MINISTRY OF

EDUCATION AND TRAINING

FPT UNIVERSITY

Capstone Project Document

Real-time Monitoring System of electrical parameters for the FSC-MDF Wooden board Factory of Kim Tin Group Corporation

SP24SE062_GSP04	
Group Members	Bùi Ngọc Huy - SE150819 Nguyễn Nhật Huy - SE150807 Lê Xuân Đại - SE151340 Trần Trung Kiên - SE150683 Lê Tiến Thịnh - SE151082
Supervisor	Mrs. Nguyễn Thị Cẩm Hương
Product Owner	Mr. Đỗ Tân Khoa
Ext Supervisor	N/A
Capstone Project code	GSP24SE04

- Ho Chi Minh, 04/2024 –

Acknowledgment

We express our heartfelt gratitude to our esteemed supervisor, Mrs. Nguyễn Thị Cẩm Hương, for her invaluable guidance and meticulous review of our RMS project. Her unwavering support has empowered our team to integrate theory into practical application seamlessly.

We sincerely appreciate our diligent reviewers for their insightful advice and constructive feedback. Their probing questions and suggestions have been instrumental in enhancing the quality and relevance of our project.

We also extend our heartfelt thanks to Mr. Bùi Kim Thành and Mr. Đỗ Tân Khoa, our esteemed partners from the factory. Their collaboration and input as representatives of our valued customers have been instrumental in shaping the direction and success of our project.

Furthermore, we want to thank each member of our dedicated team for their tireless efforts and unwavering commitment. We have reached this point through our collective dedication and hard work.

Finally, we thank our dedicated lecturers, supportive friends, and loving family members for their unwavering encouragement and steadfast support throughout our academic journey. Their belief in us has been a constant source of inspiration and motivation.

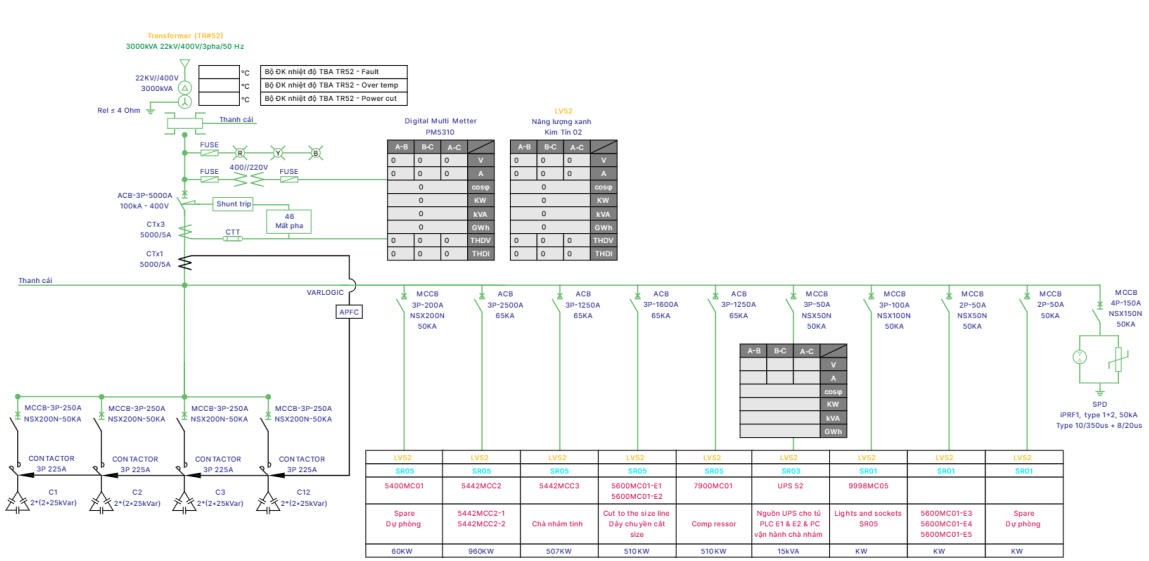
With the support of our supervisor, reviewers, partners, and loved ones, we have navigated challenges and achieved milestones in our university life.

Table of Contents

List of Tables

List of Figures

Glossary/Abbreviation

Term	Description
Administrator	Read details in the Administrator section.
BR	Business Rule.
CR	Common Requirement.
Electrical schematic diagram	An electrical schematic diagram is a graphical representation of electrical circuits, illustrating the arrangement of components, connections, and electrical parameters within a factory station. In the context of Kim Tin's factory and the RMS application, these diagrams depict the layout of electronic components, circuits, and monitoring figures such as voltage, current, power, etc. They serve as visual aids for understanding the electrical infrastructure and facilitating real-time monitoring features within the RMS application.  *Figure 1 Example of an electrical schematic diagram*

FSC-MDF	FSC MDF refers to a **M**edium-**D**ensity **F**iberboard certified by the **F**orest **S**tewardship **C**ouncil (FSC). The FSC is an international organization that promotes responsible stewardship of forests worldwide to minimize the negative impact of commercial timber sourcing. MDF is an engineered wood product that binds strands, particles, fibers, veneers, wood boards, and adhesives. It's commonly used to manufacture furniture, kitchen cabinets, flooring, picture frames, and wooden children's toys, among other products.
PLC	- A Programmable Logic Controller, or PLC, is a ruggedized computer used for industrial automation. These controllers can automate a specific process, machine function, or production line. - The Kim Tin's factory is divided into multiple clusters ("cụm"); each cluster consists of various stations ("trạm"), including tens of mechanical machines directly involved in the production process. On top of all clusters, a PLC device gathers machines' input and exposes associated output based on its initial programmed logic made by the factory's employees. - The RMS will mainly connect to this PLC device to get the input for the functionalities.  *Figure 2 A PLC Siemens S7 1200 - Main data point of the RMS project*
PLC Data block	A data block DB is a memory area used to save the values of the parameters written during the execution of the PLC program.

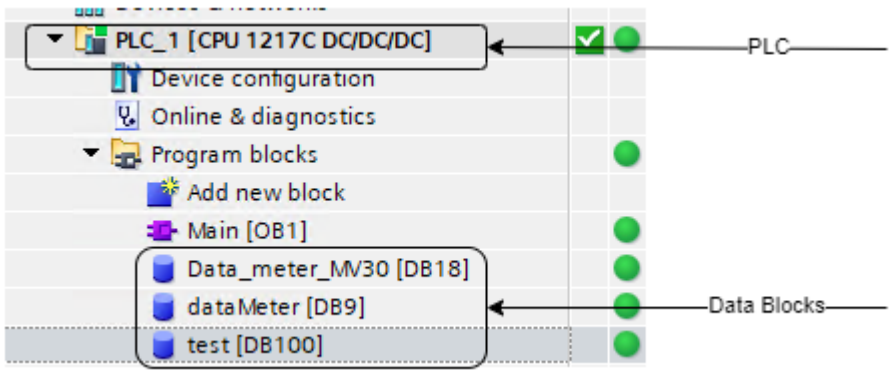


Figure 3 PLC and Data blocks configuring from a view of TIA Portal 16

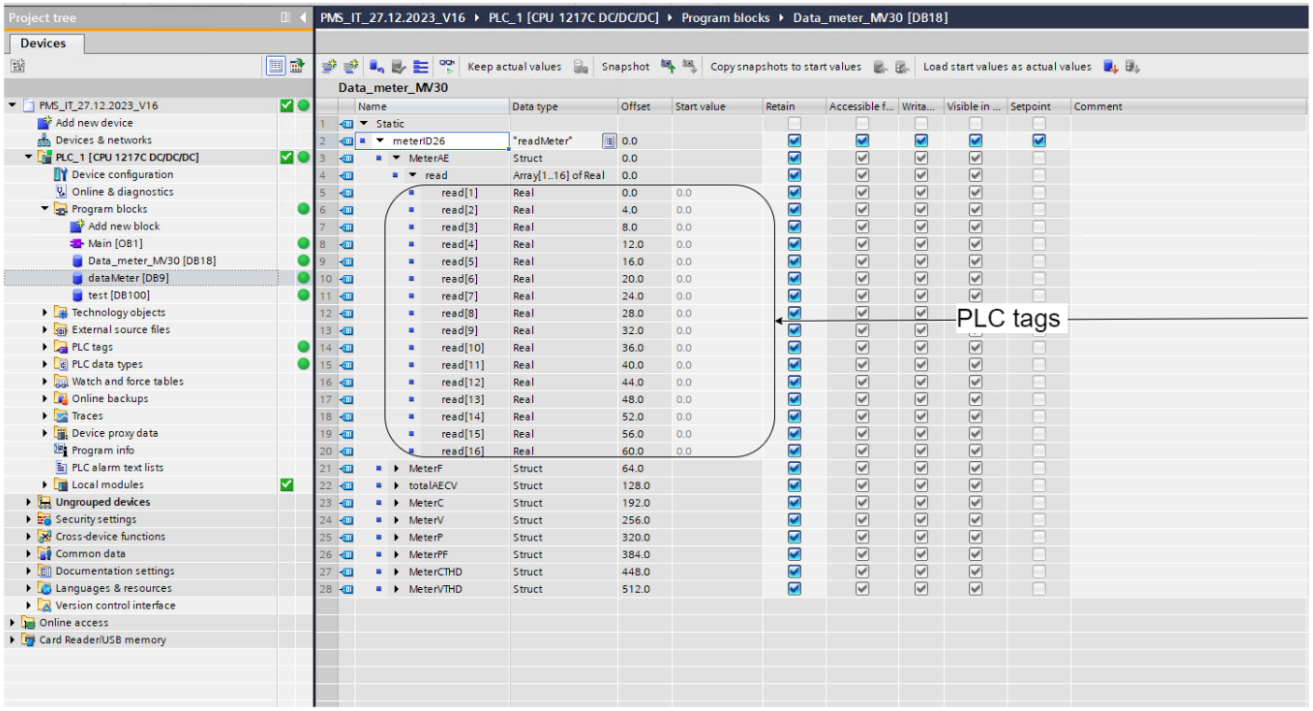
PLC Tag	A Tag is a memory block with a binary value in a data block. Today, most PLC variables are ‘tag-based,’ meaning each variable is given a data type that defines how the variable will be represented.  *Figure 4 List of PLC Tags in a data block of a PLC, viewing on TIA Portal 16* For an external service to accurately read the current value from a PLC Tag, the following fields are required: - Data Block: Indicates the number of the Data Block to which the Tag belongs. - Offset: Specifies the Tag's address within the series of bytes in the Data Block. - Data Type: Defines the data type of the value associated with the Tag.
Station	A station is a unit used to group machines in the entire chain of Kim Tin’s factory; each station is represented using an electrical schematic diagram on 1 tab on the Monitoring screen.
Supervisor	Read detail at the Supervisor section.

Table 1 Glossaries and Abbreviations

I Project Introduction

1. Overview

1.1. Project Information

- Project name: Real-time Monitoring System of electrical parameters for the FSC-MDF Wooden board Factory of Kim Tin Group Corporation.
- Project code: GSP24SE04.
- Group code: SP24SE062.
- Group name: HBC.
- Software type: Web Application.

1.2. Project Team

Full name	Role	Email	Mobile
Mrs. Nguyễn Thị Cẩm Hương	Lecturer	huongntc2@fe.edu.vn	N/A
Mr. Bùi Kim Thành	Product champion	N/A	N/A
Mr. Đỗ Tân Khoa	Product owner	N/A	0988334405
Bùi Ngọc Huy	Leader	huybnse150819@fpt.edu.vn	0937046839
Nguyễn Nhật Huy	Member	huynnse150807@fpt.edu.vn	0783329134
Trần Trung Kiên	Member	kienttse151340@fpt.edu.vn	0783329134
Lê Xuân Đại	Member	dailxse150683@fpt.edu.vn	0352654331
Lê Tiến Thịnh	Member	thinhltse151082@fpt.edu.vn	0936572310

Table 2 Stakeholders in the Project

2. Product Background

2.1. Kim Tin's Factory Overview

Kim Tin Group Corporation has operated an MDF wooden board factory since 2018 in Nam Dong Phu Industrial Park, Dong Phu district, Binh Phuoc Province. The facility is structured into clusters featuring mechanical machines and PLC devices crucial for manufacturing.

Each cluster ("cụm") comprises multiple stations ("trạm"), each equipped with several mechanical machines that play a vital role in manufacturing. A PLC device oversees on top of all clusters, collecting machine data and executing outputs based on their programmed logic. The

manufacturing process is divided into multiple stages, each utilizing different machines and varying energy consumption levels to produce high-quality wooden boards.

2.2. Kim Tin’s Factory Desktop Application

The factory's employees have been utilizing a desktop application for real-time monitoring of energy consumption and production parameters. A glimpse into the application's interface is provided in Figure 5.

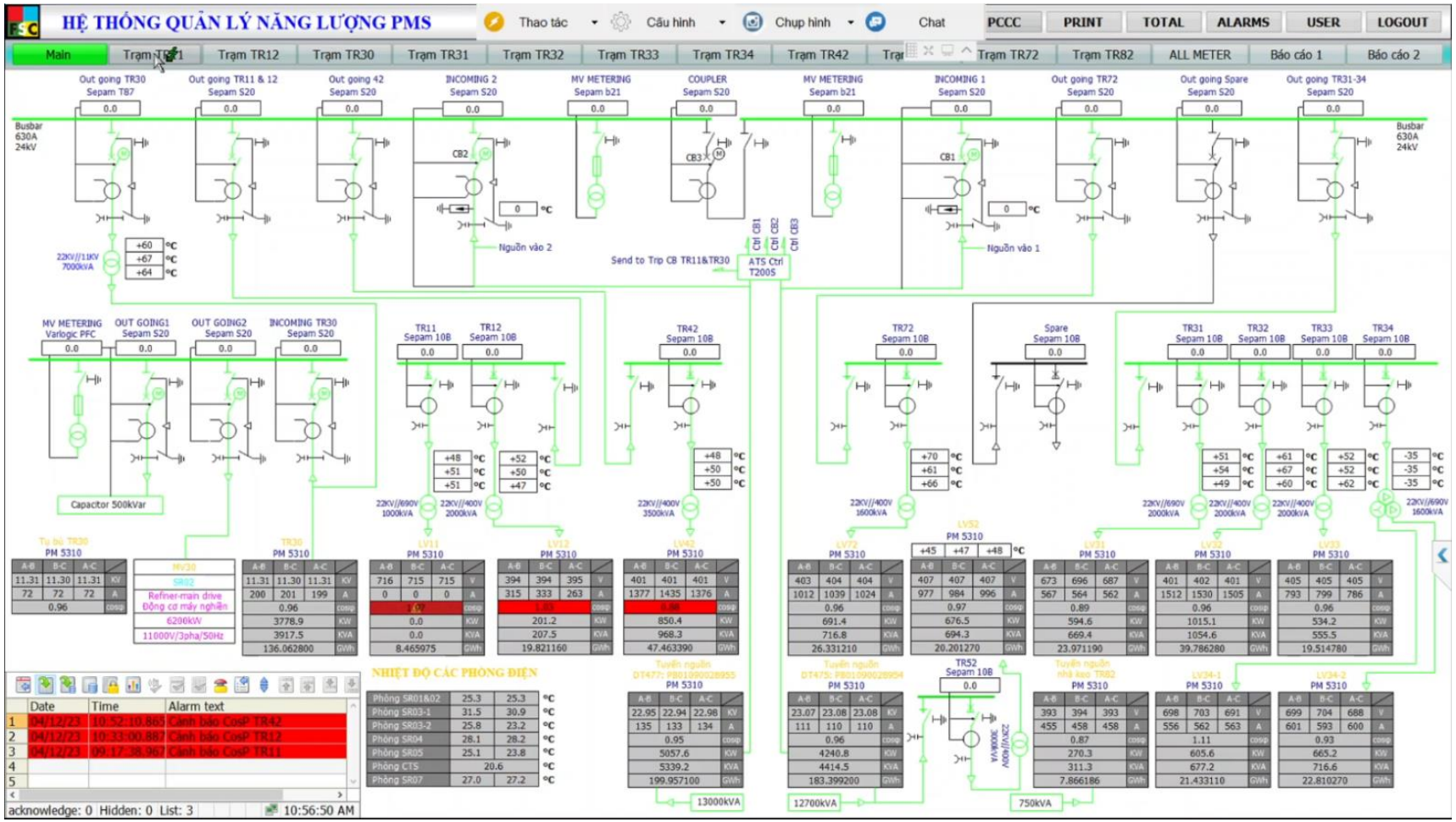


Figure 5 Old RMS Monitoring user interface.

Despite its functionality, the RMS desktop application has encountered several issues that hinder operational efficiency:

- **Poor Accessibility:** The desktop application requires installation and pre-configuration, limiting its accessibility.
- **Inadequate Warning Systems:** The app's alarm system is inefficient due to its limited display and visibility. It also lacks external notifications like email or SMS, reducing effectiveness in urgent situations.
- **Deficient Reporting Features:**
    - **Lack of Dynamic Data Interaction:** The application lacks dynamic abilities for data manipulation, such as the option to sort, search, filter, or show/hide columns, which would streamline reporting tasks and enhance user interaction with the data.
    - **Absence of Visual Aids:** Without charts or other visual aids, data comprehension is slow and challenging.
- **Extensibility Issues:** The application's architecture, constrained by initial development time limitations and technology choices, is poorly designed, making it challenging to implement new features or address existing problems.

Recognizing these issues, stakeholders at the factory have expressed the need to transition from a desktop-based application to a web-based platform. The move to a web-based Real-time Monitoring System is driven by the desire to improve accessibility, enhance feature sets, and provide a more user-friendly and efficient tool for managing the factory's operations.

3. Business Opportunity

Developing a web-based Real-time Monitoring System presents a substantial business opportunity for the MDF wooden board factory. By replacing the current desktop application with a contemporary, user-friendly, and adaptable web-based platform, users will experience increased efficiency, better control over power management, and cost reduction.

Furthermore, a web-based system allows for seamless integration with the hardware and platforms, opening possibilities for additional services and revenue sources.

4. Software Product Vision

For factory operators who demand seamless real-time data insights, the Web-Based Real-time Monitoring System is a comprehensive monitoring solution that transcends traditional desktop applications. It is designed to deliver advanced remote monitoring, dynamic data reports, and intuitive user interaction. This system empowers operators to enhance operational efficiency, minimize energy costs, and foster sustainable industrial practices. Unlike the limited and rigid desktop-based RMS, our product offers a scalable, user-centric platform with robust features that adapt to the evolving needs of modern factories. Our solution demonstrates our dedication to

innovation, operational improvement, and meeting the diverse requirements of stakeholders in the manufacturing sector.

5. Project Scope and Limitation

5.1. Major Features

The capstone project will focus on the following major features of the new web-based Real-time Monitoring System:

- **Monitoring:**
  - Ensuring a stable connection to the required PLC, handling a data stream of 500 - 600 parameters per second, and accurately displaying this information on the user interface.
  - Each factory station screen must display the associated electrical schematic diagram and its parameters correctly.
- **Alarm:**
  - Allow users to create alarms to alert users of any unexpected events in the factory.
  - Must support two types of alarms, one to check existing conditions in the PLC before and one to check custom conditions created by the user without the need to add/modify the PLC logic.
- **Reporting:** Developing a comprehensive reporting feature with table-format data and charts for efficient decision-making.
- **User Management:** Creating a set of user management functions that enable different levels of access and control within the application.

5.2. Scope

5.2.1. In Scope

- Creating a web-based Real-time Monitoring System to replace the old desktop-based RMS.
- Meet all of these acceptance criteria:
  - AC1 - Features: a web-based Real-time Monitoring System with all the features mentioned in the Major Features section.
  - AC2 – Real-time: Effectively real-time monitoring with acceptable cycle time.
  - AC3 – Enhanced the overall UI/UX.
  - AC4 – Ensure integrity: apply any technique to ensure any further code changes from other developer teams do not break the current functions of the application.
- Operating environment: ensuring the web-based system operates successfully within the factory's internal network environment, tailored for internal use rather than public release.
- After delivery:
  - Comprehensive handover of the codebase, including all source code and development documentation, to ensure maintainability and future development for the factory.
  - Deliver a technical manual and guidelines to assist the factory's technical staff in managing and troubleshooting.

5.2.2. Out Scope

- Development of features or functionalities not defined in the Software Requirement Specification chapter.
- Development of a responsive web application that can display and function correctly on a screen size below 1200px width.

5.3. Limitations & Exclusions

- The system is designed to read data from the factory's PLCs only, with no functionality to write or modify data, ensuring the integrity of the factory's operational processes.

II Project Management Plan

1. Overview

1.1. Scope & Estimation

#	WBS Item	Complexity	Est. Effort (man-days)
1.	***Initializing***		***15***
1.1	Kick-off meeting	Simple	2
1.2	Collect requirements	Complex	10
1.3	Create project Introduction	Medium	3
2.	***Planning***		***5***
2.1	Create a Project Management Plan	Complex	5
3.	***Executing***		
3.1	***Analysis***		***20***
3.1.1	Analyze requirements	Complex	10
3.1.2	Create a Software Requirement Specification	Complex	10
3.2	***Designing***		***50***
3.2.1	Design Monitoring UI & Architecture	Complex	10
3.2.2	Design Alarm UI & Architecture	Complex	10
3.2.3	Design Report UI & Architecture	Complex	10

3.2.4	Design Authentication UI & Architecture	Medium	5
3.2.5	Design User Management UI & Architecture	Medium	5
3.2.6	Write Software Design Document	Complex	10
3.3	***Implementation***		***83***
3.3.1	**Implement Monitoring Features**		***14***
3.3.1.1	View a circuit diagram of each station	Simple	4
3.3.1.2	Reflect real-time figures of circuit diagrams	Complex	10
3.3.2	**Implement Alarm Features**		***21***
3.3.2.1	View alarm history	Simple	2
3.3.2.2	Get notified via UI	Medium	5
3.3.2.3	Get notified via email	Complex	10
3.3.2.4	Manage alarms	Simple	4
3.3.3	**Implement Report Features**		***29***
	View historical reports	Medium	5
	Search historical reports	Simple	2
	View statistical charts	Complex	10
	Automated data capturing	Complex	10
	Export historical reports	Medium	5
3.3.4	**Implement Authentication**		***7***
	Login & Logout	Medium	5
	Authorize based on roles	Simple	2
3.3.5	**Implement User Management Features**		***12***
	Manage users	Complex	10

	View & Update profile	Simple	2
3.4	***Testing***		***18***
3.4.1	Create test cases	Medium	5
3.4.2	Unit test	Medium	6
3.4.3	System test	Complex	7
3.4.4	Create a Test document	Complex	5
4.	***Monitoring and controlling***		***20***
4.1	Track progress	Complex	10
4.2	Track documents	Complex	10
5.	***Closing***		***15***
5.1	Create Final Project Report	Complex	10
5.2	Create reports	Medium	5
	Total Estimated Effort (man-days)		***226***

Table 3 Project Estimation

1.2. Project Objectives

Quality

ID	Testing Stage	Test Coverage	No. of Defects	% of Defect	Notes
1	Reviewing	160	40	25	Reviewing both code and design
2	Unit Test	200	50	25	
3	System Test	60	20	33,33	

Table 4 Project Quality Objectives

Milestone Timeliness (%): 80

Allocated Effort (man-days): 14

1.3. Project Risks

ID	Risk Description	Impact	Possibility	Response Plans
1	Technology risk: unknown what technology is needed and satisfies the requirements	Critical	Medium	- Consult the client to scrutinize the technology of the old application. - Investigate and create a custom version of available technologies to match the requirements.
2	The client does not provide essential materials and information promptly	High	Medium	- Ask mentors to urge clients. - Make design and implementation as generic as possible. - Assume some parts of the requirements.
3	The application is not compatible with the PLC at the factory	High	Low	- Spend about a month to fix the issue. If the issue can't be fixed, we switch to the PLC simulation for demonstration.
4	Members fail to deliver tasks on time	Medium	Medium	- Host retrospective meetings to improve processes and overcome hindrances.

Table 5 Project risks

2. Management Approach

2.1. Project Process

The project will adopt the Scrum Agile methodology, characterized by 2-week sprints, for several key reasons:

- **Iterative Development:** Scrum facilitates iterative development, allowing the project team to focus on delivering high-value features in short, manageable cycles.
- **Continuous Improvement:** The methodology supports continuous improvement through regular sprint reviews and retrospectives, ensuring the project stays aligned with stakeholder needs.
- **Structured Flexibility:** Scrum provides a structured framework that is also flexible, accommodating changes in project requirements and enabling the team to adapt quickly.

- **Collaboration and Transparency:** The approach fosters a collaborative environment with high transparency among team members and stakeholders.
- **Rapid Feedback Incorporation:** Scrum ensures that stakeholder feedback is rapidly incorporated into the development process, leading to a final product that closely aligns with user needs and expectations.

2.2. Quality Management

To manage quality objectives effectively, our project implements the following strategies:

- **Effective Meeting Notes:** We maintain detailed notes from all meetings, centralizing goals, decisions, and action items to ensure alignment and accountability.
- **Responsive Team Meetings:** Team meetings are convened in response to new requirements from the client, ensuring that all changes are communicated to and understood by the entire team.
- **Centralized Requirement Management:** Requirements and their versions are centralized to maintain transparency and clarity on the project's scope and deliverables.
- **Standardized Ways of Working:** We establish and document standardized ways of working for all tools, platforms, and methods used in the project, ensuring consistency and shared knowledge across the team.
- **Style and Quality Gate Checks:** The main codebase is subject to code style guidelines and quality gate checks to uphold coding standards and ensure high-quality outputs.
- **Progress Reports:** Progress is reported three times weekly to keep the internal team informed and engaged with the project's development.
- **Weekly Technical Discussions and Support:** A dedicated session is held once a week for the team to discuss technical challenges, share knowledge, and support one another, fostering a collaborative environment and promoting continuous improvement.
- **Sprint Retrospectives:** Each sprint concludes with a retrospective to reflect on the process, celebrate successes, and identify areas for improvement.

These strategies enhance our existing quality management practices, including regular code reviews, continuous integration with automated testing, and periodic performance evaluations to ensure the system's stability and efficiency.

2.3. Training Plan

Training Area	Participants	When, Duration	Waiver Criteria
Java Spring Boot	All members	Week 1 - 3	Mandatory
Adobe Illustrator	Lê Xuân Đại	Week 1 - 3	Non-designers
Integrating PLC	All members	Week 3 - 5	Mandatory

Kubernetes	All members	Week 7 - 8	Mandatory

Table 6 Project's training plan

3. Project Deliverables

Sprint	Deliverable
1-2	Report 1 (Project Introduction Document)
2-3	Report 2 (Project Management Plan)
4-7	Report 3 (Software Requirement Specification)
6-9	Report 4 (Software Design Document)
8-11	Report 5 (Test Documentation)
11-12	Report 6 (Software User Guides)
12-13	Report 7 (Final Project Report)

Table 7 Project's deliverables

4. Responsibility Assignments

Role	Responsibility	PIC
Product Champion	- Specify user requirements. - Review products and documents.	Mr. Bùi Kim Thành
Product Owner	- Approve and sign the enterprise agreement and requirements.	Mr. Đỗ Tân Khoa
Scrum Master	- Manage process. - Track backlog. - Quality management. - Arrange meetings. - Support other team members.	Bùi Ngọc Huy
Business Analyst	- Get and analyze requirements directly from the Product Owner. - Produce and Manage documents.	Lê Tiến Thịnh, Trần Trung Kiên

DevOps	- Setup testing & development environment. - Design and implement CI & CD flows.	Bùi Ngọc Huy, Nguyễn Nhật Huy
Back-end	- Design system architecture. - Implement architecture design. - Collaborate with the Front-end team to integrate UI and APIs.	Nguyễn Nhật Huy, Bùi Ngọc Huy, Lê Tiến Thịnh
Front-end	- Implement UI design. - Collaborate with the Back-end team to integrate UI and APIs.	Lê Xuân Đại, Trần Trung Kiên
Designer	- Design wireframe & prototype.	Lê Xuân Đại

Table 8 Project's responsibility assignments

5. Project Communication

Communication Item	Who/ Target	Purpose	When, Frequency	Type, Tool, Method(s)
Working with supervisors	Supervisor and team members	- Review the progress. - Review project document. - Support the technical and business analysis. - Give advice and suggestions.	Weekly	Google Meet and Zalo / At University
Working in team	Team members	- Discuss technical and business topics. - Raise and handle problems. - Notify the essential events and notes of the project.	Always	Slack, Messenger, and Google Meet
Tracking progress	Team members	- Track the status of tasks. - Track issues and defects.	Always	Slack, Jira, and GitHub
Archives	Team members	- Share project materials & documents. - Upload meeting minutes.	Always	Confluence and OneDrive

Table 9 Project's communication plan

6. Configuration Management

6.1. Document Management

To effectively manage and document our project's information, we utilize a combination of OneDrive, Confluence, and Jira:

- Confluence: advanced document management with comments, versioning, etc.
- OneDrive: securely archive documents, records, images, and diagrams.
- Jira: assign and keep track of features, user stories, and tasks.

6.2. Source Code Management

We have opted to utilize the platform GitHub to manage our source code. By implementing this version control system, our team members can collaboratively work on the source code in a more efficient, convenient, and streamlined manner, enabling us to resolve conflicts with ease. The system is handy for reviewing changes made by collaborators, allowing for a thorough examination of modifications made over time.

6.3. Tools & Infrastructure

Category	Tools / Infrastructure
Core Technology	React Typescript (Frontend) and Spring Boot (Backend)
IoT	Apache PLC4X
Database	Microsoft SQL Server
Deployment server	AWS EC2 and VPS
Deployment Management	GitHub Actions and Kubernetes
IDEs/Editors	Visual Studio Code and IntelliJ IDEA
Diagramming	DrawIO, Mermaid, and PlantUML
Documentation	MS Office
Version Control	GitHub Actions (Source Codes), Confluence and OneDrive (Documents)
Project management	Jira (Tasks) and Slack (Communication)

Table 10 Project's tools and infrastructure

III Software Requirement Specification

1. Product Overview

The new web-based RMS system will replace Kim Tin's Factory's current RMS desktop application. The Web-Based Real-time Monitoring System offers a complete platform for factory operators to access real-time data, dynamic reports, and user-friendly interaction for monitoring energy consumption and production parameters at Kim Tin's MDF wooden board factory.

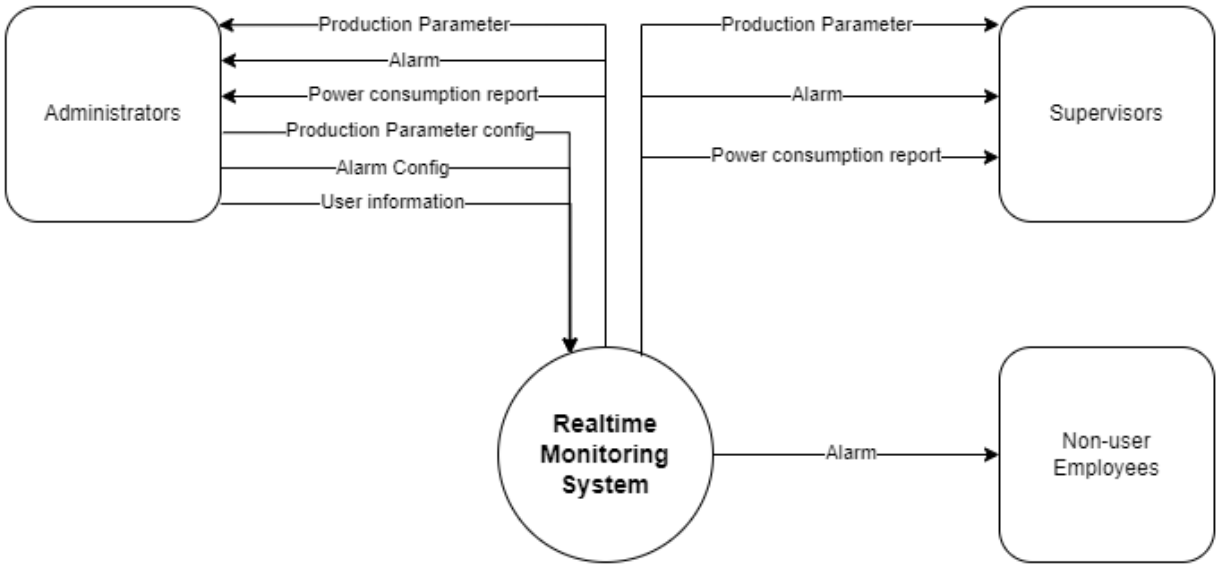


Figure 6 Context Diagram of the RMS

Refer to the Project Scope and Limitation section of the chapter Project Introduction for a detailed project scope. The Acceptance Criteria in that section are written from the business perspective. Here are the associated AC from a technical standpoint:

- AC1 - Features: a web-based Real-time Monitoring System with all the features mentioned in the Major Features section.
- AC2 – Real-time: Effectively real-time monitoring with acceptable cycle time.
- AC3 – Enhanced the overall UI/UX.
- AC4 - Ensure integrity: apply any technique to ensure any further code changes from other developer teams do not break the current functions of the application. *This will be accomplished by ensuring that all functional test cases pass with at least 75% line of code coverage in the continuous integration (CI) process.*

2. User Requirements

2.1. Actors

2.1.1. Unauthenticated User

Unauthenticated users are those who have yet to log in and can only access the login page.

2.1.2. Supervisor

A supervisor is a person who keeps an eye on the manufacturing process. They need real-time data on production parameters and factory reports to make quick, informed decisions.

2.1.3. Administrator

An administrator takes charge of overall system management, which includes user access and alarm configurations. Their responsibilities involve setting up user accounts, managing user permissions, and overseeing all alarm conditions.

2.1.4. System Handler

The System Handler is not a tangible actor or specific components within the system; it serves as a categorization actor for scheduled tasks that produce results for the actual actors. These tasks include data streaming, alarm notification distribution, and periodic reporting data capture.

System Handler represents a grouping of time-dependent processes critical for the functioning of the Real-time Monitoring System, ensuring timely execution and coordination of essential functions to benefit the system's users.

2.2. Usecase diagram

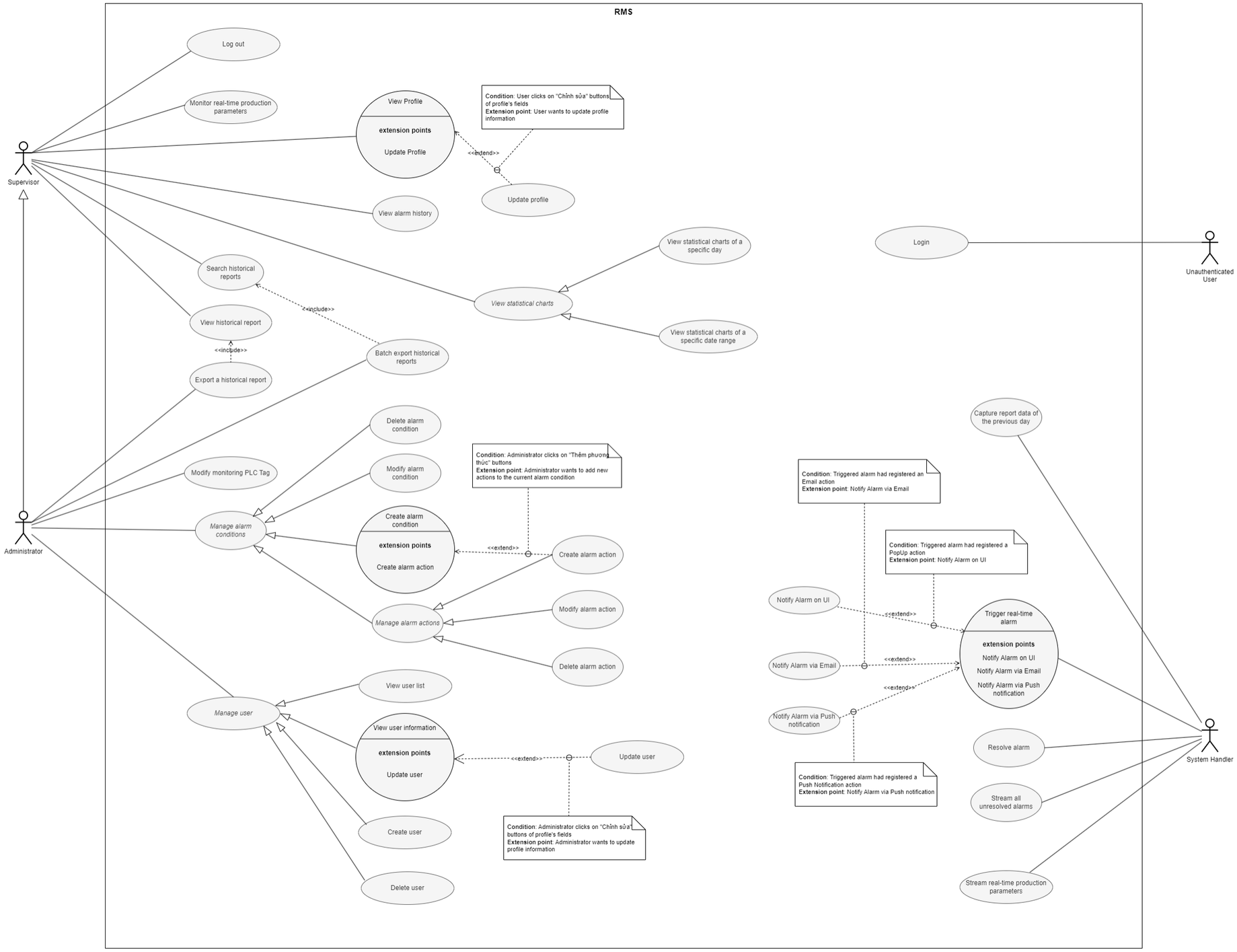


Figure 7 [Usecase diagram]

2.3. List of usecase with description

ID	Usecase	Feature group	Actors	Description
1	Login	Authentication and User management	Unauthenticated User	Unauthenticated users securely log in with their unique username and password combination.
2	View profile		Supervisor, Administrator	Users can view their information.
3	Update profile		Supervisor, Administrator	Users can update their information.
4	View user list		Administrator	Administrators can view a list of existing users in the system.
5	View user information		Administrator	Administrators can view information of an existing user the system.
6	Update user information		Administrator	Administrators can update information of existing users in the system.
7	Create user		Administrator	Administrators can create new users to grant them permissions to access the system.
8	Delete user		Administrator	Administrator can delete user accounts.
9	Monitor real-time production parameters	Monitoring	Supervisor, Administrator	Users can monitor all actual real-time production parameters of all stations at the factory.
10	Modify monitoring PLC Tag		Administrator	Administrators can modify the monitoring PLC Tag of each production parameter on system.
11	Stream production parameters		System Handler	The system continuously streams new production parameters from the factory with a delay of 1 second between 2 fetches.

12	View alarm history	Alarm	Supervisor, Administrator	Users can view the list of all alarm histories that were triggered in the past.
13	Receive alarm notifications on the UI		Supervisor, Administrator	Users using the app will receive a notification popup whenever a new alarm is triggered.
14	Receive alarm notifications via email		Supervisor, Administrator	Users with the email registered in an alarm action will receive an email whenever a new alarm is triggered.
15	Receive alarm push notifications via mobile phone		Supervisor, Administrator	Users who installed the ntfy application on their phone and subscribed to the RMS's notification channel will receive a push notification whenever a new alarm is triggered.
16	Quickly view all unresolved alarms		Supervisor, Administrator	Users can view the list of all unresolved alarms on the Monitoring ("Giám sát") screen of the RMS application.
17	Continuously check for alarm conditions		System Handler	The RMS system continuously checks against all current conditions for timely triggering, notification delivering and resolving.
18	Delete alarm condition		Administrator	Administrators can delete an alarm condition.
19	Modify alarm condition		Administrator	Administrators can modify the allowed fields of an alarm condition.
20	Create a new alarm condition.		Administrator	Administrators can create a new alarm condition, including accompanying notification actions.
21	Delete alarm action		Administrator	Administrators can delete an alarm action of an alarm condition.

22	Modify alarm actions		Administrator	Administrators can modify the list of alarm condition's accompanying actions and their information.
23	Create alarm action		Administrator	Administrators can create a new alarm action for an alarm condition.
24	View historical reports	Report	Supervisor, Administrator	Users can view the reports that were recorded in the past.
25	Search historical reports		Supervisor, Administrator	Users can find reports using filters and sorts.
26	View statistical charts of a specific day		Supervisor, Administrator	Users can view charts to gain insights into whose data belongs to reports within a specific day.
27	View statistical charts of a specific date range		Supervisor, Administrator	Users can view charts to gain insights into whose data belongs to reports within a specific date range.
28	Export a historical report		Administrator	Administrators can export a report to an Excel file, and export reports are based on predefined templates.
29	Batch Export historical reports		Administrator	Administrators can export reports to Excel files, and export reports are based on predefined templates.
30	Capture data from the previous day.		System Handler	The RMS system automatically captures data from the previous day daily.

Table 11 Usecases with description

3. Functional Requirements

3.1. System Functional Overview

3.1.1. Screen Flow

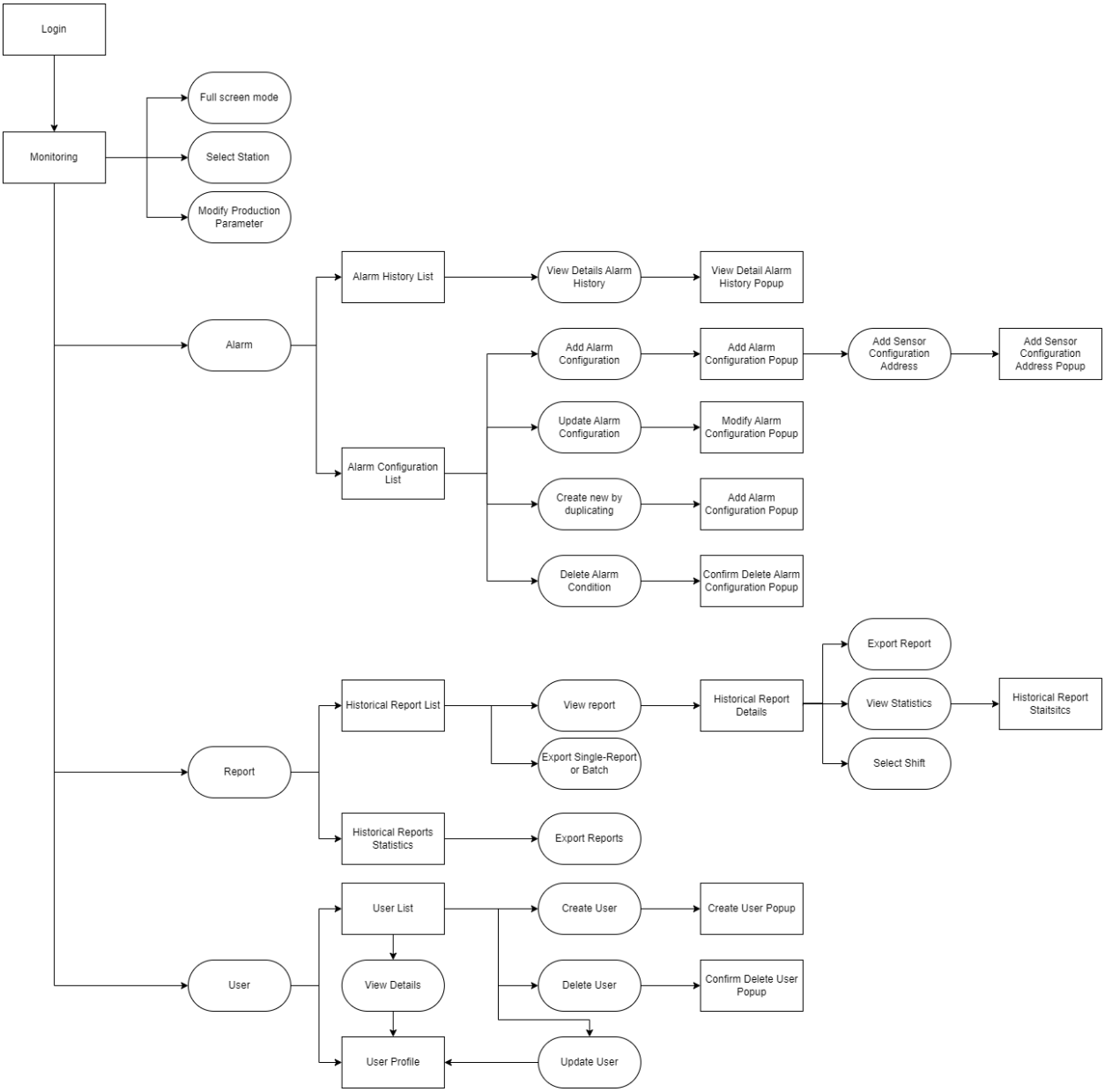


Figure 8 [Project's screen flow diagram]

3.1.2. Screen Description

No	Screen	Feature	Description
1	Login	Authentication	Unauthenticated users can log in using a valid combination of username and password.
2	View Main station	Monitoring	Streaming 294 different production parameters on the factory's Main station plant layout.
3	View TR11 Station		Streaming 38 different production parameters on the factory's TR11 station plant layout.
4	View TR12 Station		Streaming 230 different production parameters on the factory's TR12 station plant layout.
5	View TR30 Station		Streaming 52 different production parameters on the factory's TR30 station plant layout.
6	View TR31 Station		Streaming 38 different production parameters on the factory's TR31 station plant layout.
7	View TR32 Station		Streaming 38 different production parameters on the factory's TR32 station plant layout.
8	View TR33 Station		Streaming 38 different production parameters on the factory's TR33 station plant layout.
9	View TR34 Station		Streaming 70 different production parameters on the factory's TR34 station plant layout.
10	View TR42 Station		Streaming 90 different production parameters on the factory's TR42 station plant layout.
11	View TR52 Station		Streaming 70 different production parameters on the factory's TR52 station plant layout.
12	View TR72 Station		Streaming 38 different production parameters on the factory's TR72 station plant layout.
13	View TR82 Station		Streaming 124 different production parameters on the factory's TR82 station plant layout.

14	View All Meter Station		Streaming 908 production parameters on the factory's All Meter station plant layout.
15	Edit PLC Tag information popup		Allow the administrator to input the PLC Tag value of a production parameter.
16	View Alarm History List	Alarm	Displaying a list of alarm histories. Each alarm item in the list includes these columns: - Alarm type – "Loại". - PLC Tag – "Địa chỉ biến". - Station – "Trạm". - Message – "Thông báo". - Triggered At – "Ban đầu cảnh báo". - Resolved At – "Kết thúc cảnh báo". Users can perform these abilities with the list: - Search. - Sort by any column. - Show/hide any column but the "Thao tác" column. - Shrink/expand row distance. - Open the list in full screen.
17	View Alarm Detail popup		Displaying a detailed view of the associated information of an alarm history.

18	View Alarm Condition List		Displaying a list of alarm conditions.
			Each alarm condition item includes:
			- Condition type – “Loại”.
			- PLC Tag – “Địa chỉ biến”.
			- Severity – “Mức độ nghiêm trọng”, represented by the color of an associated dot.
			  o Red (#d32f2f) for “Khẩn cấp” type.
			  o Orange (#ed6c02) for “Quan trọng” type.
			  o Blue (#0288d1) for “Thông báo” type.
			- Message – “Thông báo”.
			- Check interval – “Chu kì kiểm tra”.
			- Time delay – “Độ trễ”.
			
			Users can perform these abilities with the list:
			- Search.
			- Sort by any column.
			- Show/hide any column but the “Thao tác” column.
			- Shrink/expand row distance.
			- Open the list in full screen.

19	Create an Alarm Condition Popup		A popup for creating a new alarm condition with three steps: Step 1: Condition config requires these fields: - Station (select) – “Trạm”. - Severity (select) – “Mức độ khẩn cấp”. - PLC Tag (select) – “Địa chỉ biến”. - Check interval (integer input) – “Chu kì kiểm tra”. - Time delay (integer input) – “Độ trễ”. - Predefine/Custom alarm (switch) – “Thiết lập cơ bản” / “Thiết lập nâng cao”. Step 2: Notification config requires these fields: - Message (text input) – “Thông báo”. - Notification action (select) – “Phương thức cảnh báo”. Step 3: Confirm - Display all input information from previous steps.

20	Edit an Alarm Condition Popup		A popup for updating an existing alarm condition with two steps: Step 1: Condition config requires these fields: - Station (disabled) – "Trạm". - Severity (select) – "Mức độ khẩn cấp". - PLC Tag (disabled) – "Địa chỉ biến". - Check interval (integer input) – "Chu kì kiểm tra". - Time delay (integer input) – "Độ trễ". Step 2: Configure the alarm's actions, which requires these fields: - Message (text input) – "Thông báo". - Notification action – "Phương thức cảnh báo": create/update/delete actions
21	View Historical Report List	Report	Displaying a list of historical reports. Each historical report item in the list includes these columns: - Order number – "STT". - Report type – "Cụm sản xuất". - Recording date – "Ngày xuất báo cáo". Users can filter a list of historical reports using these fields: - Report type (multiple select) – "Cụm sản xuất". - Range date of recording date: (select) – "Khoảng thời gian báo cáo". - Order by (select) – "Sắp xếp theo". - Type of ordering (select) – "Thứ tự sắp xếp". Users can export historical reports: - A specific historical report. - A list of selected historical reports. - A list of all historical reports. Users can view historical report details by clicking "Xem báo cáo" button.

22	View Historical Report Details		Displaying a detailed view of the associated information of a historical report. Users can view the associated information of a historical report by shift: - "Ca ngày" (tab). - "Ca sáng" (tab). - "Ca tối" (tab). Users can perform these abilities with the electrical meter reading tables: - Search. - Show/hide any column. - Open the table in full screen. Users can export the chosen historical report by clicking the "Tải báo cáo" button. Users can view the statistics of a report by clicking the button: "Xem thống kê".
23	View Historical Report Statistics for a specific date.		Displaying the charts to depict the total historical electricity consumption: - A pie chart depicts the total electricity consumption between shifts. - A bar chart to depict the total electricity consumption with time groups. - A stacked bar chart depicting the total electricity consumption by device. Users can customize the chart by showing/hiding selectively displaying relevant datasets. Users can export the chosen report by clicking the "Tải báo cáo" button. Users can view the history report by clicking the "Xem báo cáo" button.

24	View Historical Report Statistics for a specific date range across multiple reports.		Displaying the list of cards to depict the total historical electricity consumption with time groups. Displaying the charts to depict the total historical electricity consumption: - A pie chart depicts the total electricity consumption between report types. - A multiple-line chart to depict the total electricity consumption with report types over time. - A stacked bar chart to depict the total electricity consumption by device of each report type over time. Users can customize the chart by showing/hiding selectively displaying relevant datasets. Users can filter a specific date range to get historical report statistics using these fields: - Recording date range (date picker) – "Ngày bắt đầu"/"Ngày kết thúc". - Step (select), available values: o "Xem theo ngày". o "Xem theo tuần". o "Xem theo tháng". o "Xem theo năm". - Number of steps (integer input) – "Ngày" / "Tuần" / "Tháng" / "Năm". Users can export historical reports for chosen date ranges.
25	View User List	User management	Display a list of users. Each user item in the list includes: - No. – "STT". - Full name – "Họ và tên". - Username – "Tên đăng nhập". - Email. - Role – "Vai trò" Users can perform these abilities with the list: - Search. - Sort by any column.

			- Show/hide any column but the “Thao tác” column. - Shrink/expand row distance. - Open the list in full screen. User can manage user account by clicking these buttons: - “Tạo người dùng”. - “Xem chi tiết”. - “Xóa người dùng”.
26	View User Profile		Displaying a detailed view of the user's associated information. Users can update the information with the list: - Full name – “Họ và tên”. - Email. - Password – “Mật khẩu”. Users can delete their account by clicking the “Xóa tài khoản” button.
27	Create User Popup		A popup for creating a new user with the list fields: - Full name (text input) – “Họ và tên”. - Email (text input). - Role (select) – “Vai trò”. - Username (text input) – “Tên đăng nhập”. - Password (text input) – “Mật khẩu”. - Confirm password (text input) – “Nhập lại mật khẩu”.
28	Delete User Popup		A popup to confirm user deletion.

Table 12 List of screens and pop-ups grouped by feature

3.1.3. Screen Authorization

No	Screen	Feature	Unauthenticated User	Supervisor	Administrator
1	Login	Authentication	X		
2	View Main station	Monitoring		X	X
3	View TR11 Station			X	X
4	View TR12 Station			X	X
5	View TR30 Station			X	X
6	View TR31 Station			X	X
7	View TR32 Station			X	X
8	View TR33 Station			X	X
9	View TR34 Station			X	X
10	View TR42 Station			X	X
11	View TR52 Station			X	X
12	View TR72 Station			X	X
13	View TR82 Station			X	X
14	View All Meter Monitoring			X	X
15	Edit PLC Tag popup				X
16	View Alarm History List	Alarm		X	X
17	View Alarm History Detail Popup			X	X
18	View Alarm Condition List				X
19	Create an Alarm Condition Popup				X

20	Edit an Alarm Condition Popup				X
22	View historical report list	Report		X	X
23	View historical report details			X	X
24	View historical report statistics of a specific date			X	X
25	View historical report statistics of a date range across multiple reports			X	X
26	View user list	User management			X
27	View user profile			X	X
28	Create a new user popup				X
29	Update user popup				X
30	Delete user popup				X

Table 13 Screen authorization

3.1.4. Non-screen functions

No.	Feature	Description
1	Authorization	Enhances security by limiting user access based on predefined roles, preventing unexpected actions, and protecting sensitive data from potential breaches.

Table 14 List of non-screen functions

3.1.5. Entity Relationship Diagram

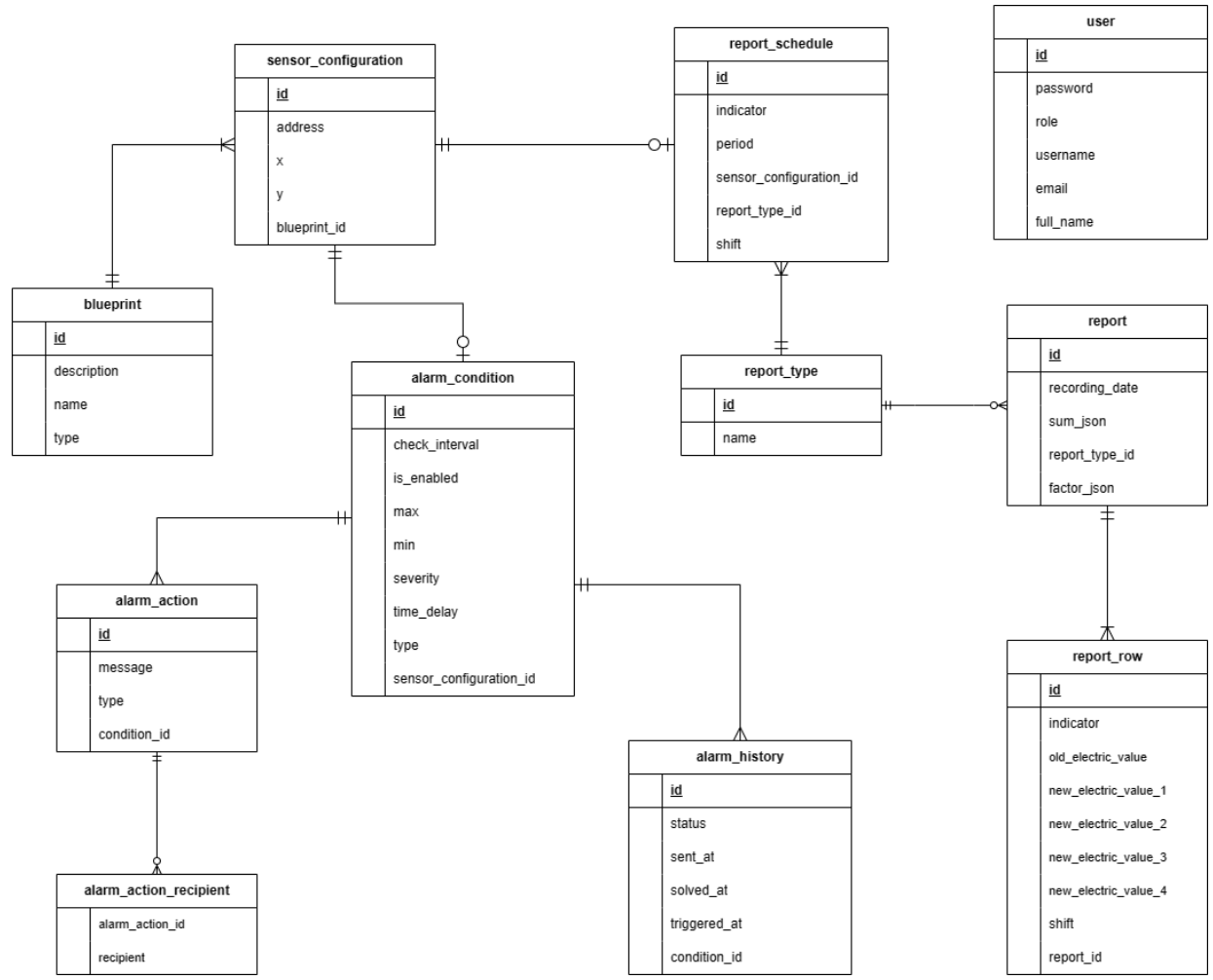


Figure 9 Entity Relationship Diagram

3.1.5.a) Blueprint

Description: For grouping and categorizing the SensorConfiguration records.

No	Field	Description
1	id	Unique identifier of the blueprint.
2	description	Meaningful description of the purpose of this blueprint.
3	name	To indicate what feature/screen this blueprint is used for.

4	type	To grouping the blueprints by features. Available values: - Monitoring: represent the Monitoring Station, shall have 13 blueprints of this type. - Alarm: a blueprint of this type is used to group all SensorConfiguration used for the Alarm feature. - Report: a blueprint of this type is for grouping all SensorConfiguration used for the Report feature.

Table 15 Blueprint entity's fields

3.1.5.b) SensorConfiguration

Description: Represent a production parameter or an associated PLC Tag.

No	Field	Description
1	id	Unique identifier of the Sensor Configuration.
2	address	The PLC Tag (some place called variable or address) must be in the format "%DB{data_block}:{offset}:{data_type}." In which data_block, offset, and data_type are required elements of a Tag in the PLC.
3	x	The horizontal coordinate of the Tag's value is displayed in the UI.
4	y	The vertical coordinate of the Tag's value is displayed in the UI.
5	blueprint_id	The id of the associated Blueprint.

Table 16 Sensor Configuration entity's fields

3.1.5.c) AlarmCondition

Description: Store all necessary information for a full alarm condition.

No	Field	Description
1	id	Unique identifier of the Alarm Condition.
2	check_interval	The number of seconds between 2 consecutive checks.
3	is_enabled	Represent whether this alarm condition enabled or not.
4	min	The minimum value of the allowed range for the value of the associated SensorConfiguration's PLC Tag; this can be null for the alarm condition type "Predefined"
5	max	The maximum value of the allowed range for the value of the associated SensorConfiguration's PLC Tag; this can be null for the alarm condition type "Predefined"

6	severity	Level of urgency. Available values: “Urgent,” “High,” and “Low.”
7	time_delay	The total time that this condition occurs continuously before triggering a new alarm.
8	type	Types of the condition. Available values: - Predefined: The alarm and logic have already been stored in the PLC. This type of alarm condition is stored in a PLC Tag with the bool data type, representing whether the condition is met. - Custom: represents the alarm condition that has not been stored in the PLC before, and its condition is stored in the RMS.
9	sensor_configuration_id	The associated Sensor Configuration is used to read the value from and check against the condition. If: - This belongs to a Blueprint with the “Predefined” type: the SensorConfiguration has the address field point to the PLC Tag storing the condition itself. - This belongs to a Blueprint with the “Custom” type: the SensorConfiguration has the address field point to the PLC Tag storing the value to check against.

Table 17 Alarm Condition entity’s fields

3.1.5.d) AlarmAction

Description: Represent an action triggered when associated alarm condition is matched.

No	Field	Description
1	id	Unique identifier of the Alarm Action.
2	message	The message to show or send to the user when the associated alarm occurs.
3	type	Type of channel this action will send to. Available values: - Popup: send the message to the application's UI. - Email: send a message to all registered emails. - PushNotification: send a notification to all prepared mobile devices.
4	condition_id	The associated Alarm Condition that triggered this alarm.

Table 18 AlarmAction entity’s fields

3.1.5.e) AlarmActionRecipient

Description: Store recipients for sending emails when the associated alarm actions are triggered.

No	Field	Description
1	alarm_action_id	The associated Alarm Action that has Email type.

2	recipient	The list of email recipients for sending emails as notifications.

Table 19 AlarmActionRecipient entity's fields

3.1.5.f) AlarmHistory

Description: Represent an alarm that was triggered in the application.

No	Field	Description
1	id	Unique identifier of the Alarm History.
2	status	Status of the alarm. Available values: - Triggered: a newly triggered alarm. - Sent: an alarm with an associated list of actions performed successfully. - Solved: an alarm with the associated condition is no longer met since this alarm was created.
3	sent_at	The exact time the alarm's status is changed to "Sent."
4	solved_at	The exact time the alarm's status is changed to "Solved."
5	triggered_at	The exact time the alarm's status is changed to "Trigger."
6	condition_id	The associated Alarm Condition that triggered this alarm.

Table 20 Alarm History entity's fields

3.1.5.g) ReportSchedule

Description: Contains essential information for automated report generation.

No	Field	Description
1	id	Unique identifier of the ReportSchedule.
2	indicator	Unique identifier of a report row that the electricity consumption figure belongs to.
3	period	Period of the captured electricity figure. Available values: - Old: the starting point of a shift. - New1: Value at the first checkpoint the electricity consumption figure is captured. - New2: Value at the second checkpoint the electricity consumption figure is captured. - New3: Value at the third checkpoint the electricity consumption figure is captured. - New4: Value at the fourth checkpoint the electricity consumption figure is captured.

4	shift	Shift when the electricity consumption figure is captured. Available values: - I: the first shift from 6:00 a.m. to 6:00 p.m. - II: the second shift from 6:00 p.m. to 6:00 a.m.
5	report_type_id	The associated Report Type that the electricity consumption figures belong to.
6	sensor_configuration_id	The associated Sensor Configuration contains the PLC Tag's address, whose value represents the electricity consumption figure.

Table 21 ReportSchedule entity's fields

3.1.5.h) ReportType

Description: For grouping and categorizing the Report records.

No	Field	Description
1	id	Unique identifier of the ReportType.
2	name	Name of the report type.

Table 22 ReportType entity's fields

3.1.5.i) Report

Description: Represent a report including electricity consumption figures within a day.

No	Field	Description
1	id	Unique identifier of the Report.
2	recording_date	The date when the report is created.
3	sum_json	A stringified JSON file that stores all required calculated sums for quick querying.
4	factor_json	A stringified JSON file that stores all required factors for calculation operations.
5	report_type_id	The associated Report Type that the report belongs to.

Table 23 Report entity's fields

3.1.5.j) ReportRow

Description: Represent a row in an associated Report.

No	Field	Description
1	id	Unique identifier of the ReportRow.
2	indicator	Unique identifier of a report row that the electricity consumption figure belongs to.

3	shift	Shift when the electricity consumption figure is captured. Available values: - I: the first shift from 6:00 a.m. to 6:00 p.m. - II: the second shift from 6:00 p.m. to 6:00 a.m.
4	old_electric_value	The electricity consumption figure at the starting point of a shift.
5	new_electric_value1	The electricity consumption figure at the first checkpoint.
6	new_electric_value2	The electricity consumption figure at the second checkpoint.
7	new_electric_value3	The electricity consumption figure at the third checkpoint.
8	new_electric_value4	The electricity consumption figure at the fourth checkpoint.
9	report _id	The associated Report that the report row belongs to.

Table 24 ReportRow entity's fields

3.1.5.k) User

Description: Represent an end user using the application.

No	Field	Description
1	id	Unique identifier of the User.
2	password	The password for the user to log in along with the username.
3	role	The role determines the user's permissions to perform specific operations. Available values: - Admin: the role that can manage other accounts. - Supervisor: the role provides access to basic operations.
4	username	The username for the user to log in along with the password.
5	email	The user's email for contact.
6	full_name	The full name of the user.

Table 25 User entity's fields

3.2. Feature Authentication and User Management

3.2.1. Login

- Function trigger: When an unauthenticated user clicks the "Đăng nhập" button.
- Screen layout:

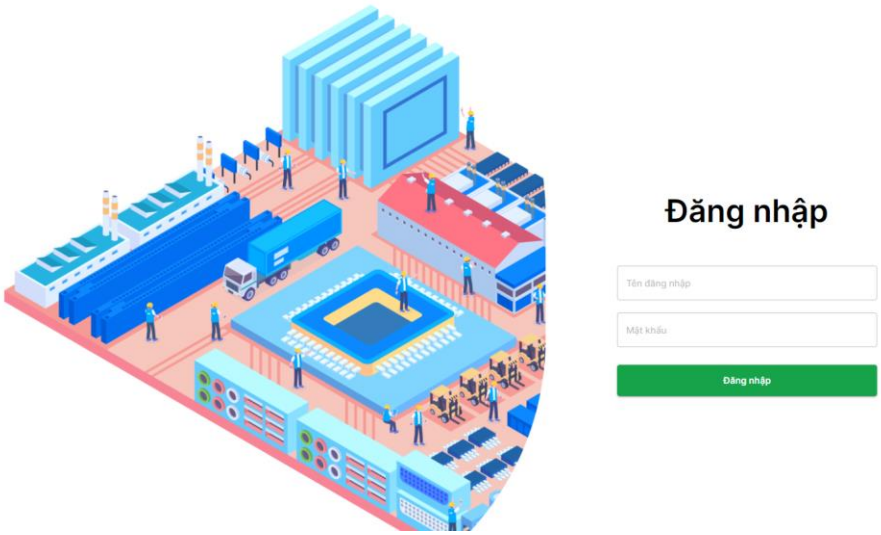


Figure 10 Login screen layout

- Function description: Unauthenticated users log into the system with a username and password to verify their role and get access to allowed resources and screens.
- Data validation:
  - "Tên đăng nhập"
    - Type: text field.
    - Min length: 2 .
    - Max length: 50.
  - "Mật khẩu"
    - Type: text field.
    - Min length: 8.
    - Max length: 16.

3.2.2. View user profile

- Function trigger: Users click on the avatar item from the top right corner of the navbar, then select the "Thông tin cá nhân" item from the dropdown menu.
- Function description: Users can view their information.
- Screen layout:

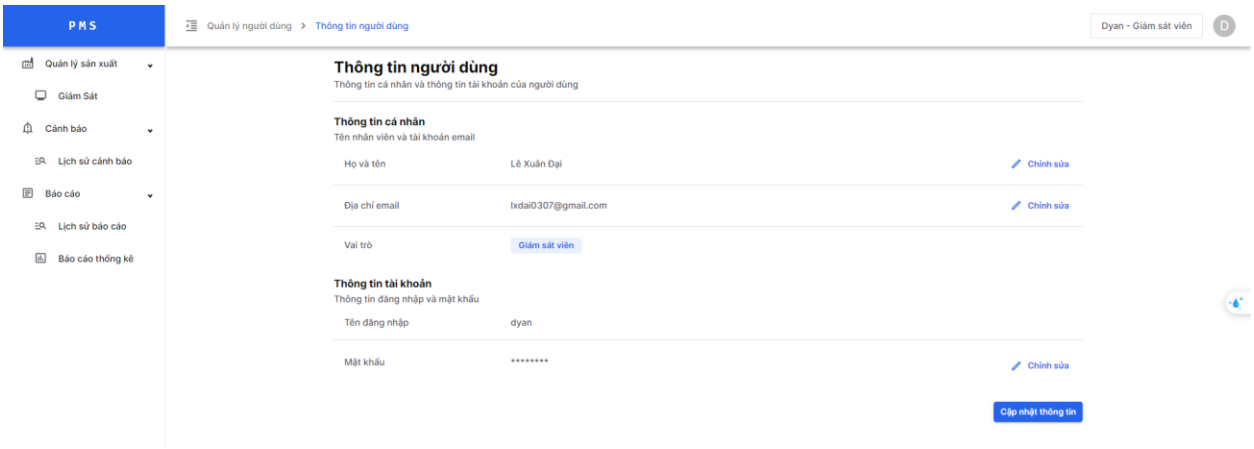


Figure 11 View user profile screen layout

- Function details: Users can view their profile information, which includes:
  - "Họ và tên".
  - "Địa chỉ email".
  - "Vai trò".
  - "Tên đăng nhập".

3.2.3. Update profile

- Function trigger: Users click on the avatar item from the top right corner of the navbar, then select the "Thông tin cá nhân" item from the dropdown menu and click the "Chỉnh sửa" button to update an item.
- Function description: Users update their information.
- Screen layout:

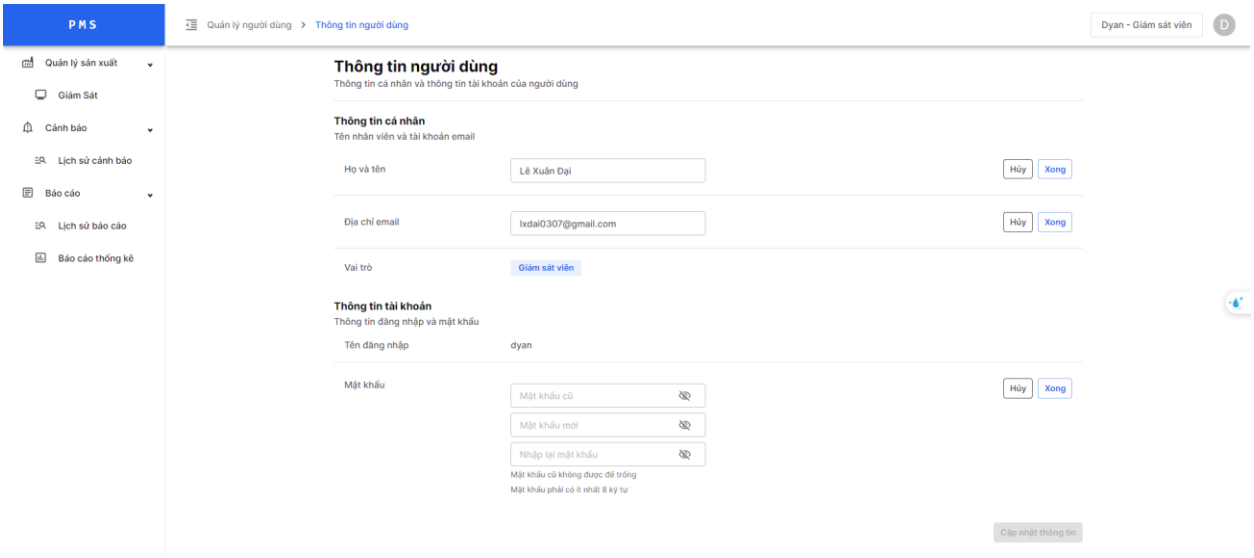


Figure 12 Update profile screen layout

- Function details:
    - Users can update user information, which includes:
        - “Họ và tên”.
        - “Địa chỉ email”.
        - “Mật khẩu”.
    - To update password:
        - Enter the current password.
        - Enter a new password.
        - Confirm the new password.
- Business Rules: BR-02, BR-03, BR-04.
- Data Validation:
    - “Họ và tên”
        - Type: text field.
        - Min length: 2.
        - Max length: 50.
    - “Địa chỉ email”
        - Type: text field.
        - Must be in a valid email format.
    - “Mật khẩu cũ”
        - Type: hidden text field.
        - Min length: 8.
        - Max length: 16.
    - “Mật khẩu mới” and “Nhập lại mật khẩu”
        - Type: hidden text field.
        - Min length: 8.
        - Max length: 16.
        - Two values must be the same.

3.2.4. View User List

- Function trigger: Administrators click on the “Quản lý người dùng” on the left sidebar of the screen.
- Function description: Administrator can view a system user list.
- Screen layout:

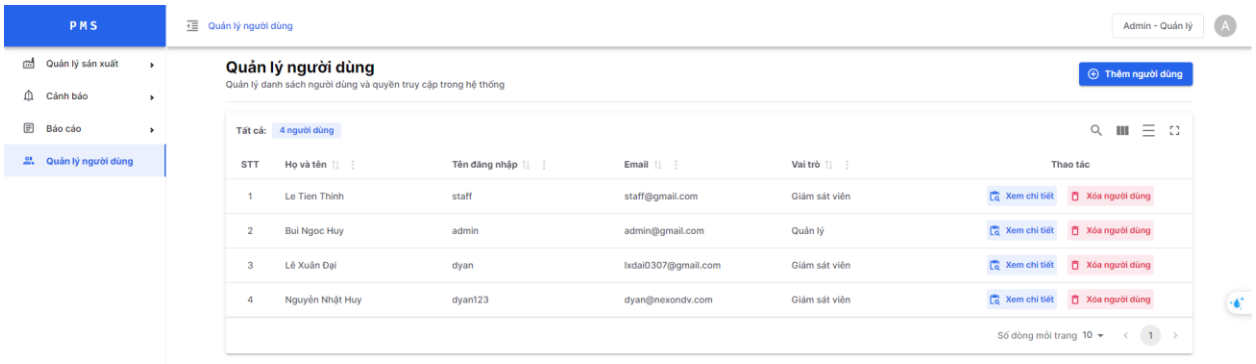


Figure 12 View user list screen layout

- Function detail:
    - Administrators can view the list of users and can perform these abilities with the list:
        - Search.
        - Sort by any column.
        - Show/hide any column but the “Thao tác” column.
        - Shrink/expand row distance.
        - Open the list in full screen.
    - Administrators can manage user accounts by clicking these buttons:
        - “Tạo người dùng”
        - “Xem chi tiết”
        - “Xóa người dùng”
- Business rules: BR-06.

3.2.5. Create Users

- Function trigger: On the top right corner of the “View User List” screen, administrators click on the “Thêm người dùng” button, and it shows a pop-up to fill out the user information.
- Function description: The administrator creates a new user.
- Screen layout:

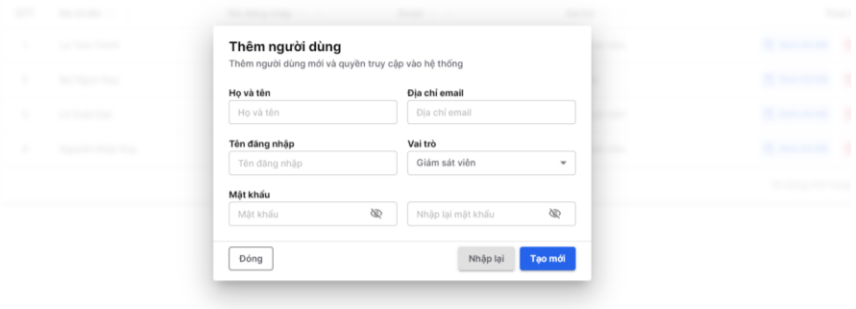


Figure 13 Create a user screen layout

- Function detail: Administrators shall fill out the fields, includes:
    - "Họ và tên".
    - "Địa chỉ email".
    - "Vai trò".
    - "Tên đăng nhập".
    - "Mật khẩu".
    - "Xác nhận mật khẩu".
- Business Rules: BR-01, BR-02, BR-06.
- Data Validation:
    - "Họ và tên"
        - Type: text field.
        - Min length: 2.
        - Max length: 50.
    - "Địa chỉ email"
        - Type: text field.
        - Must be in a valid email format.
    - "Tên đăng nhập"
        - Type: text field.
        - Min length: 2.
        - Max length: 50.
    - "Vai trò"
        - Type: select.
        - Available values: "Giám sát viên" and "Quản lý".
    - "Mật khẩu" and "Nhập lại mật khẩu"
        - Type: hidden text field.
        - Min length: 8.
        - Max length: 16.
        - Two values must be the same.

3.2.6. View User Information

- Function trigger: On the "View User List" screen, administrators click the "Xem chi tiết" button for an item in the user list.
- Function description: The administrator views information of users using the system.
- Screen layout:

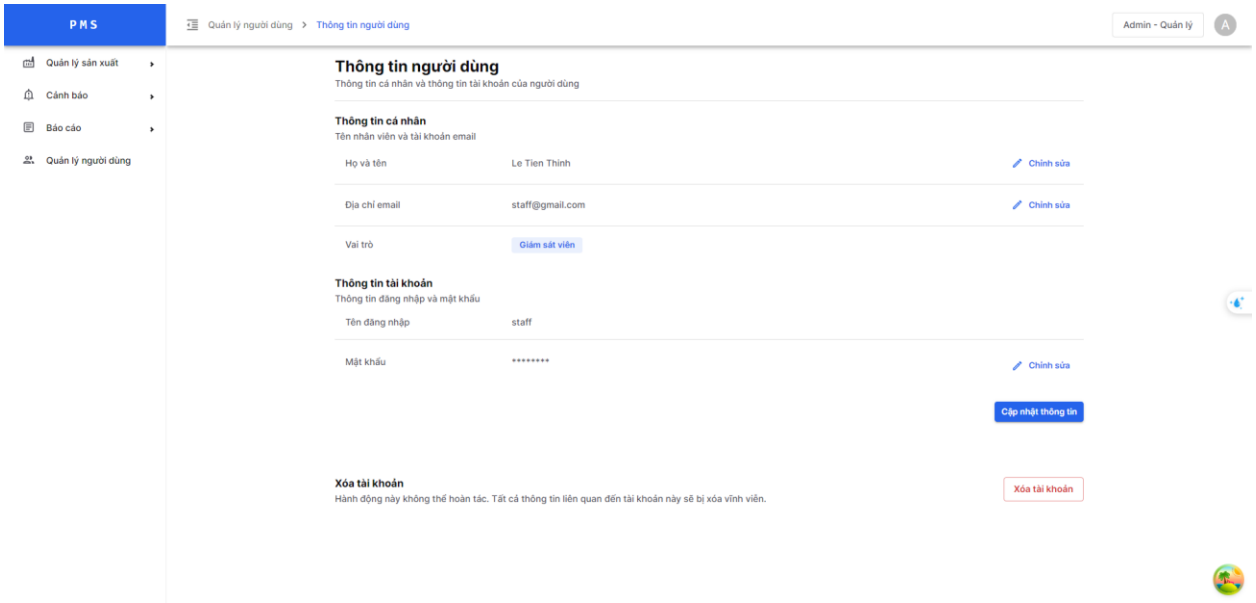


Figure 14 View user information screen layout

- Function detail: Users can view user information, which includes:
  - "Họ và tên".
  - "Địa chỉ email".
  - "Vai trò".
  - "Tên đăng nhập".
- Business rules: BR-06.

3.2.7. Update User Information

- Function trigger: On the "View User List" screen, administrators click the "Xem chi tiết" button for an item in the user list and click the "Chỉnh sửa" button to update an item.
- Function description: The administrator can update the information of users using the system.
- Screen layout:

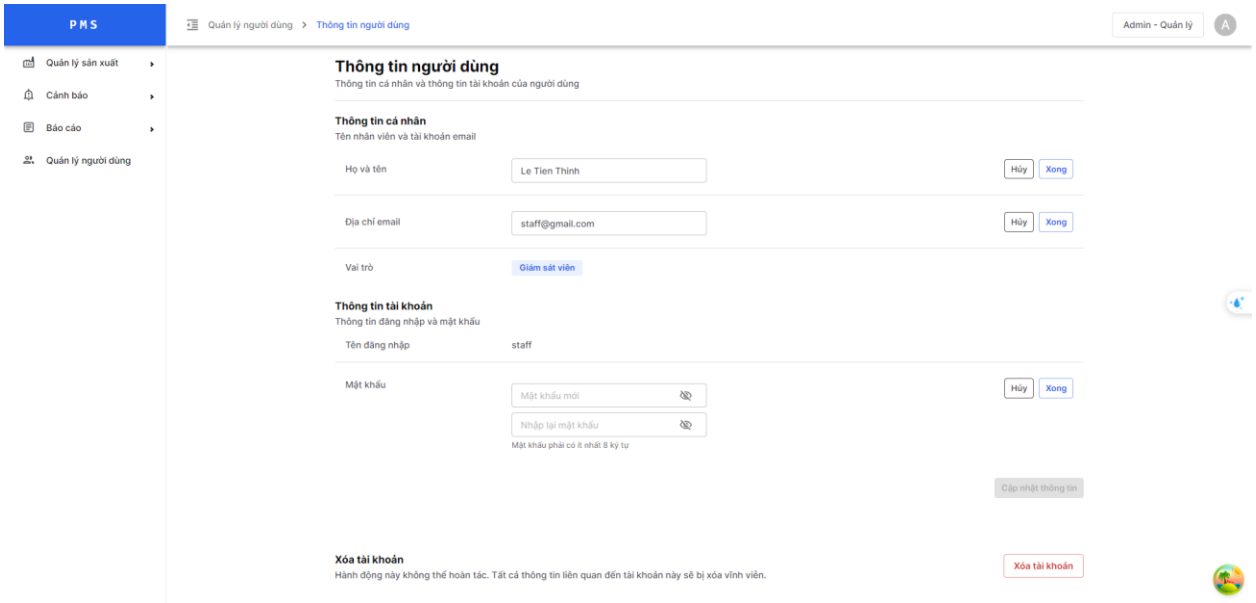


Figure 15 Update User Information screen layout

- Function detail:
    - Users can update user information, which includes:
        - "Họ và tên".
        - "Địa chỉ email" (This information cannot be changed to an existing one).
        - "Mật khẩu".
    - To update password:
        - Enter a new password.
        - Confirm new password.
- Business rules: BR-02, BR-03, BR-04, BR-06.
- Data Validation:
    - "Họ và tên"
        - Type: text field.
        - Min length: 2.
        - Max length: 50.
    - "Địa chỉ email"
        - Type: text field.
        - Must be in a valid email format.
    - "Mật khẩu mới" and "Nhập lại mật khẩu"
        - Type: hidden text field.
        - Min length: 8.
        - Max length: 16.
        - Two values must be the same.

3.2.8. Delete Users

- Function trigger: On the “View User List” screen, administrators click the “Xóa người dùng” button for an item in the user list.
- Function description: Administrator deletes user accounts.
- Screen layout:

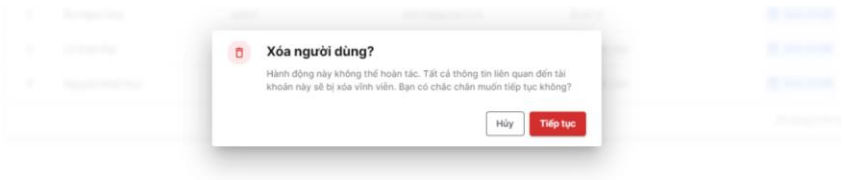


Figure 16 Delete user confirmation popup.

- Function detail: Administrators must confirm deletion before removing users.
- Business rules: BR-05

3.3. Feature Monitoring

3.3.1. Production Parameter Monitoring

- Function trigger: Users click on the “Giám sát” item on the left sidebar of the screen.
- Function description: Users visually view the real-time production parameters on each station’s plant layout. Display a cell with “.....” value for those not streamed from the RMS backend.
- Screen layout:
    - Main Station

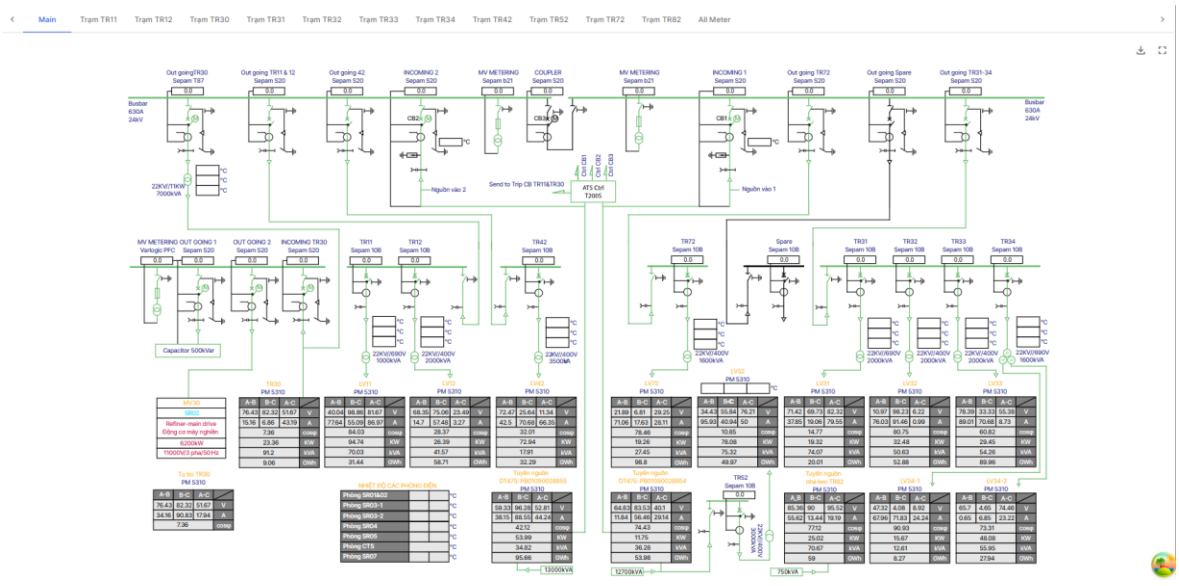


Figure 17 Main station electrical schematic diagram screen.

- TR11 Station

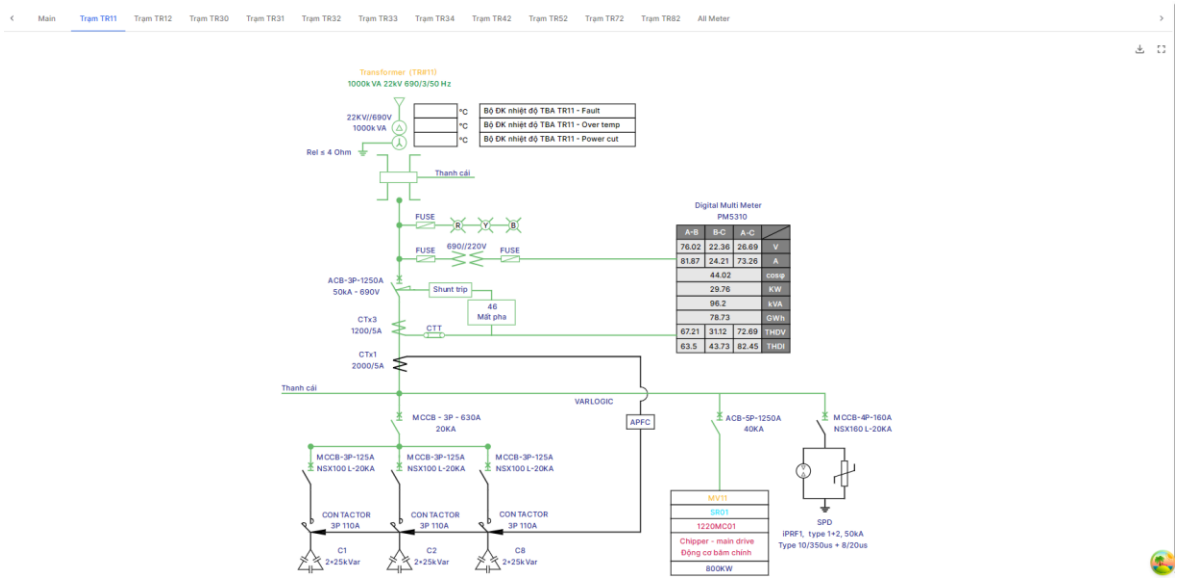


Figure 18 TR11 station electrical schematic diagram screen.

- TR12 Station

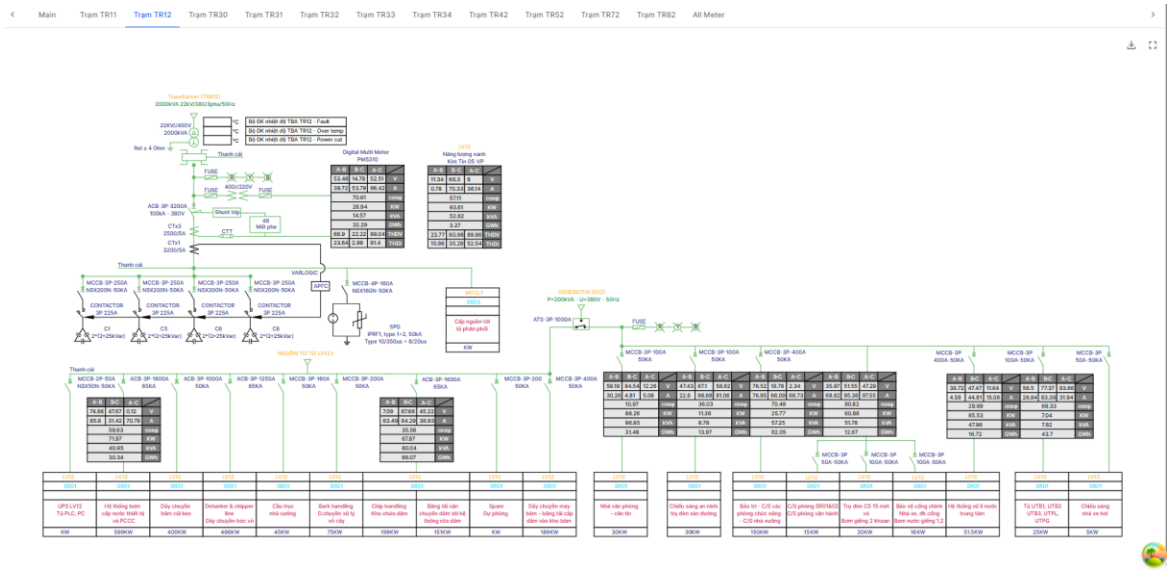


Figure 19 TR12 station electrical schematic diagram screen.

- TR30 Station

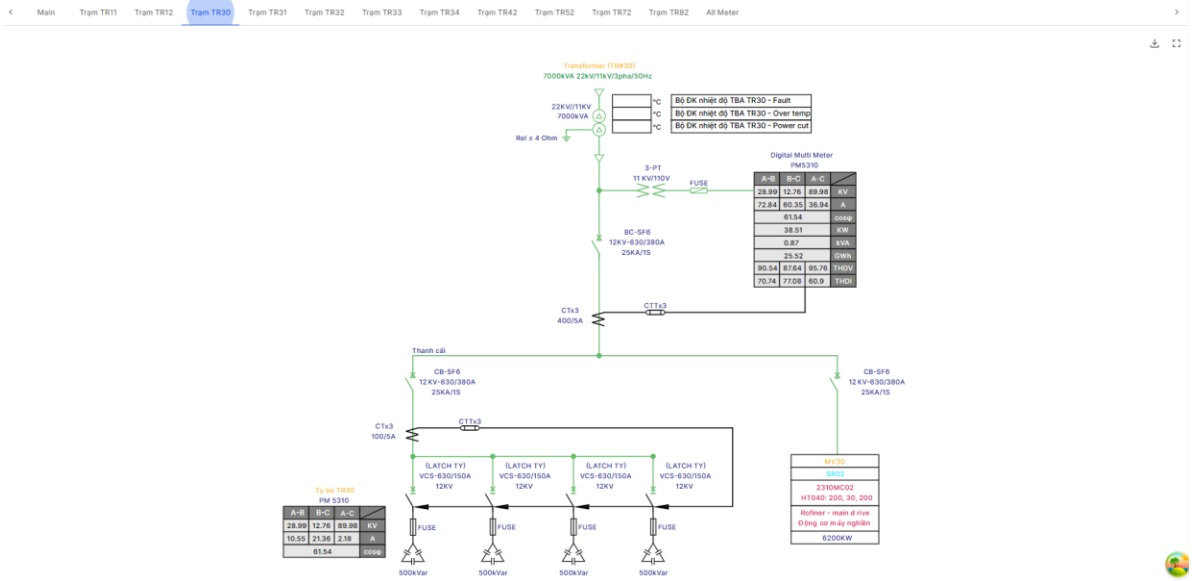


Figure 20 TR30 station electrical schematic diagram screen.

- TR31 Station

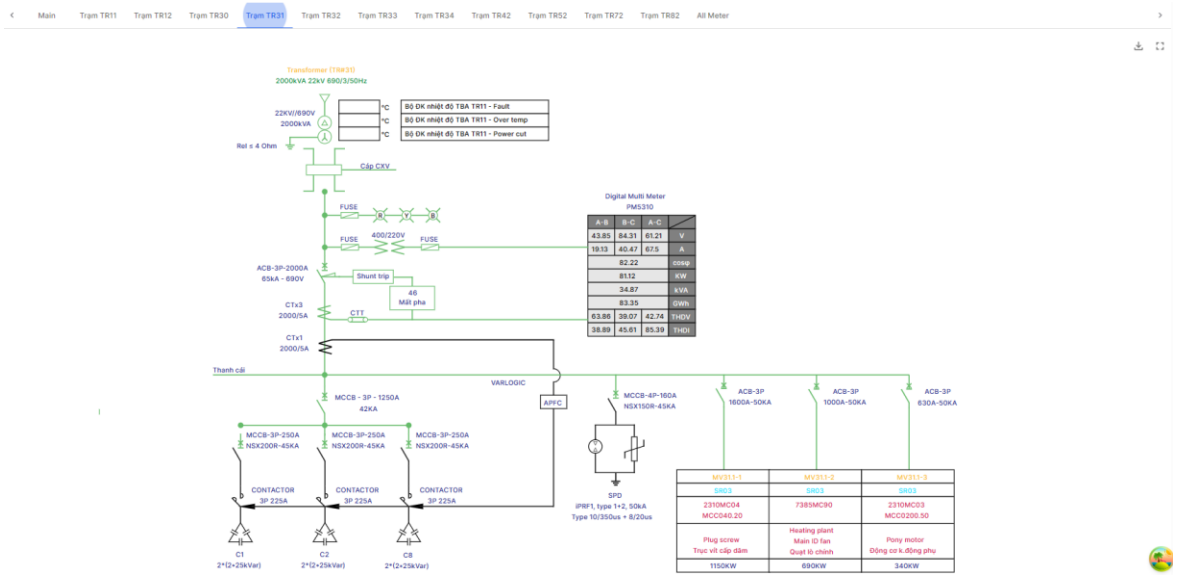


Figure 21 TR31 station electrical schematic diagram screen.

- TR32 Station

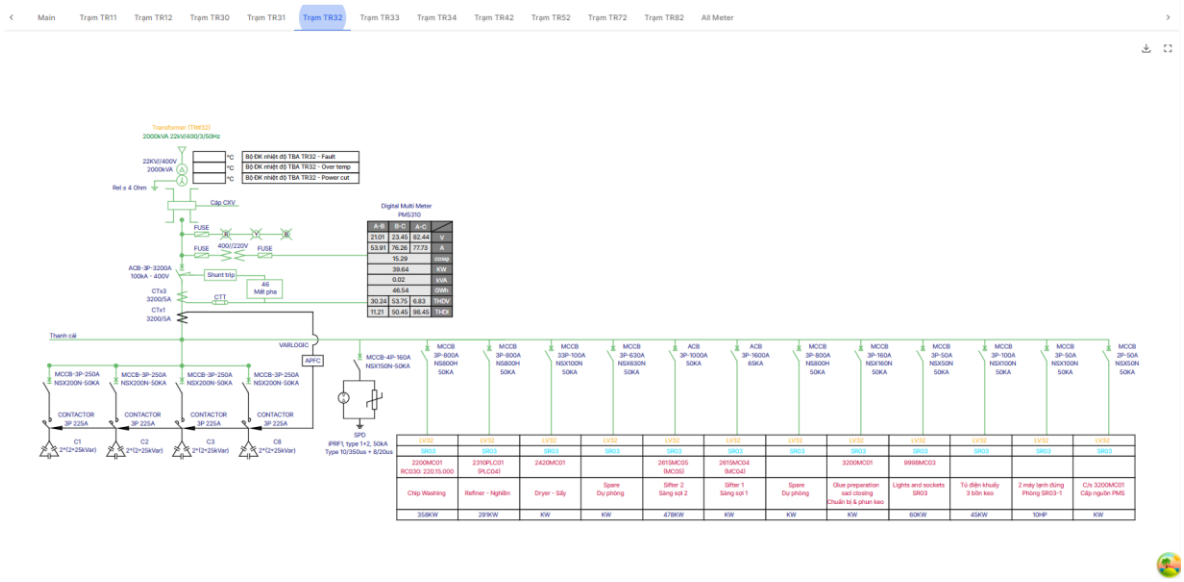


Figure 22 TR32 station electrical schematic diagram screen.

- TR33 Station

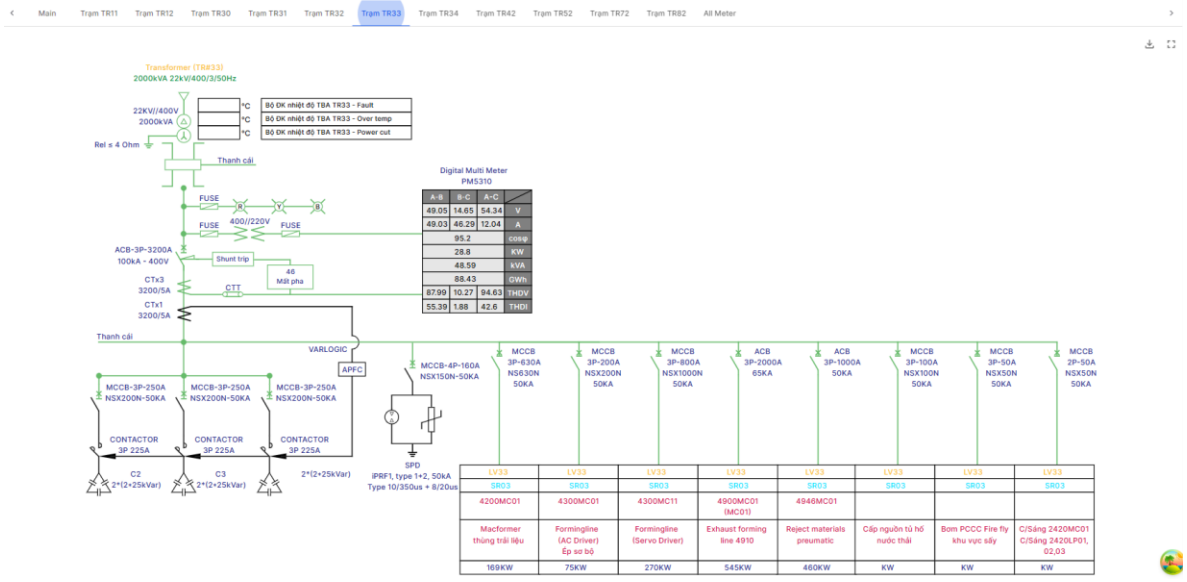


Figure 23 TR33 station electrical schematic diagram screen.

- TR34 Station

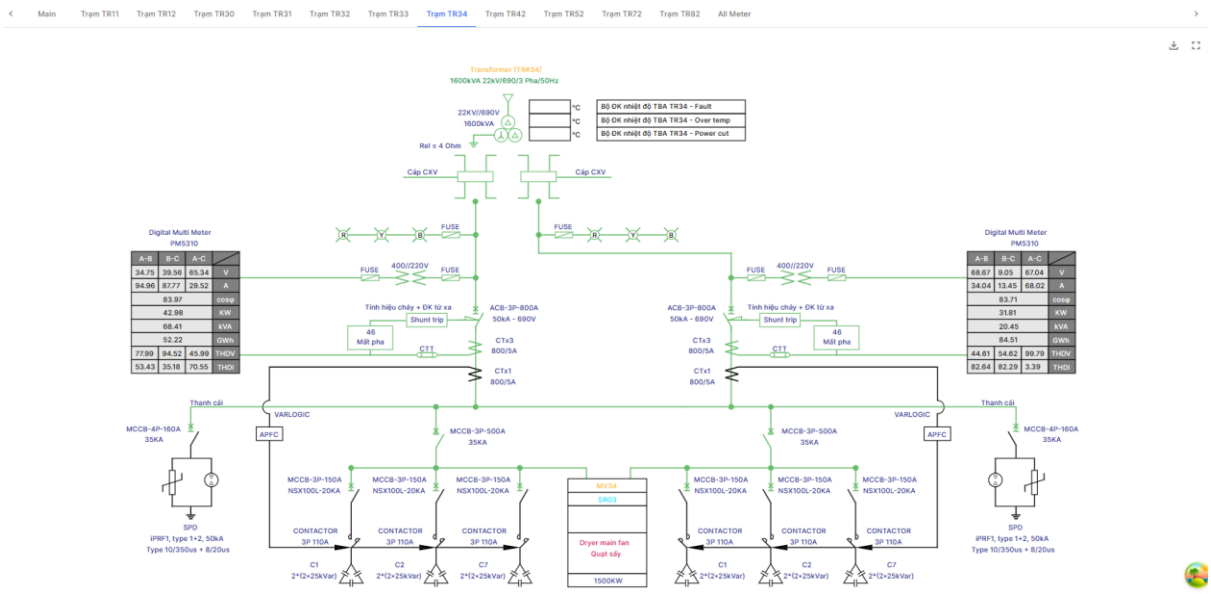


Figure 24 TR34 station electrical schematic diagram screen.

- TR42 Station

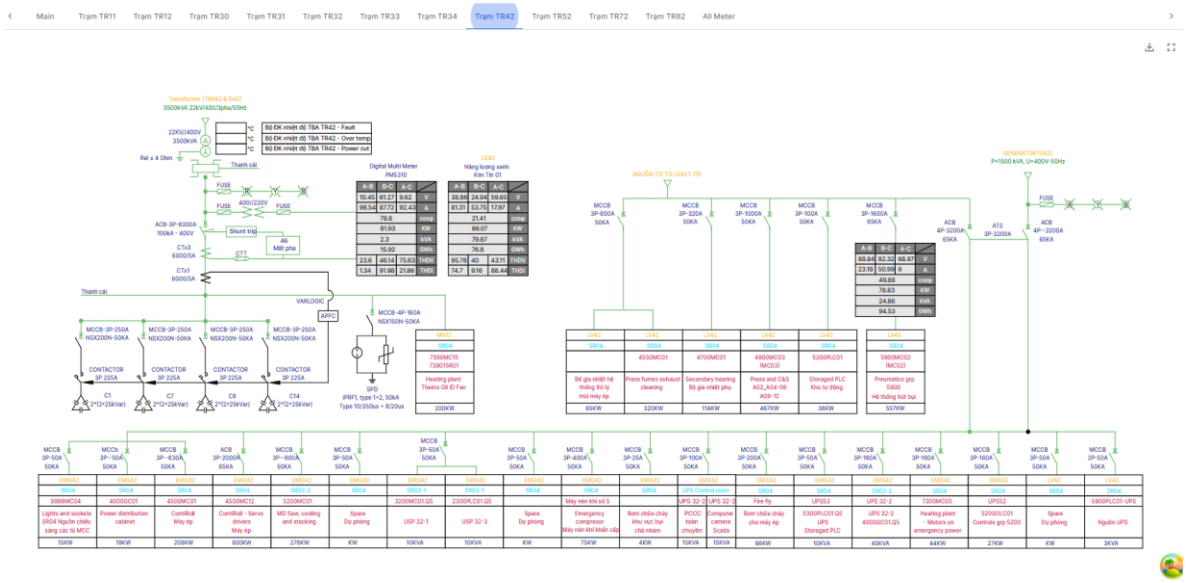


Figure 25 TR42 station electrical schematic diagram screen.

- TR52 Station

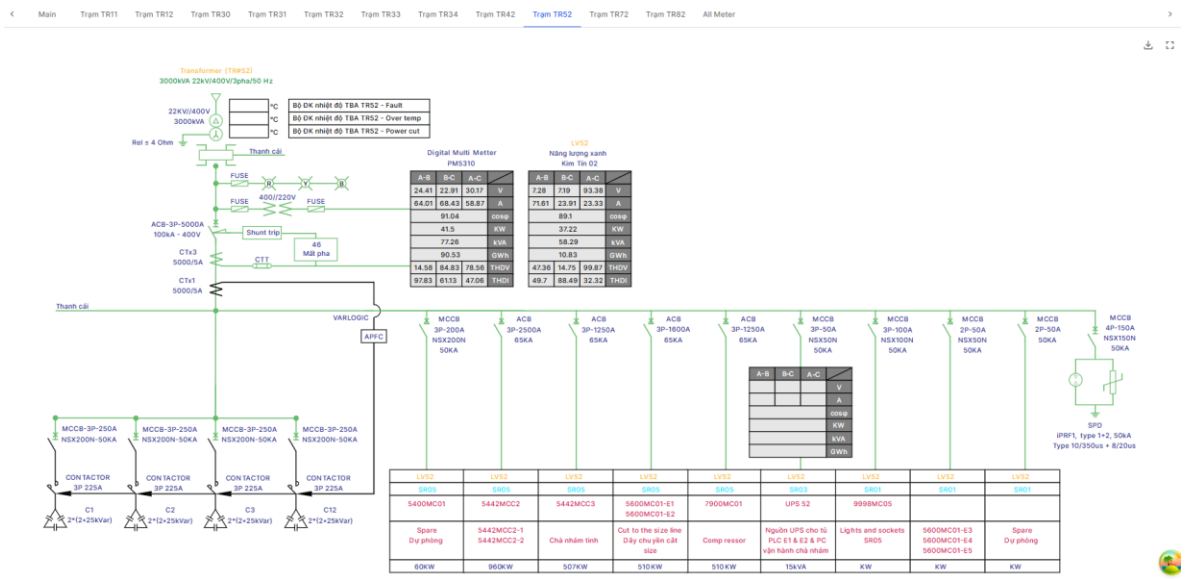


Figure 26 TR52 station electrical schematic diagram screen.

- TR72 Station

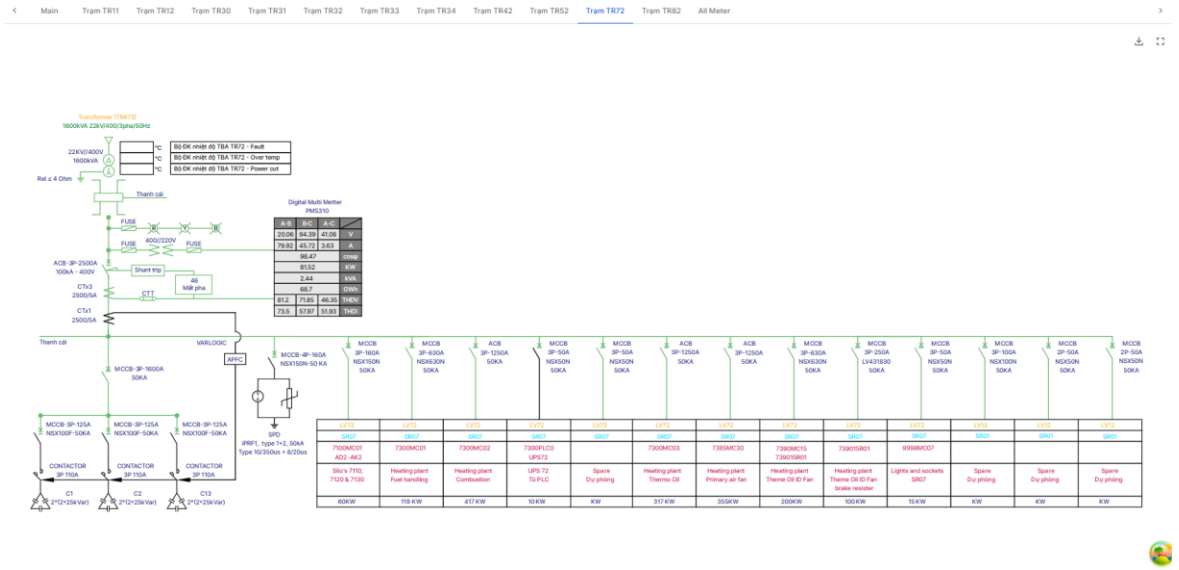


Figure 27 TR72 electrical schematic diagram screen.

- TR82 Station

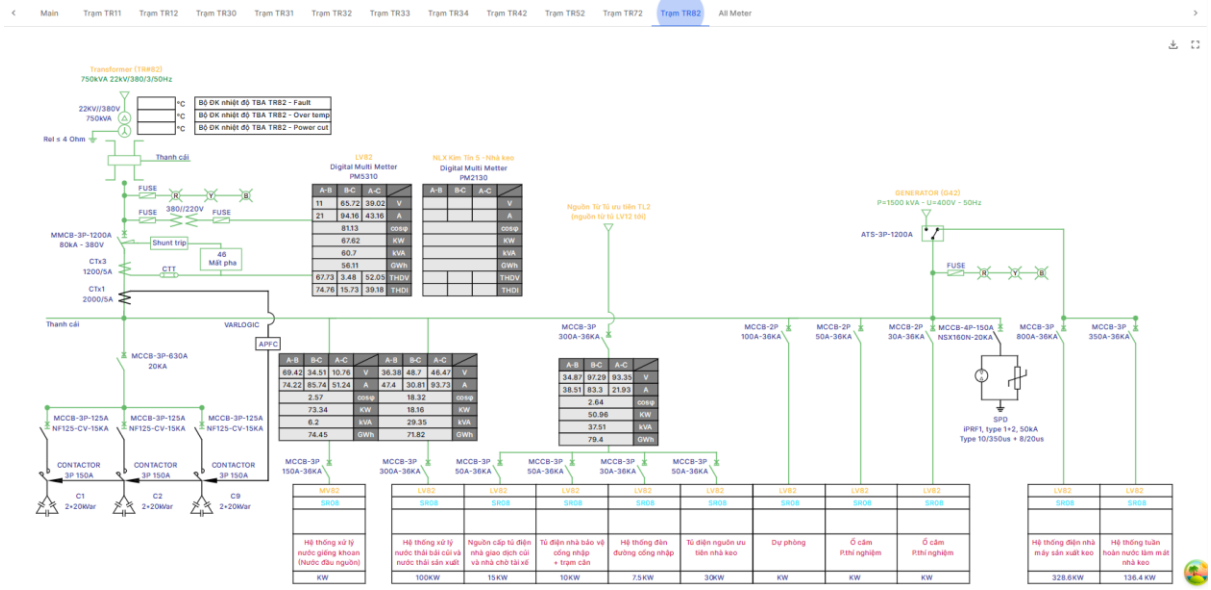


Figure 28 TR82 electrical schematic diagram screen.

- All Meter Station

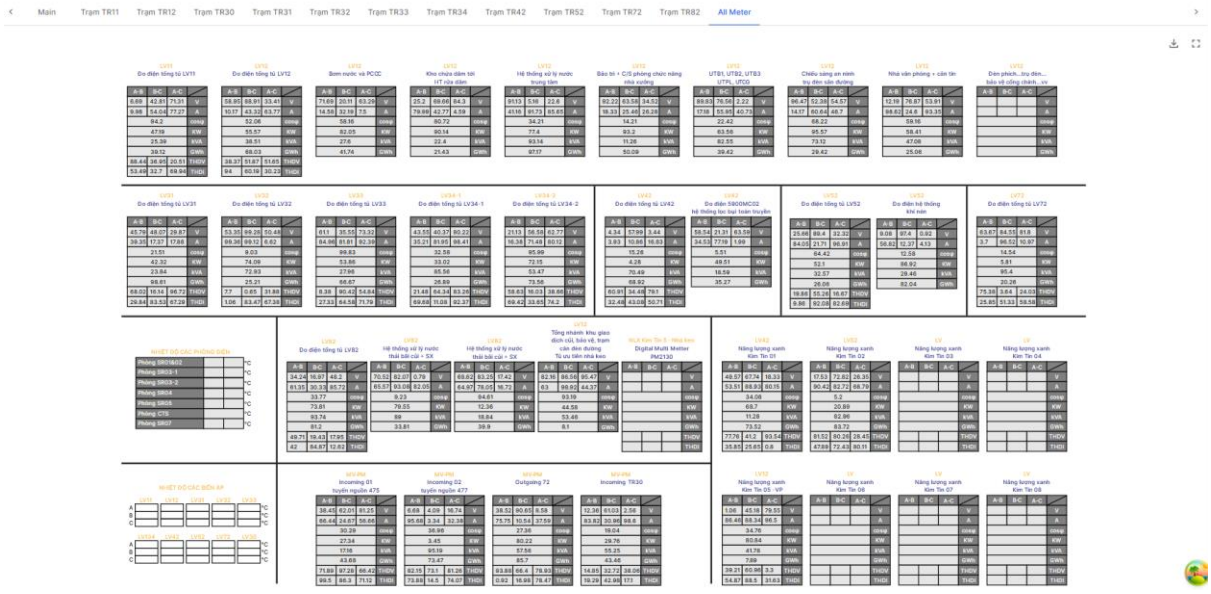


Figure 29 All meter station electrical schematic diagram screen.

- Function detail: Users shall be able to quickly navigate between station electrical schematic diagrams to view each station's production parameters in the factory.
- Business rules: BR-24.

3.3.2. Modify Production Parameter

- Function trigger: On any station monitoring screen, the administrator shall click the button “Bật chế độ thiết lập” on the top right corner of the screen.
- Function description: The administrator hovers over the production parameter that needs to be updated and then clicks the Pencil button. A popup will appear for the administrator to enter the required information of the PLC Tag associated with the production parameter.
- Screen layout:
    - Hovering state:

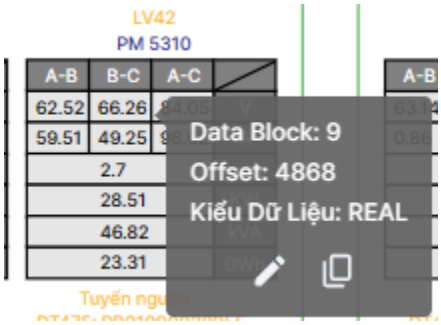


Figure 30 Modify production parameter tooltip.

    - Editing popup

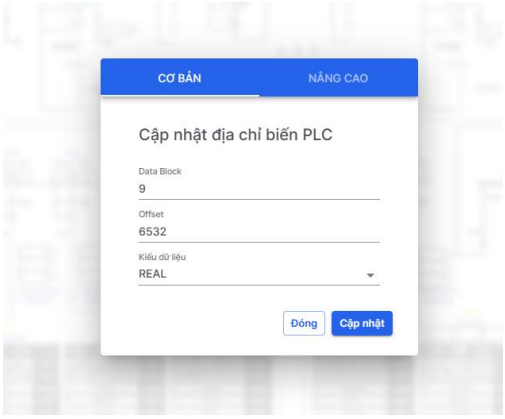


Figure 31 Editing sensor configuration popup.

- Function detail: The administrator shall be able to edit the information of the associated PLC Tag via two modes:
    - Basic (“Cơ bản”): administrator inputs the fields Data block, Offset, Data type.
    - Advance (“Nâng cao”): administrator inputs a string in the format %DB{data_block}:{offset}:{data_type}
- Business rules: BR-07, BR-08
- Data validation:
    - “Địa chỉ” in the “Nâng cao mode”:
        - Type: text field

- Must be in the format "%DB{data_block}:{offset}:{data_type}". In that:
    - data_block:
        - Type: integer
    - offset:
        - Type: double
    - data_type:
        - Type: text
        - Format: uppercase
        - Must be a valid data type for the PLC.
- Example of correct input: “%DB100:2000:REAL”, “%DB2:2.0:BOOL”

3.3.3. Stream Monitoring data

- Function trigger: A scheduler shall run continuously after the application starts, with a delay of 1 second between 2 fetches.
- Function description: The RMS system fetches the current production parameters from the PLCs at the factory on each run.
- Screen layout: N/A.
- Function detail: On each fetch, the RMS reads all values of the PLC Tags of all production parameters in the system and then sends all those values to the user interface.
- Business rules: BR-07.

3.4. Feature Alarm

3.4.1. Continuously Check against Alarm Conditions

- Function trigger: A scheduler shall run continuously after the application starts, with a delay of 1 second between 2 checks.
- Function description: The RMS system checks the production parameters against all alarm conditions and then creates or resolves the alarms accordingly.
- Screen layout: N/A
- Function detail: The RMS reads all alarm conditions information on each check. Three possible cases will happen:
    - When a condition is met, create a new associated alarm.
    - When an alarm has just been created, distribute associated notification actions.
    - When the condition of an alarm is not met anymore, resolve that alarm.
- Business rules: BR-09, BR-11, BR-12, BR-13, BR-14, BR-15, BR-16, BR-17, BR-18, BR-19.

3.4.2. Alarm Email Notification

- Function trigger: When a newly created alarm has registered the Email action type.
- Function description: The system sends the alarm message to all registered emails in that action.
- Screen layout: N/A
- Function detail: The system sends the alarm message to all registered emails in that action; each sent can be automatically retried three times in case any error happens.
- Business rules: BR-21.

3.4.3. Alarm Popup Notification

- Function trigger: When a newly created alarm has a PopUp action type registered.
- Function description: The system shows all active users a popup of alarm information for continuously 10 seconds.
- Screen layout:

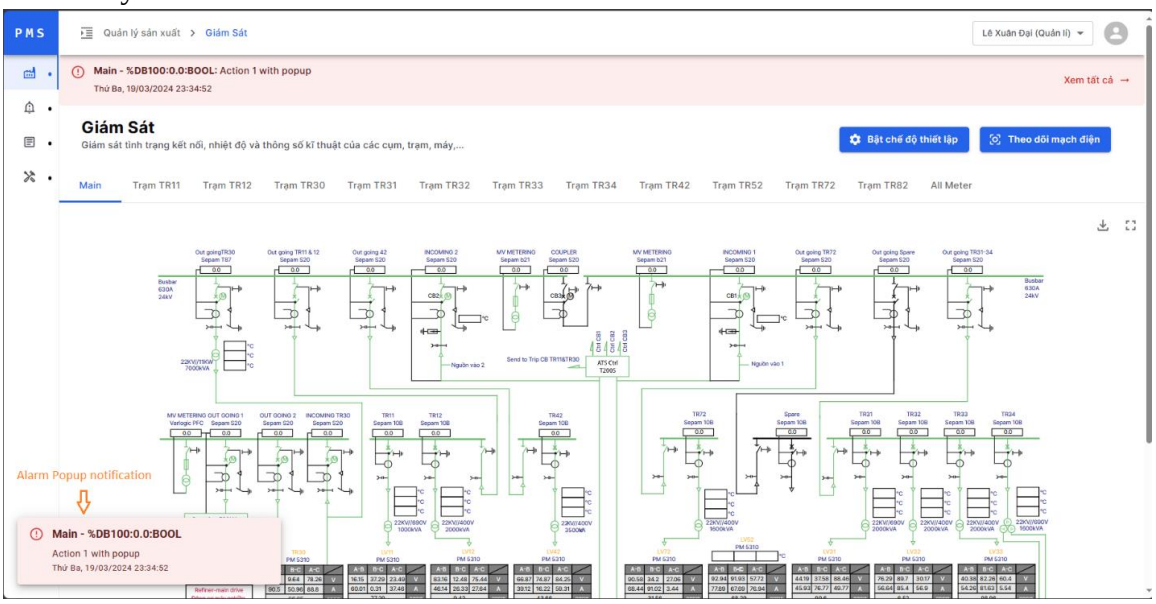


Figure 32 Alarm popup notification and carousel.

- Function detail: The system shows a popup of alarm information to all active users for 10 seconds, then moves that alarm information to the list of unresolved alarms until that alarm is resolved.
- Business rules: BR-20.

3.4.4. Alarm Push Notification

- Function trigger: When a newly created alarm has registered push notification action type.

- Function description: Users who installed the ntfy application on their phone and subscribed to the RMS notification channel will receive a push notification whenever a new alarm is triggered.
- Screen layout: N/A
- Function detail: The users must follow instructions to install the ntfy application on their phone and subscribe to the RMS notification channel. After completing all the setups, those users can receive a push notification whenever a new alarm is triggered.
- Business rules: BR-23.

3.4.5. View Alarm Histories

- Function trigger: Users click the “Lịch sử cảnh báo” item on the left sidebar of the screen.
- Function description: List all resolved alarms that happened in the past.
- Screen layout:

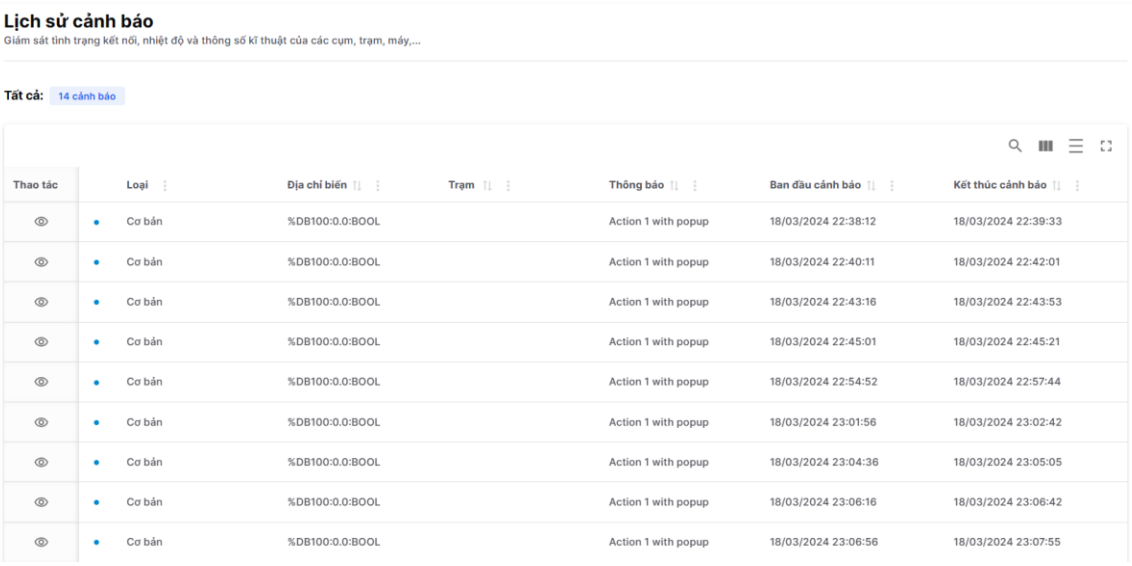


Figure 33 Alarm histories screen.

- Function detail: List all resolved alarms that happened in the past, including these fields:
  - Thao tác.
  - Loại.
  - Địa chỉ biến.
- Business rules: BR-26.

3.4.6. View Alarm Conditions

- Function trigger: Users click the “Cấu hình cảnh báo” item on the left sidebar of the screen.
- Function description: List all alarm conditions.

- Screen layout:

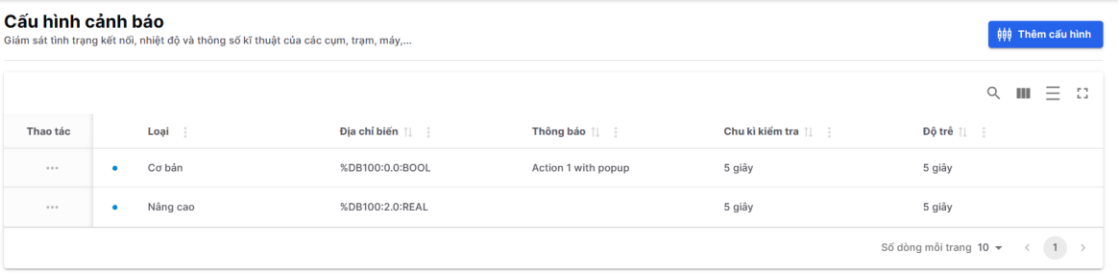


Figure 34 Alarm conditions screen.

- Function detail: List all alarm conditions in the table format, including these columns:
  - Thao tác (Edit, Create by duplicating and Delete)
  - Loại alarm (“Cơ bản” and “Nâng cao”)
  - Địa chỉ biến
  - Mức độ
  - Thông báo
  - Chu kì kiểm tra
  - Độ trễ
- Business rules: BR-09, BR-10.

3.4.7. Create Alarm Condition

- Function trigger: On the “Cấu hình cảnh báo” screen, the administrator clicks the button “Thêm cấu hình” on the top left of the screen.
- Function description: The administrator creates a new alarm condition.
- Screen layout:

- Step 1

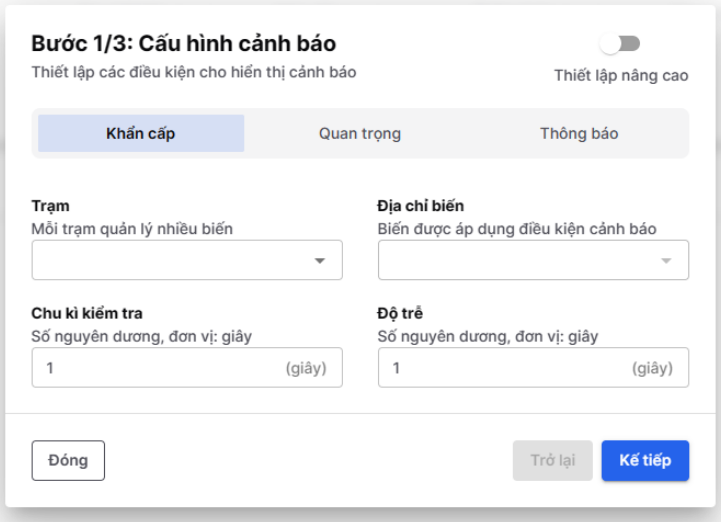


Figure 35 Step 1 of creating Alarm condition popup.

- Step 2:

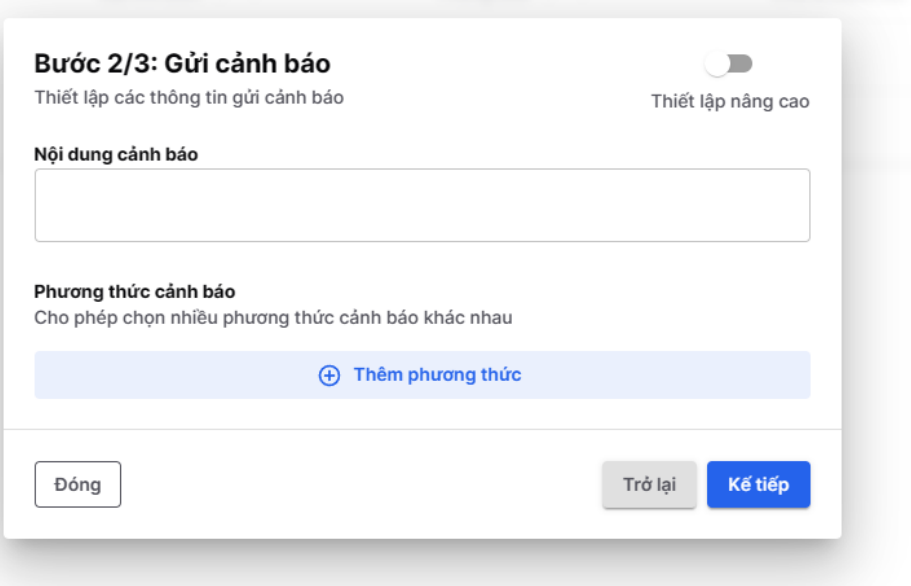


Figure 36 Step 2 of creating Alarm condition popup.

- Step 3:

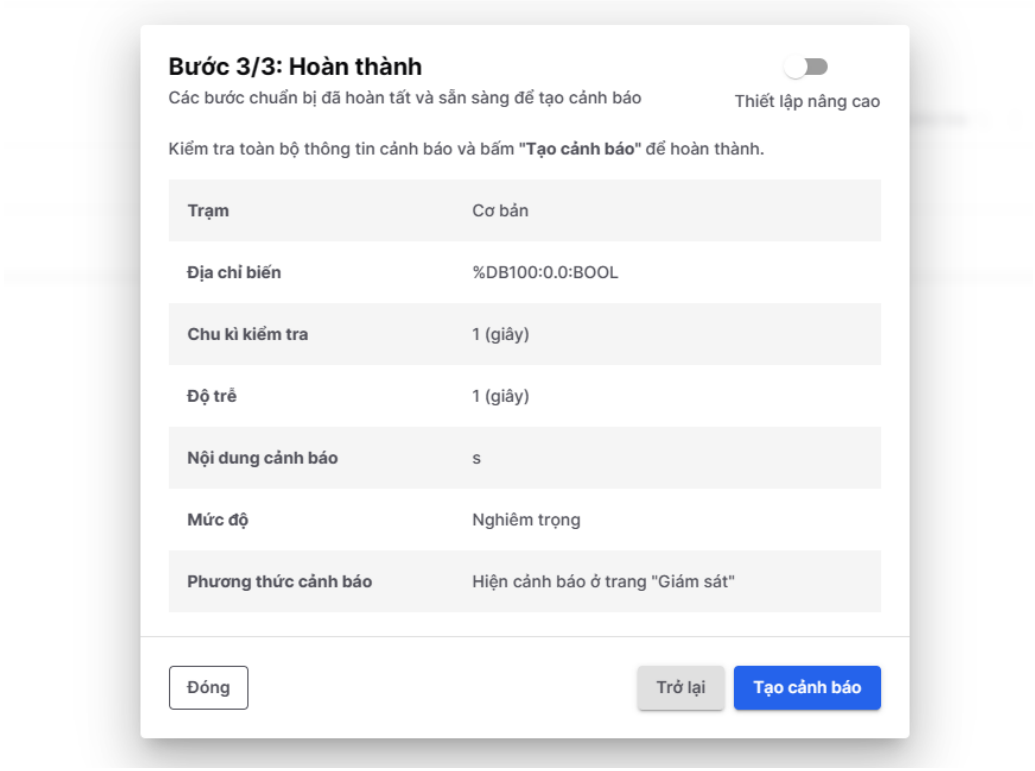


Figure 37 Step 3 of creating Alarm condition popup.

- Function detail: The administrator creates a new alarm condition of 2 different types:
  - Predefined alarm (“Thiết lập cơ bản”): The alarm and logic have already been stored in the PLC. Each alarm is stored in a PLC Tag with the bool data type, representing whether the condition is met. When creating this type of alarm, the administrator is required to enter:
    - PLC Tag of the alarm condition – “Địa chỉ biến.”
    - Alarm message – “Nội dung cảnh báo”.
    - Check interval – “Chu kì kiểm tra”.
    - Time delay – “Độ trễ”.
    - Severity – “Mức độ”.
  - Custom alarm (“Thiết lập nâng cao”): an alarm not stored in the PLC before. When creating this type of alarm, the administrator is required to enter the following:
    - PLC Tag of the production parameter to check against the condition – “Địa chỉ biến.”
    - Alarm message – “Nội dung cảnh báo.”
    - Check interval – “Chu kì kiểm tra”
    - Time delay – “Độ trễ”
    - Condition check type:
      - In range – “Trong khoảng”

                - Greater than or equal – “Lớn hơn hoặc bằng”
                - Smaller than or equal – “Nhỏ hơn hoặc bằng”
            - Min – “Giới hạn dưới”
            - Max – “Giới hạn trên”
- Data validation:
    - “Trạm”:
        - Type: select
        - Available values: all stations from the Monitoring screen
    - “Địa chỉ biến”:
        - Type: select
        - Available values: all available PLC Tags from the selection Station (“Trạm”)
    - “Chu kì kiểm tra”:
        - Type: integer
        - Min: 1
        - Max: 59
    - “Độ trễ”:
        - Type: integer
        - Min: 1
        - Max: 59
    - “Nội dung cảnh báo”:
        - Type: text field
        - Min length: 1
        - Max length: 255
    - “Phương thức cảnh báo”:
        - Type: multi-select.
        - Must select at least 1 value.
        - Available values:
            - “Hiện cảnh báo ở trang Giám sát”
            - “Thông báo qua email”
            - “Cảnh báo qua thiết bị di động”
- Business rules: BR-09, BR-10, BR-19, BR-22, BR-27, BR-28, BR-29, BR-30, BR-31, BR-34.

3.4.8. Update Alarm Condition

- Function trigger: On the “Cấu hình cảnh báo,” the administrator clicks the associated triple dots button (“Thao tác” column), then click “Chỉnh sửa cảnh báo.”
- Function description: The administrator updates information on an alarm condition
- Screen layout:

- Step 1:

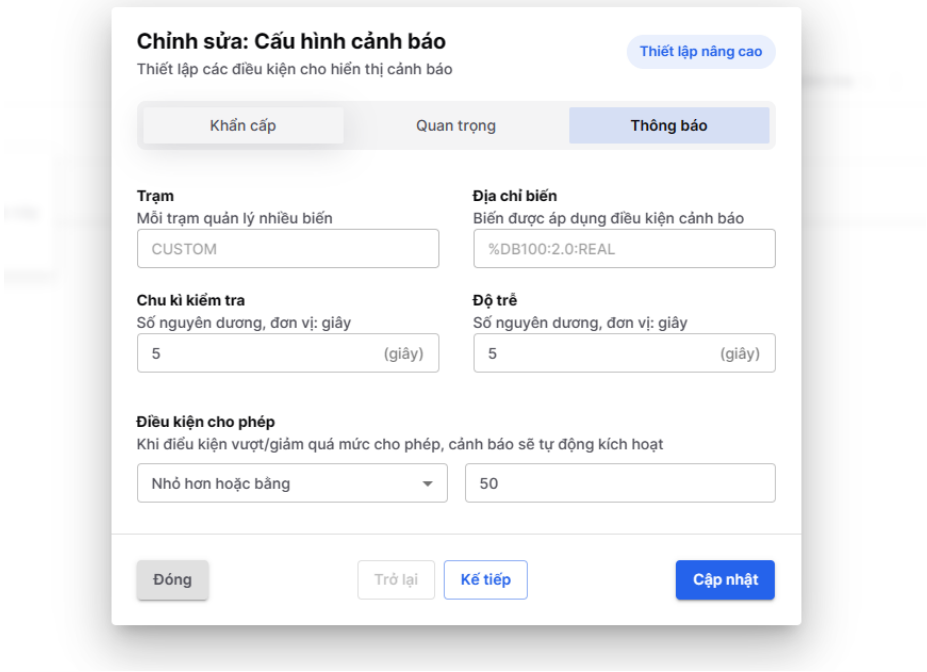


Figure 38 Step 1 of updating Alarm condition popup.

- Step 2:

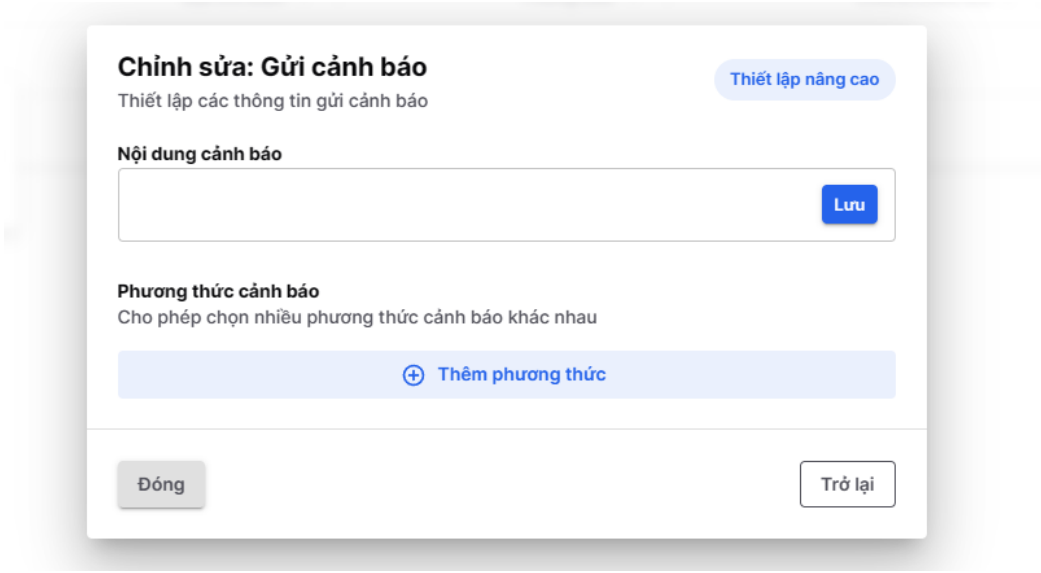


Figure 39 Step 2 of updating Alarm condition popup.

- Function detail: The administrator is allowed to update this information of any existing alarm condition:

    - Check interval – “Chu kì kiểm tra”
    - Time delay – “Độ trễ”
    - Alarm message – “Nội dung cảnh báo.”
    - Min
    - Max
- Business rules: BR-09, BR-10, BR-19, BR-22, BR-27, BR-28, BR-29, BR-30, BR-31, BR-32, BR-33, BR-34
- Data validation:
    - “Chu kì kiểm tra”:
        - Type: integer
        - Min: 1
        - Max: 59
    - “Độ trễ”:
        - Type: integer
        - Min: 1
        - Max: 59
    - “Nội dung cảnh báo”:
        - Type: text field
        - Min length: 1
        - Max length: 255
    - “Phương thức cảnh báo”:
        - Type: multi-select.
        - Must select at least 1 value.
        - Available values:
            - “Hiện cảnh báo ở trang Giám sát”
            - “Thông báo qua email”
            - “Cảnh báo qua thiết bị di động”

3.4.9. Delete Alarm Condition

- Function trigger: On the “Cấu hình cảnh báo” list screen, the administrator clicks the associated triple dots button (“Thao tác” column), then clicks “Xóa cảnh báo.”
- Function description: The administrator deletes an existing alarm condition.
- Screen layout - Confirmation popup:

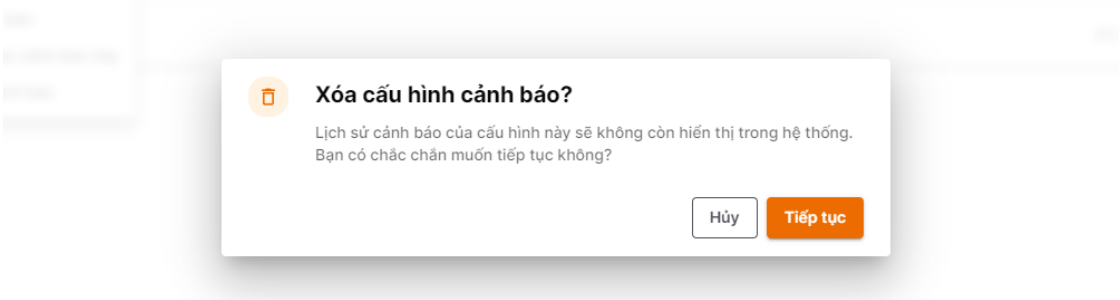


Figure 40 Delete Alarm condition confirmation popup.

- Function detail: The administrator can delete any alarm condition that does not have an unresolved alarm attached.
- Business Rules: BR-10, BR-25, BR-39.

3.4.10. Create Alarm Action

- Function trigger: On the second step of the Update Alarm Condition ("Chỉnh sửa cảnh báo") popup, the administrator can create a new alarm action by clicking the "Thêm phương thức" button.
- Function description: The administrator can add a new action to an alarm condition to add notification ability to the alarm condition.
- Screen layout:

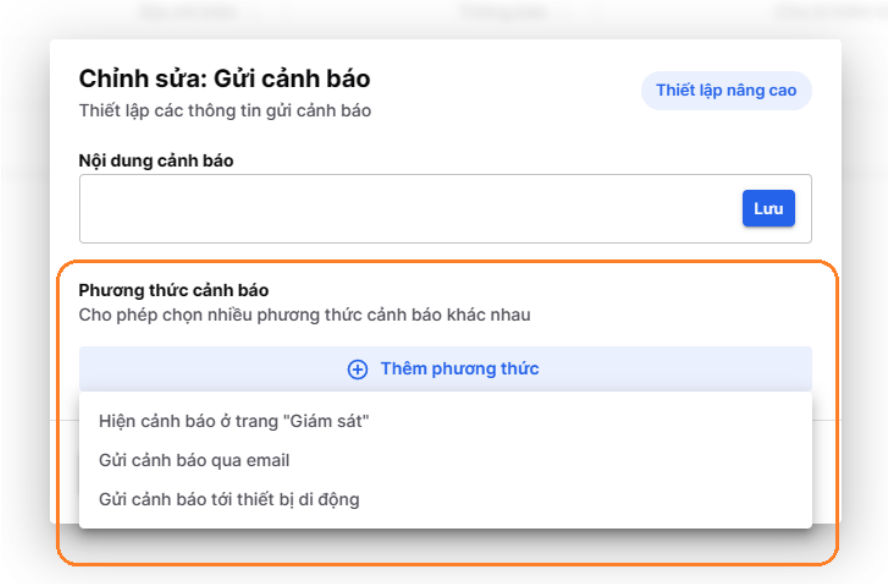


Figure 41 Create Alarm Action option when on the Creating Alarm Condition popup.

- Function detail: The administrator can add between 2 types of action, including "Hiện ở trang giám sát" or "Gửi cảnh báo qua email". For the "Gửi cảnh báo qua email," the administrator can select between the available emails of the application's users.
- Business rules: BR-10, BR-19, BR-34, BR-22.
- Data validation:
    - "Phương thức cảnh báo":
        - Type: multi-select
        - Must select at least 1 value.
        - Available values:
            - "Hiện cảnh báo ở trang Giám sát"

- “Thông báo qua email”
- “Cảnh báo qua thiết bị di động”

3.4.11. Update Alarm Action

- Function trigger: On the second step of the Update Alarm Condition (“Chỉnh sửa cảnh báo”) popup
- Function description: The administrator can modify an existing action of an alarm condition.
- Screen layout:

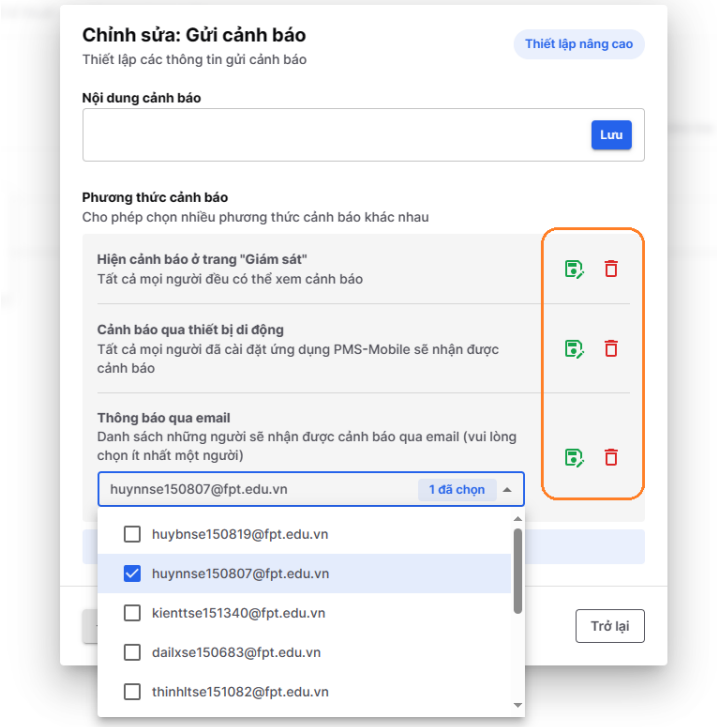


Figure 42 Update Alarm Action option when on the Creating Alarm Condition popup.

- Function detail: The administrator can update the alarm message once for all alarm actions or update the list of emails of action of type “Gửi cảnh báo qua Email.”
- Business rules: BR-10, BR-19, BR-34, BR-22.

3.4.12. Delete Alarm Action

- Function trigger: On the second step of the Update Alarm Condition (“Chỉnh sửa cảnh báo”) popup, the administrator clicks the Trash icon button.
- Function description: The administrator can delete an existing action of the current alarm condition.
- Screen layout: N/A

- Function detail: N/A
- Business rules: BR-34

3.5. Feature Report

3.5.1. Search Historical Report

- Function trigger: Users click the “Lịch sử báo cáo” item on the left sidebar of the screen.
- Function description: List all available reports recorded in the past.
- Screen layout:

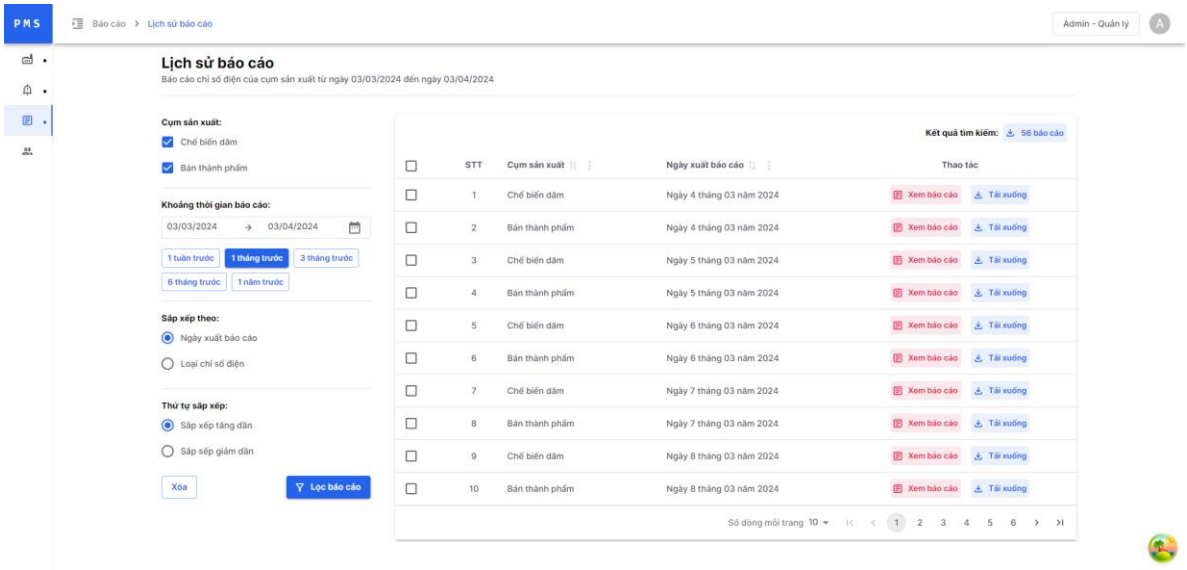


Figure 43 Historical reports screen.

- Function detail: List all available reports recorded in the past based on filtering and sorting with pagination.
    - Filters include report types and date ranges of recording dates.
      Date ranges contain two types: absolute and relative date ranges.
    - Sorts include report type and recording date.
- Data validation:
    - “Khoảng thời gian báo cáo”:
        - Start date must be before the end date.

3.5.2. Export Historical Report

- Function trigger: On the Historical Report List screen, users click the “Tải xuống” for an item in the historical list.
- Function description: Download reports as Excel files.
- Screen layout:

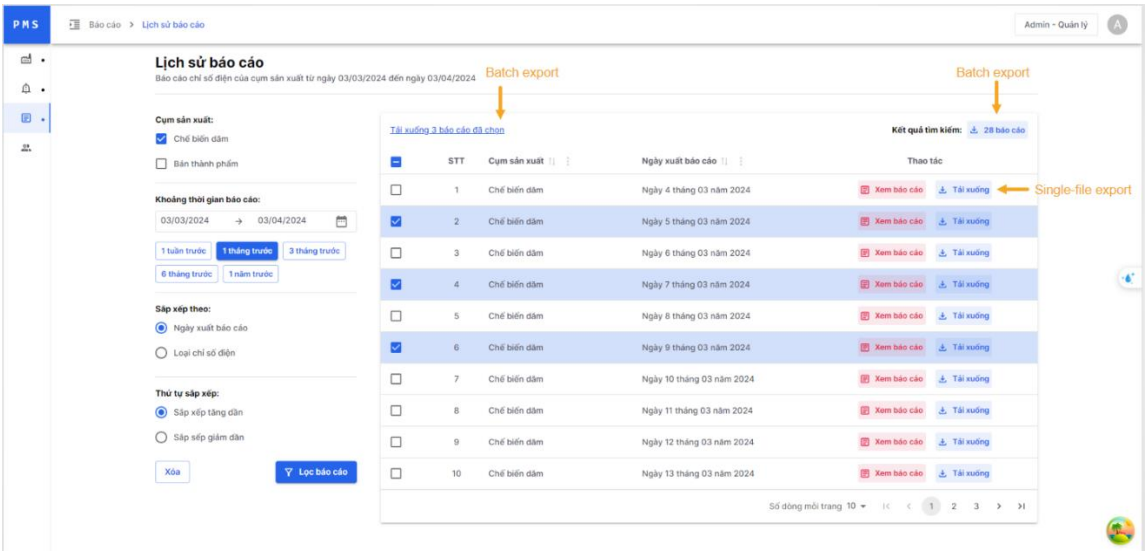


Figure 44 Export historical export option on the Historical reports screen.

- Function detail: Download reports as Excel files.
    - There are two export types: batch export for multiple reports and single-file export for specific reports.
    - Excel files are automatically generated by the Capture Data of the Previous Day feature.
- Business rules: BR-35, BR-36, BR-37, BR-38

3.5.3. View Historical Report

- Function trigger: On the Historical Report List screen, users click the “Xem báo cáo” button for an item in the historical list.
- Function description: View details of a report recorded in the past.
- Screen layout:
    - “Cả ngày” tab

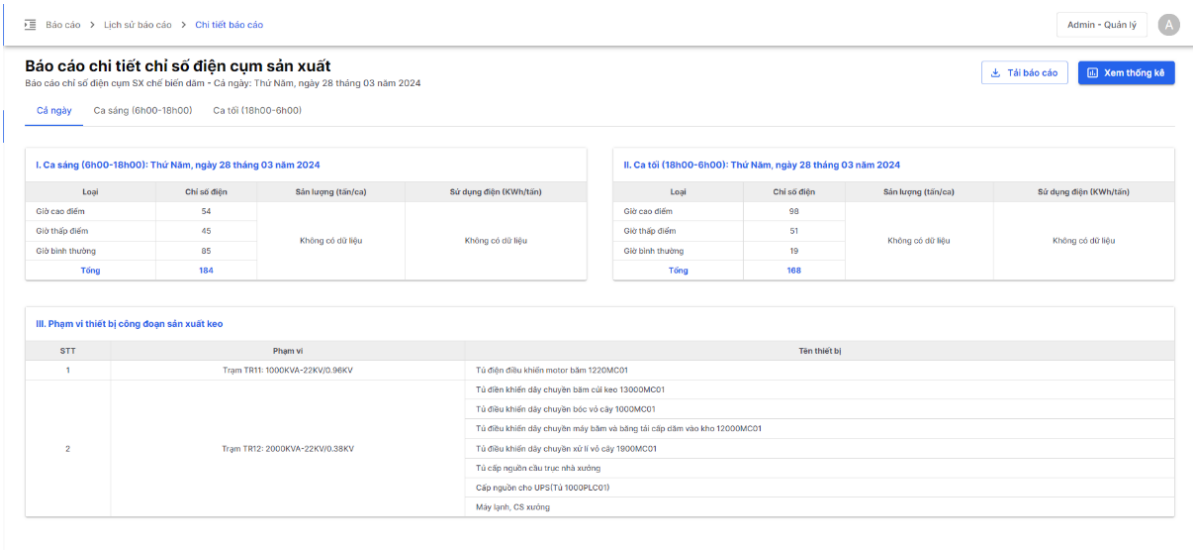


Figure 45 View 1 historical report screen.

- "Ca sáng" tab:

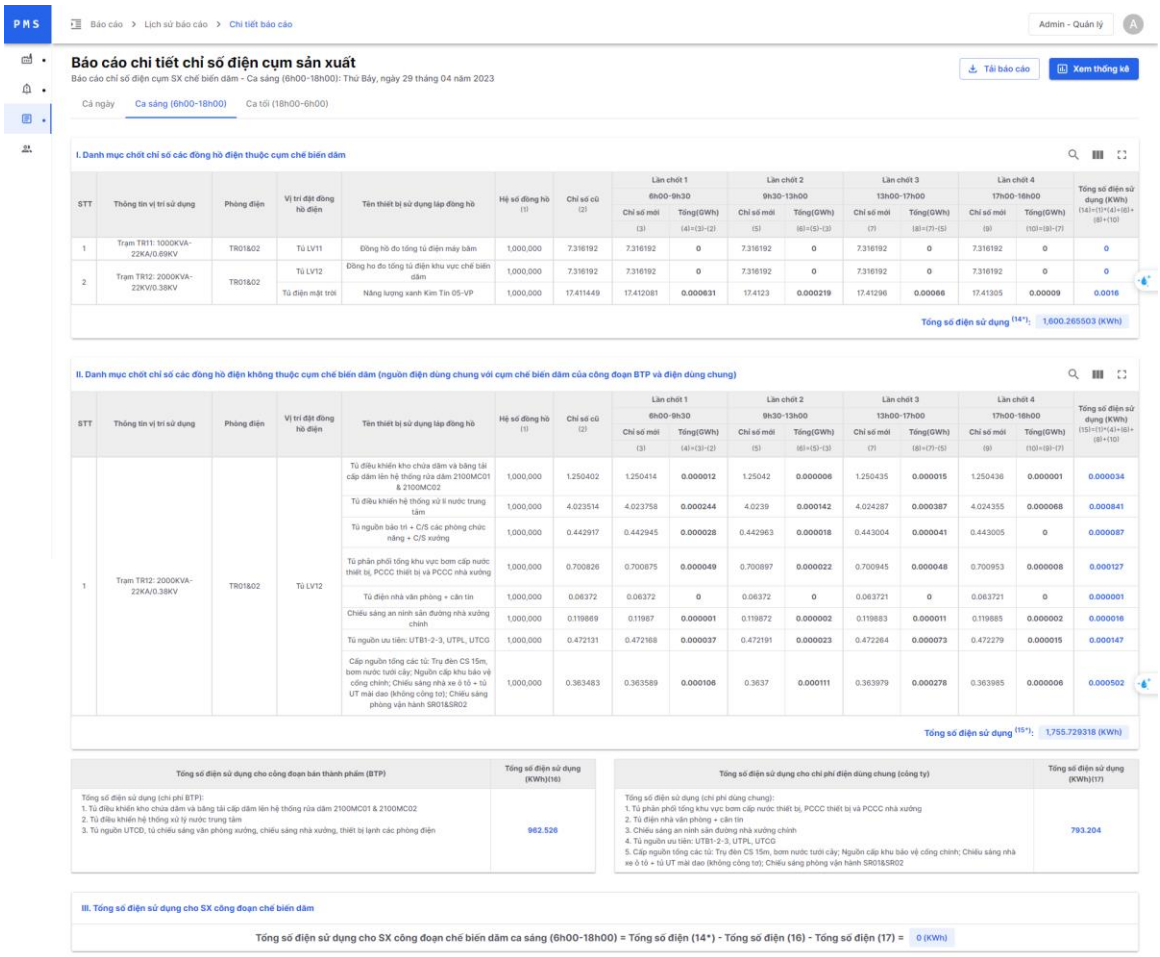


Figure 46 "Ca sáng" tab of the historical report screen.

- Ca tối" tab:

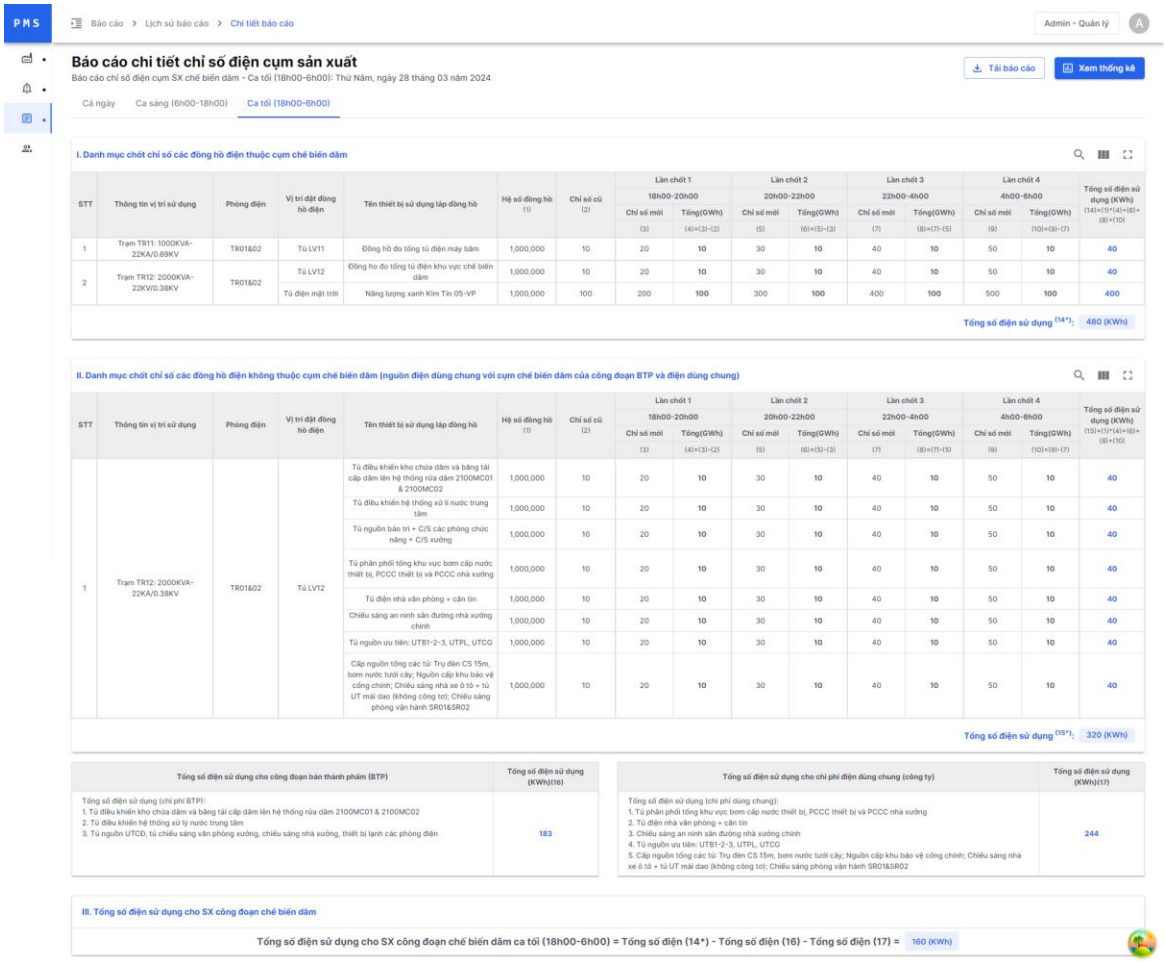


Figure 47 "Ca tối" tab of the historical report screen.

- Function detail: View details of a report recorded in the past. The report contains 3 sections, namely "Cả ngày", "Ca sáng" and "Ca tối". Each section includes several categories, and a category contains many data rows.

3.5.4. View Statistical Charts of a specific date

- Function trigger: On the top View Historical Report Details screen, users click the "Xem thống kê" button.
- Function description: View charts whose data belong to historical report to gain insights about total electricity consumption.
- Screen layout:

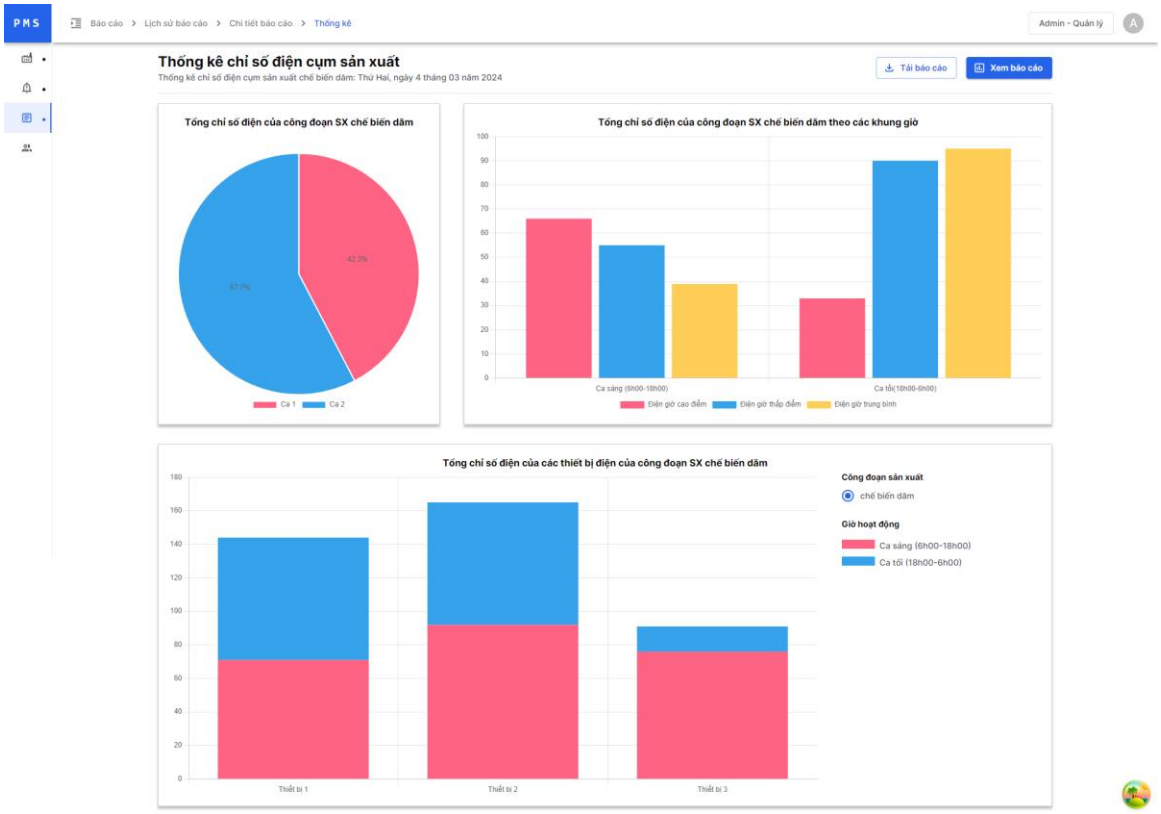


Figure 48 Charts tab of the Historical report screen.

- Function detail: Each historical report will have a corresponding historical report statistic.

3.5.5. View Statistical Charts of a specific date range

- Function trigger: Users click the “Báo cáo thống kê” item on the left sidebar of the screen.
- Function description: View charts whose data belong to historical reports to gain insights about total electricity consumption.
- Screen layout:

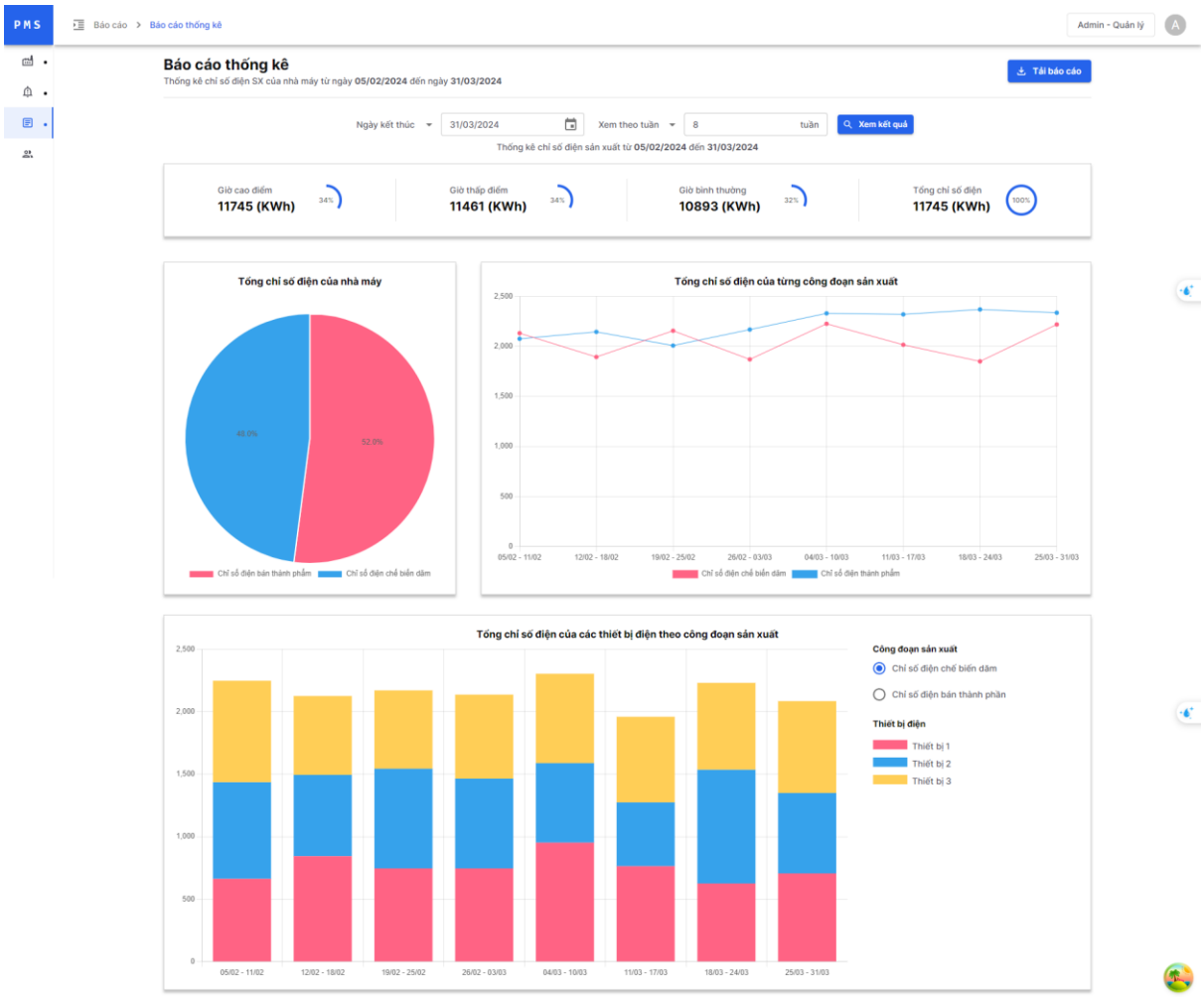


Figure 49 View time range Reports and Statistic screen.

- Function detail: Analyze statistics from all available historical reports based on filtering. Filters include recording date range, step, and number of steps.
- Data validation:
    - "Xem theo..." – type: integer
        - "Xem theo ngày":
            - Min: 2
            - Max: 12
        - "Xem theo tuần":
            - Min: 2
            - Max: 10
        - "Xem theo tháng":
            - Min: 2
            - Max: 12
        - "Xem theo năm":
            - Min: 2.

- Max: 5.

3.5.6. Capture Data of the Previous Day

- Function trigger: The PLC has a boolean PLC Tag representing whether the PLC had captured enough data of yesterday or not. RMS continuously checks for the value of this Tag, if the value is true, this function will be triggered.
- Function description: Read and save the previous day's data into the Database.
- Screen layout: N/A.
- Function detail: Create a scheduler to check the trigger PLC Tag daily at 6 a.m. to decide whether to capture the values of provided Tags into RMS's database.
  More specifically, at a specific time of the day (6 a.m.), RMS checks a trigger Tag to decide whether to capture figures.
  - If the value is True, RMS must capture the values of all provided Tags into the database and simultaneously generate an excel report file containing this data, saving it to the server. Subsequentially, RMS sets the trigger Tag back to false.
  - The trigger Tag is only updated to True by the PLC.
  - If the value is False, RMS will do nothing.

4. Non-functional Requirements

4.1. External Interfaces

4.1.1. User Interface

- The language for the RMS application is Vietnamese.
- Not supported for screen size of below 1200px.

4.1.2. Hardware Interface

- The RMS application interfaces with **PLC Siemens S7-1200** for real-time data acquisition and monitoring.

4.1.3. Communication Interface

- The communication protocol used for interaction between the RMS application and the PLC is the **S7 Protocol**.

4.2. Quality Attributes

4.2.1. Usability

- User interface components must be labeled appropriately and be friendly to Vietnamese users.
- Navigation flows are streamlined to minimize user effort and facilitate efficient task completion.

4.2.2. Security

- User username/password pair for authentication to control access to the RMS application.
- Logging and monitoring mechanisms are implemented to track user activities.

5. Requirement Appendix

5.1. Common Requirements

CR1: All required fields in the forms need to be filled out.

5.1. Business Rules

ID	Rule Definition
BR-01	"Tên đăng nhập" must be unique.
BR-02	"Địa chỉ email" must be unique.
BR-03	"Tên đăng nhập" is not allowed to be modified after a user is created.
BR-04	"Vai trò" is not allowed to be modified after a user is created.
BR-05	There must be always contains at least 1 user with Administrator role in the system.
BR-06	Only Administrator is allowed to manage User (view, create, update, and delete).
BR-07	Combination of data block, offset, and data type for a monitoring cell must be a valid PLC Tag in the connected PLC.
BR-08	Only Administrator is allowed to Modify PLC Tag on Monitoring screen
BR-09	There are 2 available types for an Alarm Condition: "Cảnh báo cơ bản" and "Cảnh báo nâng cao".
BR-10	Only Administrator is allowed to manage Alarm Condition (view, create, update, and delete).
BR-11	After any type of Alarm Condition is triggered and there's no Alarm History created for this Alarm Condition, a newly Alarm History with status Triggered will be created for this Alarm Condition.

BR-12	There are 3 available statuses for an Alarm History: Triggered, Sent, and Solved.
BR-13	A Predefined Alarm Condition ("Cảnh báo cơ bản") is triggered when the value of the associated PLC Tag is true.
BR-14	A Custom Alarm Condition ("Cảnh báo nâng cao") with min ("Giới hạn dưới") set only is triggered when the value of the associated PLC Tag goes greater than that min value.
BR-15	A Custom Alarm Condition ("Cảnh báo nâng cao") with max ("Giới hạn trên") set only is triggered when the value of the associated PLC Tag goes smaller than that max value.
BR-16	A Custom Alarm Condition ("Cảnh báo nâng cao") with both max ("Giới hạn trên") and min ("Giới hạn dưới") set is triggered when the value of the associated PLC Tag goes smaller than the min value or greater than the max value.
BR-17	An Alarm History of a Predefined Alarm Condition changes the status from Triggered or Sent to Solved when the value of the associated PLC Tag in the Condition switch from true to false.
BR-18	An Alarm History of a Custom Alarm Condition changes the status from Triggered or Sent to Solved when the value of the associated PLC Tag in the Condition is back in the allowed min/max range.
BR-19	There are 3 available types for an Alarm Action: "Hiện cảnh báo ở trang Giám sát", "Thông báo qua email" and "Cảnh báo qua thiết bị di động" .
BR-20	If a triggered Alarm Condition has a "Hiện cảnh báo ở trang Giám sát" Alarm Action, display that Alarm Condition's message via a toast on the UI .
BR-21	If a triggered Alarm Condition has a "Thông báo qua email" Alarm Action, send that Alarm Condition's message to the list of emails listed in the Alarm Action.
BR-22	If an Alarm Condition has a "Thông báo qua email" Alarm Action, that Action is required to have at least 1 email as the recipients.
BR-23	If a triggered Alarm Condition has a "Cảnh báo qua thiết bị di động" Alarm Action, send that Alarm Condition's message to the ntfy endpoint for ntfy to notify all registered users.
BR-24	There must be always a section on the "Giám sát" screen to display all unresolved Alarm Conditions.
BR-25	When an Alarm Condition is deleted, all its associated Alarm Histories must be deleted also.
BR-26	Only Alarm Histories with status Solved are displayed on the Alarm History page.
BR-27	One PLC Tag is allowed to be specified in one Alarm Condition only.
BR-28	A Custom Alarm Condition's associated PLC Tag ("Địa chỉ biến") can be selected from any Monitoring stations or "Nâng cao" station.
BR-29	A Predefined Alarm Condition's PLC Tag ("Địa chỉ biến") can be selected from the "Cơ bản" station only.

BR-30	Newly created Sensor Configuration when creating a Predefined Alarm Condition must belong to the "Cơ bản" station.
BR-31	Newly created Sensor Configuration when creating a Custom Alarm Condition must belong to the "Nâng cao" station.
BR-32	"Địa chỉ biến" is not allowed to be modified after an Alarm Condition is created.
BR-33	Alarm type ("Cơ bản" and "Nâng cao") is not allowed to be modified after an Alarm Condition is created.
BR-34	An Alarm Condition requires to have at least 1 Alarm Action.
BR-35	Only Administrator is allowed to export Excel reports.
BR-36	There are 2 available types of Report, corresponded with 2 stages of the factory: “Chế biến dăm” and “Chế biến bán thành phẩm”.
BR-37	Exported “Chế biến dăm” Excel report must be in the format of the given template file from the factory: 1.BM theo doi chi so dien che bien dam.xlsx.
BR-38	Exported “Chế biến bán thành phẩm” Excel report must be in the format of the given template file from the factory: 2.BM theo doi chi so dien BTP.xlsx.
BR-39	Alarm Condition that has an unresolved alarm attached is not allowed to be deleted.

Table 26 Business rules

5.2. Application Messages List

Message Code	Message Type	Context	Content
MSG-01	Toast - Info	Monitoring - Edit PLC Tag	Đã bật chế độ thiết lập! Di chuyển chuột lên các số để tiến hành thiết lập.
MSG-02	Error message under input field	Monitoring - Edit PLC Tag - Empty data block	Data Block không được trống.
MSG-03	Error message under input field	Monitoring - Edit PLC Tag - Empty offset	Offset không được trống
MSG-04	Toast - Success	Monitoring - Edit PLC Tag	Cập nhật địa chỉ thành công! Vui lòng tải lại trang để thấy dữ liệu mới nhất

MSG-05	Error message under input field	Alarm - Add Alarm - checkInterval greater than 59	Chu kì kiểm tra không được lớn hơn 59
		Alarm - Update Alarm - checkInterval greater than 59	
MSG-06	Error message under input field	Alarm - Add Alarm - checkInterval smaller than 0	Chu kì kiểm tra phải là số nguyên dương
		Alarm - Update Alarm - checkInterval smaller than 0	
MSG-07	Error message under input field	Alarm - Add Alarm - timeDelay greater than 59	Độ trễ không được lớn hơn 59
		Alarm - Update Alarm - timeDelay greater than 59	
MSG-08	Error message under input field	Alarm - Add Alarm - timeDelay smaller than 0	Độ trễ phải là số nguyên dương
		Alarm - Update Alarm - timeDelay smaller than 0	
MSG-09	Error message under input field	Alarm - Add Alarm - No action selected	Ít nhất phải chọn một phương thức cảnh báo
		Alarm - Update Alarm - No action selected	
MSG-10	Error message under input field	Alarm - Add Alarm - No email selected for email action	Vui lòng chọn ít nhất 1 người nhận qua email.
		Alarm - Update Alarm - No email selected for email action	
MSG-11	Toast - Success	User - Delete user successfully	Xóa người dùng thành công
MSG-12	Error message	User - Add/update user - Invalid name	Tên tối thiểu có 2 kí tự

	under input field		Tên tối đa có 50 kí tự
MSG-13	Error message under input field	User - Add/update user - Invalid email	Email không được để trống
			Email không hợp lệ
MSG-14	Error message under input field	User - Add/update user - Invalid username	Tên đăng nhập tối thiểu có 6 kí tự
			Tên đăng nhập tối đa có 16 kí tự
MSG-15	Error message under input field	User - Add/update user - Invalid password	Mật khẩu không trùng khớp
			Mật khẩu không hợp lệ

Table 27 Application Messages list

IV Software Design Description

1. System Design

1.1. System Architecture

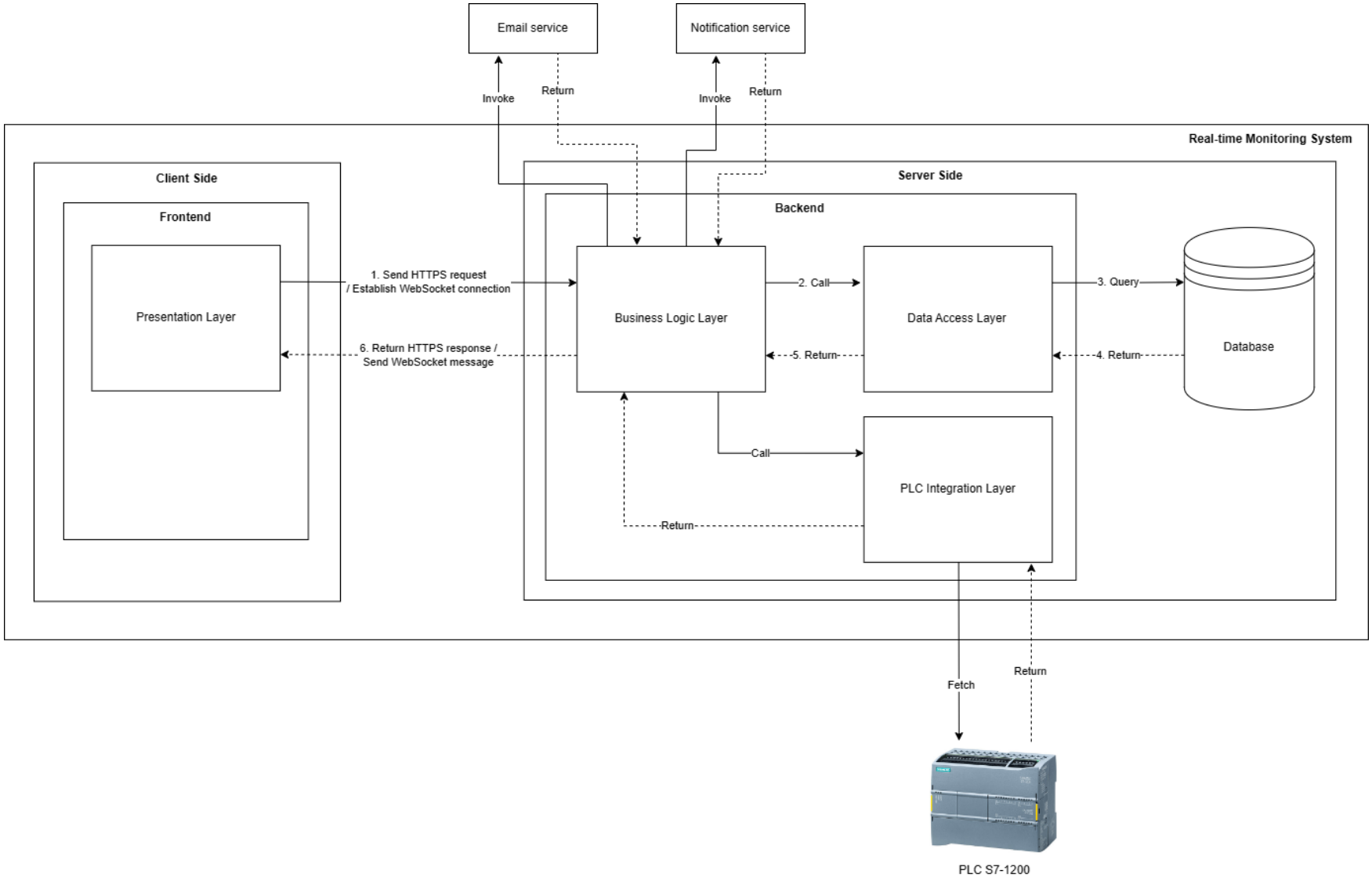


Figure 50 System architecture.

1.1.1. Frontend Architecture

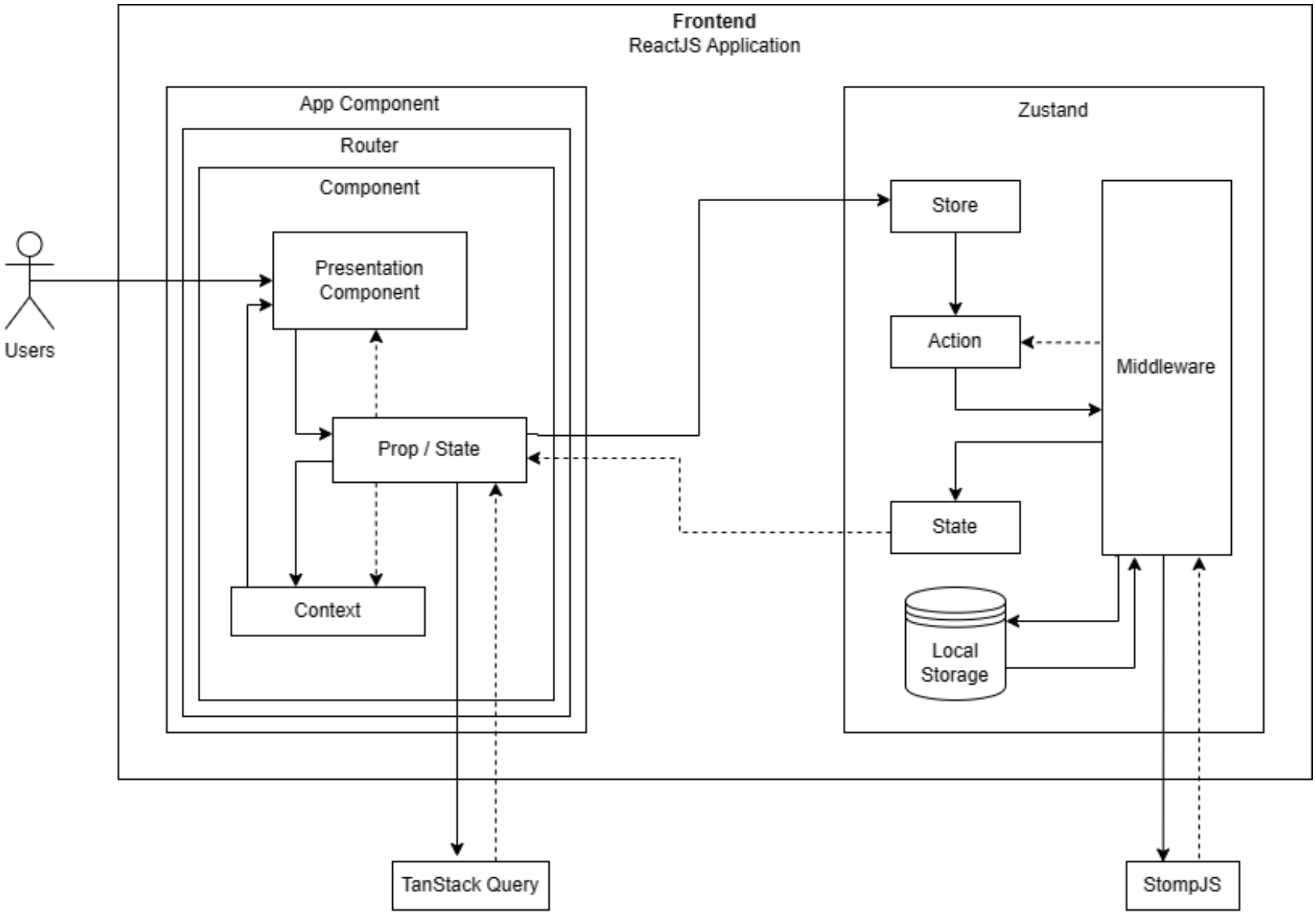


Figure 51 Frontend Architecture

The information on the RMS's Frontend Architecture is given as follows:

- **Presentation**: start with the app component that contains all other ReactJS parts, namely Router, Presentation Components, Prop / State, and Context. These are essential parts of the ReactJS application that allow it to operate seamlessly.
    - **Router**: router config helps the application to navigate between routes flexibly.
    - **Presentation Components**: Typescript functions that encapsulate UI declaration and render based on the Router and the Props / States.
    - **Prop / State**: local variables whose changes make the Presentation Components re-render the UI to reflect the changes.
    - **Context:** contains states that are rarely accessed in multiple components.
- **State Management**: use Zustand as a library for managing global states that need access.
- **APIs Invocation:** use TanStack Query library to manage REST API calls and StompJS to handle WebSocket connections and messages.

1.1.2. Backend Architecture

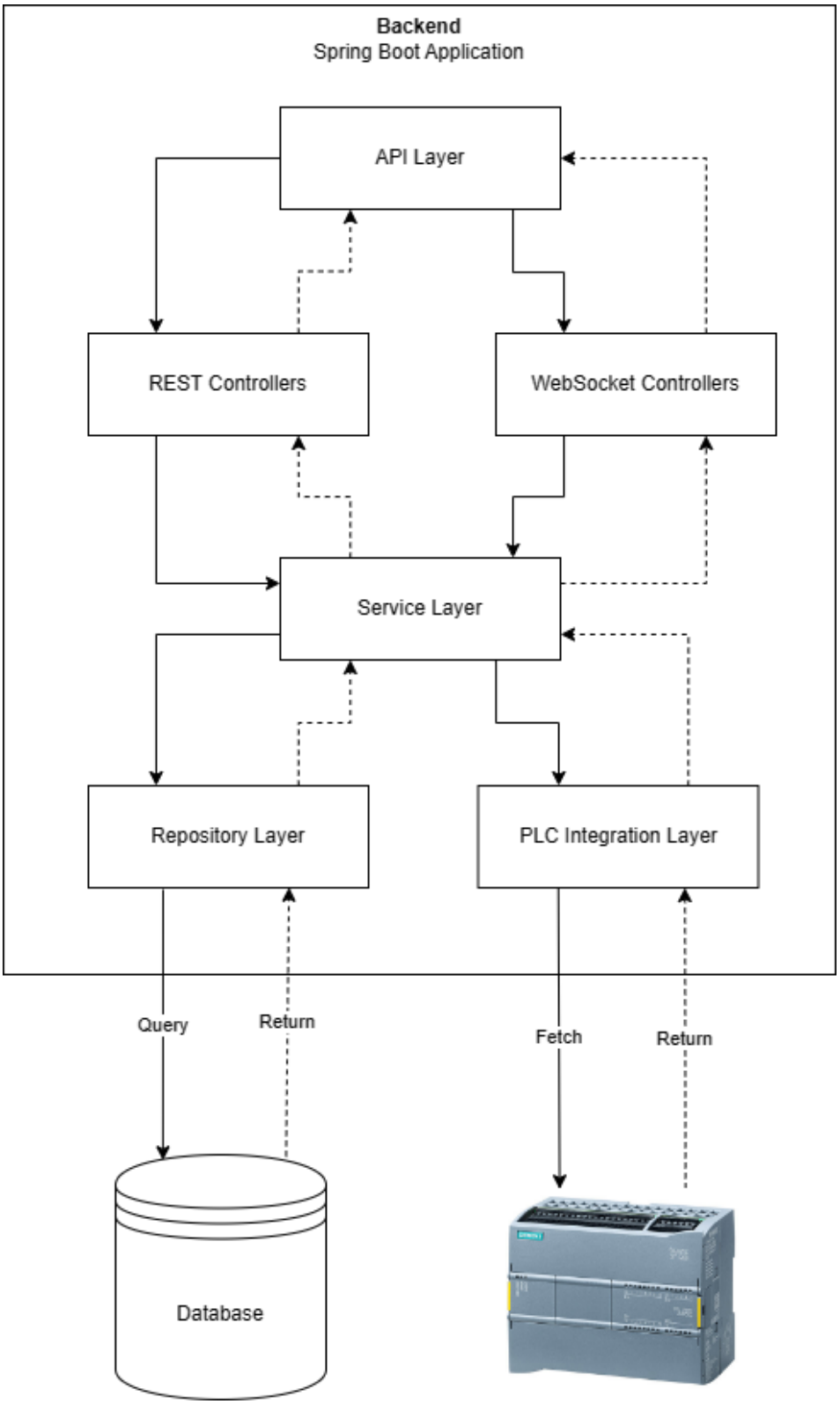


Figure 52 Backend Architecture

The information on the RMS's Backend Architecture is given as follows:

- **API layer:** Spring Boot supports creating web APIs using controllers or minimal APIs. There are 2 types of APIs, namely REST and WebSocket APIs, for each of which there are Controller classes handling requests of the corresponding API type.
    - **REST Controllers:** classes annotated with @RestController have methods for handling REST APIs.
    - **WebSocket Controllers:** classes annotated with @Controller have methods for handling WebSocket API connections.
- **Service Layer**: classes annotated with @Service that allow controllers to access business logic and resources without interacting with those resources directly by-passing messages through a separate interface. It works separately from how to connect data to the database and retrieve data but also focuses on dealing with logic to solve problems with available data.
- **Repository layer:** classes generated by Spring Data JPA from predefined interfaces take responsibility for connecting data logic to a database. These classes encapsulate the logic to access data sources as they centralize common data access functionality, providing better maintainability and decoupling the infrastructure or technology used to access databases from the domain model layer.
- **PLC Integration Layer:** classes extended and modified based on the PLC4X library that allows RMS to communicate with the PLC devices through TCP/IP connections.

1.1.3. DevOps Architecture

DevOps supports 2 main processes: Implement & Build flow and Deploy flow.

1.1.3.a) Implement & Build

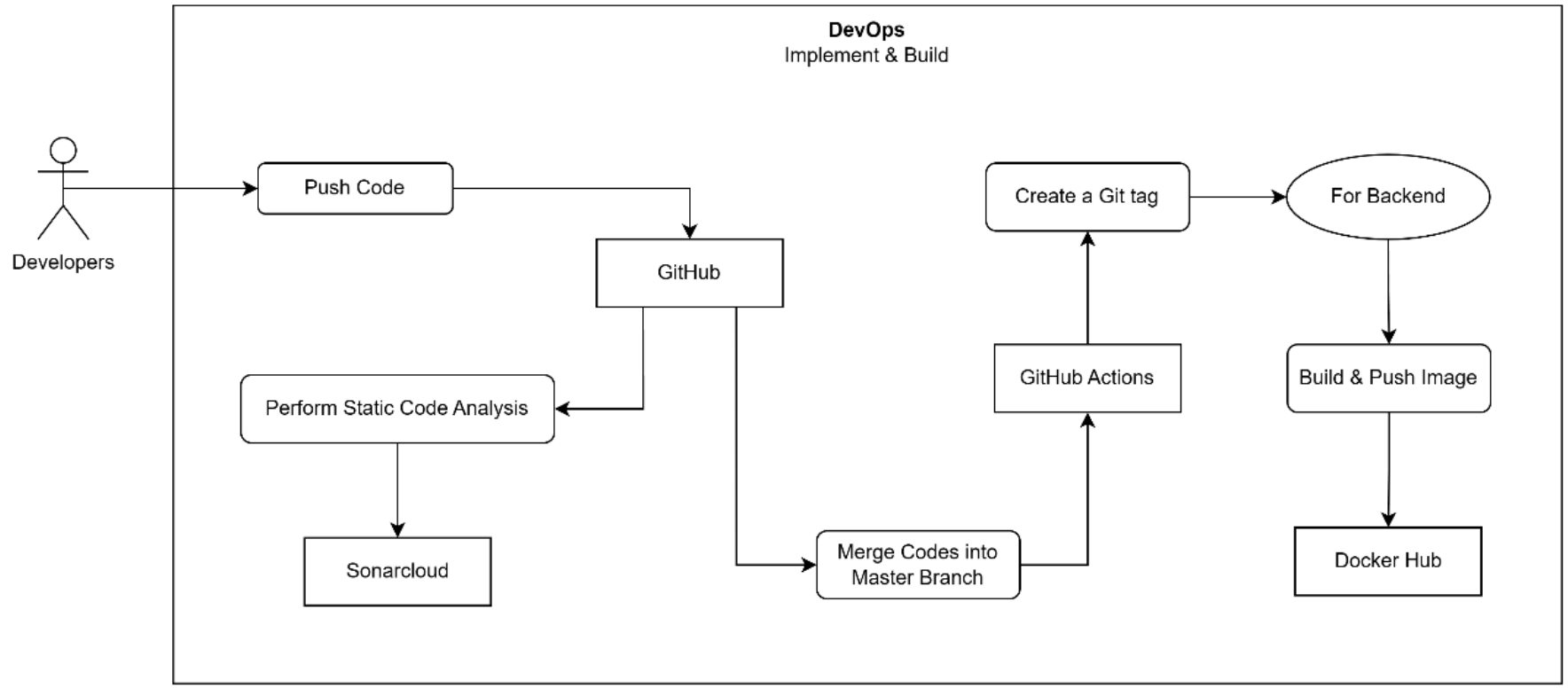


Figure 53 Implement and build flows.

The information on the DevOps's Implement & Build flow is given as follows:

- **GitHub**: a version control system used for managing source codes of the Frontend and Backend.
- **Sonarcloud**: a platform specialized for static code analysis, including detecting vulnerabilities and code smells.
- **GitHub Actions**: a flagship feature of GitHub allows software workflows to be automated through pipelines so that repetitive tasks of code integration can be reduced, like performing static code analysis, assigning new versions, and building and pushing Docker images to the hub.
- **Docker Hub:** a platform for storing and distributing Docker images.

1.1.3.b) Deploy

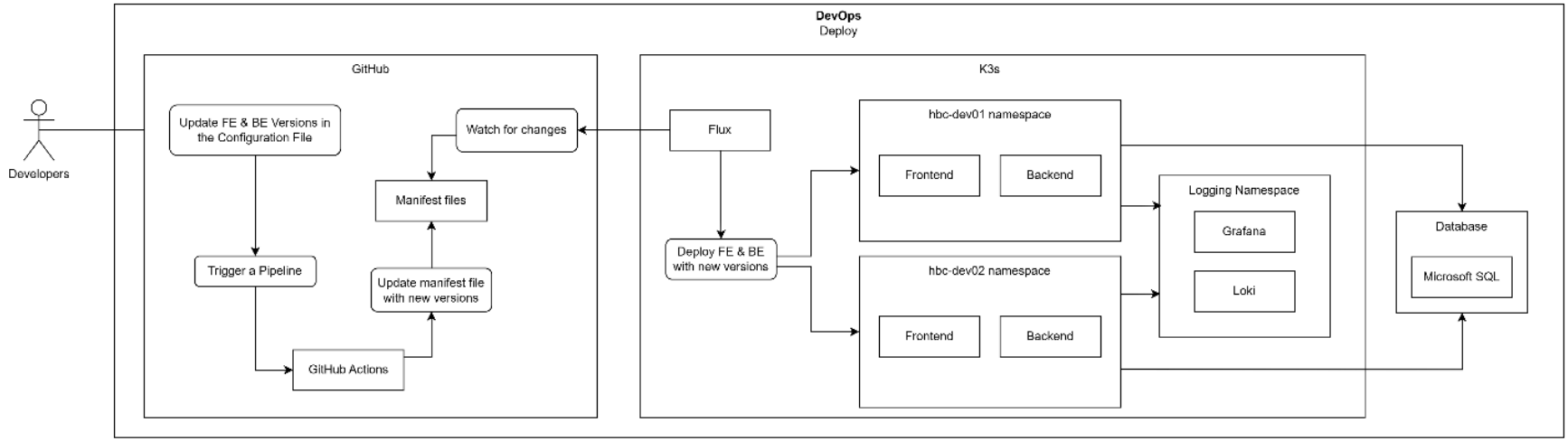


Figure 54 Deployment flow.

The information on the DevOps's Deploy flow is given as follows:

- **GitHub**: store and manage codes reflecting the infrastructure.
- **GitHub Actions**: pipelines for deploying new versions of the Frontend and Backend applications.
- **Manifest files:** files as sources of infrastructure configurations for centralizing and managing.
- **K3s**: a simplified version of Kubernetes, a container orchestration tool for managing containerized applications regarding deployment, scaling, etc.
- **Flux**: a tool for keeping Kubernetes clusters in sync with sources of configuration (like Git repositories), and automating updates to configuration when there is new code to deploy.
- **Nginx:** A web server and a reverse proxy for routing requests to Frontend and Backend.
- **VPS**: A server functions as a virtualized instance within our infrastructure, hosted and managed by a third-party service.
- **Grafana & Loki**: an open-source platform and library for centralized observability, including loggings and dashboards.
- **Microsoft SQL**: database for storing application essential data.

1.2. Package Diagram

1.2.1. Frontend Package Diagram

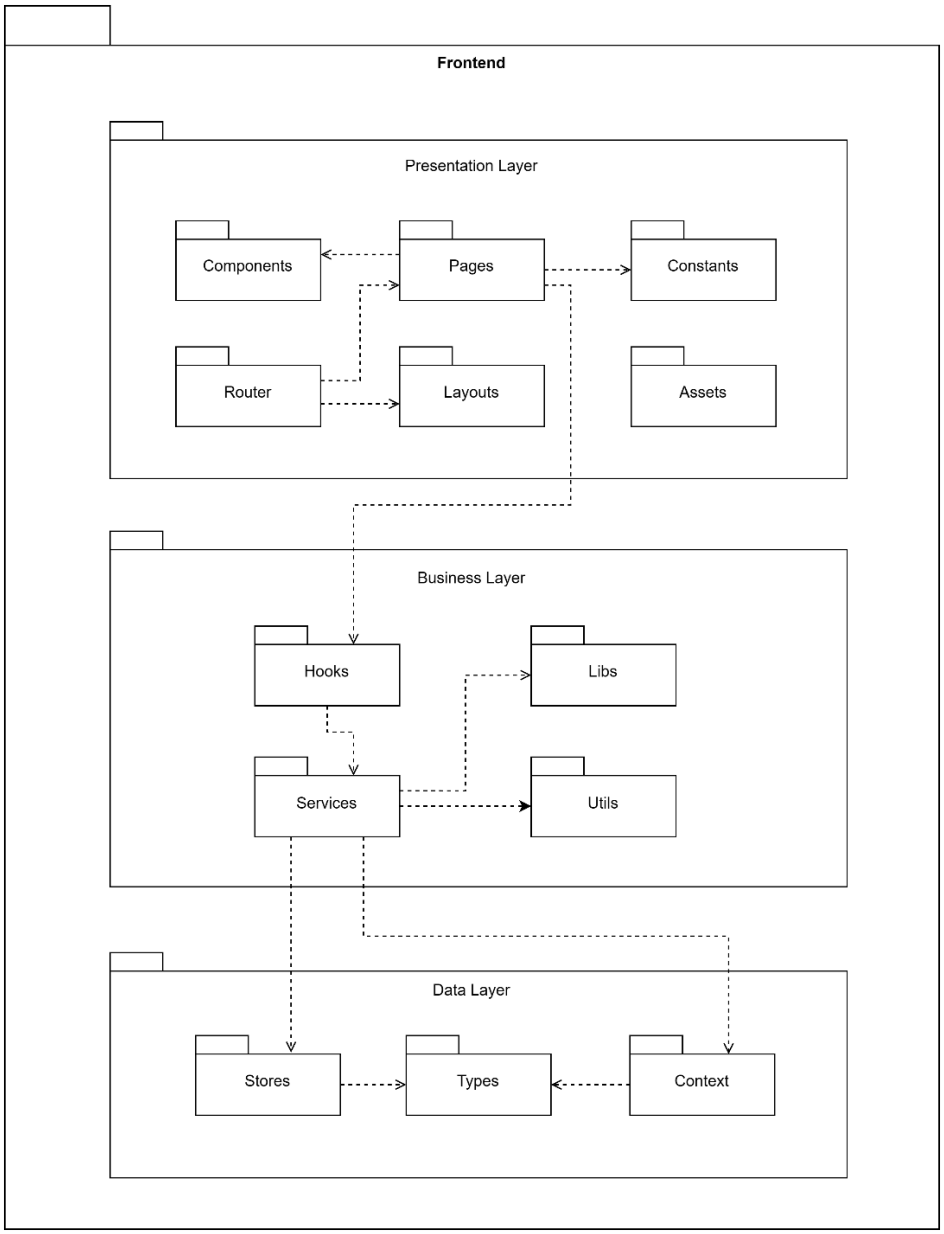


Figure 55 Frontend Package Diagram

ID	Package	Description
1	Components	Reusable UI elements, such as buttons, forms, or cards, that make up the interface.
2	Constants	Fixed values used throughout the application, such as data used for UI rendering.
3	Pages	The individual pages of the application, such as the homepage, login page, or account detail page.

4	Router	Navigation logic of UI pages.
5	Layouts	Predefined UI components that are used across pages such as headers and footers.
6	Assets	Any static files, such as images, fonts, vector logos, used by the application.
7	Hooks	Custom hooks provide a way to reuse stateful logic between components including calling APIs through services.
8	Services	Reusable functions for making side effects such as call REST APIs.
9	Libs	A gateway for adding custom configurations to external libraries before using them.
10	Utils	Reusable functions contain stateless operations such as format numbers, parse an array, etc.
11	Stores	Zustand stores include multiple ever-changing states need global access to avoid code duplications.
12	Types	Custom types that are used globally across functions and components.
13	Context	The seldom-changing states that need global access.

Table 28 Packages of the frontend

1.2.2. Backend Package Diagram

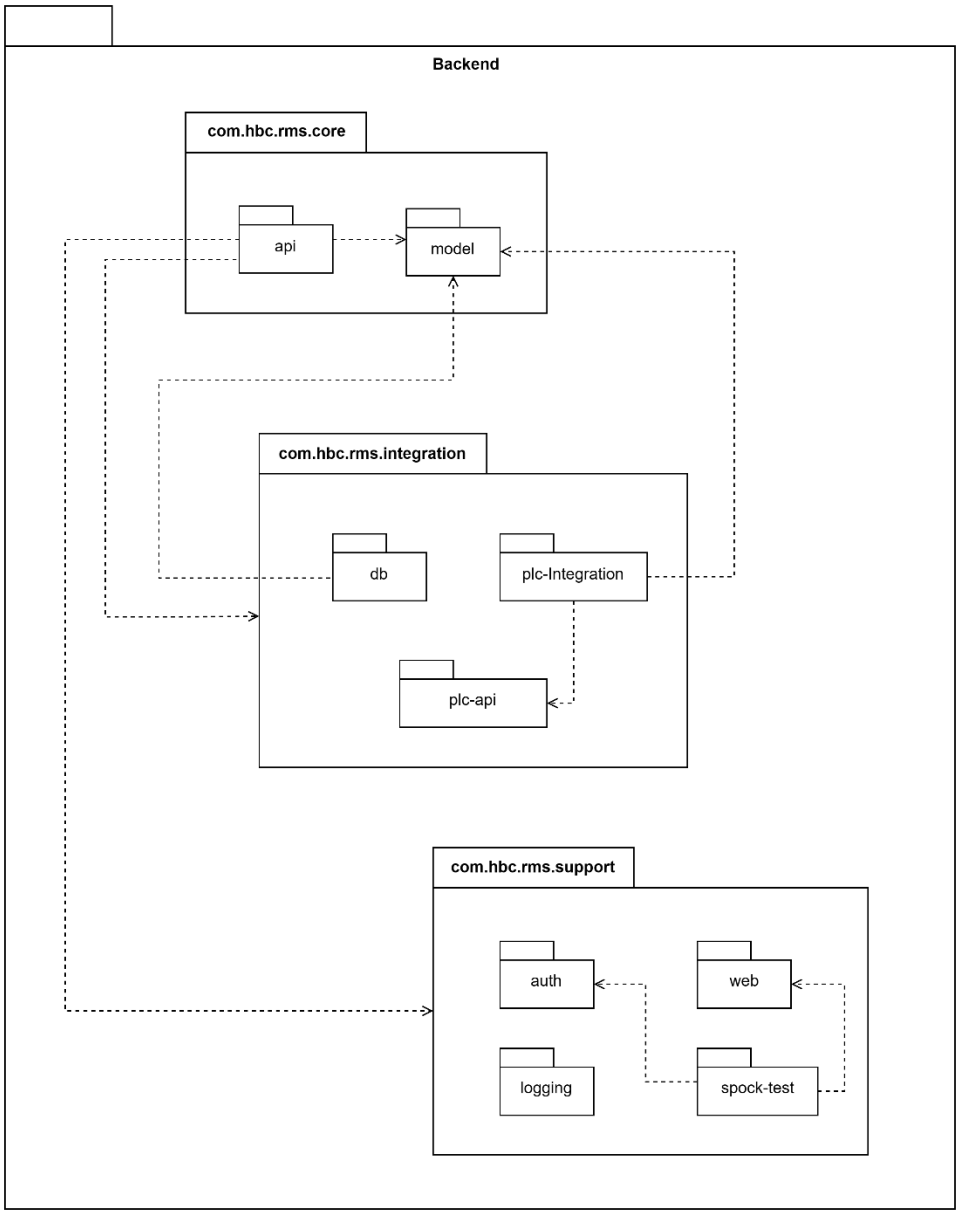


Figure 56 Package diagram of the backend.

ID	Package	Description
1	api	Essential components for handling REST APIs and WebSocket APIs such as controllers and services.
2	model	Defined classes that are used across multiple packages and classes as models.

3	db	Spring Data JPA-related classes for persisting data into database such as repositories and entities.
4	plc-integration	A class with scrapping logic for fetching data from the PLCs.
5	plc-api	Interfaces for api package to invoke the plc-integration package.
6	auth	Configurations for authentication and authorization.
7	web	Utility classes that are used across packages.
8	logging	Configurations for logging throughout the application.
9	spock-test	Classes contain test scenarios for automating functional test executions.

Table 29 Packages of the backend

2. Database Design

2.1. Physical Database Schema

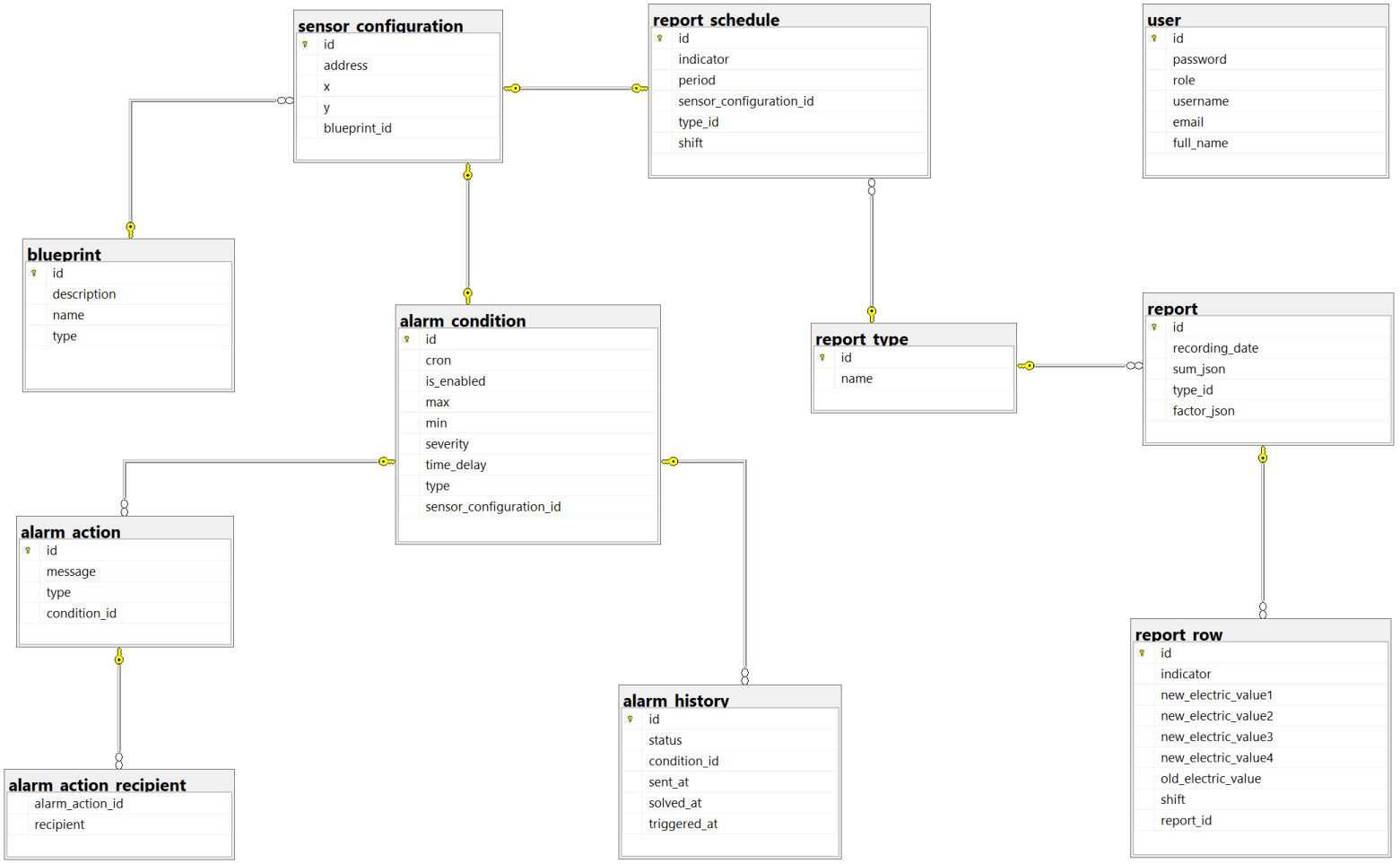


Figure 57 Physical database schema.

2.2. Table Descriptions

2.2.1. blueprint

ID	Field name	Type	Allow nulls	Unique	PK/FK	Note
	id	bigint			PK	
	description	varchar(255)	X			
	name	varchar(255)	X	X		
	type	varchar(255)	X			

Table 30 blueprint table's fields

2.2.2. sensor_configuration

ID	Field name	Type	Allow nulls	Unique	PK/FK	Note
	id	bigint			PK	
	address	varchar(255)	X			
	x	float	X			
	y	float	X			
	blueprint_id	bigint			FK	

Table 31 sensor_configuration table's fields

2.2.3. alarm_condition

ID	Field name	Type	Allow nulls	Unique	PK/FK	Note
	id	bigint			PK	
	cron	varchar(255)	X			
	is_enabled	bit	X			
	max	float	X			
	min	float	X			
	severity	varchar(255)	X			
	time_delay	int	X			
	type	varchar(255)	X			
	sensor_configuration_id	bigint			FK	

Table 32 alarm_condition table's fields

2.2.4. alarm_action

ID	Field name	Type	Allow nulls	Unique	PK/FK	Note
	id	bigint			PK	
	message	varchar(255)	X			
	type	varchar(255)	X			
	condition_id	bigint			FK	

Table 33 alarm_action table's fields

2.2.5. alarm_action_recipient

ID	Field name	Type	Allow nulls	Unique	PK/FK	Note
	alarm_action_id	bigint		X	FK	
	recipient	varchar(255)	X			

Table 34 alarm_action_recipient table's fields

2.2.6. alarm_history

ID	Field name	Type	Allow nulls	Unique	PK/FK	Note
	id	bigint			PK	
	status	varchar(255)	X			
	condition_id	bigint			FK	
	sent_at	datetimeoffset	X			
	solved_at	datetimeoffset	X			
	triggered_at	datetimeoffset	X			

Table 35 alarm_history table's fields

2.2.7. report_schedule

ID	Field name	Type	Allow nulls	Unique	PK/FK	Note
	id	bigint			PK	
	indicator	varchar(255)	X			
	period	varchar(255)	X			
	sensor_configuration_id	bigint			FK	
	type_id	bigint			FK	

	shift	varchar(255)	X			

Table 36 report_schedule table's fields

2.2.8. report_type

ID	Field name	Type	Allow nulls	Unique	PK/FK	Note
	id	bigint			FK	
	name	varchar(255)	X			

Table 37 report_type table's fields

2.2.9. report

ID	Field name	Type	Allow nulls	Unique	PK/FK	Note
	id	bigint			PK	
	recording_date	datetimeoffset	X			
	sum_json	varchar(6500)	X			
	type_id	bigint			FK	
	factor_json	varchar(6500)	X			

Table 38 report table's fields

2.2.10. report_row

ID	Field name	Type	Allow nulls	Unique	PK/FK	Note
	id	bigint			PK	
	indicator	varchar(255)	X			
	new_electric_value1	float	X			
	new_electric_value2	float	X			
	new_electric_value3	float	X			
	new_electric_value4	float	X			
	old_electric_value	float	X			
	shift	bigint	X			
	report_id	bigint			FK	

Table 39 report_row table's fields

2.2.11. user

ID	Field name	Type	Allow nulls	Unique	PK/FK	Note
	id	bigint			PK	
	password	varchar(255)				
	role	varchar(255)	X			
	username	varchar(255)		X		
	email	varchar(255)		X		
	full_name	nvarchar(255)	X			

Table 40 user table's fields

3. Detailed Design

3.1. Sequence Diagram

3.1.1. PLC Integration

Summary: This diagram illustrates the system interactions related to the PLC integration.

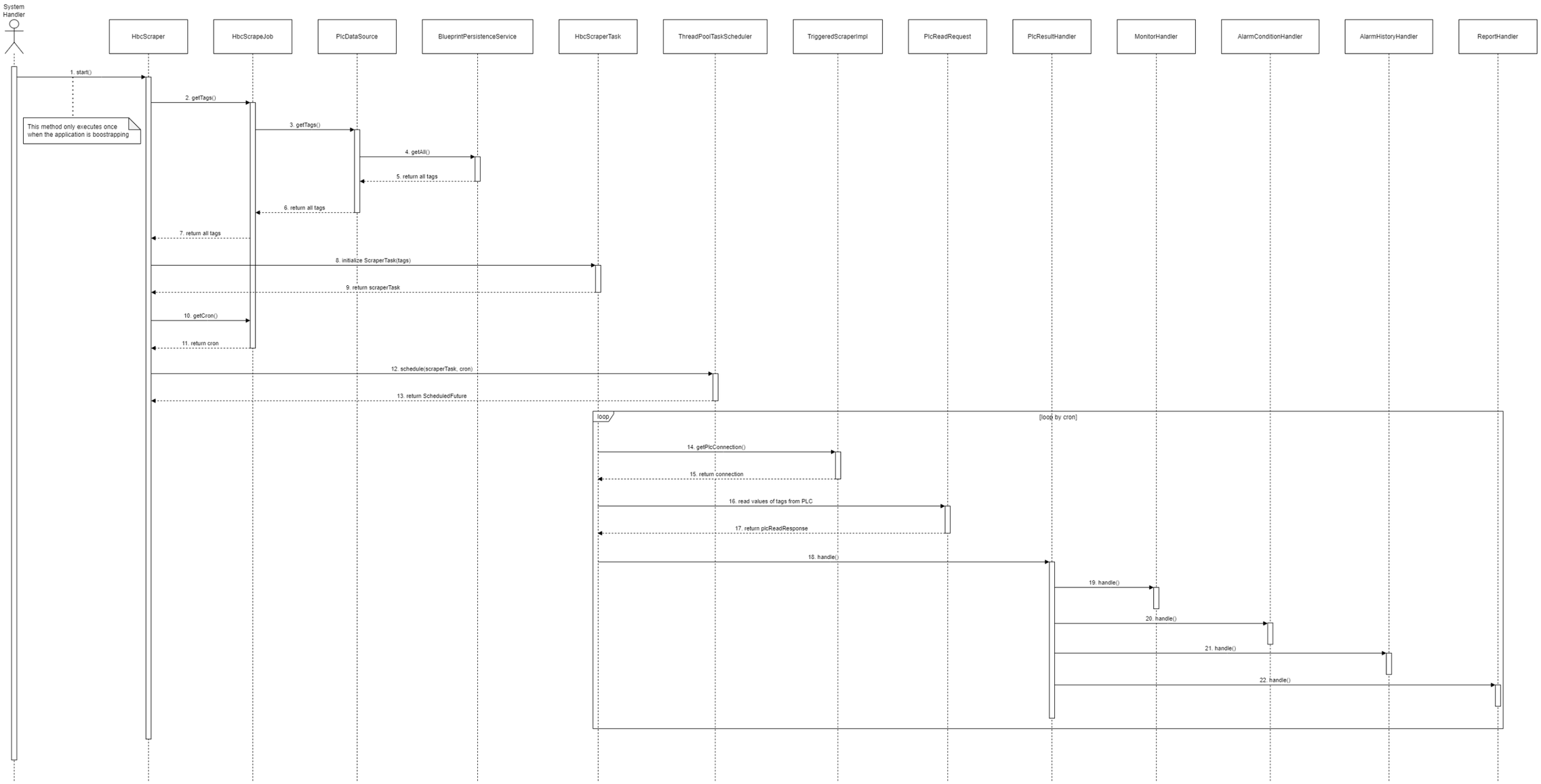


Figure 58 [Sequence diagram of PLC Integration flow]

3.1.2. Monitoring Feature

Summary: This diagram illustrates the system interactions related to the Monitoring feature after the PLC integration is invoked on the second basis.

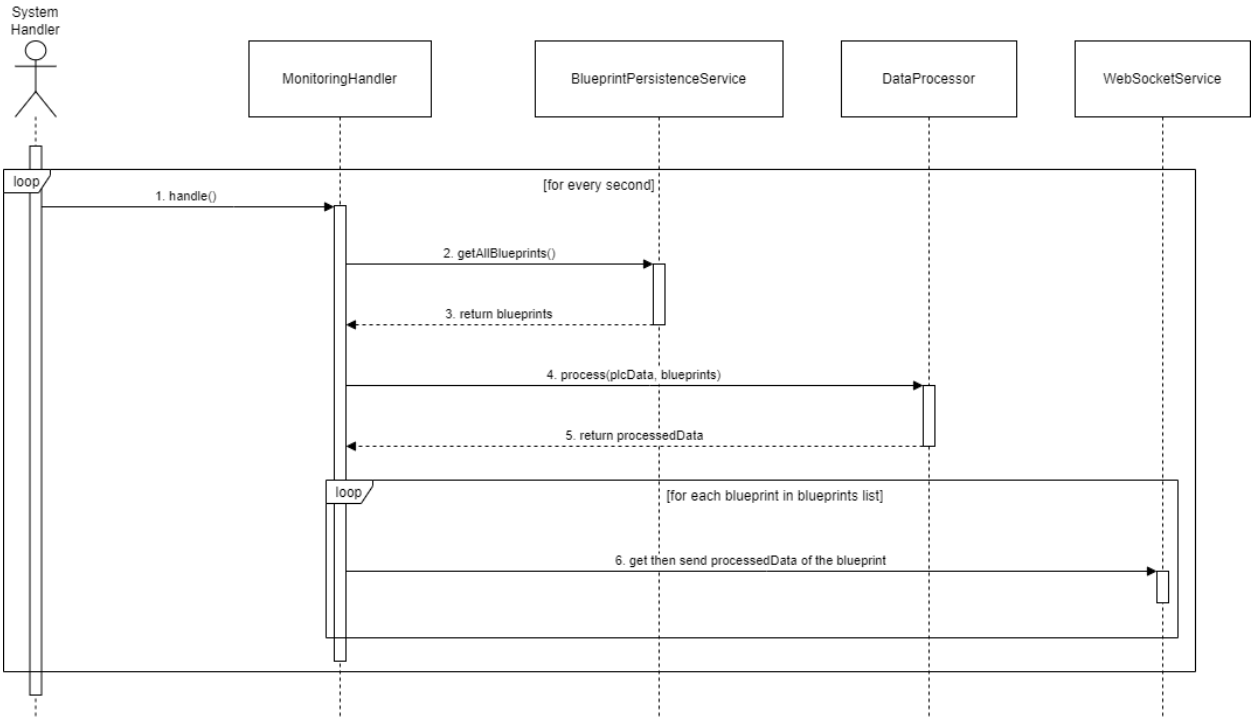


Figure 59 Sequence diagram of the Monitoring flow

3.1.3. Alarm Feature

Summary: This diagram illustrates the system interactions related to the Alarm feature after the PLC integration is invoked on the second basis.

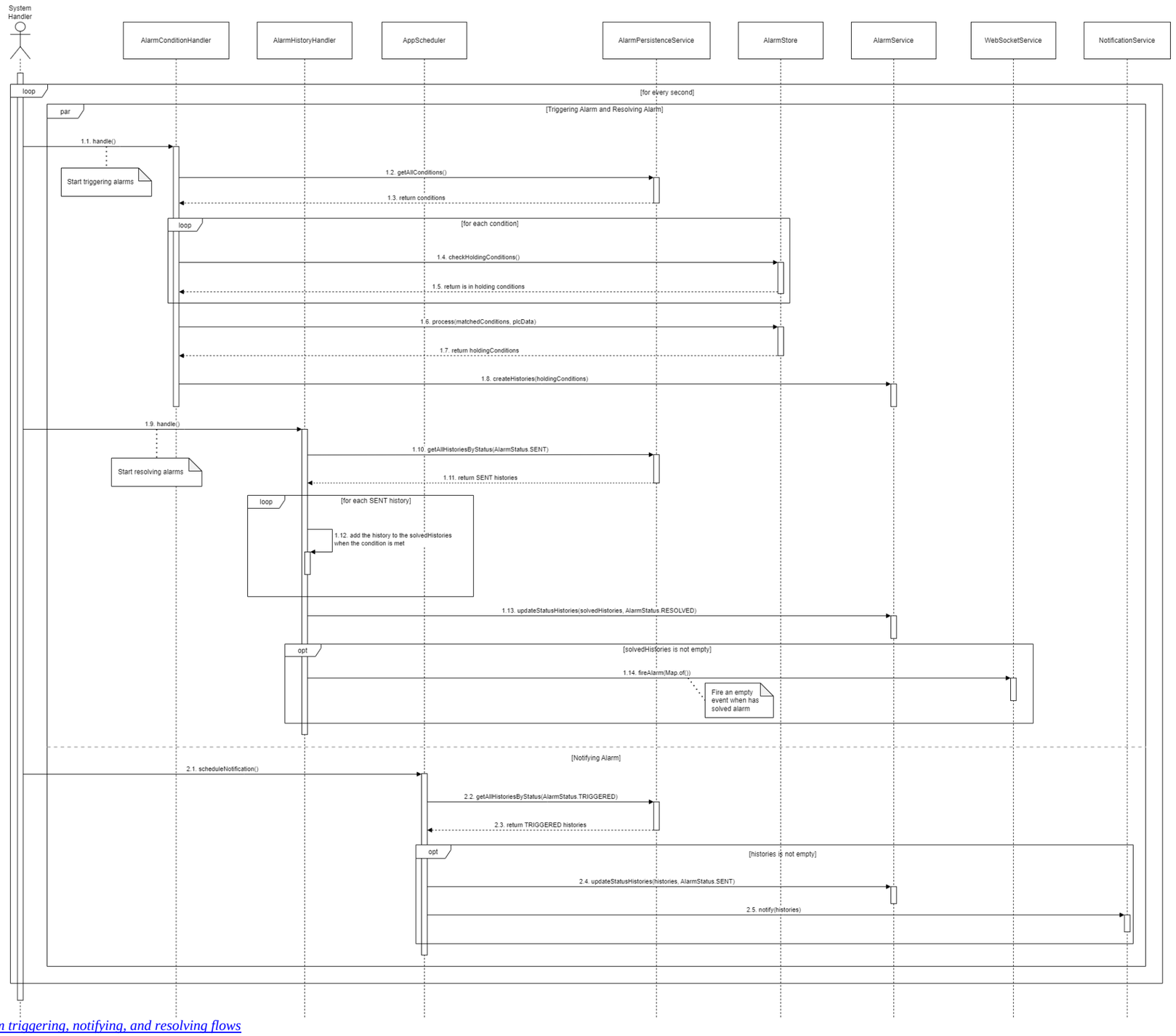


Figure 60 [Sequence diagram of the alarm triggering, notifying, and resolving flows]

3.1.4. Report Feature

3.1.4.a) Outbound flow

Summary: This diagram illustrates the system interactions related to View Statistical Charts of a specific date range function.

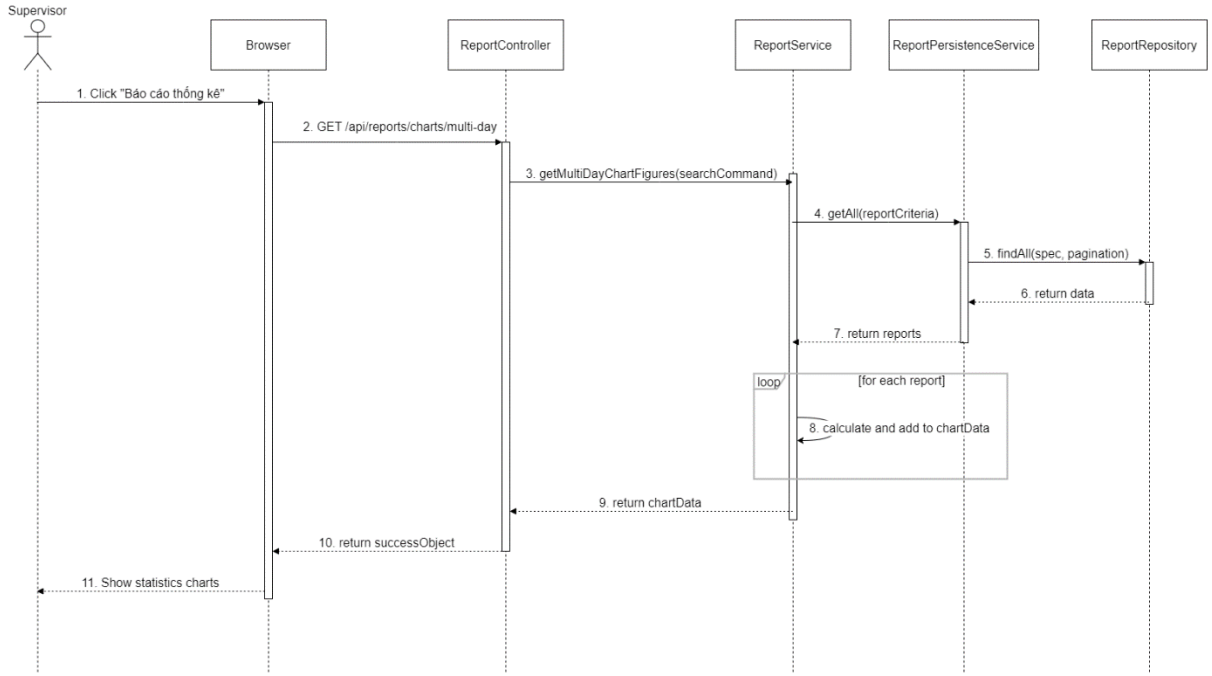


Figure 61 Sequence diagram of the ChartController in Report feature group

3.1.4.b) Inbound flow

Summary: This diagram illustrates the system interactions related to the Capture Data of the Previous Day function.

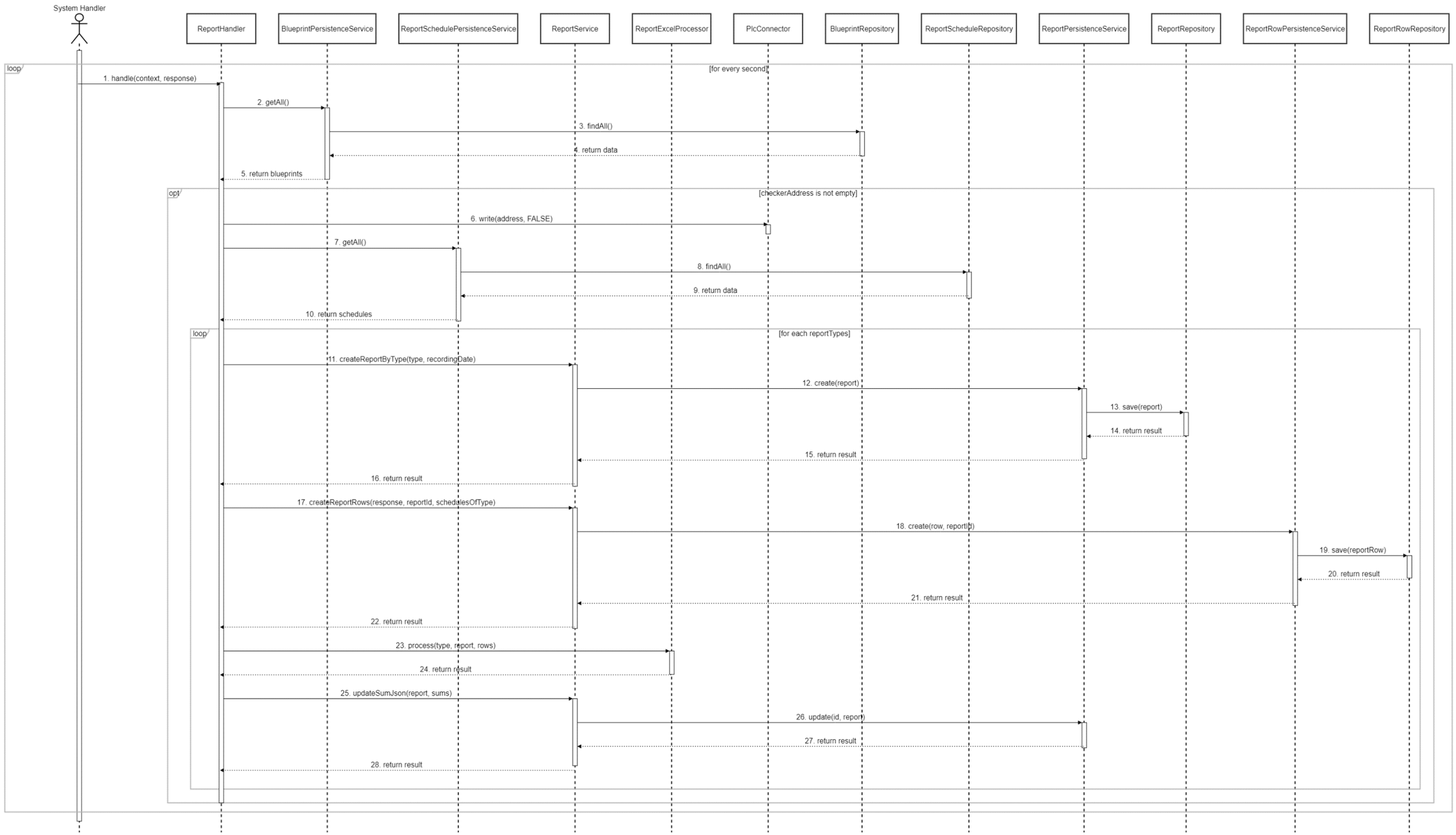


Figure 62 [Sequence diagram of Report capturing flow]

3.2. Class Diagram

3.2.1. User Management Feature

Below is the implementation class diagram of the function User Management in the Software Requirement Specification. The diagram depicts classes and relationships involving processes of handling REST APIs. In addition, to reduce the diagram's complexity, the necessary fields and classes are visualized only, not all in the implementation.

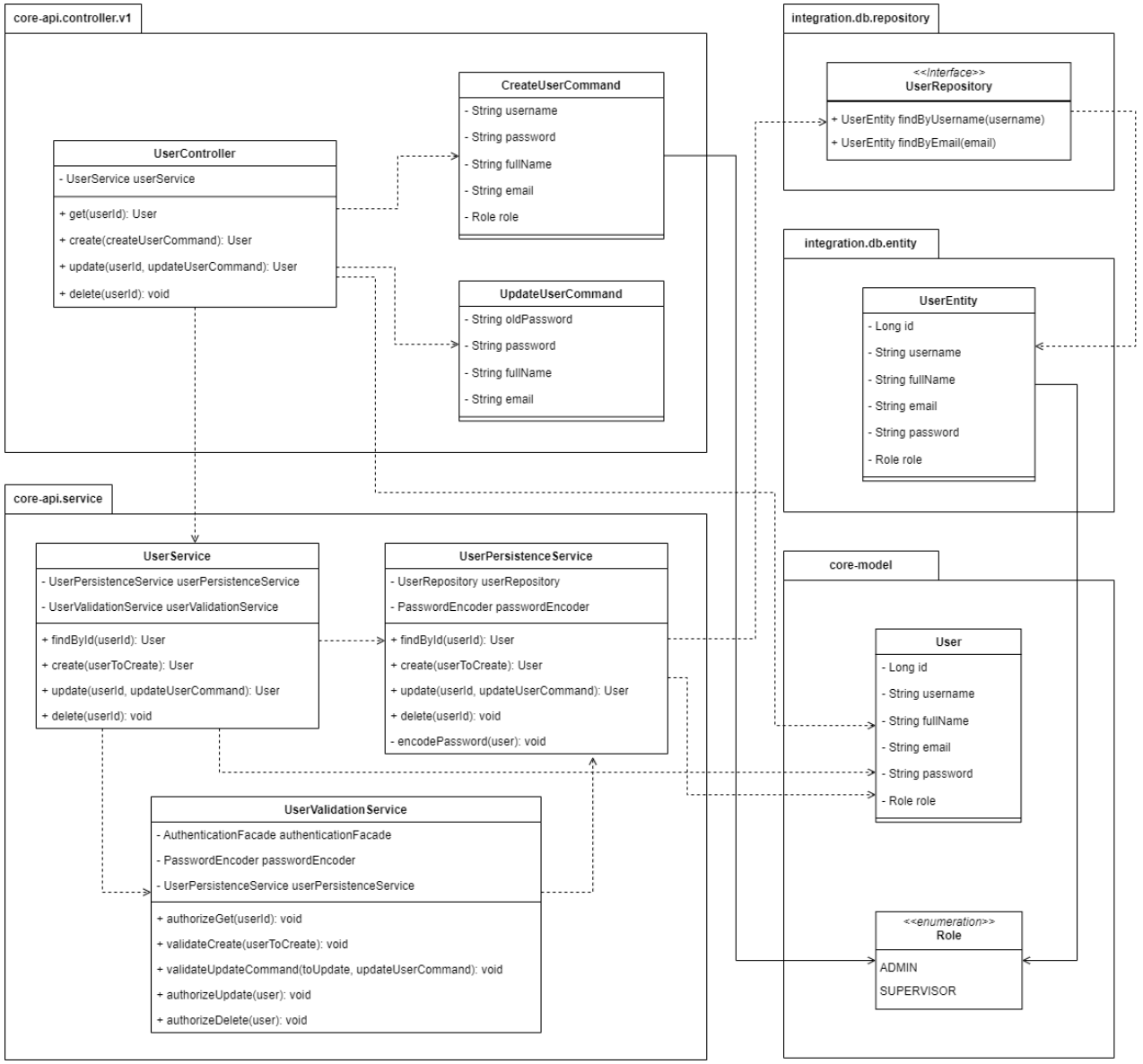


Figure 63 Class diagram of the User management feature group

3.2.1.a) UserController class

Description: contains method to handle REST APIs.

Field name	Type	Description
userService	UserService	View the UserService class section.

Table 41 UserController class's fields

Method	Parameter(s)	Return type	Description
get	userId - identifier of the user	User	Get a specific user information based on the given identifier by invoking appropriate methods of service classes.
create	createUserCommand – an instance of the CreateUserCommand class	User	Create a new user based on the given information by invoking appropriate methods of service classes.
update	updateUserCommand - an instance of the UpdateUserCommand class	User	Update an existing user's information by invoking appropriate methods of service classes.
delete	userId - identifier of the user	void	Delete an existing user based on the given identifier by invoking appropriate methods of service classes.

Table 42 UserController class's methods

3.2.1.b) CreateUserCommand class

Description: contains user information sent from the creating user API and each field is attached with constraints to perform validation.

Field name	Type	Description
username	String	The username of the user.
password	String	The password of the user.
fullName	String	The full name of the user.
email	String	The email with validation including matching standard email convention.
role	Role	View the Role section.

Table 43 CreateUserCommand class's fields

Method	Parameter(s)	Return type	Description
N/A	N/A	N/A	N/A

Table 44 CreateUserCommand class's methods

3.2.1.c) UpdateUserCommand class

Description: contains user information sent from the updating user API and each field is attached with constraints to perform validation.

Field name	Type	Description
oldPassword	String	The old password of the user.
password	String	The password of the user.
fullName	String	The full name of the user.
email	String	The email with validation including matching standard email convention.

Table 45 UpdateUserCommand class's fields

Method	Parameter(s)	Return type	Description
N/A	N/A	N/A	N/A

Table 46 UpdateUserCommand class's methods

3.2.1.d) UserService class

Description: contains methods invoked by the controller methods to perform appropriate operations.

Field name	Type	Description
userPersistenceService	UserPersistenceService	View the UserPersistenceService class section.
userValidationService	UserValidationService	View the UserValidationService class section.

Table 47 UserService class's fields

Method	Parameter(s)	Return type	Description

findById	userId - identifier of the user	User	Get a specific user information based on the given identifier by invoking appropriate methods of upstream service classes.
create	userToCreate – an instance of the User class	User	Create a new user based on the given information by invoking appropriate methods of upstream service classes.
update	userId - identifier of the user updateUserCommand - an instance of the UpdateUserCommand class	User	Update an existing user's information by invoking appropriate methods of upstream service classes.
delete	userId - identifier of the user	void	Delete an existing user based on the given identifier by invoking appropriate methods of upstream service classes.

Table 48 UserService class's methods

3.2.1.e) UserValidationService class

Description: contains methods invoked by downstream classes to validate user information or perform authorized operations.

Field name	Type	Description
authenticationFacade	AuthenticationFacade	Provide methods for authorizing users using Spring Security functionalities.
passwordEncoder	PasswordEnconder	An instance provided by Spring Security to perform encoding operations.
userPersistenceService	UserPersistenceService	View the UserPersistenceService class section.

Table 49 UserValidationService class's fields

Method	Parameter(s)	Return type	Description
authorizeGet	userId - identifier of the user	void	Check if the request sender has appropriate permissions to get user information.

validateCreate	userToCreate - an instance of the User class	void	Validate the email and username before creating the user.
validateUpdateCommand	toUpdate - an instance of the User class updateUserCommand - an instance of the UpdateUserCommand class	void	Validate the email and password before updating the user information.
authorizeUpdate	state - an instance of the User class	void	Check if the request sender has appropriate permissions to update user information.
authorizeDelete	user - an instance of the User class	void	Check if the request sender has appropriate permissions to delete user information.

Table 50 UserValidationService class's methods

3.2.1.f) UserPersistenceService class

Description: contains methods invoked by downstream services to interact with the Database.

Field name	Type	Description
userRepository	UserRepository	View the UserRepository interface section.
passwordEncoder	PasswordEnconder	An instance provided by Spring Security to perform encoding operations.

Table 51 UserPersistenceService class's fields

Method	Parameter(s)	Return type	Description
findById	userId - identifier of the user	User	Get a specific user information based on the given identifier by invoking appropriate methods of repository classes.
create	userToCreate – an instance of the User class	User	Create a new user based on the given information by invoking appropriate methods of repository classes.

update	userId - identifier of the user updateUserCommand - an instance of the UpdateUserCommand class	User	Update an existing user's information by invoking appropriate methods of repository classes.
delete	userId - identifier of the user	void	Delete an existing user based on the given identifier by invoking appropriate methods of repository classes.
encodePassword	user – an instance of the User class	void	Encode the raw password and replace it with encoded one.

Table 52 UserPersistenceService class's methods

3.2.1.g) UserRepository interface

Description: an interface for Spring Data JPA to create a repository class containing essential methods to interact with the UserEntity class in the database. Below are custom methods that are not provided by Spring Data.

Method	Parameter(s)	Return type	Description
findByUserName	username	Optional<UserEntity>	Get the first user that matches the given username by querying the database.
findByEmail	email	Optional<UserEntity>	Get the first user that matches the given email by querying the database.

Table 53 UserRepository class's methods

3.2.1.h) UserEntity class

Description: a class represents the table User in the database.

Field name	Type	Description
id	Long	The identifier of the user.
username	String	The username of the user.
fullName	String	The full name of the user.
email	String	The email of the user.
password	String	The password of the user.
role	Role	View the Role enumeration section.

Table 54 UserEntity class's fields

Method	Parameter(s)	Return type	Description
N/A	N/A	N/A	N/A

Table 55 UserEntity class's methods

3.2.1.i) User class

Description: a class whose instances are used for transferring user information between methods.

Field name	Type	Description
id	Long	The identifier of the user.
username	String	The username of the user.
fullName	String	The full name of the user.
email	String	The email of the user.
password	String	The password of the user.
role	Role	View the Role enumeration section.

Table 56 User class's fields

Method	Parameter(s)	Return type	Description
N/A	N/A	N/A	N/A

Table 57 User class's methods

3.2.1.j) Role enumeration

ADMIN: represent for the admin actor.

SUPERVISOR: represent for the supervisor actor.

3.2.2. Monitoring Feature

Below is the implementation class diagram of the Monitoring flow, which is related to the function Production Parameter Monitoring in the Software Requirement Specification. The diagram depicts classes and relationships involving processes after the PLC integration is invoked on the second basis. In addition, to reduce the complexity of the diagram, the necessary fields and classes are visualized only, not all fields and classes in the actual implementation.

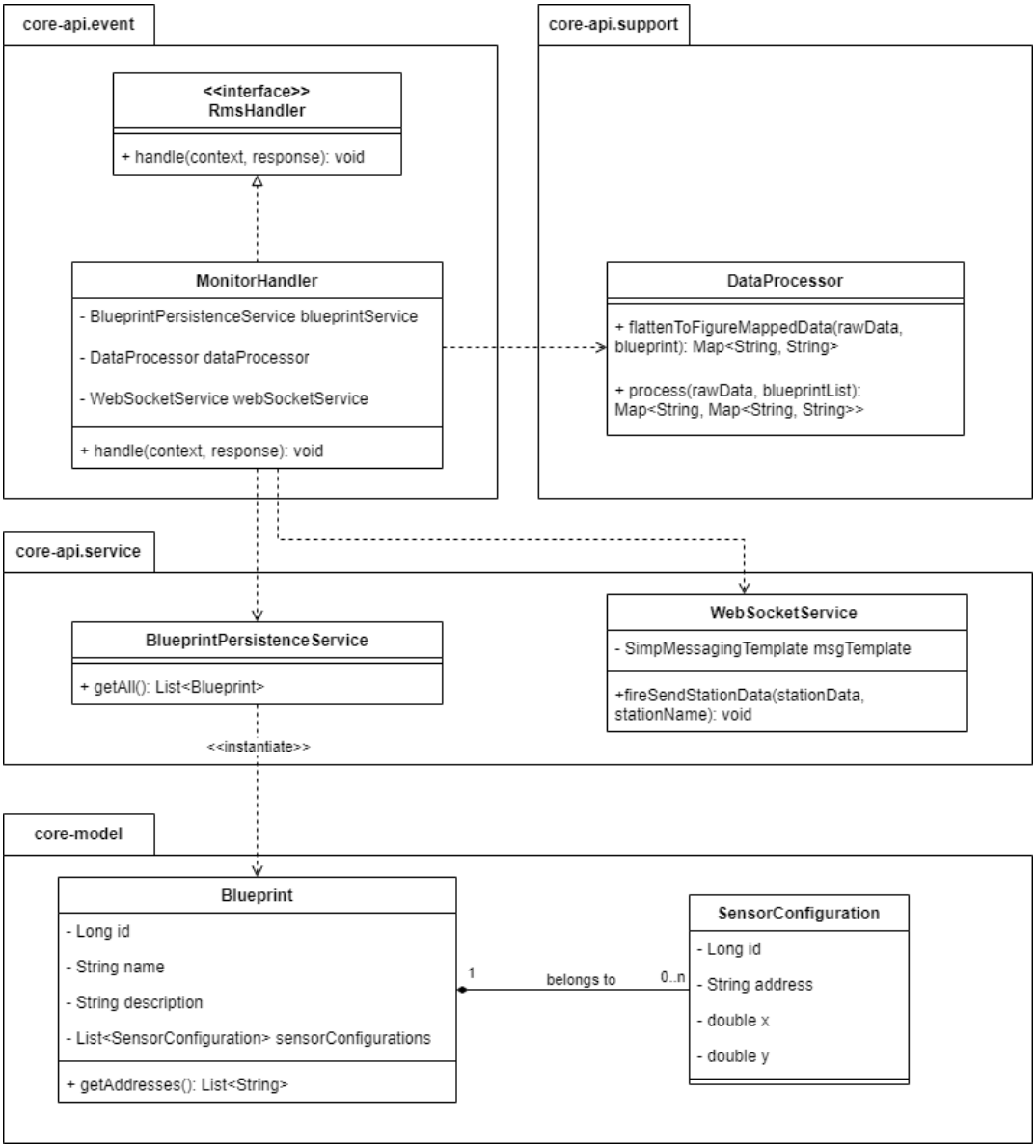


Figure 64 Class diagram of the Production Parameter Monitoring feature

3.2.2.a) RmsHandler interface

Description: handlers are objects that directly receive data from the PLC and then handle it based on their purposes.

Method	Parameter(s)	Return type	Description

handle	context – metadata of the job fetching data from the PLC. response – data received from the PLC.	void	Enforce all implementation classes to implement this method, read and extract needed data received from the PLC, and then handle those data depending on the purpose of those classes.

Table 58 RmsHandler class's methods

3.2.2.b) MonitorHandler class

Description: contains method to handle the process of parsing received data from the PLC then stream it back to the UI via the WebSocket.

Field name	Type	Description
blueprintService	BlueprintPersistenceService	View the BlueprintPersistenceService class section.
dataProcessor	DataProcessor	View the DataProcessor class section.
webSocketService	WebSocketService	View the WebSocketService class section.

Table 59 MonitorHandler class's fields

Method	Parameter(s)	Return type	Description
handle	context – metadata of the job fetching data from the PLC. response – data received from the PLC.	void	Parse the received data from the PLC, then group those data by blueprint and stream back to the UI using the WebSocket.

Table 60 MonitorHandler class's methods

3.2.2.c) DataProcessor class

Description: contains methods for converting data from raw PLC data to a readable Map structure for UI.

Field name	Type	Description
N/A	N/A	N/A

Table 61 DataProcessor class's fields

Method	Parameter(s)	Return type	Description
flattenToFigure MappedData	rawData – data received from the PLC. blueprint	Map<String, String>	Get all addresses of the SensorConfigurations in the Blueprint, then convert the rawData into a map format, whose key is the PLC Tag, and the value is the actual value at the Tag of the current fetch.
process	rawData – data received from the PLC. blueprintList	Map<String, Map<String, String>>	Loop through all MONITORING Blueprints in the system; on each Blueprint, call the flattenToFigureMappedData to convert the data to map format.

Table 62 DataProcessor class's methods

3.2.2.d) BlueprintPersistenceService class

Field name	Type	Description
N/A	N/A	N/A

Table 63 BlueprintPersistenceService class's fields

Method	Parameter(s)	Return type	Description
getAll	None	List<Blueprint>	Get all available Blueprints in the system.

Table 64 BlueprintPersistenceService class's methods

3.2.2.e) WebSocketService class

Description: Service for aggregating data and calling the publish method of the actual WebSocket.

Field name	Type	Description
template	SimpMessagingTemplate	The necessary object of the Spring STOMP library, provides methods for sending messages to a user.

Table 65 BlueprintPersistenceService class's fields

Method	Parameter(s)	Return type	Description
fireSendStationData	stationData – a map storing all current values read from the PLC. stationName – the station (or blueprint) name.	void	Stream the stationData to the WebSocket endpoint in the format "/topic/{stationName}".

Table 66 BlueprintPersistenceService class's methods

3.2.2.f) Blueprint class

Description: for grouping the SensorConfiguration in the project

Field name	Type	Description
id	Long	Unique identifier of the Blueprint.
name	String	Human readable identifier of the Blueprint, labeled with station name or feature name.
description	String	Description of the Blueprint.
sensorConfigurations	SensorConfigration	View the SensorConfiguration class section.

Table 67 Blueprint class's fields

Method	Parameter(s)	Return type	Description
getAddresses	None	List<String>	Get all addresses (PLC Tag) of all sensorConfigurations belonging to this Blueprint.

Table 68 Blueprint class's methods

3.2.2.g) SensorConfiguration class

Field name	Type	Description
id	Long	Unique identifier of the SensorConfiguration.
address	string	PLC Tag.
x	Double	The x in coordination of this PLC Sensor Configuration in the Electrical Schematic diagram on the user interface.
y	Double	The y in coordination of this PLC Sensor Configuration in the Electrical Schematic diagram on the user interface.

Table 69 SensorConfiguration class's fields

Method	Parameter(s)	Return type	Description
N/A	N/A	N/A	N/A

Table 70 SensorConfiguration class's methods

3.2.3. Alarm Feature

Below is the implementation class diagram of the Alarm triggering, notifying, and resolving flows. The diagram depicts classes and relationships involving processes of each flow after the PLC integration is invoked on the second basis. Although these are 3 separate flows, visualizing them all into 1 diagram shall offer a more comprehensive overview of the alarm lifecycle in the factory: triggering the alarm → notifying the user of the alarm → resolving the alarm when the condition is not met anymore.

In addition, to reduce the complexity of the diagram, the necessary fields and classes are visualized only, not all fields and classes in the actual implementation.

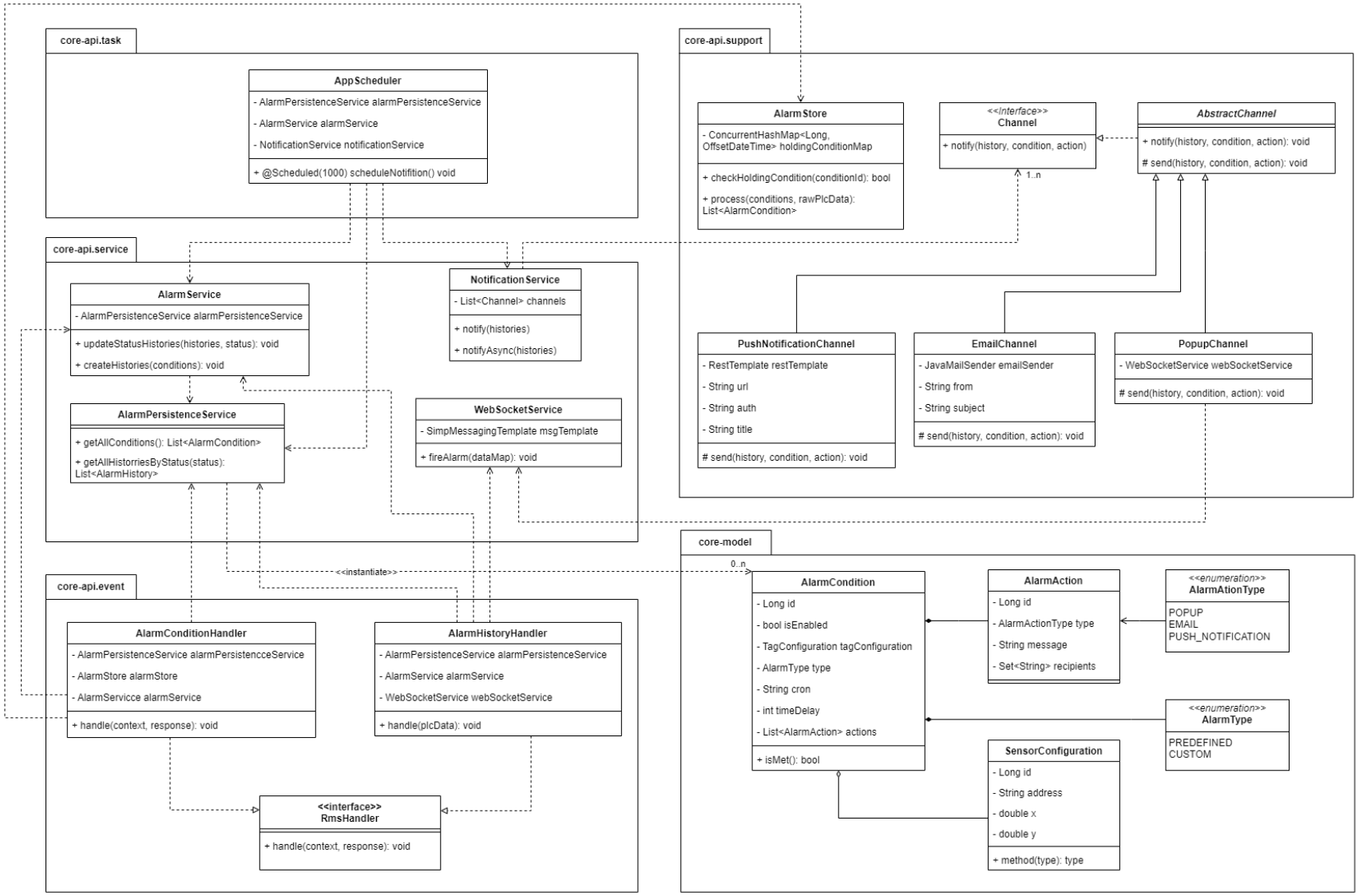


Figure 65 Class diagram of the Alarm triggering, notifying and resolving flows

3.2.3.a) RmsHandler interface

Description, fields, and methods: view the RmsHandler interface section.

3.2.3.b) AlarmHistoryHandler class

Description: contains a method for resolving an unresolved alarm condition.

Field name	Type	Description
alarmPersistenceService	AlarmPersistenceService	View the AlarmPersistenceService class section.
alarmService	AlarmService	View the AlarmService class section.
webSocketService	WebSocketService	View the WebSocketService class section.

Table 71 AlarmHistoryHandler class's fields

Method	Parameter(s)	Return type	Description
handle	context – metadata of the job fetching data from the PLC response – data received from the PLC	void	Compare received PLC Data to unsolved histories in the system to find any history that can be solved.

Table 72 AlarmHistoryHandler class's methods

3.2.3.c) AlarmConditionHandler class

Description: contains method for comparing SensorConfigurations' values against the Condition to detect that any alarm should be triggered.

Field name	Type	Description
alarmPersistenceService	AlarmPersistenceService	View the AlarmPersistenceService class section.
alarmService	AlarmService	View the AlarmService class section.
alarmStore	AlarmStore	View the AlarmStore class section.

Table 73 AlarmConditionHandler class's fields

Method	Parameter(s)	Return type	Description

handle	context – metadata of the job fetching data from the PLC. response – data received from the PLC	void	Compare received PLC Data to all alarm conditions in the system to trigger alarms promptly.

Table 74 AlarmConditionHandler class's methods

3.2.3.d) AppScheduler class

Description: contains method to run continuously each second.

Field name	Type	Description
alarmPersistenceService	AlarmPersistenceService	View the AlarmPersistenceService class section.
alarmService	AlarmService	View the AlarmService class section.
notificationService	NotificationService	View the NotificationService class section.

Table 75 AppScheduler class's fields

Method	Parameter(s)	Return type	Description
@Scheduled(1000) scheduleNotification	None	void	Run continuously every second since the app started. On each run, it queries all the TRIGGERED histories to send the notify action accordingly.

Table 76 AppScheduler class's methods

3.2.3.e) AlarmService class

Description: Service for querying and modifying AlarmHistories with additional logic.

Field name	Type	Description
alarmPersistenceService	AlarmPersistenceService	View the AlarmPersistenceService class section.

Table 77 AlarmService class's fields

Method	Parameter(s)	Return type	Description

createHistories	conditions – list of Alarm Conditions	void	Loop through all the conditions, on each one, and check if this condition is having an unsolved alarm history already, if not, create a new history of this alarm condition.
updateStatusHistories	histories – list of Alarm Histories status – new status for the histories	void	Update the status of the histories.

Table 78 AlarmService class's methods

3.2.3.f) NotificationService class

Field name	Type	Description
channels	List<Channel>	List of available Channels.

Table 79 NotificationService class's fields

Method	Parameter(s)	Return type	Description
notify	histories	void	Loop through all alarm histories; on each history, loop through all its actions and call notifyAsync().
notifyAsync	history condition action	void	Loop through all channels and notify an alarm based on given inputs.

Table 80 NotificationService class's methods

3.2.3.g) AlarmPersistenceService class

Description: Service for directly querying and modifying AlarmCondition and AlarmHistories without additional logic.

Field name	Type	Description
N/A	N/A	N/A

Table 81 AlarmPersistenceService class's fields

Method	Parameter(s)	Return type	Description
getAllConditions	None	List<AlarmCondition>	Get all available alarm conditions in the database.

getAllHistoriesByStatus	status – the status to query	List<AlarmHistories>	Get all available alarm histories in the database with the given status.

Table 82 AlarmPersistenceService class's methods

3.2.3.h) WebSocketService class

Description: Service for aggregating data and calling the publishing method of the actual WebSocket.

Field name	Type	Description
template	SimpMessagingTemplate	The necessary object of the Spring STOMP library, provides methods for sending messages to a user.

Table 83 WebSocketService class's fields

Method	Parameter(s)	Return type	Description
fireAlarm	alarmData – a map storing some necessary information of an alarm	void	Call convertAndSend() method of the SimpMessagingTemplate to send a message to the user.

Table 84 WebSocketService class's methods

3.2.3.i) AlarmStore class

Description: Class for storing, deciding whether to add to or delete from holdingConditions list – conditions met before but still not in enough timeDelay.

Field name	Type	Description
holdingConditionMap	ConcurrentHashMap	Storing all AlarmCondition that is met but hasn't been kept for a `timeDelay` seconds yet.

Table 85 AlarmStore class's fields

Method	Parameter(s)	Return type	Description
checkHoldingCondition	Long conditionId – Id of the Condition to check	boolean	Check if the condition exists in the holdingConditionMap already

process	conditions rawData – raw data received from the PLC	List<AlarmCondition>	Loop through all conditions; on each one, it checks if the condition is met and is met for how long already to add it to the holdingConditionMap.

Table 86 AlarmStore class's methods

3.2.3.j) Channel interface

Description: channel objects are objects that handle the actual notification sending, in e.g., email, WebSocket, push notification.

Field name	Type	Description
N/A	N/A	N/A

Table 87 Channel interface's fields

Method	Parameter(s)	Return type	Description
notify	history condition action	void	Enforce all implementation classes to explicitly implement this method to notify the user of the history and condition.

Table 88 Channel interface's methods

3.2.3.k) AbstractChannel abstract class

Description: implementation of the Channel interface, which contains notify(), recover(), filter(), and send() methods with some retriable mechanism on exception when sending.

Field name	Type	Description
N/A	N/A	N/A

Table 89 AbstractChannel abstract class's fields

Method	Parameter(s)	Return type	Description
notify	history condition action	void	Enforce all child classes to have this method to filter if the channel type matches the action type, if yes then process to the send function.

send	history condition action	void	Enforce all child classes to have this method to do the actual send to the user.

Table 90 AbstractChannel abstract class's methods

3.2.3.l) EmailChannel class

Description: implementation of the AbstractChannel abstract class, which contains a method for sending the alarm with action type = EMAIL to the registered user email.

Field name	Type	Description
mailSender	JavaMailSender	Contains a method to send an email to the user.
from	String	Default "from" email of the RMS required for an email to be able to be sent.
subject	String	Default "subject" field of the email

Table 91 EmailChannel class's fields

Method	Parameter(s)	Return type	Description
send	history condition action	void	Query all the recipients from the action, then create the necessary Email object for the JavaMailSender to send email.

Table 92 EmailChannel class's methods

3.2.3.m) PopupChannel class

Description: implementation of the AbstractChannel abstract class, which contains the method for firing an alarm to the UI.

Field name	Type	Description
webSocketService	WebSocketService	View the WebSocketService class section.

Table 93 PopupChannel class's fields

Method	Parameter(s)	Return type	Description
send	history condition action	void	Fire an alarm message to the "/topic/alarm" WebSocket endpoint.

Table 94 PopupChannel class's methods

3.2.3.n) PushNotificationChannel class

Description: implementation of AbstractChannel abstract class, which contains the method for calling to a configured endpoint of the ntfy 3rd-party notification service.

Field name	Type	Description
restTemplate	RestTemplate	A REST API client object
url	String	URL destination provided by the ntfy 3rd-party push notification service.
auth	String	Authentication secret for the ntfy service.
title	String	Content of the notification.

Table 95 PushNotificationChannel class's fields

Method	Parameter(s)	Return type	Description
send	history condition action	void	Create necessary objects for sending an API call to ntfy service and called it with the above properties. The ntfy will send a push notification when receive a call to the configured endpoint.

Table 96 PushNotificationChannel class's methods

3.2.3.o) AlarmCondition class

Description: alarm condition model

Field name	Type	Description
id	Long	Unique identifier of the AlarmCondition.
isEnabled	boolean	Is this condition enabled in the RMS system.
sensorConfiguration	SensorConfiguration	The SensorConfiguration with the PLC Tag has the value this condition wants to check against.
type	AlarmType	View the AlarmType enumeration section.
cron	String	Cron string of how frequent this condition should be checked in the system.
timeDelay	int	How many seconds the condition should be met before triggering an alarm.
actions	List<Action>	List of AlarmAction class.

Table 97 AlarmCondition class's fields

Method	Parameter(s)	Return type	Description

isMet	None	boolean	Contains some logic to verify if a condition is met, it has 2 different checking logics for 2 different types of alarm.

Table 98 AlarmCondition class's methods

3.2.3.p) AlarmAction class

Description: define what action that the system should do when the belonging AlarmCondition is triggered.

Field name	Type	Description
id	Long	Unique identifier of the AlarmAction.
type	AlarmActionType	View the AlarmActionType enumeration section.
message	string	Message of this action.
recipients	Set<String>	Required at least 1 recipient in the set if this action is EMAIL type. The other types don't need this field.

Table 99 AlarmAction class's fields

Method	Parameter(s)	Return type	Description
N/A	N/A	N/A	N/A

Table 100 AlarmAction class's methods

3.2.3.q) AlarmActionType enumeration

EMAIL: Send the alarm to registered recipients.

POPUP: Toast an alarm message for 10s for all users who have active RMS sessions.

PUSH_NOTIFICATION: Send a push notification to all users who installed the ntfy application and subscribe to the RMS's notification channel.

3.2.3.r) AlarmType enumeration

PREDEFINED: An alarm represented by a bool PLC Tag in the PLC. If the value of this Tag is true, then the condition is triggered, and vice versa.

CUSTOM: An alarm that is configurable from the RMS UI allows the user to set some alarm conditions without the need to modify the current PLC code whenever a new demand for an alarm is needed.

3.2.3.s) SensorConfiguration class

Description, fields, and methods: View the SensorConfiguration class section.

3.2.4. Report Feature

Below is the implementation class diagram of the Report flow, which is related to the Capture Data of the Previous Day and View Statistical Charts of a specific date range feature in the Software Requirement Specification. The diagram depicts classes and relationships involving processes of each flow after the PLC integration is invoked on the second basis.

In addition, to reduce the complexity of the diagram, the necessary fields and classes are visualized only, not all fields and classes in the actual implementation.

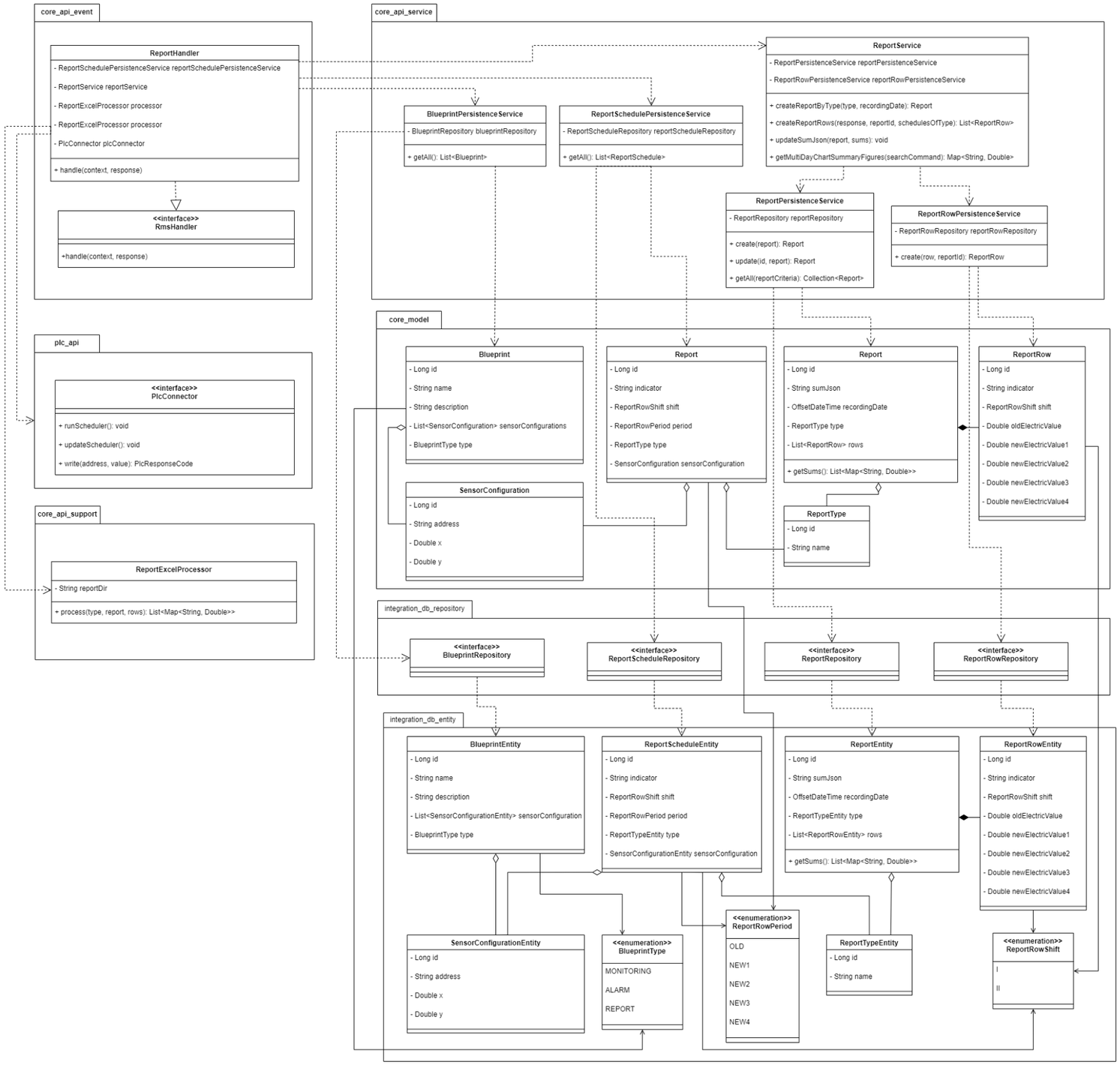


Figure 66 [Class diagram of the Capture Data of the Previous Day and View Statistical Charts of a specific date range functions]

3.2.4.a) RmsHandler interface

Description, fields, and methods: view the RmsHandler interface section.

3.2.4.b) ReportHandler class

Description:

Field name	Type	Description
reportSchedulePersistenceService	ReportSchedulePersistenceService	View the ReportSchedulePersistenceService section.
reportService	ReportService	View the ReportService section.
processor	ReportExcelProcessor	View the ReportExcelProcessor section.
blueprintService	BlueprintPersistenceService	View the BlueprintPersistenceService section.

Table 101 ReportHandler class's fields

Method	Parameter(s)	Return type	Description
handle	context – metadata of the job fetching data from the PLC. response – data received from the PLC	void	Check the value of the flag PLC Tag and map the received data from the PLC to the database.

Table 102 ReportHandler class's methods

3.2.4.c) ReportSchedulePersistenceService class

Description: Service for directly querying Reports without additional logic.

Field name	Type	Description
reportScheduleRepository	ReportScheduleRepository	Repository layer for report schedule.

Table 103 ReportSchedulePersistenceService class's fields

Method	Parameter(s)	Return type	Description
getAll	None	List<ReportSchedule>	Get all report schedules in the database.

Table 104 ReportSchedulePersistenceService class's methods

3.2.4.d) ReportService class

Description:

Field name	Type	Description
reportPersistenceService	ReportPersistenceService	View the ReportPersistenceService section.
reportRowPersistenceService	ReportRowPersistenceService	View the ReportRowPersistenceService section.
reportTypePersistenceService	ReportTypePersistenceService	View the ReportTypePersistenceService section.

Table 105 ReportService class's fields

Method	Parameter(s)	Return type	Description
createReportByType	type recordingDate	Report	Used to create a new Report of a ReportType.
createReportRows	response reportId schedulesOfType	List<ReportRow>	Used to store electrical information in the database
updateSumJson	report sums	void	Update sum to database
getOneDayChartFigures	reportId	OneDayChartResponse	Get information about the chart for one day
getMultiDayChartSummaryFigures	searchCommand	Map<String, Double>	Get chart summary for multiple days
getMultiDayChartFigures	searchCommand	MultiDayChartResponse	Get information about the chart for multiple days
getMissingDatesInReportsGroupByType	searchCommand	Map<String, List<OffsetDateTime>>	Get missing dates in reports

Table 106 ReportService class's methods

3.2.4.e) ReportTypePersistenceService class

Description: Service for directly querying ReportType without additional logic.

Field name	Type	Description
reportTypeRepository	ReportTypeRepository	Repository layer for report type.

Table 107 ReportTypePersistenceService class's fields

Method	Parameter(s)	Return type	Description
getAll	None	List<ReportType>	Get all report types in the database.

Table 108 ReportTypePersistenceService class's methods

3.2.4.f) ReportPersistenceService class

Description: Service for directly querying Report without additional logic.

Field name	Type	Description
reportRepository	ReportRepository	Repository layer for report.

Table 109 ReportPersistenceService class's fields

Method	Parameter(s)	Return type	Description
getAll	criteria	List<Report>	Get all report in the database.
getById	id	Report	Get a report by id in the database.
create	report	Report	Create a new report in database.
update	id report	Report	Update the report in database.

Table 110 ReportPersistenceService class's methods

3.2.4.g) ReportRowPersistenceService class

Description: Service for directly querying ReportRow without additional logic.

Field name	Type	Description
reportRowRepository	ReportRowRepository	Repository layer for report row.

Table 111 ReportRowPersistenceService class's fields

Method	Parameter(s)	Return type	Description
create	reportRow	ReportRow	Create a report row in the database.

Table 112 ReportRowPersistenceService class's methods

3.2.4.h) ReportExcelProcessor class

Description: the class for processing Excel templates.

Method	Parameter(s)	Return type	Description
process	reportType report reportRows	List<Map<String, Double>>	Fill in electric data and calculate sum.

Table 113 ReportExcelProcessor class's methods

3.2.4.i) PlcConnector interface

Description: the interface is responsible for interacting with the PLC

Method	Parameter(s)	Return type	Description
write	address value	PlcResponseCode	Write a value to the address.

Table 114 PlcConnector interface's methods

3.2.4.j) ReportRowShift enumeration

I: first shift in one day

II: second shift in one day

3.2.4.k) ReportRowPeriod enumeration

OLD: locate the old electricity value in the Excel template

NEW1: locate the new electricity value 1 in the Excel template

NEW2: locate the new electricity value 2 in the Excel template

NEW3: locate the new electricity value 3 in the Excel template

NEW4: locate the new electricity value 4 in the Excel template

3.3. Activity Diagram

3.3.1. Alarm Feature

Summary: This diagram illustrates the process of triggering, notifying, and resolving alarms.

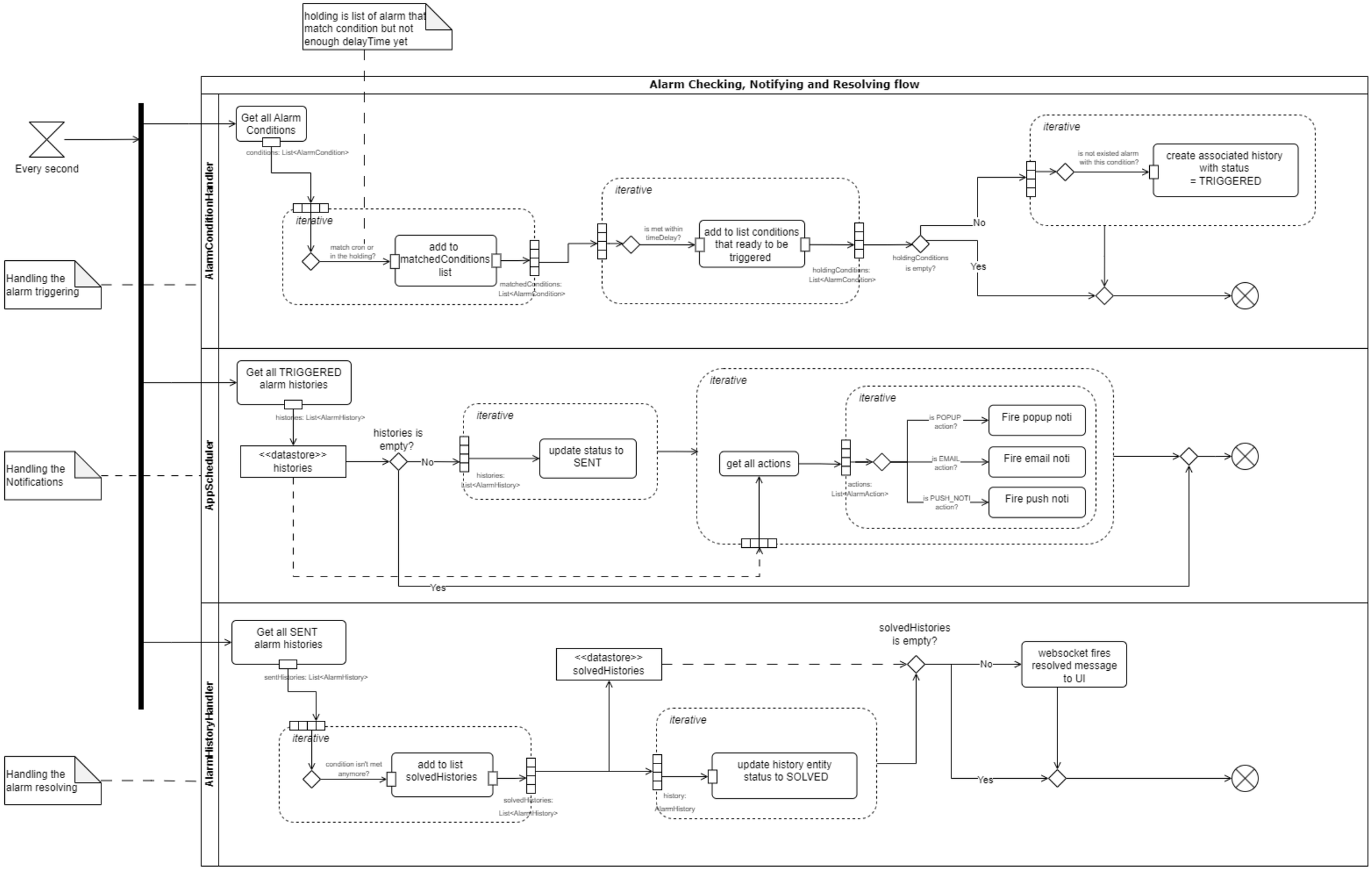


Figure 67 [*Activity diagram of the process of triggering, notifying, and resolving alarms*]

V Software Testing Document

1. Overview

1.1. Scope of Testing

As stated in the AC4 given in the Project Scope section in the Software Requirement Specification, which is "*by ensuring that all functional test cases pass with at least 75% line of code coverage to ensure the codebase's integrity*," the Scope of Testing for this project shall need to have enough functional test cases and integration test cases to cover every existed usecases from the Software Requirement Specification.

1.2. Test Strategy

Objective	Technique	Completion criteria
Automatedly testing the system from the end user's perspective, ensuring correct functionality.	Functional testing	Have at least 1 happy and 1 negative test case for each usecase, function, or endpoint. All registered test cases in the codebase must be passed. Enforce to have 75% line-of-code coverage in the project.
Focusing on checking the appearance, functionality, and usability of the visual elements of an interface.	User interface testing	Ensure the UI has consistency and accessibility in the interface.
Replicating the user behaviors with the software in a complete environment to verify all main available workflows	End-to-end testing	All conducted workflows pass the expected behaviors.
Letting intended users use the software in the real world to see if the system addresses their needs.	User acceptance testing	Receive approval from the main users on the features agreed upon in the requirement.

Table 115 Test Strategy

1.3. Test Plan

1.4. Human Resources

Worker/Doer	Role	Responsibilities

Bùi Ngọc Huy	Leader	Preparing test plans and test environments. Set up necessary tools/libraries for functional testing.
Lê Tiến Thịnh	Member	Write automated functional test cases. End-to-end testing.
Trần Trung Kiên		
Lê Xuân Đại		User interface testing. End-to-end testing.
Nguyễn Nhật Huy		API testing.
Mr. Bùi Kim Thành	Project Owner	User acceptance testing.

Table 116 Human Resources

2. Test Environment

2.1. Test Cases

SP24SE062_RMS-Functional-Tests_v1.0.xls

2.2. Test Reports

SP24SE062_RMS-Test-Report_v1.0.xlsx

VI Release Package & User Guides

1. Deliverable Package

No.	Deliverable Item	Description
1	Project Schedule / Tracking	We use Jira for task management, Confluence for saving feature requirements, software designs, and meeting minutes, and OneDrive for storing client's materials and documents. - Jira: SP24SE062_RMS-Jira.xlsx. - OneDrive: HBC Capstone. - Confluence: Software Development (atlassian.net).
2	Source Codes	We separated our project into 3 private repositories, back-end, front-end, and mobile, respectively, on GitHub. - FE: UI. - BE: API. - DevOps: Devops.
3	Database Script(s)	Script for PLC, Blueprints, etc., to seed data (data sources).
5	Final Report Document	This is the final document of RMS.

6	Slide	A PowerPoint file to present the project information and workflows to audiences.

Table 117 Deliverable Package

2. Installation Guides

2.1. System Requirements

2.1.1. Hardware requirements

2.1.1.a) Server/PC

Name	Minimum	Recommended
Internet connection	LAN Cable, Wi-Fi (4 Mbps)	LAN Cable, Wi-Fi (8 Mbps)
Processor	Intel Pentium	Intel Core i5 or higher
Memory	2GB RAM	4GB or higher
Storage	2GB	5GB or higher

Table 118 Server/PC requirements

2.1.1.b) Mobile

Name	Minimum	Recommended
Internet connection	Wi-Fi (4 Mbps)	Wi-Fi (8 Mbps)
Processor	Exynos 3475	MediaTek MT6768
Memory	1GB RAM	2GB or higher
Storage	1GB	2GB or higher

Table 119 Mobile requirements

2.1.2. Software requirements

Software	Name/Version	Description
Operating system	Windows 10/Windows 11/Windows Server 2022	Operating system for Server/PC
NodeJS	18.12.0 or newer	For front-end
Google Chrome	Latest Google Chrome version	
Java	17	For back-end

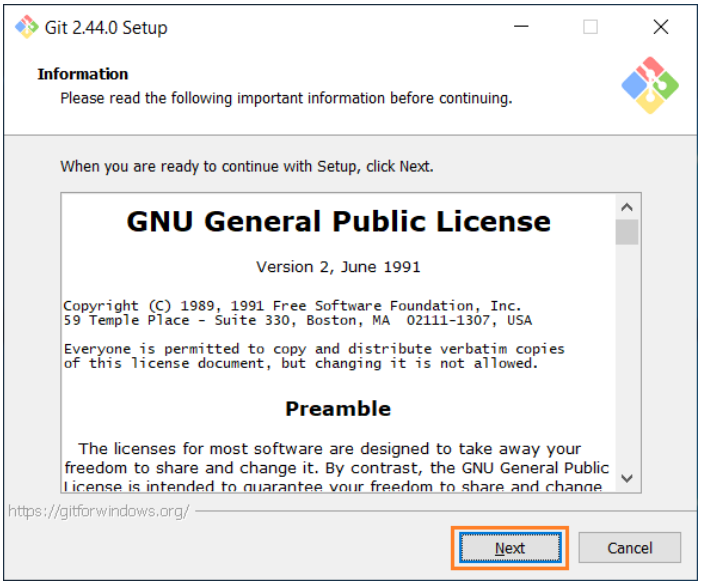
Table 120 Software requirements

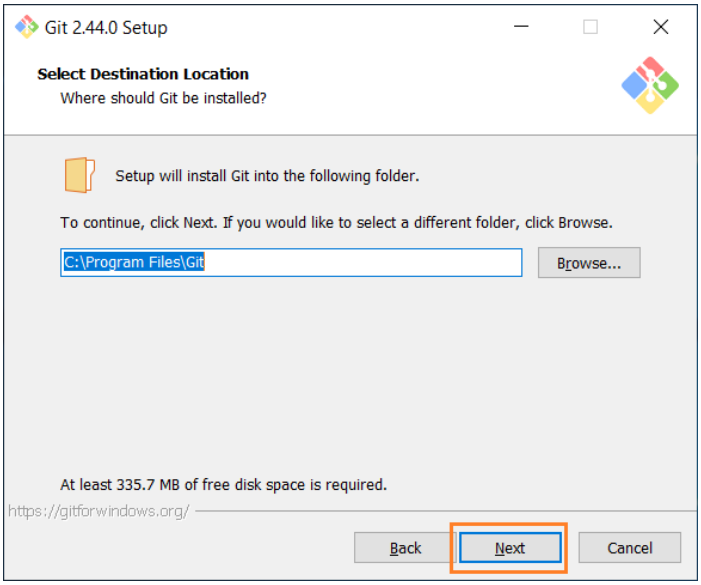
2.2. Installation Instruction

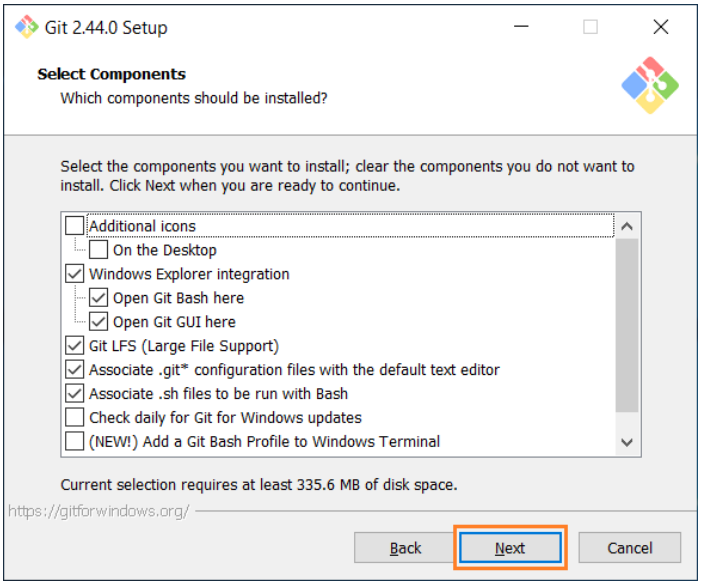
2.2.1. Development environment instruction

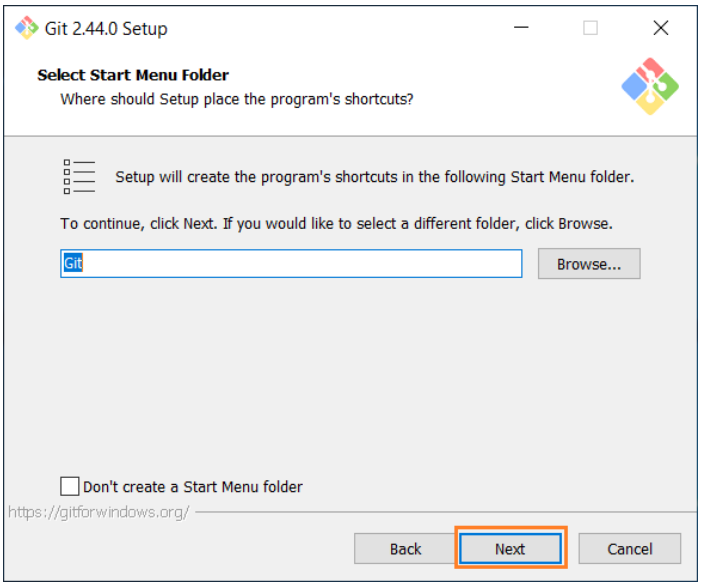
2.2.1.a) Version Control System

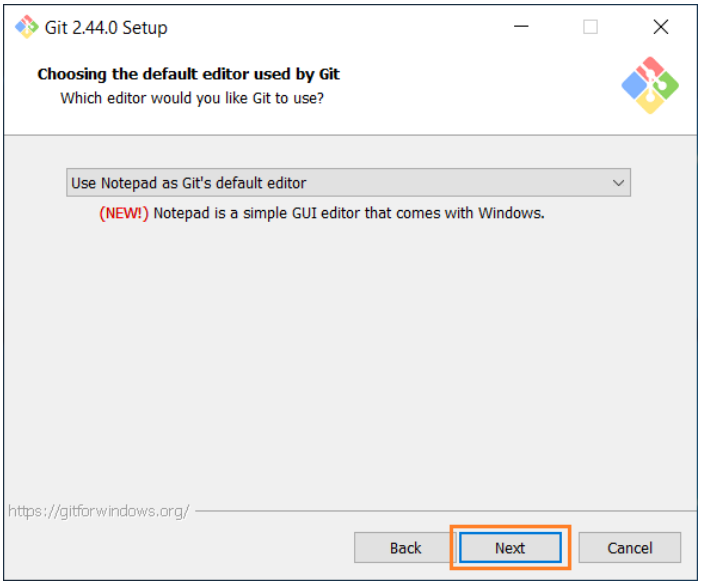
- Step 1: Visit the official Git website, and download the appropriate installer for your operating system.
- Step 2: After the download finishes, execute the installer executable file.
- Step 3: Follow the installation wizard instructions. You can generally stick with the default settings unless you have specific preferences.

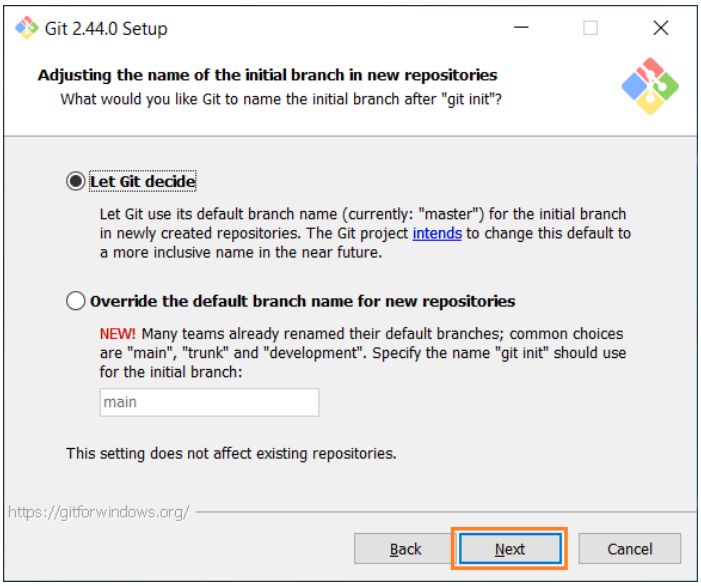


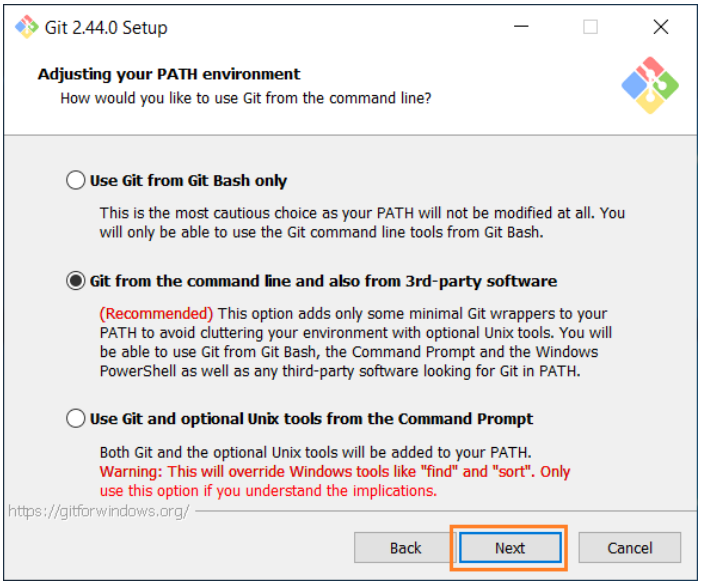


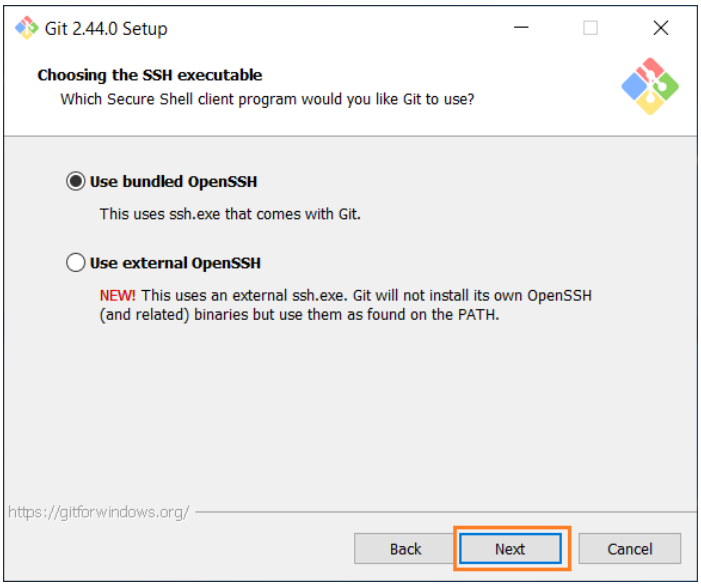

Git 2.44.0 Setup
Select Components
Which components should be installed?
Select the components you want to install; clear the components you do not want to install. Click Next when you are ready to continue.
Additional icons
On the Desktop
Windows Explorer integration
Open Git Bash here
Open Git GUI here
Git LFS (Large File Support)
Associate .git* configuration files with the default text editor
Associate .sh files to be run with Bash
Check daily for Git for Windows updates
(NEW!) Add a Git Bash Profile to Windows Terminal
Current selection requires at least 335.6 MB of disk space.
https://gitforwindows.org/
Back
Next
Cancel

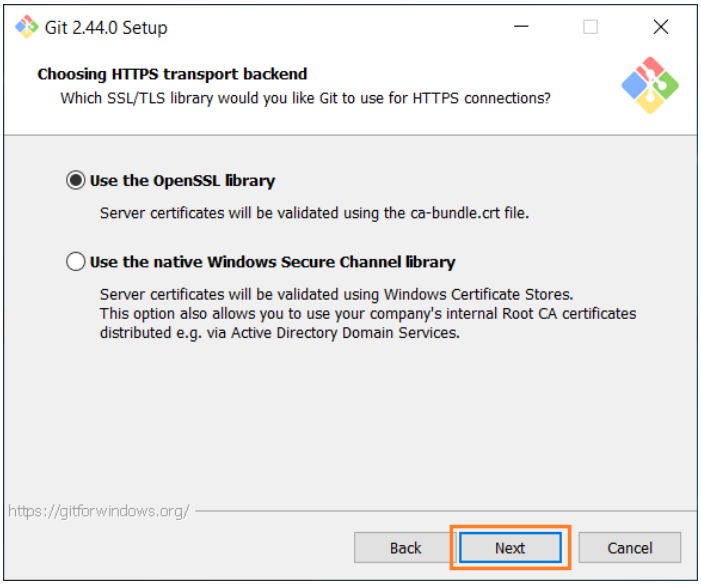

Git 2.44.0 Setup
Select Start Menu Folder
Where should Setup place the program's shortcuts?
Setup will create the program's shortcuts in the following Start Menu folder.
To continue, click Next. If you would like to select a different folder, click Browse.
Git
Browse...
Don't create a Start Menu folder
https://gitforwindows.org/
Back
Next
Cancel

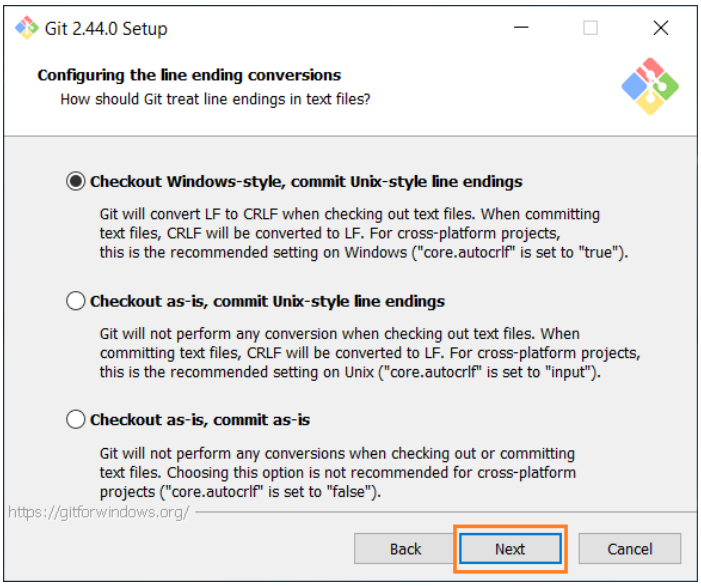

Git 2.44.0 Setup
Choosing the default editor used by Git
Which editor would you like Git to use?
Use Notepad as Git's default editor
(NEW!) Notepad is a simple GUI editor that comes with Windows.
https://gitforwindows.org/
Back
Next
Cancel

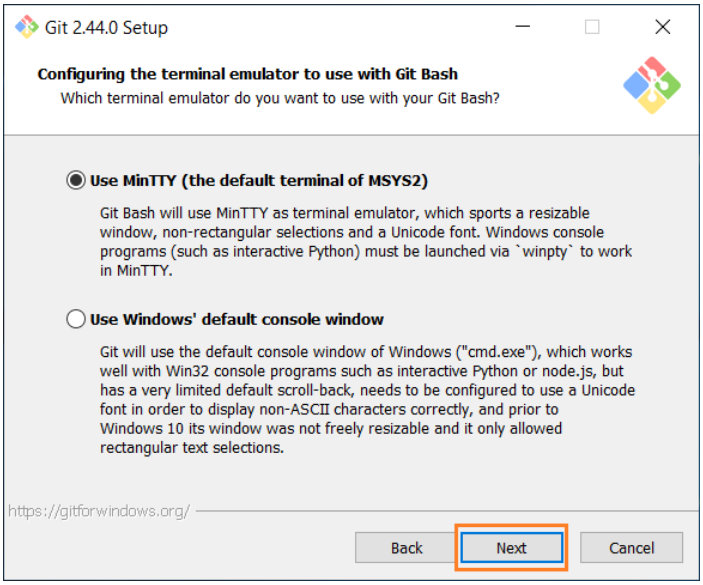

Git 2.44.0 Setup
Adjusting the name of the initial branch in new repositories
What would you like Git to name the initial branch after "git init"?
Let Git decide
Let Git use its default branch name (currently: "master") for the initial branch in newly created repositories. The Git project intends to change this default to a more inclusive name in the near future.
Override the default branch name for new repositories
NEW! Many teams already renamed their default branches; common choices are "main", "trunk" and "development". Specify the name "git init" should use for the initial branch:
main
This setting does not affect existing repositories.
https://gitforwindows.org/
Back
Next
Cancel

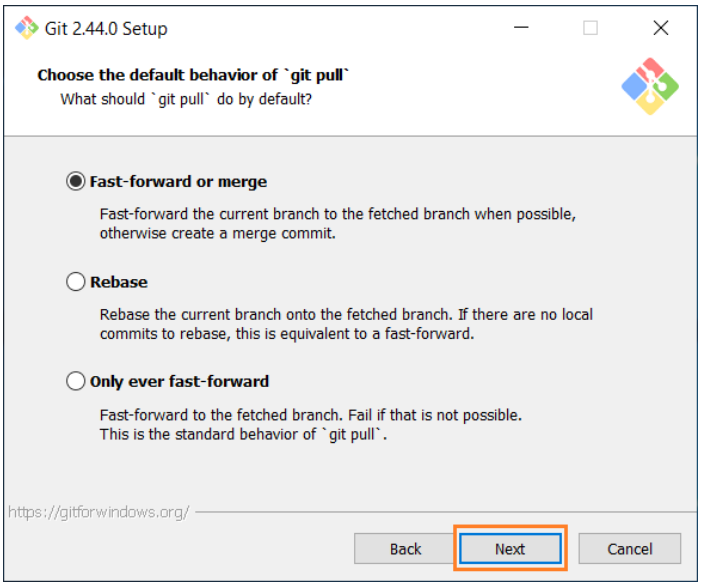

Git 2.44.0 Setup
Adjusting your PATH environment
How would you like to use Git from the command line?
Use Git from Git Bash only
This is the most cautious choice as your PATH will not be modified at all. You will only be able to use the Git command line tools from Git Bash.
Git from the command line and also from 3rd-party software
(Recommended) This option adds only some minimal Git wrappers to your PATH to avoid cluttering your environment with optional Unix tools. You will be able to use Git from Git Bash, the Command Prompt and the Windows PowerShell as well as any third-party software looking for Git in PATH.
Use Git and optional Unix tools from the Command Prompt
Both Git and the optional Unix tools will be added to your PATH.
Warning: This will override Windows tools like "find" and "sort". Only use this option if you understand the implications.
https://gitforwindows.org/
Back
Next
Cancel

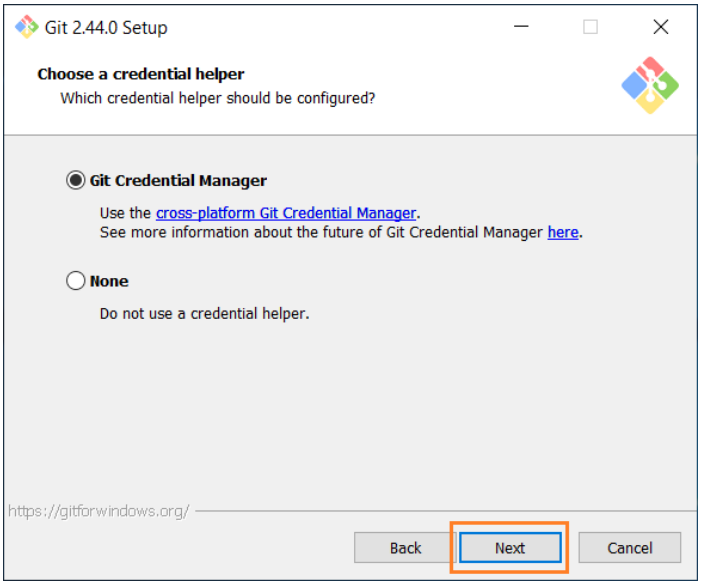

Git 2.44.0 Setup
Choosing the SSH executable
Which Secure Shell client program would you like Git to use?
Use bundled OpenSSH
This uses ssh.exe that comes with Git.
Use external OpenSSH
NEW! This uses an external ssh.exe. Git will not install its own OpenSSH (and related) binaries but use them as found on the PATH.
https://gitforwindows.org/
Back
Next
Cancel

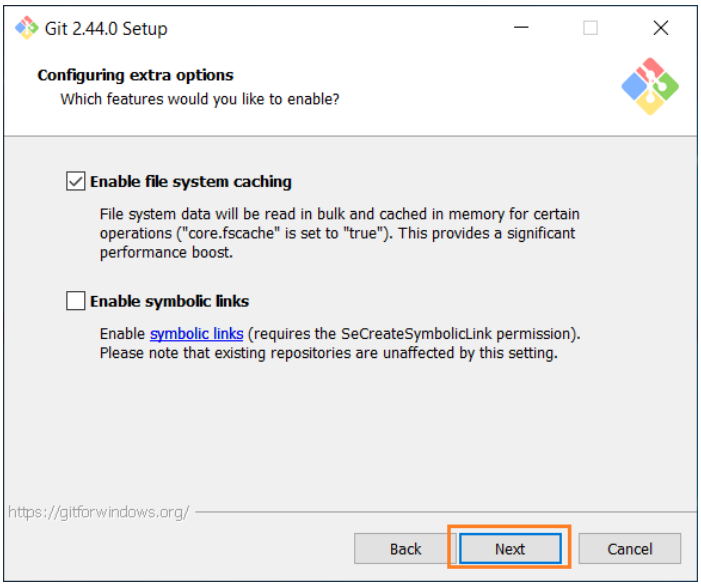

Git 2.44.0 Setup
Choosing HTTPS transport backend
Which SSL/TLS library would you like Git to use for HTTPS connections?
Use the OpenSSL library
Server certificates will be validated using the ca-bundle.crt file.
Use the native Windows Secure Channel library
Server certificates will be validated using Windows Certificate Stores.
This option also allows you to use your company's internal Root CA certificates
distributed e.g. via Active Directory Domain Services.
https://gitforwindows.org/
Back
Next
Cancel

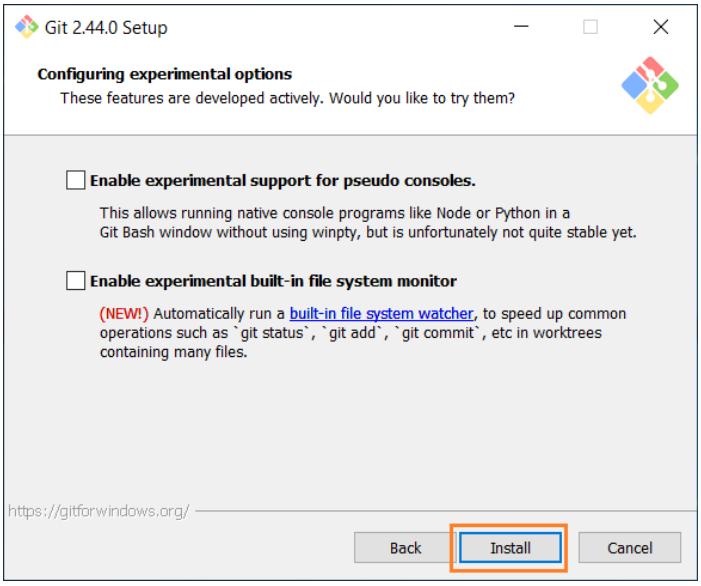

Git 2.44.0 Setup
Configuring the line ending conversions
How should Git treat line endings in text files?
Checkout Windows-style, commit Unix-style line endings
Git will convert LF to CRLF when checking out text files. When committing
text files, CRLF will be converted to LF. For cross-platform projects,
this is the recommended setting on Windows ("core.autocrlf" is set to "true").
Checkout as-is, commit Unix-style line endings
Git will not perform any conversion when checking out text files. When
committing text files, CRLF will be converted to LF. For cross-platform projects,
this is the recommended setting on Unix ("core.autocrlf" is set to "input").
Checkout as-is, commit as-is
Git will not perform any conversions when checking out or committing
text files. Choosing this option is not recommended for cross-platform
projects ("core.autocrlf" is set to "false").
https://gitforwindows.org/
Back
Next
Cancel


Git 2.44.0 Setup
Configuring the terminal emulator to use with Git Bash
Which terminal emulator do you want to use with your Git Bash?
Use MinTTY (the default terminal of MSYS2)
Git Bash will use MinTTY as terminal emulator, which sports a resizable window, non-rectangular selections and a Unicode font. Windows console programs (such as interactive Python) must be launched via `winpty` to work in MinTTY.
Use Windows' default console window
Git will use the default console window of Windows ("cmd.exe"), which works well with Win32 console programs such as interactive Python or node.js, but has a very limited default scroll-back, needs to be configured to use a Unicode font in order to display non-ASCII characters correctly, and prior to Windows 10 its window was not freely resizable and it only allowed rectangular text selections.
https://gitforwindows.org/
Back
Next
Cancel


Git 2.44.0 Setup
Choose the default behavior of `git pull`
What should `git pull` do by default?
Fast-forward or merge
Fast-forward the current branch to the fetched branch when possible, otherwise create a merge commit.
Rebase
Rebase the current branch onto the fetched branch. If there are no local commits to rebase, this is equivalent to a fast-forward.
Only ever fast-forward
Fast-forward to the fetched branch. Fail if that is not possible. This is the standard behavior of `git pull`.
https://gitforwindows.org/
Back
Next
Cancel


Git 2.44.0 Setup
Choose a credential helper
Which credential helper should be configured?
Git Credential Manager
Use the cross-platform Git Credential Manager.
See more information about the future of Git Credential Manager here.
None
Do not use a credential helper.
https://gitforwindows.org/
Back
Next
Cancel

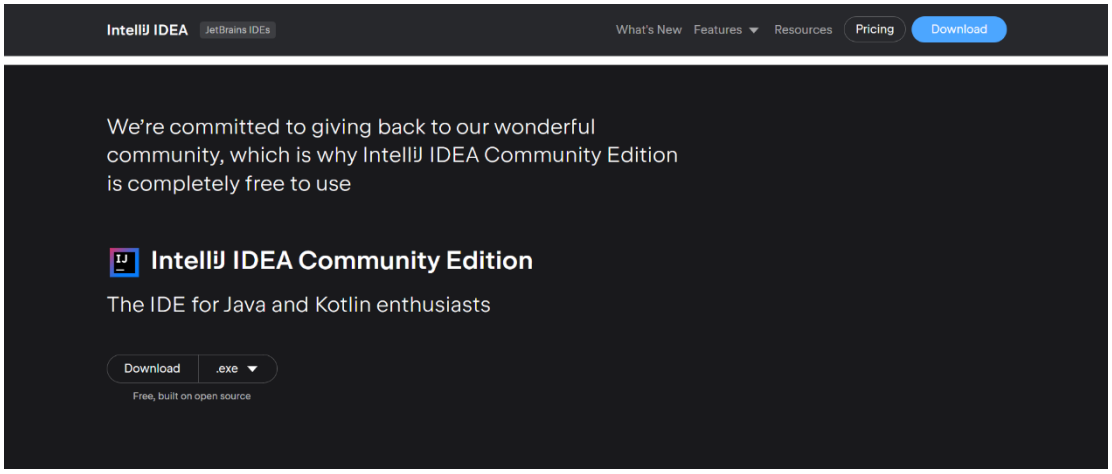

Git 2.44.0 Setup
Configuring extra options
Which features would you like to enable?
Enable file system caching
File system data will be read in bulk and cached in memory for certain operations ("core.fscache" is set to "true"). This provides a significant performance boost.
Enable symbolic links
Enable symbolic links (requires the SeCreateSymbolicLink permission).
Please note that existing repositories are unaffected by this setting.
https://gitforwindows.org/
Back
Next
Cancel



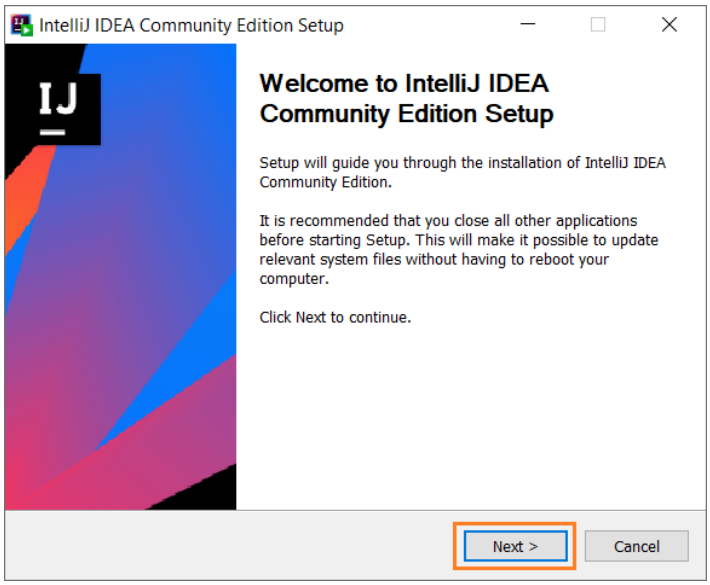
- Step 4: After installation, open a terminal and type **git --version** to confirm that Git has been installed successfully.

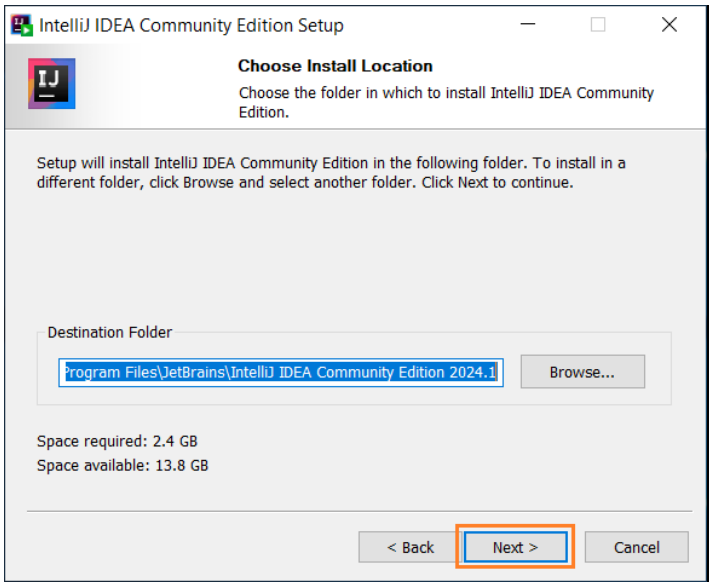
2.2.1.b) Back-end

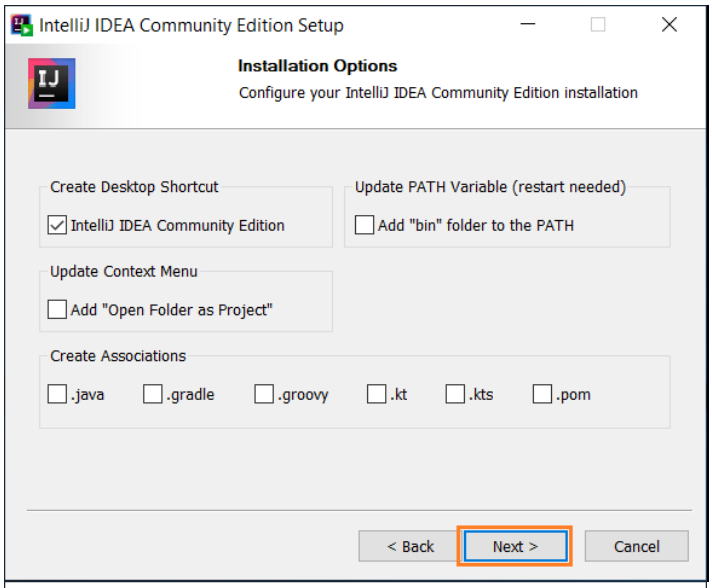
- Install the IntelliJ IDEA Community Edition.
    - Step 1: Go to the IntelliJ website to download the IDE

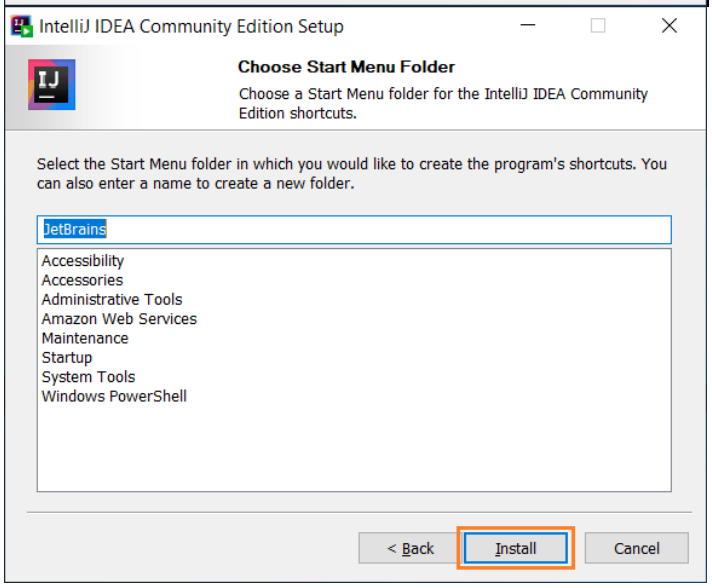


    - Step 2: From your Downloads folder, double-click the bootstrapper ideaIC.exe to start the installation.
    - Step 3: Choose Yes if you receive a User Account Control notice.
    - Step 4: Follow the installation wizard instructions. You might be asked to choose installation options such as location, creating desktop shortcuts, etc.


IntelliJ IDEA Community Edition Setup
IJ
Welcome to IntelliJ IDEA Community Edition Setup
Setup will guide you through the installation of IntelliJ IDEA Community Edition.
It is recommended that you close all other applications before starting Setup. This will make it possible to update relevant system files without having to reboot your computer.
Click Next to continue.
Next >
Cancel


IntelliJ IDEA Community Edition Setup
IJ
Choose Install Location
Choose the folder in which to install IntelliJ IDEA Community Edition.
Setup will install IntelliJ IDEA Community Edition in the following folder. To install in a different folder, click Browse and select another folder. Click Next to continue.
Destination Folder
rogram Files\JetBrains\IntelliJ IDEA Community Edition 2024.1
Browse...
Space required: 2.4 GB
Space available: 13.8 GB
< Back
Next >
Cancel

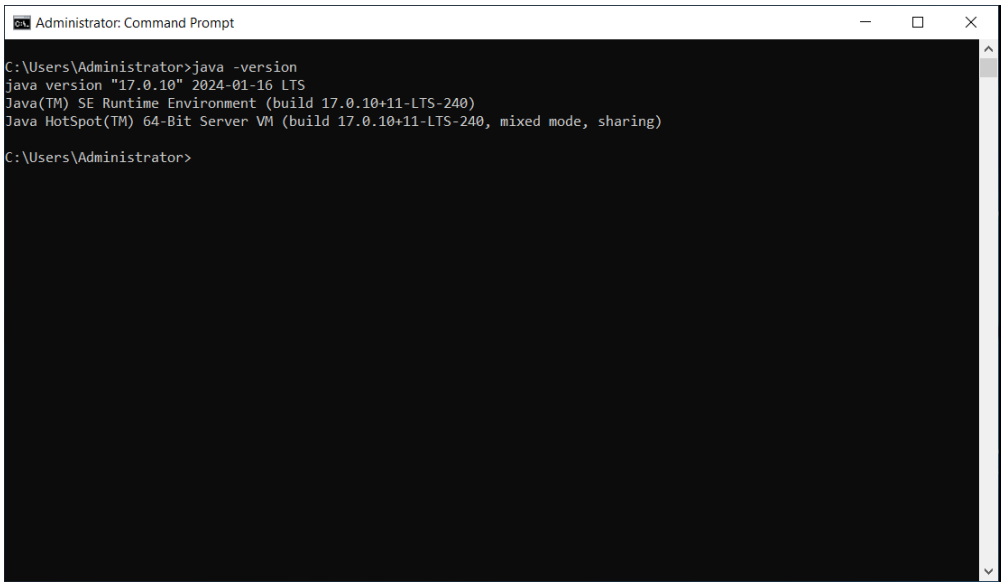




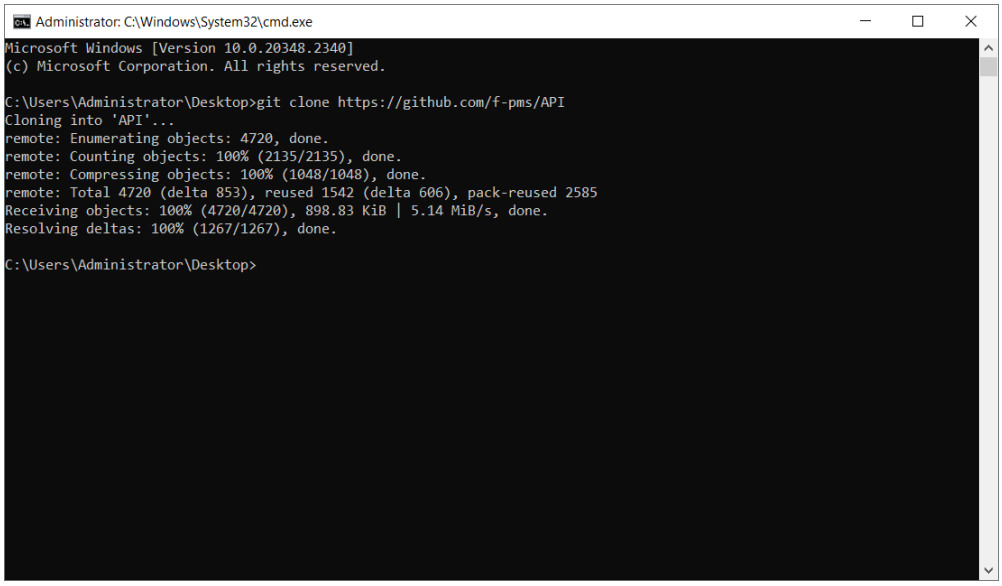
- Install Java SDK 17
    - Step 1: Go to the official Oracle website to download Java 17.
    - Step 2: Run the installer executable file once the download is complete.
    - Step 3: Follow the installation wizard instructions. You might be asked to choose the installation location and accept license agreements.


Java(TM) SE Development Kit 17.0.10 (64-bit) - Setup
Java
ORACLE
Welcome to the Installation Wizard for Java SE Development Kit 17.0.10
This wizard will guide you through the installation process for the Java SE Development Kit 17.0.10.
Next >
Cancel
Java(TM) SE Development Kit 17.0.10 (64-bit) - Destination Folder
Java
ORACLE
This will install the Java(TM) SE Development Kit 17.0.10 (64-bit), which requires 420MB on your hard drive. Click the "Change" button to change the installation folder.
Install Java(TM) SE Development Kit 17.0.10 (64-bit) to:
C:\Program Files\Java\jdk-17\
Change...
Back
Next
Cancel

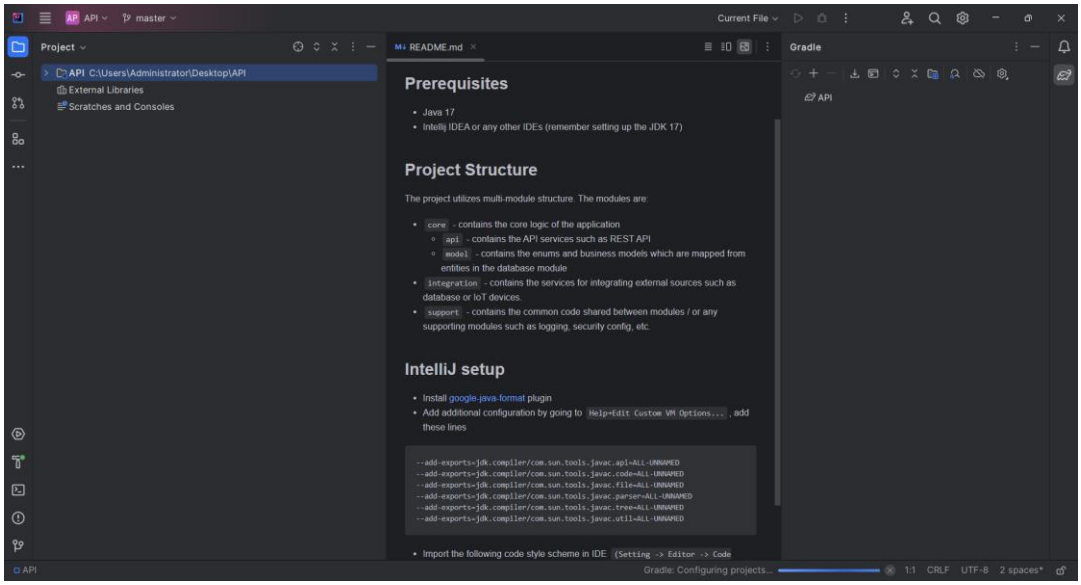
- Step 4: Open a terminal and execute the command **java -version** to verify that Java is successfully installed.



- Setup the back-end codebase
  - Step 1: Open a terminal, and type **git clone https://github.com/f-pms/API**

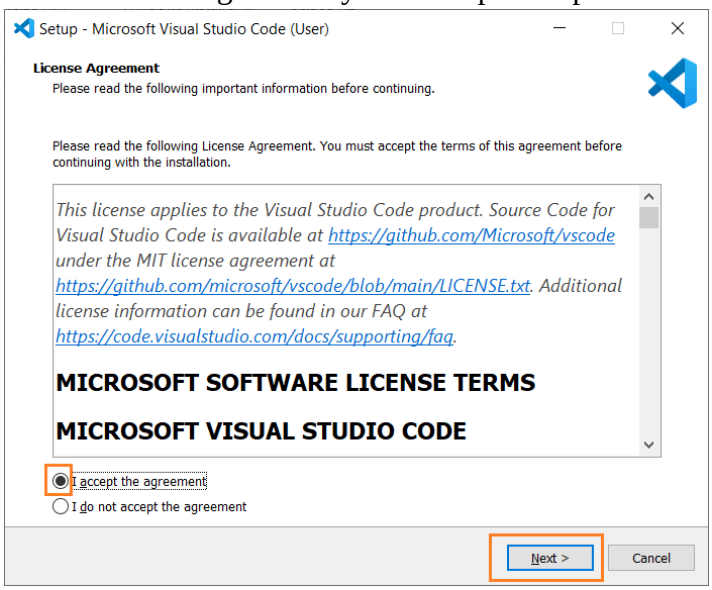


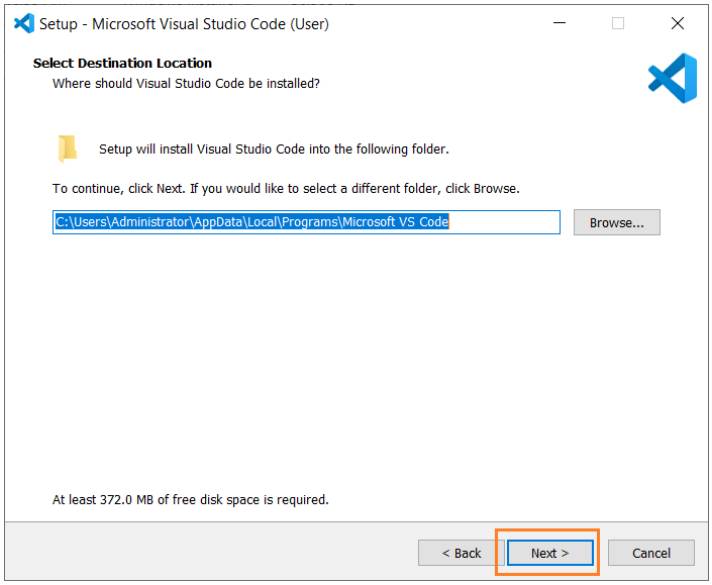
- Step 2: Launch IntelliJ and open the codebase

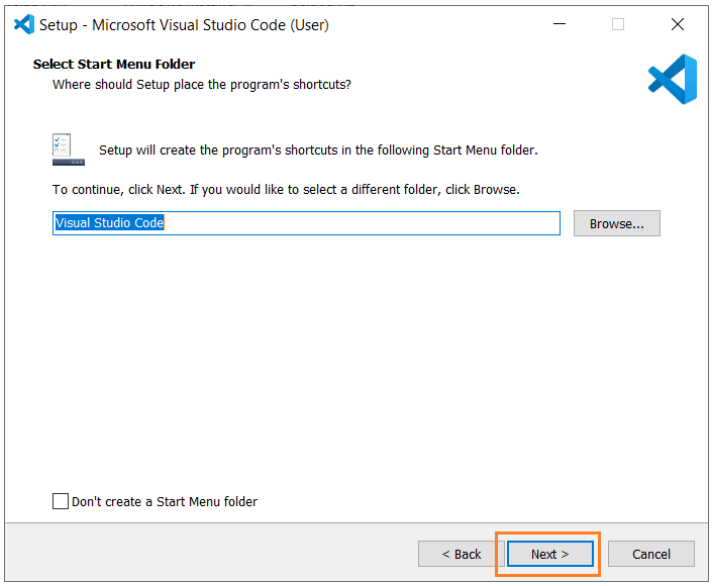


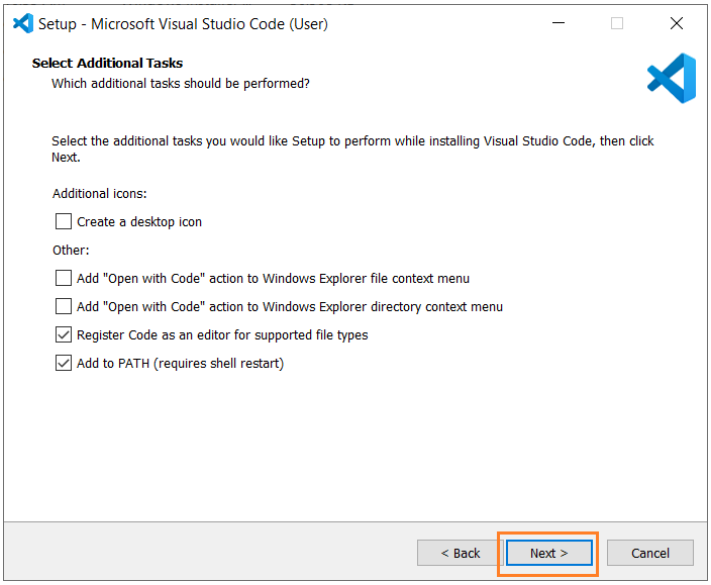
2.2.1.c) Front-end

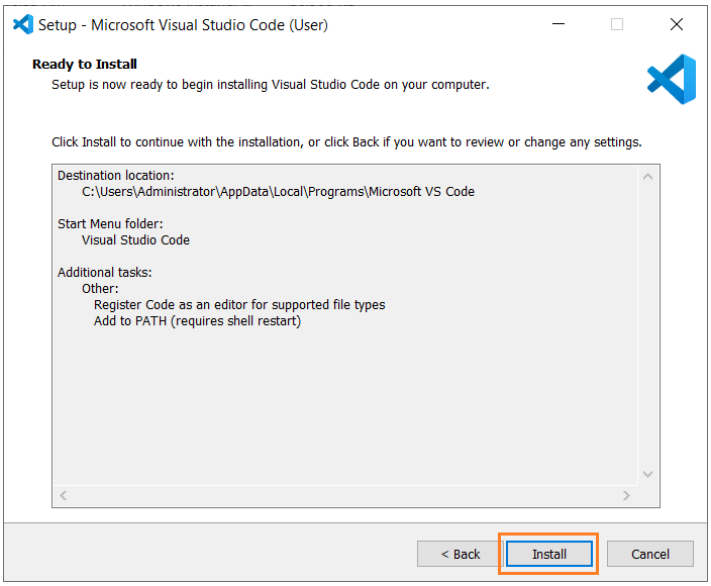
- Install VSCode IDE
  - Step 1: Go to the VSCode website to download the VSCode installer.
  - Step 2: Run the installer executable file once the download is complete.
  - Step 3: Follow the installation wizard instructions. You can generally stick with the default settings unless you have specific preferences.



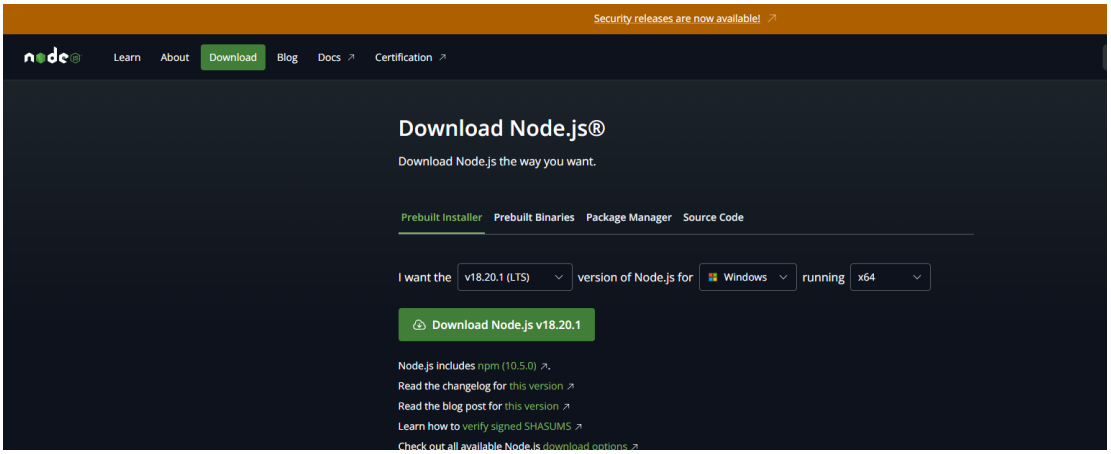

Setup - Microsoft Visual Studio Code (User)
Select Destination Location
Where should Visual Studio Code be installed?
Setup will install Visual Studio Code into the following folder.
To continue, click Next. If you would like to select a different folder, click Browse.
C:\Users\Administrator\AppData\Local\Programs\Microsoft VS Code
Browse...
At least 372.0 MB of free disk space is required.
< Back
Next >
Cancel


Setup - Microsoft Visual Studio Code (User)
Select Start Menu Folder
Where should Setup place the program's shortcuts?
Setup will create the program's shortcuts in the following Start Menu folder.
To continue, click Next. If you would like to select a different folder, click Browse.
Visual Studio Code
Browse...
Don't create a Start Menu folder
< Back
Next >
Cancel

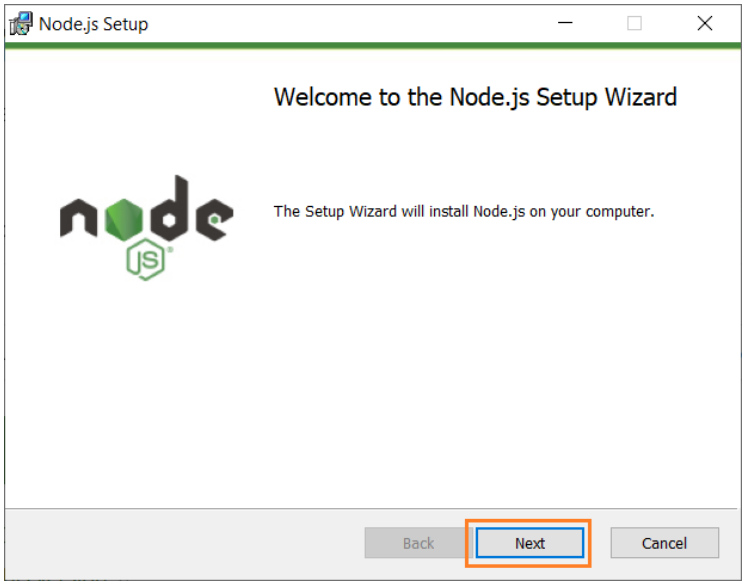


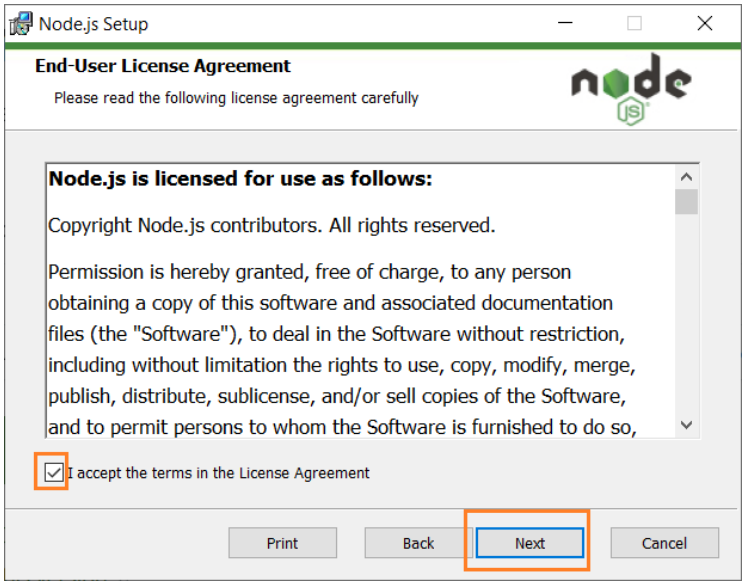


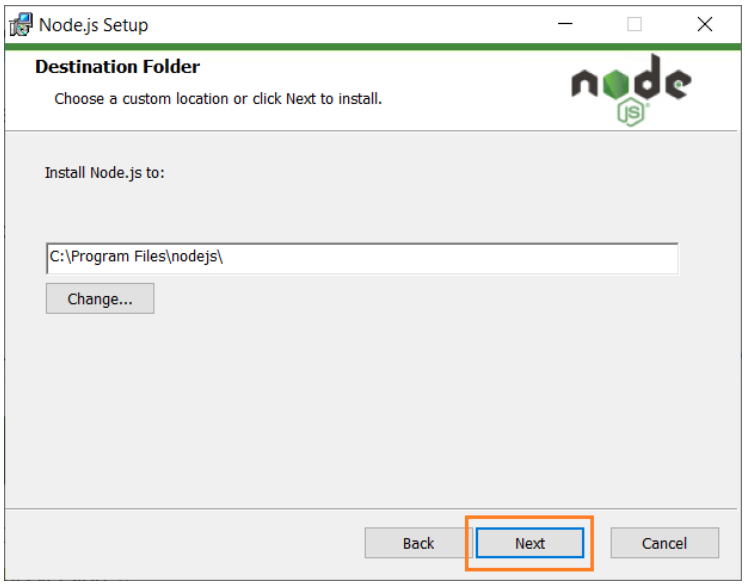
- Install Node.js v18:
  - Step 1: Go to the Node.js website to download Node.js installer.

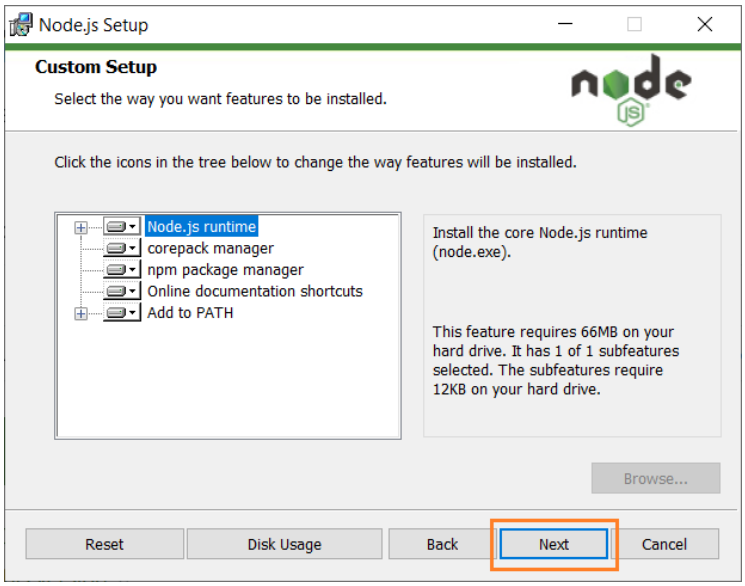


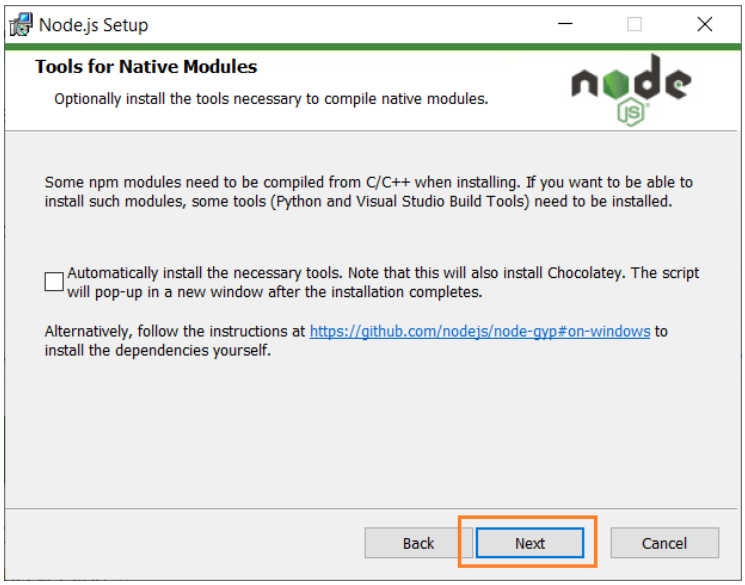
- Step 2: Once the download is complete, run the installer executable file.
- Step 3: Follow the installation wizard instructions. You can generally stick with the default settings unless you have specific preferences.

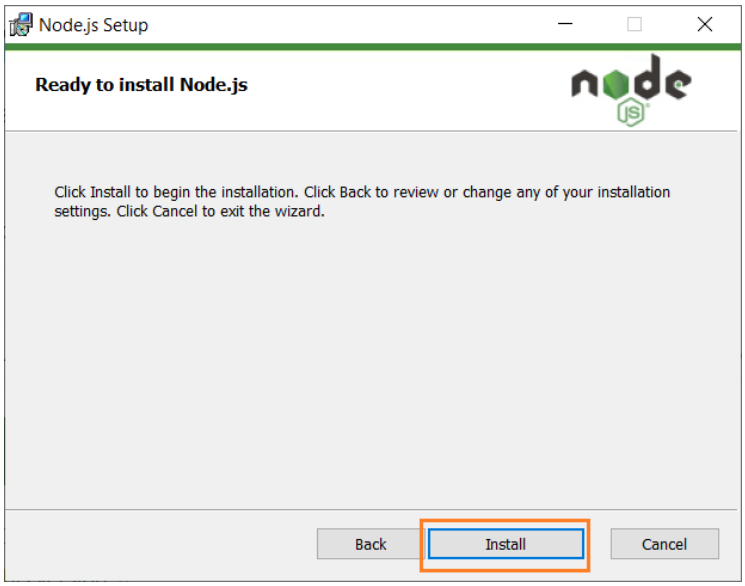




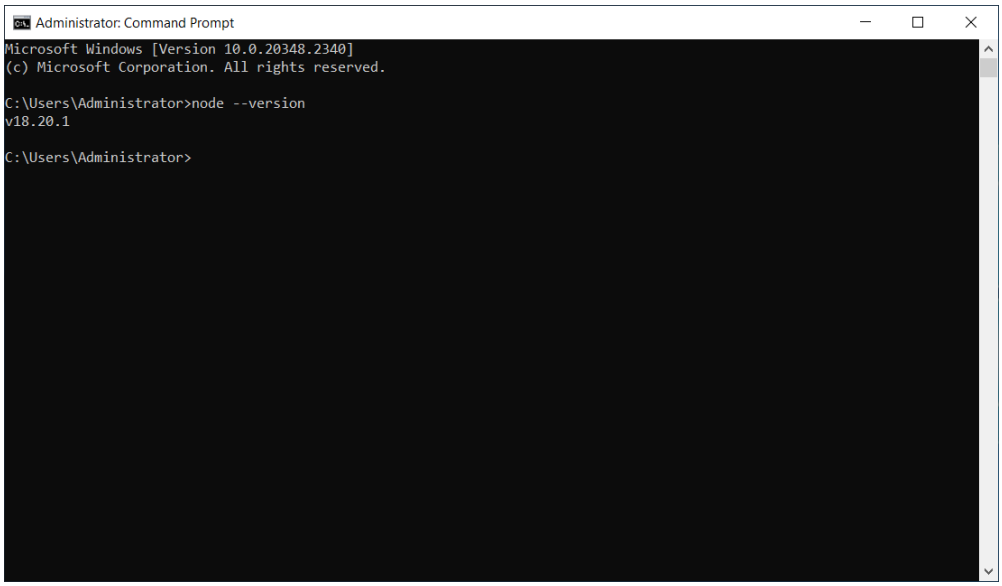

Node.js Setup
Destination Folder
Choose a custom location or click Next to install.
Install Node.js to:
C:\Program Files\nodejs\
Change...
Back
Next
Cancel


Node.js Setup
Custom Setup
Select the way you want features to be installed.
Click the icons in the tree below to change the way features will be installed.
Node.js runtime
corepack manager
npm package manager
Online documentation shortcuts
Add to PATH
Install the core Node.js runtime (node.exe).
This feature requires 66MB on your hard drive. It has 1 of 1 subfeatures selected. The subfeatures require 12KB on your hard drive.
Browse...
Reset
Disk Usage
Back
Next
Cancel

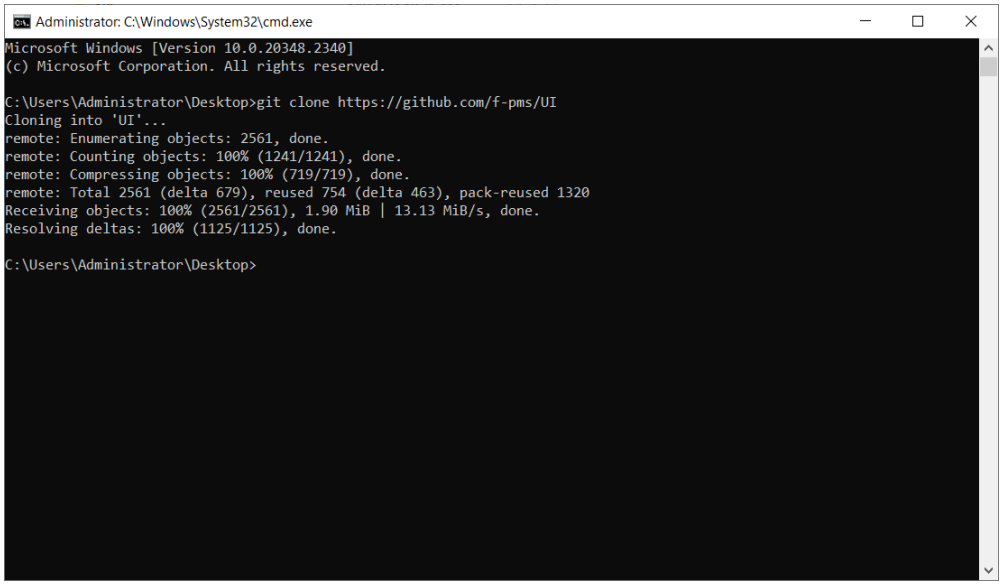

Node.js Setup
Tools for Native Modules
Optionally install the tools necessary to compile native modules.
Some npm modules need to be compiled from C/C++ when installing. If you want to be able to install such modules, some tools (Python and Visual Studio Build Tools) need to be installed.
Automatically install the necessary tools. Note that this will also install Chocolatey. The script will pop-up in a new window after the installation completes.
Alternatively, follow the instructions at https://github.com/nodejs/node-gyp#on-windows to install the dependencies yourself.
Back
Next
Cancel


Node.js Setup
Ready to install Node.js
Click Install to begin the installation. Click Back to review or change any of your installation settings. Click Cancel to exit the wizard.
Back
Install
Cancel

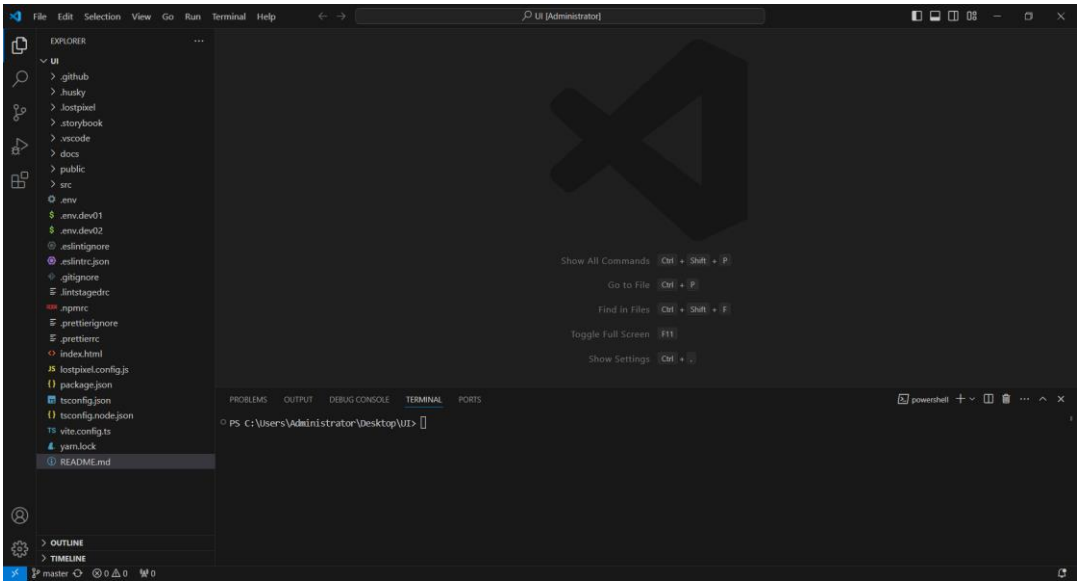
Step 4: Open a terminal, type **node --version** to verify that Node.js is installed successfully.



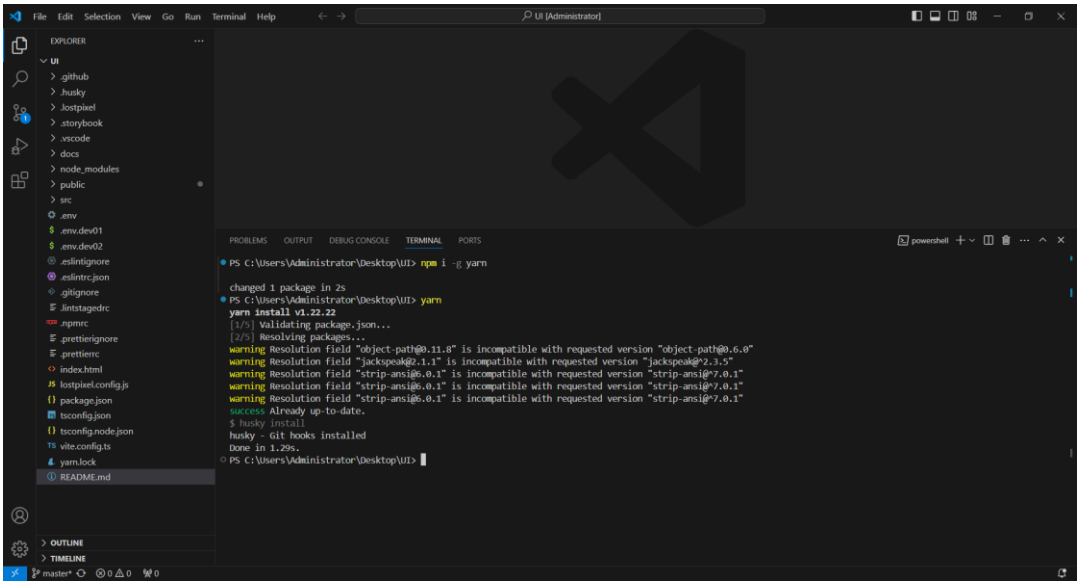
- Setup the front-end codebase:
    - Step 1: Open a terminal and type **git clone https://github.com/f-pms/UI**.



- Step 2: Launch VSCode IDE to open the codebase.



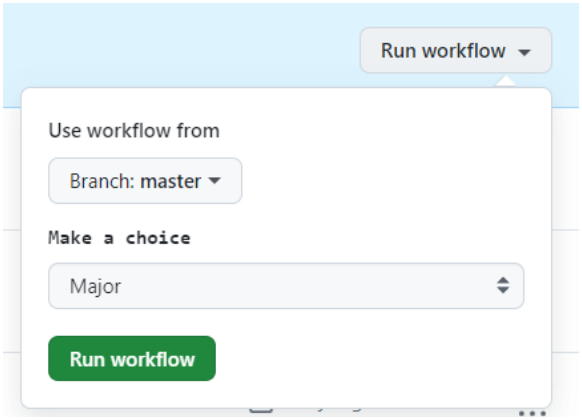
- Step 3: Install the package manager yarn using the **npm i -g yarn** command.
- Step 4: Install all packages using the **yarn** command.



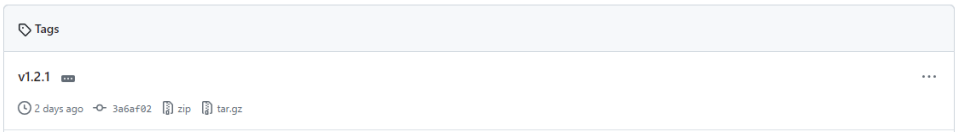
2.2.1.d) Deploy New Versions to The Development Environment

- Make new changes:
  - Step 1: Create a branch with the command **git checkout -b <branch-name>**.
  - Step 2: Make your desired changes to the codebase.
  - Step 3: Once you've made changes, stage the files you want to commit by inputting **git add .** command.
  - Step 4: Commit the changes with a descriptive commit message **git commit -m "Your descriptive commit message here"**.
  - Step 5: Push changes **git push origin <branch-name>**.
- Merge new changes:

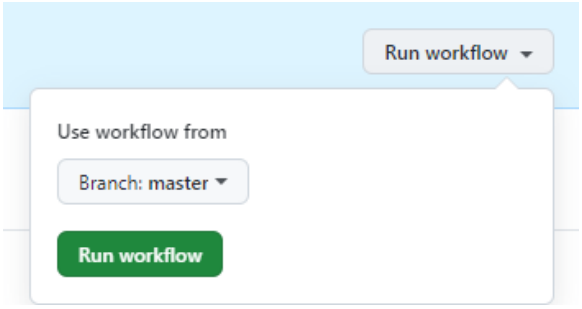
    - Step 1: Navigate to the “Pull Requests” tab and click “New pull request”. Select your branch as the “compare” branch and the master branch as the “base” branch.
    - Step 2: Click the “Squash and merge” button on GitHub.
- Release new changes:
    - Step 1: Use API job or UI job to create a new release
    - Step 2: Choose Major/Minor/Patch based on semantic versioning. Click “Run workflow”.



    - Step 3: Get the latest tag in the API repository or UI repository.



    - Step 4: Update the manifest section within the release.yml file, ensuring to include the specified tag mentioned above.
    - Step 5: Use the Update Deployment Versions job to apply the latest version.

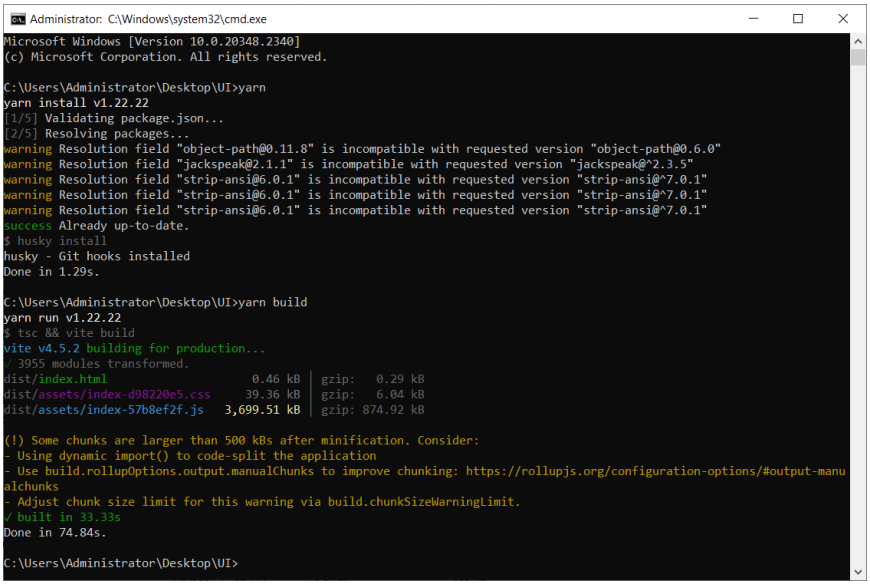


2.2.2. Production environment instruction

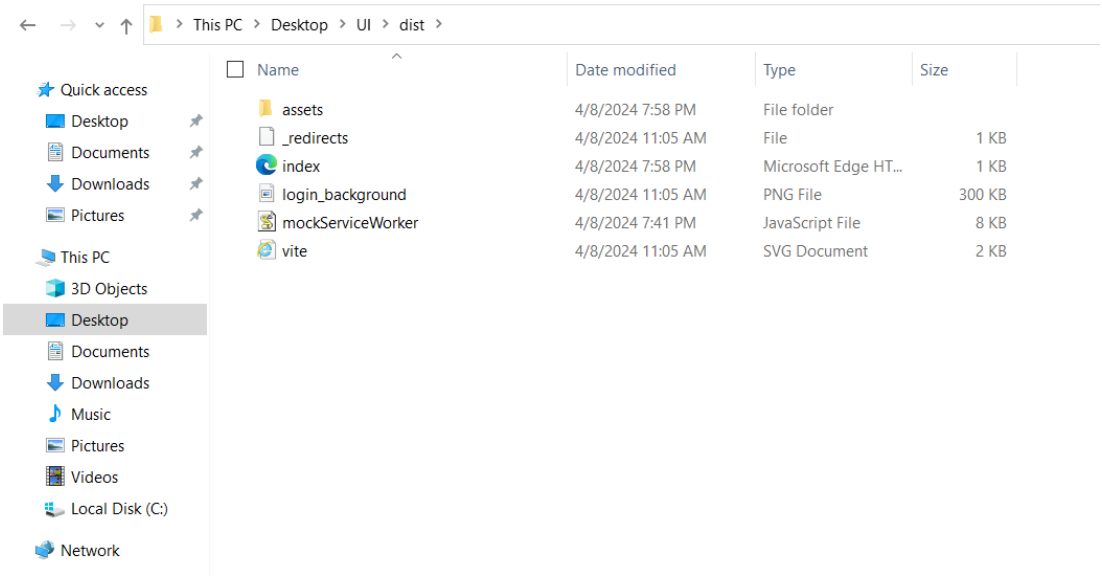
2.2.2.a) Build Front-end Codebase

- Step 1: Open a terminal at the front-end codebase.
- Step 2: Run **yarn** in the terminal to install all packages.

- Step 3: Run **yarn build** in the terminal to build front-end codebase.

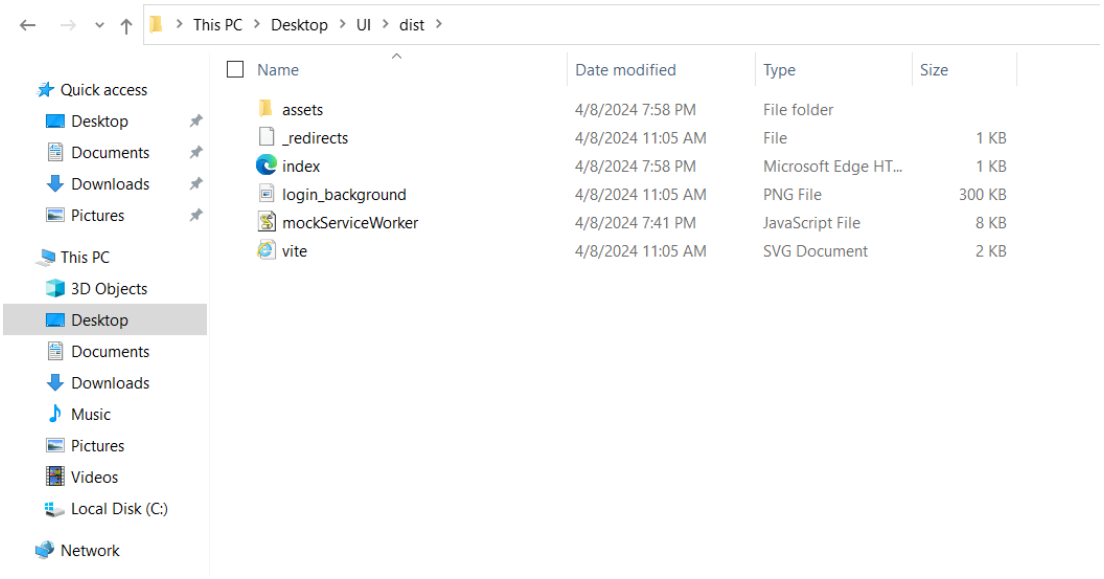


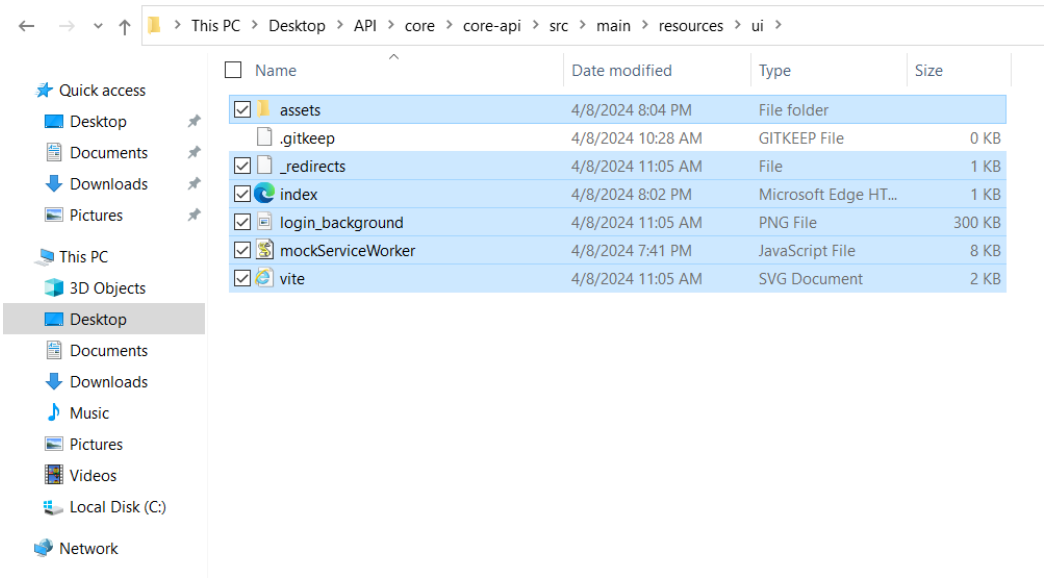
After building, you have a **dist** folder.



2.2.2.b) Build Back-end Codebase

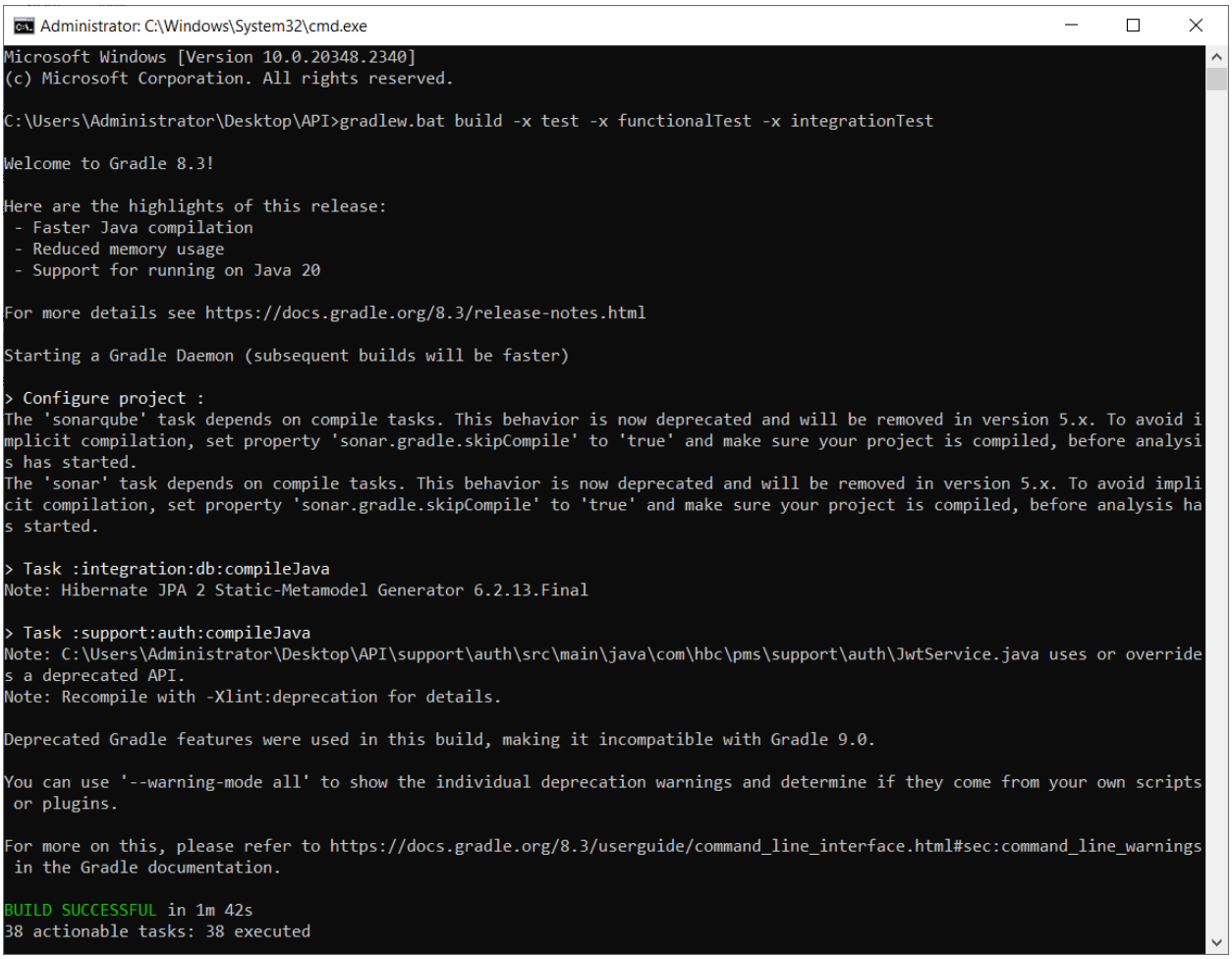
- Step 1: Copy all files in the front-end **dist** folder to the **ui** folder in the back-end **core\core-api\src\main\resources\ui** path



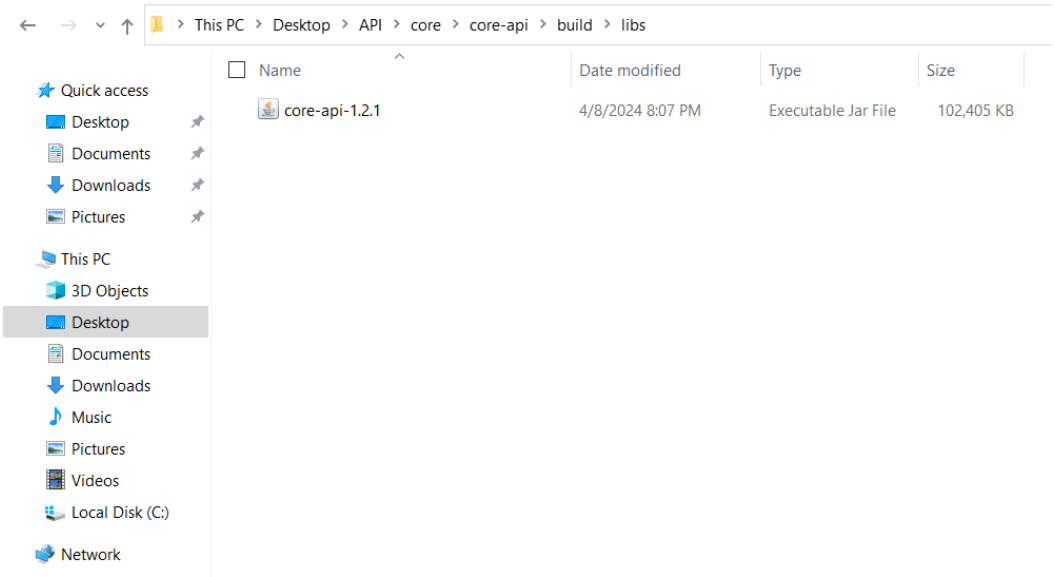


- Step 2: Open a terminal at back-end codebase

- Step 3: Run **gradlew.bat build -x test -x functionalTest -x integrationTest** to build back-end codebase



- Step 4: After building, you have a jar file in the back-end **core\core-api\build\libs** folder.



2.2.2.c) Run RMS Application

- Step 1: Configure environment variables to configure the application setting.
- Step 2: Open a terminal at the jar file path and type **java -jar <jar-name>.jar** with **<jar-name>** as the name of the jar file. For example, **java -jar RMS.jar**.

3. User Manual

3.1. Overview

Workflow	Description
Monitor and modify Tags	As a supervisor / an administrator, after navigating to the monitoring page, user can view the real-time production parameters on each station's plant. As an administrator, after clicking a button at the monitoring page, user can modify the address of every figure.
Create alarms and receive notifications	As an administrator, after navigating to the alarm page, the user can create predefined and custom alarms. As a supervisor / an administrator, after creating a predefined and a custom alarm, user can receive notifications based on the actions defined in the alarms.
View statistical reports	As a supervisor / an administrator, after navigating to the search report page, user can filter and view a specific report along with its associated charts. As an administrator, after viewing a specific report, the user can download the report as an Excel file. As a supervisor / an administrator, after navigating to the statistics page, user can view the statistical charts and figures by adjusting the date range.
Manage users	As an administrator, user can perform management operations including creating new users, delete existing users, and update user information.

Table 121 User Manual overview

3.2. Monitor and Modify Tags

3.2.1. Monitoring

- Description: Supervisors visually view the real-time production parameters on each station's plant layout.

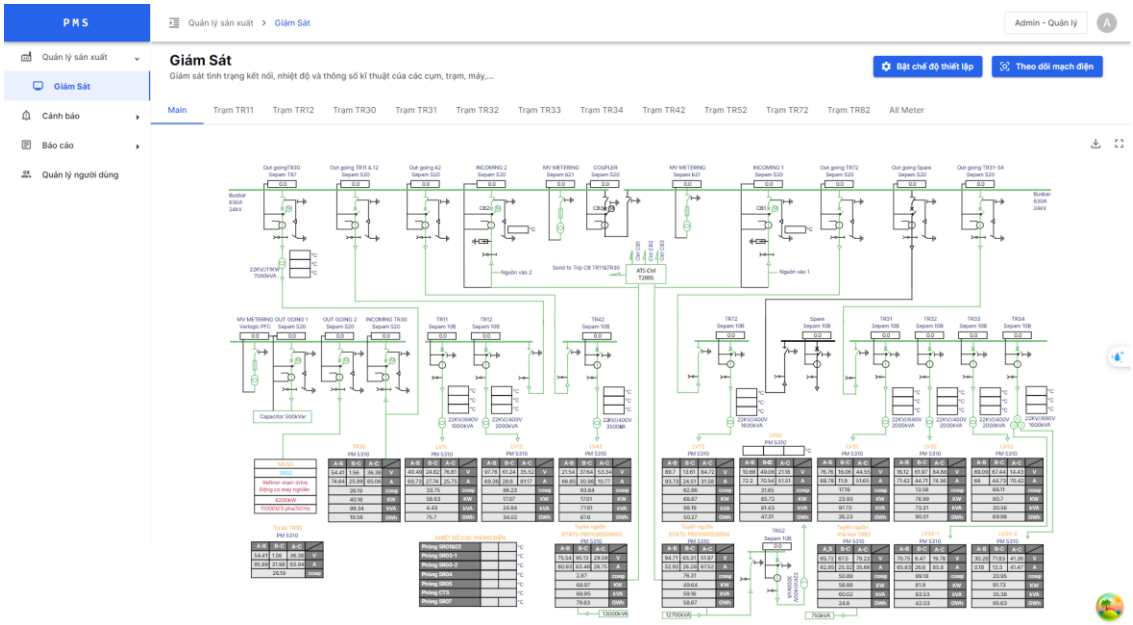


Figure 68 Supervisors switch between station tabs (e.g., "Main", "Trạm TR11") to view parameters

3.2.2. Modify Tags

- Description: Administrators edit the information of the associated PLC Tag.

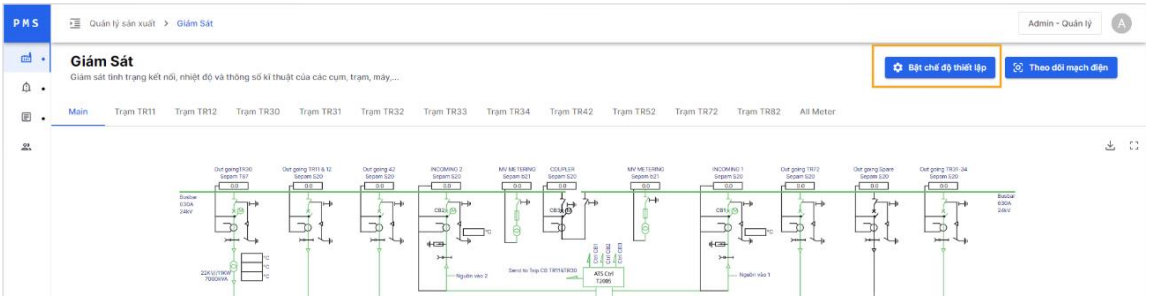


Figure 69 Administrators click "Bật chế độ thiết lập" button to switch to edit mode

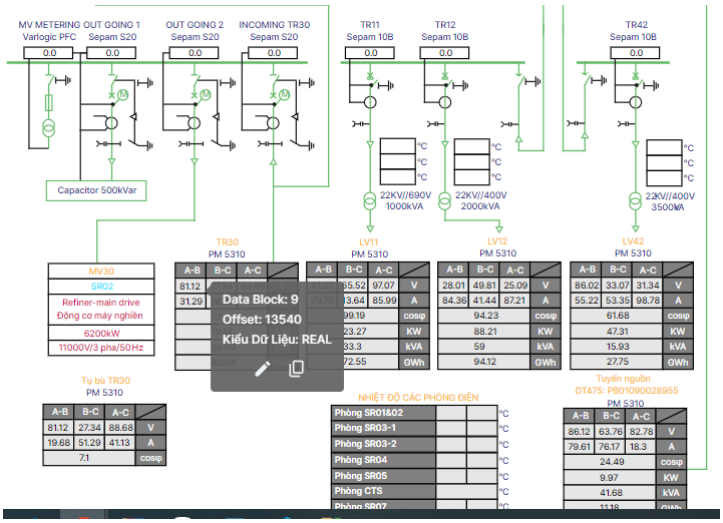


Figure 70 Administrators hover over the production parameter that needs to be updated

Figure 71 Administrators define the production parameter by entering the required information for its associated PLC Tag

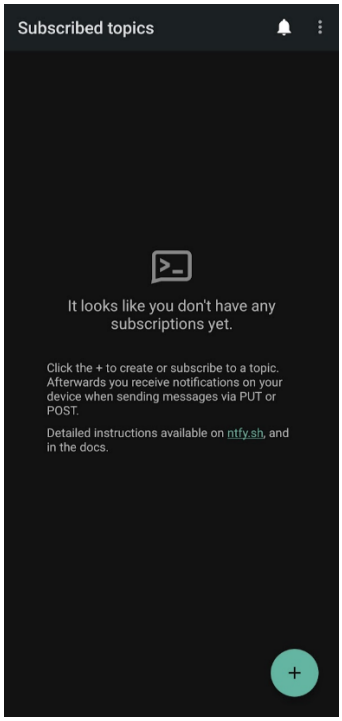
3.3. Create Alarms and Receive Notifications

3.3.1. Setup for Push Notification

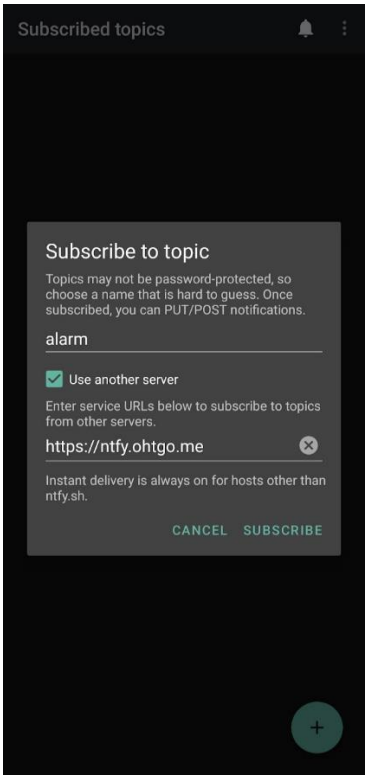
The staff at the factory, who need to receive Alarm via mobile's push notification, must follow these setup instructions:

1. Download the ntfy application
    a. Android: https://play.google.com/store/apps/details?id=io.heckel.ntfy
    b. iOS: https://apps.apple.com/us/app/ntfy/id1625396347
2. Subscribe to the alarm topic
    a. Open the installed app

b. Click **Add** button



c. Fill the ntfy server URL and topic name
topic: alarm
server: https://ntfy.ohtgo.me



d. Click **Subscribe** button

3.3.2. Create condition's alarm

- Description: Administrators create alarms based on either predefined PLC Tags or user-defined PLC Tags displayed on monitoring screens.

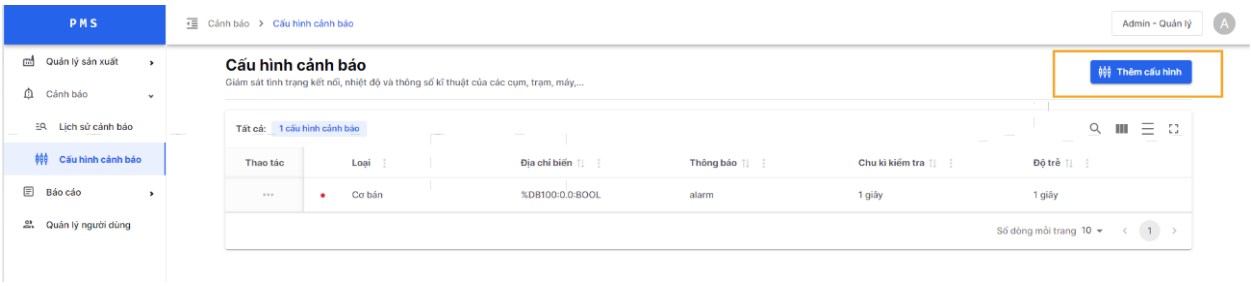


Figure 72 Administrators click "Thêm cấu hình" button to create new alarm configuration

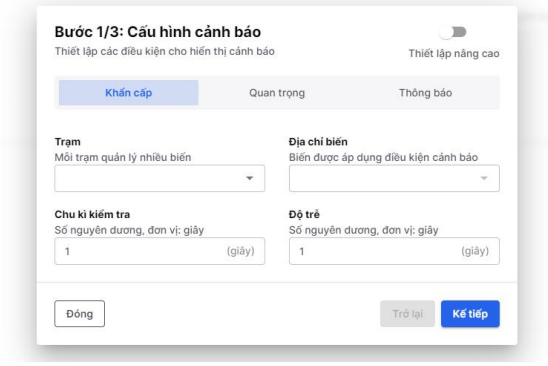


Figure 73 Administrators create alarms based on predefined PLC Tag

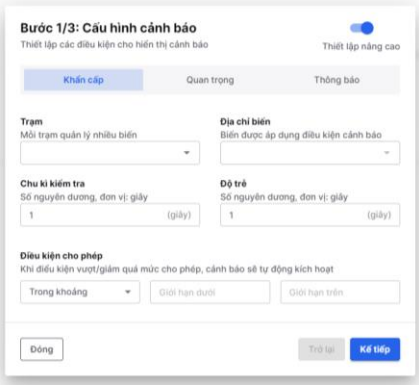


Figure 74 Administrators create alarms based on PLC Tags displayed on monitoring screens

3.3.3. Create actions' alarm

- Description: Administrators configure alarms for actions, specifying a message and notification methods.

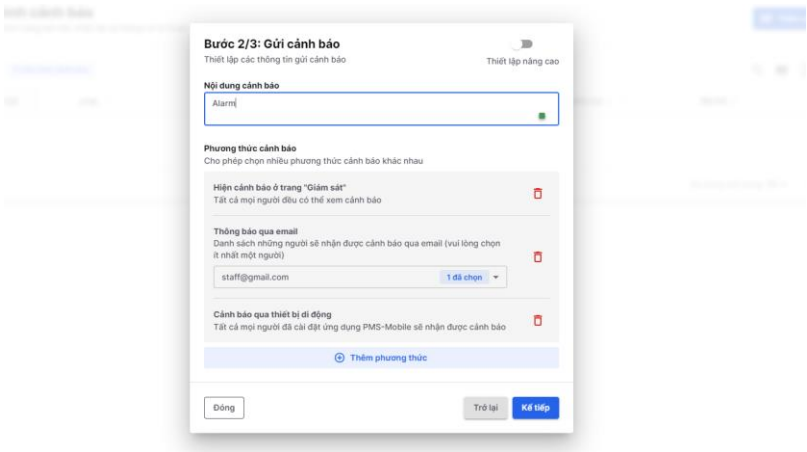


Figure 75 Administrators configure alarms for actions

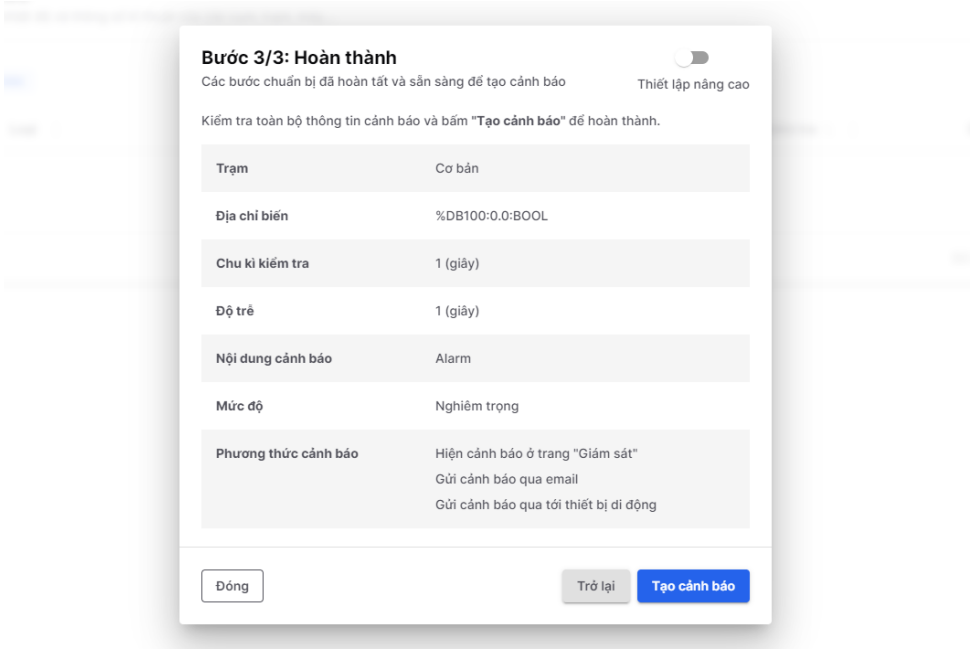


Figure 76 Administrators review and submit the new alarm configuration

3.3.4. Receive notifications

- Description: Users are notified when a defined alarm is triggered.

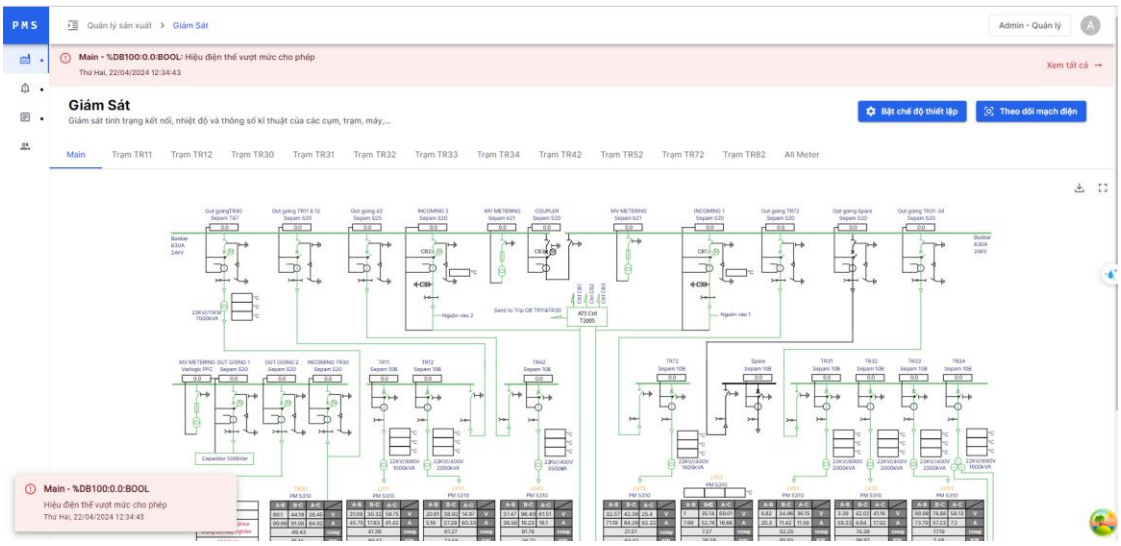


Figure 77 Users receive notifications in the monitoring screen

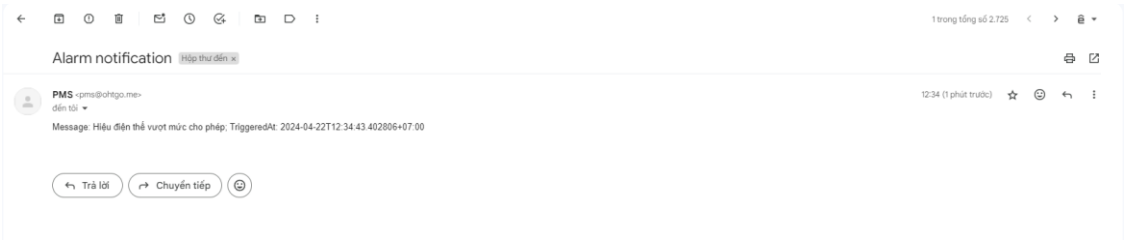


Figure 78 Users receive email notifications

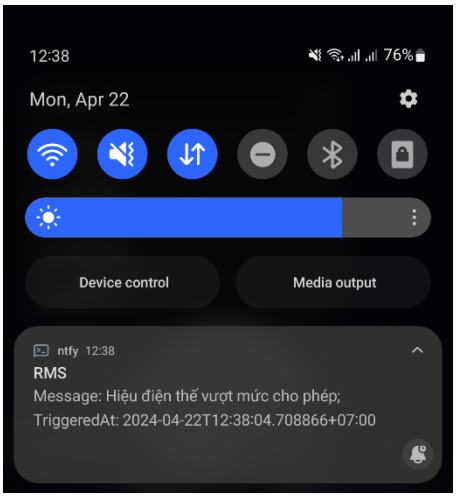


Figure 79 Users receive notifications on their mobile apps

3.3.5. View alarm histories

- Description: Supervisors view a historical list of resolved alarms.

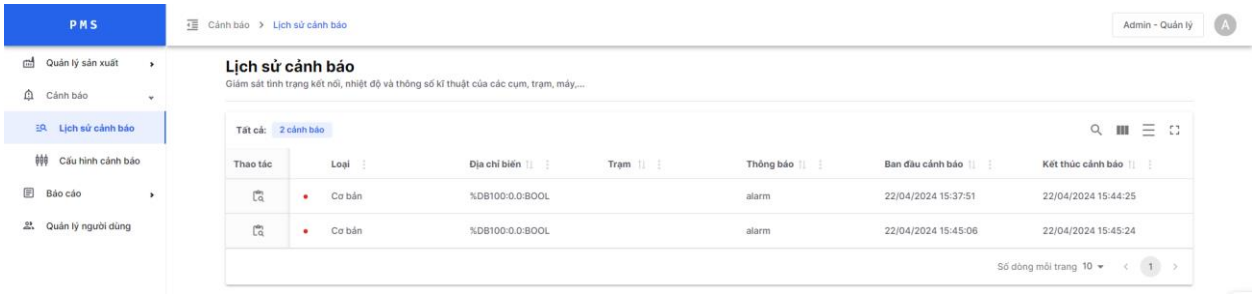


Figure 80 Supervisors view a historical list of resolved alarms.

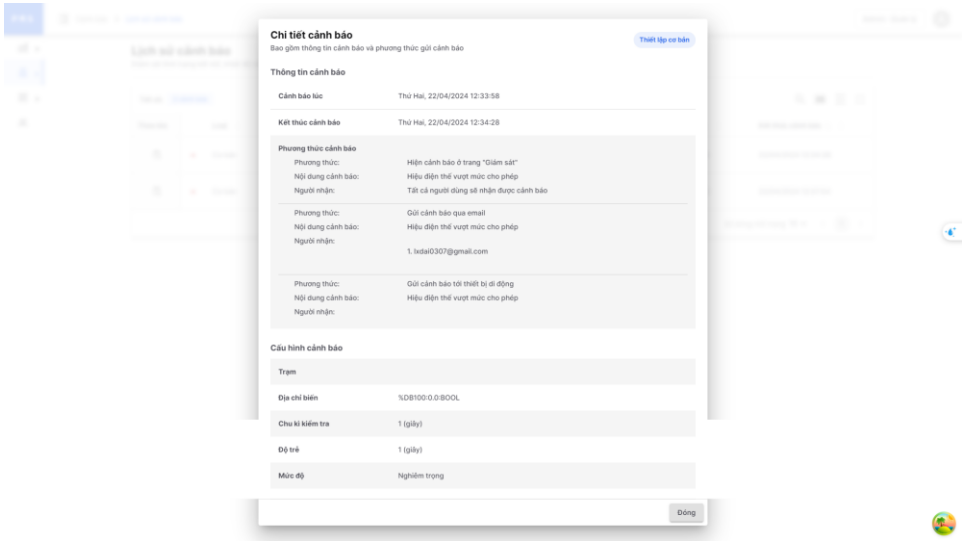


Figure 81 Supervisors view details of resolved alarm

3.4. View Reports

3.4.1. Search historical report

- Description: Supervisors filter historical reports based on specific criteria (e.g., date range, report type), and sort reports by various fields (e.g., date, report type).

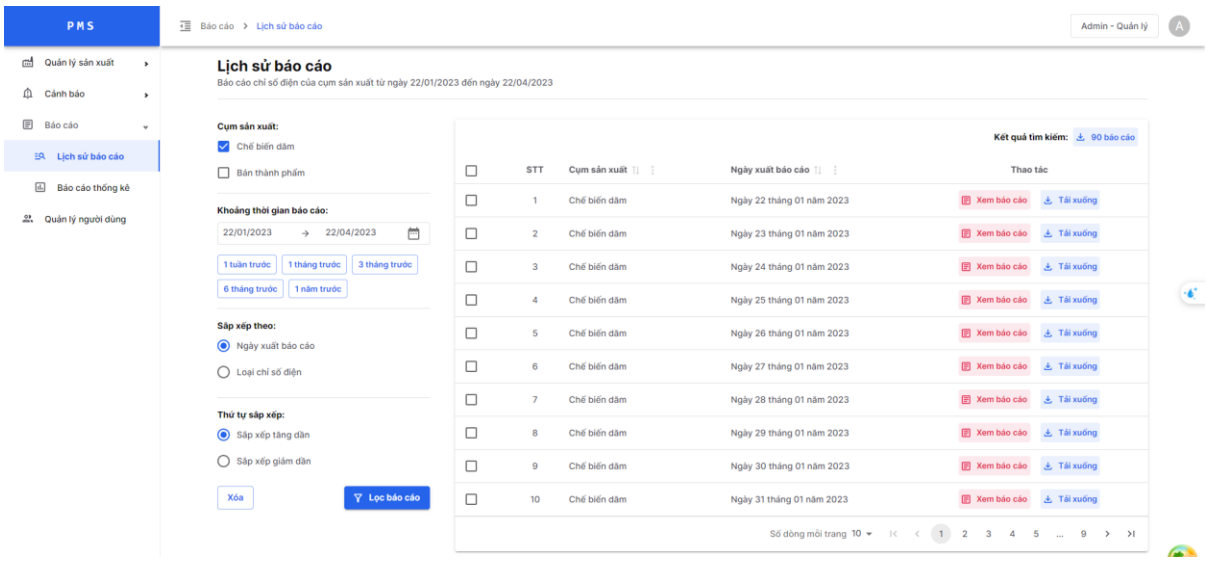


Figure 82 Supervisors search and view a list of historical reports

3.4.2. Export historical report

- Description: Administrators can export reports and view them in Excel format.

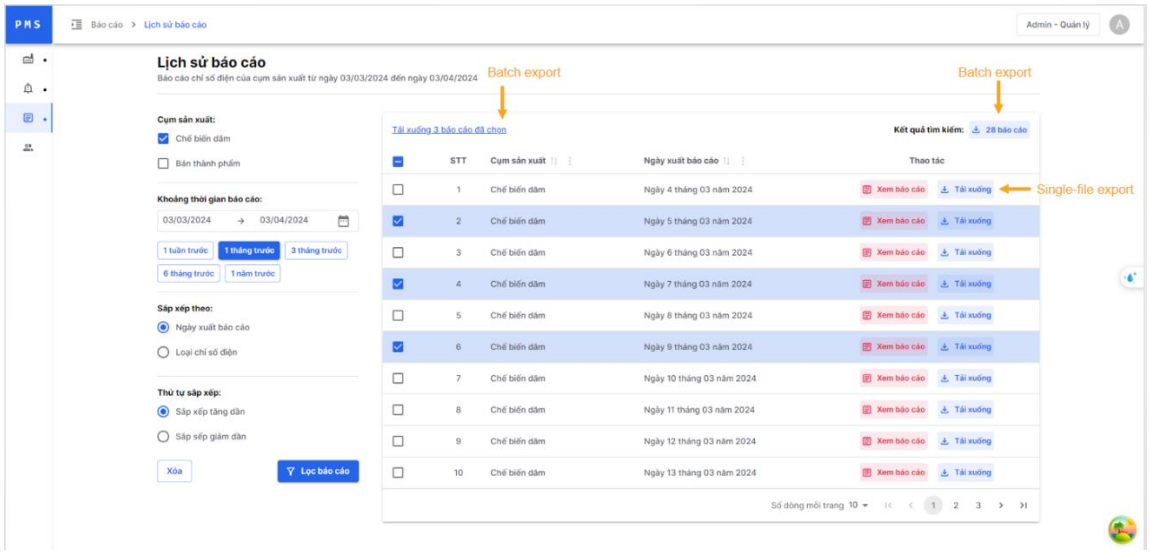


Figure 83 Administrators select reports to export

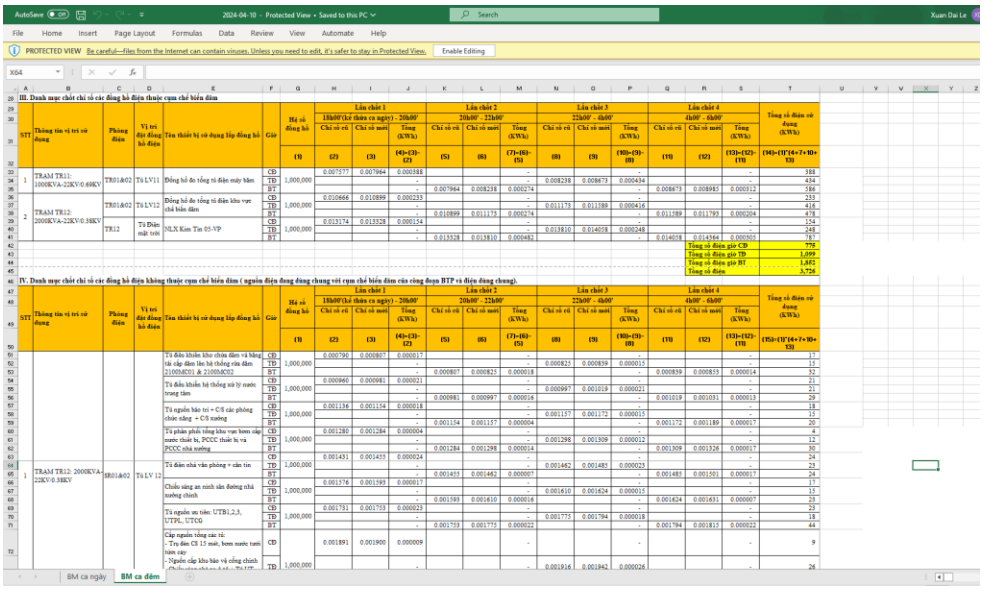


Figure 84 Administrators export reports in Excel format

3.4.3. View historical report details

- Description: Supervisors view historical report details which contains 3 sections, namely “Cả ngày”, “Ca sáng” and “Ca tối”. Each section includes several categories, and a category contains many data rows.

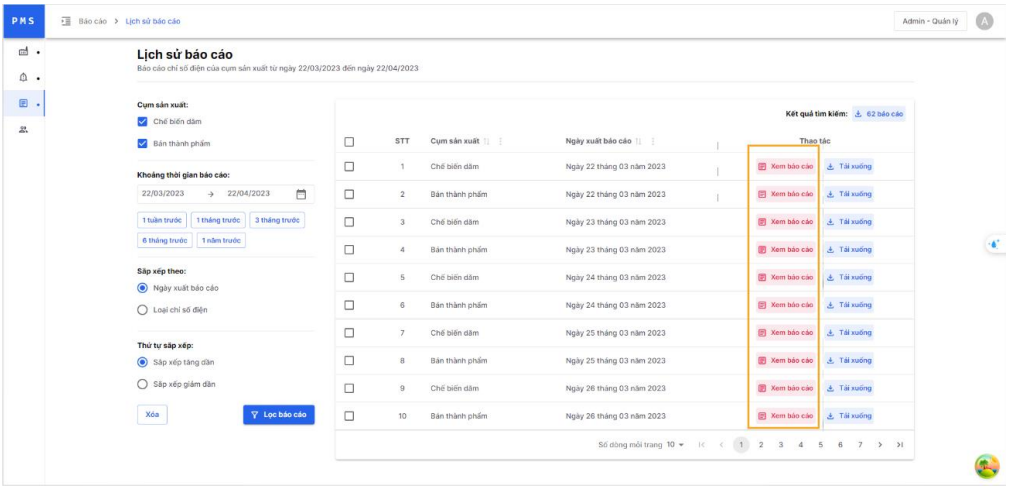


Figure 85 Supervisors click "Xem báo cáo" button to view historical report details

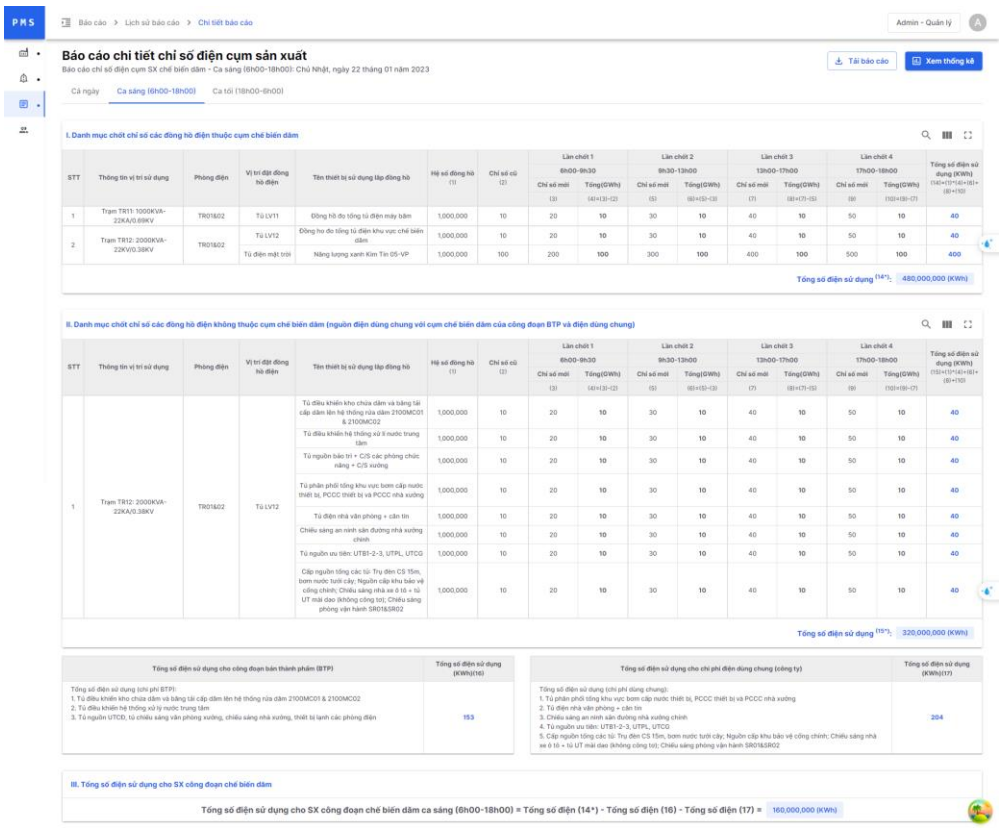


Figure 86 Supervisors view details of historical report

3.4.4. View Statistical Charts of a report

- Description: Supervisors view charts whose data belong to historical reports to gain insights about total electricity consumption.

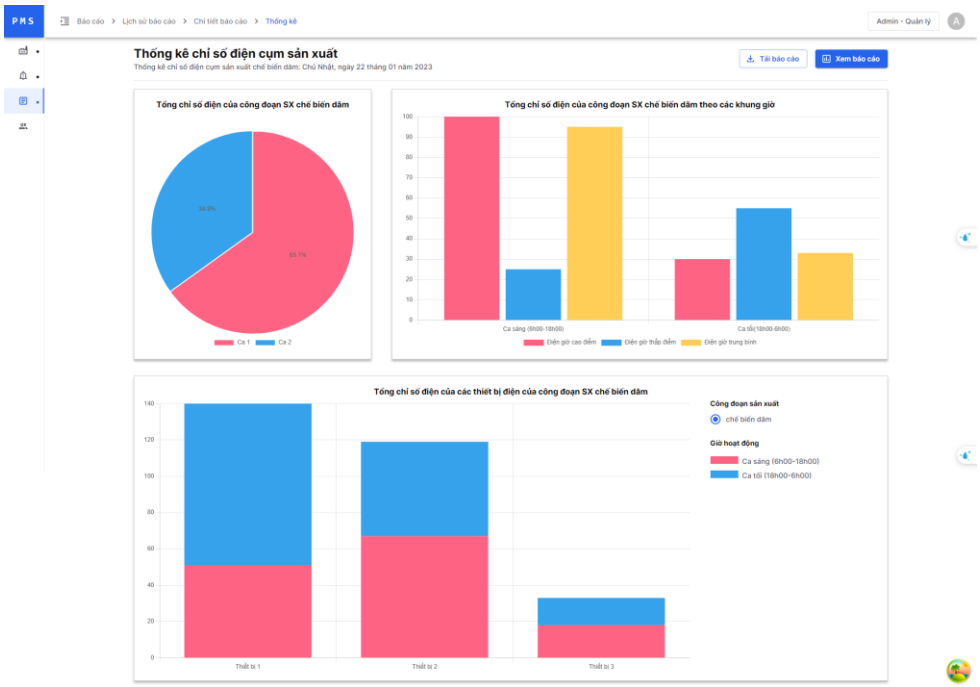


Figure 87 Supervisors view statistic charts of historical report

3.4.5. View Statistical Charts of a specific date range

- Description: Supervisors view charts whose data belong to historical reports across specific date range to gain insights about total electricity consumption.

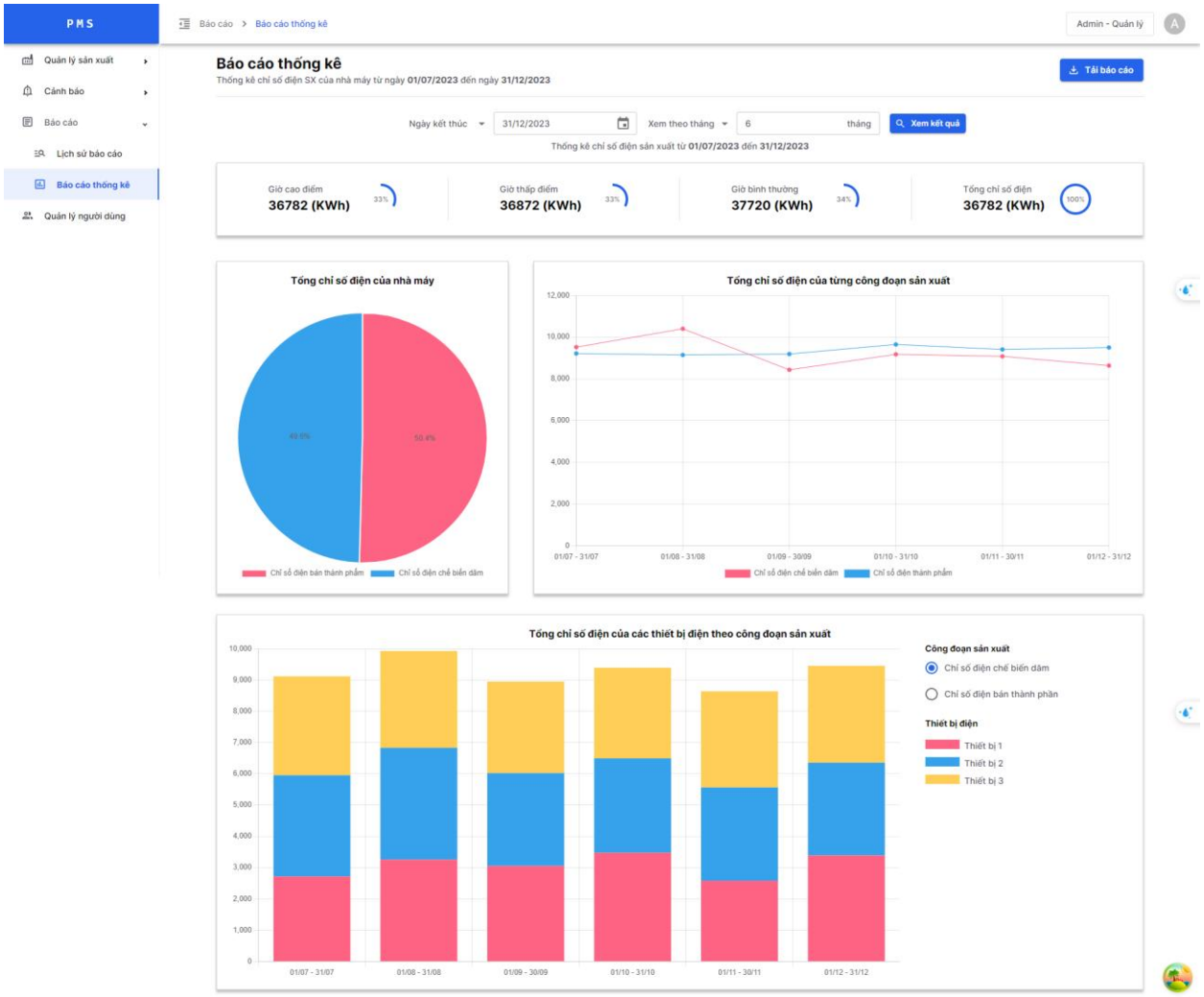


Figure 88 Supervisors view statistic chart of historical reports across specific date range

3.5. Manage Users

3.5.1. Create User

- Description: Administrators create new users.

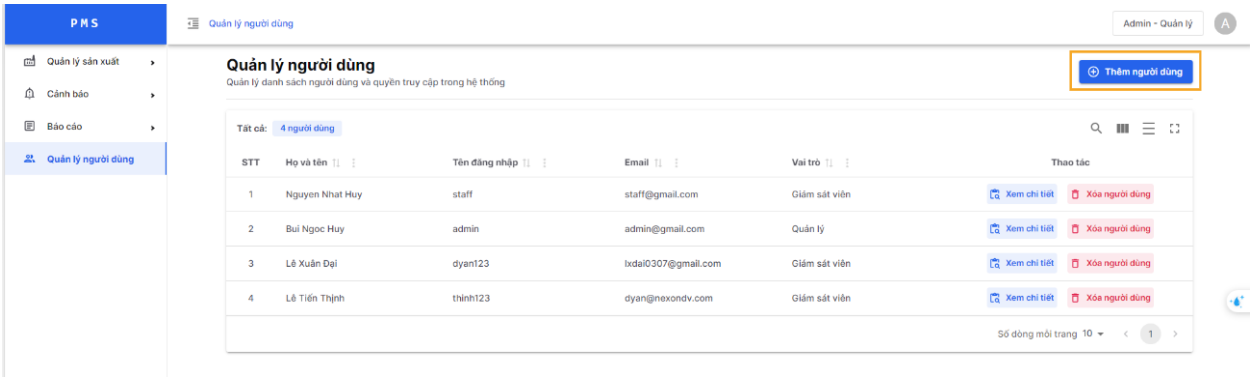


Figure 89 Administrators click "Thêm người dùng" to create new user

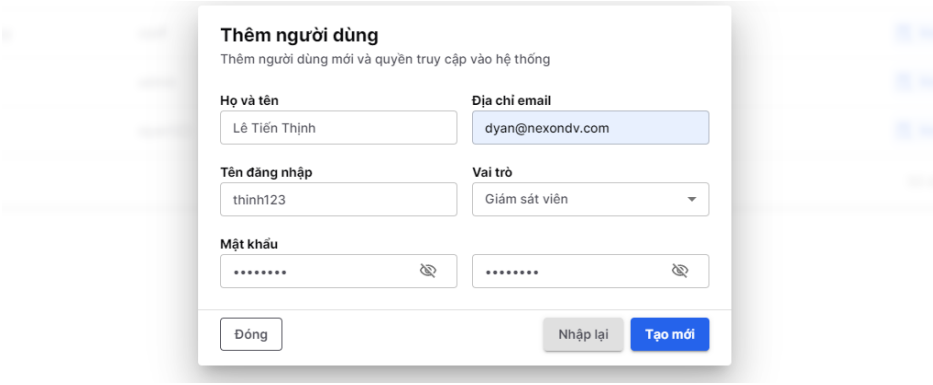


Figure 90 Administrators create new user

3.5.2. Update User

- Description: Administrators update user information.

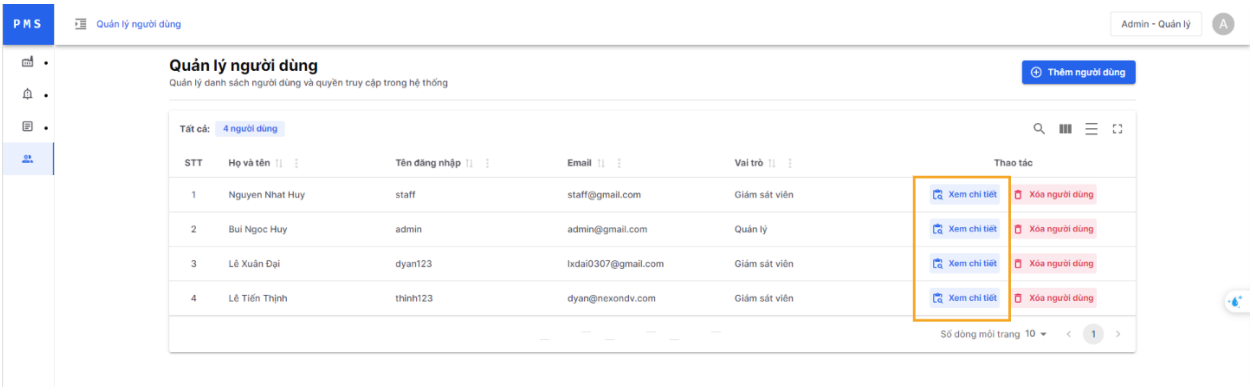


Figure 91 Administrators click "Xem chi tiết" button to view user information

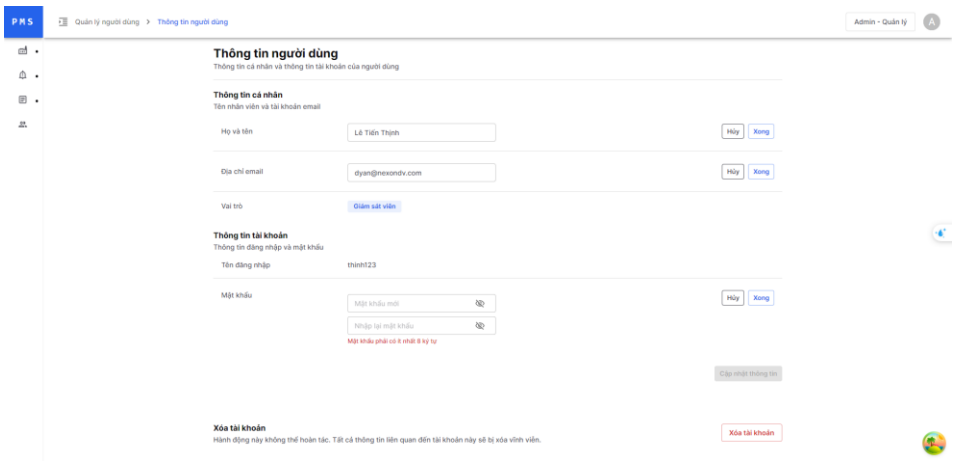


Figure 92 Administrators update user information

3.5.3. Delete User

- Description: Administrators delete a user.

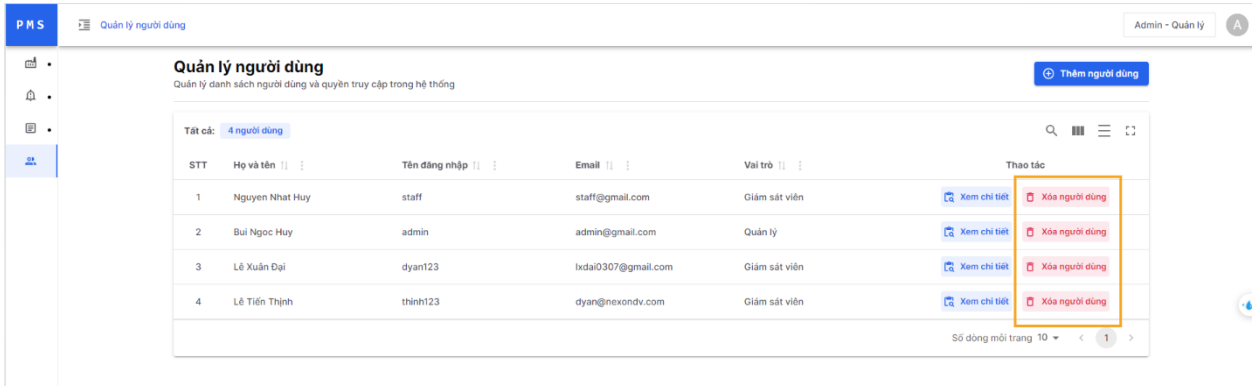


Figure 93 Administrators click "Xóa người dùng" button to delete a user

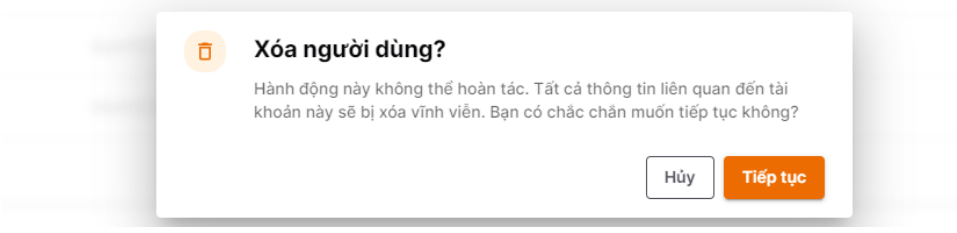


Figure 94 Administrators confirm the deletion of a user