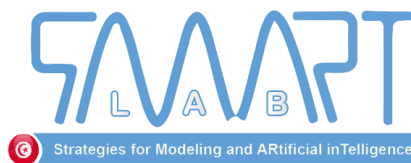


Tunisian Republic
Ministry of Higher Education and Scientific Research
University of Tunis
Higher Institute of Management of Tunis
IT Department

Graduation Project Report
In Business Computing
Specialty: Business Information System

Conception & Development of SMARTLAB
Web Application

Host Organization



Framed By
Anouer Bennajeh

Elaborated By
Jihen Aridhi
Roudeina Hamdi

Dedication

To my parents

Throughout my life's journey, you have been my unwavering pillars of support, my guiding lights, and my greatest source of love and inspiration. Your unconditional love, sacrifices, and constant encouragement have stayed with me throughout my studies.

To my childhood friend Nour

As we have grown and evolved, our bond has only deepened. You have been my companion, confidant, and partner in countless adventures. I am grateful for your help and support during my graduation project.

To my project partner Jihen

I want to take a moment to express my deepest appreciation for your exceptional partnership throughout this project. Our collaboration has not only been productive but also enjoyable. I am truly grateful to have had the opportunity to work with you as my project partner.

Roudeina Hamdi

Dedication

This journey would not have been possible without having the dearest people to my heart constantly by my side.

To my beloved parents and siblings

You have been the pillars of my life journey. Without you, I wouldn't have been the person that I am today, and it's all because of the support and easement you provided for me, not only throughout my education, but also throughout various sides of my life. I would love to express my absolute appreciation towards you, for your unconditional love and support and patience, and more importantly for being such an inspiration and guidance to me, and I will keep on admiring you wholeheartedly.

To my project partner Roudeina

I would like to express my gratitude towards you, for your collaborative spirit and your enjoyable company, and especially for what we have learned together throughout the past few years. And I am grateful for the experience we have gained together.

Jihen Aridhi

Acknowledgment

Before any development of this professional experience, it seems appropriate to begin this report with thanks to those who taught us a lot during this internship.

We would like to express our heartfelt thanks and deep gratitude to our academic and professional supervisor Mr. Anouer Bennajeh for his skills, his professional qualities, his availability and above all his advice, he never stop helping and encouraging us during the realization of this graduation project.

We would like to thank all the SMARTLAB laboratory managers for their help with great sincerity and gratitude.

On our part, we hope that our conduct and our learning have left a good impression of the ISG and affirms its image.

TABLE OF CONTENT

GENERAL INTRODUCTION **1**

CHAPTER ONE: INTRODUCTION **2**

INTRODUCTION **2**

- 1. PROJECT FRAMEWORK 2
- 2. ANALYSIS OF THE EXISTING 3
- 3. CONSIDERED SOLUTION 4
- 4. APPLICATION DEVELOPMENT METHODOLOGY 5

CONCLUSION **9**

CHAPTER TWO: PLANNING AND ARCHITECTURE **10**

INTRODUCTION **10**

ACTORS IDENTIFICATION **10**

SPECIFICATION OF NEEDS **11**

- 1. FUNCTIONAL REQUIREMENTS 11
- 2. NON-FUNCTIONAL REQUIREMENTS 14

TEAMS AND ROLES **15**

PRODUCT BACKLOG **16**

STRUCTURING IN SPRINTS AND PLANNING **20**

CONCLUSION **20**

CHAPTER THREE: SPRINT 1: BUILDING FRONT OFFICE **21**

INTRODUCTION **21**

SPRINT BACKLOG **21**

FUNCTIONAL SPECIFICATION **22**

ANALYSIS **22**

- 1. USE CASE DIAGRAM 22
- 2. TEXTUAL DESCRIPTION 23
- 3. SYSTEM SEQUENCE DIAGRAM 24

CONCEPTION	24
1. CLASS DIAGRAM	24
2. DETAILED SEQUENCE DIAGRAM	25
IMPLEMENTATION	26
1. RELATIONAL SCHEMA	26
.2 INTERFACES	27
CONCLUSION	29

CHAPTER FOUR: SPRINT 2: BUILDING MEMBERS BACK OFFICE

INTRODUCTION	30
SPRINT BACKLOG	30
FUNCTIONAL SPECIFICATIONS	32
ANALYSIS	33
1. USE CASE DIAGRAM	33
2. TEXTUAL DESCRIPTION	35
3. SYSTEM SEQUENCE DIAGRAM	38
CONCEPTION	43
1. CLASS DIAGRAM	43
2. DETAILED SEQUENCE DIAGRAM	46
IMPLEMENTATION	51
1. RELATIONAL SCHEMA	51
2. INTERFACES	51
CONCLUSION	56

CHAPTER FIVE: SPRINT 3: BUILDING ADMINISTRATOR BACK OFFICE

INTRODUCTION	57
SPRINT BACKLOG	57
FUNCTIONAL SPECIFICATIONS	60
ANALYSIS	60
1. USE CASE DIAGRAM	61
2. TEXTUAL DESCRIPTION	62
3. SYSTEM SEQUENCE DIAGRAM	67
CONCEPTION	71
1. CLASS DIAGRAM	71
2. DETAILED SEQUENCE DIAGRAM	73

IMPLEMENTATION	80
1. RELATIONAL SCHEMA	80
2. INTERFACES	81
CONCLUSION	84
ARCHITECTURE & DEVELOPMENT ENVIRONMENT	85
INTRODUCTION	85
1. PROJECT ARCHITECTURE	85
2. SOFTWARE DEVELOPMENT ENVIRONMENT	89
CONCLUSION	91
<u>GENERAL CONCLUSION</u>	<u>92</u>

TABLE OF FIGURES

Figure 1: SMARTLAB Logo	2
Figure 2: General Use Case Diagram	12
Figure 3: General Class Diagram	13
Figure 4: Netizen's Use Case Diagram	22
Figure 5: "Registrate" Use Case Diagram	22
Figure 6: "Registrate" System Sequence Diagram	24
Figure 7: "Registrate" Class Diagram	24
Figure 8: Sprint 1 Relational Schema	26
Figure 9: "Registrate" Interface	28
Figure 10: Member's Use Case Diagram	32
Figure 11: "Manage Account" Use Case Diagram	33
Figure 12: "Manage News" Use Case Diagram	33
Figure 13: "Manage Events" Use Case Diagram	34
Figure 14: "Manage Articles" Use Case Diagram	34
Figure 15: "Authentication" System Sequence Diagram	38
Figure 16: "Update Account" System Sequence Diagram	39
Figure 17: "Update Event" System Sequence Diagram	40
Figure 18: "Add News" System Sequence Diagram	41
Figure 19: "Delete Article" System Sequence Diagram	42
Figure 20: "Authentication" Class Diagram	43
Figure 21: "Manage Account" Class Diagram	43
Figure 22: "Manage News" Class Diagram	44
Figure 23: "Manage Events" Class Diagram	44
Figure 24: "Manage Article" Class Diagram	45
Figure 25: "Authentication" Detailed Sequence Diagram	46
Figure 26: "Update Account" Detailed Sequence Diagram	47
Figure 27: "Add News" Detailed Sequence Diagram	48
Figure 28: "Update Event" Detailed Sequence Diagram	49
Figure 29: "Delete Article" Detailed Sequence Diagram	50
Figure 30: Sprint 1 Relational Schema	51
Figure 31: "Authentication" Interface	51
Figure 32: "Update Account" Interface	53
Figure 33: "Add News" Interface	54
Figure 34: "Update Event" Interface	55
Figure 35: "Delete Article" Interface	55
Figure 36: Administrator's Use Case Diagram	60

Figure 37: "Manage Partners" Use Case Diagram	61
Figure 38: "Manage Project" Use Case Diagram	61
Figure 39: "Manage Members" Use Case Diagram	61
Figure 40: "Reporting» Use Case Diagram	62
Figure 41: "Add Partner" System Sequence Diagram	67
Figure 42: "Delete Partner" System Sequence Diagram	68
Figure 43: "Update Project" System Sequence Diagram	68
Figure 44: "Accept Membership" System Sequence Diagram	69
Figure 45: "Update Teams" System Sequence Diagram	69
Figure 46: "Delete Member" System Sequence Diagram	70
Figure 47: "Reporting" System Sequence Diagram	70
Figure 48: "Manage Partners" Class Diagram	71
Figure 49: "Manage Project" Class Diagram	71
Figure 50: "Manage Members" Class Diagram	72
Figure 51: "Reporting» Class Diagram	72
Figure 52: "Add Partner" Detailed Sequence Diagram	73
Figure 53: "Delete Partner" Detailed Sequence Diagram	74
Figure 54: "Update Project" Detailed Sequence Diagram	75
Figure 55: "Accept Membership" Detailed Sequence Diagram	76
Figure 56: "Update Teams" Detailed Sequence Diagram	77
Figure 57: "Delete Member" Detailed Sequence Diagram	78
Figure 58: "Reporting" Detailed Sequence Diagram	79
Figure 59: Sprint 3 Relational Schema	80
Figure 60: "Add Partner" Interface	81
Figure 61: "Delete Partner" Interface	81
Figure 62: "Edit Project" Interface	82
Figure 63: "Update Project" Interface	82
Figure 64: "Accept Membership Request" Interface	83
Figure 65: "Update Profession"/ "Update Teams"/ "Update Privileges"/ "Delete Member" Interface	83
Figure 66: "Reporting" Interface	83
Figure 67: MVC Architecture	85
Figure 68: Deployment Diagram	88
Figure 69: Diagrams.net/Draw.io Logo	89
Figure 70: XAMPP Logo	89
Figure 71: MySQL Logo	89
Figure 72: IntelliJ IDEA Logo	90
Figure 73: PhpStorm Logo	90
Figure 74: Angular Logo	90
Figure 75: Symfony Logo	91
Figure 76: GitHub Logo	91

TABLE OF TABLES

Table 1: Classic Method VS Agile Method	6
Table 2: Advantages & Disadvantages of Scrum	8
Table 3: Actors Identification	10
Table 4: Actors & Functional Requirements	11
Table 5: Scrum Team and Roles	15
Table 6:Product Backlog	16
Table 7: Sprints & User Stories	20
Table 8: Sprint 1 Backlog	21
Table 9: "Registration" Textual Description	23
Table 10: Sprint 2 Backlog	30
Table 11: "Authentication" Textual Description	35
Table 12: "Update Account" Textual Description	35
Table 13: "Add News" Textual Description	36
Table 14: "Update Event" Textual Description	36
Table 15: "Delete Article" Textual Description	37
Table 16: Sprint 3 Backlog	57
Table 17: "Add Partner" Textual Description	62
Table 18: "Delete Partner" Textual Description	62
Table 19: "Update Partner" Textual Description	63
Table 20:"Accept Membership" Textual Description	64
Table 21: "Update Privileges" Textual Description	64
Table 22: "Update Teams" Textual Description	65
Table 23: "Delete Member" Textual Description	65
Table 24: "Reporting" Textual Description	66
Table 25: MVC architectural pattern	86

Abbreviations

API: *Application Programming Interface*

MVC: *Model View Controller*

UI: *User Interface*

XP: *Extreme Programming*

General Introduction

In today's rapidly advancing digital age, innovation is driving change across all industries, including research and development. To keep up with the evolving needs of the scientific community, laboratories are increasingly turning towards modern technological solutions to facilitate their work. One such solution is the creation of websites that act as virtual portals for information sharing.

In a field as dynamic and constantly evolving as research, it is essential to have a means of communicating new ideas and discoveries quickly and efficiently. A website provides a perfect platform for sharing information about ongoing research projects, scientific publications, and other relevant data. A laboratory website can enhance the visibility and reputation of the laboratory, attract new researchers, and improve its overall scientific impact. By providing a well-curated collection of research and scientific knowledge, the website can establish the laboratory as a credible and authoritative source of information within its field.

Our final year project takes place in this context, aiming to establish a website that enables the dissemination of news, events, projects, and articles by researchers, in order to share new ideas and discoveries as widely as possible. Additionally, it incorporates a back-office system, to manage laboratory members and their articles efficiently.

In this report, we will present the different steps that led to the completion of this application. Essentially, this report is structured around six chapters. In the first chapter, "Introduction" we will introduce our project and place it in its general context. This chapter will cover the internship framework, an analysis of the existing situation, and the work methodology. In the second chapter, "Planning and Architecture," we will specify the functional and non-functional requirements of our project and develop the overall use case diagram. The third, fourth, and fifth chapters correspond to sprint 1, sprint 2, and sprint 3, respectively. In these chapters, we will describe the conceptual part of our project by presenting the different actors, as well as some interfaces of our future work. Finally, the sixth chapter, "Architecture and Development Environment" will focus on the tools used, programming languages, and the overall architecture presentation of our project. In conclusion, we will mention the project's perspectives.

Chapter One: Introduction

Introduction

This introductory chapter is divided four main sections. First, presenting the hosting organisation SMARTLAB. Second, analysing its existing. Third, discussing possible solutions. And lastly, presenting the adopted methodology for the development of this project.

1. Project Framework

1.1. Presenting The Host Organisation

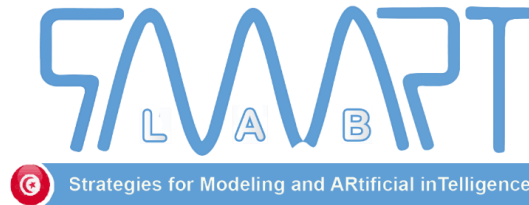


Figure 1: SMARTLAB Logo

Strategies for Modeling and Artificial Intelligence Laboratory (SMARTLAB), is a research laboratory within the Higher Institute of Management (ISG), represented by Mr. Lamjed Ben Said. The SMARTLAB research program develops intelligent solutions using statistical modeling, data science, optimization, and artificial intelligence to address real-world problems in complex environments.

1.2. State of knowledge relating to the issues of the Research Lab

On a national level

At the national level, the concerns of the SMART LAB laboratory align with the overall direction of the national research program aiming for high-quality socio-economic outcomes for Tunisia. Indeed, through the research projects proposed for the period 2019-2022, we aim to effectively leverage theoretical knowledge primarily derived from the fields of Artificial Intelligence, Optimization, and Data Science, combining them with appropriate modelling approaches. The objective is to assist users of the systems/platforms resulting from these

projects in making the best decisions at the right moments in an environment characterized primarily by dynamics, uncertainty, and a large amount of data (Big Data).

On an international level

The research themes of the SMART LAB laboratory are positioned at the intersection of the most advanced and current research activities worldwide. Therefore, at the international level, the SMART LAB laboratory maintains high-quality cooperative relationships with several research laboratories and organizations that are working on these issues.

1.3. Required Activity

Due to the expanding of the research done in the host organisation, it is inevitable that the work environment needs to be developed in order to adapt to the evolving dynamics of the contemporary world, especially that this is an IT laboratory. So, the main goal is to improve the workspace.

2. Analysis of the Existing

2.1. Current Status

The members of this laboratory are currently using public bibliographic platforms in order to publish their work, such as Google scholar, DBLP, ORCID . . .

2.2. Criticising the Existing

Up until now, the laboratory does not have a website to publish their work. As a result, the laboratory administration faces significant challenges when it comes to sharing events or projects, as they aim to attract many participants, but the information is not effectively disseminated.

Furthermore, users are unable to benefit from the articles published by the researchers at SMARTLAB because they struggle to identify the laboratory members and find their articles across numerous platforms. For this reason, it is essential to establish a platform for the SMARTLAB laboratory to effectively disseminate its articles. This platform will not only attract young researchers but also facilitate the efficient management of SMARTLAB members by the administration. By implementing this comprehensive platform, the laboratory will be able to enhance visibility, encourage collaboration, and foster a vibrant research community within SMARTLAB.

3. Considered Solution

To address the issues mentioned in the previous section, we propose the implementation of a comprehensive platform to effectively disseminate events, news, projects, partners, articles, and members' information within SMARTLAB. Additionally, a dedicated back office for administration will be established to efficiently manage SMARTLAB members and their articles. This back office will also provide members with the ability to manage their accounts and articles, ensuring greater control and convenience.

By creating this platform and back office, SMARTLAB will not only enhance the visibility and accessibility of its activities but also empower its members with the tools to actively participate and manage their contributions. This integrated approach will foster collaboration, knowledge sharing, and efficient administration within the SMARTLAB community.

Therefore, this project is divided into three essential phases:

- a. **Front-office:** This part is dedicated to the netizens or users who visit the SMARTLAB website. It allows them to gather information about the SMARTLAB laboratory and get a general idea of its activities, news, events, projects, and members. The front-office provides a user-friendly and intuitive interface to facilitate navigation and access to the desired information.
- b. **Back-office:** This part is specifically designed for the laboratory members. It enables them to manage their accounts autonomously, providing functionalities such as updating personal information, managing published articles, and collaborating with other members. Additionally, the back-office provides the site administrator with the necessary tools to effectively manage the platform, including member management, news and event publication, and overall site supervision.
- c. **Platform:** This phase involves the development and implementation of the platform itself, encompassing both the front-office and back-office. The platform is designed to ensure an optimal user experience, guaranteeing easy access to information, smooth navigation, and efficient interaction with the various functionalities offered.

By combining these three phases, the project aims to provide a comprehensive and functional platform that meets the information, management, and collaboration needs of users and members of the SMARTLAB laboratory.

4. Application Development Methodology

In order to plan our project and define tasks, choosing the right methodology is a crucial step that helps us deliver our project on time while meeting the expected functionalities. In the following, we present agile methodologies:

4.1. Agile methodologies

Agile methodology is a project management methodology that involves dividing projects into phases and emphasizing continuous collaboration and improvement. These teams follow a cyclical process that includes planning, executing, and evaluating.

Agile methods can be identified by several key characteristics and practices:

- **Iterative Development:** Agile projects are organized in short iterations or sprints.
- **Flexibility and Adaptability:** adapts to changes and aims to meet changing requirements and customer feedback.
- **Active Customer Involvement:** Agile places a high value on collaboration and customer engagement.
- **Self-Organizing and Cross-Functional Teams:** Agile teams tend to be self-organizing, meaning they can make decisions autonomously and adapt to changing circumstances.
- **Continuous Improvement:** Agile methods foster a culture of continuous improvement.
- **Emphasis on Delivering Value:** Agile projects focus on delivering value to customers sooner and frequently.
- **Agile Frameworks:** There are several popular Agile frameworks, such as Scrum, Kanban, and Extreme Programming (XP), which provide specific guidelines and practices for implementing Agile principles. [1]

4.2. Comparative study: Agile method / Classic method

In order to choose the suitable method to work with, we did a comparative study between Agile methodology and the traditional/classic project management methodology [2]:

Table 1: Classic Method VS Agile Method

Theme	Agile method	Classic method
Approach to Project Management	Agile is a flexible and iterative approach to project management. The focus is on incrementally adding value and adapting to changes throughout the project	The classical approach follows the sequential approach of project management. It includes different phases such as requirements gathering, design, development, testing and deployment. Each stage is completed before moving on to the next, with limited flexibility to make changes after one stage is complete.
Flexibility and Adaptability	Agile is very flexible and able to adapt to changing requirements and customer demands. It adapts to change and allows for continuous feedback and adjustments throughout the project.	Classical methods lack flexibility and rigidity. Once a stage is complete, it is difficult to make changes without going back to earlier stages, which is time consuming and expensive.
Customer Collaboration	Agile encourages frequent and active collaboration with customers. Clients or stakeholders are involved throughout the project, providing feedback and clarifying requirements.	Using the classical approach, customer involvement is usually limited to initial phases such as B. Requirements Gathering and Approval.
Project Monitoring and Control	Agile projects use adaptive and iterative control mechanisms. Progress is monitored through regular meetings, such as daily stand ups and sprint retrospectives.	Classical methods are based on more traditional control methods. Progress is typically measured against predefined project milestones and deliverables.
Risk Management	Agile incorporates risk management throughout the project lifecycle. Risks are	Risk management in traditional approaches is usually dealt with in

	identified early on, and mitigation strategies are implemented during each iteration.	a separate phase, usually the planning phase.
Documentation	Agile values working software or tangible results over comprehensive documentation. Documentation is usually minimal and focuses on basic information.	The classic approach emphasizes detailed documentation for each phase of the project.

4.3. Scrum

On the other hand, Scrum is an agile methodology that focuses on flexibility and adaptability. It employs an iterative and incremental approach, dividing the project into short time frames called sprints. The project requirements are organized into a product backlog, and the highest-priority items are selected for each sprint. Scrum promotes collaboration, transparency, and continuous feedback. It allows for changes and adjustments throughout the project, enabling a more responsive and customer-centric development process.

In the Scrum methodology, there are three key roles:

Scrum Master: The Scrum Master is responsible for ensuring that the Scrum framework is understood and followed by the team. They act as a facilitator, coach, and servant-leader for the team, helping to remove any obstacles or impediments that may hinder progress. The Scrum Master also organizes and leads the various Scrum ceremonies, such as the daily stand-up meetings, sprint planning, sprint review, and sprint retrospective.

Product Owner: The Product Owner represents the stakeholders and is responsible for defining and prioritizing the product backlog. They work closely with the development team to ensure that the product backlog items are well-defined, properly estimated, and aligned with the project goals and customer requirements. The Product Owner makes decisions regarding the scope and release of the product and provides clear guidance and direction to the development team.

Development Team: The Development Team is a self-organizing and cross-functional group of individuals who are responsible for delivering a potentially shippable product

increment at the end of each sprint. The team collaboratively works on the development, testing, and delivery of the product, and they have the autonomy to determine how the work is accomplished. The Development Team estimates and selects the product backlog items to be completed during a sprint and is accountable for achieving the sprint goals. [3]

Table 2: Advantages & Disadvantages of Scrum

Advantages of Scrum	Disadvantages of Scrum
Scrum can aid teams in accomplishing project deliverables rapidly and efficiently.	Scrum can sometimes be associated with scope creep, as the absence of a fixed end-date can allow for continuous changes and additions to project requirements.
By implementing Scrum, teams can ensure optimal use of both time and money.	The probability of project failure increases when individuals involved are not fully committed or lack cooperation.
Easily manageable sprints are used to divide large projects into smaller.	Implementing the Scrum framework in large teams can present significant challenges.
During the sprint review, the development work, including coding and testing, is presented and evaluated.	The success of the Scrum framework often relies on the presence of experienced team members.
This approach is particularly effective for development projects that require a fast-paced and agile workflow.	Team members may occasionally find daily meetings frustrating.
Daily scrum meetings provide visibility into the individual efforts of each team member.	The departure of a team member in the middle of a project can have a significant negative impact on its progress and outcomes.

Conclusion

Throughout this introductory chapter, we give a global view of our project, by analysing the host organization needs, criticizing the existing solution, proposing solutions, as well as suggesting possible methodologies ... which made us ready to take the next step, the planning phase, which is the subject of the next chapter.

Chapter Two: Planning and Architecture

INTRODUCTION

This chapter describes the crucial phase of the Scrum methodology, known as "Sprint Zero" or the planning phase. During this phase, the focus is solely dedicated to preparing and organizing all the necessary elements in order to start the project activities.

The first step is to identify the functional requirements, different actors, and non-functional requirements. After that, the initial Product Backlog will be created, and the sprints are planned accordingly.

ACTORS IDENTIFICATION

Table 3: Actors Identification

Netizen	Member	Administrator
This refers to the individual who uses the "Smart Lab" website to access its laboratory articles, news, events, and other information.	This user is registered on the platform and could be a doctor, PhD student, teacher researcher, or any other type of registered user.	This refers to the person who controls the back office and manages the platform.

SPECIFICATION OF NEEDS

1. Functional Requirements

1.1. Identification of actors and functional requirements

This table illustrate the actors and their functional requirements:

Table 4: Actors & Functional Requirements

Netizen	Member	Administrator
Browse: <ul style="list-style-type: none"> - View Articles - View News - View Events - View Partners - View Members Send Feedback Registrate	Authentication Account Management: <ul style="list-style-type: none"> - View Profile - Update Account Article Management: <ul style="list-style-type: none"> - Add Article - Update Article - Delete Article Event Management: <ul style="list-style-type: none"> - Add Event - Update Event - Delete Event News Management: <ul style="list-style-type: none"> - Add News - Update News - Delete News 	Authentication Partners Management: <ul style="list-style-type: none"> - Add Partner - Update Partner - Delete Partner Members Management: <ul style="list-style-type: none"> - Update Profession - Update Teams - Accept Member - View Member - Delete Member - Add Privilege to Member Projects Management: <ul style="list-style-type: none"> - Add Project - Update Project - Delete Project Reporting

1.2. Use case diagram

This figure represents the general use case diagram of our project:

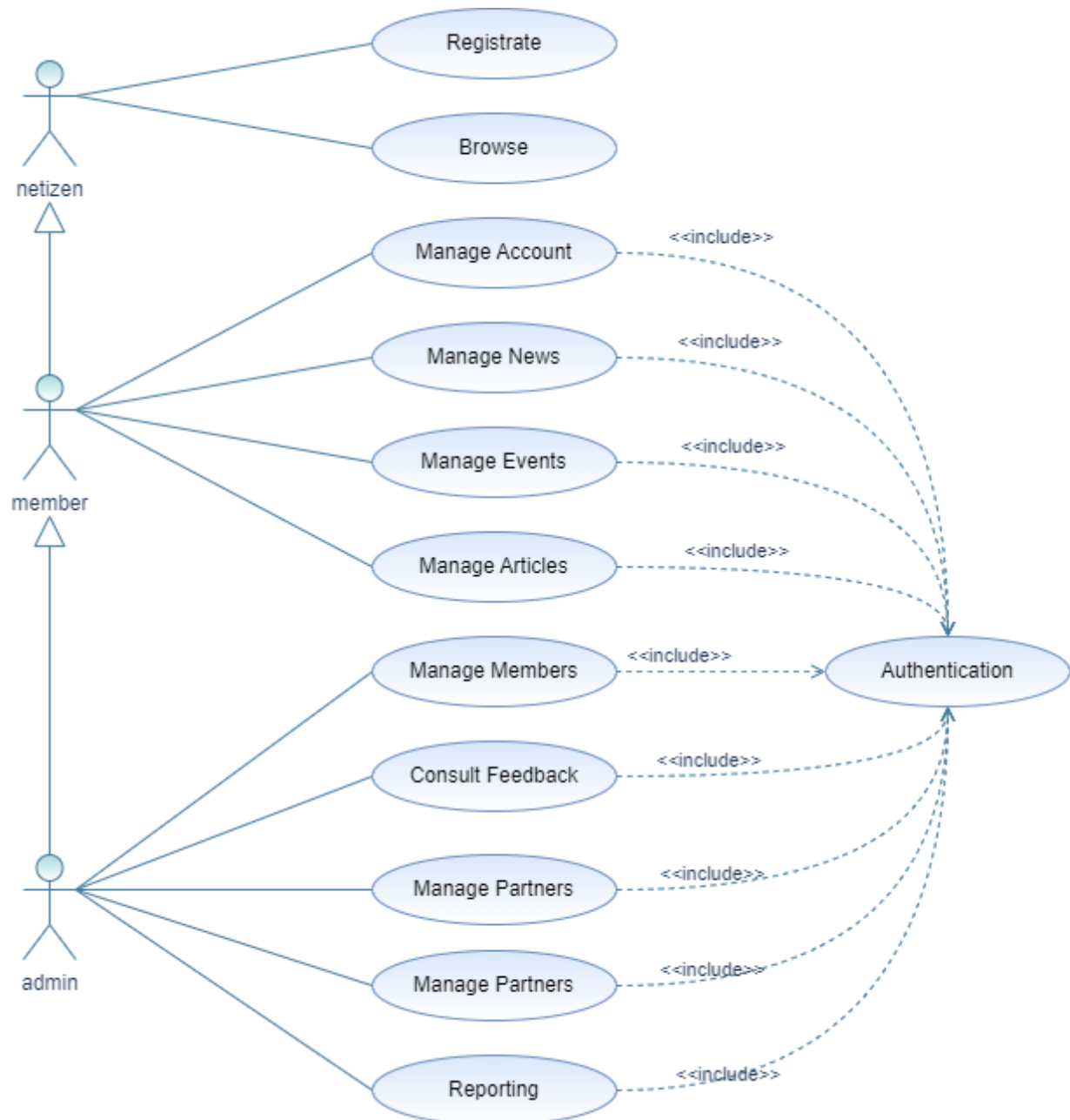


Figure 2: General Use Case Diagram

1.3. Class Diagram

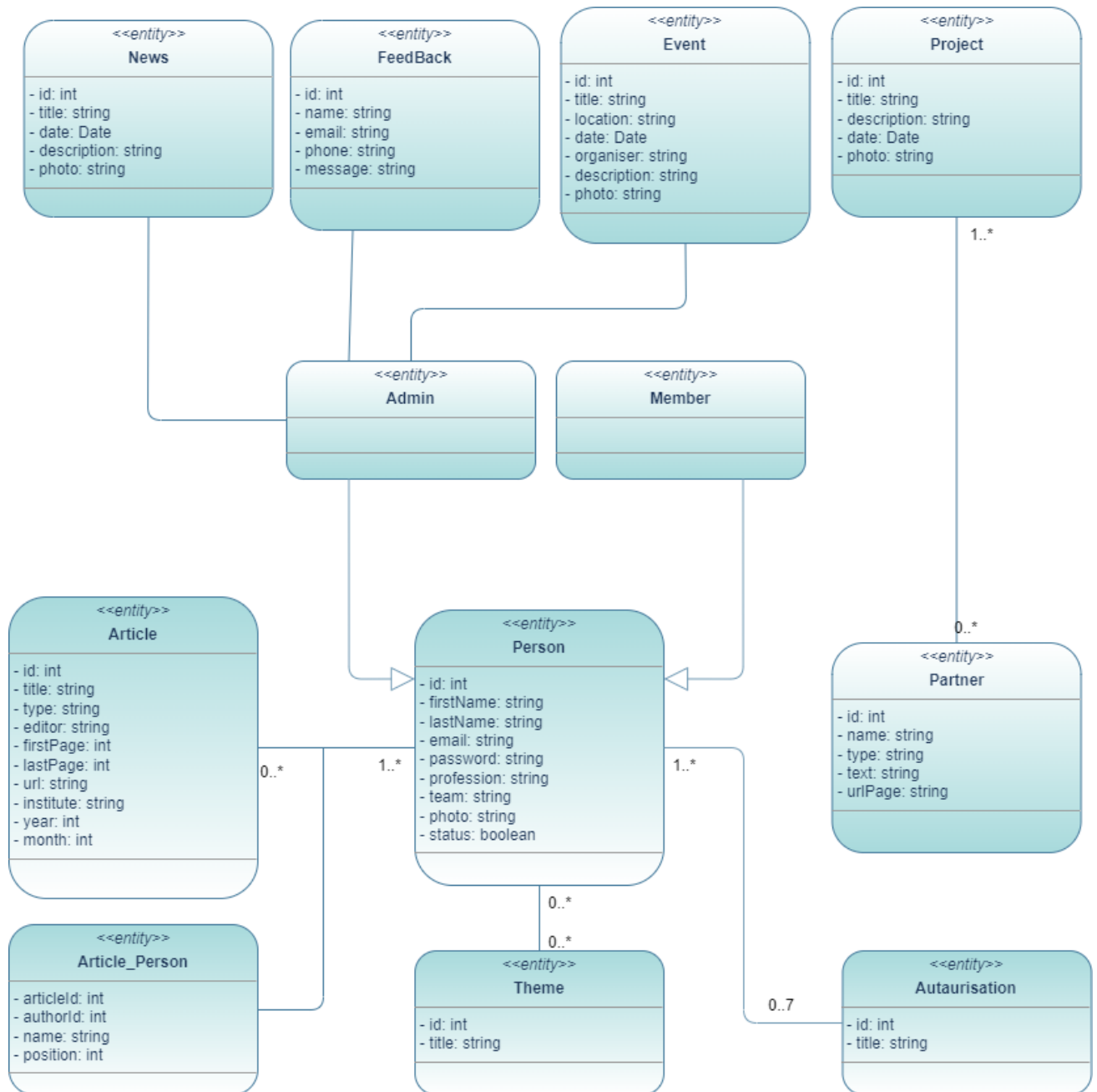


Figure 3: General Class Diagram

2. Non-functional requirements

2.1. Security

- Establishing the connection is necessary to access the platform
- If the password is forgotten, the user can only change it after having received a confirmation email and answer a security question.
- The password will be at least 8 characters long with at least a special character, an uppercase character, a character in lowercase and a number.

2.2. Ergonomics and Accessibility

- Page load times are reasonable and estimated to be less than 2 seconds
- The page size is less than 100 kB
- The contrast between background colour and writing
- Vectorial logo for high quality
- The platform is - tested on several browsers





2.3. Reusability

- Some modules of this project may be components reusable: the project is actually based on modularity.

TEAMS AND ROLES

Table 5 displays all the team members who worked on the development of SMARTLAB website:

Table 5: Scrum Team and Roles

Lamjed Ben Said	Anouer Bennajeh	Roudeina Hamdi	Jihen Aridhi
			
Product Owner	Scrum Master	Development Team	
The Scrum Product Owner is responsible for the project's or product's outcomes. Their task is to create and prioritize the product backlog, which is a list of features to be developed by the development team.	The Scrum master's role is to ensure that each team member is engaged and to help them overcome any obstacles they may encounter. They assist the team in researching and identifying solutions.	The development team's mission is to technically realize user stories. They estimate the complexity of user stories and develop products, which includes project development, testing, and configuration. The development team determines the best way to do the work and is responsible for the technical quality and technical choices made.	

PRODUCT BACKLOG

The table 6 represents the product backlog of our project that resume the functionalities of the platform:

Table 6:Product Backlog

ID	Module	ID	Functionality	User Story	Priority
1	Registrate	1.A	Registration	As an internet user, I can create an account to log in.	High
2	Authentication	2.A	Authentication	As a user of the platform, I can authenticate myself to access my account.	High
3	Account Management	3.A	Update Account	As a user of the platform, I can edit my profile.	High
		3.B	Update Password	As a user of the platform, I can change my password after entering the old one.	High
4	News Management	4.A	Add News	As an administrator of the platform, I can create (add) a news item for the Smart Lab laboratory.	High
		4.B	Update News	As an administrator of the platform, I can update a news item for the Smart Lab laboratory.	High

		4.C	Delete News	As an administrator of the platform, I can delete a news item of Smart Lab laboratory.	High
		4.D	Consult News	As a user of the platform, I can consult all the news of the laboratory.	Medium
5	Events Management	5.A	Add Event	As an administrator of the platform, I can create (add) an event item for the Smart Lab laboratory.	High
		5.B	Update Event	As an administrator of the platform, I can update an event item for the Smart Lab laboratory.	High
		5.C	Delete Event	As an administrator of the platform, I can delete an event item of Smart Lab laboratory.	High
		5.D	Consult Event	As a user of the platform, I can consult all the news of the laboratory.	Medium
6	Articles Management	6.A	Add Article	As a member of the platform, I can create (add) an event item for the Smart Lab laboratory.	High
		6.B	Update Article	As a member of the platform, I can update an event item for the Smart Lab laboratory.	High

		6.C	Delete Article	As a member of the platform, I can delete an event item of Smart Lab laboratory.	High
		6.D	Consult Article	As a user of the platform, I can consult all the news of the laboratory.	Medium
7	Members Management	7.A	Update Profession	As an administrator of the platform, I can update the profession of each member.	High
		7.B	Update Teams	As an administrator of the platform, I can update the team of each member.	High
		7.C	Accept Member	As an administrator of the platform, I can Accept or delete the membership requests.	High
		7.D	Delete Member	As an administrator of the platform, I can delete any member.	High
		7.E	Consult Member	As an administrator of the platform, I can consult the list members.	Medium
		7.F	Give Authorisations	As an administrator of the platform, I can add authorisation to members.	High

8	Partners Management	8.A	Add Partner	As an administrator of the platform, I can add a partner for the Smart Lab laboratory.	High
		8.B	Update Partner	As an administrator of the platform, I can update a partner for the Smart Lab laboratory.	High
		8.C	Delete Partner	As an administrator of the platform, I can delete a partner item of Smart Lab laboratory.	High
		8.D	View Partners	As a user of the platform, I can consult all the partners of the laboratory.	Medium
9	Projects Management	9.A	Add Project	As an administrator of the platform, I can add a project for the Smart Lab laboratory.	High
		9.B	Update Project	As an administrator of the platform, I can update a project for the Smart Lab laboratory.	High
		9.C	Delete Project	As an administrator of the platform, I can delete a project item of Smart Lab laboratory.	High
		9.D	View Project	As a user of the platform, I can consult all the projects of the laboratory.	Medium

10	Feedback Management	10.A	Consult Feedback	As an administrator of the platform, I can consult all the feedback sent to the laboratory.	Low
11	Reporting	11.A	Download PDF	As an administrator of the platform, I can download a pdf which contain all the statics that I choose.	Medium

STRUCTURING IN SPRINTS AND PLANNING

After specifying the functional and non-functional requirements of this project, and designing the product backlog, we can now start planning sprints. It has been decided to divide the tasks into three main sprints, which are consisted of:

Table 7: Sprints & User Stories

Sprint 1	Sprint 2	Sprint 3
Registrate	Authentication	Manage Members
Browse	Manage Account	Manage Feedback
	Manage News	Manage Partners
	Manage Events	Reporting
	Manage Articles	

CONCLUSION

During this chapter, we have gone through planning of the project, starting with the requirement specifications, introducing different actors and their functionalities, product backlog and finally specifying project sprints, which is what we will be explaining for the next few chapters.

Chapter Three: Sprint 1: Building Front Office

INTRODUCTION

In this chapter, we will dive into the details of the second sprint of our project, which mainly carries the functionalities of the “netizen” actor.

SPRINT BACKLOG

The table 8 represents the sprint backlog that enumerate the functionalities of this sprint:

Table 8: Sprint 1 Backlog

ID	User story	task	task
1	As a netizen, I can create an account.	1.A	Elaborate the use case, sequence, and class diagrams for the “Registration” functionality.
		1.B	Develop the “Registration” use case.
		1.C	Test the “Registration” use case.
2	As a netizen, I can consult the website of the laboratory smart lab.	2.A	Develop the interface of website.
3	As a netizen of the website, I can send my feedback to the laboratory smart lab.	3.A	Develop the class diagrams for the “Feedback” functionality.
		3.B	Develop the “Feedback” use case.
		3.C	Test the “Feedback” use case.

FUNCTIONAL SPECIFICATION

This figure 3 illustrates the main functionalities for the “netizen” actor

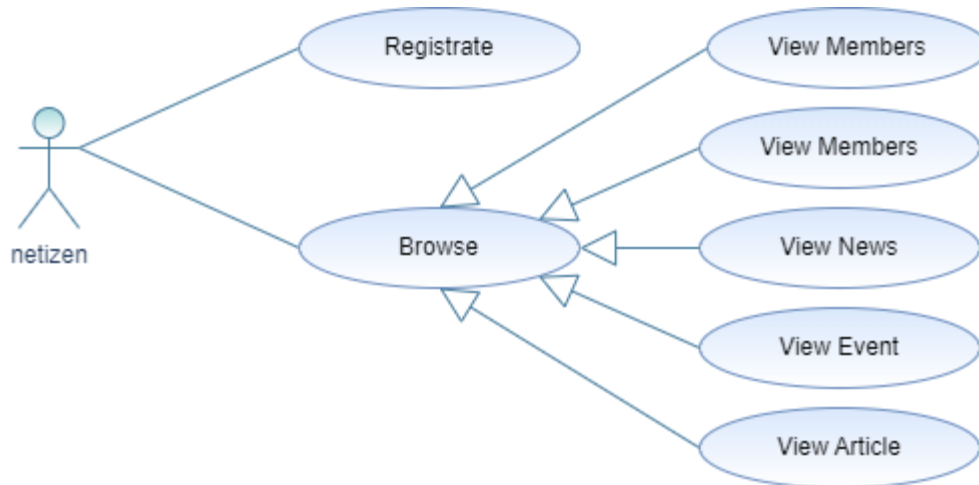


Figure 4: Netizen's Use Case Diagram

ANALYSIS

For further analysis of this sprint, we are diving into the details of each functionality that has been mentioned above.

1. Use Case Diagram

1.1. Registrare

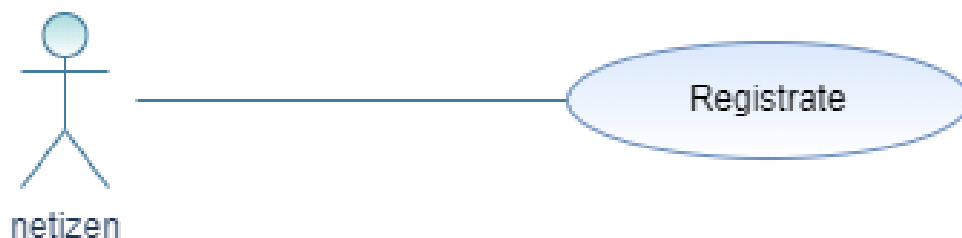


Figure 5: “Registrare” Use Case Diagram

2. Textual Description

2.1. Registrare

Table 9: "Registration" Textual Description

<i>Use Case</i>	<i>Registrare</i>
<i>Actor</i>	<i>Netizen</i>
<i>Pre-Condition</i>	<i>Enter the website</i>
<i>Post-Condition</i>	<i>Registration request sent to the administrator</i>
<i>Basic Scenario</i>	<ol style="list-style-type: none">1. <i>The netizen enters the website.</i>2. <i>The netizen goes to "Registrare" interface</i>3. <i>The netizen enters their information</i>4. <i>The netizen clicks on "registrare"</i>5. <i>System display message "account created with success"</i>
<i>Exceptional Scenario</i>	<ol style="list-style-type: none">4.a. <i>If the information is wrong system will display "incorrect"</i>4.b. <i>If member not accepted yet system will display message "you're not accepted yet"</i>
<i>Alternative Scenario</i>	<i>None</i>

3. System Sequence Diagram

3.1. Registrate

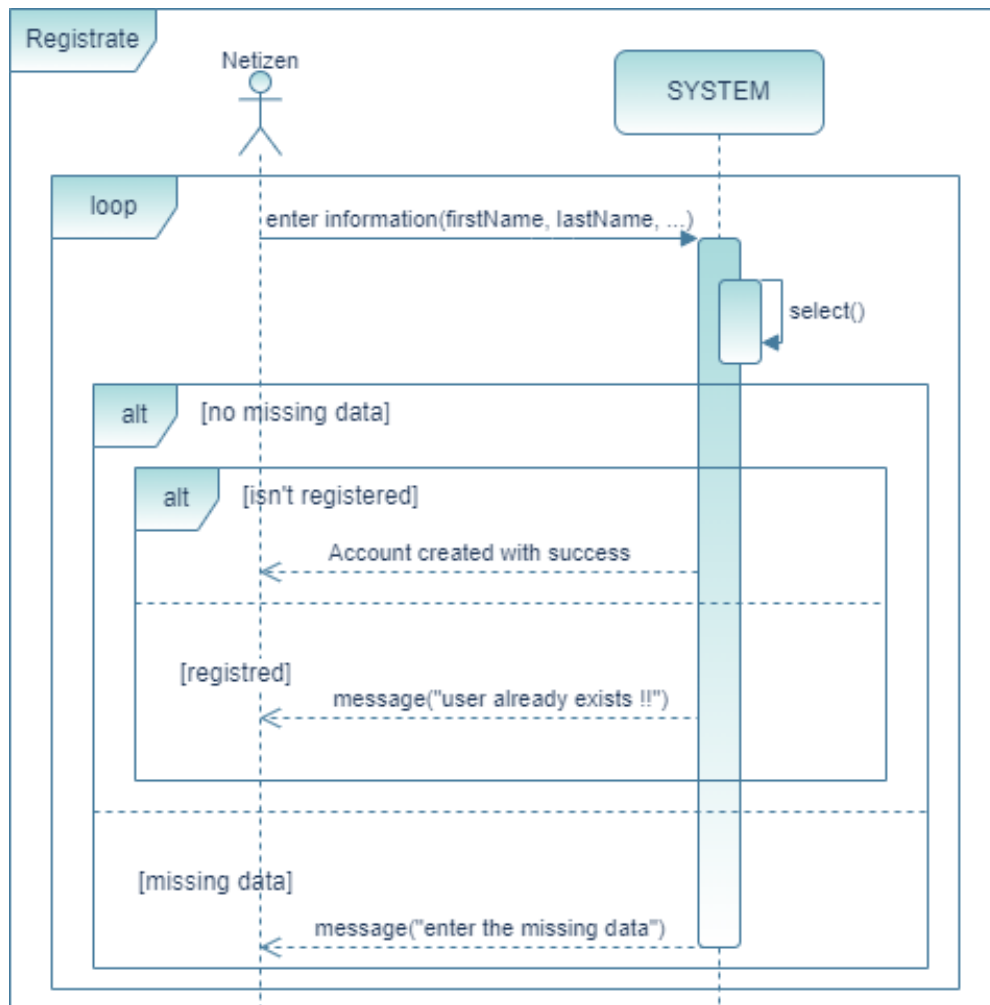


Figure 6: "Registrate" System Sequence Diagram

CONCEPTION

1. Class Diagram

1.1. Registrate

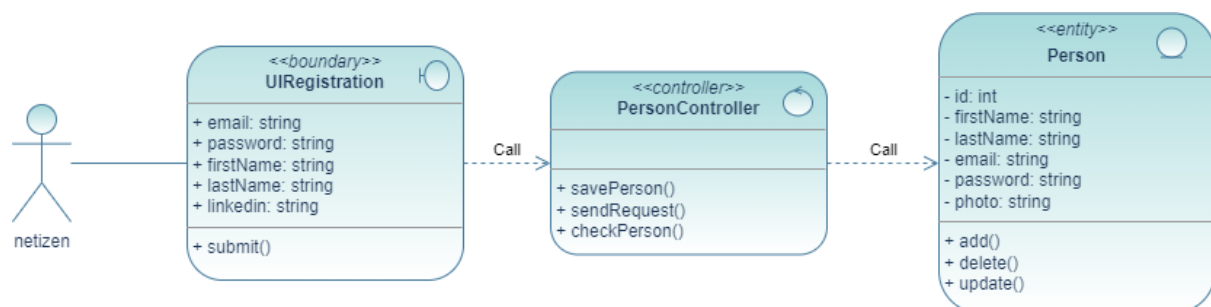


Figure 7: "Registrate" Class Diagram

2. Detailed Sequence Diagram

2.1. Registrate

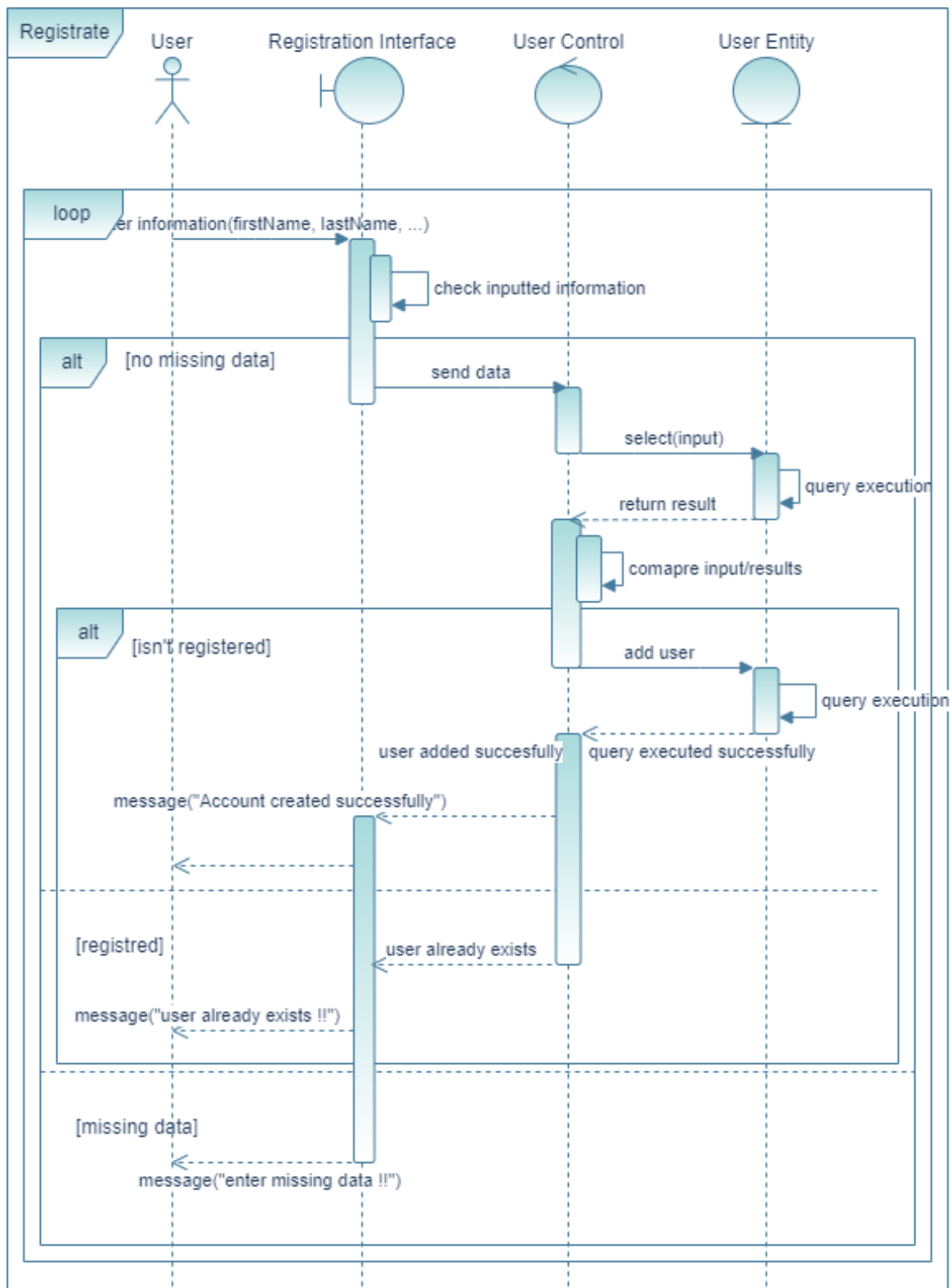


Figure 9: "Registrate" Detailed Sequence Diagram

IMPLEMENTATION

1. Relational Schema

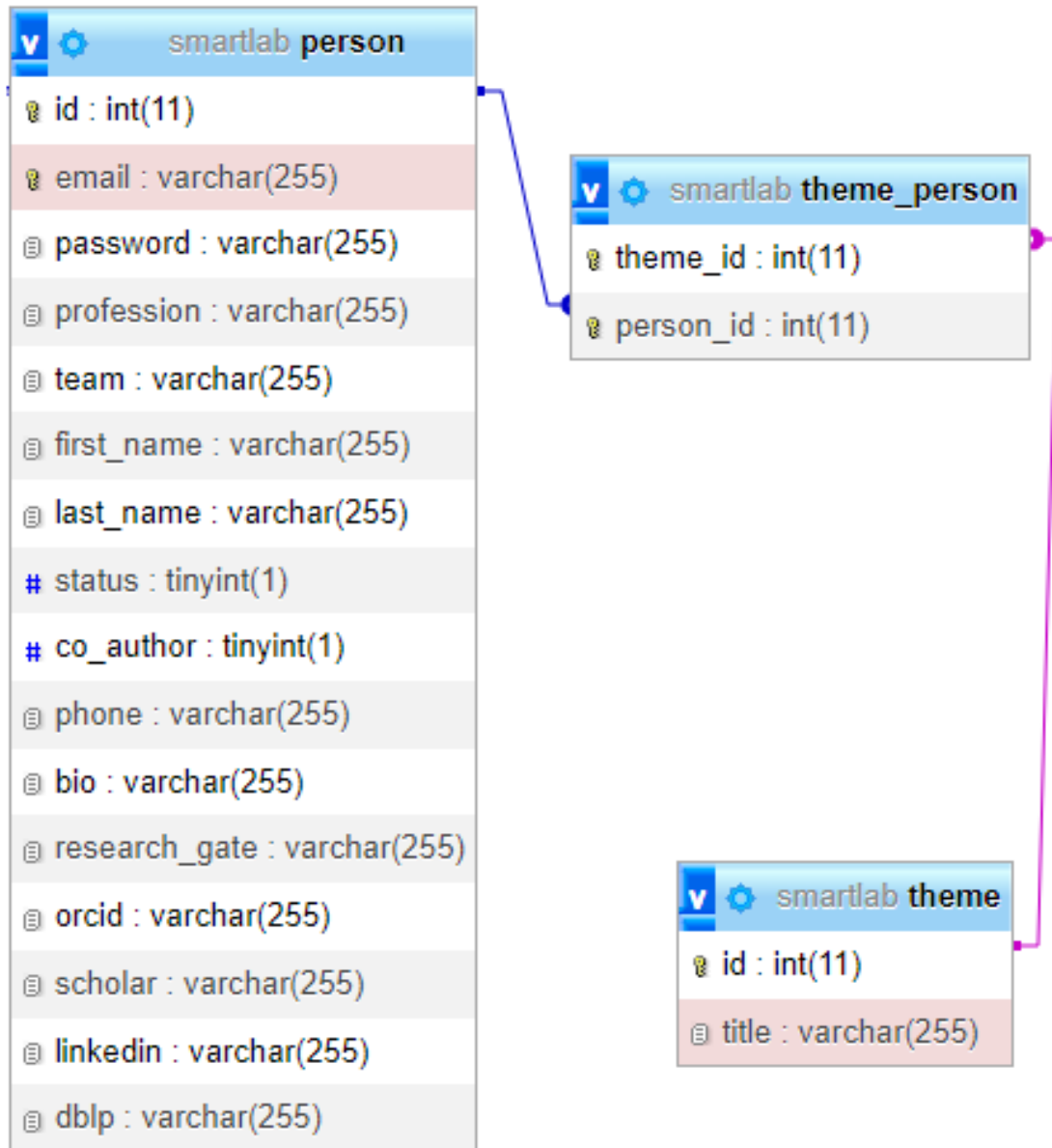



Figure 8: Sprint 1 Relational Schema

2. Interfaces

2.1. Registrare



General Information

First Name


Last Name

email

Password

Confirm your password

next



General Information

Phone

Short Bio

Curriculum vitae

previous next

Strategies for
Modeling and
ARTificial
inTelligence

General Information

your ResearchGate's url ...

your ORCID's url ...

your Google scholar's url ...

your LinkedIn's url ...

your DBLP's url ...

previous

next

Strategies for
Modeling and
ARTificial
inTelligence

General Information

Artificial intelligence

Machine learning

Deep learning

Soft computing

Multi agent system

Data mining

Web

Optimization

Cyber security

Computer science

Graph theory

Big data

more ...

previous

sign up

Figure 9: "Registrarte" Interface

Conclusion

This chapter presents a detailed analysis of the first sprint, that is consisted of tasks assigned to the netizen actor, as well as the conception, and finally an implementation.

Chapter Four: Sprint 2: Building Members Back Office

INTRODUCTION

In this chapter, we will dive into the details of the second sprint of our project, which mainly carries the functionalities of the “member” actor.

SPRINT BACKLOG

This table represents the sprint backlog that enumerate the functionalities of this sprint:

Table 10: Sprint 2 Backlog

ID	User story	task	task
1	As a member of the laboratory, I can update my account information.	1.A	Elaborate the use case, sequence, and class diagrams for the “Manage Account” functionality
		1.B	Develop the “Manage Account” use case.
		1.C	Test the “Manage Account” use case.
2	As a member of the platform, I can add news to the website.	2.A	Develop the use case, sequence, and class diagrams for the “Add News” functionality
		2.B	Develop the “Add News” use case.
		2.C	Test the “Add News” use case.
3	As a member of the platform, I can update news in the website.	3.A	Develop the use case, sequence, and class diagrams for the “Update News” functionality
		3.B	Develop the “Update News” use case.
		3.C	Test the “Update News” use case.

4	As a member of the platform, I can delete news from the website.	4.A	Develop the use case, sequence, and class diagrams for the “Delete News” functionality
		4.B	Develop the “Delete News” use case.
		4.C	Test the “Delete News” use case.
5	As a member of the platform, I can add events to the website.	5.A	Develop the use case, sequence, and class diagrams for the “Add Events” functionality
		5.B	Develop the “Add Events” use case.
		5.C	Test the “Add Events” use case.
6	As a member of the platform, I can update events in the website.	6.A	Develop the use case, sequence, and class diagrams for the “Update Events” functionality
		6.B	Develop the “Update Events” use case.
		6.C	Test the “Update Events” use case.
7	As a member of the platform, I can delete events from the website.	7.A	Develop the use case, sequence, and class diagrams for the “Delete Events” functionality
		7.B	Develop the “Delete Events” use case.
		7.C	Test the “Delete Events” use case.
8	As a member of the platform, I can add articles to the website.	8.A	Develop the use case, sequence, and class diagrams for the “Add Articles” functionality
		8.B	Develop the “Add Articles” use case.
		8.C	Test the “Add Articles” use case.
9	As a member of the platform, I can update articles in the website.	9.A	Develop the use case, sequence, and class diagrams for the “Update Articles” functionality

		9.B	Develop the “Update Articles” use case.
		9.C	Test the “Update Articles” use case.
10	As a member of the platform, I can delete articles from the website.	10.A	Develop the use case, sequence, and class diagrams for the “Delete Articles” functionality
		10.B	Develop the “Delete Articles” use case.
		10.C	Test the “Delete Articles” use case.

FUNCTIONAL SPECIFICATIONS

This figure illustrates the main functionalities for the “member” actor:

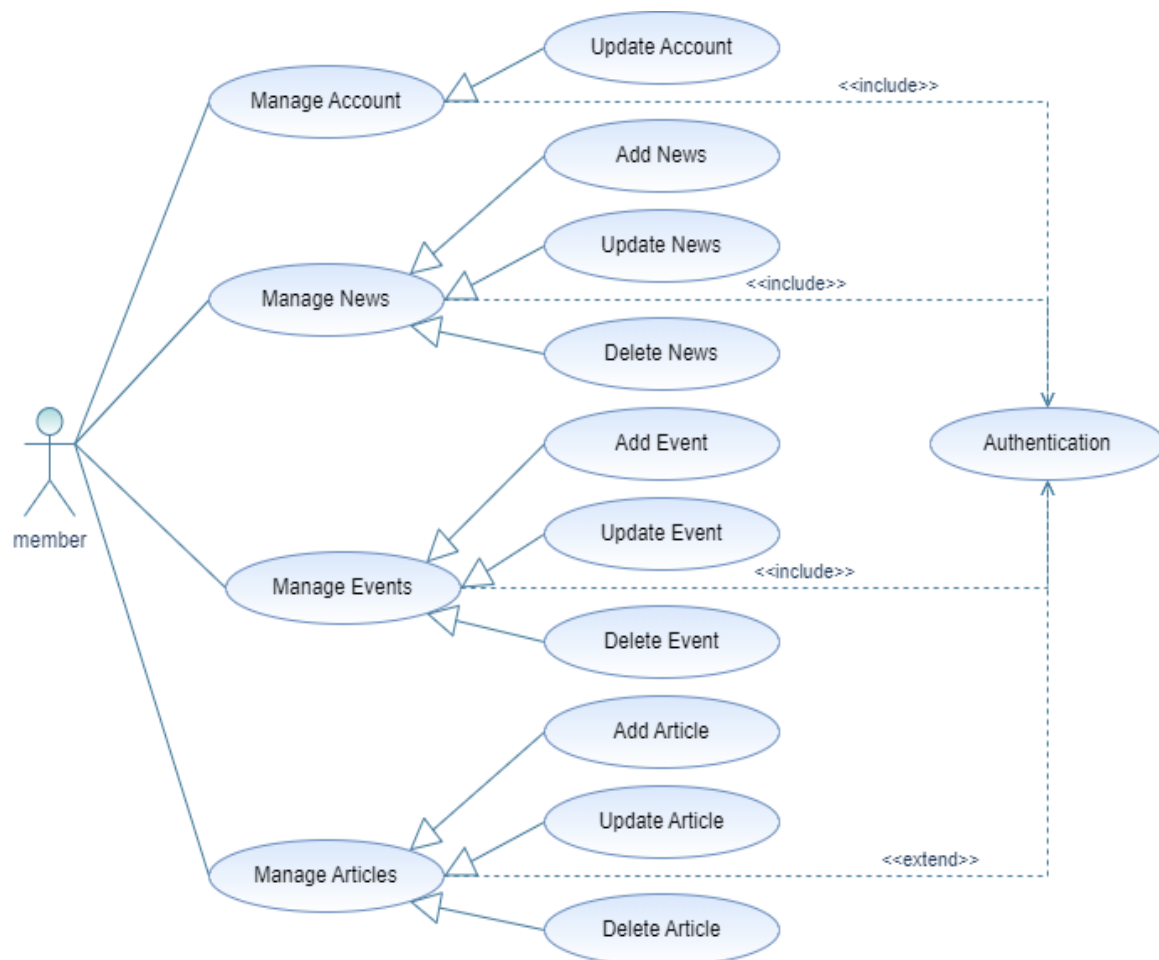


Figure 10: Member's Use Case Diagram

ANALYSIS

For further analysis of this sprint, we are diving into the details of each functionality that has been mentioned above.

1. Use Case Diagram

1.1. Manage Account

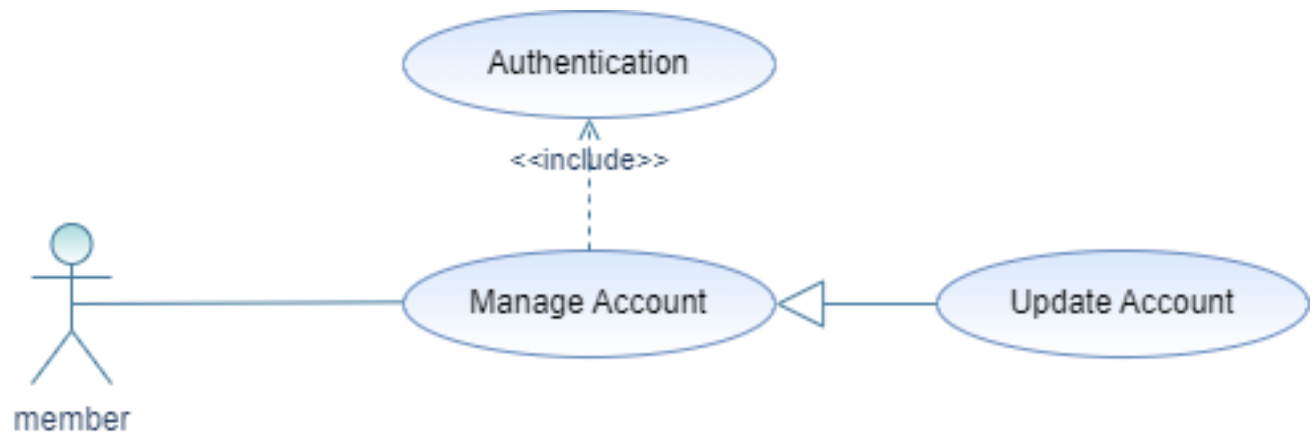


Figure 11: "Manage Account" Use Case Diagram

1.2. Manage News

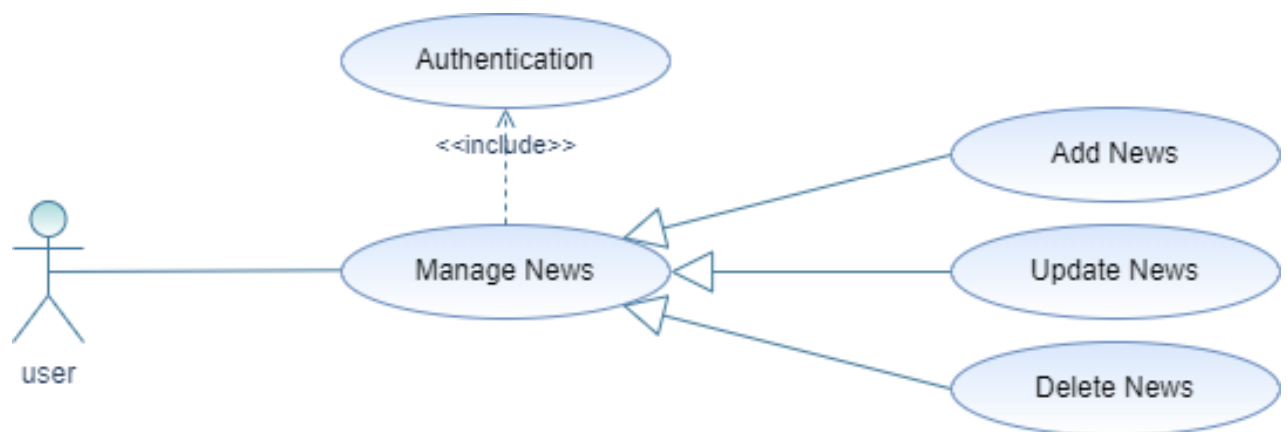


Figure 12: "Manage News" Use Case Diagram

1.3. Manage Events

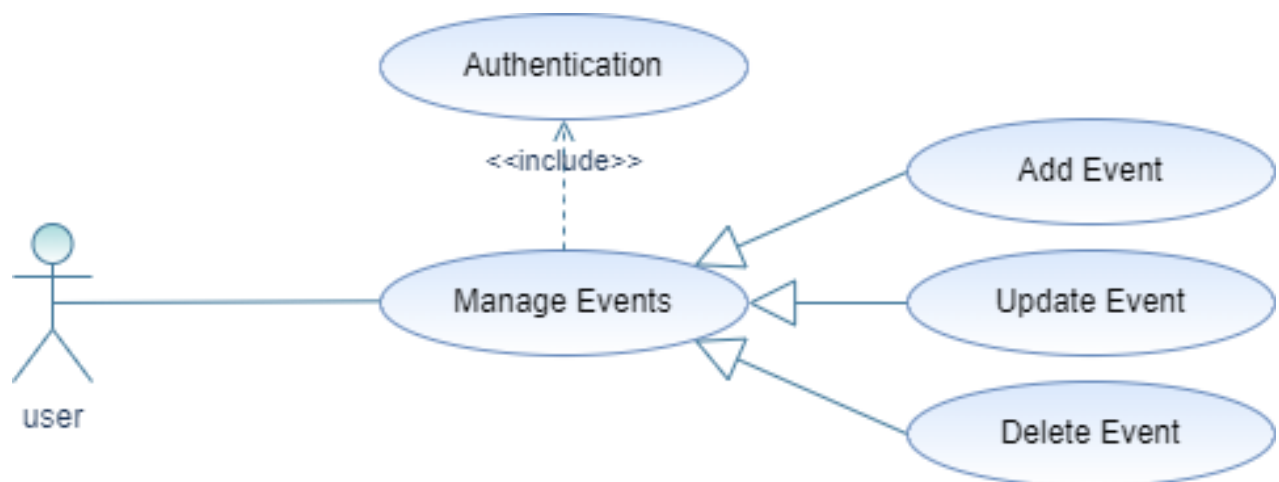


Figure 13: "Manage Events" Use Case Diagram

1.4. Manage Articles

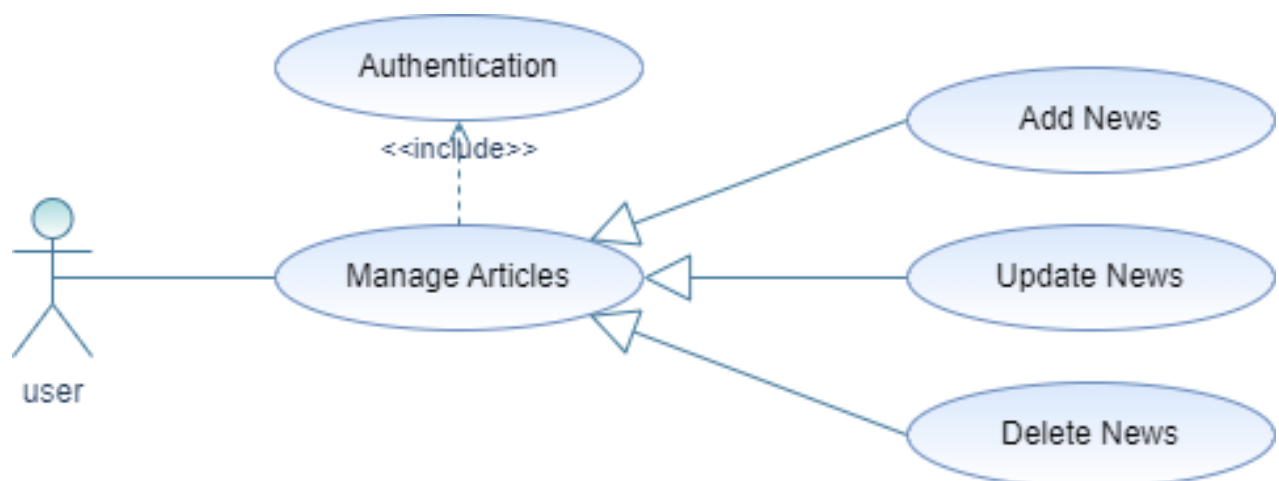


Figure 14: "Manage Articles" Use Case Diagram

2. Textual Description

2.1. Authentication

Table 11: “Authentication” Textual Description

<i>Use Case</i>	<i>Authentication</i>
<i>Actor</i>	<i>Member, Administrator</i>
<i>Pre-Condition</i>	<i>Enter the website</i>
<i>Post-Condition</i>	<i>Being Authenticated</i>
<i>Basic Scenario</i>	<ol style="list-style-type: none">1. The member enters the website.2. The member goes to “Authentication” interface.3. The member enters their email and password.4. The member clicks on “Login”.5. The System allows the member to enter their account.
<i>Exceptional Scenario</i>	<ol style="list-style-type: none">4.a. If the information is wrong system will display “incorrect”4.b. If member not accepted yet system will display message “you’re not accepted yet”
<i>Alternative Scenario</i>	<i>None.</i>

2.2. Update Account

Table 12: “Update Account” Textual Description

<i>Use Case</i>	<i>Update Account</i>
<i>Actor</i>	<i>Member</i>
<i>Pre-Condition</i>	<i>Being Authenticated</i>
<i>Post-Condition</i>	<i>Account Updated successfully</i>
<i>Basic Scenario</i>	<ol style="list-style-type: none">1. The member goes to “Update Account” interface.2. The system displays the account updating form.3. The member fills the form and clicks “submit”.4. System updates the information to the database.

<i>Exceptional Scenario</i>	<i>4.a. If the user fails to fill some of the mandatory fields, system will display a warning.</i>
<i>Alternative Scenario</i>	<i>None.</i>

2.3. Add News

Table 13: "Add News" Textual Description

<i>Use Case</i>	<i>Add News</i>
<i>Actor</i>	<i>Member</i>
<i>Pre-Condition</i>	<i>Being Authenticated</i>
<i>Post-Condition</i>	<i>News added successfully</i>
<i>Basic Scenario</i>	<ol style="list-style-type: none"> 1. The member clicks "Add News". 2. The system displays the news adding form. 3. The member fills the form and clicks "submit". 4. The system adds new news to the database.
<i>Exceptional Scenario</i>	<i>4.a. If the user fails to fill some of the mandatory fields, the system will display a warning.</i>
<i>Alternative Scenario</i>	<i>None.</i>

2.4. Update Event

Table 14: "Update Event" Textual Description

<i>Use Case</i>	<i>Update News</i>
<i>Actor</i>	<i>Member</i>
<i>Pre-Condition</i>	<i>Being Authenticated</i>
<i>Post-Condition</i>	<i>News added successfully</i>
<i>Basic Scenario</i>	<ol style="list-style-type: none"> 1. The member clicks "Update News". 2. The system displays the news adding form. 3. The member fills the form and clicks "submit". 4. The system updates news to the database.

<i>Exceptional Scenario</i>	4.a. If the user fails to fill some of the mandatory fields, the system will display a warning.
<i>Alternative Scenario</i>	None.

2.5. Delete Article

Table 15: "Delete Article" Textual Description

<i>Use Case</i>	<i>Delete News</i>
<i>Actor</i>	<i>Member</i>
<i>Pre-Condition</i>	<i>Being Authenticated</i>
<i>Post-Condition</i>	<i>News Deleted successfully</i>
<i>Basic Scenario</i>	<ol style="list-style-type: none"> 1. User selects event and clicks "Delete Article". 2. System displays a confirmation dialog, "Are you sure?". 3. The member clicks "Yes". 4. System deletes the article from the database.
<i>Exceptional Scenario</i>	None.
<i>Alternative Scenario</i>	<ol style="list-style-type: none"> 3.a. The member clicks "No". 3.b. The system cancels the task.

3. System Sequence Diagram

3.1. Authentication

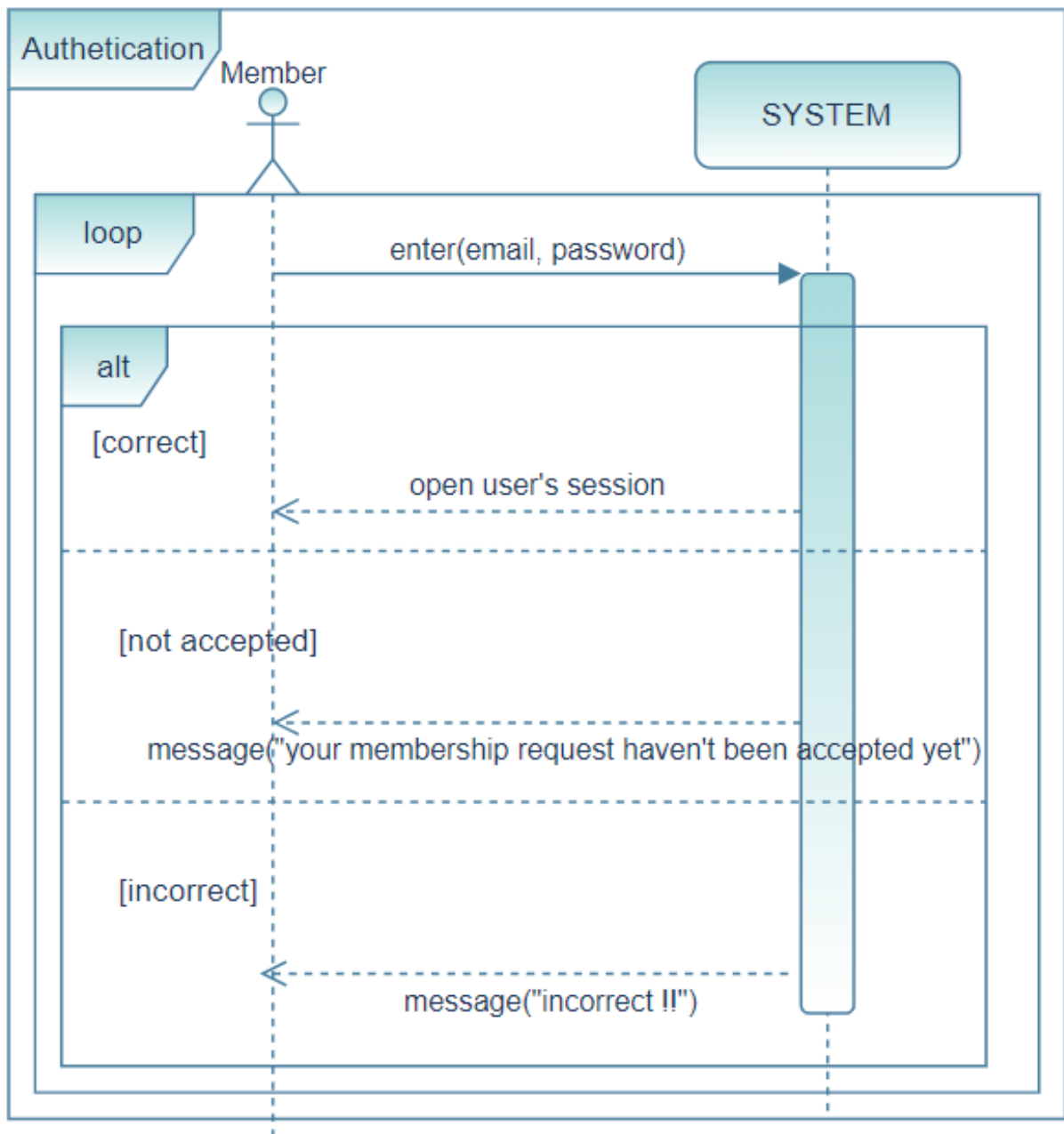


Figure 15: "Authentication" System Sequence Diagram

3.2. Update Account

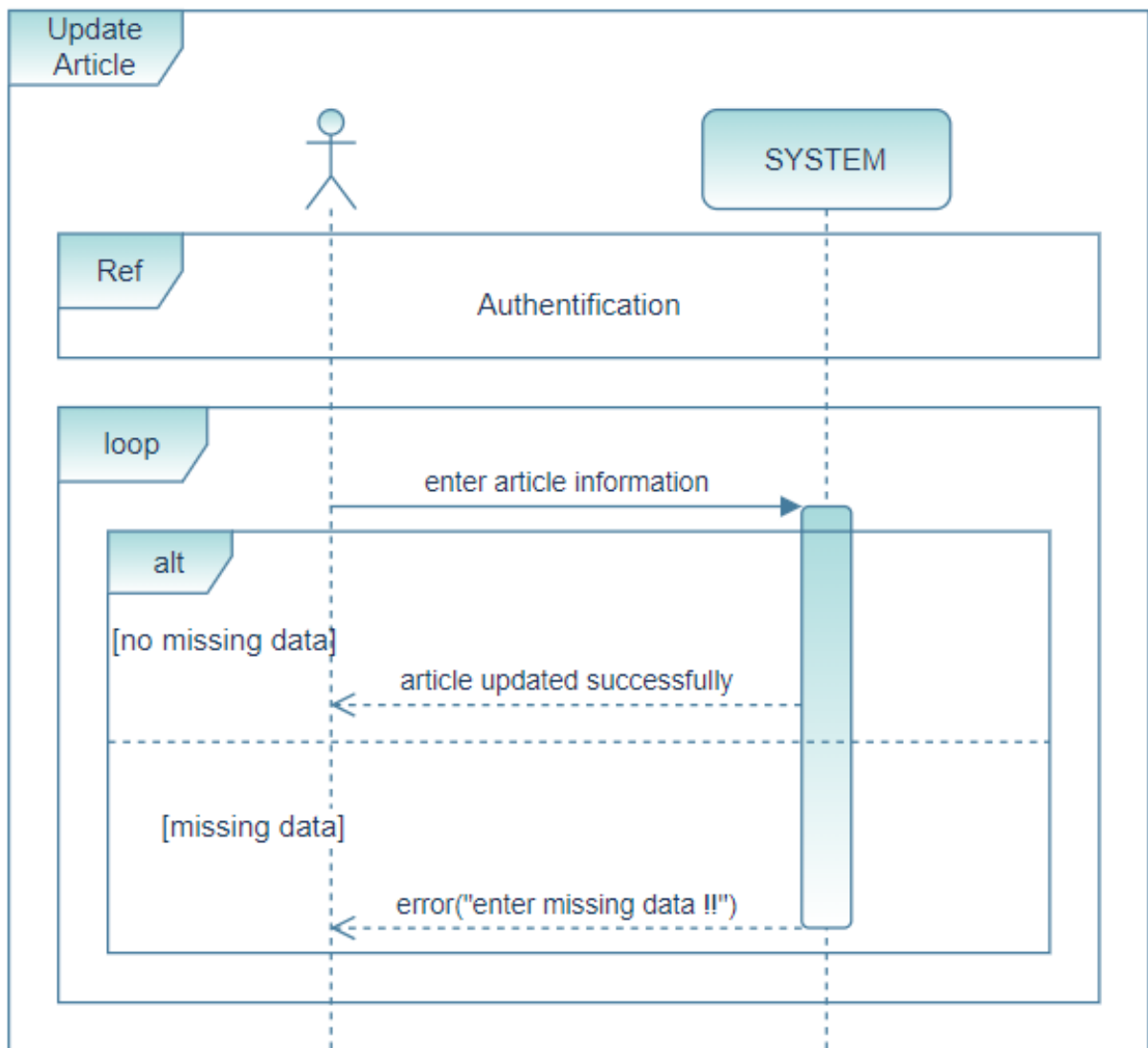


Figure 16: "Update Account" System Sequence Diagram

3.3. Update Event

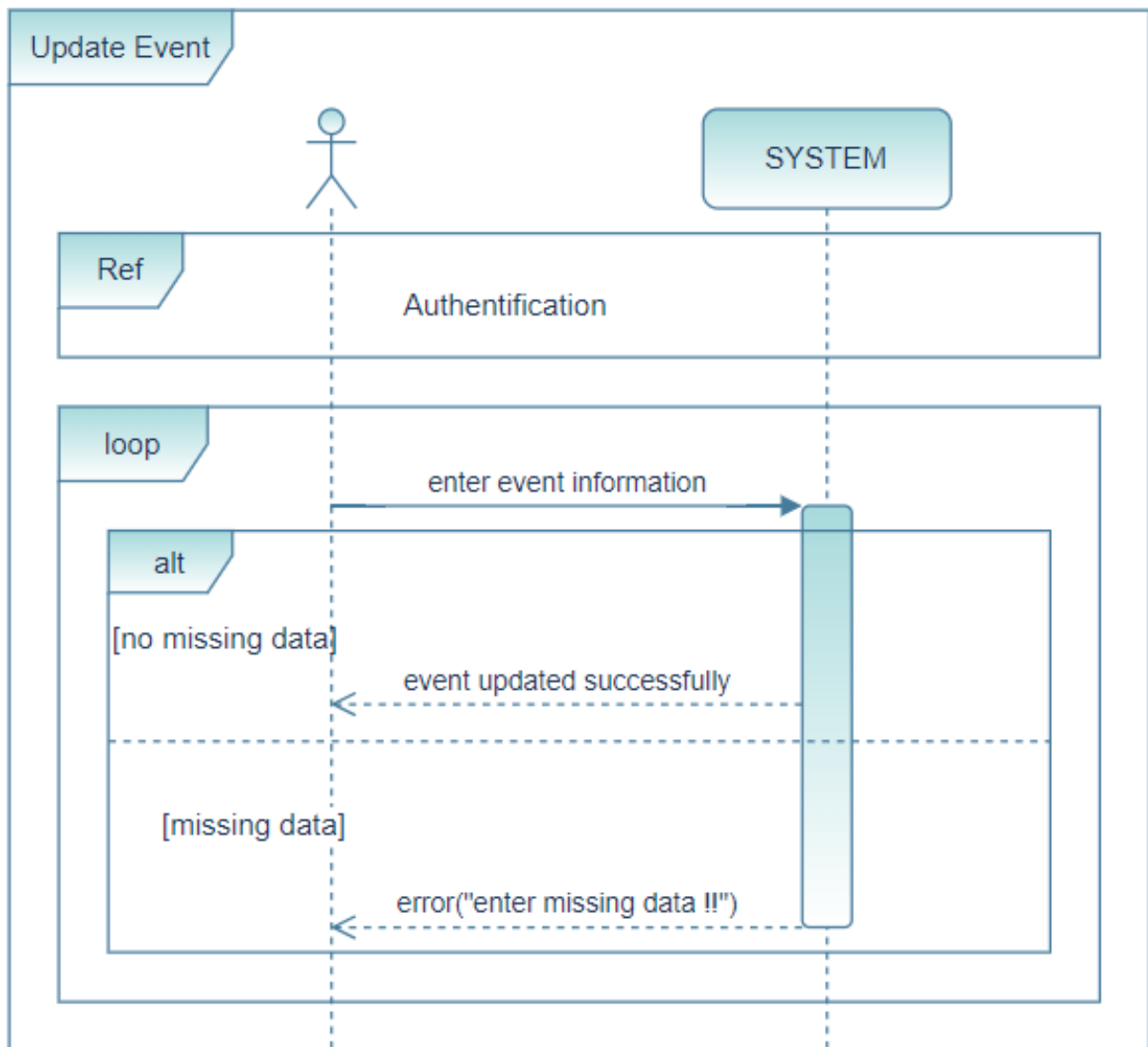


Figure 17: "Update Event" System Sequence Diagram

3.4. Add News

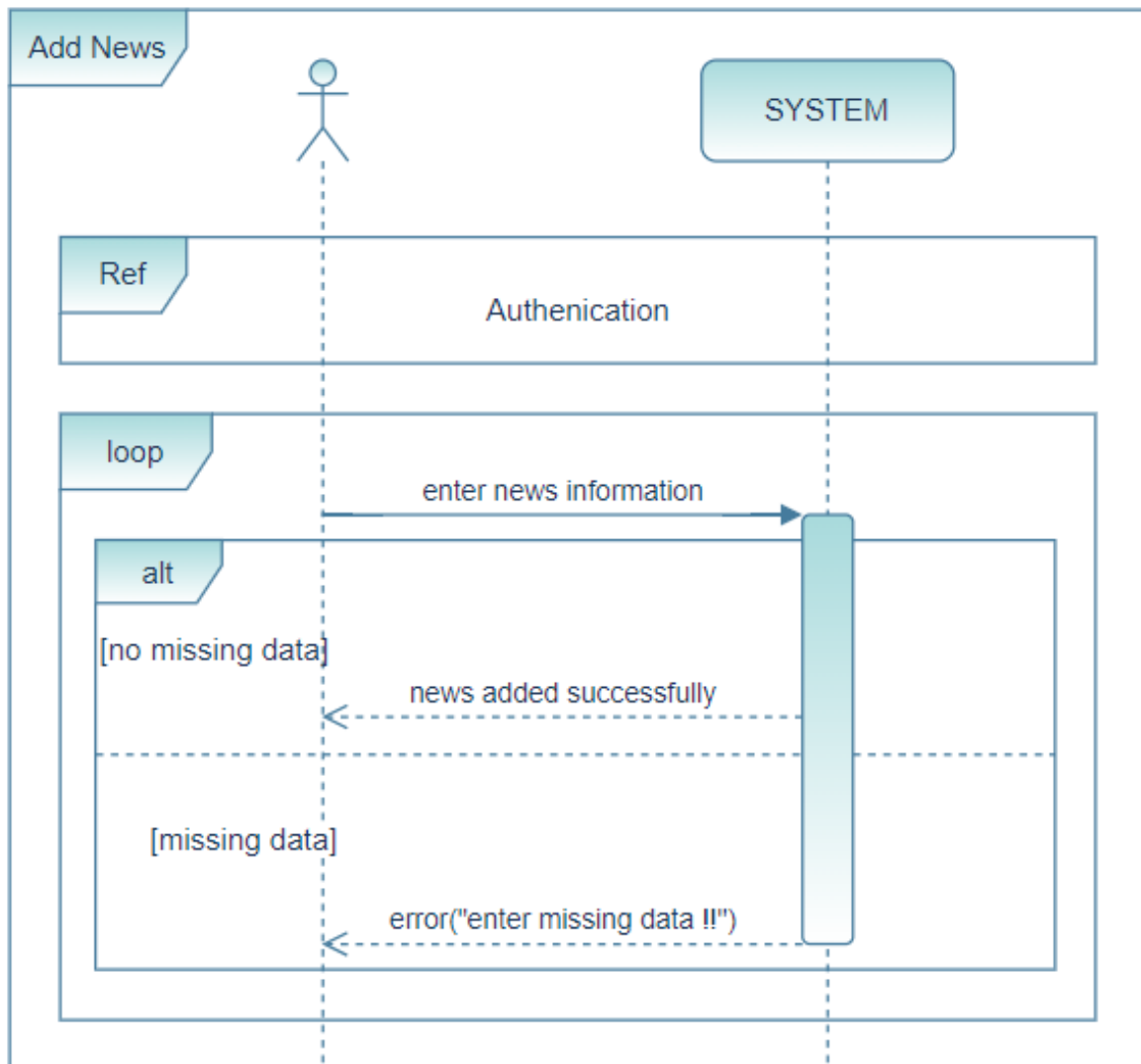


Figure 18: "Add News" System Sequence Diagram

3.5. Delete Article

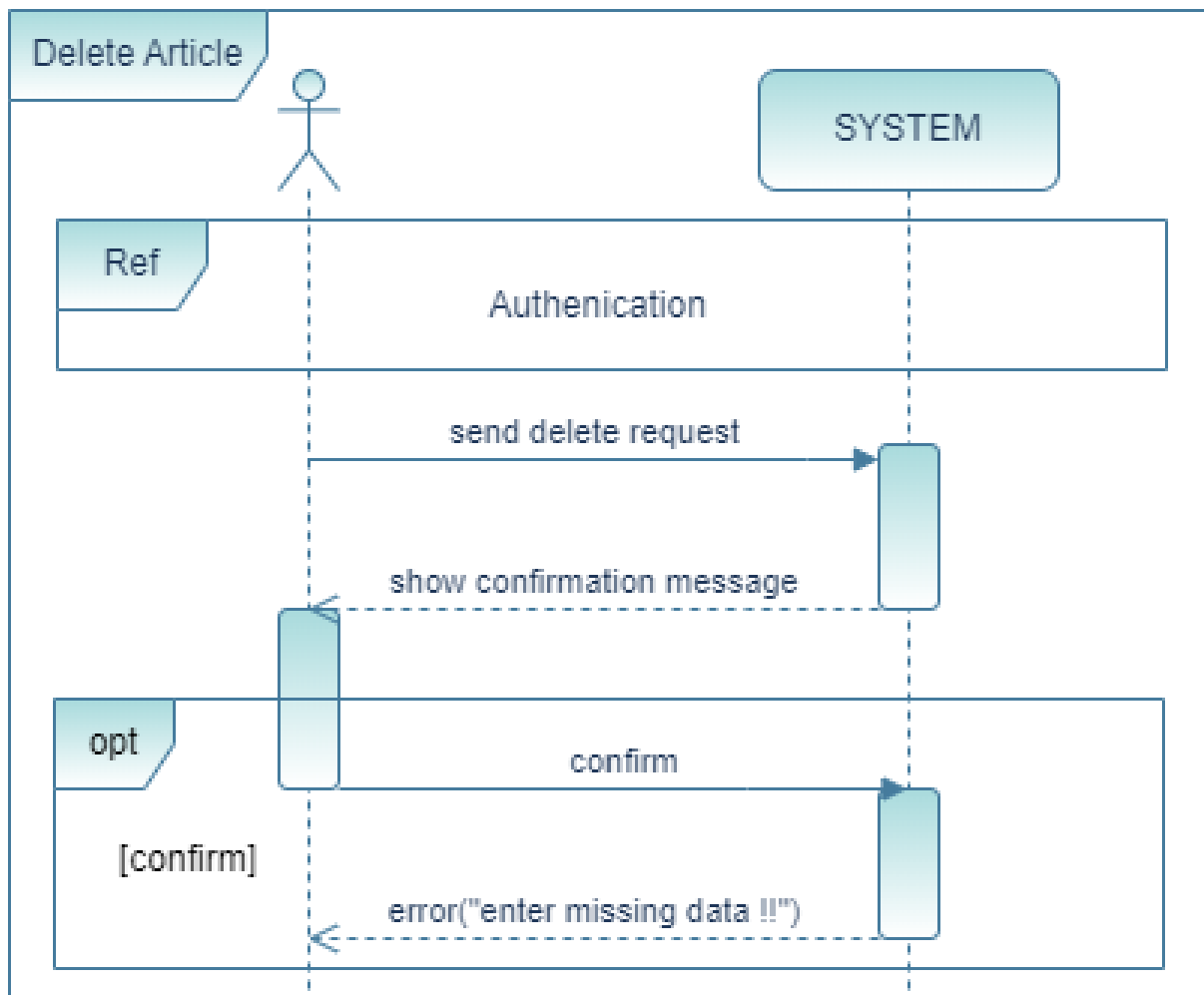


Figure 19: "Delete Article" System Sequence Diagram

CONCEPTION

1. Class Diagram

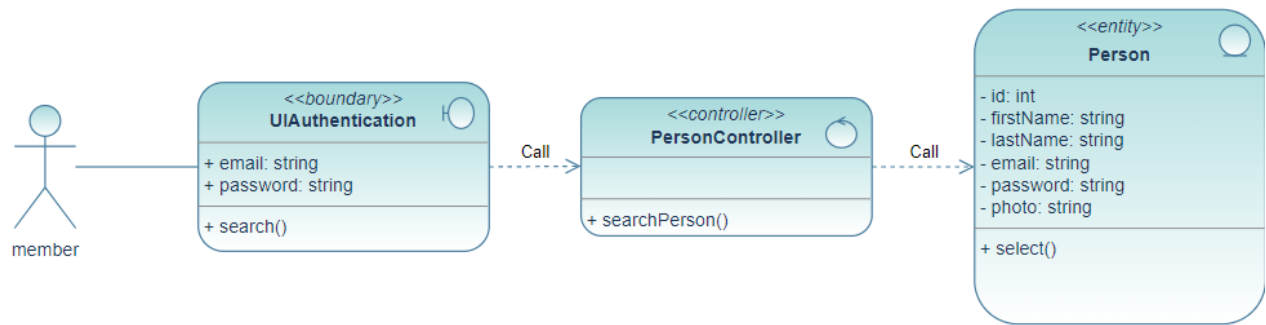


Figure 20: "Authentication" Class Diagram

1.1. Manage Account

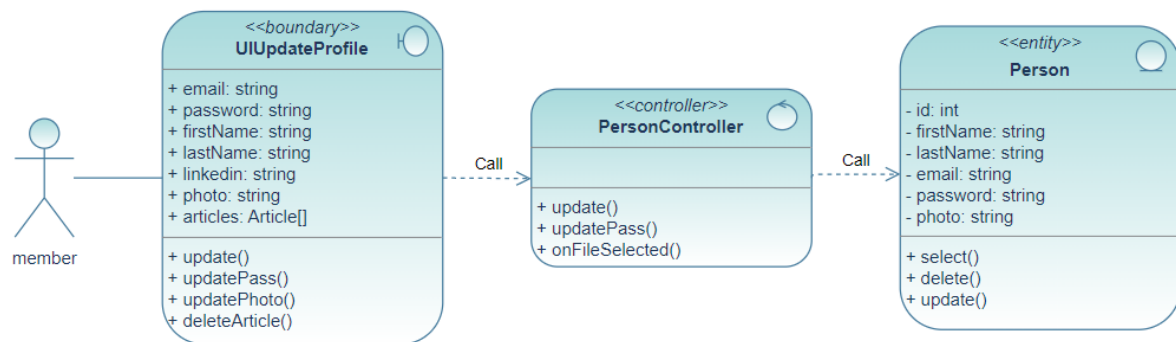


Figure 21: "Manage Account" Class Diagram

1.2. Manage News

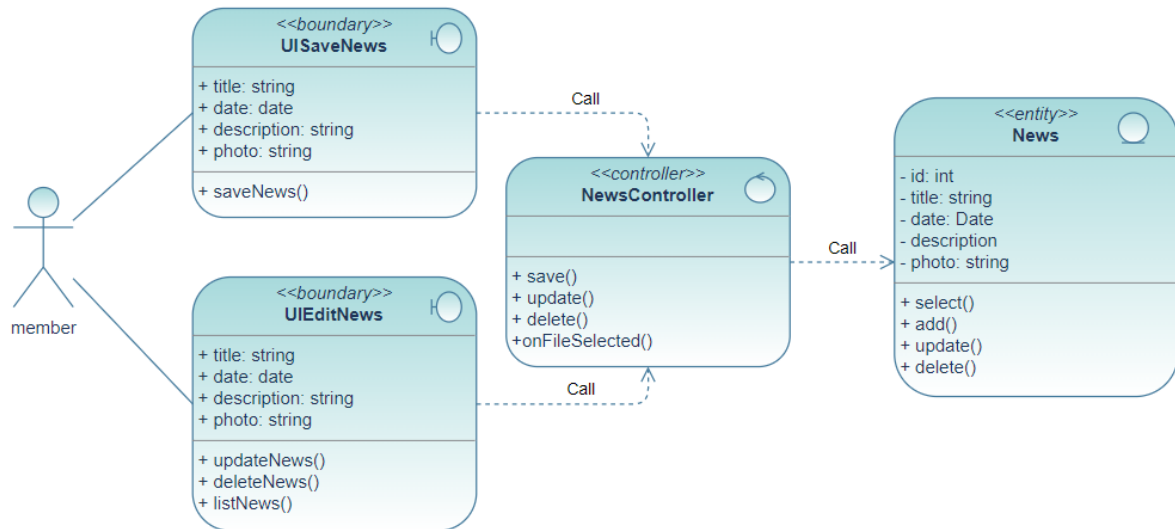


Figure 22: "Manage News" Class Diagram

1.3. Manage Events

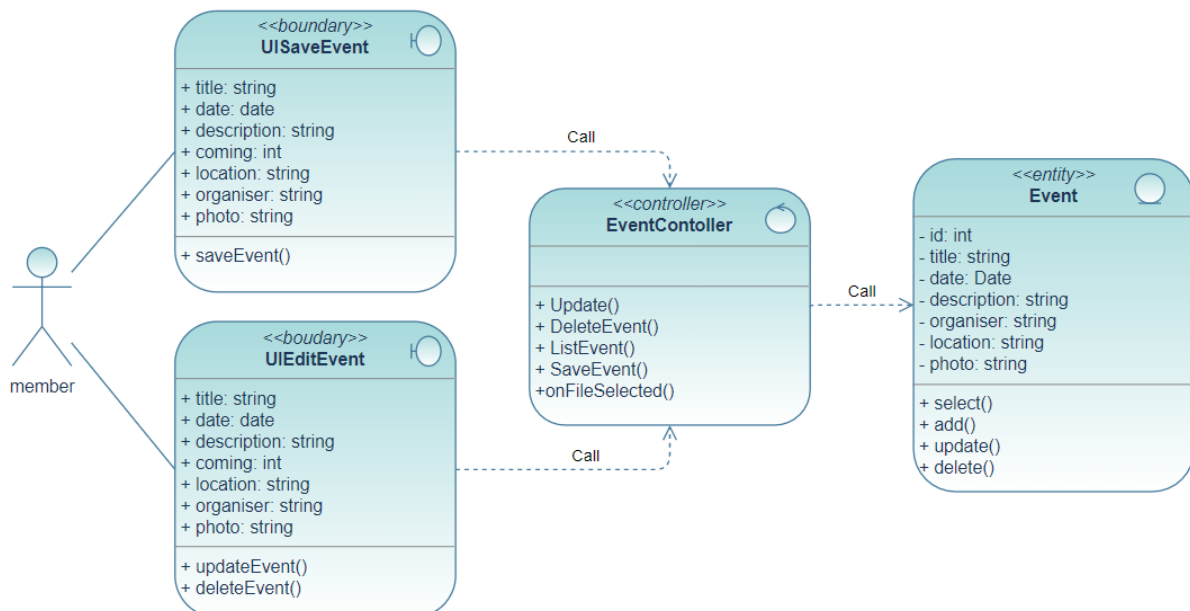


Figure 23: "Manage Events" Class Diagram

1.4. Manage Articles

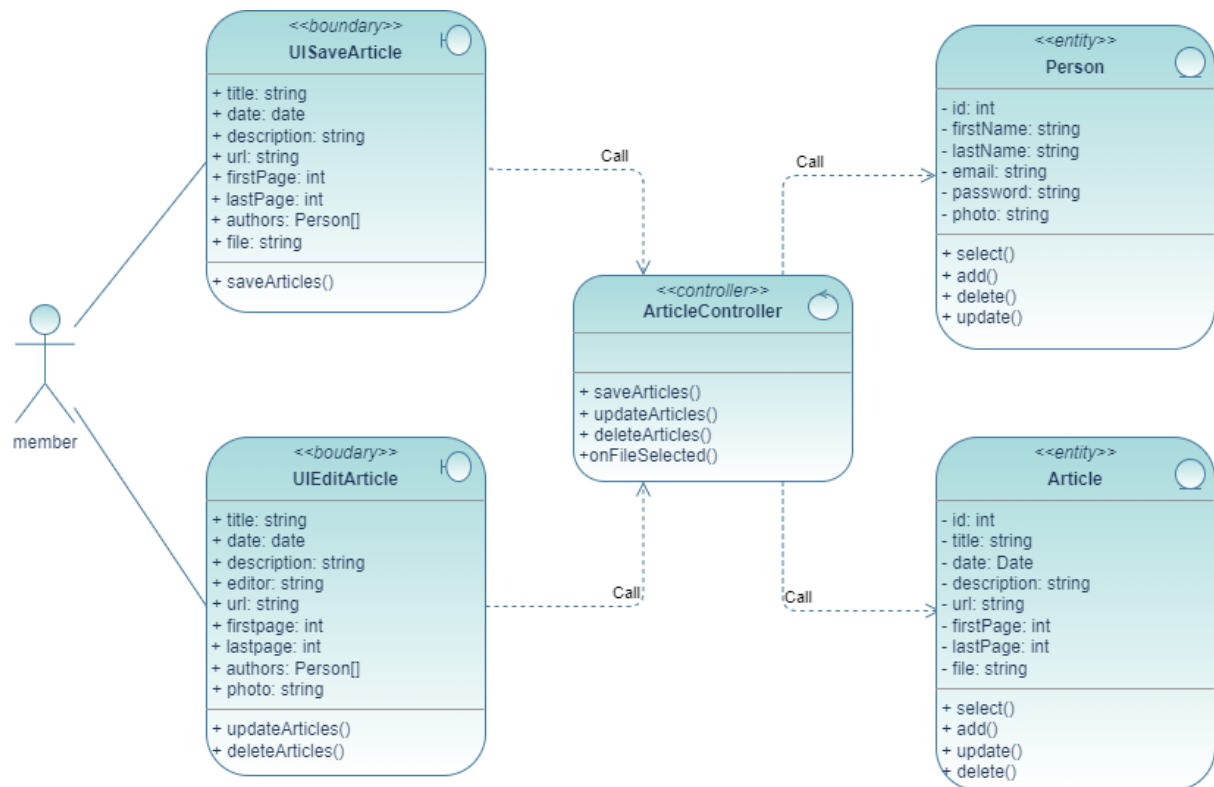


Figure 24: "Manage Article" Class Diagram

2. Detailed Sequence Diagram

2.1. Authentication

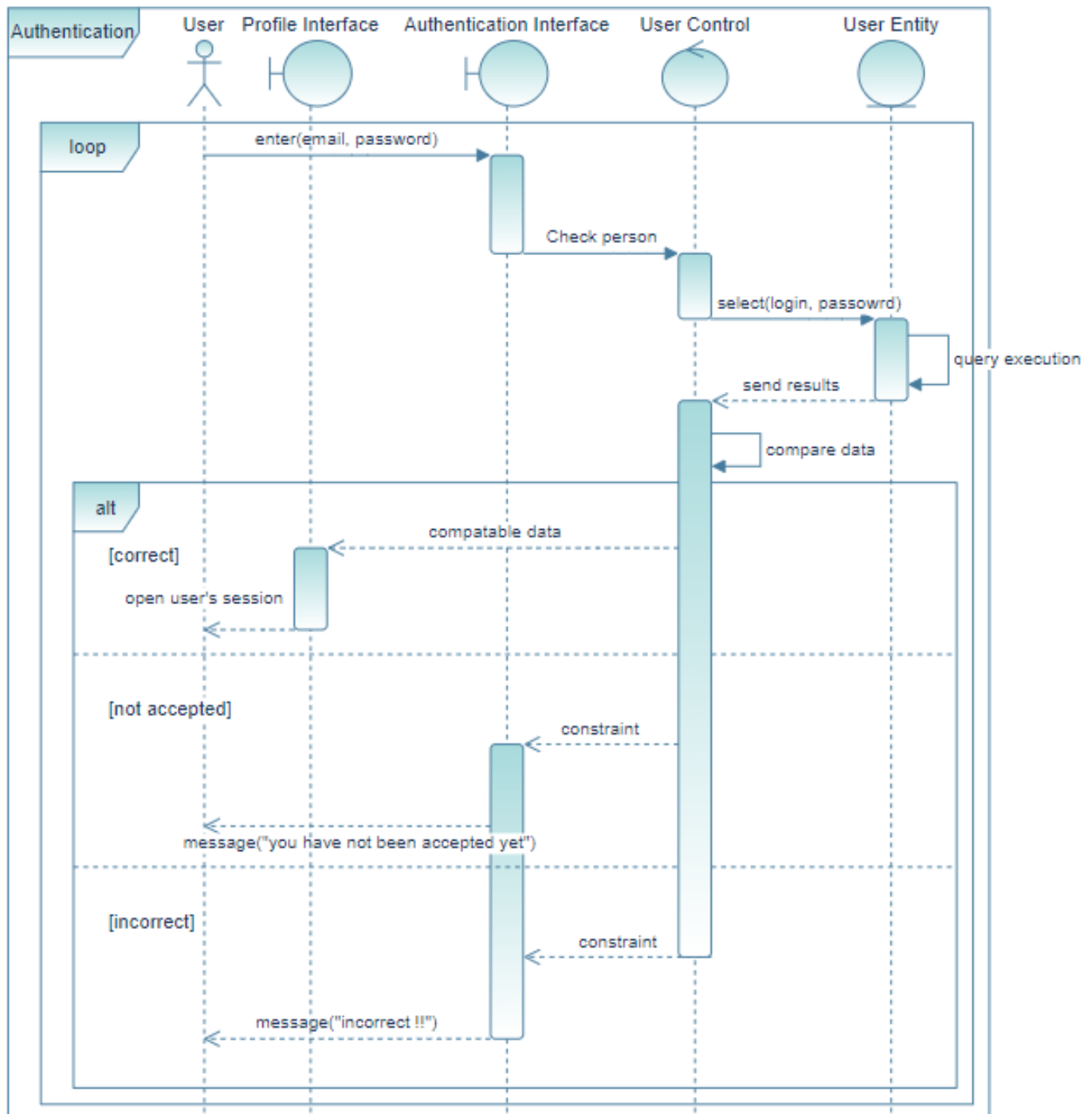


Figure 25: "Authentication" Detailed Sequence Diagram

2.2. Update Account

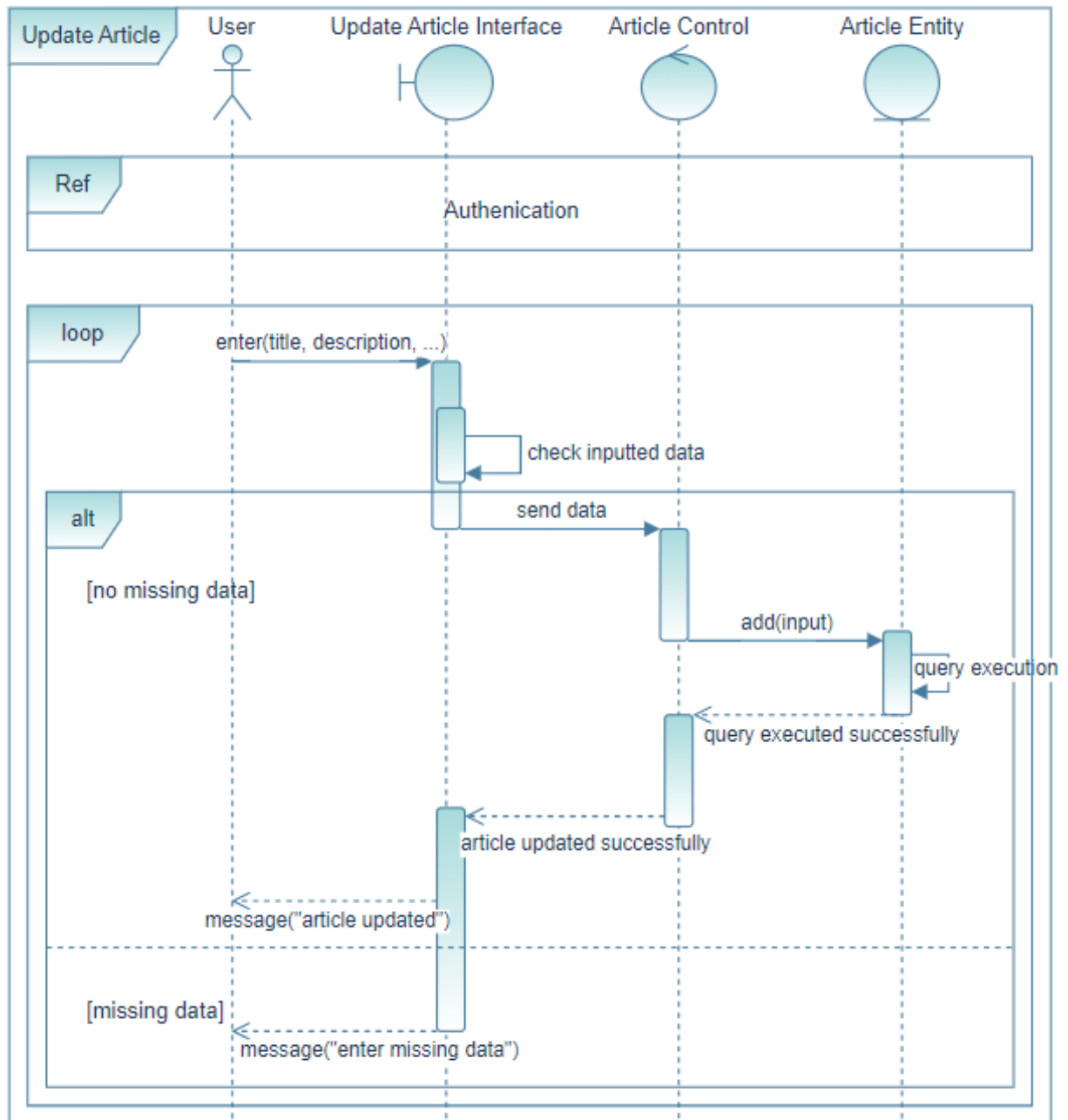


Figure 26: "Update Account" Detailed Sequence Diagram

2.3. Add News

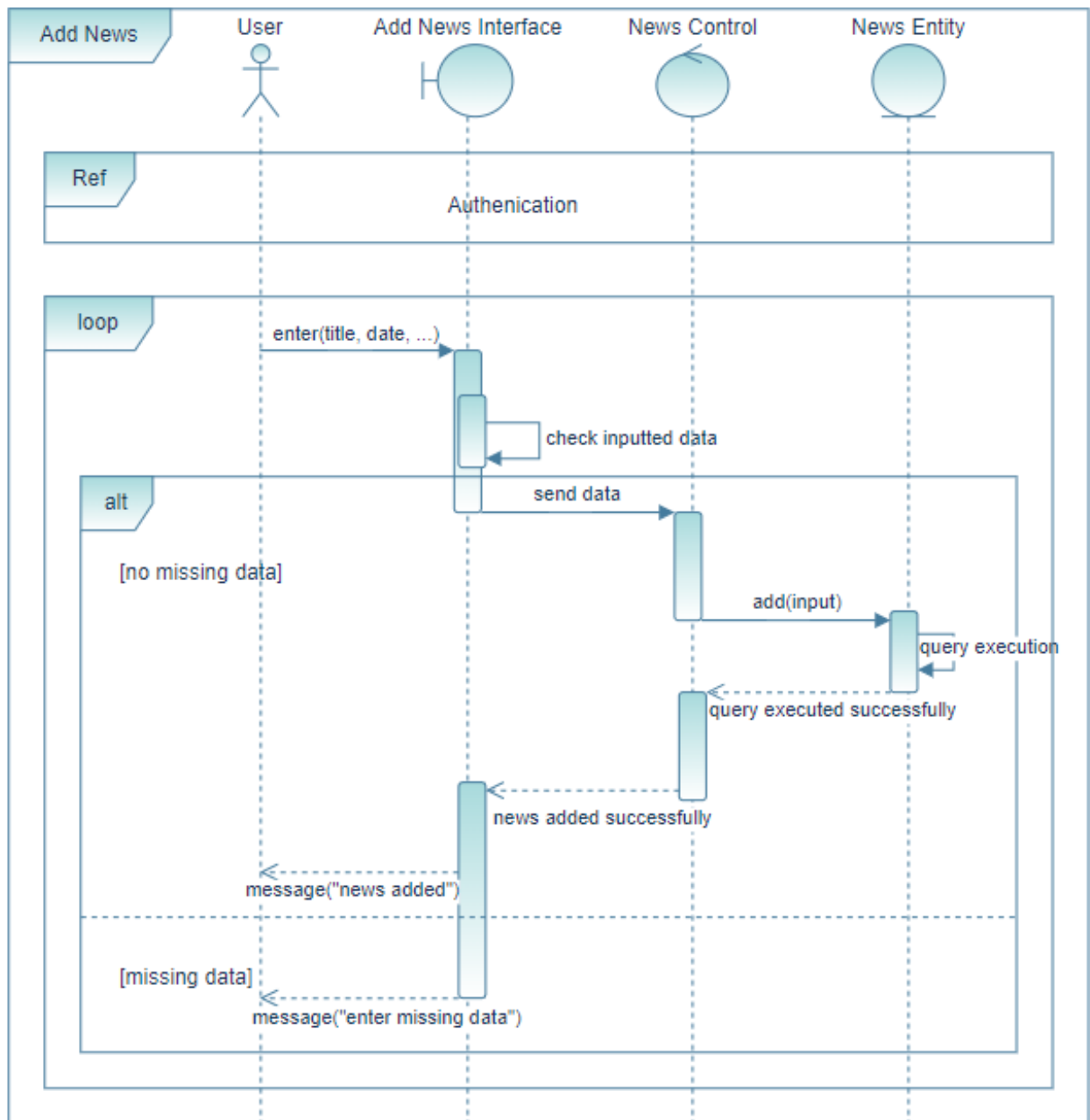


Figure 27: "Add News" Detailed Sequence Diagram

2.4. Update Event

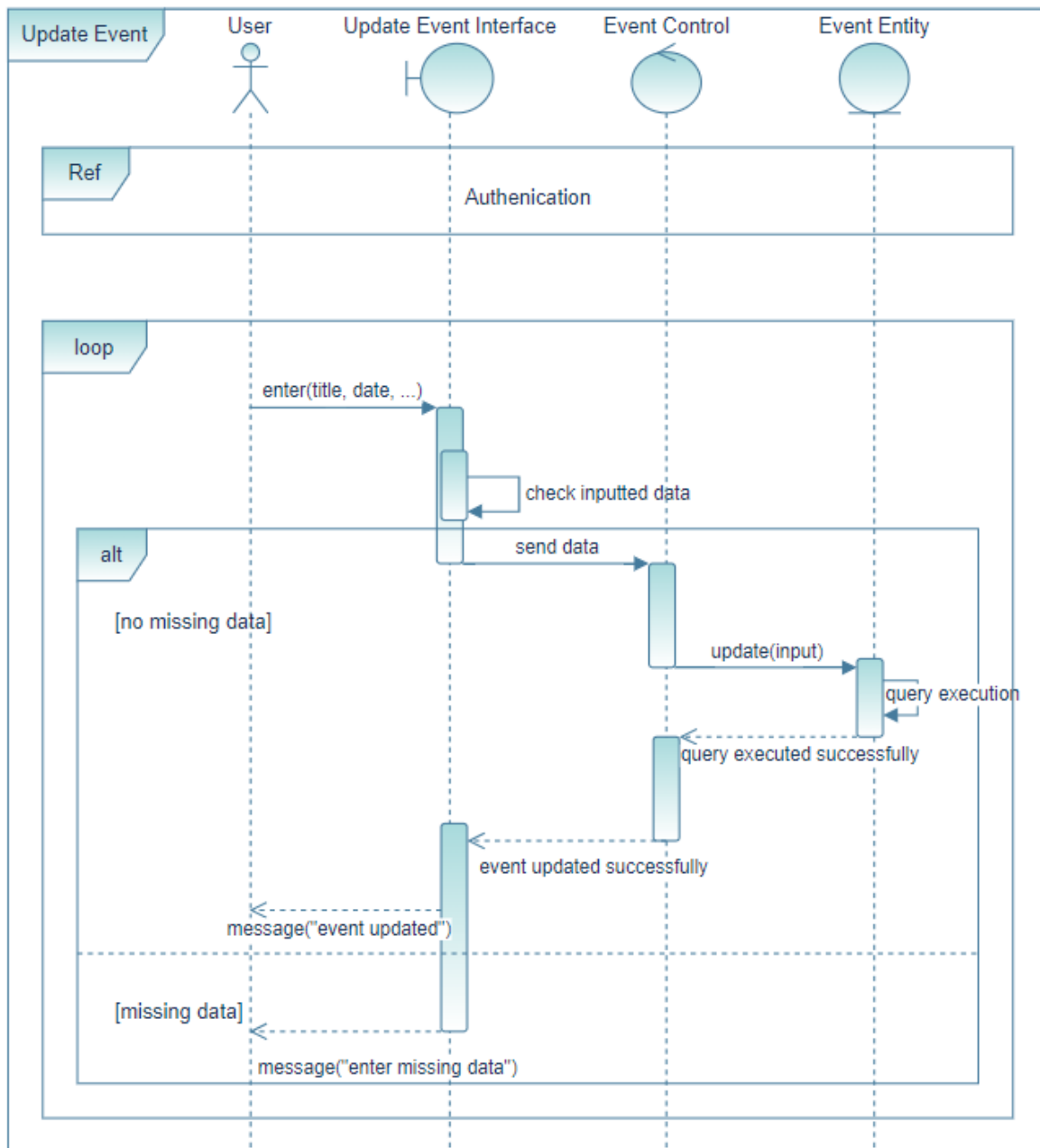


Figure 28: "Update Event" Detailed Sequence Diagram

2.5. Delete Article

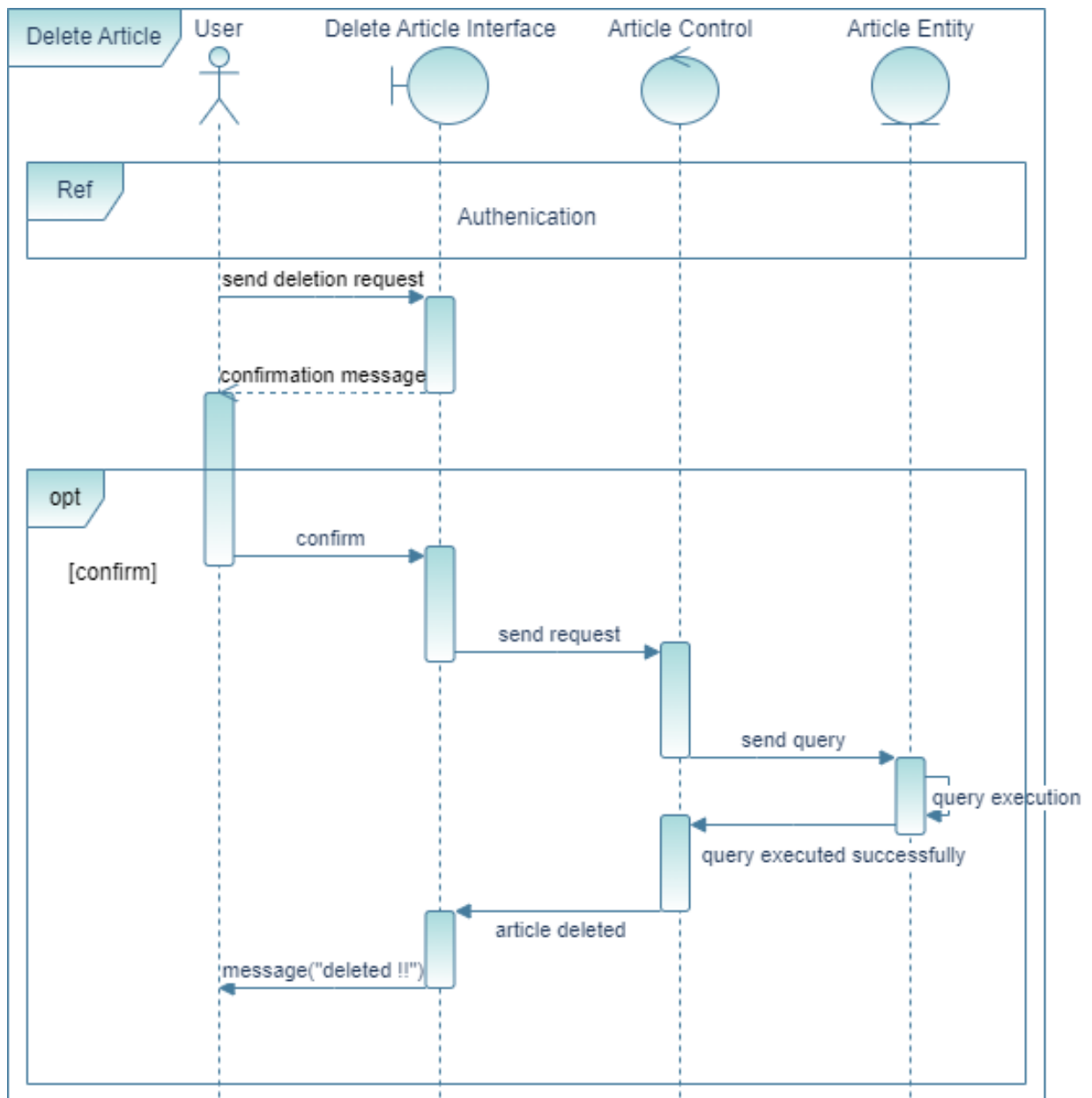


Figure 29: "Delete Article" Detailed Sequence Diagram

IMPLEMENTATION

1. Relational Schema

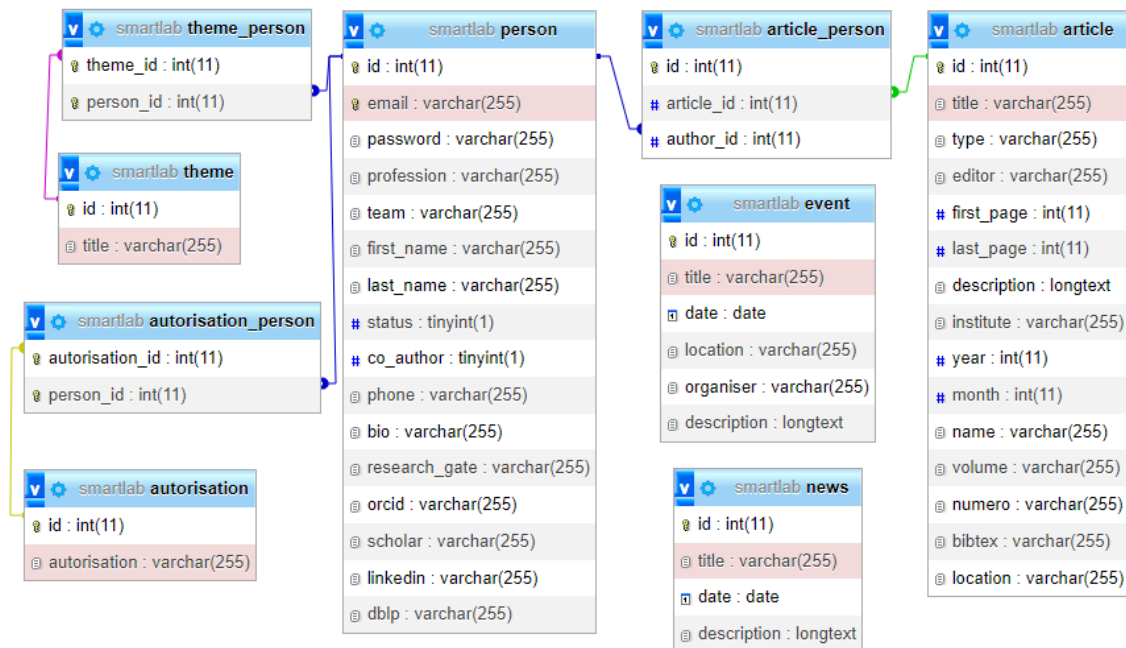


Figure 30: Sprint 1 Relational Schema

2. Interfaces

2.1. Authentication

The screenshot shows the 'Authentication' interface of the application. It features a login form on the left and a large blue banner on the right.

Login Form:

- Welcome Back !**: A heading in blue text.
- Email**: A label above a text input field containing the placeholder 'your e-mail ...'.
- Password**: A label above a text input field containing the placeholder 'your password ...'.
- Login**: A button below the password field.

Banner:

- A large blue banner with the text **Strategies for Modeling and Artificial Intelligence** in white and blue.

Figure 31: "Authentication" Interface

2.2. Manage Account



Update account

Update password

Choose File

Old password

your old password ...

Password

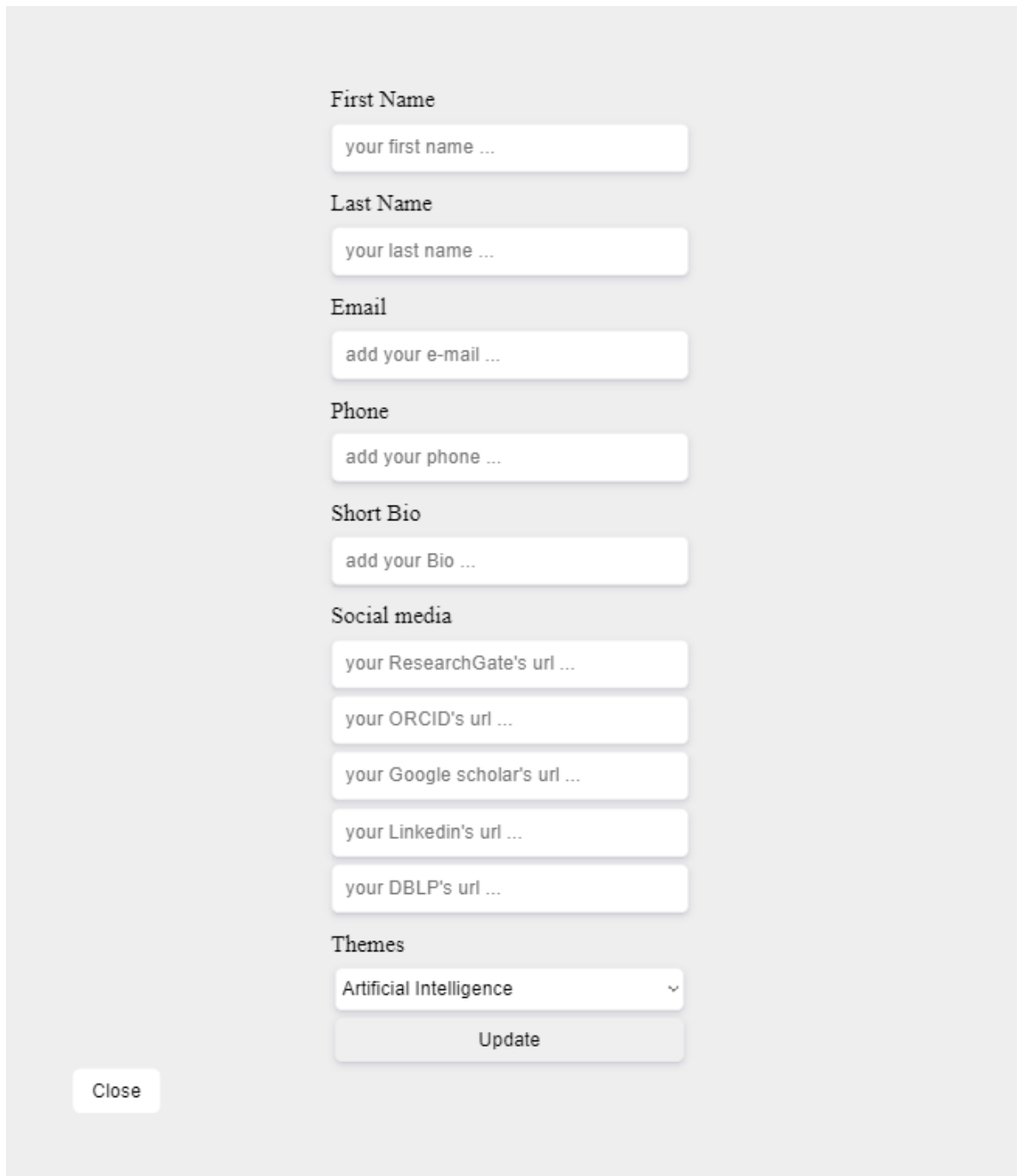
your new password ...

Confirm your password

Confirm your new password ...

Update your password

Close



The image shows a 'Update Account' interface with a light gray background. It contains several input fields for personal and professional information, a dropdown menu for themes, and an 'Update' button. A 'Close' button is located in the bottom left corner.

First Name
your first name ...

Last Name
your last name ...

Email
add your e-mail ...

Phone
add your phone ...

Short Bio
add your Bio ...

Social media

- your ResearchGate's url ...
- your ORCID's url ...
- your Google scholar's url ...
- your LinkedIn's url ...
- your DBLP's url ...

Themes
Artificial Intelligence ▾

Update

Close

Figure 32: "Update Account" Interface

2.3. Add News

News

Title

Image

No file chosen

Description

Submit

Figure 33: "Add News" Interface

2.4. Update Event

Event

Title

New Event

Location

ISG

Organiser

SMARTLAB

Date

11/05/2023

Image

Choose File

No file chosen

Description

posting new event, organised by SMARTLAB at ISG

Submit

Figure 34: "Update Event" Interface

2.5. Delete Article

Title	Type	Date	ADD ARTICLE	
testing file	TYPE	24-04-2023	Update	Delete
2	TYPE	21-04-2023	Update	Delete
111	TYPE	21-04-2022	Update	Delete
testing multiple authors	TYPE3	21-04-2021	Update	Delete
first article	TYPE1	20-04-2021	Update	Delete

Figure 35: "Delete Article" Interface

Conclusion

This chapter presents a detailed analysis of the second sprint, that is consisted of tasks assigned to the member actor, as well as the conception, and finally an implementation.

Chapter Five: Sprint 3: Building Administrator Back Office

INTRODUCTION

In this chapter, we will dive into the details of the second sprint of our project, which mainly carries the functionalities of the “administrator” actor.

SPRINT BACKLOG

The table 16 represents the sprint backlog that enumerate the functionalities of this sprint:

Table 16: Sprint 3 Backlog

ID	User story	task	task
1	As an administrator of the platform, I can update the members' privileges.	1.A	Elaborate the use case, sequence, and class diagrams for the “Update Privilege” functionality
		1.B	Develop the “Update Privileges” use case.
		1.C	Test the “Update Privileges” use case.
2	As an administrator of the platform, I can update the members' professions.	2.A	Develop the use case, sequence, and class diagrams for the “Update Profession” functionality
		2.B	Develop the “Update Profession” use case.
		2.C	Test the “Update Profession” use case.
3	As an administrator of the platform, I can update the members' teams.	3.A	Develop the use case, sequence, and class diagrams for the “Update Teams” functionality
		3.B	Develop the “Update Teams” use case.

		3.C	Test the “Update Teams” use case.
4	As an administrator of the platform, I can delete members.	4.A	Develop the use case, sequence, and class diagrams for the “Delete Members” functionality
		4.B	Develop the “Delete Members” use case.
		4.C	Test the “Delete Members” use case.
5	As an administrator of the platform, I can add partners to the website.	5.A	Develop the use case, sequence, and class diagrams for the “Add Partners” functionality
		5.B	Develop the “Add Partners” use case.
		5.C	Test the “Add Partners” use case.
6	As an administrator of the platform, I can update partners in the website.	6.A	Develop the use case, sequence, and class diagrams for the “Update Partners” functionality
		6.B	Develop the “Update Partners” use case.
		6.C	Test the “Update Partners” use case.
7	As an administrator of the platform, I can delete Partners from the website.	7.A	Develop the use case, sequence, and class diagrams for the “Delete Partners” functionality
		7.B	Develop the “Delete Partners” use case.
		7.C	Test the “Delete Partners” use case.
8	As an administrator of the platform, I can add Project from the website.	8.A	Develop the use case, sequence, and class diagrams for the “Add Project” functionality

		8.B	Develop the “Add Project” use case.
		8.C	Test the “Add Project” use case.
9	As an administrator of the platform, I can Update Project from the website.	9.A	Develop the use case, sequence, and class diagrams for the “Update Project” functionality
		9.B	Develop the “Update Project” use case.
		9.C	Test the “Update Project” use case.
10	As an administrator of the platform, I can Delete Project from the website.	10.A	Develop the use case, sequence, and class diagrams for the “Delete Project” functionality
		10.B	Develop the “Delete Project” use case.
		10.C	Test the “Delete Project” use case.
11	As an administrator of the platform, I can Download Pdf from the website.	11.A	Develop the use case, sequence, and class diagrams for the “Reporting” functionality
		11.B	Develop the “Reporting” use case.
		11.C	Test the “Reporting” use case.

FUNCTIONAL SPECIFICATIONS

This figure 34 illustrates the main functionalities for the “administrator” actor:

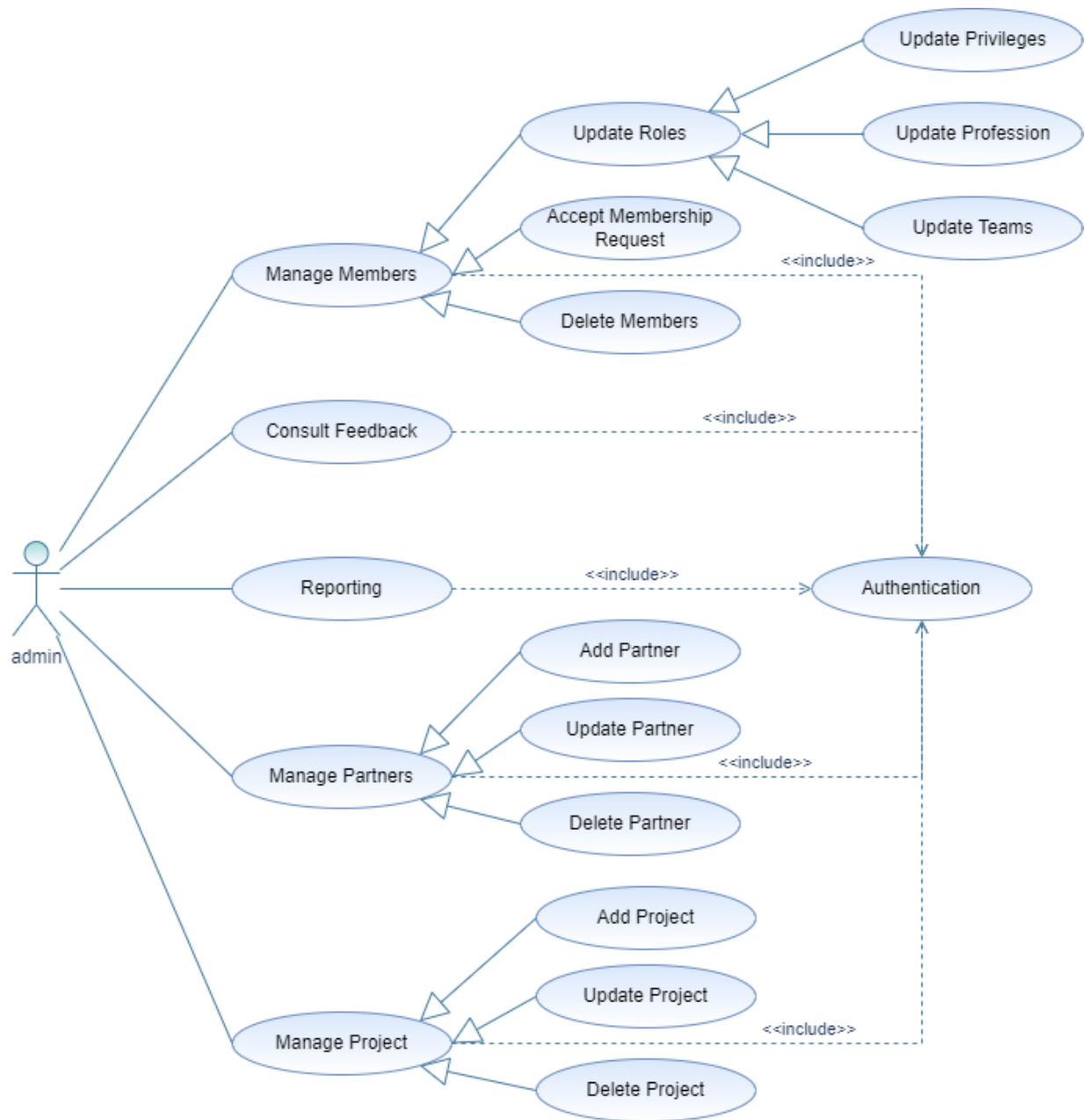


Figure 36: Administrator's Use Case Diagram

ANALYSIS

For further analysis of this sprint, we are diving into the details of each functionality that has been mentioned above.

1. Use Case Diagram

1.1. Manage Partners

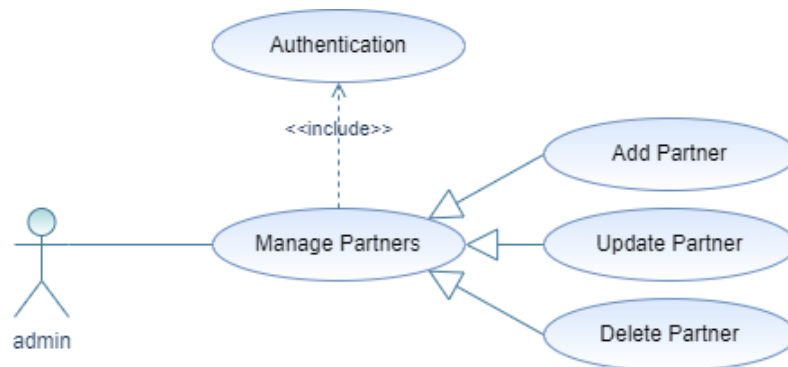


Figure 37: "Manage Partners" Use Case Diagram

1.2. Manage Projects

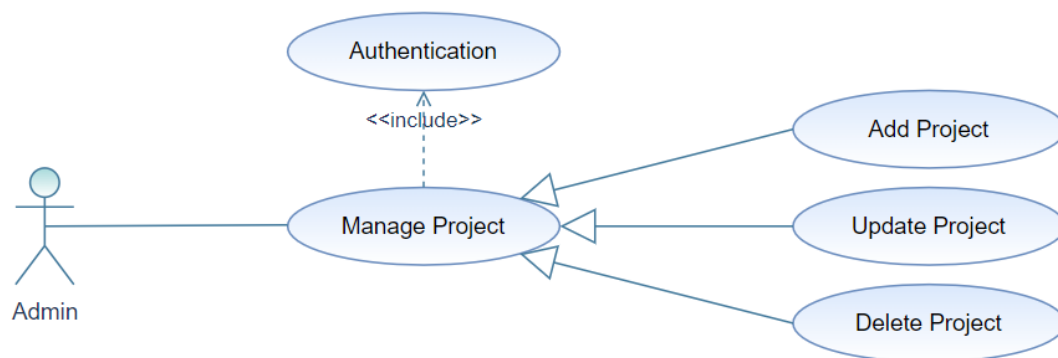


Figure 38: "Manage Project" Use Case Diagram

1.3. Manage Members

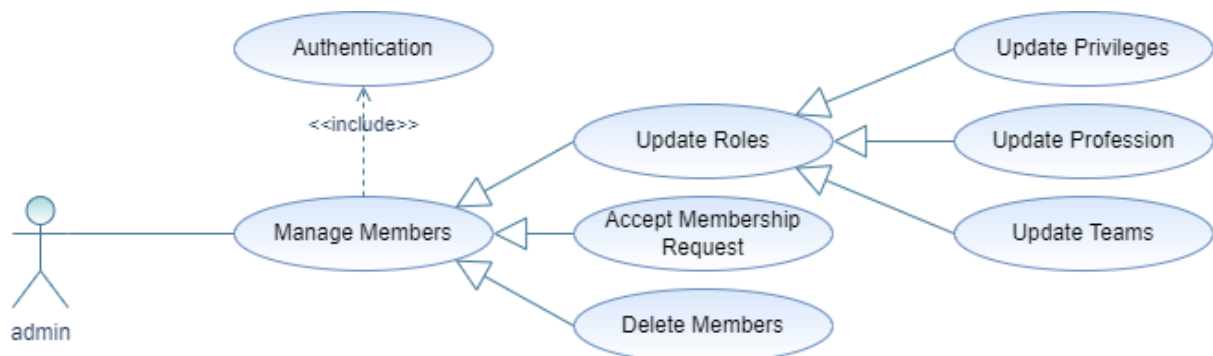


Figure 39: "Manage Members" Use Case Diagram

1.4. Reporting



Figure 40: "Reporting» Use Case Diagram

2. Textual Description

2.1. Add Partner

Table 17: "Add Partner" Textual Description

Use Case	Add Partner
Actor	Administrator
Pre-Condition	Being Authenticated
Post-Condition	Partner added successfully
Basic Scenario	<ol style="list-style-type: none">1. The administrator clicks "Add Partner".2. The system displays the partner adding form.3. The administrator fills the form and clicks "submit".4. The system adds new partner to the database.
Exceptional Scenario	<ol style="list-style-type: none">4.a. If the administrator fails to fill some of the mandatory fields, the system will display a warning.
Alternative Scenario	None.

2.2. Delete Partner

Table 18: "Delete Partner" Textual Description

Use Case	Delete Partner
Actor	Administrator
Pre-Condition	Being Authenticated
Post-Condition	Partner Deleted successfully

<i>Basic Scenario</i>	<ol style="list-style-type: none"> 1. The administrator selects a partner and clicks “Delete Partner”. 2. System displays a confirmation dialog, “Are you sure?”. 3. The administrator clicks “Yes”. 4. System deletes the partner from the database.
<i>Exceptional Scenario</i>	None.
<i>Alternative Scenario</i>	<ol style="list-style-type: none"> 3.a. The administrator clicks “No”. 3.b. The system cancels the task.

2.3. Update Project

Table 19: "Update Partner" Textual Description

<i>Use Case</i>	<i>Update Partner</i>
<i>Actor</i>	Administrator
<i>Pre-Condition</i>	Being Authenticated
<i>Post-Condition</i>	Partner updated successfully
<i>Basic Scenario</i>	<ol style="list-style-type: none"> 1. The administrator clicks “Update Project”. 2. The system displays the partner updating form. 3. The administrator fills the form and clicks “submit”. 4. The system updates project to the database.
<i>Exceptional Scenario</i>	<ol style="list-style-type: none"> 4.a. If the administrator fails to fill some of the mandatory fields, the system will display a warning.
<i>Alternative Scenario</i>	None.

2.4. Accept Membership Request

Table 20: "Accept Membership" Textual Description

<i>Use Case</i>	<i>Accept Membership</i>
<i>Actor</i>	<i>Administrator</i>
<i>Pre-Condition</i>	<i>Being Authenticated</i>
<i>Post-Condition</i>	<i>New member added successfully</i>
<i>Basic Scenario</i>	<ol style="list-style-type: none">1. The administrator goes to the "Requests" interface.2. The system displays all the membership requests.3. The administrator clicks "accept".4. The system adds new member to the database.
<i>Exceptional Scenario</i>	<i>None.</i>
<i>Alternative Scenario</i>	<ol style="list-style-type: none">3.a. The administrator clicks "delete".3.b. The system deletes the request.

2.5. Update Privileges

Table 21: "Update Privileges" Textual Description

<i>Use Case</i>	<i>Update Privileges</i>
<i>Actor</i>	<i>Administrator</i>
<i>Pre-Condition</i>	<i>Being Authenticated</i>
<i>Post-Condition</i>	<i>The members' privileges are updated successfully</i>
<i>Basic Scenario</i>	<ol style="list-style-type: none">1. The administrator goes to the "Members" interface.2. The system displays all the members with their corresponding privileges.3. The administrator alters the privileges for each member and clicks "submit".4. The system alters the members' privileges in the database.
<i>Exceptional Scenario</i>	<i>None.</i>

<i>Alternative Scenario</i>	<i>None.</i>
-----------------------------	--------------

2.6. Update Teams

Table 22: "Update Teams" Textual Description

<i>Use Case</i>	<i>Update Teams</i>
<i>Actor</i>	<i>Administrator</i>
<i>Pre-Condition</i>	<i>Being Authenticated</i>
<i>Post-Condition</i>	<i>The members' teams are updated successfully</i>
<i>Basic Scenario</i>	<ol style="list-style-type: none"> 1. The administrator goes to the "Members" interface. 2. The system displays all the members with their corresponding teams. 3. The administrator alters the teams for each member and clicks "submit". 4. The system alters the members' teams in the database.
<i>Exceptional Scenario</i>	<i>None.</i>
<i>Alternative Scenario</i>	<i>None.</i>

2.7. Delete Member

Table 23: "Delete Member" Textual Description

<i>Use Case</i>	<i>Delete Partner</i>
<i>Actor</i>	<i>Administrator</i>
<i>Pre-Condition</i>	<i>Being Authenticated</i>
<i>Post-Condition</i>	<i>Member Deleted successfully</i>
<i>Basic Scenario</i>	<ol style="list-style-type: none"> 1. The administrator goes to the "Members" interface. 2. The administrator selects a member and clicks on "Delete". 3. The System displays a confirmation dialog, "Are you sure?".

	4. The administrator clicks “Yes”. 5. System deletes the member from the database.
<i>Exceptional Scenario</i>	<i>None.</i>
<i>Alternative Scenario</i>	4.a. The administrator clicks “No”. 4.b. The system cancels the task.

2.8. Reporting

Table 24: "Reporting" Textual Description

<i>Use Case</i>	<i>Reporting</i>
<i>Actor</i>	<i>Administrator</i>
<i>Pre-Condition</i>	<i>Being Authenticated</i>
<i>Post-Condition</i>	<i>PDF Downloaded</i>
<i>Basic Scenario</i>	1. The administrator goes to the “Reporting” interface. 2. The administrator chooses the year and subjects of the reporting. 3. The administrator clicks “Generate PDF”. 4. The administrator clicks “Download PDF”.
<i>Exceptional</i>	2.a. If the administrator fails to fill the year or at least one of the subjects.
<i>Alternative Scenario</i>	<i>None.</i>

3. System Sequence Diagram

3.1. Add Partner

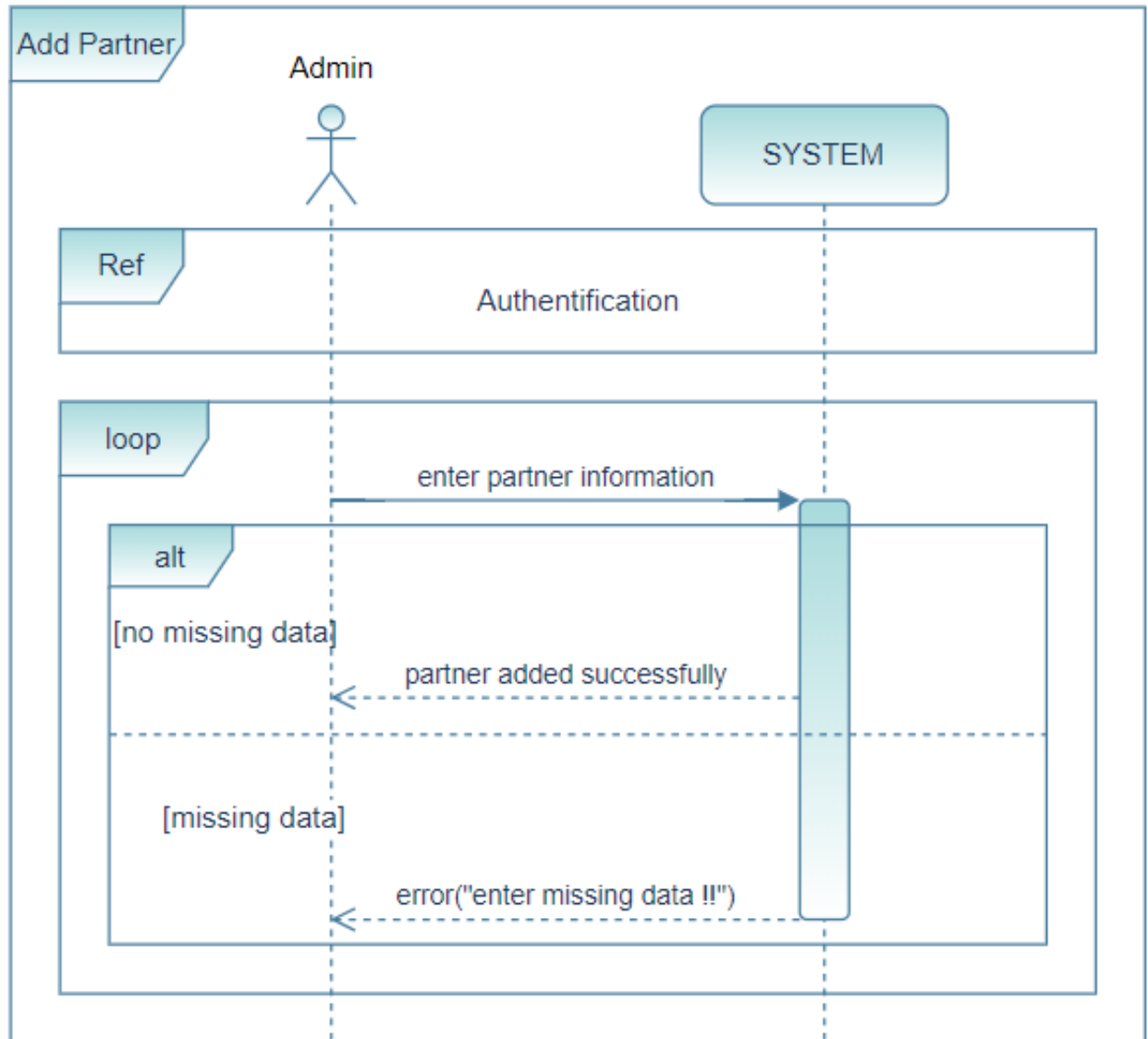


Figure 41: "Add Partner" System Sequence Diagram

3.2. Delete Partner

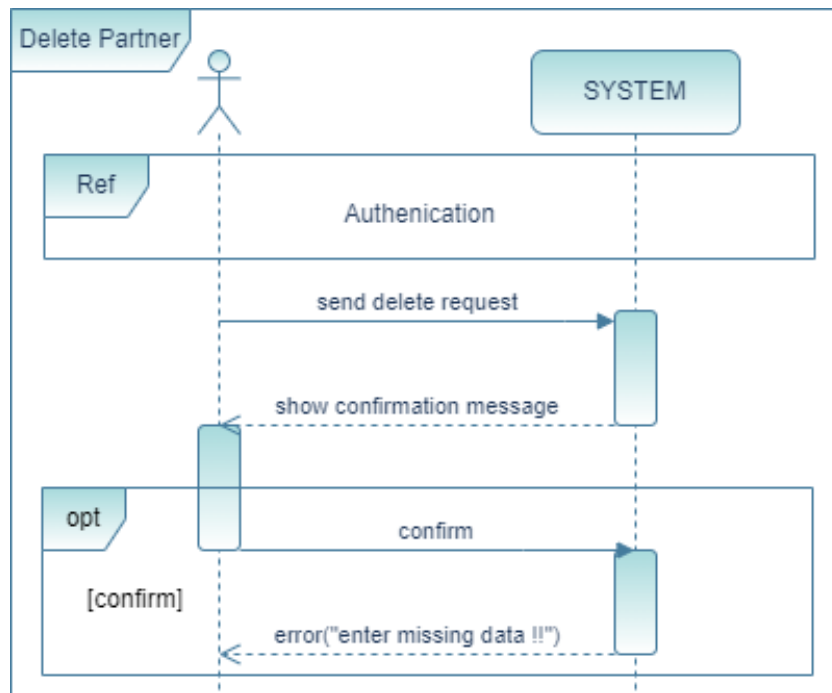


Figure 42: "Delete Partner" System Sequence Diagram

3.3. Update Project

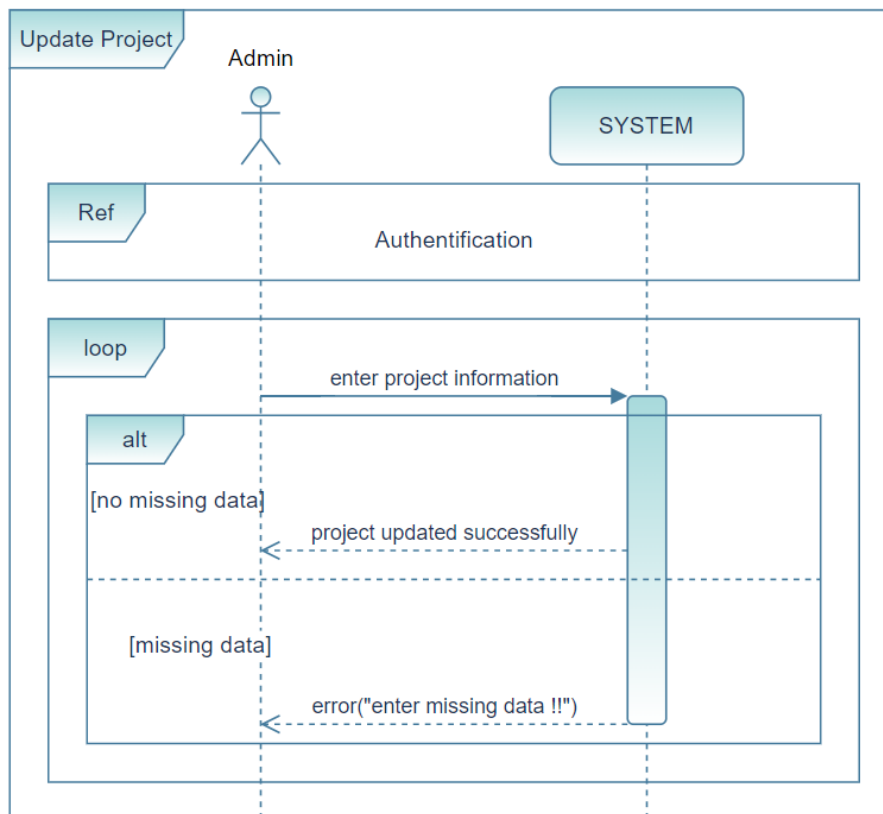


Figure 43: "Update Project" System Sequence Diagram

3.4. Accept Membership Request

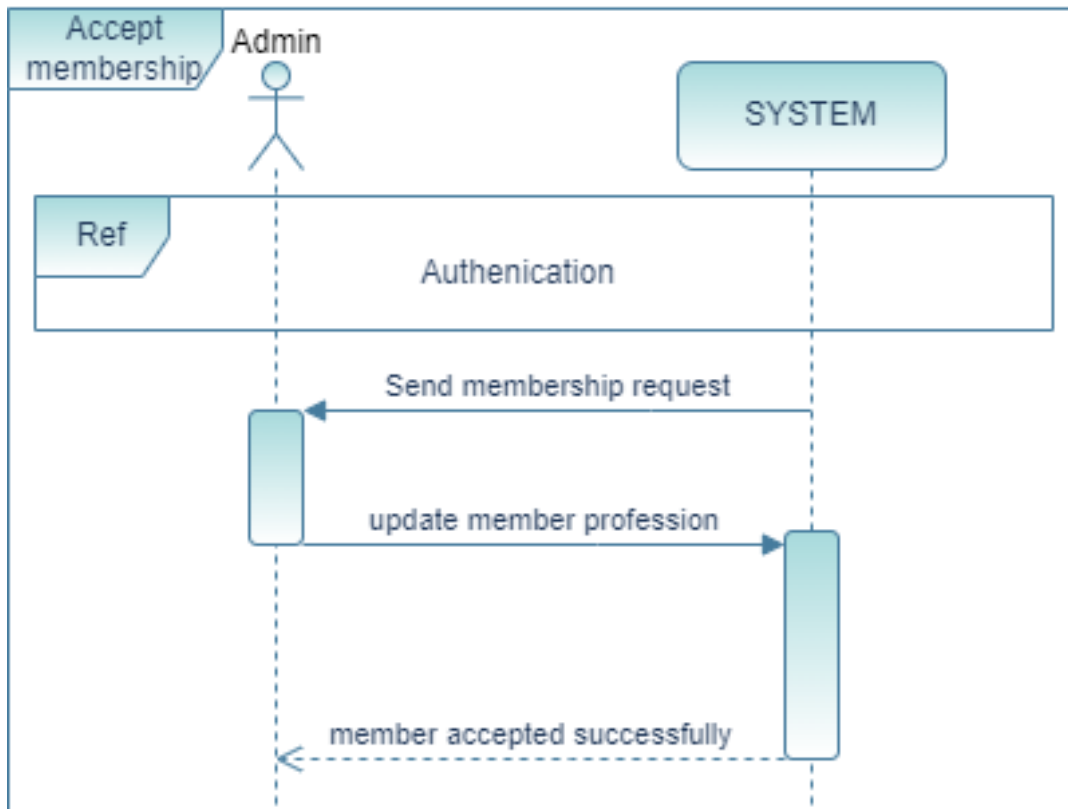


Figure 44: "Accept Membership" System Sequence Diagram

3.5. Update Team

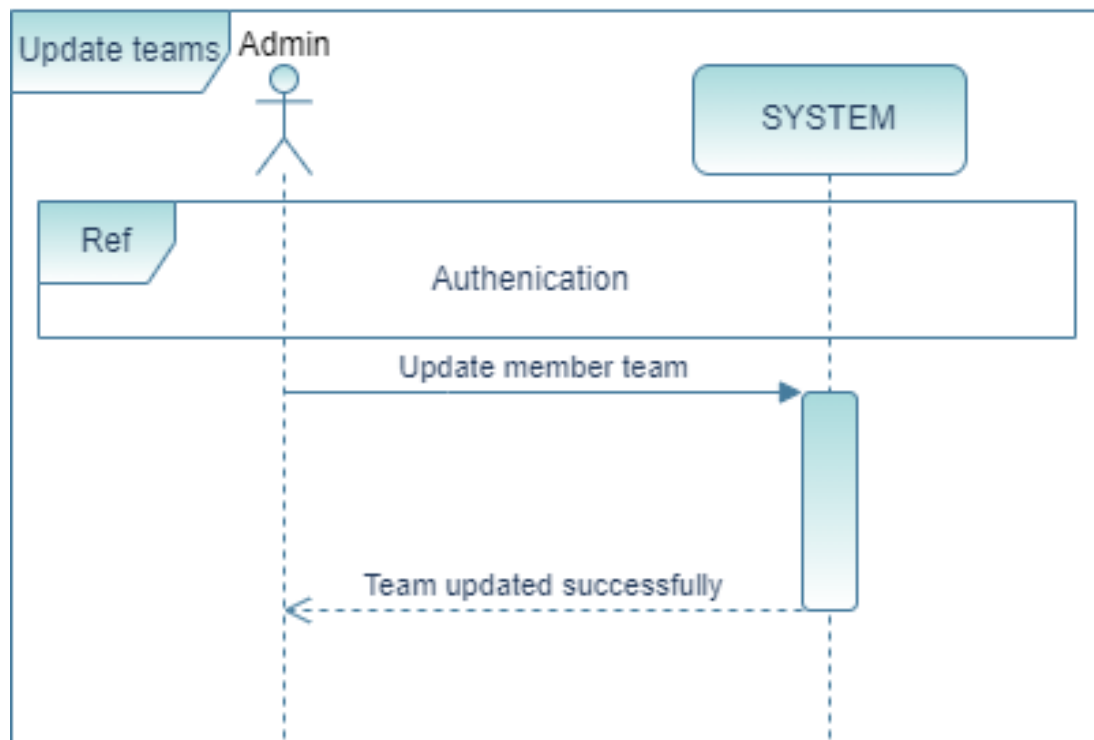


Figure 45: "Update Teams" System Sequence Diagram

3.6. Delete Member

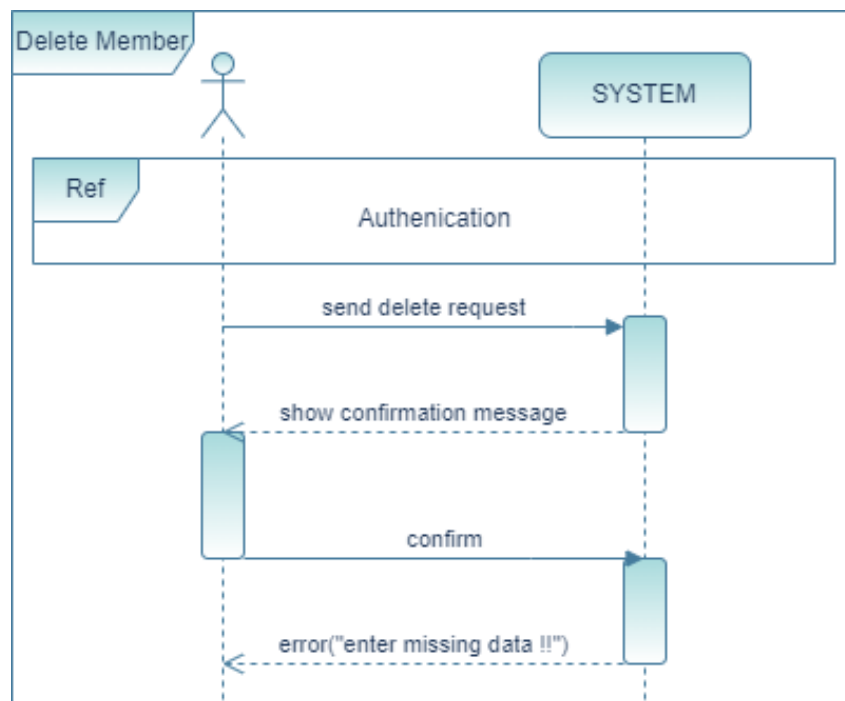


Figure 46: "Delete Member" System Sequence Diagram

3.7. Reporting

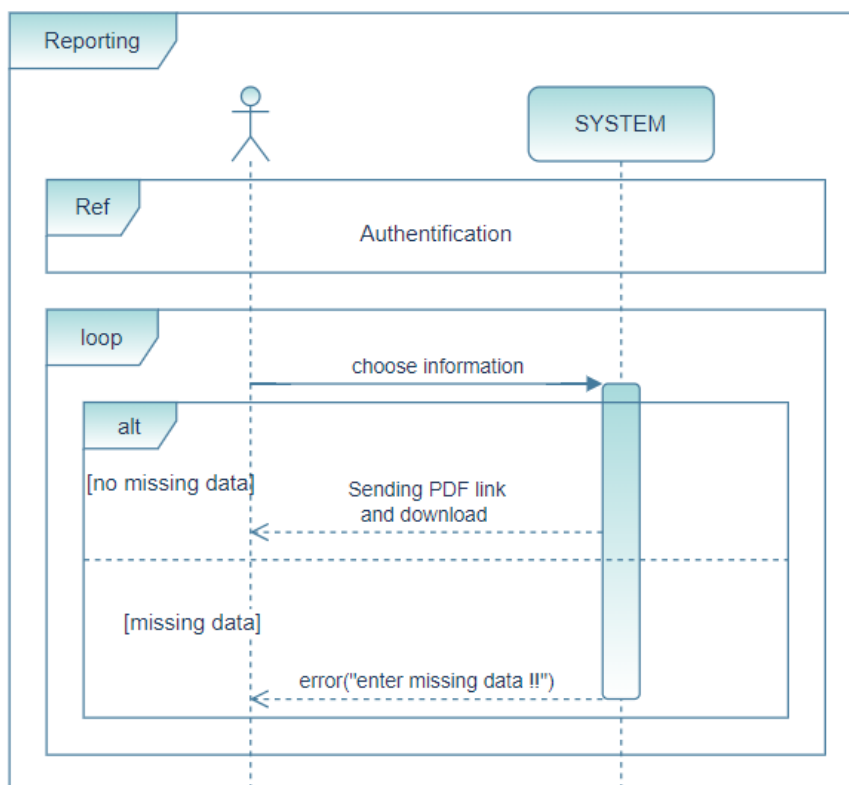


Figure 47: "Reporting" System Sequence Diagram

CONCEPTION

1. Class Diagram

1.1. Manage Partners

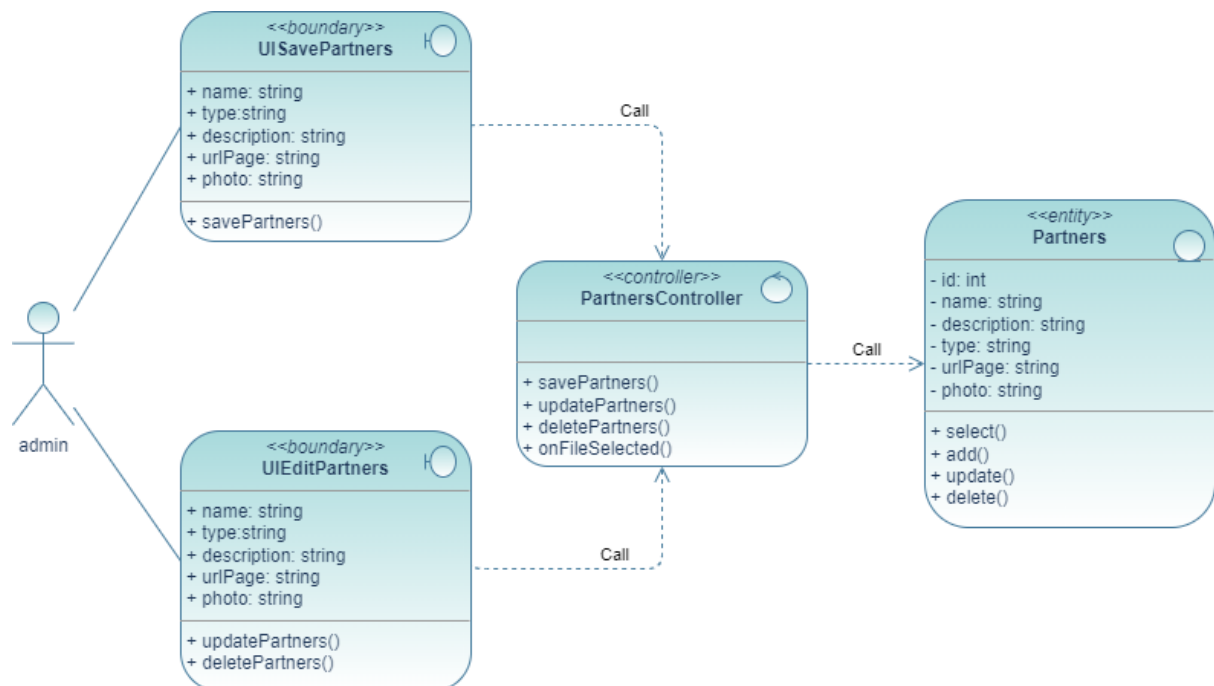


Figure 48: "Manage Partners" Class Diagram

1.2. Manage Projects

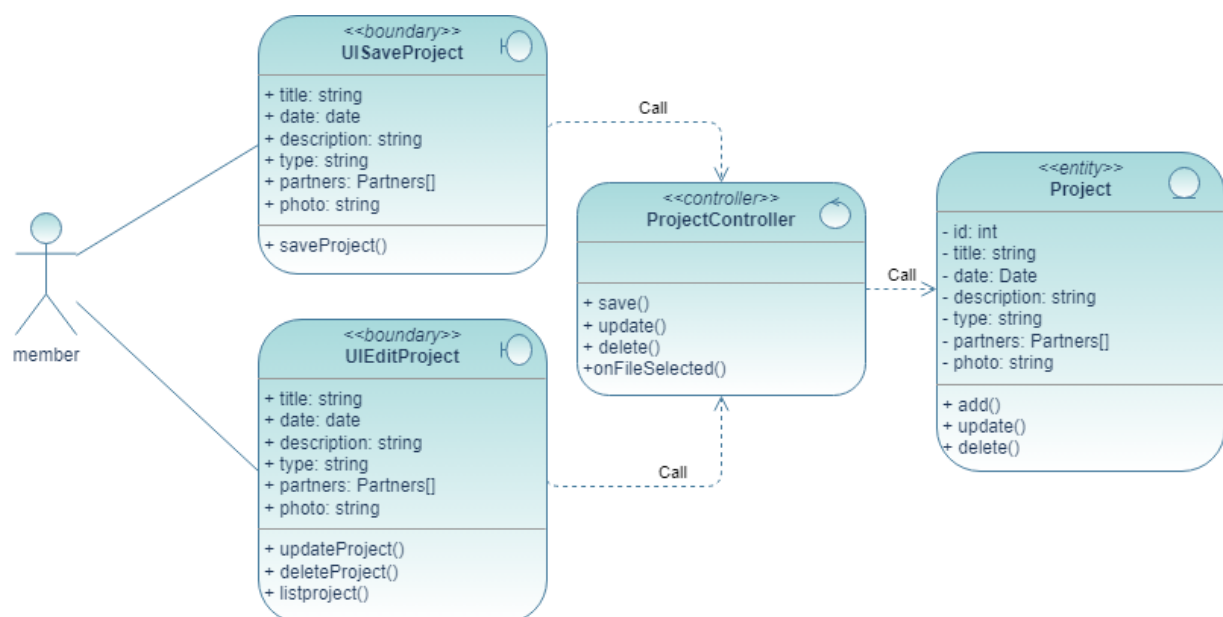


Figure 49: "Manage Project" Class Diagram

1.3. Manage Members

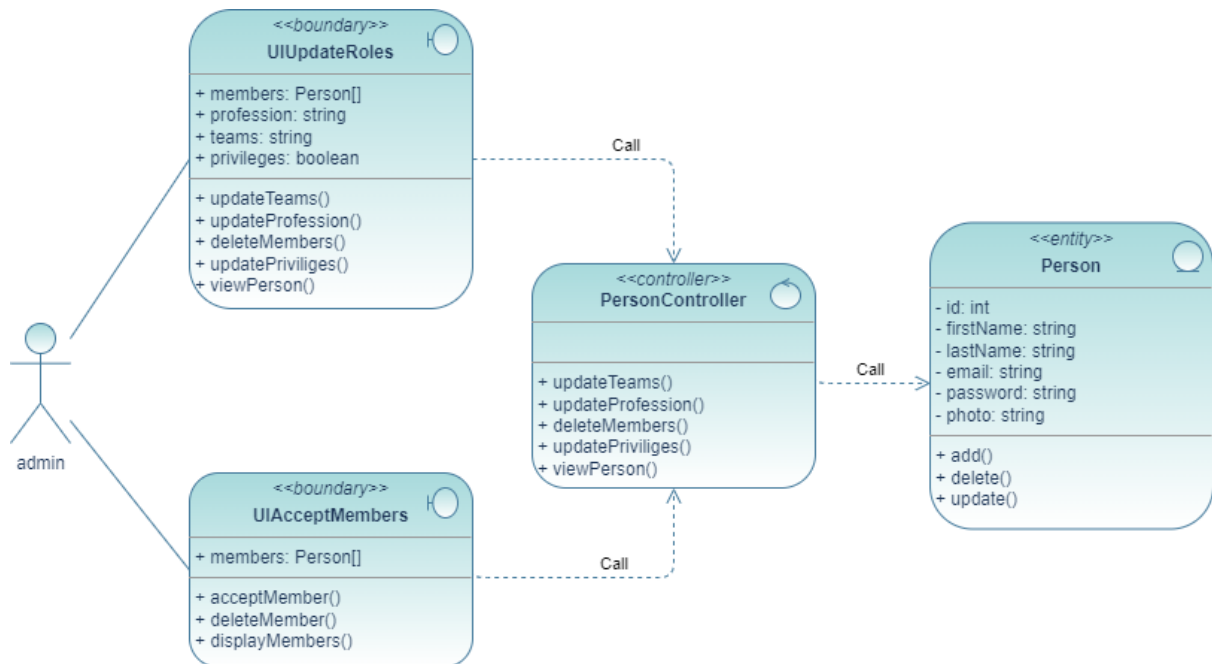


Figure 50: "Manage Members" Class Diagram

1.4. Reporting

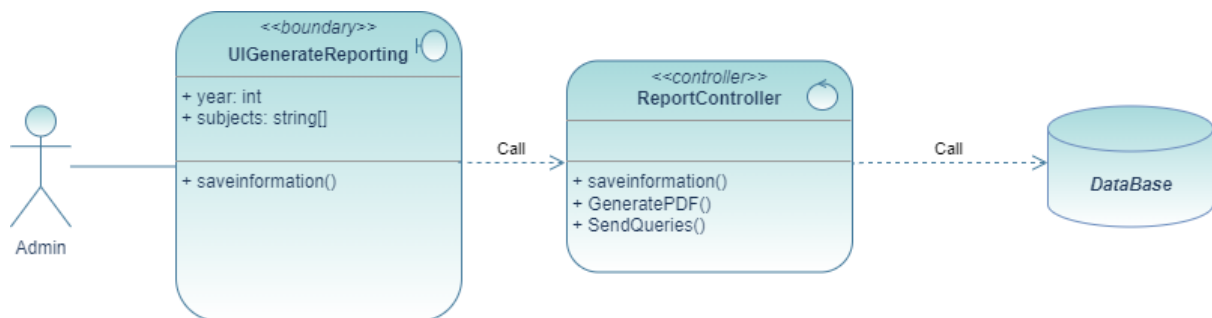


Figure 51: "Reporting" Class Diagram

2. Detailed Sequence Diagram

2.1. Add Partner

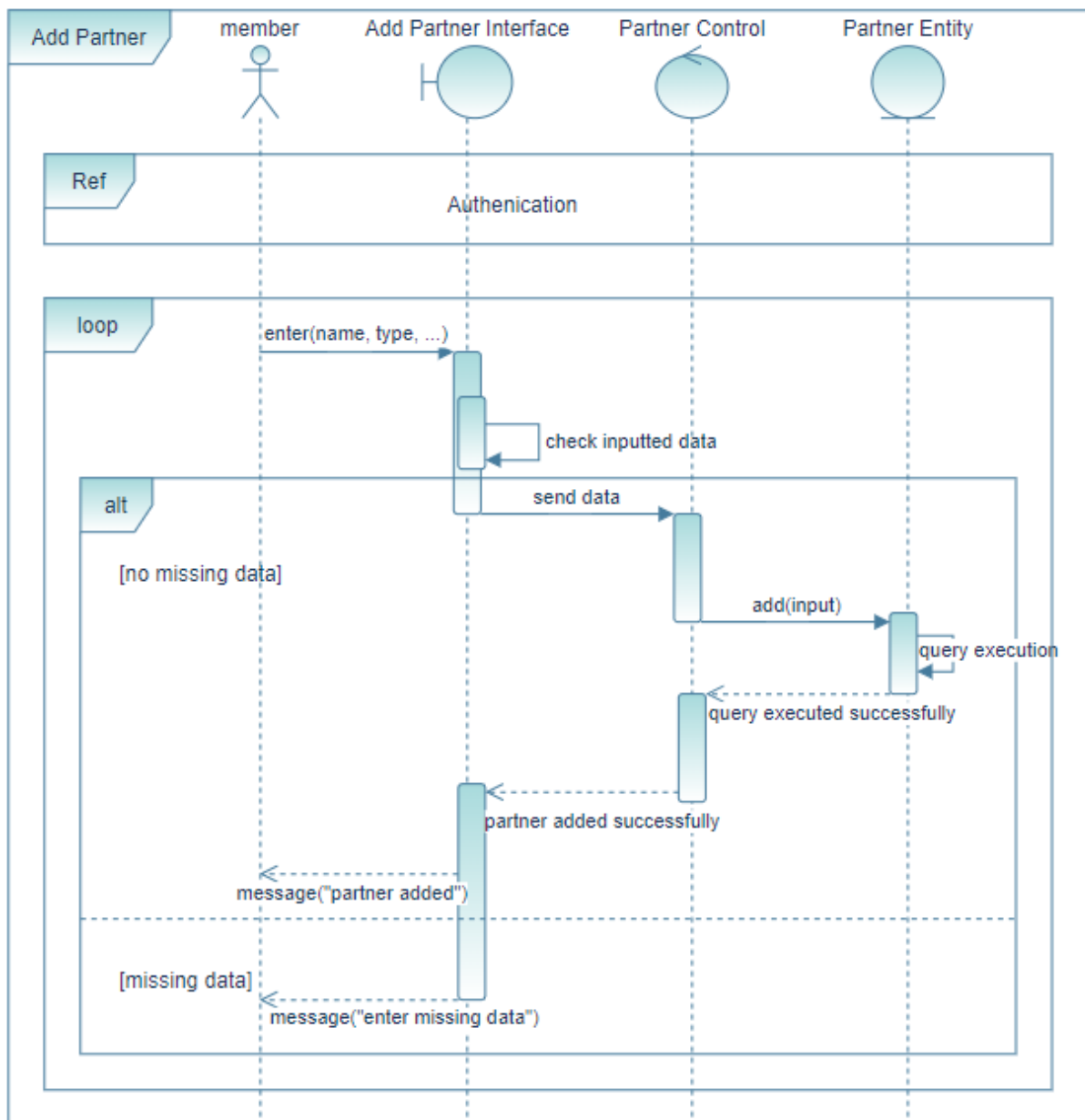


Figure 52: "Add Partner" Detailed Sequence Diagram

2.2. Delete Partner

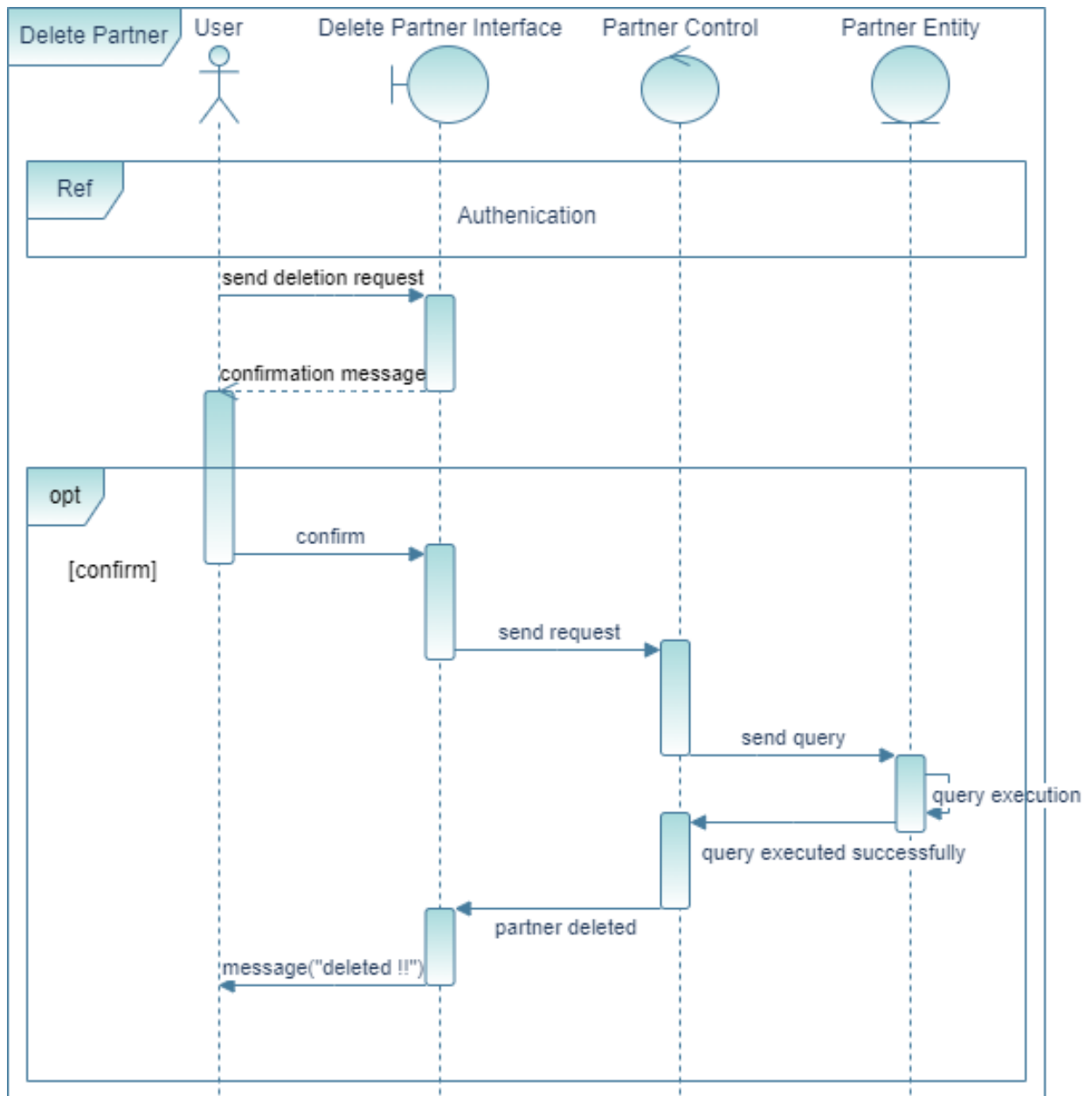


Figure 53: "Delete Partner" Detailed Sequence Diagram

2.3. Update Project

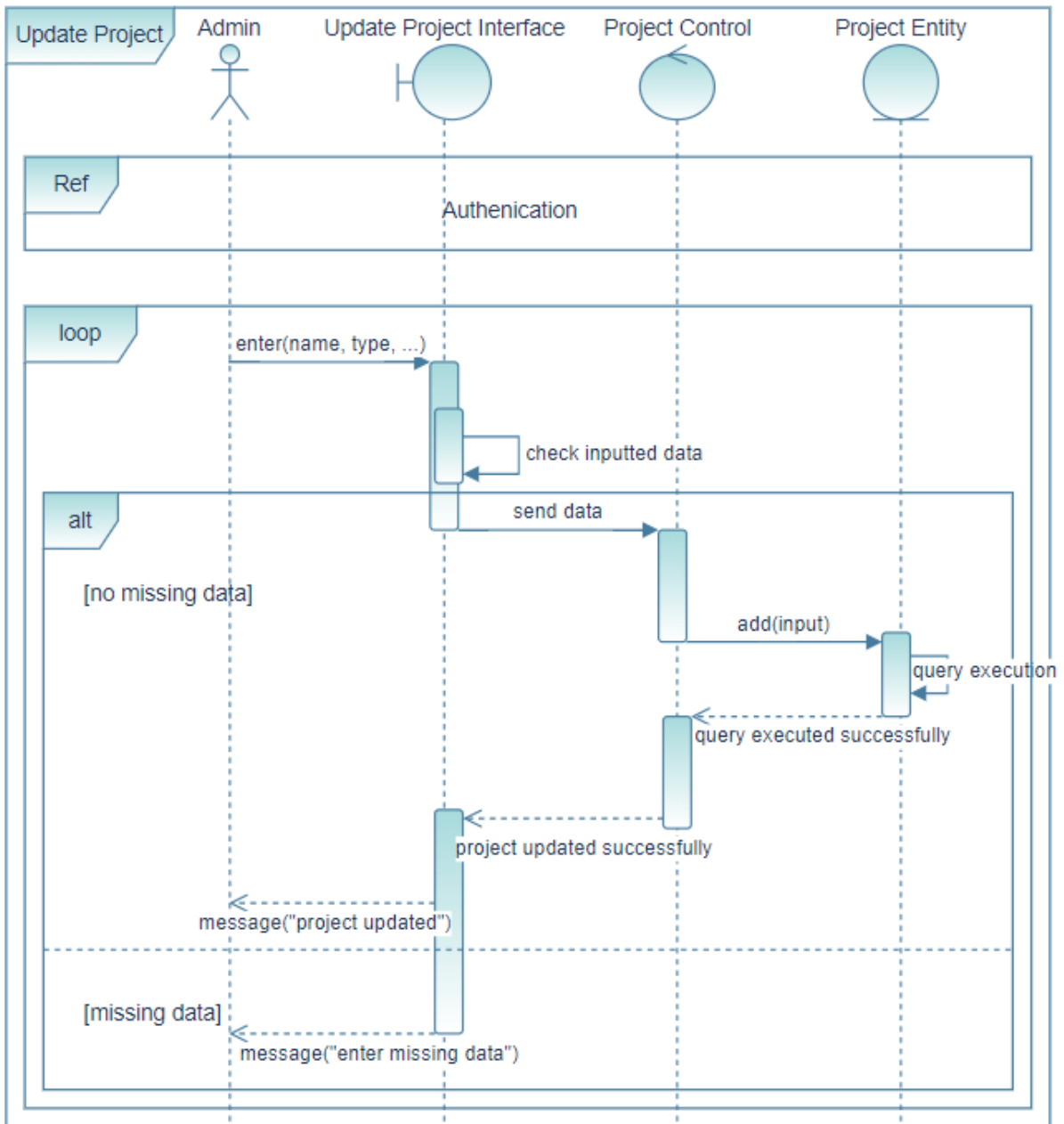


Figure 54: "Update Project" Detailed Sequence Diagram

2.4. Accept Membership Request

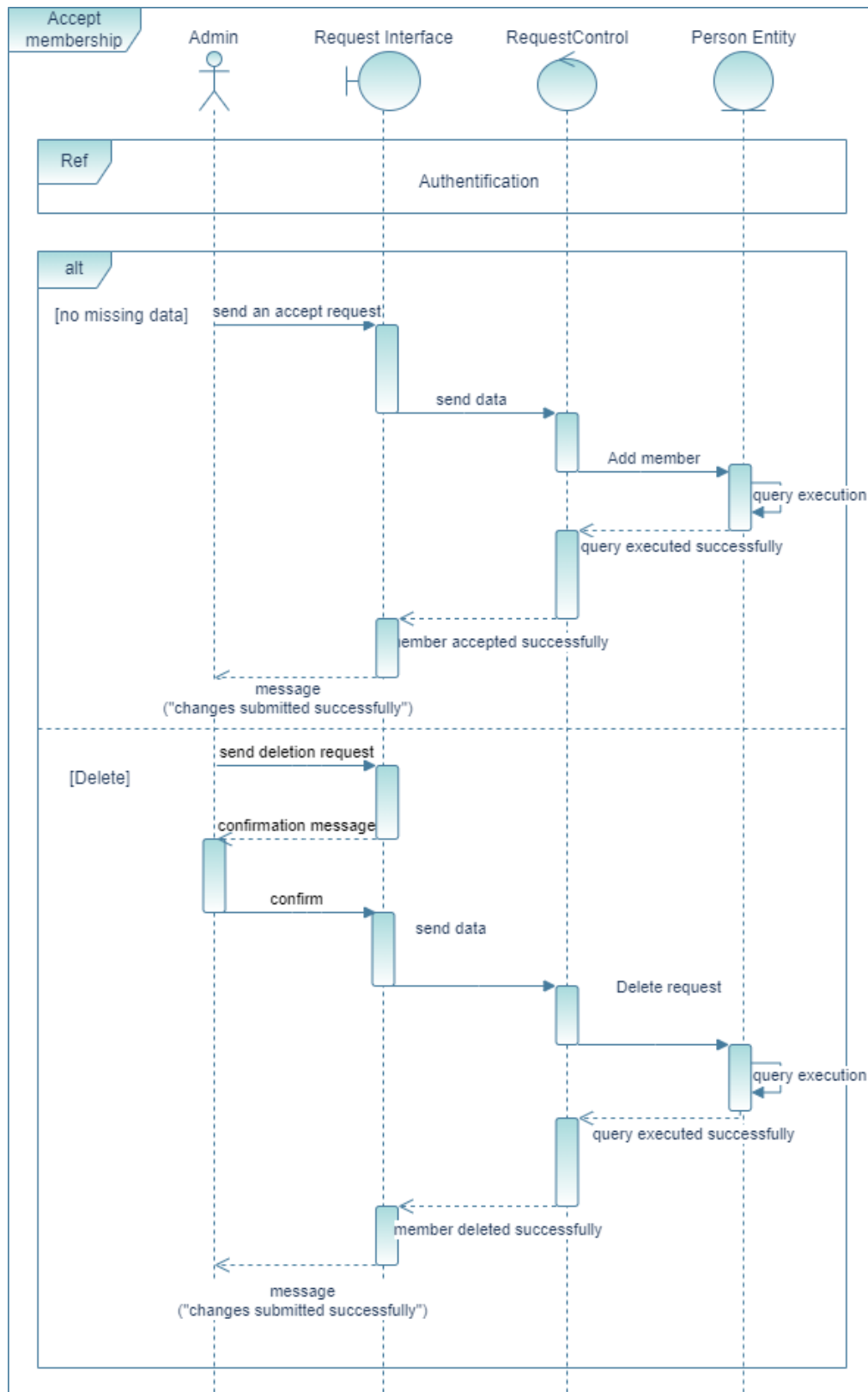


Figure 55: "Accept Membership" Detailed Sequence Diagram

2.5. Update Team

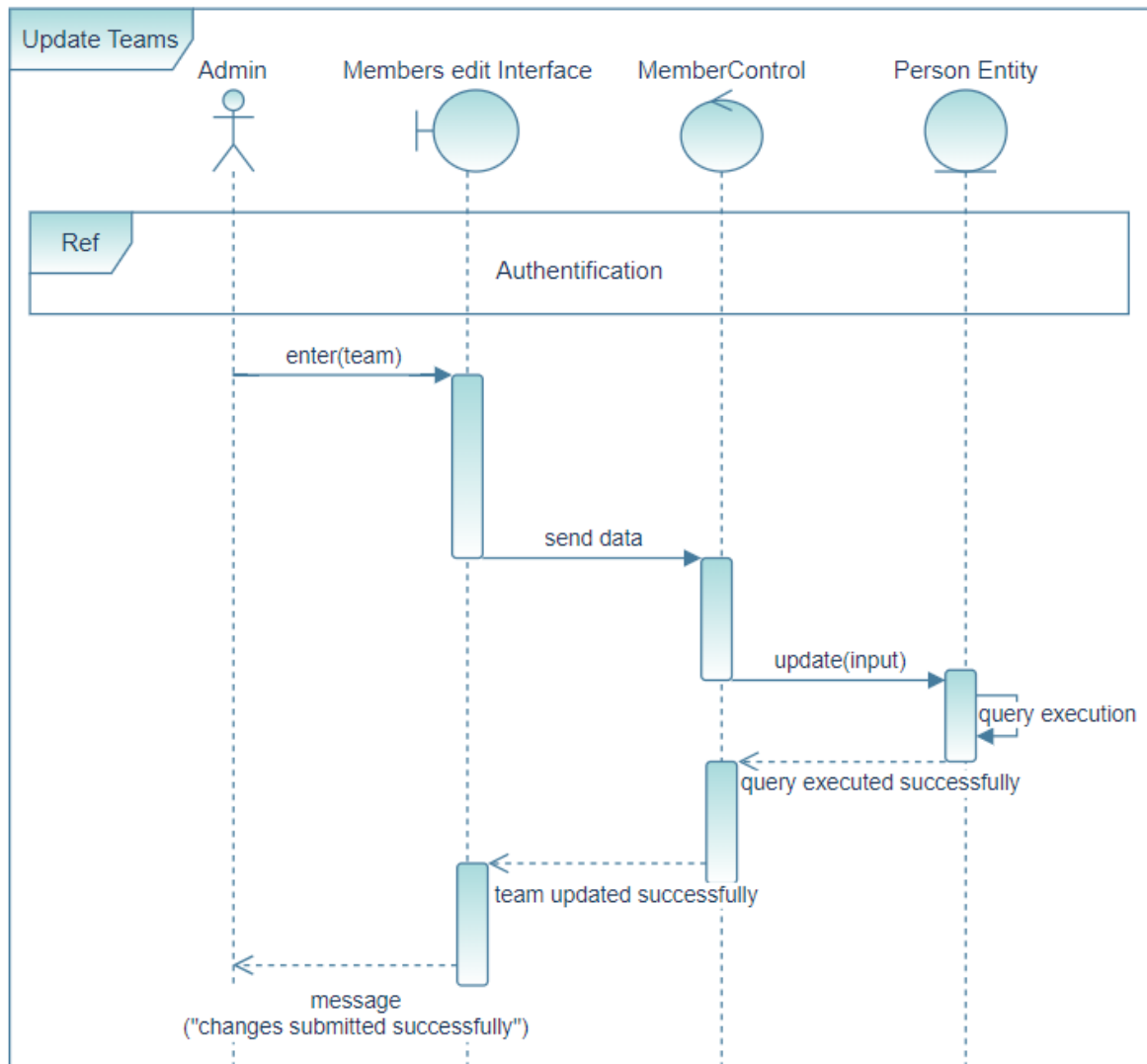


Figure 56: "Update Teams" Detailed Sequence Diagram

2.6. Delete Member

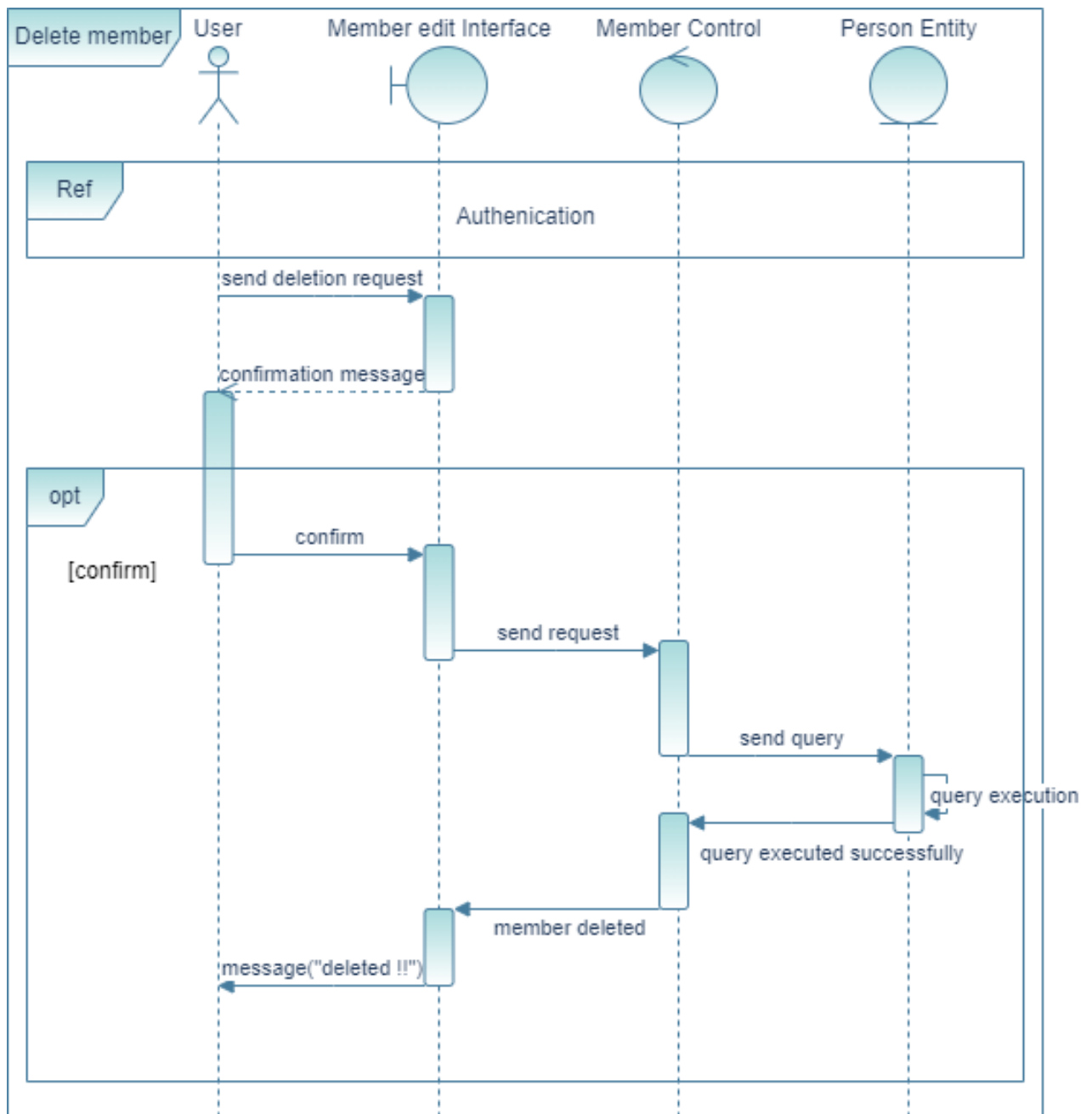


Figure 57: "Delete Member" Detailed Sequence Diagram

2.7. Reporting

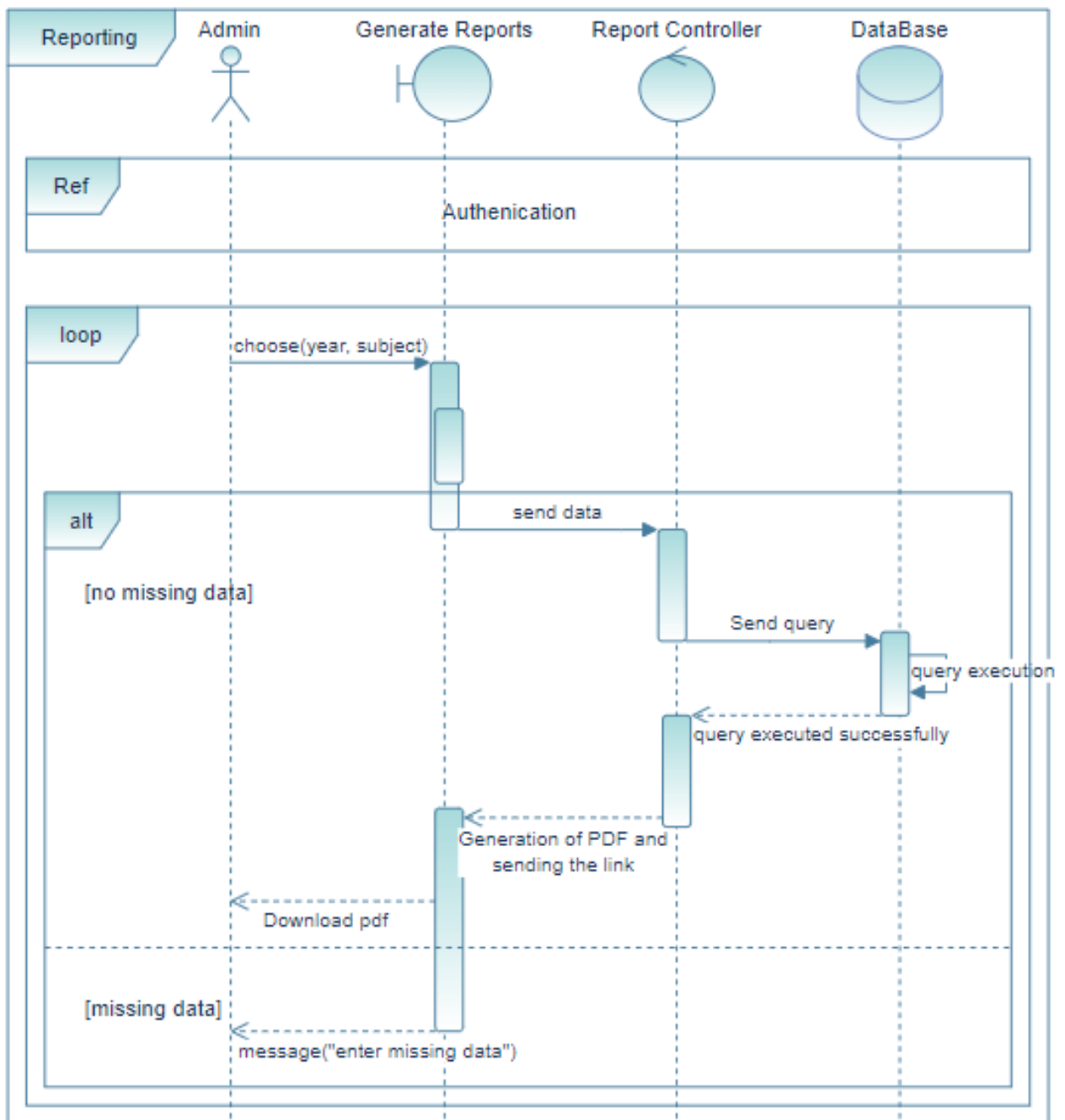


Figure 58: "Reporting" Detailed Sequence Diagram

IMPLEMENTATION

1. Relational Schema

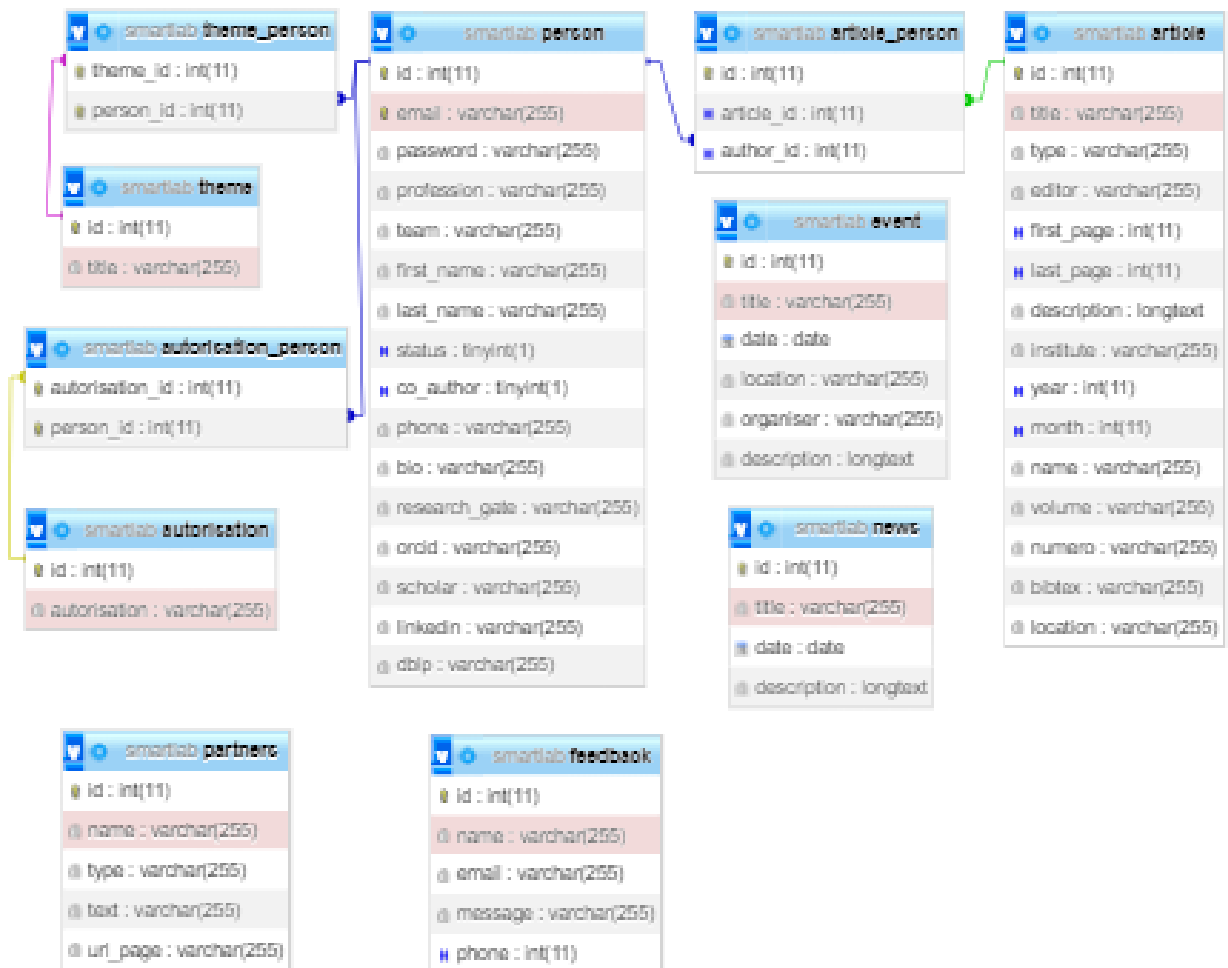
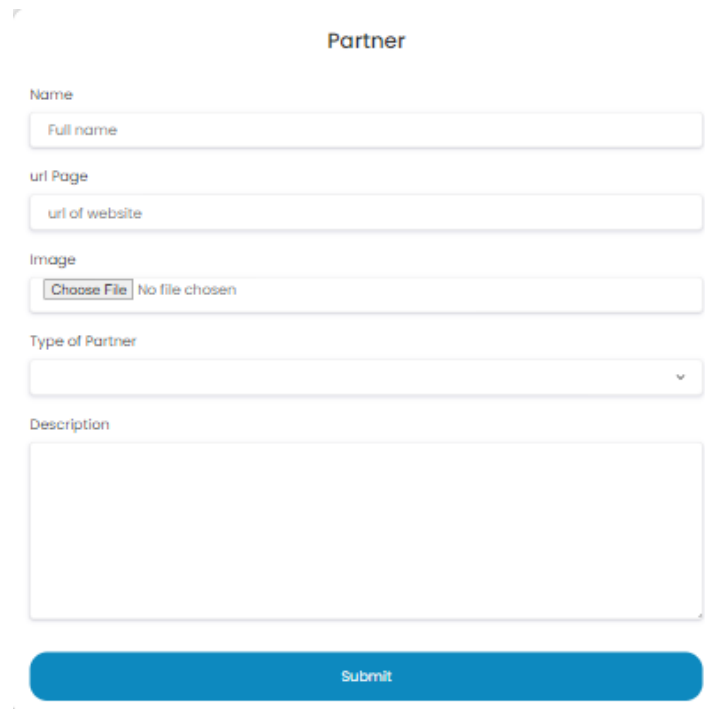


Figure 59: Sprint 3 Relational Schema

2. Interfaces

2.1. Add Partner



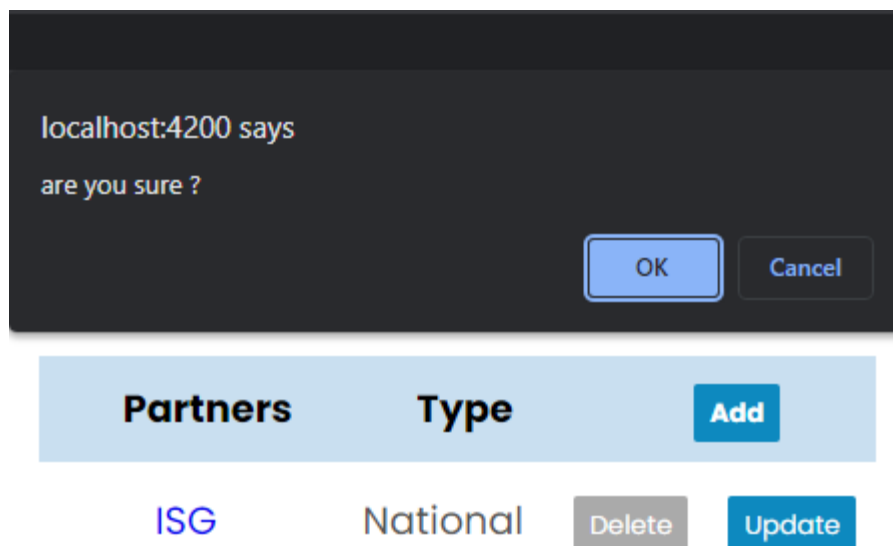
The "Add Partner" form is titled "Partner" and contains the following fields:

- Name:** A text input field with the placeholder "Full name".
- url Page:** A text input field with the placeholder "url of website".
- Image:** A file selection interface with a "Choose File" button and the text "No file chosen".
- Type of Partner:** A dropdown menu.
- Description:** A large text area.

A blue "Submit" button is located at the bottom of the form.

Figure 60: "Add Partner" Interface

2.2. Delete Partner



The "Delete Partner" interface consists of a confirmation dialog and a table of partners.

Confirmation Dialog: A dark gray box with the text "localhost:4200 says are you sure ?" and two buttons: "OK" (blue) and "Cancel" (gray).

Partners Table: A table with the following structure:

Partners	Type	
ISG	National	<div>Delete Update</div>

Figure 61: "Delete Partner" Interface

2.3. Update Project

Title	Description	Date	Type	Add Project
Coding	Project about coding with students	21-01-2020	National	<button>Update</button> <button>Delete</button>

Figure 62: "Edit Project" Interface

Update Project

Title

type

location

Date

Image

Aucun fichier choisi

Description

Project about coding with students

Update

Figure 63: "Update Project" Interface

2.4. Accept Membership Request

Name	EMAIL	DECISION
roudeina hamdi	rhamdi@gmail.com	<button>ACCEPT</button> <button>Delete</button>

Figure 64: "Accept Membership Request" Interface

2.5. Update Profession/ Update Teams/ Update Privileges/ Delete Member

Member	Delete	Profession	Teams	News	Events	Members	Feedback	Requests	Partners
roudeina hamdi	<button>DELETE</button>	Researcher	HR	✗	✗	✓	✓	✗	✗
Jihen Aridhi	<button>DELETE</button>	Ph.D Student	Development	✓	✓	✓	✓	✓	✓
<button>SUBMIT CHANGES</button>									

Figure 65: "Update Profession"/ "Update Teams"/ "Update Privileges"/ "Delete Member" Interface

2.6. Reporting

REPORTING FORM

Year

☐ Articles

☒ Members

☐ Projects

☒ Events

☐ News

☐ Partners

Generate PDF

Download PDF

Figure 66: "Reporting" Interface

Conclusion

This chapter presents a detailed analysis of the third sprint, that is consisted of tasks assigned to the administrator actor, as well as the conception, and finally an implementation.

Architecture & Development Environment

Introduction

In this chapter, we are diving into the final phase of developing the application. This part covers the highlights of the adopted architecture MVC, as well as the developing environment which is consisted of the employed tools used in order to develop.

1. Project Architecture

1.1. Logical Architecture: MVC Architectural Pattern

The MVC (Model-View-Controller) architectural pattern is a design pattern commonly used in software development to separate the concerns of an application into three main components: Model, View, and Controller. [4]

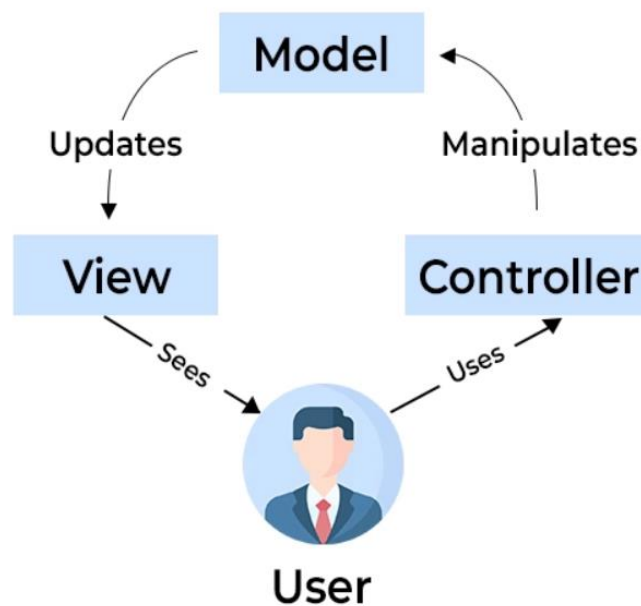


Figure 67: MVC Architecture

Table 25: MVC architectural pattern

Component	Responsibility	Interaction
Model	<p>Models represent the data and business logic of the application.</p> <p>Model components are responsible for data manipulation, validation, and interaction with databases or other data sources.</p>	<p>The Model component is independent of the other components.</p> <p>It is accessed and manipulated by the Controller to retrieve or update data.</p> <p>It notifies the View of any changes to the data so that the UI can be updated accordingly.</p> <p>It may utilize services or repositories to interact with databases or external data sources.</p>
View	<p>The View is responsible for the presentation layer of the application. It represents the user interface (UI) through which users interact with the application.</p> <p>It focuses on the visual aspects and user experience, ensuring that the data is presented in an appropriate and meaningful way.</p>	<p>The View receives data from the Model to display it to the users.</p> <p>It does not directly interact with the Model but retrieves data through the Controller.</p> <p>It notifies the Controller of user actions or input, such as button clicks or form submissions.</p> <p>It may subscribe to events or observables to stay updated with changes in the Model.</p>

Controller	<p>The Controller acts as the intermediary between the Model and the View.</p> <p>The Controller interacts with the Model to retrieve and manipulate data and prepares it to be displayed in the View.</p> <p>It coordinates the flow of data between the Model and the View, ensuring that they remain decoupled and independent of each other.</p>	<p>The Controller receives user input from the View and initiates actions based on that input.</p> <p>It communicates with the Model to fetch or modify data as required.</p> <p>It updates the Model with new data and notifies the View of any changes.</p> <p>It may use services, repositories, or other helper classes to perform data operations.</p>
-------------------	--	---

1.2. MVC Benefits

We opted for the MVC architectural pattern due to its numerous advantages.

MVC facilitates a clear division of responsibilities by partitioning the application into three distinct elements. This separation empowers us to concentrate on specific aspects of the application without impeding other areas. It enhances the organization and maintainability of the code since modifications in one component are less likely to impact others.

The modular structure of MVC fosters code reusability. Models, views, and controllers can be developed independently and reused across various sections of the application or even in separate projects. This diminishes development time and effort while enhancing overall productivity.

MVC encourages collaboration among team members working on different components of the application. With well-defined responsibilities and interfaces for each component, we can concurrently work on various parts of the system. This parallel development approach amplifies productivity and minimizes conflicts during code integration.

Lastly, MVC simplifies modifications and extensions to the application as requirements evolve. The component separation permits changes to be made to one part without affecting the others, thereby reducing the likelihood of unintended side effects. This adaptability to change is crucial for the long-term maintenance and evolution of the application. [5]

1.3. Implementation of the MVC pattern in our project:

In the backend, we have implemented entities which represent the Model component in the MVC pattern, and our controllers are the controller component that will receive incoming requests and handle the application's business logic. We organized our controllers to follow the MVC principles, so we implemented separate controllers for different entities.

In the frontend, we have implemented components which contain html, CSS, typescript files, representing the View component in the MVC pattern. We divided our application into individual components, each responsible for a specific part of the user interface, those components should focus on UI-related logic and should not contain business logic or data manipulation code, they receive data from models and bind it to the template to display to the users. services in Angular can act as the Controller component in the MVC pattern, we implemented them to encapsulate the application's business logic and data manipulation operations. Services interact with the backend APIs to retrieve or update data, in order to process it and pass it to the angular components.

1.4. Deployment Diagram

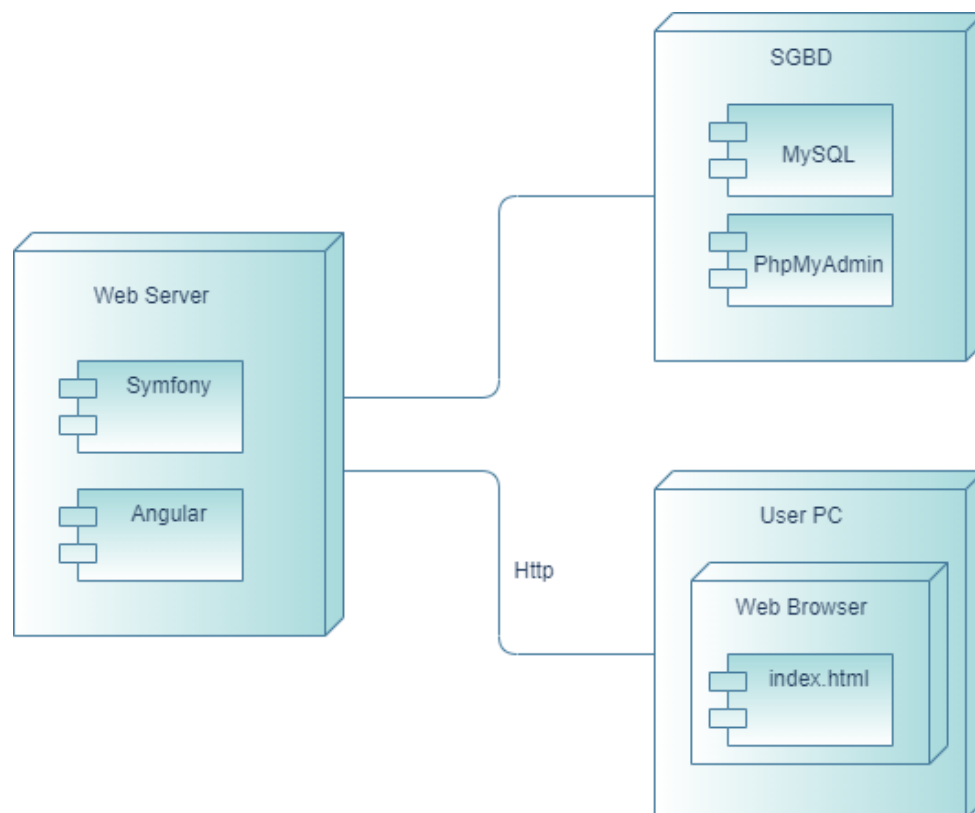


Figure 68: Deployment Diagram

2. Software Development Environment

2.1. Diagrams.net/Draw.io

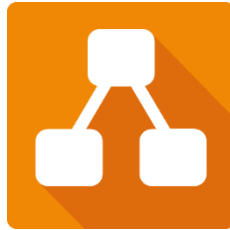


Figure 69: Diagrams.net/Draw.io Logo

Draw.io is an open-source technology stack for building diagramming applications, and the world's most widely used browser-based end-user diagramming software. [6]

2.2. XAMPP



Figure 70: XAMPP Logo

XAMPP is an open-source package of web solutions that includes Apache distribution for many servers and command-line executables along with modules such as Apache server, MariaDB, PHP, and Perl. Among those tools, *phpMyAdmin* is used. [7]

2.3. MySQL



Figure 71: MySQL Logo

MySQL, the most popular Open-Source SQL database management system, is developed, distributed, and supported by Oracle Corporation.[8]

2.4. IntelliJ IDEA

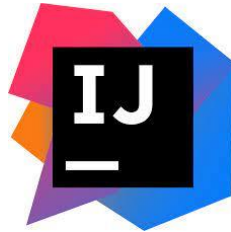


Figure 72: IntelliJ IDEA Logo

IntelliJ IDEA is a cross-platform integrated development environment (IDE) written in Java for developing computer software written in Java, Kotlin, Groovy, and other JVM-based languages. Throughout this project, it has been used to develop the front-end. [9]

2.5. PhpStorm



Figure 73: PhpStorm Logo

PhpStorm is an innovative, Java-based integrated development environment (IDE) engineered by JetBrains for PHP and web developers. Throughout this project, it has been used to develop the back-end. [10]

2.6. Angular



Figure 74: Angular Logo

Angular is an application-design framework and development platform for creating efficient and sophisticated single-page apps. Using HTML, CSS, TypeScript/JavaScript. [11]

2.7. Symfony



Figure 75: Symfony Logo

Symfony is an open-source PHP web application framework which follows an MVC architectural pattern. It provides a set of reusable components, libraries, and tools to streamline the development process. [12]

2.8. GitHub



Figure 76: GitHub Logo

GitHub is a web-based version control and collaboration platform for software developers. It has been used throughout this project in order to exchange updates between us. [13]

Conclusion

In this Chapter, we explore the realisation phase, which consists of presenting the adopted architecture, and various tools that are used in the process of developing the application.

General Conclusion

The focus of this work has been the design and development of a web application for the research laboratory SMARTLAB. In This case, the report describes the sequence of actions taken in order to achieve the assigned goal, in which we have been able to plan the architecture of this project, according to “SCRUM” method, dividing it into three sprints.

Throughout the remainder of the report, we explore each sprint by taking a deeper look into analysis, conception and implementation. The first sprint, is consisted of developing the whole website interface, as well as allowing the netizens to registrate into the laboratory. The second sprint, is consisted of the ensemble of tasks that can be assigned to a regular member of the laboratory, such as managing their account, managing articles, news and events. The third sprint, is consisted of the ensemble of tasks that are assigned to an administrator, such as managing partners, managing projects, managing members and reporting. The last chapter of this report is dedicated to exploring the adopted architecture MVC, along with the various tools that were employed in order to realise the project.

Since this project incorporates a reporting functionality that enables the display of comprehensive statistics for the year, we have the potential to develop a dashboard that would facilitate the tracking and monitoring of these statistics. Such a dashboard would greatly assist in decision-making processes by providing valuable insights and analysis.

WEBOGRAPHY

[1] Agile Methodology

<https://digital.ai/glossary/characteristics-of-agile-development-success/>

<https://www.indeed.com/career-advice/career-development/agile-characteristics>

[2] Comparative study: Agile method / Classic method

<https://www.forbes.com/advisor/business/agile-vs-waterfall-methodology/>

<https://www.guru99.com/waterfall-vs-agile.html>

[3] Scrum, Advantages & Disadvantages

<https://www.atlassian.com/agile/scrum>

<https://www.simplilearn.com/scrum-project-management-article?fbclid=IwAR10xD9iSVSCVvwaBHK-JAJuuroZ-AvH76O6JnAZESW-yPtx24mt3C2Hgjo>

[4] MVC Architectural Pattern

<https://towardsdatascience.com/everything-you-need-to-know-about-mvc-architecture-3c827930b4c1>

<https://www.geeksforgeeks.org/mvc-design-pattern/>

[5] MVC Benefits

<https://profoundedutech.com/blog/6-benefits-of-using-mvc-model-for-effective-web-application-development/>

<https://www.geeksforgeeks.org/benefit-of-using-mvc/>

[6] Diagrams.net/Draw.io

<https://www.drawio.com/about>

[7] XAMPP

<https://www.javatpoint.com/xampp>

[8] MySQL

<https://dev.mysql.com/doc/refman/8.0/en/what-is-mysql.html>

[9] IntelliJ IDEA

https://en.wikipedia.org/wiki/IntelliJ_IDEA

[10] PhpStorm

<https://monovm.com/blog/what-is-phpstorm/#What-is-PhpStorm?>

[11] Angular

<https://angular.io/docs>

[12] Symfony

<https://symfony.com/doc/current/index.html>

[13] GitHub

<https://www.techtarget.com/searchitoperations/definition/GitHub>

Conception & Development of SMARTLAB Web Application

Roudeina Hamdi

Jihen Aridhi

ABSTRACT

The present project consists on realizing a Web Application for SMARTLAB.

The developed application helps SMARTLAB's members to publish their articles, to post their laboratory's news, events and projects.

Our platform was developed with new web technologies, citing PHP, TypeScript, HTML5, CSS3... by relying on architecture suitable to the MVC architectural pattern.

Furthermore, this project was developed through SCRUM.

RESUME

Le projet actuel consiste à réaliser une application Web pour SMARTLAB.

L'application développée aide les membres de SMARTLAB à publier leurs articles, à poster les actualités, les événements et les projets de leur laboratoire.

Notre plateforme a été développée avec de nouvelles technologies web, notamment PHP, TypeScript, HTML5, CSS3, en s'appuyant sur une architecture adaptée au modèle architectural MVC.

De plus, ce projet a été développé selon la méthode SCRUM.