doi: 10.24912/tesla.v20i2.2994.

Siddhi, Fadhlul Wafi, Basuki Rahmat, and Sofia Naning Hertiana. 2022. "Sistem Monitoring Kesehatan Sapi Dengan Metode Waterfall Melalui Internet Of Things." *E-Proceeding of Engineering* 8(6):3952–61.

Syafruddin, R., Dika Surya Permana, Yakob Liklikwatil, and Givy Devira Ramady. 2021. "RANCANG BANGUN SISTEM MONITORING KAPASITAS DAN KADAR PH AIR PADA TOREN BERBASIS ARDUINO NANO." *Jurnal Online Sekolah Tinggi Teknologi Mandala* 16(2):59–65.

Syahputri, Wulan Dian, Ade Pratama, and Anggri Yulio Pernanda. 2022. "Perancangan Sistem Informasi Program Kerja Organisasi Kemahasiswaan Berbasis Web." *Decode: Jurnal Pendidikan Teknologi Informasi* 3(1):22–29. doi: 10.51454/decode.v3i1.68.

This is an Creative Commons License This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.