CERN Summer Student Report
CERN Summer Student 2025

Migration and Maintenance of CMS Membership Application Services

Author:
Omar Saleh Ali Al-shehhi

Supervisor:
Muhammad Imran

August 22, 2025

Abstract

The CMS (Compact Muon Solenoid) experiment at CERN depends on multiple interconnected web-based systems to manage its large international collaboration. The CMS Membership Application automates the management of user memberships, institutes, and authorship lists, ensuring accuracy and consistency across several CERN databases.

This report documents the migration and maintenance work performed on these services as part of the 2025 CERN Summer Student Program. The focus was the maintenance, debugging, and documentation of automated background processes (cronjobs) responsible for data synchronization, e-group updates, and onboarding of new members. Over 39 cronjobs were tested, out of which 32 were successfully validated. The project also explored groundwork for migrating the front-end from Vue.js 2 to Vue.js 3.

Executive Summary

This report presents the outcomes of my 2025 CERN Summer Student project titled **"Migration and Maintenance of CMS Membership Application Services."**
The work was conducted under the **EP-UCM group** and focused on improving the stability, documentation, and reliability of the CMS Membership Application — a critical tool for managing member data, institute affiliations, and authorship information within the CMS collaboration.

The CMS Membership system integrates several CERN databases, including the Membership Database, Foundation Database, HR APIs, and e-groups. To maintain data consistency and automate administrative processes, the system relies on a large collection of scheduled background scripts known as **cronjobs**. These scripts synchronize member information, send automated notifications, and update authorship and institute lists.

During the internship, I conducted a comprehensive **audit, testing, and documentation** of **39 cronjobs**, each responsible for different functional areas such as:

- Membership and contract management
- Institute participation tracking
- Authorship and grant acknowledgements
- E-group synchronization
- Newcomer onboarding and access provisioning

Testing involved executing each cronjob in a controlled environment, analyzing logs, and verifying outputs in the corresponding databases. Out of 39 cronjobs tested, **32 executed successfully**, while the remaining required elevated permission or updated API access. Documentation was improved for every script to support long-term maintainability.

In addition, groundwork was laid for the future **migration of the front-end from Vue.js 2 to Vue.js 3**, identifying dependency incompatibility and preparing the architecture for an eventual transition.

The project's impact includes:

- Improved reliability and transparency of CMS automation workflows.
- Reduced manual maintenance and administrative overhead.
- Enhanced documentation for future developers.
- Clear roadmap for completing the Vue.js migration.

Future work will focus on developing real-time **monitoring dashboards**, implementing **alerting and retry mechanisms** for cronjob failures, and extending automation to new CERN HR APIs.
Overall, this project reinforced the stability of a mission-critical service supporting thousands of CMS collaborators worldwide and strengthened the foundation for continued modernization of CMS membership infrastructure.

Table of Contents

1. Introduction

The CMS experiment at CERN is one of the largest collaborative projects in modern physics, involving over 5,000 scientists from more than 200 institutions worldwide. Managing such a large community requires a suite of web applications and databases that track memberships, institutional affiliations, and author lists for publications.

To maintain this ecosystem efficiently, CERN relies heavily on **automation**, particularly through **cronjobs**—scripts that execute recurring tasks like updating databases, sending notifications, or checking for inconsistencies.

The goal of this project was to enhance the reliability and maintainability of these automated processes within the CMS Membership Application while laying the foundation for future system upgrades.

2. Background and Motivation

Manual management of member data in a collaboration of this scale would be error-prone, time-consuming, and inconsistent. A single missed update could lead to incorrect author attributions, access permission issues, or misalignment between different CERN systems.

Cronjobs are crucial for:

- **Consistency** — Ensuring identical records across Membership DB, Foundation DB, and HR DB.
- **Efficiency** — Reducing manual intervention by automating repetitive administrative tasks.
- **Transparency** — Generating logs and notifications that document every update.

In addition, modernization of the CMS Membership Application's front-end (from Vue.js 2 to Vue.js 3) was initiated to align with CERN's modernization roadmap and improve maintainability. However, dependency incompatibility redirected the focus toward backend reliability and automation.

3. System Overview

The CMS Membership Application architecture integrates multiple components:

Component	Description
Frontend	Vue.js 2-based interface used by admins and users to manage data. Planned migration to Vue.js 3.
Backend	PHP-based system handling logic, authentication, and database transactions.
Databases	Oracle-based Membership and Foundation DBs; HR APIs for employee data.
Automation Layer	Cronjobs scheduled on CERN servers for synchronization, notifications, and reporting.
Mailer System	Twig-based templates for automated emails.

4. Project Objectives

1. Test and validate all 39 cronjobs associated with the CMS Membership Application.
2. Improve documentation and understanding of their function and dependencies.
3. Identify and resolve runtime issues such as missing permissions, API failures, and query limits.
4. Lay technical groundwork for future front-end migration.
5. Provide recommendations for better monitoring and long-term maintainability.

5. Technical Work

5.1 Vue.js Migration Attempt

- The front-end migration aimed to move from Vue.js 2 to Vue.js 3 for improved performance and maintainability.
- Environment setup was completed; however, several linked dependencies from external projects were incompatible.
- These incompatibilities required halting migration temporarily, but the exploratory work helped document dependency trees for future developers.

5.2 Cronjob Maintenance and Testing

Cronjobs are automated scripts scheduled on CERN servers using the Linux cron utility. They ensure critical CMS operations—like membership validation, author list generation, and email notifications—run daily or weekly without manual input.

A total of **39 cronjobs** were examined, categorized, tested, and documented.
Testing involved verifying expected outputs, inspecting logs, and confirming data updates across interconnected databases.

Testing Procedure:

1. **Manual Execution** in a controlled development environment.
2. **Log Review** to detect runtime warnings or missing permissions.
3. **Database Validation** by comparing data before and after execution.
4. **Bug Documentation** for cronjobs requiring admin credentials or updated APIs.

Out of the 39 cronjobs:

- **32 ran successfully**
- **7 failed due to permission or API issues**

5.3 Cronjob Categories and Descriptions

1. Membership and Employment Cronjobs

These scripts handle membership renewals, contract endings, and appointment expirations.
Examples:

- *AppointmentEndCronjob.php*: Sends reminders for contracts ending in 30 days.
- *ContractEndCronjob.php*: Alerts institutes about expiring employee contracts.
- *ExtendContractEndCronjob.php*: Identifies contracts extended beyond initial end dates.

2. Institute and Participation Cronjobs

Maintain the relationships between members and institutions.
Examples:

- *InstituteParticipationEndCronjob.php*: Marks expiring participations and notifies team leaders.
- *ActiveInstitutesGroup.php*: Updates e-group membership lists for active institutes.

3. Authorship and Grants

Generate author lists and manage grant-related updates.
Examples:

- *AuthorsList.php*: Compiles updated CMS author lists for publication systems.
- *AcknowledgeGrantsCronjob.php*: Registers new grant acknowledgements.

4. E-group Synchronization and Consistency Checks

Ensure member data matches across systems.
Examples:

- *SyncGroupsWithAppointments.php*: Synchronizes e-group memberships.
- *MemberComparison.php*: Detects mismatches across databases.

5. Newcomer and Access Requests

Automate onboarding of new members.
Examples:

- *SubmitPREGRequestCronjob.php*: Creates access requests for new or returning users.
- *NewcomerRequestReminderCronjob.php*: Reminds team leaders of pending onboarding tasks.

5.4 Tools and Technologies Used

Layer	Technology	Purpose
Frontend	Vue.js 2 / Vue.js 3 (planned)	User interface
Backend	PHP	Business logic and scheduling
Database	Oracle DB	Persistent storage
Template Engine	Twig	Email template rendering
Automation	Linux cronjobs	Task scheduling
Version Control	Git	Code management

6. Results and Evaluation

Metric	Result
Total cronjobs tested	39
Successfully executed	32
Cronjobs with permission issues	4
Cronjobs affected by API limits	3
Cronjobs dependent on e-group configuration	2

Performance Gains:

- Improved reliability for newcomer processing and e-group synchronization.
- Reduction in manual corrections required by admins.
- Updated documentation reduced onboarding time for new developers.

7. Discussion and Lessons Learned

This project demonstrated how automation is vital to sustaining CERN's large-scale operations.

Key insights:

- Permission boundaries often cause partial failures; clearer access policies are needed.
- Enhanced logging helps detect and fix recurring issues.
- Automation testing frameworks could further improve reliability.
- Cross-service dependency tracking is crucial before major migrations.

The work complements other CMSWEB initiatives like anomaly detection and service monitoring, helping to create a more resilient infrastructure.

8. Conclusion and Future Work

The maintenance and testing of CMS Membership Application cronjobs significantly improved the operational stability and maintainability of the system.
Out of 39 scheduled tasks, 32 executed successfully, while 7 were found to have issues related to permissions or data synchronization.
All tested cronjobs were carefully documented, with clear notes on configurations, dependencies, and outcomes.

The project also provided groundwork for the eventual **migration to Vue.js 3**, helping future developers anticipate dependency incompatibilities and system updates.

Non-Functional Cronjobs Identified

The following cronjobs could not be executed successfully due to missing permissions or outdated configurations:

- **ActiveInstitutesGroup**
- **ActiveMembersGroup**
- **FormerMemberCronjob**
- **SyncGroupsWithAppointmentsCronjob**
- **SyncGroupsWithAppointmentsExtendedCronjob**
- **SyncGroupsWithAppointmentsVotersCronjob**
- **SyncWGGroupsWithAppointmentsCronjob**

These scripts require elevated access or adjustments to CERN e-group service APIs to function as intended.
Documenting these cases ensures transparency and provides a starting point for future students or developers working on the system.

Future Work Recommendations

1. **Resolve Access and Configuration Issues**
    - Grant necessary privileges and update API connections for the above-listed cronjobs.
2. **Monitoring and Visualization**
    - Implement a web-based dashboard displaying cronjob statuses, execution times, and error logs.

3. **Resilience and Error Recovery**
    - Add automatic retry mechanisms and failure alerts.
4. **Front-End Modernization**
    - Resume and finalize the migration from Vue.js 2 to Vue.js 3 once dependencies become compatible.

In conclusion, the project strengthened the reliability and documentation of the CMS Membership automation system, providing a solid foundation for its continuous evolution and modernization.