People's Democratic Republic of Algeria
Ministry of Higher Education and Scientific Research
University M'hamed Bougara - Boumerdès



Faculty of Science
Computer Science Department

Domain : Mathématiques Informatique

Sector : Informatique

Speciality : Systèmes d'informatique

N° de l'Arrêté d'habilitation de la spécialité : arrêté n °872 du 26/07/2016

Final dissertation submitted in partial fulfillment of the requirements for the Academic Bachelor's Degree

Theme

University Club Management System

Presented by :
ARAB Lydia
BENABDELMOUMENE Feriel
GUERAR Nesrine
Kebar Mouad

Defended on 04/06/2026 before the jury composed of
Ould Babaali Kawther : Examiner
Iza Lyla : Supervisor

Acknowledgements

First and foremost, we would like to thank Almighty God for giving us the strength, courage, and determination to accomplish this project.

We would also like to express our deepest gratitude to our beloved parents for their continuous support, encouragement, and sacrifices throughout our lives and academic journey.

Our sincere thanks go to Mrs. IZA for her availability, valuable advice, guidance, and constructive remarks, which greatly contributed to improving the quality of this work.

We are also very grateful to the members of the jury for accepting to evaluate and examine our work.

Finally, we would like to thank all the teachers of the Faculty of Sciences for their efforts and for the knowledge they have shared with us throughout our studies

Dedication

First of all, to our dear parents, grandparents, and families, for all the love, support, and care they have given us throughout our lives and studies.

May this modest work be the result of your prayers, encouragement, and wishes for our success. May Almighty God bless you with health, happiness, and a long life.

Then, to all our friends and classmates for their kindness and friendship.

Special thanks to our classmates and friends: BENBDELMOUMENE Feriel Lina, ARAB Lydia, and KEBAR Mouad, for their support, understanding, and help during the realization of this work.

Finally, to everyone who helped and supported us, directly or indirectly, in completing this project.

Table of contents

General Introduction 1

Chapter 1 – Study of the existing situation 2

Introduction ……..2
1.1 Description of the project context……..2
1.2 Introduction to study case……..2
1.3 Our objective in this project……..3
1.4 Problematics ……..3
1.5 Solutions ……..4
conclusion……..4

Chapter 2 – Analysis and design

Introduction ……..5
2.1 What's UML? ……..5
2.2 Functional definition……..5
2.2.1 Functional requirements ……..5
2.2.2 Non-functional requirements……..5
2.3 Actors of the system and functions……..5
2.4 Use case diagram.……..6
2.4.1 Use case diagrams of the system.……..6
2.4.1.1 Explanation of Use Cases……..7
2.4.1.2 General Use Case diagrams of the system..……..7
2.4.1.3 Detailed Use Case diagrams of the system..……..8
2.5 Textual description of the main use cases……..11
2.5.1 Textual description for Manage Requests……..11
2.5.2 Textual description for Submit Request……..11
2.5.3 Textual description for Check notifications……..12
2.5.4 Textual description for Share post……..13
2.5.5 Textual description for Manage department/club system.……..14
2.5.6 Textual description for Authentication……..14
2.6 Sequence Diagrams……..15
2.6.1 Check notification sequence diagram.……..15
2.6.2 Manage requests Sequence diagram.……..15
2.6.3 Authentification sequence diagram.……..17
2.6.4 Share post sequence diagram.……..18
2.6.5 Manage members sequence diagram.……..19
2.6.6 Submit request sequence diagram.……..21
2.7 Class diagram.……..21
2.7.1 The data dictionary……..21
2.7.2 System classes……24
2.7.3 Management rules……26
2.7.4 Class diagram.……26
2.8 Logical Data Model……27

2.8.1 Rules for transitioning from class diagrams to relational model……27
2.8.2 The relational model of the system……27
conclusion……27

Chapter 3 – Realization

Introduction ……28
3.1 Environment and development tools……28
3.1.1 Environment materiel……28
3.1.2 Presentation of the graphical interfaces……28
3.2 Presentation of the application……29
3.2.1 Explore as guest page……29
3.2.2 Reply to join request……29
3.2.3 President profile page……30
3.2.4 Manager profile page……31
3.2.5 Instructor profile page……32
3.2.6 Member profile page……32
3.3 Security features……33
Conclusion……34

General conclusion 35

Table of figures

Figure 1: Organizational chart of Catalyst Chemistry Club.....……3

Figure 2: General Use Case diagram.……8

Figure 3: Visitor use case diagram.……8

Figure 4: Member use case diagram.……9

Figure 5: Instructor use case diagram.……..9

Figure 6: Manager use case diagram.……10

Figure 7: President/Vice President use case diagram.……10

Figure 8: Check notification sequence diagram.……16

Figure 9: Manage request sequence diagram.……...17

Figure 10: Authentication sequence diagram.……..18

Figure 11: Share post sequence diagram.……..19

Figure 12: Manage department/club system.……20

Figure 13: Submit request sequence diagram.……..21

Figure 14: Class diagram of the system.……26

Figure 15: Explore as guest page……29

Figure 16: Reply to join request.……...30

Figure 17: President profile page.……30

Figure 18: Manager profile page……..31

Figure 19: Instructor profile page.……32

Figure 20: : Member profile page.……33

Table of tables

Table 1: Resume of use case diagrams of the system……..7

Table 2: Textual description for “Manage Requests” use case……..11

Table 3: Textual description for “Submit Requests” use case…….12

Table 4: Textual description for “Check Notification” use case……..13

Table 5: Textual description for “Share posts” use case…….13

Table 6: Textual description for “Manage Members” use case……..14

Table 7: Textual description for “Authentication” use case…….15

Table 8: The data dictionary……24

Table 9: The system classes……...25

List of abbreviations

UML	Unified Modeling Language
UP	Unified Process
SQL	Structured Query Language
HTML	Hyper Text Markup Language
CSS	Cascading Style Sheets
JS	JavaScript
API	Application Programming Interface
RBAC	Role-Based Access Control
CORS	Cross-Origin Resource Sharing
RAM	Random Access Memory
GO	Gigaoctet (Gigabyte)
GHz	Gigahertz
JPEG	Joint Photographic Experts Group
JPG	Joint Photographic Experts Group
PNG	Portable Network Graphics
GIF	Graphics Interchange Format
MB	Megabyte

General Introduction

The internet has transformed the way we learn and work. Today, people can study almost any subject from home through online courses, whether free or paid. As universities embrace digital transformation, managing their activities efficiently has become more important than ever.

University clubs are a key part of student life. They encourage participation, help students build new skills, and bring them together. However, many universities still manage their clubs using outdated methods — paper-based registration, informal messaging, and little to no tracking of members or events. This leads to poor organization, miscommunication, and missed opportunities for students.

In this context, our final year project focuses on designing and building a website for managing university club. The platform gives students a clear view of the club, its activity, and how to join. Club organizers can use it to share updates, post events, and interact with their members — making the whole experience smoother for everyone involved.

The project is organized into three chapters:

- **Chapter 1 – Study of the existing situation**: This chapter presents the club's context, structure, and the main challenges it faces. It also defines our objectives and outlines the proposed solution. The goal is to provide a clear understanding of why a digital management system is needed.
- **Chapter 2 – Analysis and design**: This chapter defines the system requirements, actors, and use cases. It presents UML diagrams (use case, sequence, and class diagrams) and translates them into a relational database model. This chapter provides the technical blueprint for implementation.
- **Chapter 3 – Realization:** This chapter explains how the platform was built. It describes the hardware and software tools used for development. Then, it presents screenshots of the actual website interfaces for each actor (president, manager, instructor, and member). Finally, it covers security measures to protect user data and ensure reliable system operation.

Chapter 01:

Study of the existing situation

Introduction

This chapter describes the background of our project. It explains how a student club generally works, then introduces the Catalyst Chemistry Club as our case study. After presenting our objectives and the club's main problems, we propose a digital solution. This chapter lays the foundation for the rest of our project.

1.1 Description of the project context

A club is a student's small community where they can learn more about the field they are passionate about.

After studying the organization of a club in our university, we noticed that it follows strict rules to ensure that members learn effectively and benefit from all the club's activities and opportunities.

Once accepted as a member, a student can participate in many activities such as events and workshops, become an instructor or a manager, or withdraw from the club.

The president and managers are also responsible for tracking and managing the members' activities and progress inside the club.

Therefore, the main objectives of our project are:

1- To ensure flexible and efficient interaction between club members.
2- To provide a good and organized experience for all club members.

1.2 Introduction to study case

- **Catalyst chemistry club**



Catalyst is a student club created by students who are passionate about chemistry and scientific discovery. The club aims to bring together students who want to learn, share knowledge, and explore chemistry beyond their university courses.

The club organizes different activities such as workshops, scientific discussions, and collaborative projects to help students improve their practical and theoretical skills. It also creates opportunities for students to meet professors and experts in chemistry and encourages them to participate in research and scientific initiatives, this club consists of:

1- **Scientific Department**: Focuses on the academic and experimental activities of the club.
   - **Research Sub-department**: Prepares experiments and research projects.
   - **Workshops Sub-department**: Organizes practical laboratory workshops and activities.
   - **Training and Education Sub-department**: Provides tutorials, learning sessions, and mentoring to improve members' chemistry knowledge and skills.
2- **Events and Communication Department**: Responsible for the club's activities and public visibility.
   - **Events Sub-department**: Organizes competitions, conferences, and scientific events.
   - **Media and Communication Sub-department**: Manages social media, designs posters, and promotes the club's activities.
3- **Administration and Logistics Department**: Manages the internal organization and resources of the club.

- **Administration Sub-department**: Coordinates meetings and manages the club's organization.
- **Partnership and Sponsorship Sub-department**: Builds collaborations and looks for sponsorships.
- **Logistics and Equipment Sub-department**: Manages laboratory materials, chemicals, and equipment needed for experiments and events.

The club members:

The club works thanks to the cooperation of several members, and each one has a specific to help the club succeed and organize its activities well.

The president is responsible for managing the club and supervising all its activities. The president is helped by a vice-president, who assists in organizing and coordinating the work. Each main department has a manager who supervises the department and its tasks. In every sub-department, there is an instructor whose role is to teach and guide the members during workshops, training sessions, and other learning activities.

Organizational chart of the club

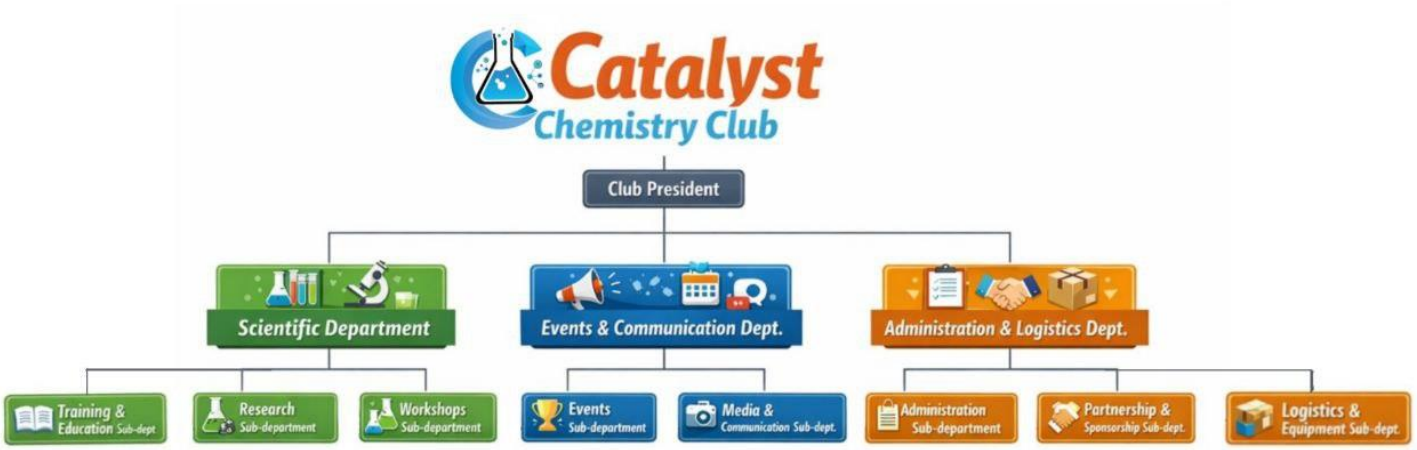


Figure 1: Organizational chart of Catalyst Chemistry Club.

1.3 Our objective in this project

These objectives are summarized below:

1- To have a complete profile for each member.
2- To facilitate members' activities.
3- To eliminate any paper work.

1.4 Problematics

After discussing with the club's president, we identified some problems affecting the club's management, which make it difficult for members to stay focused and have the best experience:

- Poor communication makes it hard to share event ideas with the president.
- Paper-based join/withdraw processes are inconvenient for everyone.
- Students must come to the university to check request responses or event information.
- Nomination process is complex and requires excessive paperwork.
- Member data stored on paper is insecure and poorly organized.
- Managing many members manually is inefficient and difficult to update.

- Students struggle to track workshops and tasks without daily presence.
- Event efforts are not documented or shared, reducing visibility and memory.

1.5 **Solutions:** Our goal now is to find suitable solutions to the previous problems.

The main issues are related to poor communication, heavy reliance on paperwork, and the lack of a digital system to manage members, events, and responsibilities. As a result, processes such as proposing events, joining or leaving the club, managing member information, and selecting new instructor/managers/vice-president or a president become slow, complicated, and inefficient.

To address these problems, we propose developing a website that includes several features designed to simplify and improve the club's management:

- Develop a website to digitize and improve club management processes.
- Replace paperwork with electronic forms for all requests and proposals.
- Provide notifications so users can track their requests easily.
- Share events, workshops, and media online to inform and attract students.
- Insure the security of students' information.

Conclusion

In this chapter, we presented the Catalyst Chemistry Club's structure and defined our three main objectives. Through discussion with the club's president, we identified key problems such as poor communication, paperwork, and lack of a digital system. To solve these issues, we proposed developing a website to digitize requests, send notifications, share events, and secure member data. The next chapter will present our methodology.

Chapter 02:

Analysis and design

Introduction

This chapter covers the analysis and design of the proposed system. It defines UML, lists functional and non-functional requirements, and identifies the system's actors. It then presents use case diagrams, sequence diagrams, the class diagram, and finally the relational model.

2.1 What's UML?



Unified Modeling Language (UML) is a standardized visual language used in software engineering to model and design systems, it provides diagrams and symbols to represent how a system works.

UML helps developers, analysts, and stakeholders understand the system clearly. It represents both static aspects (structure) and dynamic aspects (behavior), and using UML improves communication, reduces complexity, and supports better system development.

2.2 Functional definition:

UP (Unified Process) distinguishes two types of requirements in application development:

1- **Functional requirements:** These are the system's functionalities. They specify I/O behavior. And what the system does.
2- **Non-functional requirements:** These encompass everything related to quality, reliability, performance, legal aspects, etc.

2.2.1 Functional requirements

1- **Forms Management**
- The system will replace all paper forms with digital ones.

2- **Replies and Notifications**
- The system must allow every form to be replied to by the receiver.
- The system shall allow the sender to check his notifications.

3- **Posts and Updates**
- The system must allow the managers, instructors, vice-president and the president to post information, pictures, and event details to keep members updated.

4- **Member/Instructors and Manager Control**
- The system shall allow adding, removing, and modifying members, instructors, and manage information.

2.2.2 Non-functional requirements

1- The system must protect users' personal data.
2- The interface must be simple and user-friendly to help users better manage their workspace and facilitate their activities.
3- The application must ensure reliability and ease of maintenance.

2.3 Actors of the system and functions:

The system includes several types of users

1- **Main actors:** the core users essential for the system to function:
- **Member:** A student officially in the club who participates in events, a member mainly can submit requests, and check notifications.
- **Instructor:** A skilled member who organizes and leads workshops, instructors can share the information about workshops.

- **Manager:** A member responsible for managing domains, members, and events, so essentially they handle member management, approve requests, and post updates.
- **President:** The leader who supervises the club and makes key decisions, also he oversee all activities, and manage managers.
- **Vice-President:** Assists the president and replaces them when necessary.

2- Secondary actors: helps the main indirectly to keep the system running smoothly:

- **Visitor:** A non-member who browses and views public information, he can explore the space with limited access to the publications.

2.4 Use case diagram

A Use Case Diagram is a type of UML diagram that describes how users interact with a system by showing actors, use cases, and the relationships between them.

Its main purpose is to present the system's functionality from the user's point of view without focusing on technical details. It helps both developers and users understand what the system is expected to do. Use case diagrams are useful for clarifying requirements and improving communication during the early stages of system design

2.4.1 Use case diagrams of the system: All the principal use related to their concerned actor are resumed in the table below:

Actor	**Type**	**Use cases**
Visitor	Secondary actor	**C0:** Explore the visitor space
Member	Primary actor	**C0:** Authentication **C1:** View profile **C2:** Submit requests **C3:** Check notifications
Instructor	Primary actor	**C0:** Authentication **C1:** View profile **C2:** Submit request **C3:** Check notifications **C4:** Share publications
Manager	Primary actor	**C0:** Authentication **C1:** View profile **C2:** Submit requests **C3:** Check notifications **C4:** Share posts **C5:** Manage requests **C6:** Manage department system
President	Primary actor	**C0:** Authentication **C1:** View profile **C3:** Share publications **C4:** Manage requests **C6:** Manage the club's information system

Vice-president	Primary actor	Same as the president

Table 1: Resume of use case diagrams of the system.

2.4.1.1 Explanation of Use Cases:

- **View profile:** Allows users to access and review their personal information, roles, and activity within the system.
- **Submit request**: Members can submit various types of requests, such as applying for positions (instructor, manager, vice-president, president), proposing event ideas, joining events, or requesting to join/leave a department. All requests are directed to the appropriate authority (president or manager) depending on their type.
- **Manage requests:** president, and managers review and respond to incoming requests.
- **Check notifications:** Members view responses to their requests through notifications, and they can be notified about their manager/instructor posts.
- **Share posts:** Instructors, managers, and the president post workshops, events, and updates.
- **Manage the department system:** Managers handle adding, removing, and tracking department members.
- **Manage the club system:** The president monitors and tracks all members in the club.

2.4.1.2 General Use Case diagrams of the system



Figure 2:General Use Case diagram.

2.4.1.3 Detailed Use Case diagrams of the system

1- **Visitor use case diagram:** The Visitor can explore the club's public space without logging in. This allows them to view general information about the club, its activities, and posts (limited access):

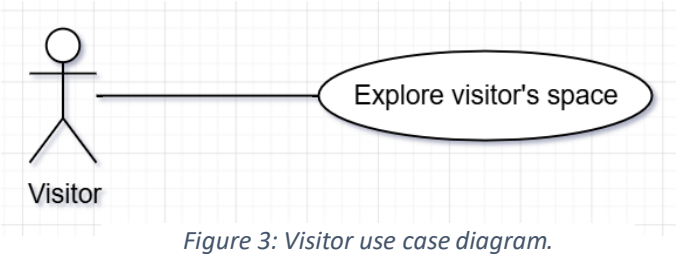


Figure 3: Visitor use case diagram.

2- **Member use case diagram:** The Member logs in to view their profile, notifications, and submits different requests (event idea, nomination, withdraw…) to the manager or president using electronic forms:

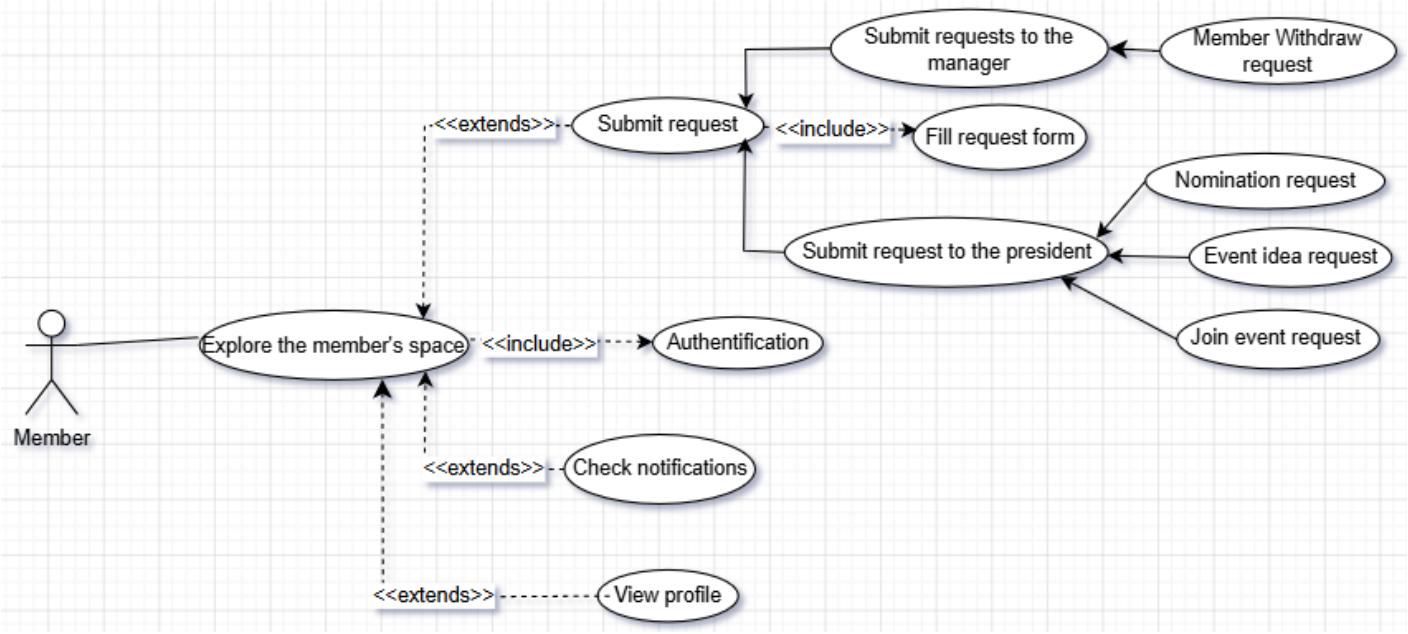


Figure 4: Member use case diagram.

3- **Instructor use case diagram:** The Instructor inherits all Member actions (view profile, check notifications, submit requests) and can additionally share posts. This extra feature allows them to publish information or announcements within the club:

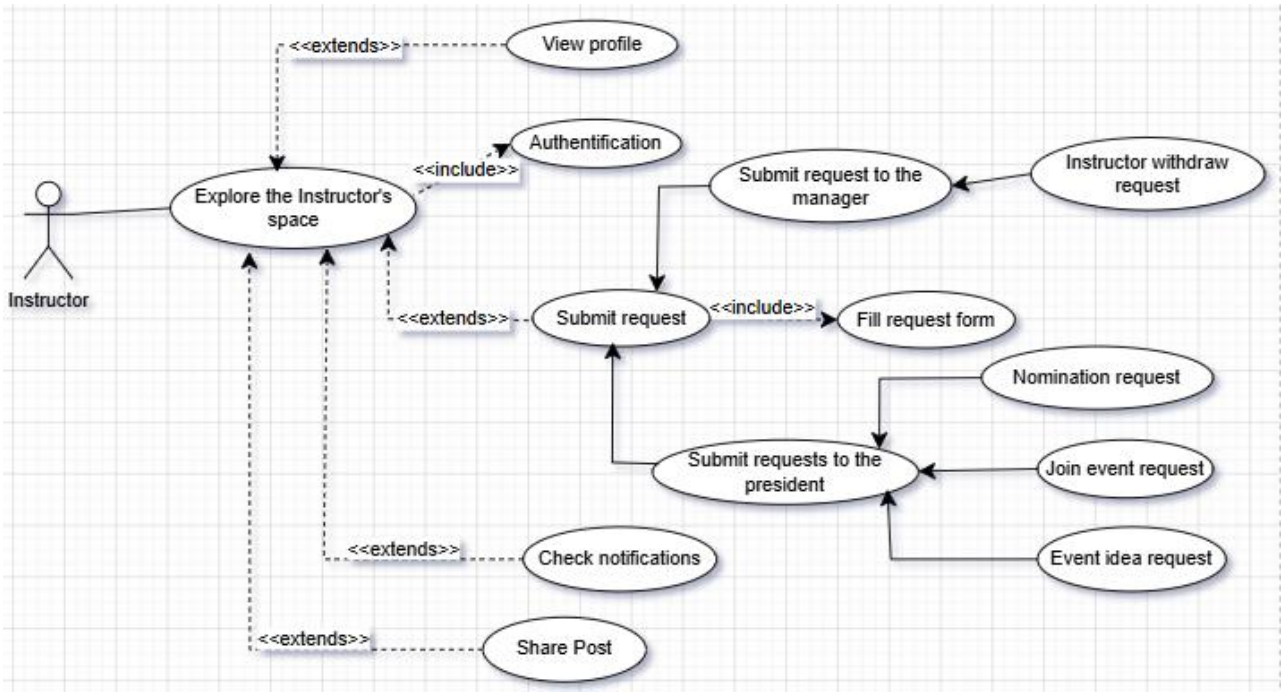


Figure 5: Instructor use case diagram.

4- **Manager use case diagram:** The Manager inherits all Instructor features (view profile, check notifications, submit requests, share posts) and additionally manages

their department system. This includes adding, modifying, or removing members, and sending replies to member requests:

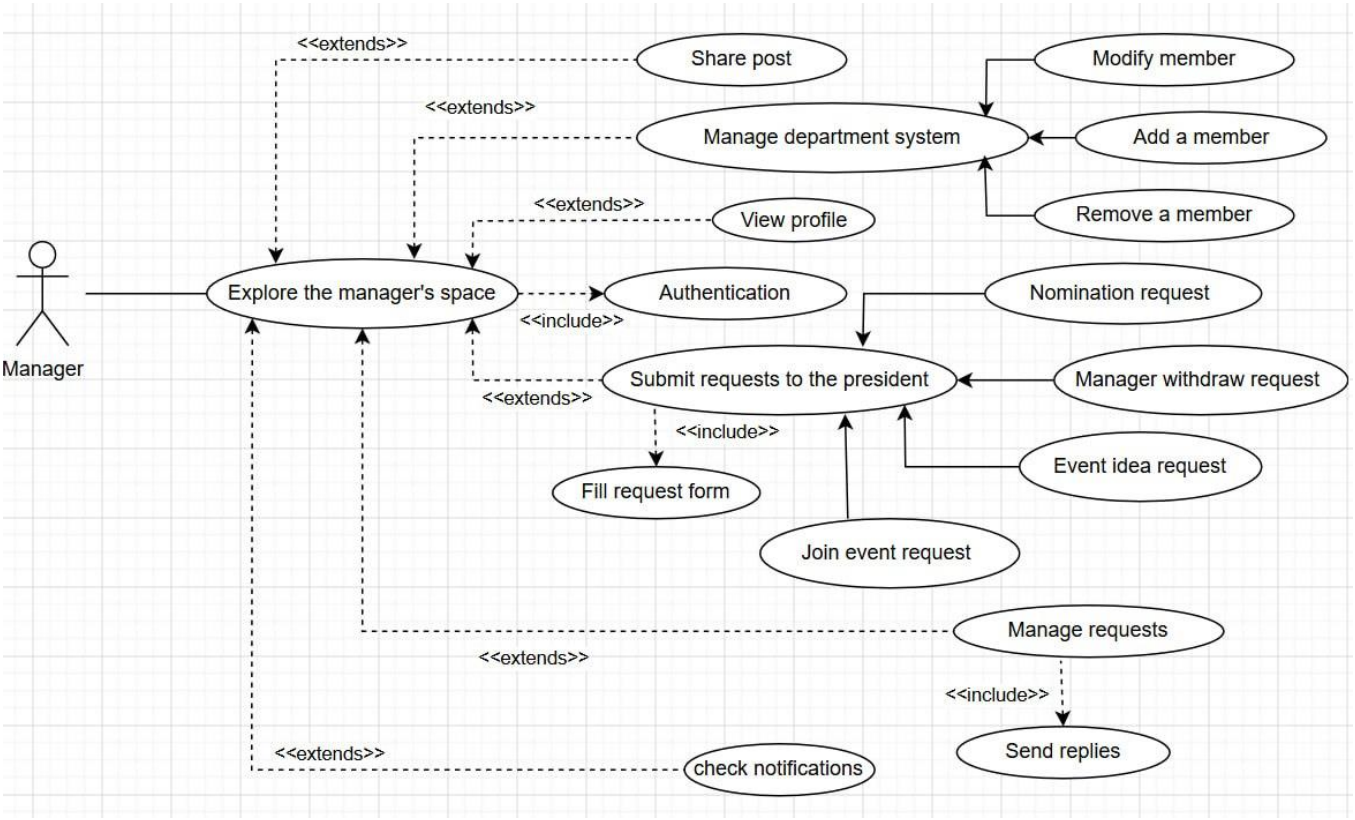


Figure6: Manager use case diagram.

5- **President/ Vice President use case diagram:** The President and Vice-president can log in to view their profile, share posts, and send replies. They also have the authority to add, remove, or modify manager information:

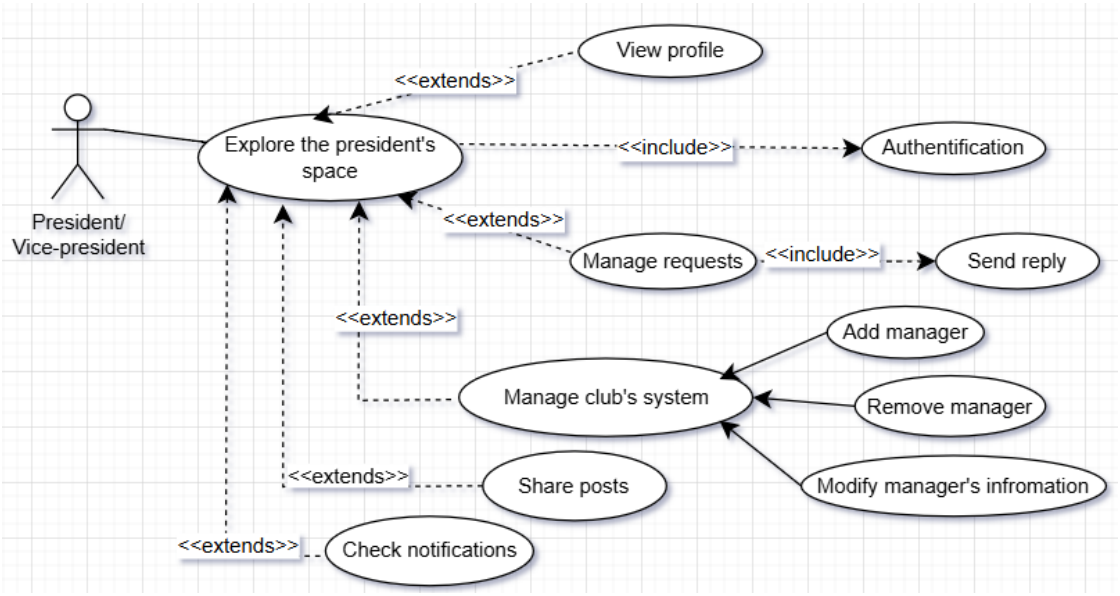


Figure 7: President/Vice President use case diagram

2.5 Textual description of the main use cases: A textual description explains how a user interacts with a system through a use case. It describes the main steps, actors, conditions, and outcomes. It provides a clear and simple view of the process.

2.5.1 Textual description for Manage Requests

Manage Requests	
Brief description	
Title	Manage Requests
Aim	Allow responsible actors to review and respond to requests.
Actors	President/Vice-president, Manager
Description of the sequences	
Precondition	The user is authenticated Member submitted a request
Main sequence	1- The user accesses their workspace 2- The user chose “Requests” option 3- The system displays the list of received requests 4- The user selects a request 5- The user chooses to accept or reject the request 6- The user press “Submit” 7- The system validate the answer 8- The database records the decision 9- The system notifies the user about the result
Alternative sequence	Same scenario until step 6: If the answer was not validated - System shows error message and ask to try again
Post condition	The request is processed and updated

Table 2: Textual description for “Manage Requests” use case.

2.5.2 Textual description for Submit Request

Submit Requests	
Brief description	
Title	Submit Request
Aim	Allow users to send requests to the appropriate authority (president/vice-president, or manager).
Actors	Instructor, Manager, Member

Description of the sequences	
Precondition	The user is authenticated.
Main sequence	1- The user accesses their workspace 2- The user selects the option “Submit request” 3- The system displays the request form 4- The user selects the type of request 5- The user fills in the required information 6- The user submits the request 7- The system sends the request to the appropriate actor 8- The system confirms that the request has been sent
Alternative sequence	Same scenario until step 6: If required fields are missing or incorrect: - The system displays an error message and asks the user to correct the form.
Post condition	The request is successfully recorded and sent to the concerned authority.

Table 3:Textual description for “Submit Requests” use case.

2.5.3 Textual description for Check notifications

Check Notifications	
Brief description	
Title	Check notifications
Aim	Allows users to view notifications related to their requests, updates, and system activities
Actors	President/Vice-president, Manager, Instructor, Member
Description of the sequences	
Precondition	The actor is authenticated
Main sequence	1- The user access their workspace 2- The user selects the option “Notification” 3- The system requests the user’s notifications from the database 4- The database returns the list of notifications 5- The system displays the notifications list ordered by date 6- The user marks the notification as “seen” 7- The system sends a request to update the notification status to “seen” 8- The database confirms the update

	9- The system updates the interface (notification marked as Seen)
Alternative sequence	Same scenario until step 3 If there are no notification exists: - The data base returns an empty set - The system displays “no notification yet”
Post condition	Notifications are displayed and updated as seen in the database

Table 4: Textual description for “Check Notification” use case.

2.5.4 Textual description for Share post

Share post	
Brief description	
Title	Share post
Aim	Allows authorized users to publish posts such as events and workshops
Actors	Instructor, Manager, President /vice-president
Description of the sequences	
Precondition	The actor is authenticated and have publishing rights
Main sequence	1- The user accesses their workspace 2- The user selects the option “Share post” 3- The system displays the content creation form 4- The user fills in the form (title, description, date, etc.) 5- The user uploads images (optional) 6- The user clicks “Publish” 7- The system validates the data 8- The system sends the data to be stored 9- The system displays a success message.
Alternative sequence	Same scenario until step 6 If data is invalid: - Returns validation errors - The application Displays error messages and asks for correction
Post condition	The content is stored in the database and visible to users through the application

Table 5: Textual description for “Share posts” use case.

2.5.5 Textual description for Manage department/club system

Manage department/club system	
Brief description	
Title	Manage department/club system
Aim	Allows the user to add and remove and modify student information in the system
Actors	Manager , President/vice-president
Description of the sequences	
Precondition	The manager/president is logged into the system
Main sequence	1- The user logs into the system 2- The user accesses the space 3- The user accesses the Department/Club system 4- The system displays the list of members. 5- The user clicks add / remove / modify student. 6- The user enters the student information needed. 7- The user confirms the action. 8- The system updates and displays the member list.
Alternative sequence	Same scenario until step 7: In case of incorrect information: - The database detects incorrect or incomplete information. - The database sends an error message. - The system displays an error message and asks the user to try again.
Post condition	Members are added or removed successfully The member list is updated

Table 6: Textual description for "Manage Members" use case.

2.5.6 Textual description for Authentication

Authentication	
Brief description	
Title	Authentication
Aim	Allow the user to access the system by verifying their identity

Actors	All the actors
Description of the sequences	
Precondition	The user must have a valid account and have a password and username
Main sequence	1-The user accesses the log in page 2-The system displays the log in form 3-The user enters username and password 4-The system checks if the information is valid 5-The system sends the user to their profile page
Alternative sequence	Same scenario until step 3: In case of incorrect information: - The website detects incorrect or incomplete information. - The website displays an error message (invalid password or username) - The user is asked to correct the information and try again.
Post condition	the user access the system

Table 7: Textual description for "Authentication" use case.

2.6 Sequence Diagrams

A sequence diagram is a type of UML diagram that shows how different objects or actors interact with each other over time. It focuses on the order of messages exchanged between them to complete a specific process or use case, using a timeline from top to bottom.

2.6.1 Check notification sequence diagram:

Check Notification allows an authenticated user to view their list of notifications, and automatically update its status in the database:

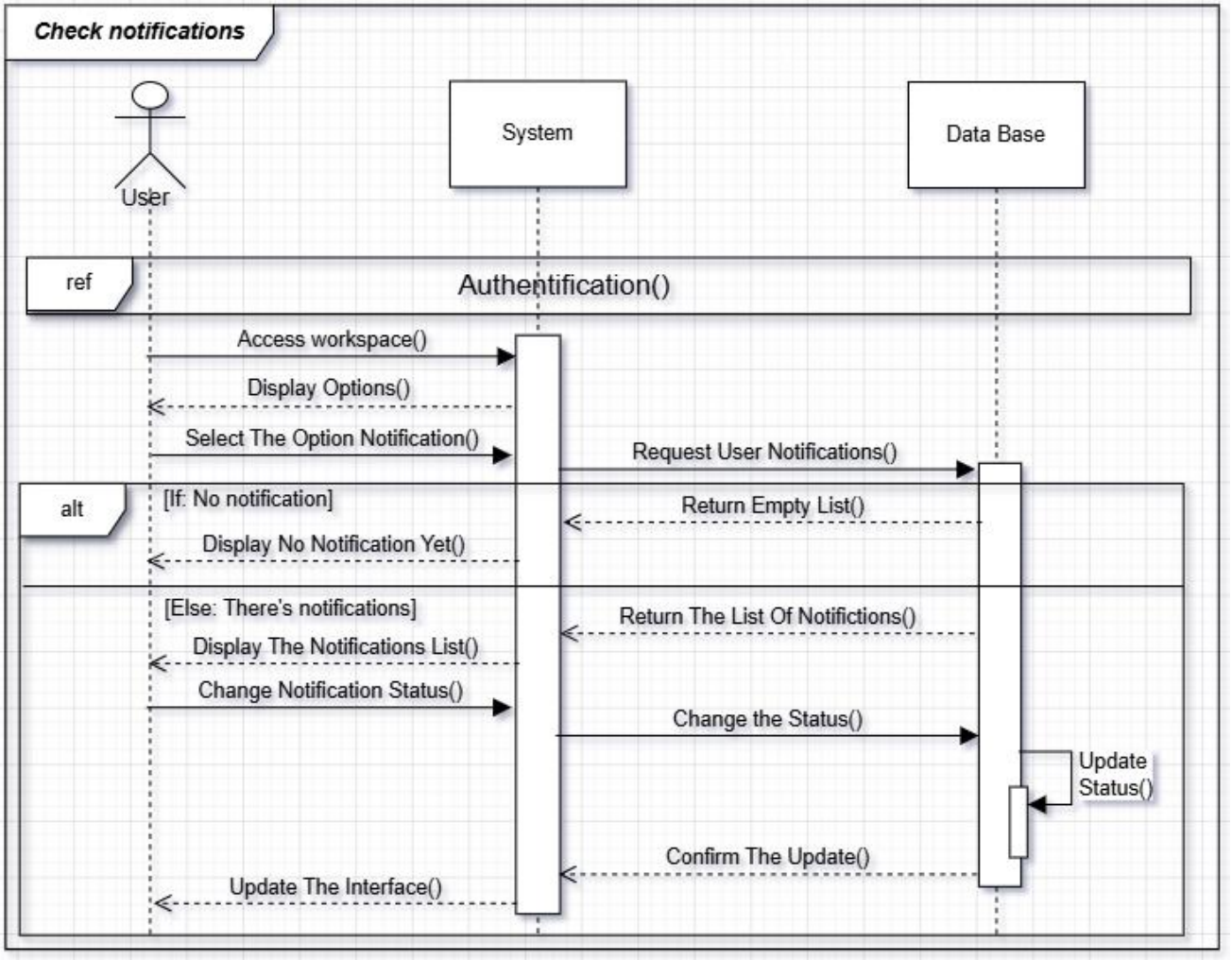


Figure 8: Check notification sequence diagram.

2.6.2 Manage requests Sequence diagram: Manage Requests allows a manager or president to view, answer, and validate member requests. The system notifies the member:

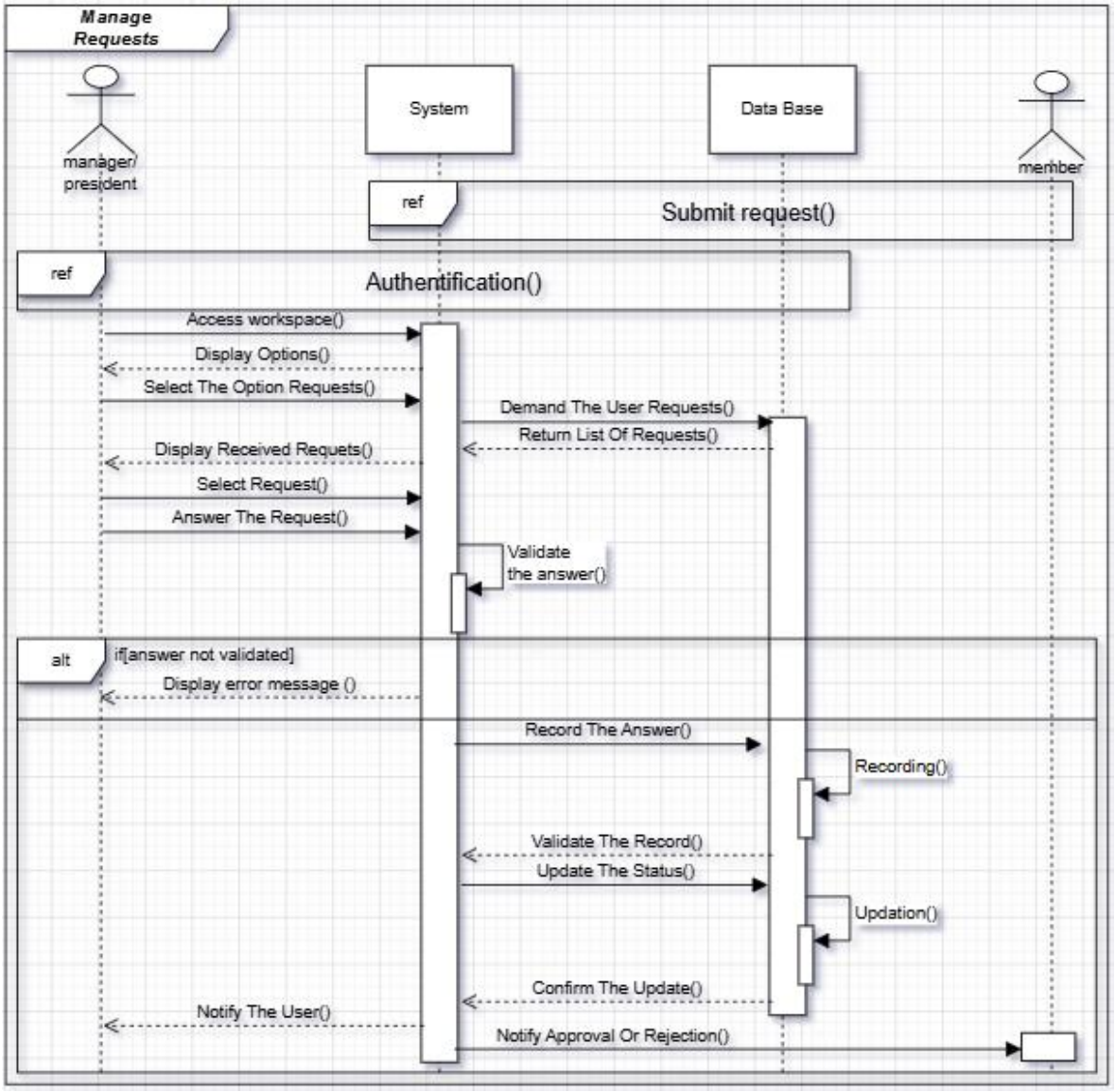


Figure 9: Manage request sequence diagram.

2.6.3 Authentification sequence diagram: Authentication verifies a user's identity by checking their entered username and password against the database, allowing access only if the information is correct:

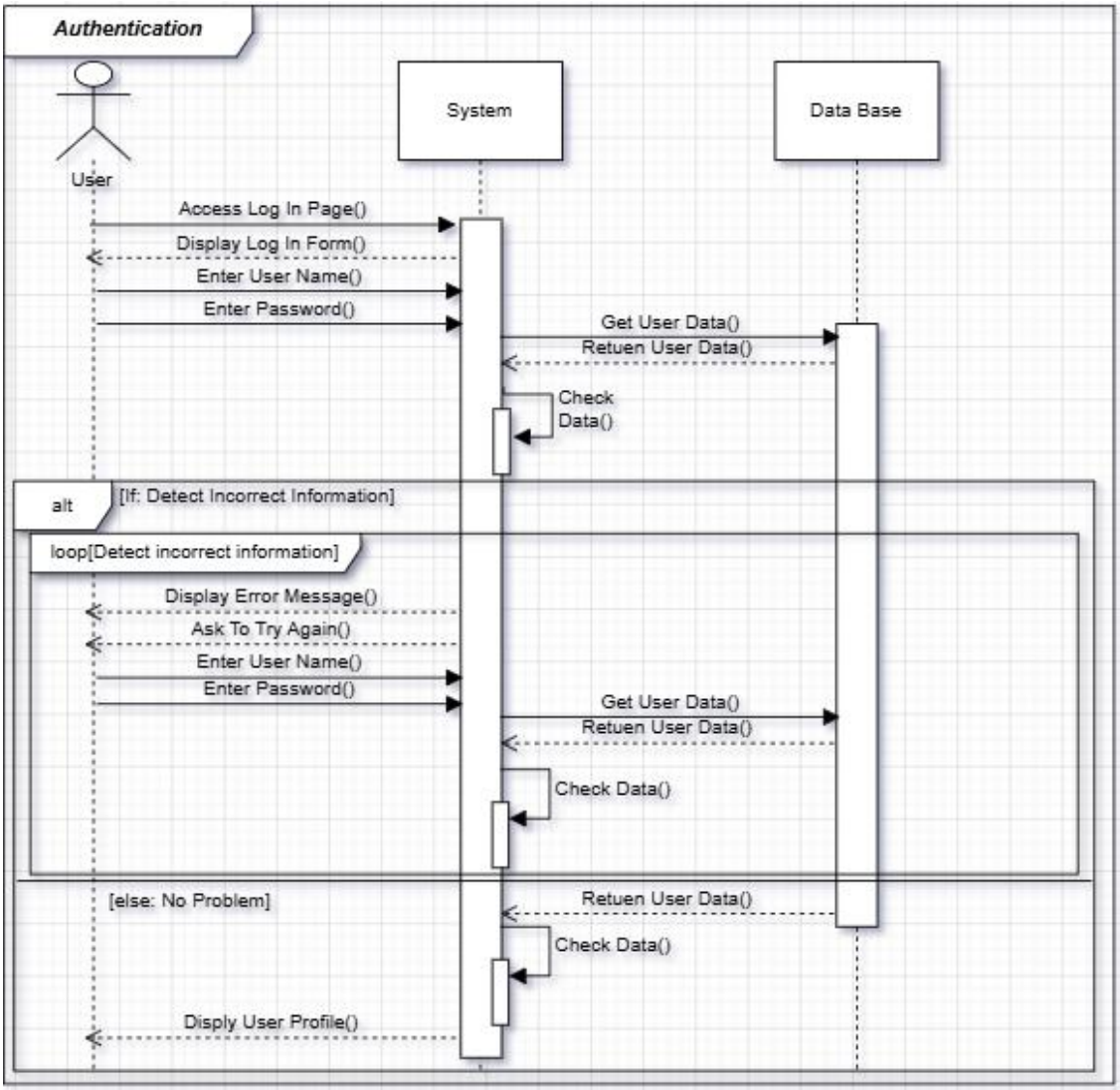


Figure 10: Authentification sequence diagram.

2.6.4 Share post sequence diagram: Share Posts allows an authenticated user to create a post by filling out a form and optionally uploading an image. The system validates the data, stores it in the database, and then updates the posts list for everyone to see:

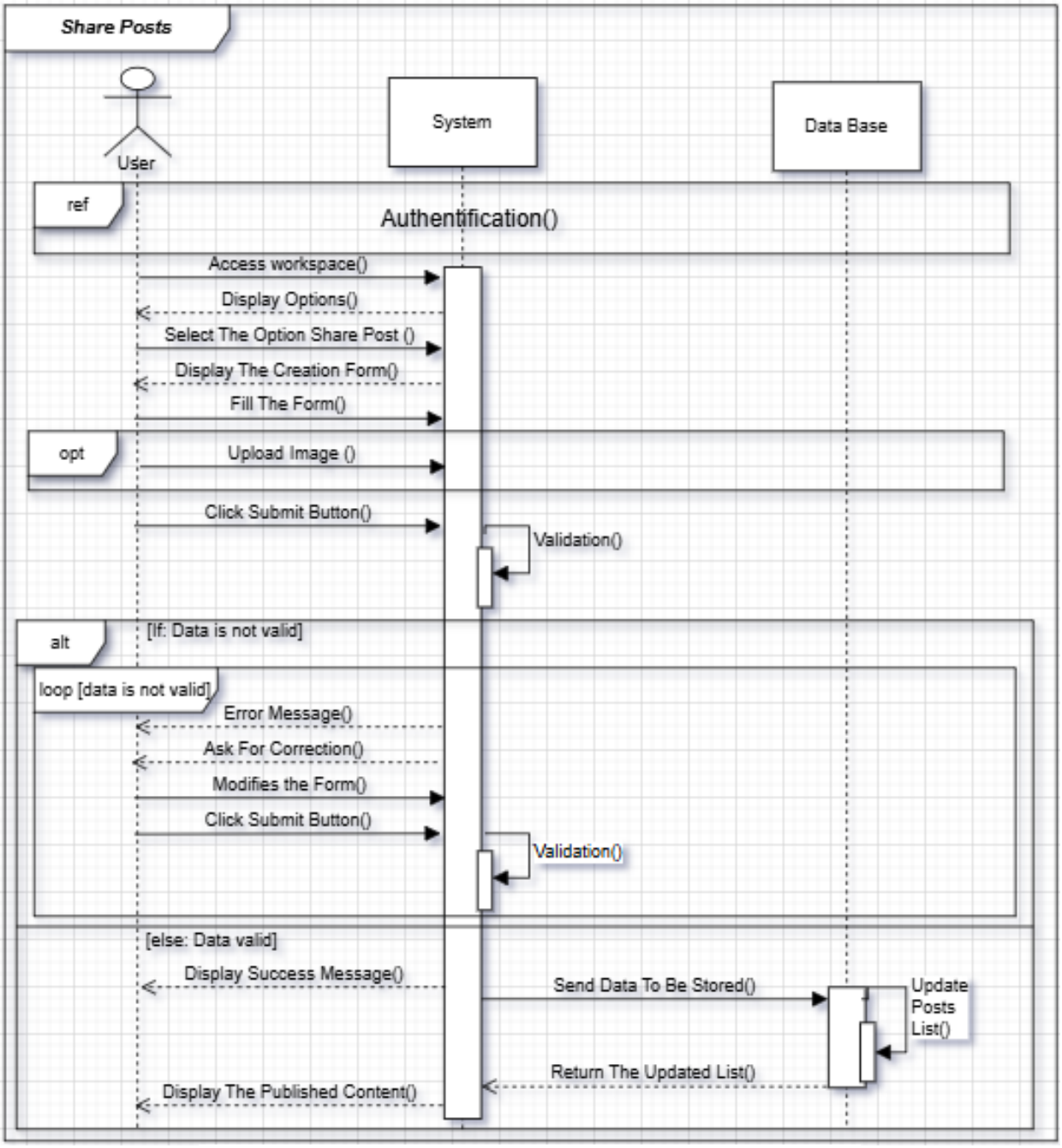


Figure 11: Share post sequence diagram.

2.6.5 Manage members sequence diagram: Manage Department/Club System allows an authorized user (manager or president) to add, remove, or modify member information and roles within the department or club:

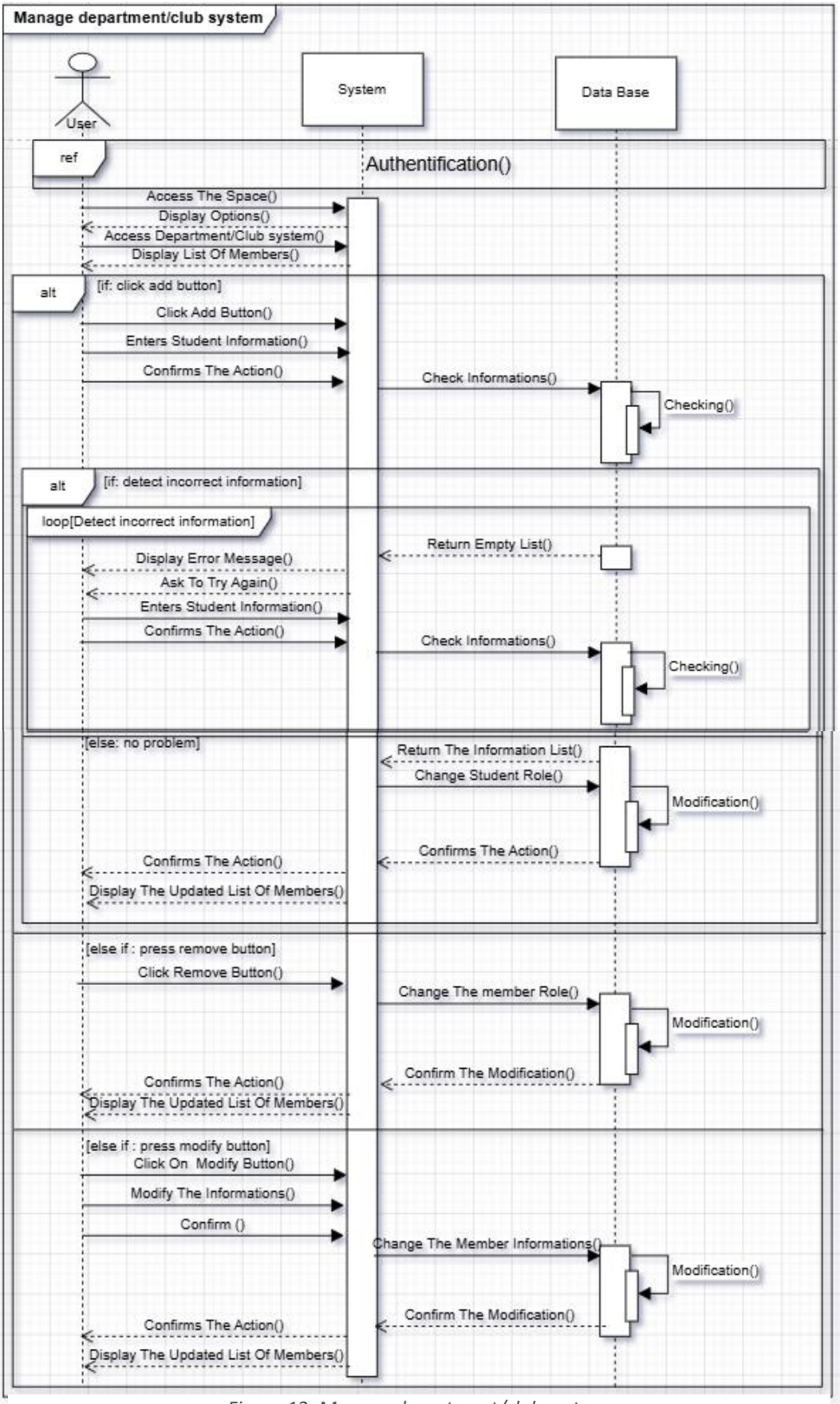


Figure 12: Manage department/club system.

2.6.6 Submit request sequence diagram: Submit Request allows a member to fill out and submit an electronic request form. The system notifies the appropriate manager or president:

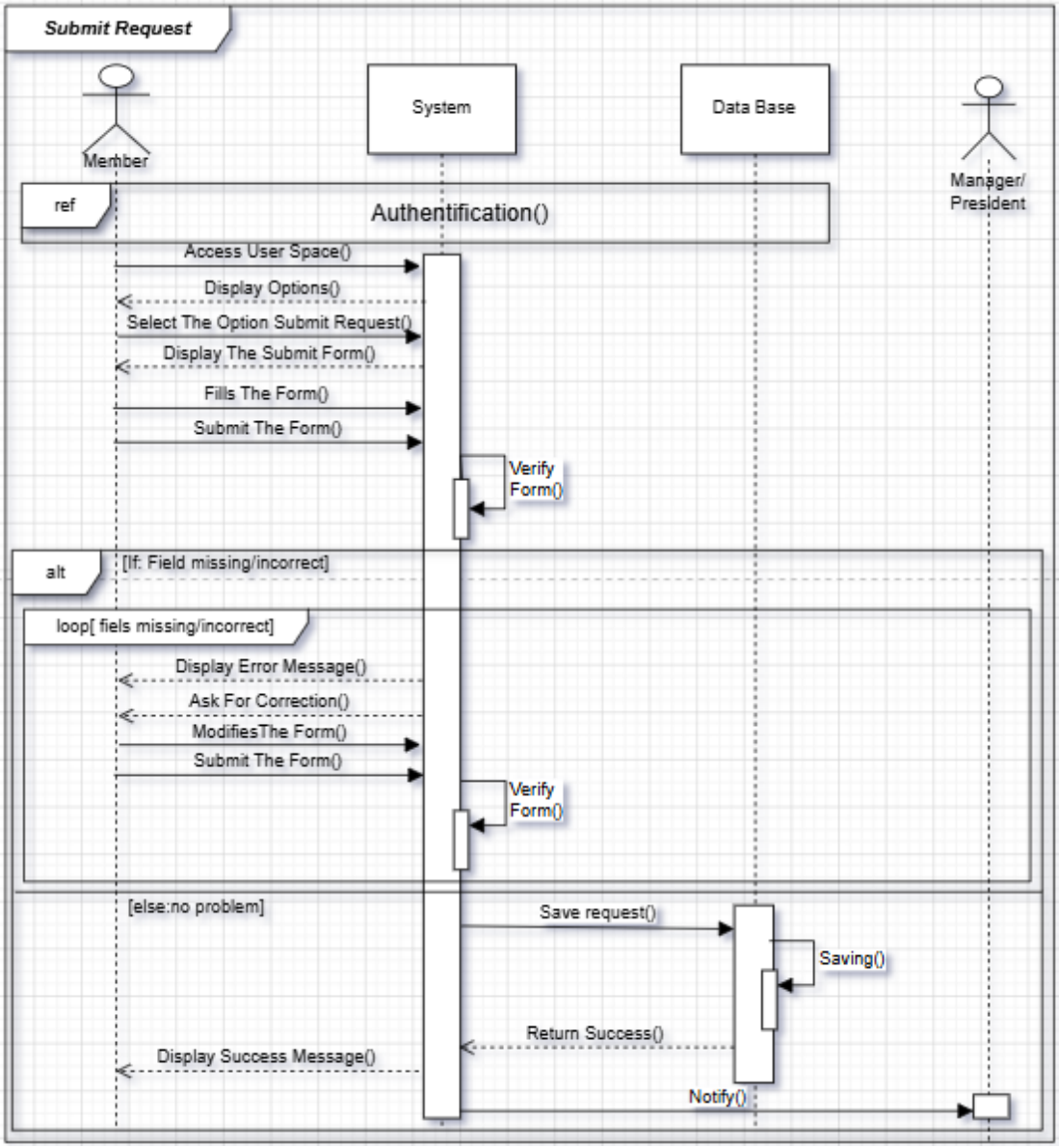


Figure 13: Submit request sequence diagram .

2.7 Class Diagram

A class diagram is a UML diagram that represents the static structure of a system by showing its classes, attributes, methods, and relationships. It illustrates how different parts of the system are connected, including associations, inheritance, and dependencies. It is commonly used in object-oriented design as a blueprint for system implementation.

2.7.1 The data dictionary: The different attributes are described in the table below:

Attribute name	Wording	Type	Length
UserID	The unique identifier of the student	BIGINT	64-bit
Username	The user name	TEXT	VARCHAR(50)
Password	The user password	TEXT	VARCHAR(255)
Email	The student email	TEXT	VARCHAR(100)
Faculty_name	The student's faculty name	TEXT	VARCHAR(100)
Level	The student's level	TEXT	VARCHAR(20)
Role	The user's role in the club	VARCHAR	50
DepartmentID	The Identifier of the department	INTEGER	32-bit
Token_expiry	Expiration time for password reset token	TIMESTAMP	-
Reset_token	Password reset token	TEXT	VARCHAR(255)
Membership_date	The member's membership date	DATE	-
EventID	The identifier of the event	SERIAL	-
Event_title	The title of the event	TEXT	VARCHAR(200)
Event_description	The Description of the event	TEXT	VARCHAR(500)
Event_date	The date of the event	DATE	-
Event_location	The location of the event	TEXT	VARCHAR(200)
RequestID	The identifier of the request	SERIAL	-
Type_request	The type of the request	TEXT	VARCHAR(50)
Details_request	The details of the request	TEXT	VARCHAR(500)
Status_request	The status of the request	TEXT	VARCHAR(20)
Created_at	Timestamp when request was created	TIMESTAMP	-
NotificationID	The identifier of the notification	SERIAL	-

Date_notification	Timestamp when notification was created	TILESTAMP	-
Sender	The sender of the notification	TEXT	VARCHAR(50)
Details_notification	The details of the notification	String	100
Type_notification	The type of the notification	String	10
Status_notification	The status of the notification	String	05
ClubID	The identifier of the club	SERIAL	-
Name_club	The name of the club	TEXT	VARCHAR(100)
Description_club	The description of the club	TEXT	VARCHAR(500)
Creation_date_club	The creation date of the club	DATE	-
WorshopID	The identifier of the workshop	SERIAL	-
Workshop_title	The title of the workshop	TEXT	VARCHAR(200)
Workshop_description	The description of the workshop	TEXT	VARCHAR(500)
Workshop_date	The date of the workshop	DATE	-
Workshop_location	The location of the workshop	TEXT	VARCHAR(200)
PostID	The identifier of the post	SERIAL	-
Type_post	The type of the post	TEXT	VARCHAR(20)
Poster	The poster of the post	BIGINT	64-bit
Date_post	The date of the post	DATE	-
Description_post	The description of the post	TEXT	VARCHAR(500)
Title_post	The title of the post	TEXT	VARCHAR(200)
Image	The image of the post (optional)	TEXT	-
Departmnet_name	The name of the department	TEXT	VARCHAR(100)
Description_department	The description of the department	TEXT	VARCHAR(500)

Sub_departmentID	The Identifier of the sub-department	SERIAL	-
Sub_departmnet_name	The name of the sub-department	TEXT	VARCHAR(100)
Sub_description_department	The description of the sub-department	TEXT	VARCHAR(500)
HiddenAT	Timestamp when request/notification was hidden	TIMESTAMP	-

Table 8: The data dictionary.

2.7.2 System classes: The different classes that make up the system are described in the table below:

Class	Attributes	Methods
User	UserID Username Password Email Faculty_name Level Role token_expiry reset_token DepartmentID	Authentification() Check_notification() Submit_request() View_profile()
Member	Membership_date()	-
Instructor	-	Share_posts()
Manager	DepartmentID	Share_posts() Manage_requests() Manage_department_system()
President	-	Share_posts() Manage_requests() Manage_club_system()
Vice-president	-	-
Event	EventID Event_title Event_description Event_date Event_location	-

Request	RequestID Type_request Details_request Status_request Created_at	-
Notification	NotificationID Time_notification Sender Details_notification Type_notification Status_notification	-
Club	ClubID Name_club Descripton_club Creation_date_club	-
Department	DepartmentID Department_name Description_departem ent	-
Post	PostID Type_post Poster Date_post Description_post Title_post Image	-
Workshop	WorkshopID Workshop_title Workshop_description Workshop_date Workshop_location	-
Sub-department	Sub-departmentID Sub-department_name Description_sub-department	-
UserHiddenRequests	UserID RequestID HiddenAt	-
UserHiddenNotifications	UserID NotificationID HiddenAt	-

Table 9: The system classes.

2.7.3 Management rules: are the rules manipulating the system to guarantee the consistency, integrity and validity of the data handled by the system:

Rule 01) The club is composed of multiple departments.
Rule 02) Each department is managed by only one manager.
Rule 03) A student can be a member in the club and a club has multiple members.
Rule 04) A student can be an instructor, manager, vice-president or a president.
Rule 05) Any student can check his notification, send and receive requests.
Rule 06) The president is the leader of the club.
Rule 07) The instructors are the responsible for creating workshops, and the president can create an event
Rule 08) The managers are the responsible for managing the departments.
Rule 09) Instructors, managers, vice-president and president are allowed to share posts.

2.7.4 Class diagram

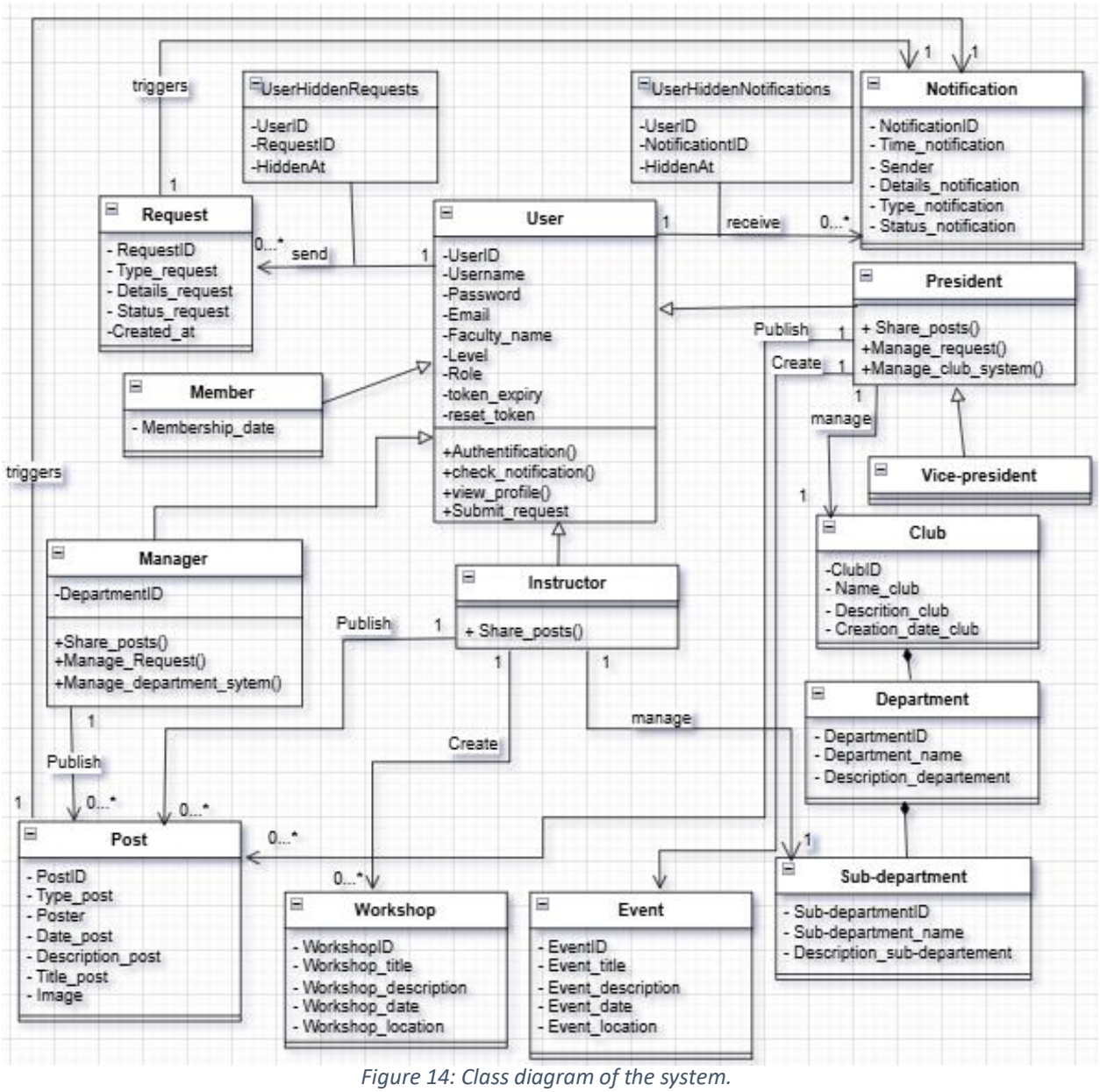


Figure 14: Class diagram of the system.

2.8 Logical Data Model

2.8.1 Rules for transitioning from class diagrams to relational model

To move from a class diagram to the relational model, follow these steps and apply them to each class in the diagram:

- **Class to table:** One table per concrete class. Attributes become columns; identifier becomes primary key.
- **1-to-1:** Add a foreign key in either table with a unique constraint.
- **1-to-many:** Add foreign key in the table on the "many" side.
- **Many-to-many:** Create a junction table with two foreign keys as composite primary key.
- **Inheritance:** Use single table (type column), concrete table (one table per child), or class table (one table per class).
- **Aggregation/Composition:** Map like 1-to-many.
- **Qualified association:** Include the qualifier in the target table's primary key.

2.8.2 The relational model of the system

User: (UserID, Username, Password, Email, Faculty_name, Level, Role, token_expiry, reset_token, # DepartmentID)
Request: (RequestID, Type_request, Details_request, Status_request, Created_at, #UserID).
Notification: (NotificationID, Time, Details_notification, Type_notification, Status_notification, Sender, ,#UserID, #RequestID, #PostID).
Post: (PostID, Type_post, Title_post, Description_post, Date_post, Image, #UserID).
Club: (ClubID, Name_club, Description_club, Creation_date_club).
Event: (EventID, Event_title, Event_description, Event_date, Event_location, #UserID).
Workshop: (WorkshopID, Workshop_title, Workshop_description, Workshop_date, Workshop_location,#UserID).
Department: (DepartmentID, Department_name, Description_department, #ClubID).
Sup-department: (Sub-depatmentID, Sub-department_name, Description_sub-department,
#DepartmentID)
Member: (#UserID, # Sub_departmentID, Membership_date).
Instructor: (#UserID, # Sub_departmentID)
Manager: (#UserID, #DepartmentID).
President: (#UserID, #ClubID).
Vice-president: (#UseID,# ClubID).
UserHiddenRequests: (#UserID, #RequestID, HiddenAt)
UserHiddenNotifications: (#UserID, #NotificationID, HiddenAt)

Conclusion

This chapter provided a complete technical design for the club management system. We defined actors, use cases, sequence diagrams, and the class diagram with its data dictionary. We then mapped the class diagram to a relational model, ready for database implementation. The next chapter will cover implementation and testing.

Chapter 03:

Realization

Introduction

This chapter explains how the platform was built. It describes the tools, technologies, and environment used to develop the system, screenshots of the actual website are included to show how each interface looks and works. Finally, it covers the security measures implemented to protect user data and ensure the platform runs smoothly and reliably.

3.1 Environment and development tools: Development environments and tools are essential for building websites, as they provide the necessary framework and features for design, coding, testing, and deployment. They simplify and streamline the entire development process:

3.1.1 Environment materiel: For the completion of this project, we have provided this computer with the following specifications:

1- Processor: Intel(R) Core(TM) i5-8250 CPU @01.60GHz.
2- Memory: 4.00 of Ram.
3- Hard Drive: 512GO.
4- Operating System: Windows 10.
5- System type: 64-bit operating system, x64 processor.

3.1.2 Presentation of the graphical interfaces

1- Visual Studio Code



A lightweight and powerful code editor. It supports multiple programming languages and offers extensions to enhance functionality. It is widely used by developers for its speed and flexibility.

2- pgAdmin4 / PostgreSQL



PostgreSQL is a free, open-source relational database management system known for its reliability and advanced SQL features. pgAdmin4 is a graphical tool that makes it easy to manage PostgreSQL databases without using the command line. Together, they provide a complete environment for developing and managing databases efficiently.

3- HTML/CSS/Java Script



Technologies for building modern websites. HTML (Hyper Text Markup Language) is the foundation of any web page. It structures the content by defining elements such as headings, paragraphs, images, links, forms, and multimedia. HTML gives the page its meaning and organization.



CSS (Cascading Style Sheets) controls the visual appearance of the page. It manages colors, fonts, spacing, layout, and responsive design to make the page attractive and user-friendly.



JavaScript is a programming language that adds interactivity and dynamic behavior. It allows the page to respond to user actions like clicks, form submissions, and keyboard input. JavaScript can also fetch data from servers and update the page without reloading.

4- Node.JS

Is an open-source, cross-platform runtime that allows you to run JavaScript outside the browser, it is built on Chrome's V8 engine and uses an event-driven, non-blocking architecture, this makes it efficient for building fast, scalable server-side and network applications.

3.2 Presentation of the application: In this section, we will present some graphical interfaces of our web application:

In addition to the explore interface for the visitor and the log in and welcome interfaces, our application contains five personal spaces accessible by the main actors (president/vice president, managers, instructors and members).

3.2.1 Explore as guest page: This is the Explore as Guest page, where a visitor can view the club's departments and sub-departments. A guest can also submit a join request to any sub-department by clicking the "Join Department" option fill the form and click "Submit", the system sends the request to the appropriate department manager, also the guest can view limited posts without logging in:

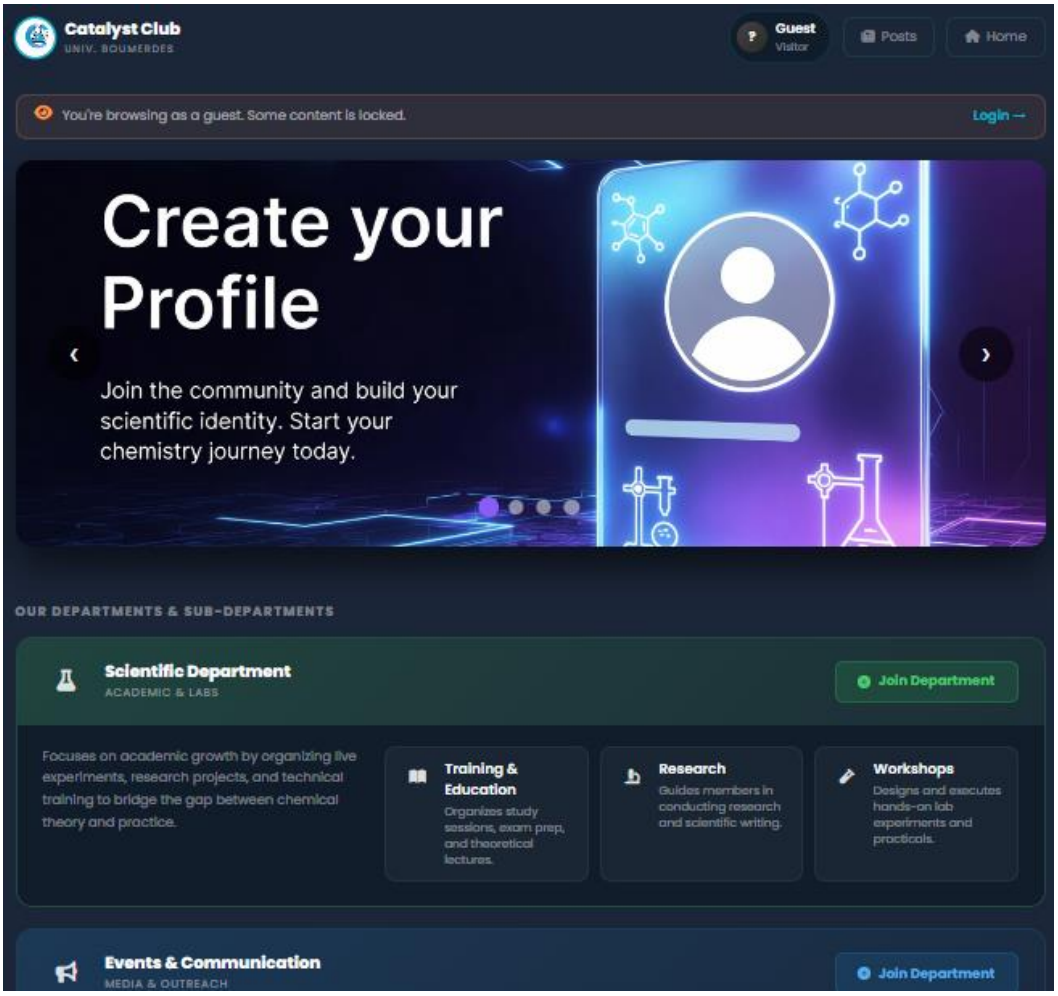


Figure 15: Explore as guest page.

3.2.2 Reply to join request: This is the approval email sent to the user after a manager approves their join request. It includes the manager's response, the user's login

credentials (username and password), and a recommendation to change the password after first login.

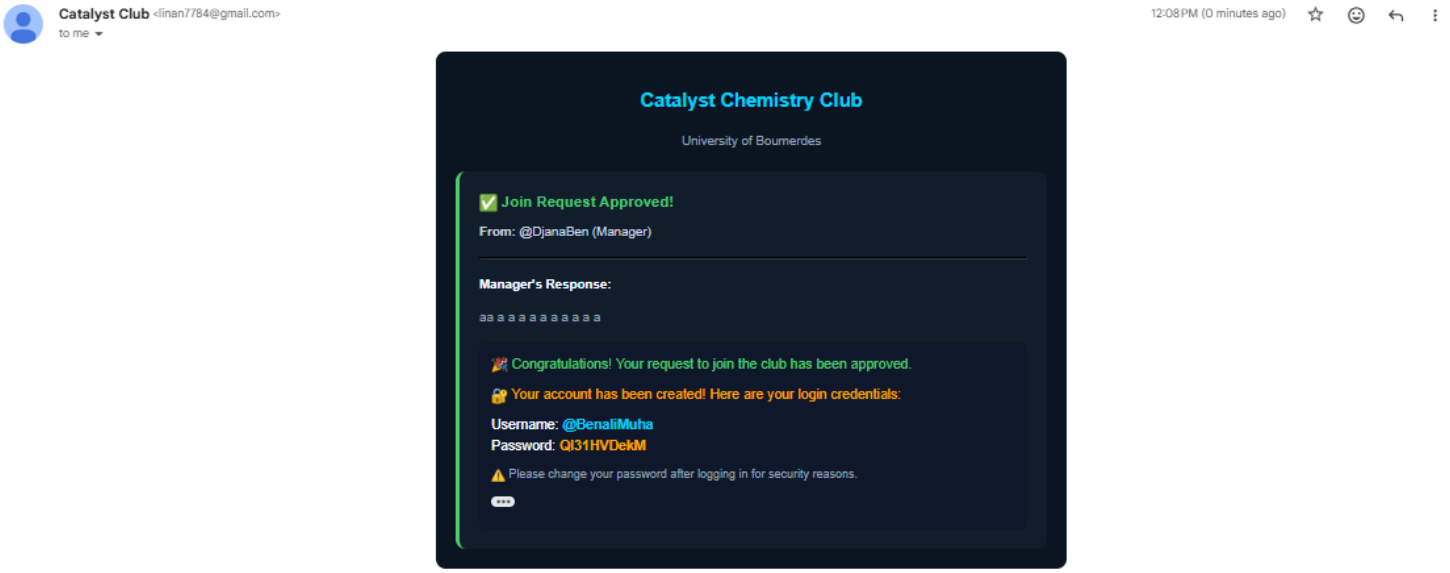


Figure 16: Reply to join request.

3.2.3 President profile page: This is the president profile page, where the president can view their personal information, role, and posted content. From this dashboard, the president can also access notifications, manage requests, share posts, and manage the club system:

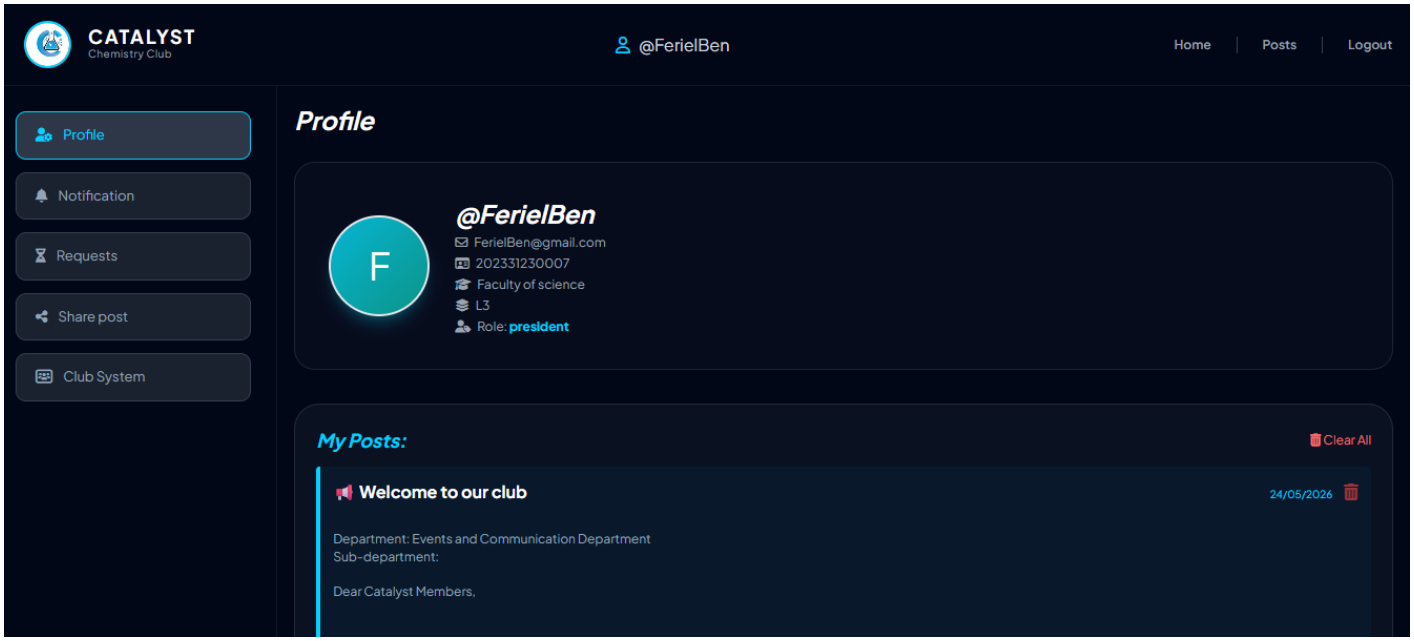


Figure 17: President profile page.

- **Notification:** The president only receives notifications when a manager or instructor shares a post. This allows the president to track all posts shared by members without having to check the "Posts" section every time.

- **Requests:** Here, the president can view and reply to requests sent by members, such as Nomination Request, Event Idea Request, Join Event Request (any member can send this request), and Manager Withdraw Request (sent only by a manager), the reply will be sent to the appropriate member as a notification.

- **Share Post:** This option is available to the president, managers, and instructors. They can post announcements, events, or workshops, and all members can see these posts.

- **Club System:** This page allows the president to view all club members and their information, such as username, registration number, role, and faculty.. . The president can add, modify, or remove a manager, and each action is followed by an automatic notification to the affected manager. The president can also withdraw from their position by changing their personal information to the member who will inherit the role. Additionally, the president can view statistical information such as the total number of members and departments.

3.2.4 Manager profile page:

This is the Manager Profile page, where a manager can view their personal information, active roles, and assigned department the requests he submitted and posts he shared. From this dashboard, the manager can also access notifications, manage requests, submit requests, share posts, and manage their department system:

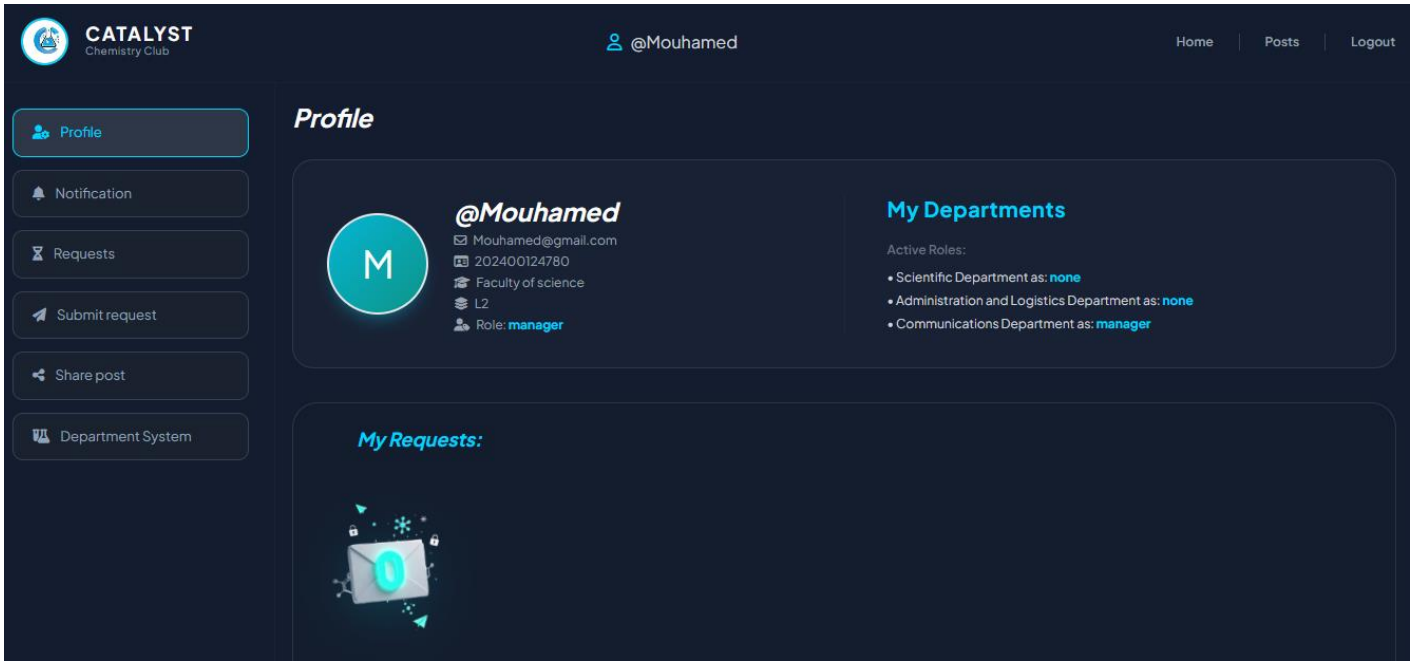


Figure 18: Manager profile page.

- **Notification:** In this section, the manager can view notifications sent by the president, either when the president shares a post or replies to a request from the manager.

- **Requests:** Here, the manager can view and reply to requests sent by three types of users: guests (join requests), members, or instructors (withdraw requests).

- **Submit request:** the manager can send requests to the president (the same we mentioned earlier in the president profile page).

- **Department system:** In this section, the manager can view all information about members of their department. They can add, remove, or modify members and instructors, search for a member by registration number or username, and view statistics such as the number of members, departments, and sub-departments.

3.2.5 Instructor profile page: it looks exactly like the manager profile page but with less features in the side bar:

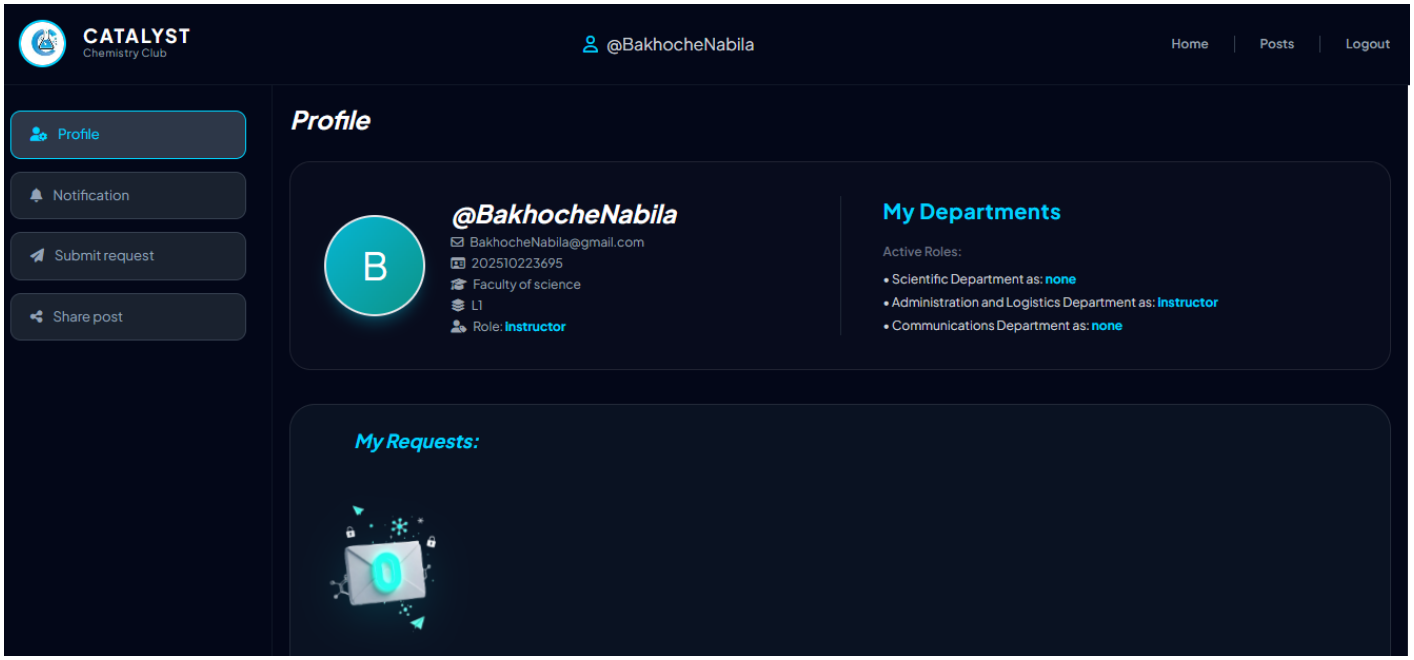


Figure 19: Instructor profile page.

- **Notification:** In this section, the instructor can see replies to their requests. Notifications are also sent to the instructor whenever the president or their department manager shares a post.

- **Submit Request:** In this section, the instructor can submit requests to the manager or president, using the same request types mentioned earlier in the previous pages.

3.2.6 Member profile page: Is the simpler profile that looks like the previous ones but with less features:

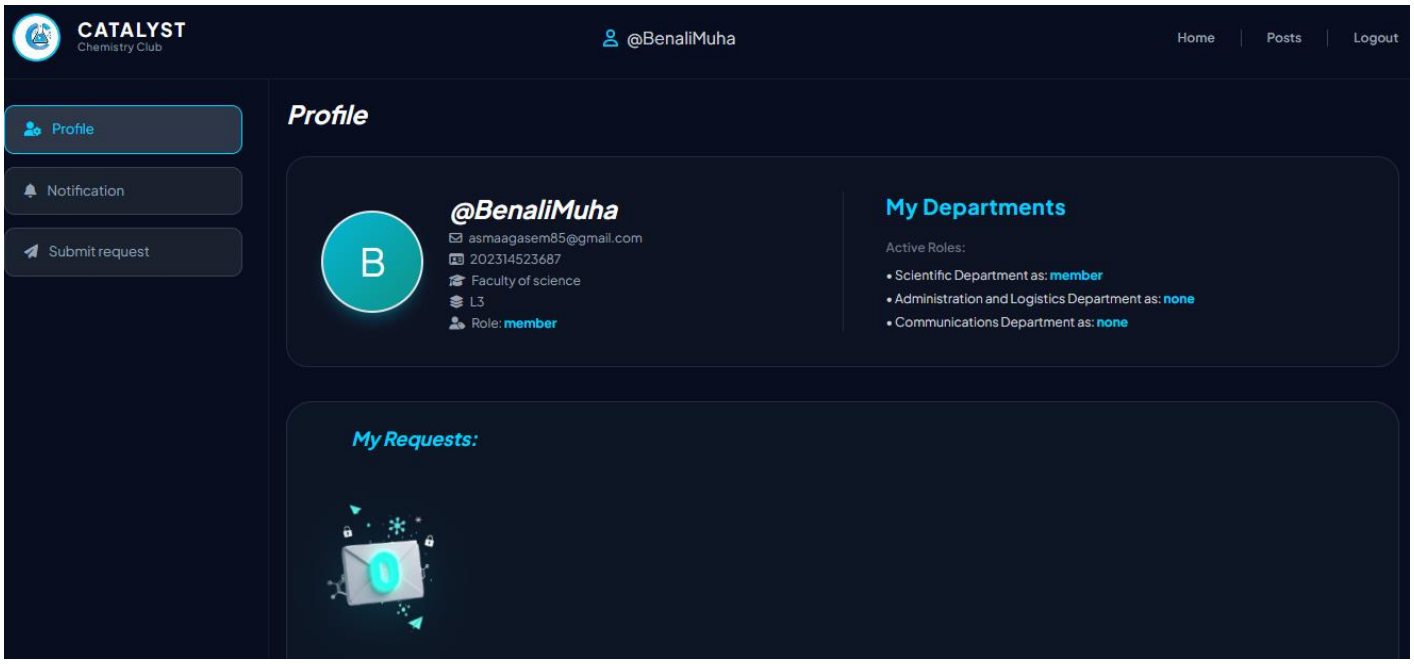


Figure 20: Member profile page

3.3 **Security features:** To protect user data and ensure the platform runs safely, the following security measures were implemented:

1- Validations

- The API enforces strict input validation on all user data. This includes password strength, student ID format, username format, email format, and minimum word counts for text fields.
- The system verifies that the provided registration number (ID) and username match in the database to confirm the user's identity.
- Additionally, the system checks that a user is already a club member before being promoted to manager or instructor, and ensures that a department does not already have a manager before adding a new one, and that a sub-department does not already have an instructor before assigning one.
- These validations ensure data integrity and prevent invalid or malicious input from being processed.

2- SQL Injection Prevention

- All database queries use parameterized inputs instead of string concatenation

3- Login validation

- Username and password are required and verified against hashed password in database

4- Role-Based Access Control (RBAC)

- Each endpoint checks if the user has permission

5- Duplicate Prevention

- **Email uniqueness:** Cannot register with an email already in use
- **User ID uniqueness:** Cannot register with an ID that already has an account
- **Instructor uniqueness:** Only one instructor per sub-department
- **Manager uniqueness:** Only one manager per department

6- File Upload Security (Images)

- **File size limit:** 5 MB maximum
- **Allowed file types:** Only JPEG, JPG, PNG, GIF
- File extension and MIME type are both validated

7- Password Reset Security

- **Reset token:** Generated using crypto. randomBytes(32)– cryptographically secure
- **Token expiry:** 1hour expiration time
- **Two-step verification:** Requires both email AND username to match before generating reset link
- Token is cleared after password reset

8- Notification and Request Security

- **Hidden requests/notifications:** Soft delete system (data is not permanently deleted)
- **Request status tracking:** Each request has a status.

9- Automatic Password Generation for New Members

- **Random password generation:** Combines letters and numbers, then shuffled for randomness
- **Password sent via email:** with recommendation to change after first login

10- Membership Date Verification (Nominations)

- When a member submits a nomination request, the system verifies their membership date matches the database record to prevent fraud

11-Cross-Origin Resource Sharing (CORS)

- middleware is enabled to control which domains can access the API

Conclusion

In this chapter, we have presented the complete development process of the platform. We described the development environment, the technologies and tools used, and showed the final web application through clear screenshots of the interfaces. We also explained the important security measures put in place to safeguard user information and maintain system reliability.

General conclusion

This project aimed to build a website to help manage the Catalyst Chemistry Club. After studying how the club works, we found several problems. Communication was poor, there was too much paperwork, and no digital system existed to track members or events. Students had to come to the university just to check simple information, and member data was stored on paper, which is not secure.

In the first chapter, we looked at the club's structure and talked with the president. We learned about the main issues, such as hard-to-manage requests, slow nomination processes, and no easy way to share events. This helped us set clear goals for the project.

In the second chapter, we designed the system. We listed what the system must do, like replacing paper forms with digital ones, sending notifications, and allowing members to submit requests online. We also drew diagrams to show how users would interact with the system. This chapter gave us a clear plan to follow.

In the third chapter, we built the website. We used tools like HTML, CSS, JavaScript, Node.js, and PostgreSQL. We created different pages for each type of user: visitors, members, instructors, managers, and the president. Each user can do only what their role allows. We also added security features to protect user data, such as login checks, safe password handling, and file size limits for images.

Finally, the website works as we planned. It removes the need for paper, makes communication faster, and helps club organizers manage members and events easily. Students can now send requests, check responses, and see upcoming events from anywhere. This project shows that a simple website can solve real problems and make student clubs run better.

Bibliography

1. Muller, R. J. (1999). *Database Design for Smarties: Using UML for Data Modeling***.** Morgan Kaufmann.
2. Halpin, T., & Morgan, T. (2024). *Information Modeling and Relational Databases*. Morgan Kaufmann.
3. Yan, Y. (2025). *Web Programming with Modern Full Stack Technology*. Amazon.
4. Pressman, R. (2014). *Software Engineering: A Practitioner's Approach* (8th ed.). McGraw-Hill.
5. Hundhausen, C. D. (2026). *Full-Stack Web Development from the Ground Up: Principles, Practices, and Technologies*. Elsevier.

Webography

1. draw.io for all the UML diagrams: https://www.diagrams.net
2. npm for the installed packages: https://npmjs.com
3. W3Schools for tutorials and references: https://www.w3schools.com
4. JavaScript tutorial: https://www.youtube.com/watch?v=EerdGm-ehJQ