Implementation of MVC Architecture on RESTful API for Monitoring Covid-19 Patient Condition Parameters using Laravel Framework with Waterfall Method

Muhamad Ivan Fadilah[1], Tien Fabrianti Kusumasari[2], and Sinung Suakanto[3]

[1,2,3] Telkom University, Bandung, Indonesia
[1] muhamadivan@student.telkomuniversity.ac.id
[2] tienkusumasari@telkomuniversity.ac.id
[3] sinung@telkomuniversity.ac.id

Abstract. The Covid-19 pandemic has created an urgent need to monitor patients' conditions efficiently and accurately. In an effort to overcome this challenge, this study aims to implement the Model-View-Controller (MVC) architecture on the Representational State Transfer (RESTful) API to monitor the condition parameters of Covid-19 patients. The Laravel framework is used as a development foundation, with the Waterfall method approach as a guide for project management. The application of MVC architecture to RESTful APIs enables a clear separation of duties between business logic, views, and data modeling. This facilitates parallel development and easier maintenance while ensuring system cohesion and flexibility. Through the use of the Laravel framework, various features such as routing, controllers, and database management can be efficiently integrated. The Waterfall approach provides a detailed structure for the stages of analysis, design, implementation, testing, and maintenance. With the integration of the Laravel framework, the system is able to efficiently collect, manage, and report patient data. The entire study provides valuable guidance for similar implementations in the areas of health monitoring and software development.

Keywords: Covid-19, MVC, API, Laravel, Waterfall.

1 Introduction

In 2019, a variant of the virus was discovered and is now known as Covid-19. The virus was originally discovered in Wuhan, China, and can spread widely and is fatal to many people. In Indonesia, the first case of Covid-19 was confirmed in March 2020. On March 9, 2020, the World Health Organization (WHO) declared Covid-19 a global pandemic. Until now, positive cases of Covid-19 are still occurring. According to information quoted from the official WHO website on August 16, 2023, it was recorded. On a global scale, there were 769,774,646 positive cases of infection and 6,955,141 deaths. On a national scale, positive cases of infection reached 6,813,095 people and 161,916 people died [1].

M. D. Sulistiyo and R. A. Nugraha (eds.), *Proceedings of the International Conference on Enterprise and Industrial Systems (ICOEINS 2023)*, Advances in Economics, Business and Management Research 270,
https://doi.org/10.2991/978-94-6463-340-5_23

With the continuation of positive Covid-19 cases in Indonesia, as well as limited hospital capacity to provide isolation places, individuals who test positive through SWAB tests with mild or even asymptomatic symptoms such as Happy Hypoxia (severe pneumonia), may experience abnormal changes in blood oxygen levels, heart rate, and body temperature. Therefore, it is recommended that they undergo self-isolation at their respective residences [2]. This approach has the potential to reduce the growing burden on hospitals due to Covid-19 patients requiring isolation. Therefore, it is important for hospitals to have systems that allow connection with patients undergoing self-isolation or hospitalization, with the aim of reducing direct interaction between the medical team and the patient [3].

Integrating dashboards will enable healthcare professionals to monitor and improve the quality of patient care data, by facilitating timely management of health resources and supporting informed clinical decision-making [4]. Now, the use of sophisticated digital technology has a very crucial role. In this context, the use of the Internet of Things (IoT) also has an important value in monitoring and tracking intelligently against Covid-19. The combined use of IoT with artificial intelligence and cutting-edge computing technology in monitoring and tracking patients undergoing quarantine has become an integral element in the fight against the current spread of Covid-19 and its potential during the upcoming pandemic [5] . Therefore, a system is needed that is able to monitor the condition of Covid-19 patients, both those who are being treated in the hospital and those undergoing self-quarantine at home. This monitoring system can be in the form of a dashboard that presents information related to the patient's condition [1].

Some of the parameters of conditions that can be monitored include electrocardiography (ECG), heart rate (HR), respiratory rate (RR), blood oxygen level (SpO2), as well as body temperature. Information from these parameters is obtained through sensors based on the Internet of Things (IoT), where these sensors will collect data through detection in patients [6]. After that, the system will send this data to the server through services connected to the internet network. After that, this data will be presented through a website-based dashboard. This allows doctors to monitor the patient's condition with minimal interaction and can help reduce the burden on hospitals in dealing with patients affected by Covid-19 [7]. Although there has been a lot of research done on health monitoring systems, due to the nature of the Covid-19 pandemic that has recently emerged, there is rarely a specific system focused on monitoring Covid-19 in general. The majority of existing systems tend to focus on monitoring one or a few physiological parameters only. More general health monitoring systems tend to be widespread. Many of these lead to wearable devices capable of detecting multiple physiological parameters at once. However, not all of these systems integrate IoT or rely solely on local data storage. In monitoring Covid-19, a system must be able to accommodate several parameters that are closely related to the development of Covid-19 disease. IoT integration is also becoming essential to enable remote monitoring, and ideally, systems should operate in real-time [5] . The Model-View-Controller (MVC) architecture has proven successful in software development by separating business logic, views, and data modeling [8]. On the other hand, RESTful APIs have become standard in communication between applications, allowing standardized and easily accessible

data exchange via HTTP protocol [9]. The Laravel framework, with its strong features and active community, provides an attractive platform for web application development. However, although there are many frameworks and development methods available, the right selection to build a Covid-19 patient condition monitoring system remains a challenge [10].

2 Literature Review

2.1 Covid-19

Covid-19 is a term introduced by the World Health Organization (WHO) for a virus that attacks the condition of individuals first identified in Wuhan, China in late 2019. Symptoms that can appear in this disease include fever, cough, as well as difficulty in breathing, and if not managed appropriately, can have serious consequences [11].

In efforts to monitor Covid-19 patients, there are several variables that can be used to monitor the patient's condition. These parameters include electrocardiography (ECG), heart rate (HR), respiratory frequency (RR), blood oxygen level (SpO2), as well as body temperature. Data from these parameters is collected through Internet of Things (IoT)-based sensors, where these sensors take data from patient detection, and the data is then stored in a website-based dashboard. Thus, doctors can monitor patients with minimal interaction and reduce the burden on hospitals in accommodating and caring for patients affected by Covid-19 [6].

2.2 Software Engineering

Software Engineering is a scientific discipline that integrates procedures, methods, and equipment in an effort to develop computer software. This discipline is closely related to the creation of software in a structured, systematic, and measurable manner. This process involves stages ranging from planning, analysis, design, deployment, testing, to software maintenance. In addition, the scope of software engineering also includes the principles and techniques applied to produce high-quality software. Aspects such as software project management, software quality evaluation, as well as software trials are also integral parts of software engineering [12].

In another concept, this discipline addresses all elements of software development, from the initial idea to the operational and maintenance phases. This principle ensures that the resulting software is in accordance with the needs of the user and operates optimally. A systematic and structured approach is also applied in software development to achieve the expected results [13].

2.3 Software Development

Software development (also known as application development, software design, software design, software development, software application development, enterprise application development, or platform development) is the development of software products. This process has its own complexity and often involves large financial investments as well as the participation of many individuals. Activities in software

development involve the application of design concepts through the creation of software code, database settings, and relevant content elements. In the context of software development, there are five main stages which include requirements, analysis and design, implementation, documentation, testing, as well as deployment, installation, and maintenance [14]. Efforts to design software consistently with time and cost efficiency can be realized through the adoption of popular and structured software development methodologies [15]. Software development methodology is a combination of practical approach and prevailing principles. Some examples of software development methodologies include Scrum, Waterfall Model, Rapid Application Development (RAD), and Spiral Model [16].

2.4 Software Architecture

In the software development phase, accurate planning includes not only problem analysis and software design, but also considers the selection of architectures that fit the purpose and scope of the project. It is important to choose the right type of architectural pattern to optimize deployment efficiency and form repeatable and redeployable solutions to address common challenges that may arise during the implementation process or in specific situations. Various types of architectures, such as client-server, Model-View-Controller (MVC), monolith, and microservices, can be adopted to meet project needs and face challenges that may arise [17].

In the framework of the application development model using the MVC pattern, elements such as Model (which pertains to the database), View (for visual display), and Controller are separated. The Model is used to manage the data, the View is in charge of setting the view, while the Controller manages the handling of requests from the client [18]. A very well-known structure in application development, especially web applications, is the Model-View-Controller (MVC) [19]. The use of the MVC concept allows the web application development process to be broken into two components, namely components related to the user side (front-end) and components that reside on the web server (back-end). The front-end and back-end parts of a website can be implemented using different frameworks [17].

2.5 Application Programming Interface (API)

An Application Programming Interface (API) is a set of instructions, functions, and protocols used by developers in the process of creating software for a specific operating system. APIs make it easy to exchange information and data between various software applications. API utilization eliminates the need to adapt to existing platforms when new platforms are added. APIs have definitions based on established standards and regulations to serve requests from the platform efficiently. Other advantages of the API include its ability to handle security aspects and access permission settings, as well as the ability to be accessed by other developers as an open API [20]. A REST API, also known as a RESTful API, is an implementation of an API (Application Programming Interface) that is based on the Representational State Transfer (REST) architecture. REST is a communication approach that uses the HTTP protocol for data exchange, and is often adopted in application development with the aim of creating systems that

have optimal performance, responsiveness, and are easy to develop, especially in terms of data exchange and communication. Each element in the system is identified through a unique Universal Resource Identifier (URI), and operations are performed via HTTP methods such as GET, PUT, POST, and DELETE. RESTful API involves four main components, namely URL Design, HTTP Methods, Representation, and Stateless Characteristics. The advantage of REST API lies in the ease for developers to build programs using blocks that are already available and can be integrated by programmers. One important aspect of REST is the stateless server characteristics of interacting with each other, allowing each server in the group to serve clients on every request. The REST approach allows the construction of web applications that can be integratively connected and accessed easily [21].

3 Methodology

The method applied to organize this research framework is using the waterfall method approach. The choice of waterfall method in this study was based on the need for a structured approach in the development of Application Programming Interface (API) for monitoring the condition of Covid-19 patients. The emphasis on this method was chosen because it provides a chronological sequence of steps, from analysis to implementation, that matches the desired complexity of API development. In addition, the waterfall approach has the capability to produce scalable solutions, clear validation at every stage, and is suitable for projects with stable initial specifications, as well as integrating patient condition parameter data with planned dashboards.

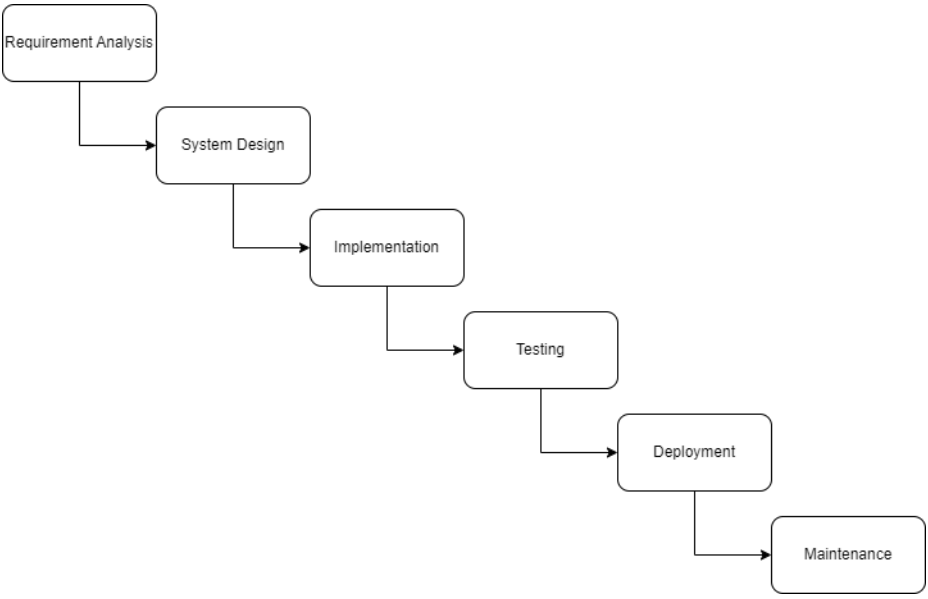

Fig. 1 Waterfall Model

4 Result and Discussion

4.1 Card Sorting

In this phase, communication was carried out with three health workers using open card sorting as a forum for collecting data and information provided by users. The data and information will then be analyzed to determine the system needs of the Covid-19 Patient condition parameter API. Thus the developed system can be in accordance with user needs. Here are the card sorting results presented in Table 1:

Table 1. Card Sorting

Feature Group	Frequency	Feature List
Main dashboard	3 of 3	Monitoring results of all patients, graph of the number of patients
	1 of 3	*Early Warning System*
Inpatient *management*	3 of 3	Patient management, ward management
	2 of 3	*Monitoring* of the patient's vital signs
Doctor's examination	2 of 3	*Discharge planning*
	3 of 3	Integrated Patient Progress Record
Nurse examination	3 of 3	Integrated Patient Progress Record
Main dashboard	3 of 3	Monitoring results of all patients, graph of the number of patients

4.2 Feature Requirements

Based on the results of user needs for designing API parameters for Covid-19 patient conditions, a system needs analysis was carried out to determine functional and non-functional needs in developing APIs in this study. Functional needs are processes that are within the scope of the system. Meanwhile, non-functional needs are the needs of supporting devices so that the system can run properly.

This functional requirement refers to the need for designing API parameters for Covid-19 patient conditions. With these needs, it can describe what must be realized in the development of APIs for designing API parameters for Covid-19 patient conditions. These functionality requirements include:

1. The system is able to provide API for the main dashboard page in the form of patient data.
2. The system is able to provide APIs for inpatient department pages in the form of patient data and vital signs, as well as ward data.
3. The system is able to provide API for the Early Warning System page in the form of changes in conditions for patients who experience critical conditions.
4. The system is able to provide APIs for master page data in the form of doctor data, including patient data, and ward data.

This non-functional need refers to the need for an API system for designing API parameters for Covid-19 patient conditions. API systems require several non-functional needs, such as:

1. The API system runs on the PHP programming language using the Laravel framework.
2. Response API in JSON format.
3. API testing using Postman.

4.3 Architectural Design

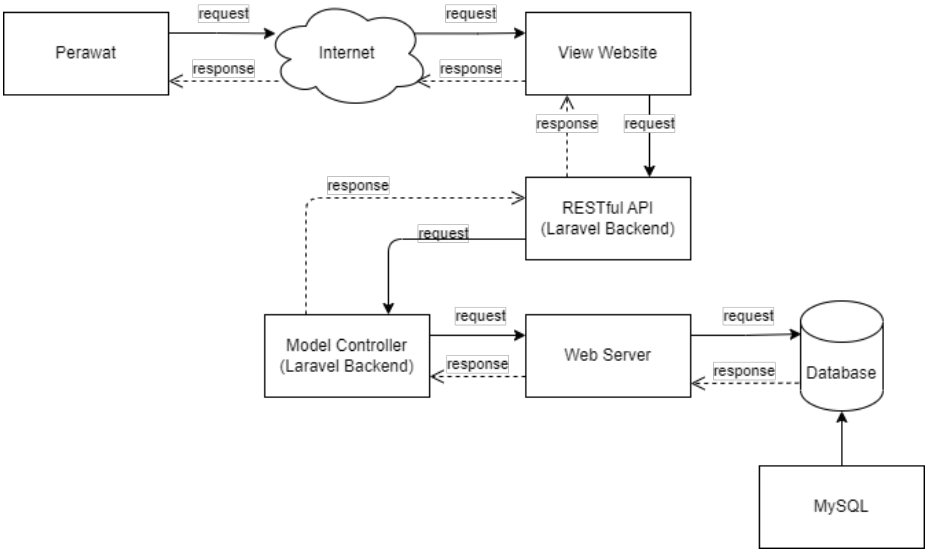

Fig. 2 API System Architecture

The following architecture is used in the API System for monitoring the condition of Covid-19 patients, users can access the website through a browser with an internet network, the browser makes a request through the url address accessed in the presentation view will make a response by displaying the website home page whose data is the result of the response from the RESTful API on the backend of the Laravel framework and the response will be forwarded to the model controller section for further processing to The model part to make server requests and requests to the database, the data from the request will be returned in the form of a response by the controller and displayed in the form of a view or presentation.

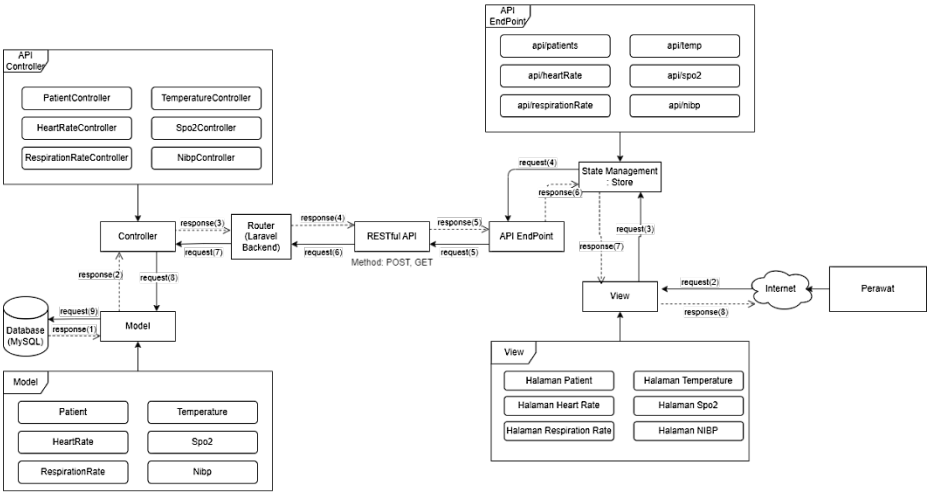

Fig. 3 MVC Architecture

The architectural design of the view controller model is shown in the architecture above, in the router section it is useful to provide access to certain pages so that they can be accessed through a web browser making it easier when moving pages from one page to another or to the opposite page, when the user accesses the website page, the first HTTP request to the controller to get an HTTP response in the form of a render view. The controller is useful for managing and validating data for the request process from the user, both between interconnected models and views, the use of controllers on the Covid-19 patient monitoring website in the form of API controllers to manage the backend. The model is useful for receiving requests from the controller and sending responses according to requests from user requests, in the form of data directly related to the database, be it in the form of patient, heart_rate, respiration_rate, temperature, spo2, nibp, ward, room, ews data. Models are also useful for relationships between tables by utilizing the eloquent feature. View is useful for displaying the rendering of website pages according to the request process from the user, so that this section is directly related to the user through HTTP requests that are set and directed by the router either moving pages from one page to another. RESTful API is useful for connecting the laravel framework (backend) with the Next.Js framework which can be used as a GET, POST, PUT, DELETE process so that it can exchange data via HTTP request and HTTP response with the presence of an API controller which is then connected to the API Endpoint.

4.4 Class Diagram

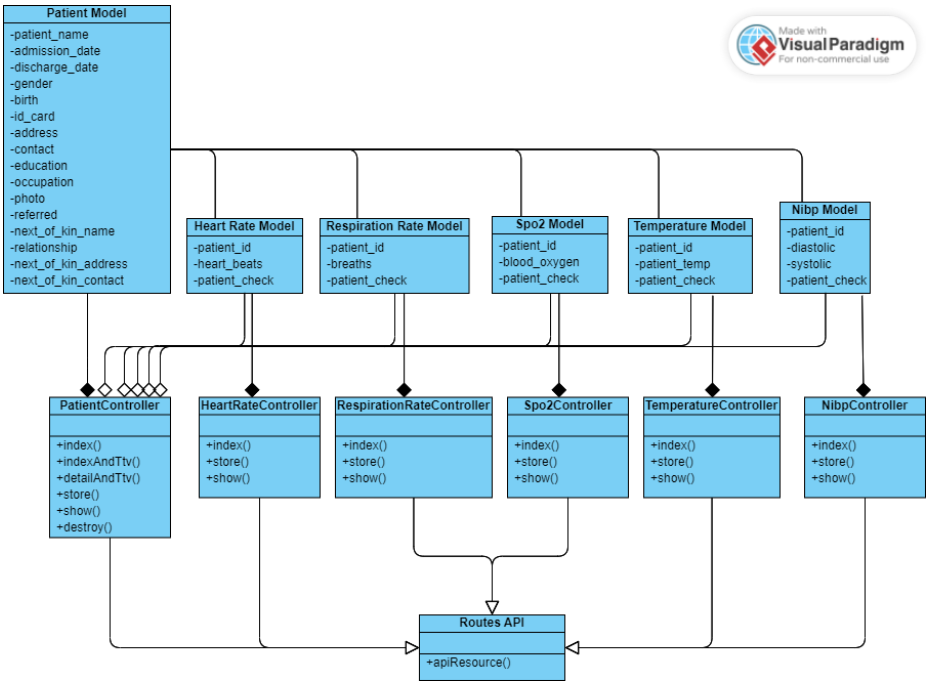

Fig. 4 Class Diagram

The diagram class illustrates the API design for Covid-19 patient condition parameters using the Laravel framework. Class diagrams are built on Model View Controller (MVC) principles to align with the selected Laravel work structure. In the class diagram, there are six models that represent the backend data to be used. In addition to these six models, there are also six controllers that function as a liaison between the model and the Application Programming Interface (API), and act as components for processing data. Furthermore, the routes API plays a role in defining how HTTP requests with specific methods and URLs will be handled by the controller or the appropriate action.

4.5 Entity Relationship Diagram

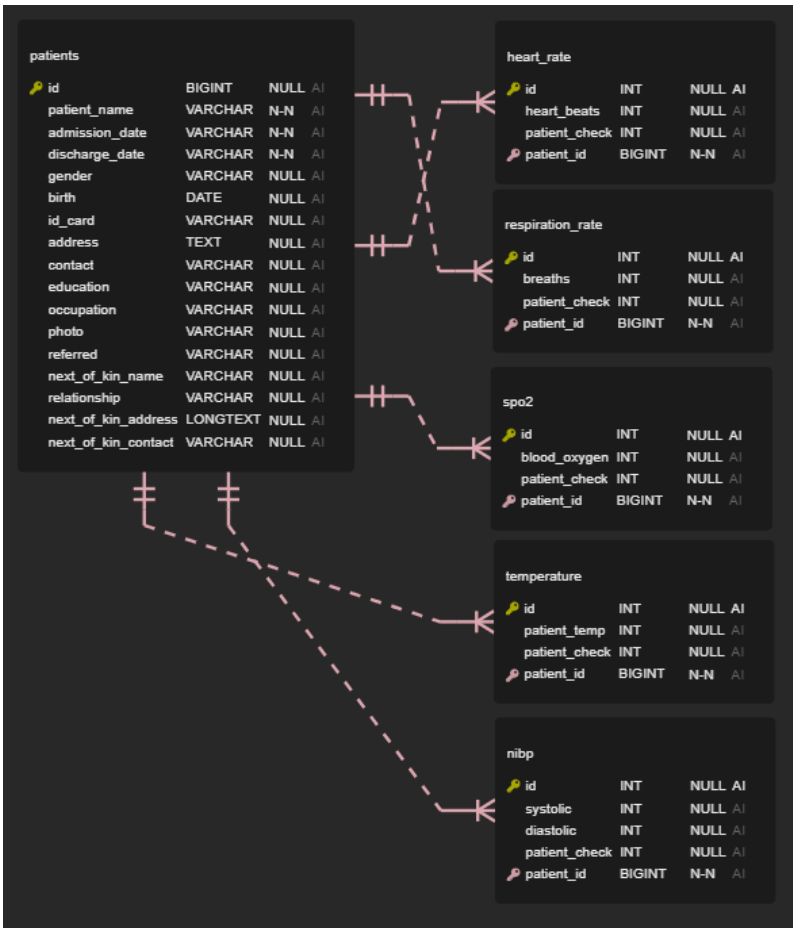

Fig. 5 ERD

In the context of this study, the Entity-Relationship Diagram (ERD) that has been generated based on the definition of tables and relationships provided has a crucial role in describing the structure and relationship between various entities in the database system under study.

4.6 API Endpoint Testing

Table 2 Patient API Table Endpoint

Method	*Endpoint*	*Function*	*Status Code*
GET	/api/patients	Display overall patient data	200 OK (Pass)
GET	/api/patients	Display one patient data by id	200 OK (Pass)
POST	/api/patients	Add patient data	201 Created (Pass)
DELETE	/api/patients	Delete patient data	200 OK (Pass)

Table 2 is the result of testing the Patient API using Postman tools. The methods used are GET, POST, and DELETE. All Results in Status Code display 200 OK and 201 Created which means that the test was successful.

Table 3 Endpoint API Table Patient Health Parameters

Method	*Endpoint*	*Function*	*Status Code*
GET	/fire/respirationRate	Displays respiration rate data from patients	200 OK (Pass)
GET	/api/respiration-Rate/{id}	Displays one respiration rate data from patients by id	200 OK (Pass)
POST	/fire/respirationRate	Add respiration rate data from patients	201 Created (Pass)
GET	/api/heartRate	Displays heart rate data from patients	200 OK (Pass)
GET	/api/heartRate/{id}	Displays one patient's heart rate data by id	200 OK (Pass)
POST	/api/heartRate	Add heart rate data from patients	201 Created (Pass)
GET	/api/temp	Displays respiration rate data from patients	200 OK (Pass)
GET	/api/temp/{id}	Displays one body temperature data from patient by id	200 OK (Pass)
POST	/api/temp	Add temperature data from patients	201 Created (Pass)
GET	/api/spo2	Display spo2 data from patients	200 OK (Pass)
GET	/api/spo2 /{id}	Display one spo2 record from patient by id	200 OK (Pass)
POST	/api/spo2	Add spo2 data from patients	201 Created (Pass)
GET	/api/nibp	Display NIBP data from patients	200 OK (Pass)
GET	/api/nibp/{id}	Display one NIBP record from patient by id	200 OK (Pass)
POST	/api/nibp	Add NIBP data from patients	201 Created (Pass)

Table 3 is the result of API testing of Patient Vital Signs such as Heart Rate, Respiration Rate, SpO2, Body Temperature, and NIBP using Postman tools. The methods used are GET, and POST. All Results in Status Code display 200 OK and 201 Created which means that the test was successful.

5 Conclusion and Future Work

The implementation of MVC architecture brings significant benefits in terms of separation of duties, allowing the development team to focus on certain aspects of the system without interfering with the functionality of others. It also contributes to easier maintenance and maintenance, as well as speeding up the development process with code reusability. Integration with RESTful APIs enables standardized and structured communication between various system components. The use of the waterfall method makes it easy to design APIs from parameter data such as heart rate, respiratory rate, oxygen saturation, body temperature, and Nibp into the dashboard system. This allows medical personnel to easily monitor the patient's condition in real-time.

Testing with black-box testing using Postman tools was successfully carried out where all endpoints managed to provide a successful response from all requests given. Thus, through this development, it is hoped that the results can make a meaningful contribution in increasing the efficiency and effectiveness of monitoring Covid-19 patients in hospitals. With a focus on designing Covid-19 patient parameter data APIs, this system can be a valuable tool for medical personnel in facing the challenges of this pandemic. Although black-box testing has been successful, be sure to continue ongoing monitoring and maintenance of the API. This will ensure that the API remains functional and responsive over time. Use IoT technology to transmit sensor data in real-time to backend systems or dashboards. This will allow medical personnel to monitor the patient's condition in real-time. By paying attention to and implementing these suggestions, it is hoped that the development of a Covid-19 patient monitoring system can be more focused and make a positive contribution to efforts to monitor patient conditions efficiently and effectively.

References

[1] A. C. Purnomo, P. Aliftiar, and Y. Darmawan, "Design a Covid-19 Information Dashboard with the RAD Method," vol. 7, no. 2, p. 2021.

[2] W. Vernandhes, N. S. Salahuddin, R. R. S. P. Sari, and T. Saptariani, "Happy Hypoxia Early Detection Tool in IoT Based for Covid-19 Patients Using SpO2 Sensor, Body Temperature and Electrocardiogram (ECG)," in *2021 6th International Conference on Informatics and Computing, ICIC 2021*, Institute of Electrical and Electronics Engineers Inc., 2021. DOI: 10.1109/ICIC54025.2021.9633002.

[3] Z. Bahaul Anwar, A. Widodo, and N. Kholis, "Internet Of Things Based Covid-19 Self-Isolation Patient Monitoring System."

[4] B. Alhmoud, D. Melley, N. Khan, T. Bonicci, R. Patel, and A. Banerjee, "Evaluating a novel, integrative dashboard for health professionals' performance in managing deteriorating patients: *A Quality Improvement Project*," *BMJ Open Qual*, vol. 11, no. 4, p. e002033, Dec. 2022, doi: 10.1136/bmjoq-2022-002033.

[5] A. N. Ejin, H. T. Yew, M. Mamat, F. Wong, A. Chekima, and S. K. Chung, "Internet of things based real-time coronavirus 2019 disease patient health monitoring system," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 12, no. 6, p. 6806, Dec. 2022, doi: 10.11591/ijece.v12i6.pp6806-6819.

[6] V. Khullar *et al.*, "IoT Fog-Enabled Multi-Node Centralized Ecosystem for Real Time Screening and Monitoring of Health Information," *Applied Sciences (Switzerland)*, vol. 12, no. 19, Oct. 2022, doi: 10.3390/app12199845.

[7] K. Zaballa *et al.*, "Social Response to Covid-19 SMART Dashboard: Proposal for Case Study," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Springer Science and Business Media Deutschland GmbH, 2022, pp. 154–165. DOI: 10.1007/978-3-031-09593-1_12.

[8] A. Alip, S. Kosasi, I. D. A. E. Yuliani, G. Syarifudin, and D. David, "Implementation of Model View Controller Architecture on Online Store Website," *Bumigora Information Technology (BITe) Journal*, Vol. 3, No. 2, pp. 135–150, Jan. 2022, doi: 10.30812/bite.v3i2.1566.

[9] O. M. A. AL-atraqchi, "A Proposed Model for Build a Secure Restful API to Connect between Server Side and Mobile Application Using Laravel Framework with Flutter Toolkits," *Cihan University-Erbil Scientific Journal*, Vol. 6, No. 2, pp. 28–35, Aug. 2022, doi: 10.24086/cuesj.v6n2y2022.pp28-35.

[10] R. Ridwan and M. Maisura, "DESIGN AND IMPLEMENTATION OF WEB SERVICES FOR THE INTEGRATION OF INFORMATION SYSTEMS FOR THE DISTRIBUTION OF Covid-19 CASES," *Cyberspace: Journal of Information Technology Education*, vol. 4, no. 2, p. 104, Oct. 2020, doi: 10.22373/cj.v4i2.7936.

[11] Handayani, D., Hadi, D. R., Isbaniah, F., Burhan, E., &; Agustin, H. (2020). Coronavirus Disease 2019. *Indonesian Journal of Respirology, 40*.

[12] R. S. Pressman and B. R. Maxim, "Software Engineering."

[13] I. Sommerville, *Software engineering*. Pearson, 2011.

[14] H.-P. Halvorsen, *Software Development A Practical Approach!* 2020. [Online]. Available: https://www.halvorsen.blog

[15] R. Narayan, "STUDY OF VARIOUS SOFTWARE DEVELOPMENT METHODOLOGIES," *EPRA International Journal of Multidisciplinary Research (IJMR)-Peer Reviewed Journal*, 2021, doi: 10.36713/epra2013.

[16] K. Risener, "A Study of Software Development Methodologies." [Online]. Available: https://scholarworks.uark.edu/csceuht/103

[17] T. Fadhilah Iskandar, M. Lubis, T. Fabrianti Kusumasari, and A. Ridho Lubis, "Comparison between client-side and server-side rendering in the web development," in *IOP Conference Series: Materials Science and Engineering*, Institute of Physics Publishing, Jun. 2020. DOI: 10.1088/1757-899X/801/1/012136.

[18] M. S. Singh, "MVC Framework: A Modern Web Application Development Approach and Working," *International Research Journal of Engineering and Technology*, 2020, [Online]. Available: www.irjet.net

[19] D. By, "APPLICATION OF MVC IN POINT OF SALES SYSTEM DEVELOPMENT (CASE STUDY OF TPOS PT. JAVASIGNA INTERMEDIA) INTERNSHIP TRACK FINAL PROJECT."

[20] "Application of RESTful Web Service and JSON on Application Programming Interface (API) Partnership-Based Broiler Development Information System Scientific Articles."

[21] F. Rizky Maulidy and S. Rohman Nudin, "DESIGN A WEB-BASED SIMDEPAD (INFORMATION SYSTEM FOR MONITORING AND EVALUATION OF CHILD DEVELOPMENT WITH DISABILITIES)," 2019.