REST API Development of the Mataram Scout Information System for Member Management and Activity Reporting using Scrum Method

I Desak Putu Diah Anggiani*, Dr. Eng. Budi Irmawati, Royana Afwani
Dept Informatics Engineering, Mataram University
Jl. Majapahit 62, Mataram, Lombok NTB, INDONESIA
Email: diahanggiani08@gmail.com, budi-i@unram.ac.id, royana@unram.ac.id

***Corresponding Author**

The development of a web-based Scout Information System for Kwartir Cabang Mataram aims to improve member and activity data management through a REST API. Existing processes using Google Forms and Excel are inefficient due to manual input, fragmented storage, and lack of real-time access, causing errors and delays. To address this, the proposed system integrates all levels of the scout organization within Mataram City into a centralized platform. Using the Scrum method, it was developed through five sprints and refined based on stakeholder feedback. The REST API includes 24 tested endpoints, ensuring seamless frontend-backend integration and future mobile compatibility. Black-box testing confirmed endpoints reliability. A feasibility survey involving 23 scout representatives gave an average score of 81.74% ("Good"), showing approval and indicating that the system successfully improves data handling compared to the previous manual workflows. These results demonstrate enhanced operational efficiency and reduced administrative delays.

***Key words*: Scout Information System, REST API, Scrum Method, Data Management, Black-Box Testing.**

I. Introduction

Scout Movement has played a vital role in Indonesia youth development, to grow values such as independent, self confidence, leadership, and solidarity [1]. Officially established on August 14, 1961, the Scout Movement has grown into one of the largest youth organizations in the country [2]. According to Indonesia's Law No. 12 of 2010 concerning the Scout Movement (Undang-Undang Nomor 12 Tahun 2010 tentang Gerakan Pramuka), scouting is a non-formal education aimed at nurturing individual potential, fostering self-discipline, and building essential life skills to prepare the next generation to contribute meaningfully to make a significant contribution to the nation [3].

In Mataram City, scouting activities within school-based scout units (gugus depan) have experienced significant growth, driven by the increasing number of members and events organized. Kwartir Cabang Mataram, the scout organization at the city level, currently oversees 7 kwartir ranting (the scout organization at the sub-district level) and a total of 255 gugus depan, with more than 20,000 registered members. However, Kwartir Cabang Mataram does not yet have a centralized database to manage members data and activities. Discussions with the scout leader of SDN 26 Mataram, who also serves as the secretary of Kwartir Cabang Mataram, revealed that the existing data management system is still inefficient. Google Forms is used by each gugus depan to collect individual member and activity data, while Microsoft Excel is used to compile the data and generate summary reports at the sub-district and city levels. Although these tools provide a digital alternative to paper-based forms, they lack features for centralized access control, real-time synchronization, and efficient data retrieval across multiple levels of the scout organization. In addition, both gugus depan and kwartir ranting are required to prepare activity reports in printed (hardcopy) format. These reports are submitted in a hierarchical manner, where gugus depan units submit their reports to the kwartir ranting, which then compiles and forwards them to the kwartir cabang. This manual, paper-based workflow is time-consuming and prone to delays, inconsistencies, and data loss. Given these inefficiencies and fragmentation, there is a clear need for a centralized, integrated system that can streamline data collection, real-time synchronization, and reporting across all levels of Kwartir Cabang Mataram.

In today's digital era, leveraging information technology is essential for improving organizational efficiency, especially in managing member data and activity reporting. Digital transformation offers a promising solution to overcome the data management challenges faced by Kwartir Cabang Mataram. To implement this transformation, a web-based scout information system is being developed to support the multi-level organizational structure of Kwartir Cabang Mataram. As part of a collaborative project, the frontend of the system is developed by another member of the research team. This study specifically focuses on developing the backend system using REST architecture, which represents a core component of digital transformation by replacing fragmented, manual workflows with integrated, real-time digital services that improve data accessibility, and operational transparency.

The REST API, built with the Next.js framework, enables seamless data integration and real-time access across organizational levels through well-designed endpoints, data processing, database integration, and backend services to support frontend features. Operating over the HTTP protocol, REST is widely applicable both on the internet and within internal network environments [4]. REST simplifies interactions between distributed systems, improving data consistency and accessibility. The system is developed using the Scrum method, which emphasizes iterative development through sprints, continuous feedback, and collaboration among cross-functional teams [5]. This approach offers the flexibility to adapt to changing requirements and helps deliver a system aligned with user needs. Until now, no REST API-based scouting information system has been designed specifically to support the three-tier Scout Movement structure in Mataram. This research addresses this gap by developing backend features for managing member data and activity reporting with role-based access control, real-time synchronization, centralized database integration, and streamlined reporting workflows. With the implementation of this system, it is expected that the management of scout data in Kwartir Cabang Mataram will become more efficient.

II. Literature Review

In recent years, various studies have been conducted on the development of backend web-based information systems. As a result, a literature review of previous research relevant to the topic has been carried out to serve as a theoretical foundation and comparison.

Study [6] highlights the use of the Scrum method in developing an information system for retail sales at Rabbani Shoes. The research implemented Scrum to build four main modules: user login, product management, user management, and transaction management. The successful implementation of all planned features according to the product backlog, underscores Scrum's effectiveness in providing structured project management.

In line with the focus on the agile Scrum methodology, study [7] discusses the development and implementation of the back-end of a financial reporting system website at SMK Multistudi High School. The back-end was developed using JavaScript, designed to handle business logic and data operations, and supported by MySQL as the database management system. Through the application of Scrum, the web-based financial reporting system successfully delivered features for user registration, authentication, supplier data management, inventory control, sales tracking, and profit and loss reporting.

Another study [8] focuses on the digitization of inventory management through a web-based system developed for Widarapayung Kulon Village Hall. The system utilized a REST API built with Node.js to integrate the front-end and back-end. The REST API was designed to efficiently support CRUD (Create, Read, Update, Delete) operations on inventory data. The development process followed the waterfall model, which was chosen due to the project's well-defined requirements and minimal changes expected during the development.

Study [9] discusses the development of a backend system for the NTB Mall MSME registration platform and an attendance application. Using an agile approach, the project employed Next.js as the full-stack framework, integrating front-end interfaces with API routing. Furthermore, Prisma ORM facilitated efficient CRUD operations and ensured data integrity through structured schemas and controlled database migrations. In addition, Postman was used to validate API endpoints prior to deployment. Overall, the integration of these tools resulted in a scalable and maintainable system well-suited for managing MSME and attendance data effectively.

In Study [10], the development of a web-based booking system for Potret Kecilmu addressed challenges in manual booking management. The study proposed a more efficient booking system built using Next.js and Prisma ORM, adopting an agile development methodology. The findings indicate a 46.15% simplification of the booking process, measured by comparing the number of steps before and after system implementation, along with notable improvements in booking management efficiency, a reduction in operational errors, and an enhanced user experience.

Study [11] discusses the development of a web-based application to monitor and control Automated Guided Vehicles (AGVs) at PT. Stechoq Robotika Indonesia. The system was developed with a REST API back-end integrated with a ReactJS front-end to enable real-time communication between the web application and AGVs. The REST API facilitated efficient data exchange and control, making it a key component in the system's functionality. The application was developed using the Agile Scrum methodology, ensuring that the system met user needs through continuous improvements and rapid adaptability. By conducting regular sprints and integrating sprint reviews and retrospectives, the development team was able to refine API functionalities based on feedback from the frontend team and end-users. This iterative approach facilitated better synchronization between backend and frontend modules, ensuring that data exchange and features were reliable and responsive. Key features implemented included login, a dashboard, station management, AGV management, task management, and robot control, all of which successfully operated without issues as confirmed by iterative black-box testing.

Several studies have highlighted the significance of applying the Agile Scrum methodology, particularly in the development of web-based information systems. Research [6], [7], [11] show that Scrum's iterative approach facilitates rapid adaptation to user needs, improves team collaboration, and ensures the successful delivery of system features. Additionally, studies [8], [11] demonstrate the effectiveness of using REST APIs for seamless communication between the back-end and front-end, enabling real-time data exchange and control. Furthermore, research [9], [10] highlight the use of Next.js and Prisma ORM in system development, ensuring smooth integration and efficient database management through structured schemas and controlled migrations. While the "Ayo Pramuka" app has been developed, it primarily operates at the national level and do not address the operational needs at the city level. This system lacks support for city-specific administrative workflows, as described by the secretary of Kwartir Cabang Mataram during the interview. This study addresses that gap by focusing on a backend system specifically designed for the three-tier Scout Movement structure in Mataram.

This research focuses on the development of the Mataram Scout Information System, adopting proven technologies such as Next.js as a backend framework to efficiently structure API routes, REST API for standardized and scalable communication between system components, and Prisma ORM for efficient and structured interaction with the database. The system is designed to be scalable through modular REST API architecture, structured schema management via Prisma, and a clear separation of concerns, enabling potential future integration with mobile apps or external services. This approach ensures maintainability and adaptability to meet the evolving needs of the Scout organization in Mataram City.

III. Research Methodology

The research method acts as a roadmap for the entire information development process, starting from the planning phase to implementation. Scrum is an agile development method that organizes work into iterative cycles called sprints, allowing teams to deliver incremental improvements [5]. This method encourages collaboration among team members and stakeholders, with regular reviews to adapt to changing requirements throughout the development process. This research applies the Scrum method in the development of the REST API for the Mataram Scout Information System, implementing each phase systematically to ensure structured progress and continuous improvement.

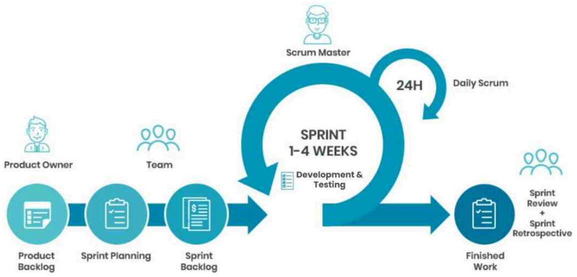


Fig. 1. Scrum Methodology

As shown in Fig. 1, the process begins with the product backlog, which contains a prioritized list of REST API features and system requirements such as user authentication endpoints, member data management endpoints, and activity reporting endpoints. These were gathered through interviews with Mataram City scouting representatives. The backlog remains dynamic, allowing updates based on evaluations at the end of each sprint. During the sprint planning phase, the team held collaborative discussions to select and break down backlog items into smaller, actionable tasks, each with an estimated duration based on team consensus. These selected items were then organized into a sprint backlog. Each sprint spans one to four weeks and involves development activities such as REST API endpoint creation, database integration, and testing. Progress during the sprint is tracked using project management tools to ensure transparency and accountability. A detailed overview of the REST API development durations can be found in the sprint backlog table. The system is developed as a full-stack application, with the back-end, which is the main focus of this paper, being developed using Next.js, and the front-end handled by another team. Both components are built and integrated together to ensure seamless communication between the REST API and the front-end interface. At the end of the sprint, a sprint review is conducted, to demonstrate the functional system increment to stakeholders. Their feedback on whether the developed features meet their needs is used to refine the product backlog and guide future development [5]. Finally, a sprint retrospective allows the team to reflect on their performance, identify challenges, and discuss improvements for future sprints. This continuous feedback loop ensures the team evolves along with the project's needs, leading to more effective and efficient development over time.

This research adopts black box testing to verify the functionality of the REST API without examining its internal code structure [12]. The testing process is carried out at the end of each sprint, following the completion of features listed in the sprint backlog. This ensures that each implemented endpoint functions correctly based on predefined system requirements [13]. Various inputs, including valid and invalid data, are used to evaluate how well the REST API handles different scenarios. For example, testing scenarios include creating new member records, retrieving member lists by unit level, and submitting activity reports. Valid inputs ensure all required fields are filled in correctly, while invalid inputs include missing parameters, incorrect data formats (e.g., invalid NTA or date), or attempts to access non-existent resources. These cases reflect real-world interactions, such as a scout leader submitting an activity report without a date or updating a member that does not exist. Postman is used to execute HTTP methods (GET, POST, etc.) and validate outputs such as response codes and messages. Test results are documented to identify deviations, which are then addressed in next sprints as part of the Scrum workflow.

IV. Result and Implementation

The implementation process began with initial system design, including the creation of the system architecture and Entity Relationship Diagram (ERD), based on requirements gathered from interviews with Mataram City scouting stakeholders. These designs provided a clear technical foundation for subsequent development.

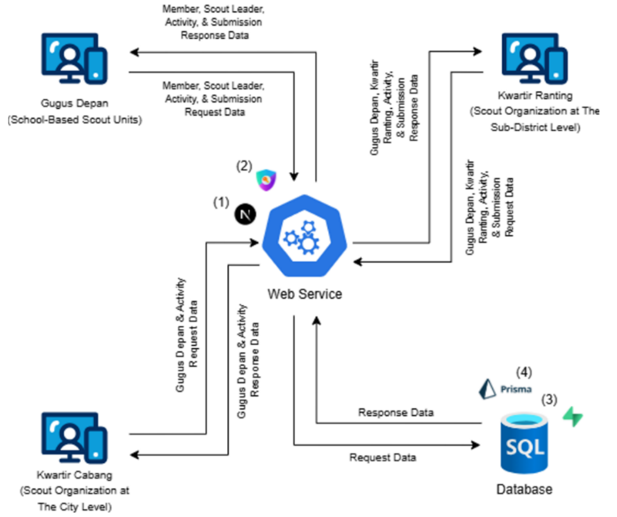

Member, Scout Leader, Activity, & Submission Response Data
Gugus Depan (School-Based Scout Units)
Member, Scout Leader, Activity, & Submission Request Data
Gugus Depan, Kwartir Ranting, Activity, & Submission Response Data
Kwartir Ranting (Scout Organization at The Sub-District Level)
Gugus Depan, Kwartir Ranting, Activity, & Submission Request Data
(2)
(1)
Web Service
Gugus Depan & Activity Request Data
Gugus Depan & Activity Response Data
(4)
Prisma
(3)
Response Data
Request Data
SQL
Database
Kwartir Cabang (Scout Organization at The City Level)

Fig. 2. System Architecture of Mataram Scout Information System

As shown in Fig. 2, the system adopts a web-based client-server architecture with a REST API approach. REST is an architectural style that uses standard HTTP methods, such as GET, POST, PUT, PATCH, and DELETE, to perform operations on resources [4]. These resources are represented by URLs and can be manipulated through interactions between the client and server. The system involves three main types of users: the gugus depan, the kwartir ranting, and the kwartir cabang. Each type of user interacts with the back-end services via a web interface to manage data such as members, scout leaders, activities, and activity submissions.

Requests from users are processed by the back-end service, which is developed using the Next.js framework (1) and functions as the REST API server. For security and access control, the system integrates NextAuth (2), a session-based authentication solution. Upon receiving a request and passing through authentication and authorization checks, the back-end service performs read/write operations on Supabase (3), which provides the PostgreSQL database. The interaction between the back-end and the database is facilitated by Prisma ORM (4), allowing for easy database queries and seamless integration with TypeScript.

To support the system's requirements, the database is designed with ten tables that store information from various entities, illustrated in Fig. 3. This design includes data for the gugus depan, kwartir ranting, kwartir cabang, member and scout leader management, rank history, activity reporting, as well as NTA submissions and verifications. The database is built using PostgreSQL, a powerful relational database management system, to ensure data integrity, scalability, and efficient querying. This design aims to ensure that the system development process is more efficient and structured.

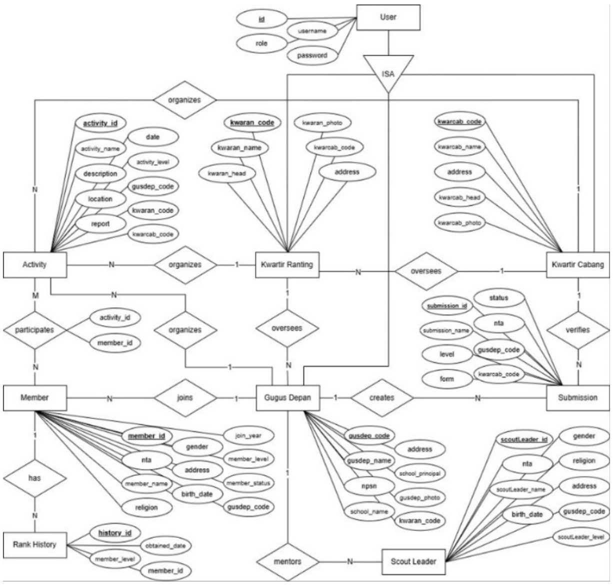


Fig. 3. ERD of Mataram Scout Information System

A. Product Backlog

In the Scrum process, the product backlog is an evolving and prioritized list of all system requirements and features. Maintained by the product owner, it provides a clear overview of the project scope and acts as the primary reference for planning upcoming sprints. To remain relevant throughout the development, the backlog is continuously updated to reflect shifting user needs and insights from previous sprints, helping ensure that each development cycle stays aligned with the project's goals.

TABLE I. PRODUCT BACKLOG

Domain	Feature Description	Priority
Registration	A feature that allows admins to create accounts within the system.	High
User Management	A feature that allows users to manage accounts of other users within their respective administrative regions.	High
Authentication	A feature that facilitates user authentication by verifying credentials for system access.	High
Member	A feature that allows users with the gugus depan role to manage member data.	High
Scout Leader	A feature that allows users with the gugus depan role to manage scout leader data.	Medium
Activity	A feature that allows users to manage activity data.	High
NTA Submission	A feature that allows users with the gugus depan role to submit NTA requests.	Low
Submission Verification	A feature that allows users with the kwartir cabang role to verify NTA submissions.	Low
Dashboard	A feature that allows users to view regional scouting organization development through visualized data charts.	High
Activity History	A feature that allows user to view a list of activities a member has participated in	High

Domain	Feature Description	Priority
Rank History	A feature that allows user to record and view a member's rank progression. Each entry includes the rank achieved and its date.	High

Following the product backlog, it is important to note that the addition of the "rank history" and "activity history" features resulted from feedback and requests from stakeholders during the sprint review session of sprint 3. During this review, stakeholders expressed the need for the system to display each member's rank progression and past activities to support more transparent and comprehensive data monitoring. This feedback was documented using the project's collaborative tool and discussed further in the subsequent sprint planning meeting. Based on the team's evaluation of priority and feasibility, these features were officially added to the sprint 4 backlog. This reflects the iterative nature of Scrum, where stakeholder input is continuously gathered and translated into actionable tasks to enhance system functionality. The product backlog in this research specifically focuses on back-end development tasks, as this study centers on the development of the REST API component of the system.

B. *Sprint Planning*

Sprint planning is a critical phase in which the developer reviews the product backlog to define the scope of work for the upcoming sprint. The most important features and requirements are selected based on the project's current priorities and then organized into a sprint backlog. Each selected item is broken down into smaller tasks with estimated durations. The sprint backlog serves as a roadmap, guiding the execution of tasks throughout the sprint to ensure steady progress toward meeting the system's development goals.

TABLE II. SPRINT BACKLOG

Domain	Description	Priority & Duration
Sprint 1		
Registration	The admin registers an account by filling out the required form fields.	High, 1 Day
Authentication	- Users with the gugus depan, kwartir ranting, and kwartir cabang roles receive a username and password, registered by either admin or the user in charge of their area. - Users log in and out using valid credentials.	High, 2 Days
User Management	- Admin can create and delete user accounts for the gugus depan, kwartir ranting, and kwartir cabang roles. - Kwartir cabang can create and delete user accounts for the kwartir ranting role. - Kwartir ranting can create and delete user accounts for the gugus depan role.	High, 3 Days
Sprint 2		
Domain	**Description**	**Priority**
Member	- Users with the gugus depan role can manage members (create, read, update, and delete) within their area. - Users can view members from each gugus depan.	High, 5 Days
Scout Leader	- Users with the gugus depan role can manage scout leaders (create, read, update, and delete) within their area. - Users can view scout leader data from each gugus depan.	Medium, 2 Days
Sprint 3		
Domain	**Description**	**Priority**
Activity	- Users can manage activities (create, read, update, and delete) they have participated in or organized. - Users can view activity reports from their respective region.	High, 5 Days
Sprint 4		
Domain	**Description**	**Priority**
Activity History	Users can view a list of activities a member has participated in.	High, 1 Day
Rank History	- Users with the gugus depan role can manage (add and delete) members rank history. The latest rank based on the most recent date is displayed on the member's data. - Users can view member's rank history.	High, 2 Days
Dashboard	Users can view data visualizations of their scout organization's progress through charts.	High, 4 Days
Sprint 5		
Domain	**Description**	**Priority**
NTA Submission	Users with the gugus depan role can add and delete NTA submissions.	Low, 2 Days
Submission Verification	Users with the kwartir cabang role can verify NTA submissions.	Low, 2 Days

C. *Sprint*

The sprint backlog outlines the development process, divided into seven sprints, with each sprint planned for completion within one to two weeks. Throughout these sprints, features are developed and assessed according to predefined functional standards, ensuring systematic progress and alignment with project goals. The following table provides a comprehensive overview of the API endpoints developed during the sprints, detailing their functionality.

TABLE III. ENDPOINTS

Code	Method	Endpoint	Description
Sprint 1			
E001	POST	/auth/register	Allow admin to register.
E002	POST	/user/admin	Allow admin to create user accounts for user with the gugus depan, kwartir ranting, and kwartir cabang role.
E003	GET	/user/admin	Allow admin to view the list of accounts they have created.
E004	DELETE	/user/admin/:id	Allow admin to delete user accounts.
E005	POST	/user/account	Allow user to create accounts for users within their respective area.
E006	GET	/user/ account	Allow user to view the list of accounts they have created.

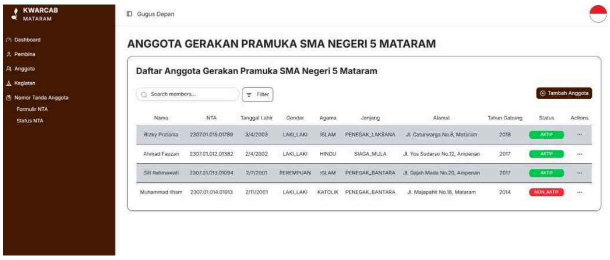
Code	Method	Endpoint	Description
E007	DELETE	/user/ account /:id	Allow user to delete user accounts.
Sprint 2			
E008	GET	/anggota	Allow viewing all members.
E009	POST	/anggota	Allow adding member.
E010	PATCH	/anggota/:id	Allow editing member.
E011	DELETE	/anggota/:id	Allow deleting member.
E012	GET	/pembina	Allow viewing all scout leaders.
E013	POST	/pembina	Allow adding scout leader.
E014	PATCH	/pembina/:id	Allow editing scout leader.
E015	DELETE	/pembina/:id	Allow deleting scout leader.
Sprint 3			
E016	GET	/kegiatan	Allow viewing all activities.
E017	GET	/kegiatan/:id	Allow viewing detailed activity.
E018	POST	/kegiatan	Allow adding activity.
E019	PATCH	/kegiatan/:id	Allow editing activity.
E020	DELETE	/kegiatan/:id	Allow deleting activity.
Sprint 4			
E021	GET	/anggota/:id /riwayat-kegiatan	Allow viewing a member's activity history.
E022	GET	/anggota/:id /riwayat-jenjang	Allow viewing a member's rank history.
E023	POST	/anggota/:id /riwayat-jenjang	Allow adding a new rank record to a member's history.
E024	DELETE	/anggota/:id /riwayat-jenjang /:riwayatId	Allow deleting rank record from a member's history.
E025	GET	/dashboard /anggotaByGender	Allows the chart to display member data based on gender.
E026	GET	/dashboard /anggotaByJenjang	Allows the chart to display member data based on rank.
E027	GET	/dashboard /anggotaByYear	Allows the chart to display total members per year.
E028	GET	/dashboard /gusdepByKwaran	Allows the chart to display total gugus depan in each kwartir ranting.
E029	GET	/dashboard /jenjangPerKwaran	Allows the chart to display total members per rank in each kwartir ranting.
E030	GET	/dashboard /kegiatanPerJenjang	Allows the chart to display total activities based on rank.
E031	GET	/dashboard /totalAnggota	Allows the chart to display total members.
E032	GET	/dashboard /totalKegiatan	Allows the chart to display total activities.
Sprint 5			
E033	GET	/ajuan	Allow viewing all submissions.
E034	POST	/ajuan	Allow adding submission.
E035	PATCH	/ajuan/:id	Allow editing submission.
E036	DELETE	/ajuan/:id	Allow deleting submission.

Sprint 1 of Development. The development activity focuses on the user registration and authentication domain. Endpoints under /auth/register, /user/admin, and /user/account were created to facilitate account creation and management. Admins can register themselves by providing required information such as username and password. Additionally, both admins and users responsible for specific areas can create user accounts for their respective regions by submitting data such as username, password, name, code, and area code.

Authentication is handled using NextAuth, which provides session-based authentication for the system. Upon successful login, a session is established to store user-specific data, such as role and area-specific codes (kwartir cabang, kwartir ranting, gugus depan). This stored session data enables access verification and enforces role-based authorization, ensuring secure and controlled user interactions across the platform.

Sprint 2 of Development. The development activity focuses on the management of members and scout leaders. Endpoints under /anggota and /pembina were created to enable viewing, creation, editing, and deletion of data. These endpoints allow users to maintain accurate and up-to-date records of individuals involved in the organization. CRUD (Create, Read, Update, Delete) operations on these entities ensure that the system can adapt to changes in personnel data efficiently.

Only users with the gugus depan role are authorized to add, edit, or delete member and scout leader data, ensuring that data management remains within the appropriate organizational level. All data at the gugus depan level can later be monitored and accessed by kwartir ranting and kwartir cabang for supervision and reporting purposes.



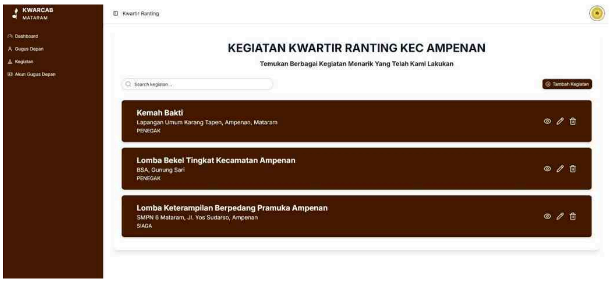
ANGGOTA GERAKAN PRAMUKA SMA NEGERI 5 MATARAM

Daftar Anggota Gerakan Pramuka SMA Negeri 5 Mataram

Nama	NTA	Tanggal Lahir	Gender	Agama	Jenjang	Alamat	Tahun Gabung	Status	Actions
Rizky Pratama	2307.01.015.01789	3/4/2003	LAKI_LAKI	ISLAM	PENEGAK_LAKSANA	Jl. Caturwarga No.8, Mataram	2018	AKTIF	···
Ahmad Fauzan	2307.01.012.01382	2/4/2002	LAKI_LAKI	HINDU	SIAGA_MULA	Jl. Yos Sudarso No.12, Ampenan	2017	AKTIF	···
Siti Rahmawati	2307.01.013.01094	2/7/2001	PEREMPUAN	ISLAM	PENEGAK_BANTARA	Jl. Gajah Mada No.20, Ampenan	2017	AKTIF	···
Muhammad Ilham	2307.01.014.01913	2/11/2001	LAKI_LAKI	KATOLIK	PENEGAK_BANTARA	Jl. Majapahit No.18, Mataram	2014	NON_AKTIF	···

Fig. 4. Member Page Integration Result

Sprint 3 of Development. The development activity focuses on the activity management module. Endpoints created under /kegiatan allow users to perform operations such as adding new activities, viewing a list of all activities, viewing detailed information about a specific activity using its ID, editing existing activity records, and deleting them when necessary. When adding an activity, users are required to upload an activity report document, which can later be downloaded to support documentation and evaluation purposes. The system also requires users to select at least one participant involved in the activity. These functionalities provide flexibility in organizing and maintaining activity data, which is essential for tracking scout events and engagements.



KEGIATAN KWARTIR RANTING KEC AMPENAN

Temukan Berbagai Kegiatan Menarik Yang Telah Kami Lakukan

Kemah Bakti
Lapangan Umum Karang Tapen, Ampenan, Mataram
PENEGAK

Lomba Bekel Tingkat Kecamatan Ampenan
BSA, Gunung Sari
PENEGAK

Lomba Keterampilan Berpedang Pramuka Ampenan
SMPN 6 Mataram, Jl. Yos Sudarso, Ampenan
SIAGA

Fig. 5. Activity Page Integration Result

Sprint 4 of Development. The development activity focuses on both analytical features and dynamic history tracking, following updated requests from stakeholders during development. Midway through development, stakeholders and the product owner requested the addition of rank history and activity history features to better reflect member progression and involvement. Endpoints under /anggota/:id/riwayat-jenjang and /anggota/:id/riwayat-kegiatan were developed to allow viewing, adding, and deleting history records. The activity history is derived from the participant relationship table, where history records appear only if the member has participated in an activity. This ensures that the system accurately reflects members actual involvement without manual input duplication.

In addition, a set of endpoints under /dashboard was implemented to support statistical visualization of data, such as total members by gender, rank, year, and distribution of units in each region. These endpoints feed data into frontend charts and visual components, helping users monitor organizational metrics at a glance.

Sprint 5 of Development. The development activity focuses on submission management through the /ajuan endpoint group. Users can submit new data entries, review submission history, edit pending submissions, or delete them if necessary. Only users with the gugus depan role are authorized to submit new data, while users with the kwartir cabang role have the authority to verify or edit pending submissions. When a kwartir cabang user edits a submission status to approved and has filled in the NTA field, the original form file uploaded by the gugus depan user will be automatically deleted from the database. This mechanism is in place to minimize storage usage and reduce the risk of retaining redundant or potentially sensitive data after verification is complete. This role-based access ensures that data submission follows the proper hierarchy, supporting a controlled and verifiable approval process within the organization.

Daily Scrum. Daily scrum meetings were held to coordinate progress on the REST API development, address technical issues, and ensure tasks stayed on track. These meetings typically lasted around 15 minutes, helping maintain effective communication and ensuring backend features were delivered on schedule and aligned with user needs.

Sprint Review. The development progressed steadily through five sprints, each focusing on essential features. Sprint 1 established user registration and authentication with role-based access. Sprint 2 developed member and scout leader management, allowing CRUD operations with role-specific permissions. Sprint 3 developed the activity management module, allowing CRUD operations on activities, with data accessible within each user's specific area. A challenge arose in handling file uploads for activity report additions and edits, which was resolved by exploring and reading the documentation for the storage service. Furthermore, during frontend integration, there were issues regarding data access for specific roles. This was addressed by configuring role-based access and storing necessary properties in the NextAuth session. Sprint 4 added rank and activity history tracking, responding to stakeholder needs, along with analytical endpoints for data visualization. Sprint 5 completed the submission management feature, enabling users to submit, edit, or delete data entries based on their roles. Product owner and stakeholders feedback shaped each sprint, ensuring the system met user needs and expectations.

Sprint Retrospective. In the sprint evaluation, a meeting with the product owner was held to reflect on the team's performance, highlight successes, and discuss areas for improvement. Throughout the development of the Mataram Scout Information System REST API, the team encountered various technical and operational challenges. On the technical side, configuring secure session-based authentication using NextAuth to support role-based access required careful handling of token sessions, dynamic role verification, and integration with the frontend to ensure protected routes responded correctly to user credentials. Additionally, managing file uploads for activity reports involved dealing with asynchronous file handling and ensuring proper cleanup mechanisms. This was particularly important when files had to be deleted after verification to reduce storage usage and preserve data security. Operationally, the team had to continuously respond to evolving stakeholder feedback, which required flexible sprint planning and prioritization adjustments. In sprint 3, stakeholders proposed the addition of rank history and activity history features to improve transparency in member tracking. This led to changes in the database schema, the addition of new relationship tables, and the development of endpoints that could dynamically retrieve and update historical data based on user participation records, ensuring that these features reflected actual system activity without manual duplication. These ongoing challenges were regularly addressed through daily scrum meetings and close collaboration with the product owner and stakeholders. Despite these obstacles, all backlog items were completed and successfully integrated into the frontend, meeting the product owner's expectations.

D. REST API Testing

In each sprint, REST API testing is conducted once the sprint's endpoints are fully developed. The sprint continues only if the tests meet the requirements. The following are the test results for the REST API endpoints developed to support the Mataram Scout Information System. Each test scenario in TABLE IV corresponds to an endpoint identified by its code in TABLE III.

TABLE IV. BLACK-BOX TESTING RESULT

Code	Scenario	Result	Status
E001	Valid request	201 Created, message “User created successfully”	Pass
	Missing required fields	400 Bad Request, message "All fields are required"	
	Register with wrong username format	400 Bad Request, message "Username cannot contain spaces"	
	Register with wrong password format	400 Bad Request, message "Password must be at least 8 characters long, contain uppercase, lowercase letters, and numbers."	
	Register with duplicate username	409 Conflict, message "Username already taken"	
E002	Valid request	201 Created, message “User created successfully”	Pass
	Missing required fields	400 Bad Request, message "All fields are required"	
	Create user account with wrong username format	400 Bad Request, message "Username cannot contain spaces"	
	Create user account with wrong code format	400 Bad Request, message "Code cannot contain spaces"	
	Create user account with wrong password format	400 Bad Request, message "Password must be at least 8 characters long, contain uppercase, lowercase letters, and numbers."	
	Create user account with duplicate username	409 Conflict, message "Username already taken"	
	Create user account with duplicate code	409 Conflict, message "Code already taken"	
	Unauthorized user tries to create account	403 Forbidden, message "Unauthorized: Only 'Superadmin' users can create account"	
E003	Retrieve accounts	200 OK, list of created accounts returned	Pass
E004	Delete existing account	200 OK, message "User deleted successfully"	Pass
E005	Valid request	201 Created, message “User created successfully”	Pass
	Missing required fields	400 Bad Request, message "All fields are required"	
	Create user account with wrong username format	400 Bad Request, message "Username cannot contain spaces"	
	Create user account with wrong code format	400 Bad Request, message "Code cannot contain spaces"	
	Create user account with wrong password format	400 Bad Request, message "Password must be at least 8 characters long, contain uppercase, lowercase letters, and numbers."	
	Create user account with duplicate username	409 Conflict, message "Username already taken"	
	Create user account with duplicate code/name	409 Conflict, message "Code or name already taken"	
	Unauthorized user tries to create account	403 Forbidden, message "Unauthorized: Only 'Kwarcab/Kwaran /Gusdep' users can create account"	
E006	Retrieve accounts	200 OK, list of created accounts returned	Pass
E007	Delete existing account	200 OK, message "User deleted successfully"	Pass
E008	Retrieve members	200 OK, list of members returned	Pass
E009	Valid request	201 Created, message “Member added successfully”	Pass
	Missing required fields	400 Bad Request, message "All fields are required"	
	Add member with duplicate NTA	409 Conflict, message "NTA already registered"	
E010	Valid request	201 Created, message “Member updated successfully”	Pass
	Edit member with duplicate NTA	409 Conflict, message "NTA already registered"	
E011	Delete existing member	200 OK, message "Member deleted successfully"	Pass
E012	Retrieve scout leaders	200 OK, list of scout leaders returned	Pass
E013	Valid request	201 Created, message “Scout leader added successfully”	Pass
	Missing required fields	400 Bad Request, message "All fields are required"	
	Add scout leader with duplicate NTA	409 Conflict, message "NTA already registered"	
E014	Valid request	201 Created, message “Scout leader updated successfully”	Pass

	Edit member with duplicate NTA	409 Conflict, message "NTA already registered"	
E015	Delete existing scout leader	200 OK, message "Scout leader deleted successfully"	Pass
E016	Retrieve activities	200 OK, list of activities returned	Pass
E017	Retrieve detail activity	200 OK, list of detail activity returned	Pass
E018	Valid request	201 Created, message “Activity added successfully”	Pass
	Missing required fields	400 Bad Request, message "All fields are required"	
	Add activity with too long description.	400 Bad Request, message "The description is too long (max 300 words)"	
	Attempt to upload a non-PDF activity report file.	400 Bad Request, message "Only PDF files are allowed"	
	Attempt to upload a file larger than 2MB.	400 Bad Request, message "File size must be less than 2MB"	
	Add activty without selecting participant.	400 Bad Request, message "At least one participant must be selected"	
E019	Valid request	201 Created, message “Activity updated successfully”	Pass
	Edit activity with too long description.	400 Bad Request, message "The description is too long (max 300 words)"	
	Attempt to upload a non-PDF activity report file.	400 Bad Request, message "Only PDF files are allowed"	
	Attempt to upload a file larger than 2MB.	400 Bad Request, message "File size must be less than 2MB"	
	Edit activty without selecting participant.	400 Bad Request, message "At least one participant must be selected"	
E020	Delete existing activity	200 OK, message "Activity deleted successfully"	Pass
E021	Retrieve member’s activity history	200 OK, member’s activity history returned	Pass
E022	Retrieve member’s rank history	200 OK, member’s rank history returned	Pass
E023	Valid request	201 Created, message “Rank history added successfully”	Pass
	Missing required fields	400 Bad Request, message "All fields are required"	
E024	Delete existing rank history	200 OK, message "Rank history deleted successfully"	Pass
E025-E032	Retrieve dashboard data based on parameters	200 OK, dashboard data returned	Pass
E033	Retrieve submissions	200 OK, list of submissions returned	Pass
E034	Valid request	201 Created, message “Submission added successfully”	Pass
	Missing required fields	400 Bad Request, message "All fields are required"	
	Attempt to upload a non-PDF submission file.	400 Bad Request, message "Only PDF files are allowed"	
	Attempt to upload a file larger than 2MB.	400 Bad Request, message "File size must be less than 2MB"	
E035	Valid request	201 Created, message “Submission updated successfully”	Pass
	Attempt to update submission without providing the "NTA" field when the status is set to "DITERIMA"	400 Bad Request, message "Field 'NTA' is required if the status is 'DITERIMA'"	
E036	Delete existing submission	200 OK, message “Submission deleted successfully"	Pass

As shown in TABLE IV, the tests were conducted to ensure each REST API endpoint correctly handles the expected inputs and returns appropriate responses, confirming the system functions as expected for both valid and invalid data scenarios. A User Acceptance Testing (UAT) session was held through a meeting with representatives of the Mataram Scout Movement to demonstrate the Scout Information System and gather feedback via a survey. The survey involved 23 respondents from various organizational levels (gugus depan, kwartir ranting, kwartir cabang). The system received an average feasibility rating of 81.74%, classified as "Good," indicating that the respondents perceive the system as efficient and suitable for improving data management compared to the previous manual workflows using Google Forms and Excel. A more detailed explanation of the UAT process is provided in a separate study authored by another member of the development team, who was responsible for the frontend implementation. This positive evaluation highlights the system’s potential to address inefficiencies in data management within Kwartir Cabang Mataram.

V. Conclusion and Suggestion

This research developed the Mataram Scout Information System using the Scrum method, focusing on backend development of a REST API to support core system functionalities. The system was designed to streamline data management and reporting processes within the Mataram City scouting organization. The REST architectural style simplified communication between components, with standardized HTTP methods enabling smooth integration between frontend and backend. Key endpoints such as /anggota, /pembina, /kegiatan, /ajuan, and /dashboard were integrated modularly without major structural changes, enhancing scalability and maintainability. The development followed all Scrum stages. The initial product backlog was based on system requirements and continuously updated based on stakeholder feedback, reflecting the project's adaptability to evolving user needs. Sprint planning was held to prioritize tasks and estimate workloads. Each sprint, with a duration of 1-2 weeks, targeted specific backend functionalities. Daily scrum enabling the team to monitor progress and resolve issues quickly. Sprint reviews served as formal checkpoints for evaluating progress and gathering feedback. Sprint retrospectives allowed the team to reflect on performance and identify areas for improvement. All planned sprint backlog items were completed and integrated into the system, and stakeholder-requested features were effectively translated into actionable tasks and delivered in subsequent sprints.

Several technical and operational challenges arose during development. One significant challenge was implementing session-based authentication and role-based access control (RBAC) using NextAuth. This required careful session handling and secure authorization logic to differentiate access for various user roles (gugus depan, kwartir ranting, and kwartir cabang), thereby ensuring a secure and organized system structure. Another major challenge involved mid-development integration of rank and activity history features, which were not part of the initial requirements but were added in response to stakeholder input during sprint reviews. Addressing this required modifying the database schema and developing dynamic endpoints to manage member progression data. These changes also demanded flexibility in sprint planning and close collaboration with the product owner to ensure alignment with evolving stakeholder needs.

The system's reliability was verified through black-box testing, which confirmed that REST API endpoints functioned correctly across a range of valid and invalid input scenarios. Additionally, a User Acceptance Testing (UAT) session involving 23 respondents from different organizational levels yielded a feasibility rating of 81.74%, classified as "Good." This result suggests that the system met both technical expectations and real-world operational needs, particularly in facilitating efficient data management and real-time activity monitoring. For future research, it is recommended to enhance the user experience by improving mobile access to the system, making it more flexible and adaptable to future needs in scouting data management. Mobile access can be improved by developing features such as offline data entry for areas with limited connectivity, push notifications for activity updates, and mobile-friendly user interfaces to support field usability by scout leaders.

References

[1] Renilda Manis Da Gomez, Elviana Pona Rato, Donatus Datoq, Marwan Marwan, Wihelmus Sede, and Fitrah Hamzah, "Membangun Karakter Bangsa Melalui Kepramukaan," *Jurnal Ilmiah Dan Karya Mahasiswa*, vol. 3, no. 1, pp. 182–187, Feb. 2025, doi: 10.54066/jikma.v3i1.3015.

[2] L. Lutfiasin, "Sejarah Pembentukan Gerakan Pramuka Dan Pengaruhnya Dalam Dunia Pendidikan Islam," *Thawalib | Jurnal Kependidikan Islam*, vol. 2, no. 1, pp. 37–52, Apr. 2021, doi: 10.54150/thawalib.v2i1.19.

[3] Undang-Undang Republik Indonesia Nomor 12 Tahun 2010 Tentang Gerakan Pramuka. [Online]. Available: https://www.pramuka.or.id/files/document/UU-Gerakan-Pramuka.pdf

[4] S. U. Meshram, "Evolution of Modern Web Services: REST API with its Architecture and Design," *International Journal of Research in Engineering, Science and Management*, vol. 4, no. 7, pp. 83–86, 2021.

[5] A. C. Sassa, I. Alves De Almeida, T. Nakagomi, F. Pereira, and M. Silva De Oliveira, "Scrum: A Systematic Literature Review," *(IJACSA) International Journal of Advanced Computer Science and Applications*, vol. 14, no. 4, pp. 173–181, 2023, doi: 10.14569/IJACSA.2023.0140420.

[6] R. Noveandini, M. S. Wulandari, and A. Hakim, "Penerapan Metode Scrum pada Rancang Bangun Sistem Informasi Penjualan Toko Sepatu Rabbani Shoes," *Seminar Nasional Teknologi Informasi dan Komunikasi STI&K (SeNTIK)*, vol. 7, no. 1, pp. 192–199, Jul. 2023.

[7] T. Wibowo and S. Salim, "Pengembangan Dan Implementasi Back End Website Sistem Laporan Keuangan Di Smk Multistudi High School Menggunakan Kerangka Kerja Agile Scrum," *NaCosPro: National Conference For Community Service Project*, vol. 4, no. 1, pp. 880–887, Aug. 2022, [Online]. Available: http://journal.uib.ac.id/index.php/nacospro

[8] S. B. Farchani, A. Kusuma, and N. Hermanto, "Implementasi REST API Dalam Pengembangan Back-End Inventory Peminjaman," *JIPI (Jurnal Ilmiah Penelitian dan Pembelajaran Informatika)*, vol. 10, no. 2, pp. 1404–1413, Jun. 2025, doi: 10.29100/jipi.v10i2.6249.

[9] R. Muhammad, G. Dennaya, and A. Zubaidi, "Pengembangan Back-End Pada Aplikasi Presensi dan Website Pendataan UMKM NTB Mall," *JBegaTI*, vol. 5, no. 2, pp. 170–181, Sep. 2024, [Online]. Available: http://begawe.unram.ac.id/index.php/JBTI/

[10] W. Ahmad Widodo, I. Ratna Indra Astutik, and N. Ariyanti, "Penerapan Framework NextJS Dan Prisma ORM Untuk Perluasan Marketplace Newborn Photography," *JOISIE (Journal Of Information Systems And Informatics Engineering)*, vol. 8, no. 2, Dec. 2024, doi: 10.35145/joisie.v8i2.4704.

[11] S. L. Mufreni *et al.*, "Sistemasi: Jurnal Sistem Informasi Development of Web-App Using Agile Scrum Method at PT. Stechoq Robotika Indonesia," *Sistemasi: Jurnal Sistem Informasi*, vol. 14, no. 2, pp. 917–931, 2025, [Online]. Available: http://sistemasi.ftik.unisi.ac.id

[12] I. R. D. Muhammad and I. V. Paputungan, "Development of Backend Server Based on REST API Architecture in E-Wallet Transfer System," *Jurnal Sains, Nalar, dan Aplikasi Teknologi Informasi*, vol. 3, no. 2, pp. 79–87, Jan. 2024, doi: 10.20885/snati.v3.i2.35.

[13] A. C. Praniffa, A. Syahri, F. Sandes, U. Fariha, and Q. A. Giansyah, "Pengujian Black Box dan White Box Sistem Informasi Parkir Berbasis Web," *Jurnal Testing dan Implementasi Sistem Informasi*, vol. 1, no. 1, pp. 1–16, 2023.