Accredited SINTA 2 Ranking
Decree of the Director General of Higher Education, Research, and Technology, No. 158/E/KPT/2021
Validity period from Volume 5 Number 2 of 2021 to Volume 10 Number 1 of 2026

Published online at: **http://jurnal.iaii.or.id**

JURNAL RESTI
(Rekayasa Sistem dan Teknologi Informasi)
Vol. 8 No. 3 (2024) 321 - 332 e-ISSN: 2580-0760

A Middleware Applications Design for Health Information Sharing

Ketut Agus Seputra[1], A.A. Gede Yudhi Paramartha[2], Gede Aditra Pradnyana[3], Kadek Yota Ernanda Aryanto[4]*
[1,2,3,4] Department of Informatics, Faculty of Engineering and Vocation, Universitas Pendidikan Ganesha, Singaraja, Indonesia
[1]agus.seputra@undiksha.ac.id, [2]yudhi.paramartha@undiksha.ac.id, [3]gede.aditra@undiksha.ac.id,
[4]yota.ernanda@undiksha.ac.id

Abstract

The interoperability between electronic health records (EHR) and electronic medical records (EMR) from various healthcare facilities for comprehensive patient care is important. However, integrating such systems, including the need for interoperability standards, data privacy, and security, is a highly challenging task, especially since patient rights in data access must be considered. The primary problem addressed is the challenge of integrating electronic health records (EHR) and electronic medical records (EMR) across various healthcare facilities to ensure comprehensive patient care while maintaining data privacy, security, and adherence to patient rights. This work presents an innovative application to consolidate patient health records from various medical facilities. It facilitates seamless data access, improving the efficiency of healthcare delivery. The GGD approach was used in developing the prototype to ensure the delivered product was able to fulfil the user requirements. Four phases are divided into six stages used in this method: research, modelling, requirements definition, framework definition, refinement, and support. The evaluation involved two phases, back-end and front-end testing, utilizing white box and black box testing. Whitebox testing delivers the average frame rendering rate of up to 56 fps, and blackbox testing has shown 100% successful results in the given task. In conclusion, the Med-OID prototype was successfully developed. It integrates and securely transmits medical records across various healthcare services well, demonstrating significant potential to enhance personalized medicine and healthcare coordination. The evaluations underscored the prototype's robustness and its capability to improve interoperability and data sharing in healthcare systems.

Keywords: *healthcare; data sharing; information systems; PHR application; distributed systems*

How to Cite: K. A. Seputra, A. G. Y. Paramartha, G. A. Pradnyana, and K. Y. E. Aryanto, "A Middleware Applications Design for Health Information Sharing", *J. RESTI (Rekayasa Sist. Teknol. Inf.)*, vol. 8, no. 3, p. 321 332, Jun. 2024
DOI: https://doi.org/10.29207/resti.v8i3.5707

1. Introduction

The state of health and individual conditions, including genomic states, have shifted from population-based healthcare approaches to individualised approaches, known as Personalized Medicine (PM) [1]. PM is utilised to tailor interventions, enhancing stratification and timing in healthcare delivery by leveraging biological information and biomarkers in patients [2]. PM aims to identify the optimal treatment for each patient, maximizing therapeutic benefits while minimizing drug side effects.

PM involves detailed patient information, from medical history and previous treatments to potential drug reactions. Three types of information are essential, that is: (1) information representing the heterogeneity of prognosis and treatment effects, (2) information related to past diagnoses, and (3) verified evidence regarding the patient's health history [1]. This information is found in the patient's medical records, which are distributed across all healthcare services the patient has accessed.

However, the adoption of PM is not yet matched by the capability of Health Information Systems (HIS) to share information with patients or among HIS within hospitals [3]. It affects the Electronic Medical Record (EMR), which, if stored in different Hospital Information Management Systems (SIMRS), cannot be used collectively, especially when patients seek treatment at different hospitals or healthcare facilities. An EMR is a lifelong electronic health record containing information about health examinations written by one or more health professionals in an integrated manner at each patient encounter [4]. It indicates that Medical Records must be accessible from multiple computers in a network to provide integrated

Received: 05-02-2024 | Accepted: 18-05-2024 | Published Online: 01-06-2024

patient health information, ensuring efficient service and care [5].

The use of information technology in supporting healthcare services, including the provision of electronic medical records, has been pursued. Electronic medical records enable healthcare providers to store and share health information without relying on paper documents. This method is considered to reduce errors, enhance service quality, improve patient satisfaction, and reduce medical errors [6]. The regulation of electronic medical records in Indonesia is permitted, as stated in the Health Ministry Regulation No. 269 of 2008, Article 2, Paragraph (1), which requires records to be written entirely and clearly or recorded electronically.

The electronic medical record system is a critical focus for healthcare providers in managing integration and open data access among healthcare services, especially regarding medical record access [7]. According to Health Minister Regulation No. 46 of 2017 [8], strengthening health informatics includes standards and system interoperability. Therefore, it is clear that the developed health information systems must support data and information access across related systems by the authorized actors. The Health Ministry's strategic plan for 2020-2024, stated in Regulation No. 21 of 2020 [9], also mentions that the development of health information technology is directed towards electronic medical record systems to support the exchange of patient medical summaries between hospitals and digitalizing medical records. The exchange of patient medical summaries greatly supports the expansion of online referral systems, including integrating private healthcare facilities into the referral system.

Several government hospitals, especially those partnered with BPJS Kesehatan, are currently using an integration feature released by BPJS called Bridging BPJS. Through this feature, hospital services have been connected, from referrals between healthcare services to diagnosis searches. From the patient's perspective, the presence of the mobile JKN has met the requirements set in Article 52 of the Medical Practice Law [10], which states that patients have the right to a complete explanation of the medical actions provided, including access to medical records. However, according to the Minister of Health Regulation Number 269 of 2008 concerning Medical Records [11], medical records can be copied and shared with family. Therefore, in this research, an exploration of the potential for interoperability of medical record data among healthcare services is conducted and integrated into a mobile application. Furthermore, using a role-based access scheme, a design for family access scenarios to medical records is provided in this work.

2. Research Methods

In this section, we deliver several related works and describe the methods used.

2.1 Related Work

Treating patients, especially those with chronic diseases, undoubtedly requires numerous consultations. Each consultation necessitates relevant patient health information related to their condition, including their treatment history. Moreover, patients often receive care from one healthcare facility to another and even across countries. Through the Electronic Health Record (EHR), the transfer of a patient's long-term healthcare information to a new provider can be facilitated via digital services [12], as illustrated in Figure 1.

The Electronic Health Record (EHR) is basically a patient's long-term health record composed of various Electronic Medical Records (EMR) from different healthcare facilities. Meanwhile, an EMR is an electronic record of a patient's healthcare created, compiled, managed, and consulted by doctors and official staff within a single healthcare service [12]. Therefore, interoperability is essential to enable all healthcare services providing EMR to exchange data with each other. However, connecting information across healthcare services is not straightforward; at least four aspects must be ensured before developing system integration: interoperability standards, data exchange terminology, data privacy and security, and related regulations.

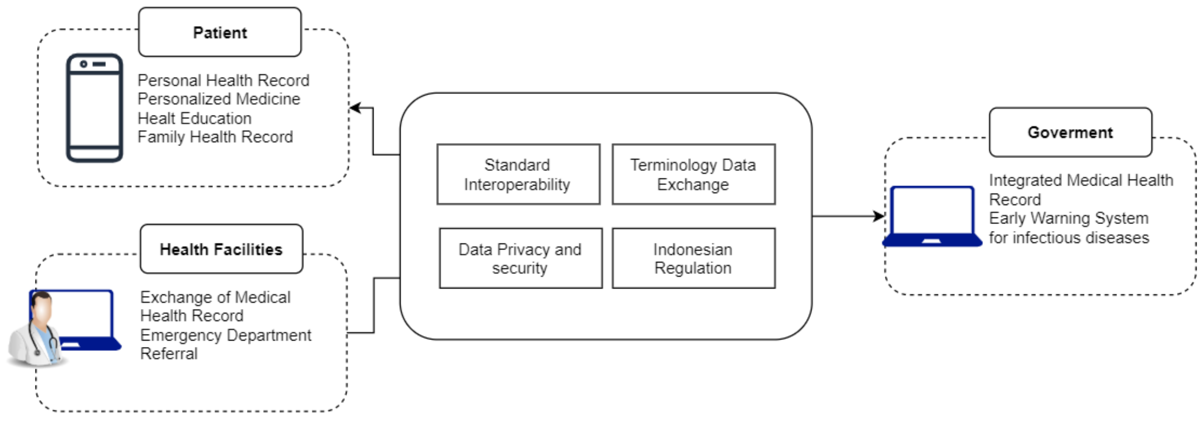

Figure 1. Flow of Healthcare Information

Some researchers have recently utilized blockchain architecture to develop EHR Sharing. Shufen et al. [13] stated that full medical record sharing allows doctors to make quick and accurate diagnoses for patients and improves hospital efficiency. However, given the highly transparent nature of blockchain, shared data must be encrypted [14]. Jack Huang et al. [15] added that the development of EHR should prioritize patient rights, especially in granting and withdrawing access consent to doctors. Additionally, EHR should not display detailed medical record information but rather provide an overview of the patient's condition over time. From a technical perspective, the developed EHR must ensure access reliability, maintain patient security and privacy, and enhance efficiency and scalability. Nevertheless, there are still many challenges in its implementation, both from the user perspective and the technology infrastructure of blockchain in healthcare [16].

2.2 Method

The research used a Research and Development (RnD) approach to find solutions to practical problems by developing a prototype product [15, 16]. This study focuses on the ultimate goal of the problem: to develop a Personal Health Record (PHR) application to integrate medical record access stored across multiple healthcare services. The prototype development approach naturally influences the system development method. The prototype was built as a mobile application using a cross-platform approach with the Flutter framework and supported by a Laravel back-end. This type of prototype is referred to as a native prototype. Native prototypes provide a more accurate representation of the final application because they use the same tools and frameworks as the final product and are helpful for testing performance, usability, and the overall user experience of the final application [19].

Several common prototype development methods include Traditional Information Technology Design (TID), Activity Centered Design (ACD), User Centered Design (UCD), Participatory Design (PD), and Goal Directed Design (GDD). Compared to other methods, GDD is more user-goal-focused [20], making it easier to determine user needs [21]. Therefore, in this work, GGD was chosen as the approach to developing the prototype to ensure the delivered product is able to fulfill user requirements. There are four phases that are divided into six stages of prototype development using the GDD approach: research, modeling, requirements definition, framework definition, refinement, and support [22], as illustrated in Figure 2.

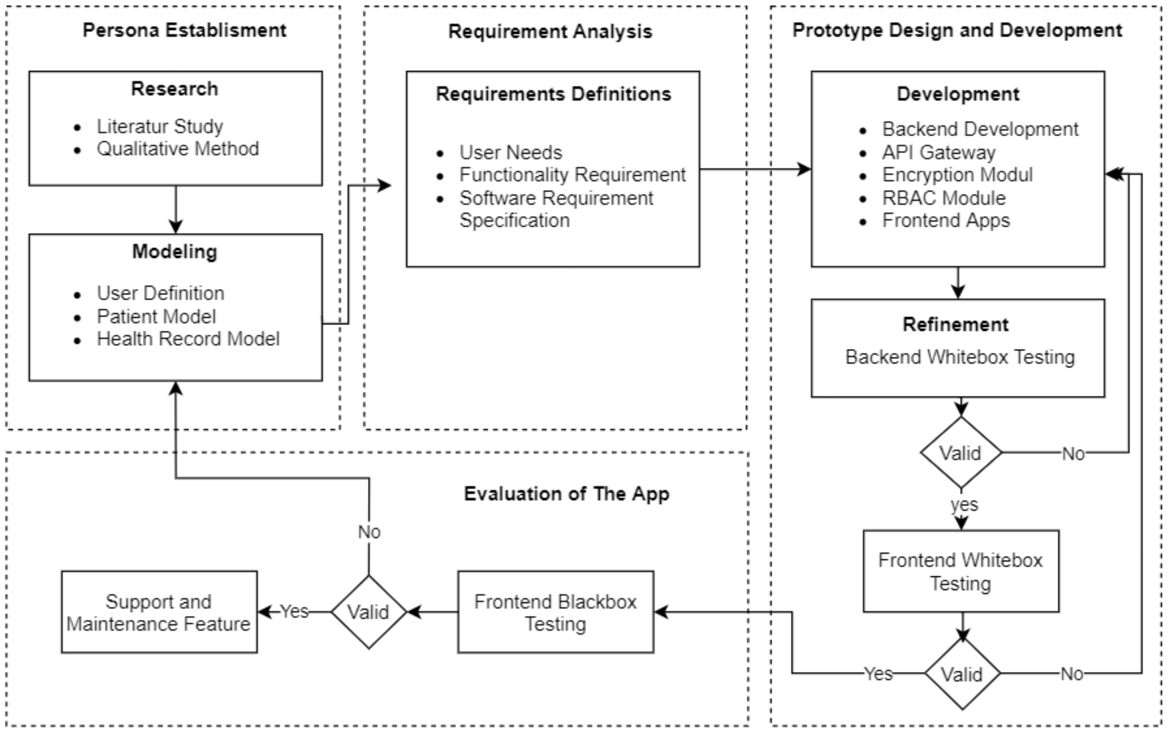

Figure 2. Workflow of the GDD Method

Qualitative data is collected through literature studies and field observations in the research stage. This data is then analyzed, among other things, to determine user needs. The output of this activity is a user model that represents the user profile, habits, needs, and user groups. The user model serves as key information in the requirements definition phase to determine the features needed in the application. Once the user needs and application features are validly mapped, this information forms the basis for developing the application design during the framework definition stage. In general, software design will conceptualize the overall product and determine the application's basic framework and visual design. The output of this stage includes the system design and application prototype. The resulting prototype is then implemented and tested during the refinement stage. This stage is crucial, among other things, to test whether the prototype meets the needs and can be used effectively. Testing methods include whitebox and blackbox testing. The results of the testing are used to improve the evaluation phase.

2.3. *User Scenario Overview*

Scenarios are introduced to deliver clarity on how the developed application works. Figure 3 illustrates the referral scenario for an 18-year-old male patient from a type B hospital to a type A hospital.

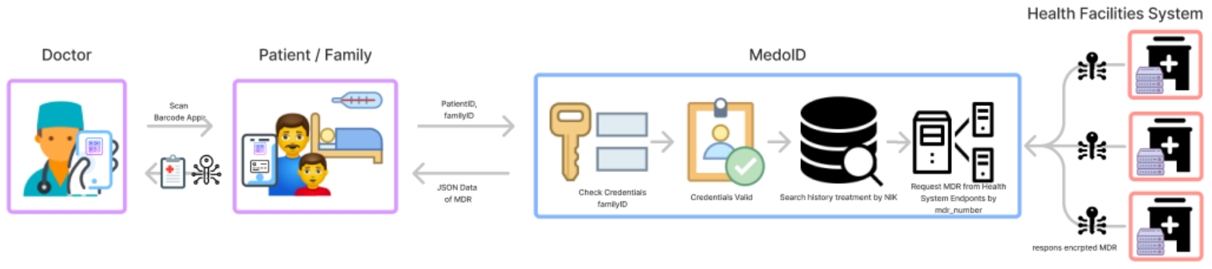

Figure 3. User Scenario Overview

The details of the scenario provided above can be described as follows.

Scenario 1: Registration. The patient registers at the admission department of a type A hospital. Immediately after registering, the on-duty doctor attends to the patient.

Scenario 2: Barcode Scan. Hospital B has conducted preliminary examinations regarding the patient's condition and decided to refer the patient to hospital A. To expedite further treatment at hospital A and with the patient unconscious, the family provides access to the patient's medical records to the on-duty doctor via an app. The doctor can then view the details of the medical records through the app by scanning a barcode presented by the patient's family.

Scenario 3: Record Access. The barcode is an access code that can be shared with the doctor by the patient or the patient's family. The barcode can contain detailed patient information, examination history, and medical records across all healthcare services. It could also be just a code for the details of the referral medical records.

Scenario 4: Role Access. The patient can determine the privacy of medical record access to other users who are considered family. Not only that, but the patient can also set the level of medical records to share.

3. Results and Discussions

In this stage, the application was developed, tested, and evaluated. The tests were done to provide performance for the back-end and front-end functionality.

3.1 Results

In the needs analysis, problem identification and qualitative data collection were conducted on the object of study using field observation methods and literature studies. The primary goal of developing this prototype is to find a solution for accessing medical records stored across multiple healthcare services through a single application by doctors, patients, and patient families.

Table 1 provides the user model for the prototype requirements.

Table 1. User Models

User	Requirements	Features
Doctor	Add and view the medical records	• Add records • View records Update records
Patient	View and grant access to medical records for family representatives	• View records Share records
Family	View medical records	View records
Operator	Setting for data integration between health information systems	Settings

Based on these findings, a system is needed that can act as a connector between healthcare systems and provide information related to patient's health history records. This health history includes the personal information of the patient and the medical record history compiled from several storage sources in healthcare services, commonly referred to as the Patient Health Record (PHR).

PHR ensures the availability of information and can be easily accessed anytime and anywhere when needed, including through gadgets [23]. It also considers various aspects related to the legality of medical record access, as stated in Article 47 of the Medical Practice Law [10], which indicates that medical records are owned by doctors, dentists, or healthcare facilities. Meanwhile, the contents belong to the patient and must kept confidential by the doctor or healthcare service. PHR is a detailed medical model and dashboard for patients to set access permissions and control health records in one place [23]. In addition to maintaining patient privacy, access control must also adhere to applicable laws.

There are four roles with distinct functions for the system: the patient, the family, the doctor, and the operator.

As regulated in Article 52 of the Medical Practice Law [10], patients have the right to receive a complete explanation of the medical actions provided, including access to their medical records. Furthermore, regarding access authority, as stated in Article 12 paragraph (4) of the Minister of Health Regulation number 269 year 2008 regarding Medical Records [11], a summary of the medical records can be provided, recorded, or copied by the patient or a person authorized by the patient or with the patient or family's written consent. Therefore, besides viewing medical records, patients can also manage the content of the medical records that can be shared and with whom it can be shared.

In principle, the right to give consent and refusal belongs to the concerned patient. However, in conditions where the patient is under guardianship, consent or refusal of medical actions can be made by the closest family members such as spouse, biological parents, children, or siblings. It refers to Article 45, paragraph (1) of the Medical Practice Law [10]. Thus, the family can record or copy the medical records in such conditions. Therefore, besides acting as a patient, a user can also act as a patient's family member. When acting as a patient's family, the user can view the patient's medical records according to the given access and then show the medical record access to the doctor as a barcode.

Subsequently, doctors can scan the barcode to view the patient's visit history and medical records. All-access history by doctors is displayed in a report format on the patient's dashboard. Meanwhile, EHR compiles medical records from various health IT sources through API endpoint configuration. Therefore, operators in each healthcare service play a role in configuring the API system for each health IT, which are then referred to as clients of the PHR and vice versa.

These four access levels will form the basis for developing user authorization settings using the Role-Based Access Control (RBAC) method.

The PHR application prototype, developed for mobile use, was subsequently named Medical Object Identifier (Med-OID).

Indonesia has successfully implemented the integration of health data through an application platform, one of which is the PHR Mobile JKN application. However, some features have not yet been included in this app, such as access to medical records for family representatives. To address the problem, a prototype of the PHR application, namely Medical Object Identifier (Med-OID), has been developed. It aimed to overcome the shortcoming by designing role-based access using QR codes that can be shared by patients with doctors and family. This feature is crucial, especially during emergency situations where the family representative is needed to make decisions. Furthermore, the decision should be based on the patient's health history. Therefore, the access granted to the family representative is essential.

The One Health Data architecture that was developed adopts a Central Repository System model. However, it has a disadvantage in that it requires expensive data center infrastructure. Therefore, Med-OID is exploring the potential for interoperability with an independent repository system approach. In this approach, the health history data remains stored at each healthcare service but can be accessed when needed by users who have already been granted access. Figure 4 provides a simple illustration of the proposed model.

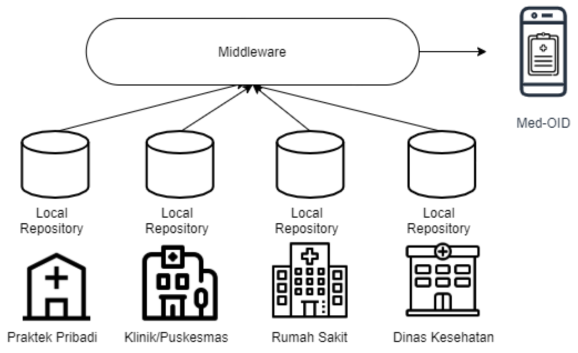

Figure 4. Independent Repository System

The users of the Med-OID application include patients, doctors, patient families, and operators. Patients and their families are general users who gain access to features such as updating personal data, health history, and referrals. As general users, doctors have access to features similar to patients, with additional access to search for patient information and medical records. Meanwhile, operator users can access services like patients with additional access to patient registration, visit history, and API configuration for Health Information Systems (HIS). Patients and doctors can use the system directly through the Med-OID website, mobile app, or HIS, which utilizes the Med-OID API. The Med-OID API used by HIS is intended to supply data to Med-OID, such as user registration and patient visit history. The second function is to supply data to HIS to obtain various information such as patient identity, visit history, and detailed medical record history. The details of the designed business process correspond to the flow diagram in Figure 5.

After the scenarios and user flows have been well mapped, the next step is identifying the key features required in the PHR application. One of the references for developing this PHR prototype is based on the Patient Care Coordination (PCC) domain of Integrating the Healthcare Enterprise (IHE) [24]. The data exchange authentication mechanism also refers to the Patient Identifier Cross-Reference Manager using a data-sharing architecture with middleware as a RESTful API [25].

There are three key features in the developed prototype: (1) the Patient Identifier Cross Reference, (2) the Exchange of Personal Health Record (XPHR), and (3) Emergency Department Referral (EDR). Through the Patient Identifier Cross-Reference feature, the system can identify and map the data sources of patient medical records. The XPHR provides for exchanging health record information used by patients across systems used by other healthcare providers based on the mapped patient profile. Meanwhile, the EDR supports the creation of patient referrals, including examinations conducted, past medical history, and medications prescribed. Thus, upon arrival at the Emergency Department (ED), the patient is identified as a referral and immediately given further treatment.

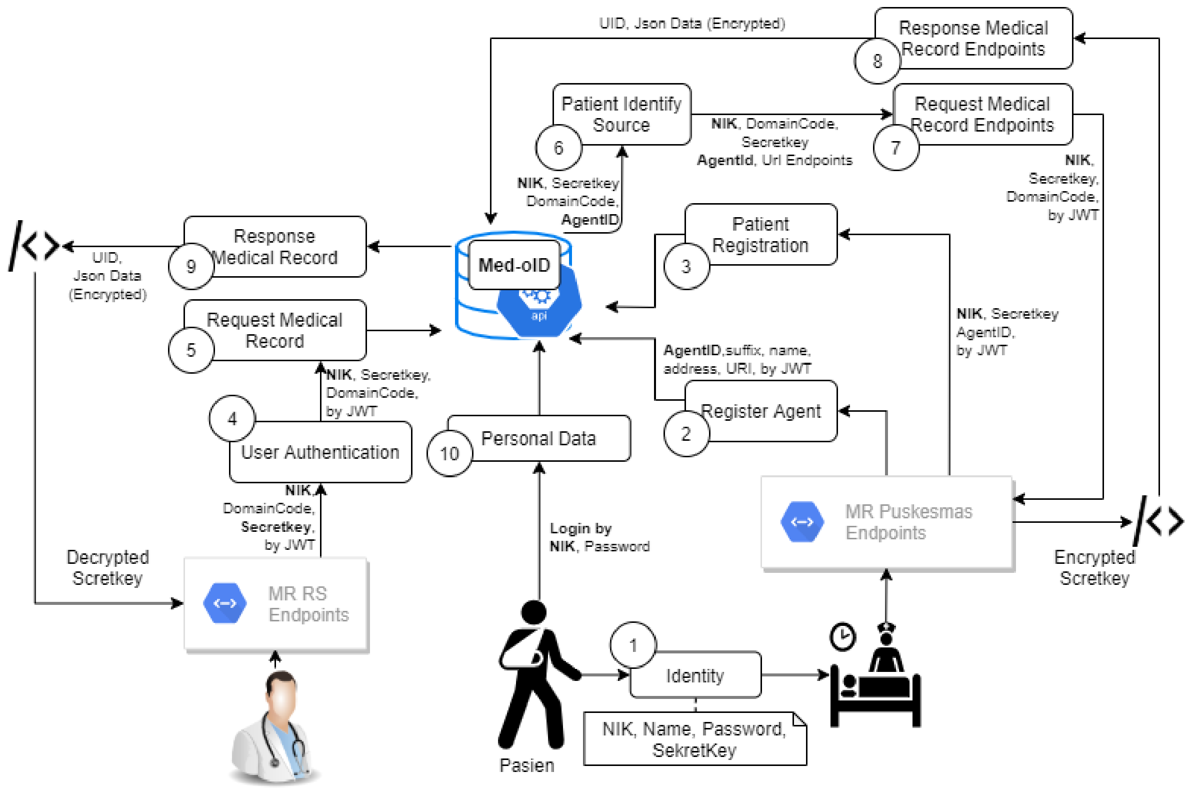
UID, Json Data (Encrypted)
8
Response Medical Record Endpoints
Patient Identify Source
6
NIK, DomainCode, Secretkey AgentId, Url Endpoints
Request Medical Record Endpoints
7
NIK, Secretkey DomainCode, AgentID
NIK, Secretkey, DomainCode, by JWT
Response Medical Record
9
UID, Json Data (Encrypted)
Med-oID
api
Patient Registration
3
Request Medical Record
5
NIK, Secretkey AgentID, by JWT
AgentID,suffix, name, address, URI, by JWT
NIK, Secretkey, DomainCode, by JWT
4
User Authentication
10
Personal Data
Register Agent
2
NIK, DomainCode, Secretkey, by JWT
MR Puskesmas Endpoints
Login by NIK, Password
Decrypted Scretkey
Encrypted Scretkey
MR RS Endpoints
1
Identity
NIK, Name, Password, SekretKey
Pasien

Figure 5. Workflow of Med-OID

The features ensure accurate linking and efficient communication of health records across various medical systems and enhance interoperability, which shows its function as middleware. This integration is crucial for maintaining data consistency, supporting real-time information exchange, and facilitating swift emergency referrals, thereby significantly improving the quality of healthcare delivery and patient safety. As middleware, it acts as a vital conduit that connects disparate healthcare technologies, streamlining processes and ensuring compliance with industry standards. Figure 6 provides a general overview of the system's functional requirements.

In addition to being reliable, the developed application must prioritize patient interests by considering information security. Therefore, the system requirements must be met, including middleware functionality, availability, privacy, security, efficiency, and scalability. The middleware functionality should solely connect patients and their healthcare visit history without storing any medical records [7]. Patients should be able to manage access to their medical records by doctors and family members. All shared information must be encrypted end-to-end to ensure data protection and should be accessible whenever they are required.

Besides utilizing existing infrastructure, the PHR should not duplicate medical records but instead use them collaboratively. This approach can significantly enhance the efficiency of storage media costs. Furthermore, the middleware is designed with a relatively straightforward workflow, making it easy for Health Information System (HIS) Operators to configure data access settings.

The explanation of the use case described in Figure 6 is provided in Table 2.

Table 2 Use Case Description

Use Case Name	XPHR
Description	The doctor accesses the patient's EMR
Actors	Doctor, Patient, and family
Preconditions	The doctor and patient have logged in
Normal Flow	1. The patient displays their barcode profile 2. The family displays the patient's barcode profile. The doctor looks up the barcode.
Alternative Flow	If it fails to work 1. The patient's family requests access to the patient's EMR profile. The patient accepts permission.
Post Condition	1. The doctor looks at the past medical record history The doctor creates a new medical record.

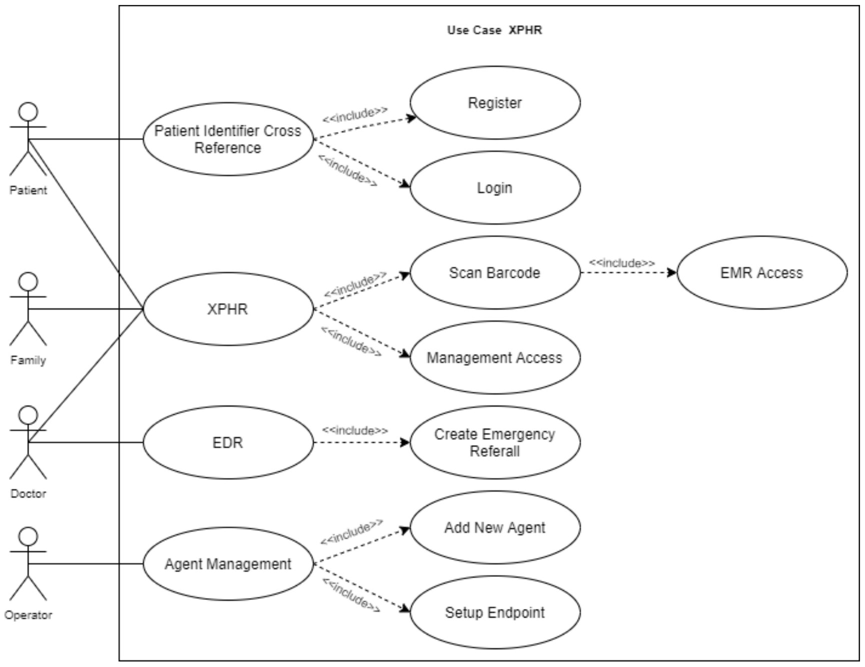

Figure 6. Use Case Diagram

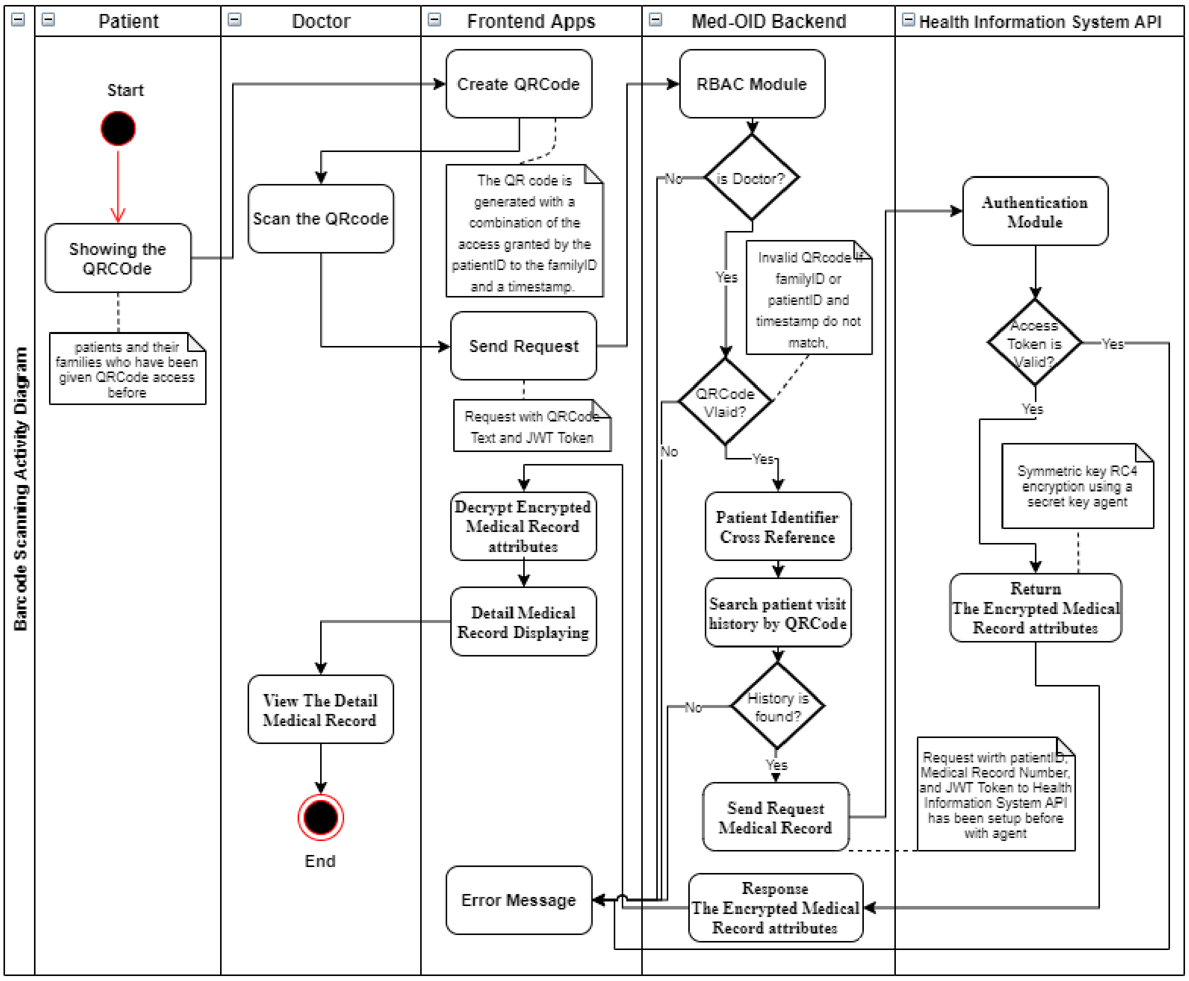
Barcode Scanning Activity Diagram
Patient
Doctor
Frontend Apps
Med-OID Backend
Health Information System API
Start
Showing the QRCOde
patients and their families who have been given QRCode access before
Create QRCode
The QR code is generated with a combination of the access granted by the patientID to the familyID and a timestamp.
Scan the QRcode
Send Request
Request with QRCode Text and JWT Token
Decrypt Encrypted Medical Record attributes
Detail Medical Record Displaying
View The Detail Medical Record
End
Error Message
RBAC Module
is Doctor?
No
Yes
Invalid QRcode if familyID or patientID and timestamp do not match.
QRCode Vlaid?
Patient Identifier Cross Reference
Search patient visit history by QRCode
History is found?
Send Request Medical Record
Response The Encrypted Medical Record attributes
Authentication Module
Access Token is Valid?
Symmetric key RC4 encryption using a secret key agent
Return The Encrypted Medical Record attributes
Request wirth patientID, Medical Record Number, and JWT Token to Health Information System API has been setup before with agent

Figure 7. Activity Diagram of the Barcode Scanning

Information security and patient privacy must be a primary concern in developing PHR, especially in sharing sensitive patient information across cloud-based healthcare services. The storage of visit history and the exchange of medical record details should employ end-to-end encryption. All attribute values displayed on the user interface must undergo a decryption process on the front-end side using a secretKey that the Health Information System operator has previously set at each connected healthcare service. This study utilizes Unified Modeling Language (UML) as a guideline in designing the prototype. UML aids in identifying the requirements and the scope of features in the application, which are illustrated through visual models. Two of the most famous and essential diagrams are the Activity Diagram and the Sequence Diagram. [26]. Figure 3 illustrates the flow diagram of patient Barcode Scanning as one of the core values of PHR. The activity diagram is created following the workflow shown in Figure 7.

Med-OID development has three phases: the back-end development phase, the encryption phase, and the front-end phase.

The back-end development phase begins with the database design using MySQL and MongoDB as the Database Management Systems (DBMS). The next step involves preparing the website using the Laravel framework as the main business logic of the system. The back-end development consists of several main modules, including the development of middleware, the RBAC module, the Patient Identifier Cross Reference, and the authentication module.

Middleware plays a crucial role as a communication interface between Med-OID as the server and the Endpoints installed in each Health Information System (HIS) as the clients. Middleware also serves as a connector between front-end applications and the back-end. The endpoints provided through the API include (1) POST for patient registration, where patient registration can be conducted through the HIS; (2) POST for visit history data, where this data aims to register the patient’s visit history including the Epid ID, Citizen ID number, medical record number, and visit date; and (3) Request for visit history (XPHR) based on the patient’s Citizen ID number, intended to handle data requests from the front-end or third-party applications regarding visit history details, patient identity details, and medical record details.

The middleware is developed using Laravel’s API library and Auth Sanctum for API authentication using the JWT (JSON Web Token) concept. Therefore, all data exchange services, whether from or to the agent, must utilize JWT authentication. Data exchange between the back-end and front-end, including inter-service exchanges, uses JSON Data. JSON was chosen for several reasons: it is lighter than other data exchange formats, easier to implement, and independent of the user interface.

The RBAC is pivotal in managing user access to application features. The RBAC development unfolds in three stages, as depicted in Figure 8, beginning with determining roles and permissions. The second stage involves defining the context-policy repository to map the user’s access repository. The third stage is the development of access control, which patients can configure. Access control extends beyond managing content, location, and time access; it also includes mechanisms for access delegation. Access delegation emerges as a critical policy, especially in emergencies.

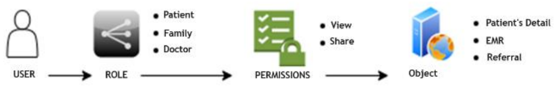

Figure 8. The RBAC Phase

The amalgamation of patient data from various healthcare services presents the potential for data inconsistency, necessitating a master data reference for patient identities. This module will serve as the master patient data, handling user registration independently and by operators. It also stores and identifies patient visit histories from each connected medical record service. The visit history data compiled from these Health Information Systems will then serve as a reference for the middleware to facilitate data exchange with the connected medical record services.

User management is a critical feature in data exchange between applications. The system must provide a secure data exchange service that aligns with the authorization granted to specific resources while safeguarding against unauthorized access attempts. Modern applications, typically developed following the frontend-backend programming paradigm, rely heavily on access provided through APIs, including API access granted to third-party systems. Therefore, protecting APIs from unauthorized access and other forms of misuse is crucial. JSON Web Tokens (JWT) manage access authentication from Med-OID to client APIs. JWT, a classic RESTful token with a straightforward workflow, is suitable for communication through RESTful APIs, offering the ease of implementation [27]. The authentication sequence diagram using JWT is succinctly illustrated in Figure 9.

At this stage, Med-OID, acting as a client, requests detailed medical record data of patients from the Endpoint Server. The Endpoint Server is an API prepared by the Health Information System (HIS), previously registered in the Patient Identifier Cross-Reference module by the operator. This module comprehensively stores endpoints to handle requests for detailed medical records and patient identity details, as well as an initial JWT token that is subsequently included in the header of each data request from Med-OID.

The next phase involves the implementation of end-to-end encryption at the agent’s endpoint as both client and data provider. End-to-end encryption ensures secure communication and prevents third parties from accessing data during its transfer from one system or device to another. The chosen encryption method is symmetric cryptography Rivest Cipher 4 (RC4). The RC4 algorithm is selected for its effectiveness and sufficient security for implementation [28]. Encryption is implemented by sending medical record data to the agent's endpoint server using a patient's secretKey, resulting in the transmitted information being in ciphertext. This information must then be translated through a decryption process using the same algorithm and key at the agent's endpoint server.

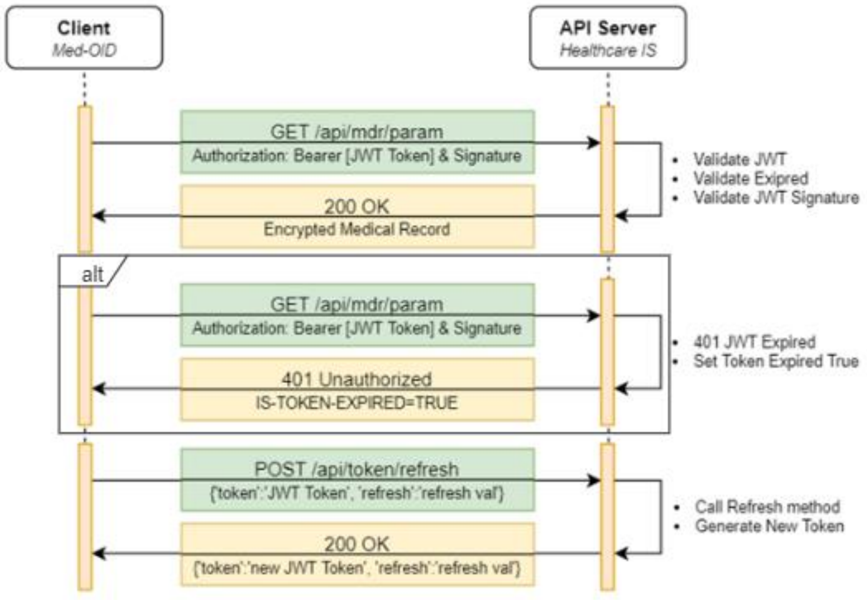

Figure 9. The Sequence Diagram of JWT Client

The final phase is front-end development. This phase begins with developing the User Interface (UI) on the client side. A website based on a Single Page Application (SPA) is developed using Vue.js. Websites developed with SPA do not require page reloads; thus, they have low bandwidth consumption and fast navigation [29]. Subsequently, mobile application development employs the Flutter framework with a Model–View–View-Model (MVVM) architecture. Flutter, a cross-platform widget-based app [30], is known for its efficiency and ease in developing high-performance mobile applications [31], [32]. Supported by Google with high standardization and security [30], [32], [33], Flutter ensures neat syntax and SDK consistency. Besides facilitating developer collaboration, the MVVM model also prevents code duplication in related functions, and the separation of logic function and UI speeds up the application testing process [34]. Figure 10 illustrates the data exchange architecture between the front-end and back-end, including between systems related to using Med-OID services.

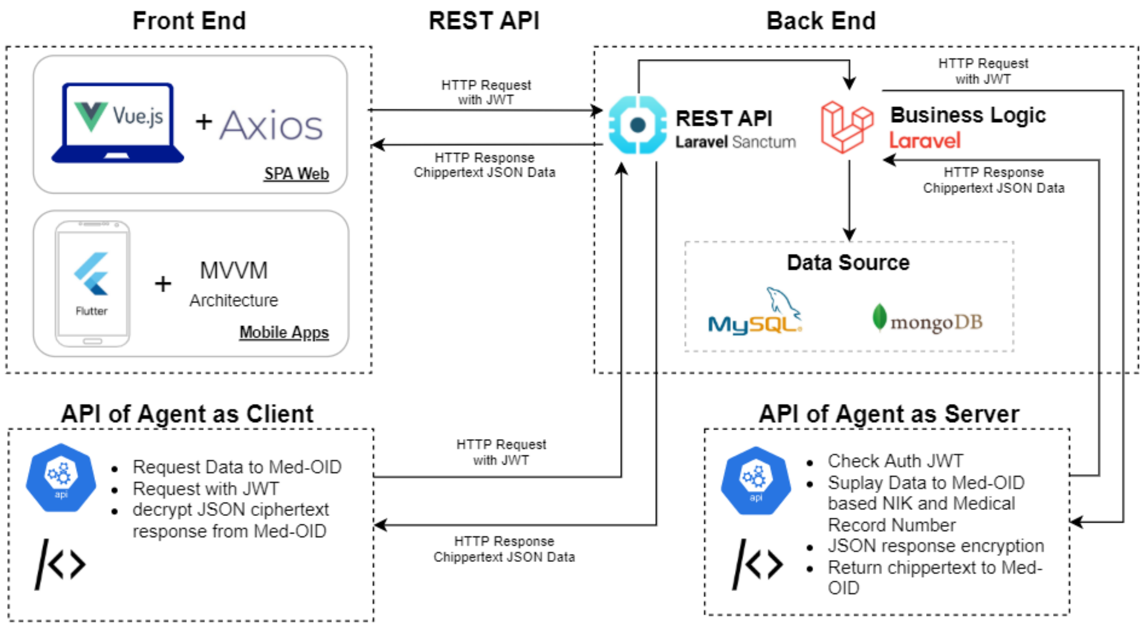

Figure 10. Software Architecture

The testing of the PHR application is conducted in two stages: Back-end testing and front-end app testing. Back-end testing uses black box testing with Postman to determine the success of the features implemented in the communication between Med-OID and the Health Information System (HIS). The results of the black box testing with sample testing can be seen in Table 3.

Table 3 Blackbox testing result

Task	Result	Conclusion
POST client visit history	The visit history data is stored, and the server provides a 201 response.	Success
Retrieve medical records for the client.	The client API responds in the form of encrypted medical record data.	Success
Medical record data description	The system can decrypt the medical record data using the agent's secretKey.	Success
GET medical details of medical records with an expired token	Client API sent a 401 response	Success

Based on the testing results, all features related to the API between Med-OID and the client API functioned well according to the designed workflow. Subsequent testing was conducted on the front-end Apps side. The initial test employed whitebox testing using Dartdev Tools. Dart DevTools is a suite of debugging and performance analysis tools for Dart and Flutter. It provides insights into the behavior of Flutter and Dart code during runtime, which can be valuable for white-box testing. The testing concentrated on performance profiling as well as debugging and inspection. Performance Profiling focused on UI threads and Raster threads. The UI measurement pertains to the time the Dart Virtual Machine requires to render a 60 FPS UI. Meanwhile, Raster measurement relates to the time the graphic processing unit (GPU) needs to render a 60 FPS UI.

Each frame must be rendered in less than 16ms to achieve an optimal performance of 60fps [35]. Frames taking longer than 16ms are categorized as janky. Performance testing is conducted based on the test cases prepared according to the User Scenario Overview in Figure 3. Testing is performed 20 times to find the average performance for each feature. Referring to Table 4, the UI threads for the register, login, and update profile features fall into the janky category. Therefore, each feature's average frame rendering rate has not reached 60fps. Several improvements in the Flutter code are necessary to reduce janky frames, including reducing the size of image files and animations and minimizing the widget stack.

Table 4. Prototype Performance Testing

Fitur	AVG UI (ms)	AVG Raster (ms)	AVG FPS (fps)
Login	17	7.6	20
Create Barcode	17	7	25
Scan Barcode Patient Detail	17.0	16.1	55
PHR	42	11.5	56
MDR Detail	40	11	34

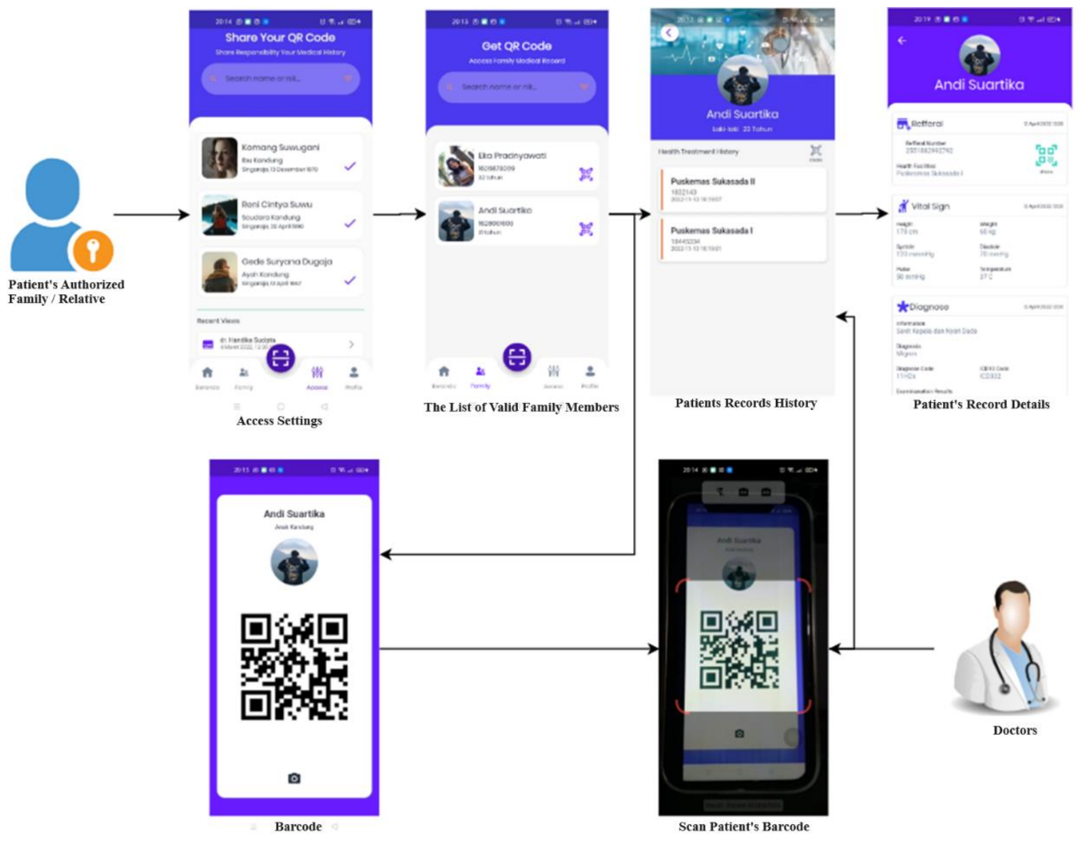

Figure 11. The Testing Scenario

Blackbox testing was conducted to assess the success of the developed application features. The testing ensured the application operated according to the planned scenarios and system design. The outline of the application testing scenario consists of five key phases. First, the patients share access to their medical records with family members. Second, the family members receive access to medical records. Third, a list of validated family members gains access to view patient details and medical records, allowing them to display the patient's barcode. Fourth, doctors or paramedics can scan the barcode to display patient details, examination history, and examination details. Last, the history and examination details data are sourced from each healthcare facility's data, accessed via APIs previously registered by each health facility's operator through a SPA-based web platform, following the previously designed architecture as seen in Figure 11.

3.2 Discussions

The Personal Health Record (PHR) application has been operating in accordance with the scenario; however, there are several notable observations, especially when involving third-party systems to display medical record details. This feature heavily depends on the health information system (HIS) at each healthcare facility in terms of access speed and data availability. Therefore, it is considered necessary to utilize local storage media on each patient's device to store the history of information access. Thus, information that has already been accessed does not need to be retrieved through the API again.

The level of readiness for technology adoption also varies across healthcare services. Some hospitals have been using desktop-based Hospital Information Management Systems (HIMS) developed by third parties. This situation could become a barrier if a new feature must be adopted and implemented in the HIMS, especially if the development contract has expired. Therefore, the next solution is to develop a specific middleware to directly handle communication from the Health Information System into the database.

A major issue in the data-sharing process relates to patient data privacy and security. In this work, a security scheme for transmitted data has been implemented. More in-depth analysis is still required to ensure the security of data-in-transit and data-at-rest, including data retention on the recipient's device, if any. Regulations related to the government's enforcing the protection of personal data must also be seriously considered for further development. Data security and patient privacy awareness training have become necessary for the involved parties.

Potential issues for the implementation of digital-based healthcare service systems were also found in the research process. First, limited resources will pose a challenge in developing similar systems. Undoubtedly, reliable resources are needed to operate and maintain this data-sharing platform. Second, ensuring patients understand and consent to their shared data becomes crucial. Building and maintaining trust that the data will be shared among the involved institutions poses a significant challenge.

4. Conclusions

In this work, a prototype of a patient health record application named Med-OID, designed to integrate medical records across various healthcare services, has been developed. The prototype demonstrates effective information transmission between systems, emphasizing the importance of data security, patient privacy, and the legal aspects of medical record access. The prototype evaluation has shown that the developed system is able to deliver the proper information transmission. Whitebox testing delivers the average frame rendering rate of up to 56 fps, and blackbox testing has shown 100% of successful results in the given task. Both testing methods were employed to evaluate the prototype's effectiveness and identify areas for improvement. The work has been successful and underscores the potential of technology to enhance personalized medicine and healthcare coordination through improved data sharing and interoperability.

Acknowledgements

The author declares no conflict of interest. This work was part of research supported by a research grant from the Ministry of Research, Technology and Higher Education Indonesia.

References

[1] W. Rogowski et al., "Concepts of 'Personalization' in Personalized Medicine: Implications for Economic Evaluation," Pharmacoeconomics, vol. 33, no. 1, pp. 49–59, Jan. 2015, doi: 10.1007/s40273-014-0211-5.

[2] S. Schleidgen, C. Klingler, T. Bertram, W. H. Rogowski, and G. Marckmann, "What is personalized medicine: Sharpening a vague term based on a systematic literature review," BMC Med. Ethics, vol. 14, no. 1, 2013, doi: 10.1186/1472-6939-14-55.

[3] S. M. Marier, "Potensi Interoperabilitas Sistem Informasi Rumah Sakit Untuk Penerapan Standar Pertukaran Data HL7," J. Sist. Inf., vol. 5341, no. October, pp. 2579–5341, 2018.

[4] D. C. A. Nugraha and I. Aknuranda, "An overview of e-Health in Indonesia: Past and present applications," Int. J. Electr. Comput. Eng., vol. 7, no. 5, pp. 2441–2450, 2017, doi: 10.11591/ijece.v7i5.pp2441-2450.

[5] E. Triyanti and E. Retna Weningsih, Manajemen Informasi Kesehatan III, Desain For. Jakarta: Pusat Pendidikan SUmber Daya Manusia Kesehatan Badan Pengembangan dan Pemberdayaan Sumber daya Manusia Kesehatan Kementerian Kesehatan Republik Indonesia, 2018.

[6] P. J. Schenarts and K. D. Schenarts, "Educational impact of the electronic medical record," J. Surg. Educ., vol. 69, no. 1, pp. 105–112, 2012, doi: 10.1016/j.jsurg.2011.10.008.

[7] K. Y. E. Aryanto, K. A. Seputra, I. N. S. W. Wijaya, I. W. Abyong, G. A. Pradnyana, and A. A. G. Y. Paramartha, "Towards Healthcare Data Sharing: An e-Health Integration Effort in Indonesian District," Proc. - 2021 Int. Semin. Appl. Technol. Inf. Commun. IT Oppor. Creat. Digit. Innov. Commun. within Glob. Pandemic, iSemantic 2021, pp. 280–284, 2021, doi: 10.1109/iSemantic52711.2021.9573250.

[8] Keputusan Menteri Kesehatan, "KMK RI no 4829 tahun 2021," no. 1635, 2021.

[9] Menteri Kesehatan Republik Indonesia, "Peraturan Menteri Kesehatan Republik Indonesia Nomor 21 Tahun 2020 Tentang Rencana Strategis Kementerian Kesehatan Tahun 2020-2024." 2020.

[10] Presiden Republik Indonesia, "Undang-Undang Republik Indonesia Nomor 29 Tahun 2004 Tentang Praktik Kedokteran," Aturan Prakt. Kedokt., pp. 157–180, 2004.

[11] Menteri Kesehatan Republik Indonesia, "Peraturan Menteri Kesehatan Republik Indonesia Nomor 269/Menkes/Per/III/2008 Tentang Rekam Medis." 2008.

[12] P. D. Jacob, "Management of patient healthcare information," in Fundamentals of Telemedicine and Telehealth, Elsevier, 2020, pp. 35–57.

[13] S. Niu, L. Chen, J. Wang, and F. Yu, "Electronic Health Record Sharing Scheme with Searchable Attribute-Based Encryption on Blockchain," IEEE Access, vol. 8, pp. 7195–7204, 2020, doi: 10.1109/ACCESS.2019.2959044.

[14] M. Usman and U. Qamar, "Secure Electronic Medical Records Storage and Sharing Using Blockchain Technology," Procedia Comput. Sci., vol. 174, pp. 321–327, 2020, doi: 10.1016/j.procs.2020.06.093.

[15] J. Huang, Y. W. Qi, M. R. Asghar, A. Meads, and Y. C. Tu, "MedBloc: A blockchain-based secure EHR system for sharing and accessing medical data," Proc. - 2019 18th IEEE Int. Conf. Trust. Secur. Priv. Comput. Commun. IEEE Int. Conf. Big Data Sci. Eng. Trust. 2019, pp. 594–601, 2019, doi: 10.1109/TrustCom/BigDataSE.2019.00085.

[16] D. B. Santoso, A. Fuad, G. B. Herwanto, and A. W. Maula, "Blockchain Technology Implementation on Medical Records Data Management: a Review of Recent Studies," J. Ris. Kesehat., vol. 9, no. 2, pp. 107–112, 2020, doi: 10.31983/jrk.v9i2.5742.

[17] L. Ma, L. Gu, and J. Wang, "Research and Development of Mobile Application for Android Platform," Int. J. Multimed. Ubiquitous Eng., vol. 9, no. 4, pp. 187–198, Apr. 2014, doi: 10.14257/ijmue.2014.9.4.20.

[18] S. Hooshangi, M. R. Arasti, D. A. Hounshell, and S. Sahebzamani, "Evolutionary learning methodology: A case study of R&amp;D strategy development," Technol. Forecast. Soc. Change, vol. 80, no. 5, pp. 956–976, Jun. 2013, doi: 10.1016/j.techfore.2012.08.017.

[19] D. Inupakutika, S. Kaghyan, D. Akopian, P. Chalela, and A. G. Ramirez, "Facilitating the development of cross-platform mHealth applications for chronic supportive care and a case study," J. Biomed. Inform., vol. 105, no. March, p. 103420, 2020, doi: 10.1016/j.jbi.2020.103420.

[20] A. Williams, "User-centered design, activity-centered design, and goal-directed design," in Proceedings of the 27th ACM international conference on Design of communication, Oct. 2009, pp. 1–8, doi: 10.1145/1621995.1621997.

[21] A. Cooper, R. Reimann, D. Cronin, C. Noessel, J. Csizmadi, and D. LeMoine, About Face: The Essentials of Interaction Design, 4th ed. Canada: Wiley Publishing, 2014.

[22] H. Duan et al., "Using Goal-Directed Design to Create a Mobile Health App to Improve Patient Compliance With Hypertension Self-Management: Development and Deployment," JMIR mHealth uHealth, vol. 8, no. 2, p. e14466, Feb. 2020, doi: 10.2196/14466.

[23] A. Tembhare, S. Sibi Chakkaravarthy, D. Sangeetha, V. Vaidehi, and M. Venkata Rathnam, "Role-based policy to maintain privacy of patient health records in cloud," Journal of Supercomputing, vol. 75, no. 9. pp. 5866–5881, 2019, doi: 10.1007/s11227-019-02887-6.

[24] I. PCC Technical Committee, "IHE Patient Care Coordination (PCC) 5 Technical Framework Volume 1 IHE PCC TF-1 Integration Profiles Revision 11.0-Final Text," vol. 1, 2016, [Online]. Available: http://www.ihe.net/PCC_Public_Comments.

[25] K. A. Seputra and L. J. E. Dewi, "A Design of Patient Registration Apps using Flutter, Laravel and, Vue JS," 2022, doi: 10.4108/eai.27-11-2021.2315532.

[26] H. Koç, A. M. Erdoğan, Y. Barjakly, and S. Peker, "UML Diagrams in Software Engineering Research: A Systematic Literature Review," in The 7th International Management Information Systems Conference, Mar. 2021, p. 13, doi: 10.3390/proceedings2021074013.

[27] R. Sreejith and S. Senthil, "Smart Contract Authentication assisted GraphMap-Based HL7 FHIR architecture for interoperable e-healthcare system," Heliyon, vol. 9, no. 4, p. e15180, Apr. 2023, doi: 10.1016/j.heliyon.2023.e15180.

[28] S. Sulistiyanto, I. Satriadi, and A. Rahman, "Electronic Archive Design With RC4 Cryptographic Based File Security," J. Comput. Networks, Archit. High Perform. Comput., vol. 6, no. 1, pp. 34–44, Dec. 2023, doi: 10.47709/cnahpc.v6i1.3298.

[29] M. Kaluža and B. Vukelić, "Comparison of front-end frameworks for web applications development," Zb. Veleučilišta u Rijeci, vol. 6, no. 1, pp. 261–282, 2018, doi: 10.31784/zvr.6.1.19.

[30] K. Shah, H. Sinha, and P. Mishra, "Analysis of Cross-Platform Mobile App Development Tools," in 2019 IEEE 5th International Conference for Convergence in Technology (I2CT), Mar. 2019, pp. 1–7, doi: 10.1109/I2CT45611.2019.9033872.

[31] W. Wu, "React Native vs Flutter, cross-platform mobile application frameworks," Metrop. Univ., no. March, p. 28, 2018.

[32] J. Christoph, D. Rösch, T. Schuster, and L. Waidelich, "Current Progress in Cross-Platform Application Development Evaluation of Frameworks for Mobile Application Development," Int. J. Adv. Softw., vol. 12, no. August 2021, 2019, [Online]. Available: http://www.iariajournals.org/software/2019,.

[33] G. Catolino, P. Salza, C. Gravino, and F. Ferrucci, "A Set of Metrics for the Effort Estimation of Mobile Apps," Proc. - 2017 IEEE/ACM 4th Int. Conf. Mob. Softw. Eng. Syst. MOBILESoft 2017, pp. 194–198, 2017, doi: 10.1109/MOBILESoft.2017.31.

[34] G. Imbugwa, M. Mazara, and S. Distefano, "Developing a Mobile Application Using Open Source Parking Management System on Etherium Smart Contracts.," J. Phys. Conf. Ser., vol. 1694, no. 1, 2020, doi: 10.1088/1742-6596/1694/1/012022.

[35] Flutter.dev, "Using the Performance view." Accessed: Sep. 01, 2021. [Online]. Available: https://flutter.dev/docs/development/tools/devtools/performance.