Course Management Support Application-iMaster.Schedule

Youssef Bassoumi - a42705

Project work presented to the Escola Superior de Tecnologia e Gestão de Bragança to obtain the Master's Degree in Information Systems under the dual diploma program with the Université Libre de Tunis.

Supervisors:

Professor José Eduardo Fernandes

Professor Paulo Alexandre Vara Alves

Professor Marwa Massaabi

Bragança

November 2020

Course Management Support Application-iMaster.Schedule

Youssef Bassoumi - a42705

Project work presented to the Escola Superior de Tecnologia e Gestão de Bragança to obtain the Master's Degree in Information Systems under the dual diploma program with the Université Libre de Tunis.

Supervisors:

Professor José Eduardo Fernandes

Professor Paulo Alexandre Vara Alves

Professor Marwa Massaabi

Bragança

November 2020

Acknowledgment

I would like to thank everyone who contributed to the success of my thesis and who helped me when writing this report. I would like to warmly thank my supervisor, Professor José Eduardo Fernandes, for having guided me during this project with his valuable advice and his sharing of his expertise. I express my sincere gratitude and my deep appreciation to my supervisor, Professor Paulo Alexandre Vara Alves, for his remarks, his precious advice, his kindness, his availability and above all for his confidence. A big thank you to my family and friends who supported me.

Abstract

Nowadays, technology plays an important role in people's life. It has shown us that it could considerably facilitate our lives and help us with our tasks and interactions. In the academic sector, some individuals will have a greater charge of creating and organizing social interactions, this is the case of the director of a master course. Indeed his position requires constantly creating and managing gatherings such as meetings, jury meetings and events. That is the reason of creating this platform. The main goal was to simplify intercommunication. Through this solution, master director can now easily access and manage their schedules and appointments according to their calendars, generate post-meeting documents directly from the platform, thus avoiding waste of time, energy and the risk of having the information incorrectly transmitted. The solution also allows the director to monitor at any time the people involved in his master course, such as students and professors and to consult their specific informations such as their curricular units, their level and even their thesis subject titles. This solution aims at being an all-purpose tool to organize, simplify and expand the Master director's management process.

Resumo

Atualmente, a tecnologia desempenha um papel importante na vida das pessoas e pode facilitar consideravelmente nossas vidas e ajudar-nos nas nossas tarefas e interações. No setor académico, alguns indivíduos terão uma responsabilidade maior de criar e organizar interações, como é o caso de um diretor de curso. Na verdade, a sua posição exige a criação e a gestão reuniões, júris e eventos. É por isso que criamos esta plataforma. O principal objetivo é simplificar a intercomunicação. Com esta solução, os diretores podem aceder e gerir as suas agendas e compromissos de acordo com seus calendários, gerar documentos pós-reunião diretamente da plataforma, evitando assim perda de tempo e energia e o risco de transmissão incorreta de informações. A solução também permite ao utilizador acompanhar a qualquer momento as pessoas envolvidas no seu curso de mestrado, como alunos e professores, podendo consultar informação específica como suas unidades curriculares e título de tese. Esta solução pretende ser uma ferramenta multifacetada para organizar, simplificar e expandir o processo de gestão de um diretor de mestrado.

Contents

1 Introduction **1**

2 State of the Art **3**

2.1 Introduction 3

2.2 Study of the existing process 3

2.3 Proposed solution 5

3 Requirement Analysis and Specification **7**

3.1 Functional needs 7

3.2 Non-functional needs 9

3.3 Use Case Diagram 9

3.3.1 Global Use case Diagram 9

3.3.2 Detailed Use Case Diagrams 10

3.3.2.1 Use case Diagram of the Management of Meeting 11

3.3.2.2 Use case Diagram of the Event 13

3.4 Class Diagram 15

3.5 Detailed Conception 16

3.5.1 Sequence Diagram «Authentication» 16

3.5.2 Sequence Diagram «Manage Meeting» 17

3.5.3 Sequence Diagram «Send E-mail» 18

3.5.4 Sequence Diagram «Generate PV» 19

3.6 The web site map 20

3.7 Database Schema 21

4 Technologies and Development Tools 23

4.1 Front-end Framework 23

4.1.1 Angular 23

4.1.2 Vue 24

4.1.3 React 24

4.2 Back-end Framework 25

4.2.1 Laravel 25

4.2.2 Django 26

4.2.3 Asp.Net 26

4.3 Technologies Choices 27

4.3.1 Front-end choice 27

4.3.2 Back-end choice 28

4.3.3 Other Technologies 28

4.3.3.1 Microservices 28

4.3.3.2 SQL Server Management Studio 30

4.3.3.3 ASP.Net Core Identity 30

4.3.3.4 Entity Framework 30

4.3.3.5 StarUml 31

4.4 General Structure 31

4.5 Architecture Used 32

5 Development of iMaster.Schedule web application 34

5.1 Introduction 34

5.2 Interfaces of the Front-end 34

5.2.1 Interface of Authentication 35

5.2.2 Interface of schedule 36

5.2.3 Interface of the weekly schedule 37

5.2.4 Interface of list of all meeting 38

5.2.5 Interface of add meeting 39
5.2.6 Interface of the post meeting "PV" 40
5.2.7 Interface of Add Event 41
5.2.8 Interface of the E-mail 42
5.2.9 Interface of Add Student 43
5.2.10 Interface of choosing curricular Units 44
5.2.11 Interface of The List of all student 45
5.2.12 Interface of Student Information 46

6 Conclusion **49**

List of Figures

2.1 Screenshot of IPB.virtual Calendar 4
2.2 Screenshot of Google Calendar 5

3.1 Global Use case Diagram 10
3.2 Use case Diagram of the Management of Meeting 11
3.3 Use case Diagram of the Event 13
3.4 Class Diagram 15
3.5 Sequence Diagram «Authentication» 16
3.6 Sequence Diagram «Manage Meeting» 17
3.7 Sequence Diagram «Send E-mail» 18
3.8 Sequence Diagram «Generate PV» 19
3.9 The web site map 20
3.10 Data Base Schema 21

4.1 Mounting and re-rendering components in React. 25
4.2 Micro Services Architecture 29
4.3 Architecture of Entity Framework 31
4.4 MVC Model 32

5.1 Interface of Authentication 35
5.2 Interface of the schedule 36
5.3 Interface of the weekly schedule 37
5.4 Interface of list of all meeting 38

5.5 Interface of add meeting 39
5.6 Interface of the post meeting "PV" 40
5.7 Interface of Add Event 41
5.8 Interface of the E-mail 42
5.9 Interface of Add Student 43
5.10 Interface of choosing curricular Units 44
5.11 Interface of the List of all student 45
5.12 Interface of Student Information 46

List of Tables

3.1 Use Case «Management of Meeting» details 12
3.2 Use Case «Management Event» details 14

Chapter 1

Introduction

Polytechnic Institute of Braganca is a Portuguese university located in Bragança and Mirandela, founded in 1983. Currently it is made up of five schools which proves the diversity of the educational paths that IPB offers for around 9000 students. Most of these are international students from different parts of the world through mobility programs, such as Erasmus partnerships. The Polytechnic Institute of Braganca offers the opportunity to study various domains in several study levels of Undergraduate Courses, Higher Professional Technical Courses, Postgraduate courses, and Masters. Having this many students and diverse courses, the university is constantly in need of better and up to date ways of organising its functioning.

The university has several masters which explains the high number of master's students and is difficult to manage the processes without a digital platform. It is within this context that the solution "iMaster.Schedule" operates, because now in IPB a master director who wants to meet with students and professors over a certain topic either has to use Google Calendars or a third party, adding the participants mail address and any relevant documents manually. Furthermore he has to create manually the document containing the information of the meeting. That is the problematic addressed by this project, which is why the aim of this project is to develop a solution that will allow master directors to go beyond these problems by combining many functionalities by using "iMASTER.Schedule".

The software introduced in this work aims to simplify the establishment's master department's calendars, events, meetings, as well as facilitate communication through its constituents. Delays in responses and scheduling meetings will be cut down, by transmitting site-wide updates to students and lectures. One of the purposes of the platform is to generate a linear path of transmitting data, reports.

"iMaster.Schedule" is a web application that contains several functions that could simplify interactions inside academies and schools. The primary users of this solution will be the Master directors who will be able to access and manage their calendars, schedule appointments and notify participants via e-mail, post and update the relevant persons on news and changes in events directly from the platform as well as generate a post-meeting document containing the minutes, main topics and conclusions of meetings. In order to develop an application that meets the expectations and requirements of future users, it is necessary to organise the work as follows:

The first chapter will introduce and describe the main idea of this project and present some information about the university.

The second chapter is the state of the art and will analyse the existing solutions such as Google Calendar and assess the lacking features before delving into the proposed solution.

The third chapter is the requirement analysis and specification,it will specify the needs of the project such as functional and non-functional needs. This chapter also identifies the actors and their roles through a detailed conception, use case and class diagrams. In the same way this chapter will show us the web site map and the database schema.

The fourth chapter will evoke the different technologies and structure of the solution from the general structure and the architecture used.

The fifth chapter will go through the steps of the implementation of the solution and will explore the detailed functionalities of the solution.

Finally, we close this report with a general conclusion allowing to synthesize all the work carried out while evoking the perspectives of the project.

Chapter 2

State of the Art

2.1 Introduction

In order to successfully develop an optimal solution, that one need to study the pre-existing ones. Assess their positive and negative aspects with the user's best interest in mind and try to incorporate the useful positive aspects within the product while cutting on any flawed or useless parts and this analysis allows us to propose a solution.

2.2 Study of the existing process

During a study of the existing, ones identified two main existing solutions, first the platform of the Polytechnic Institute of Bragança. It was observed in the figure 2.1 below that the "IPB.virtual Calendar" [1] web section, intended for the student, allows users to visit the calendars and learn detailed information on times and dates. They therefore constitute customized, simple spaces. However this solution has some drawbacks, namely:

- No appointment booking: IPB Virtual Calendar does not allow the user to schedule appointments.
- No post-meeting summary.
- A static interface: The user cannot interact directly with the calendar.

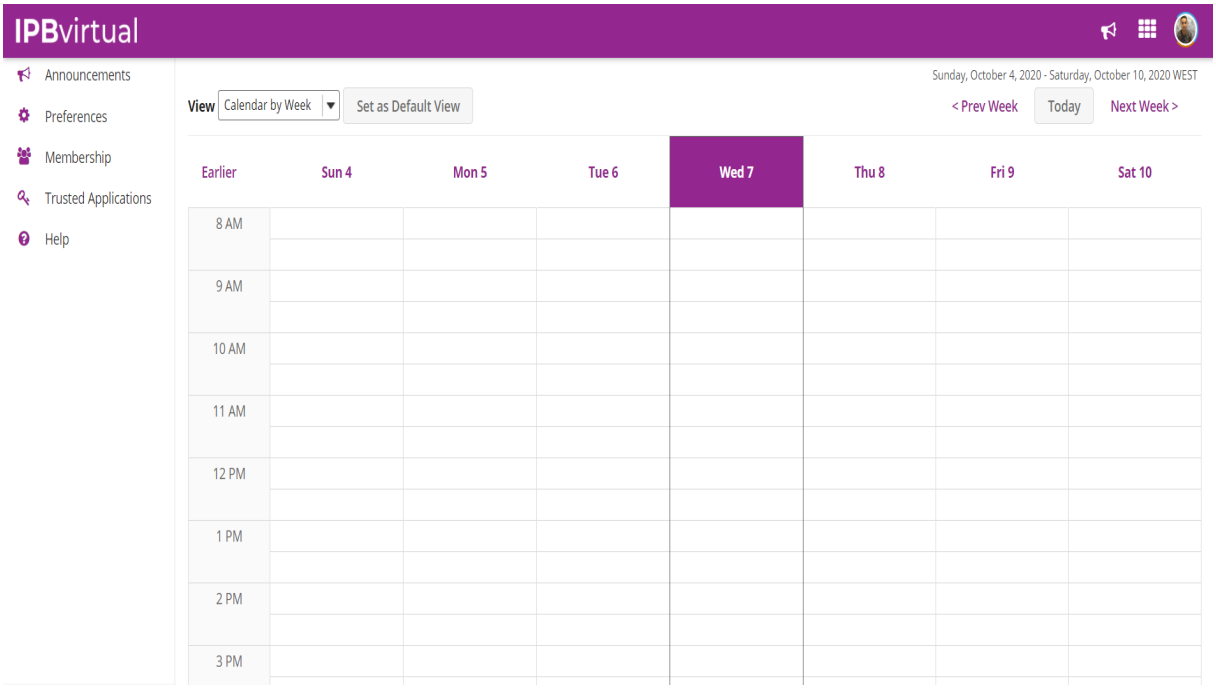


Figure 2.1: Screenshot of IPB.virtual Calendar

The second existing solution is Google Calendar [2], it is used worldwide to schedule events and also to organize meetings. Although it is a dynamic system which allows notifications and is linked to Google Hangouts, it is basically a master calendar with a bunch of individual calendars aggregated in one place, it is also integrated with Gmail to allow to create events on your calendar right from an email. Google Calendar runs entirely in "the cloud," meaning it is stored on a server to log in to, this means the possibility of accessto your calendar from any device that has an internet connection. Unfortunately, it has some negative aspects:

- No access to the IPB database: The absence of access to database means also many drawbacks since the user cannot consult any information regarding students or professors as well as curricular units. The user will also have no access to any information about the master level and there is a thesis and who is supervising it.
- No post-meeting summary: Although Google Calendar allows users to schedule

meeting, there is no option to generate a document containing the minutes of meetings as well as the highlights and conclusions.

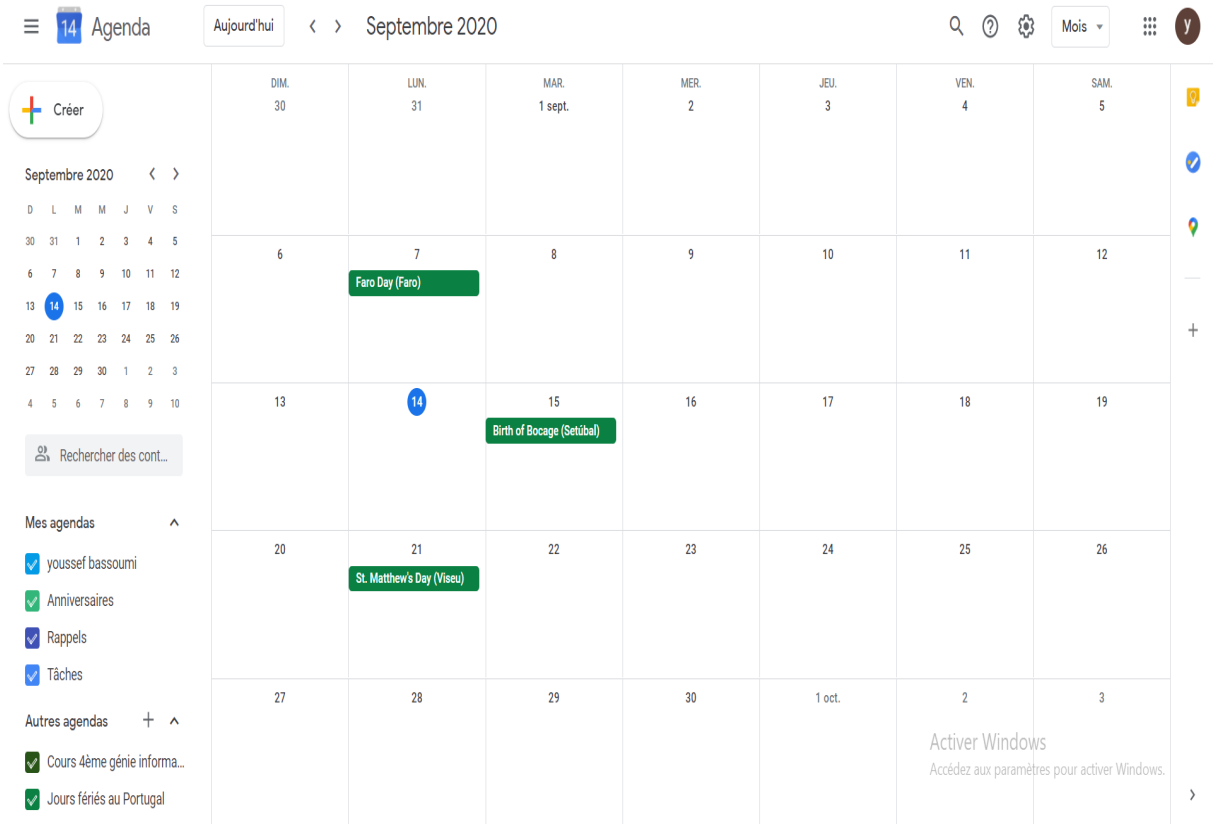


Figure 2.2: Screenshot of Google Calendar

2.3 Proposed solution

Following the study of "IPB.virtual Calendar" process and Google Calendar, several limitations were noticed. The objective of this module is to fix the issues previously stated, and to recommend the implementation of several functionalities in order to:

- Automate tasks that are processed manually.
- Facilitate research and access (24/24).

- Store information on computer media, which will ensure its security.
- Proposal meetings.
- Keep the history of previous meeting details and summaries.
- Generation of a post-meeting document containing details, addressed topics and conclusions. (Sent to participants upon completion).
- Consultation of the appointment schedule.
- Send documents to provide resources for participants of the meeting .
- Sending an email containing all the information of the meeting, jury meeting and event.
- Updating data and offering the possibility to add new students and professors.

Conclusion

This chapter allowed us to cover the state of the art and explore the existing solution. The necessity of creating such a platform was presented in detail, along with the suggested solution that will allow us to accomplish the goal. The following chapter will be focus on the specifications of the functional needs and also the nonfunctional needs that is necessary to be taken into consideration in order to satisfy the users.

Chapter 3

Requirement Analysis and Specification

In the next sub section it is essential to analyse the functional, nonfunctional elements of the project and the further details of the conception itself through some use case and sequence diagrams. Furthermore, addressing the site web map which shows how fluid and easy the user experience will be, in order to ensure accordance with the requirements of the hosting organisation. Fundamentally, in order to create a proper web application, the first step is thoroughly analysing the requirements imposed, from different perspectives, and merging them into the desired final product.

3.1 Functional needs

During the development phase, the basis of the process is represented by the implementation of the functional requirements. These include a meticulously planned structure of the features that need to be added in order to match the host organisations scope, such as:

- **Authentication:** This function is necessary to guarantee that only the intended persons will have access to the solution. As the database contains details about

both students and professors. Furthermore, every master director needs to see only his scheduled meetings and events.

- **Proposal of the meeting:** Through this function, the master director needs to be able to create meetings, jury meetings as well as events. He needs to be able to set the time and place as well as to consult the said activity within his calendar.

- **Send documents:** Upon the creation of an activity, the master director needs to be able to attach a desired document to the invitation mail. Allowing him to provide the participants with resources to consult before the meetings.

- **Consultation of the appointment schedule:** The schedule can be consulted in the calendar at any moment and the details can be modified by the master director. If there are changes in the time or place, the system will notify the participants.

- **Create event:** Aside from meetings, the user might need to create events, in which case he will need to specify the time and place and topic.

- **Reminder system notification:** After the creation of an activity (meeting, jury meeting or event) an e-mail is immediately sent to inform the participants of the time and place.

- **Creation of post-meeting PV:** After meetings, the discussed details as well as conclusions and perspectives may be featured in a pdf document and sent to the participants.

- **Aquisition of information into database:** The user must be able to fill the database by importing Excel table files in order to facilitate adding students and teachers into the system.

3.2 Non-functional needs

In order to guarantee user satisfaction, a set of non-functional needs must be considered, namely:

- **Data security:** data security throughout each step of accessing the platform, there are proper authentication and security measures implemented in order to ensure extra security of the databases. In the authentication process, access tokens were implemented in order to create safe communication and transfer of data between servers. Access to personal information is granted throughout special nominalisation and access rights, as all the resources used are administrated.
- **Extensible:** It is essential for any web platform to allow regular maintenance, improvements and additions of functions, if needed in the future.
- **Design:** As well as a functional design, the platform needs to present itself as user friendly and ensure compatibility with any other previous web applications used beforehand.

3.3 Use Case Diagram

In the development process, actors represent any type of entity that inhabits and interacts with the system but at the same time, is added as long as it helps achieving the goals of the project. For this system we have the main actor who is the master director. In relation to actors, the use case diagrams allow us to understand the purpose and layout of all the functionalities and how the user interacts with those functions as well as how these different parts interact with each other.

3.3.1 Global Use case Diagram

The global use case diagram is in overview of the functioning of the solution in which one can see the actions that the user can make and how they relate to other functionalities.

The user which is the master director, as can be seen in the diagram of figure 3.1, has the ability to manage his schedule which means managing events, meetings and jury meetings. He also has the ability to add a professor or student to the database whether individually or by lists. Furthermore, the user can consult informations about students or professors. This is only possible if the user has the ability to authenticate himself to access the platform.

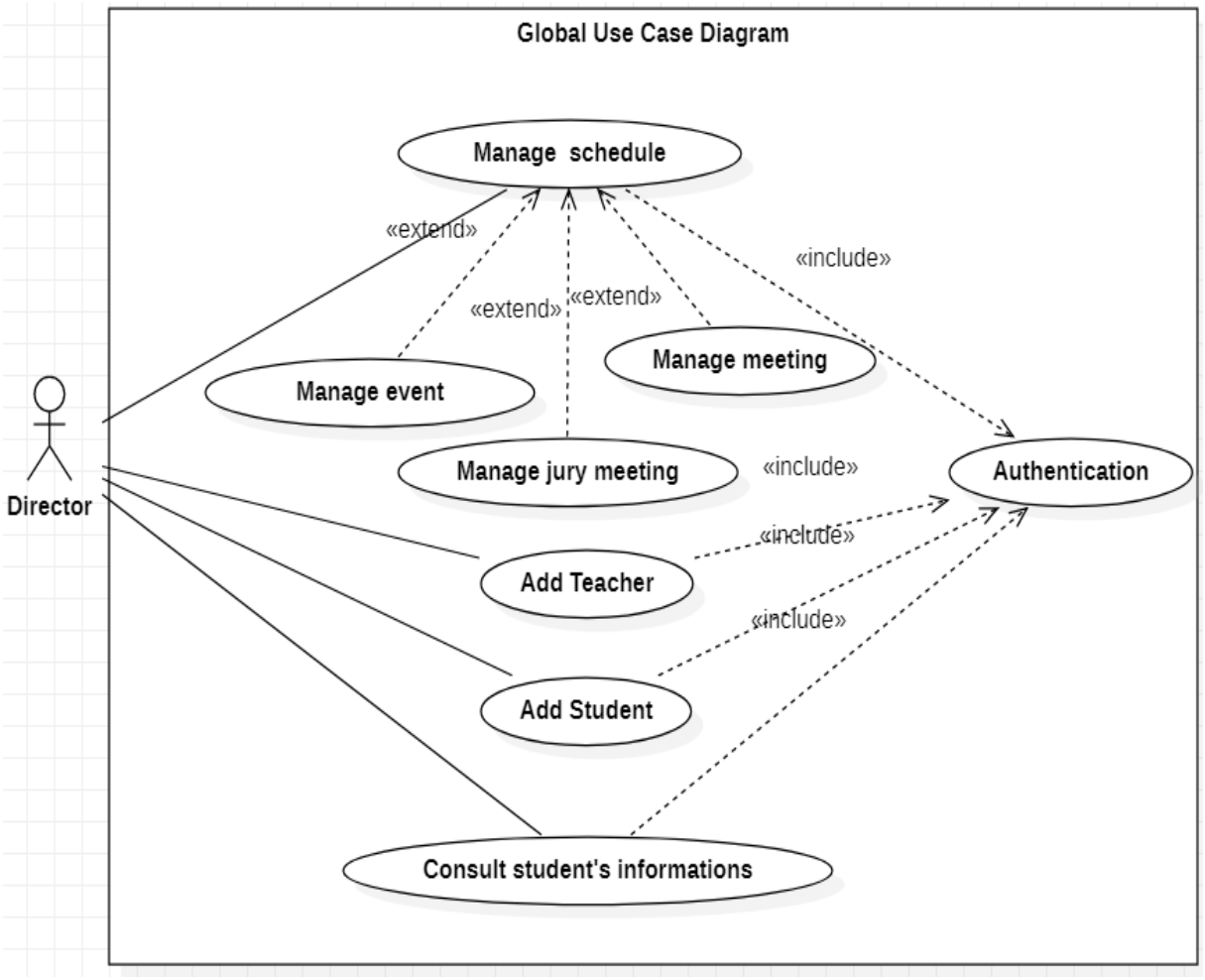


Figure 3.1: Global Use case Diagram

3.3.2 Detailed Use Case Diagrams

After seeing the global use case which showed us the general functioning of the solution and all the main actions the user can take. These main actions can be further separated

into more detailed functionalities that can be delved into by realising the detailed use case diagrams as will be shown in this section.

3.3.2.1 Use case Diagram of the Management of Meeting

Figure 3.2 is the detailed use case diagram of how the director can manage meetings. This diagram shows us that the user has the ability to manage his meetings by adding, deleting or updating them. Managing meetings also allows him to create a post meeting summary and attach relevant documents which can be sent via e-mail.

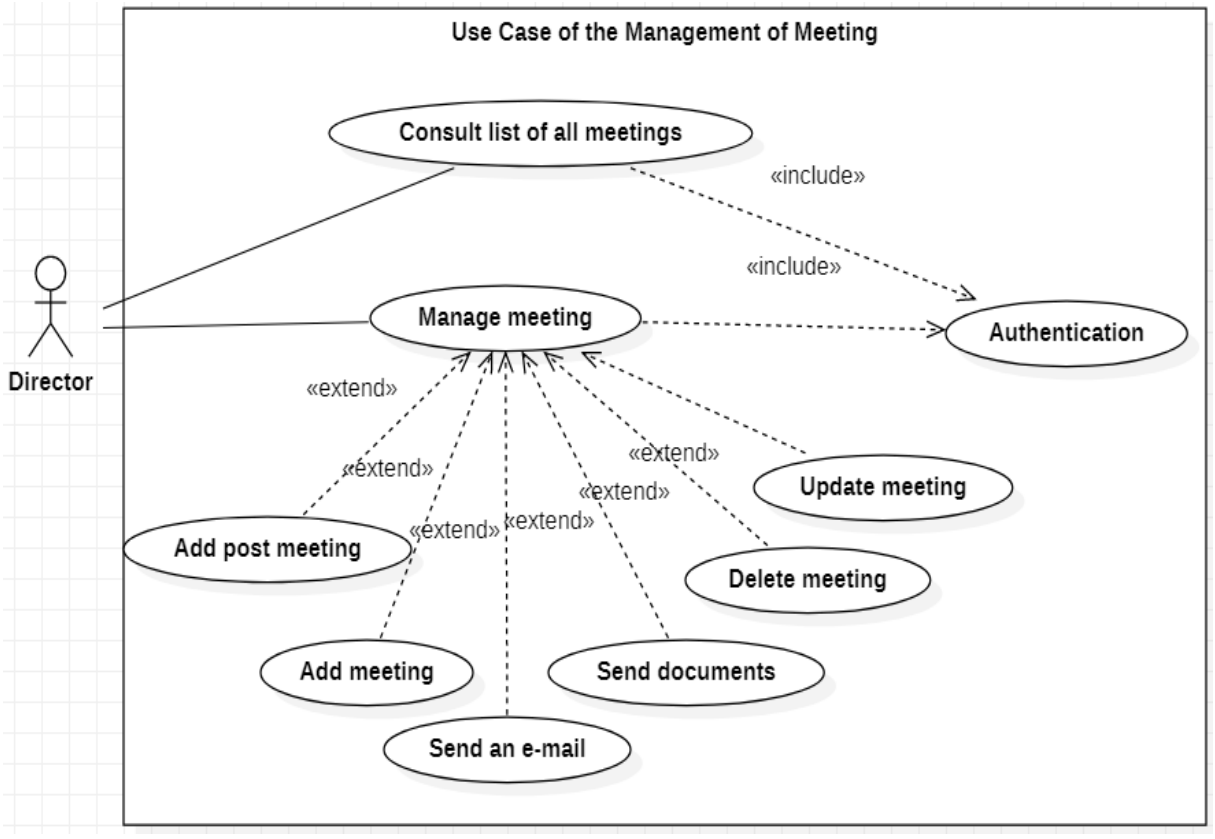


Figure 3.2: Use case Diagram of the Management of Meeting

The textual description of the “Management of Meetings” use case is illustrated by the following table in which one described the three scenarios of adding, updating and deleting a meeting.

Use Case	Management of Meeting
Actors	Director
Goal	This use case allows the Director to manage event
Scenario 1: Add Meeting	1. The Director clicks on the desired day in the schedule and chooses the add meeting from the list. 2. The Director fills the form that appears with relevant information. 3. The Director clicks on accept to add the meeting. 4. An email will be sent automatically. 5. A document will be sended with the mail if the director attach it in the form. 6. A PV will be downloaded if the director add it.
Scenario 2: Update Meeting	1. The Director accesses an existing event through the schedule or list of all meeting and click on the modify icon. 2. The Director modifies the desired informations in the form. 3. The Director clicks on confirm edit. 4. An email will be sent automatically.
Scenario 3: Delete Meeting	1. The Director accesses an existing meeting through the schedule or list of all meeting and click on the delete icon. 2. The Director clicks on OK.
Exception:	1. In Scenario 1 and 2, if the user select a student or teacher or class that is selected in the same time in another meeting or jury meeting or event. 2. In Scenario 1 and 2, if the user doesn't fill all the form, the system displays an error message.

Table 3.1: Use Case «Management of Meeting» details

3.3.2.2 Use case Diagram of the Event

Figure 3.4 is the detailed use case diagram of how the director can specifically manage one of the three types of scheduled activities which are the events including adding and modifying as well as deleting it. This is only possible if the user has the ability to authenticate himself to access the platform.

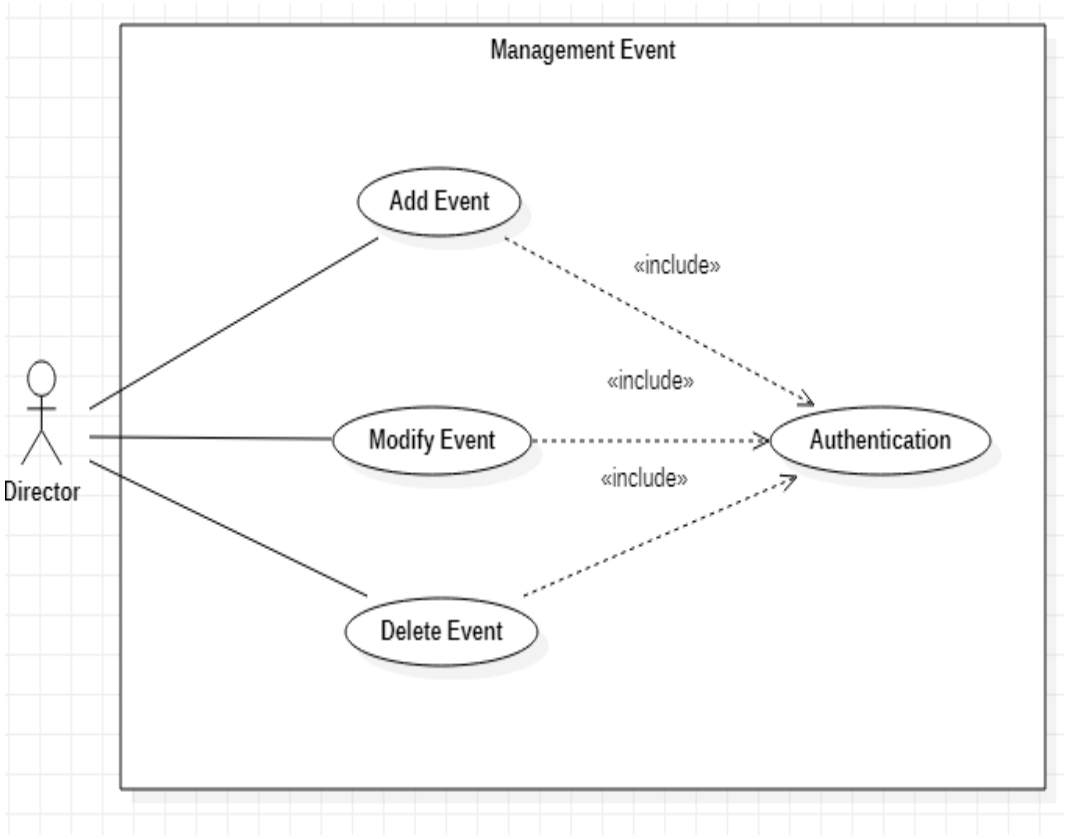


Figure 3.3: Use case Diagram of the Event

The textual description of the "Management Event" use case is illustrated by the following table in which one described the chronology of actions that will have to be carried out by the actor like the three scenarios of adding, modifying, deleting an event and by the system itself.

Use Case	Management Event
Actors	Director
Goal	This use case allows the Director to manage event
Scenario 1: Add Event	1. The Director clicks on the desired day in the schedule and chooses the add event from the list. 2. The Director fills the form that appears with relevant information. 3. The Director clicks on accept to add the event.
Scenario 2: Modify Event	1. The Director accesses an existing event through the schedule or the all events list. 2. The Director click on the modify icon. 3. The Director modifies the desired informations in the form. 4. The Director clicks on confirm edit.
Scenario 3: Delete Event	1. The Director accesses an existing event through the schedule or the all events list. 2. The Director click on the delete icon. 3. The Director clicks on OK.
Exception:	1. In Scenario 1 and 2, if the user doesn't fill all the form, the system displays an error message.

Table 3.2: Use Case «Management Event» details

3.4 Class Diagram

The general class diagram (Figure 3.5) represents the main classes which i will use to organise the database around. All the attributes are listed for each class and all classes are linked together via cardinality.

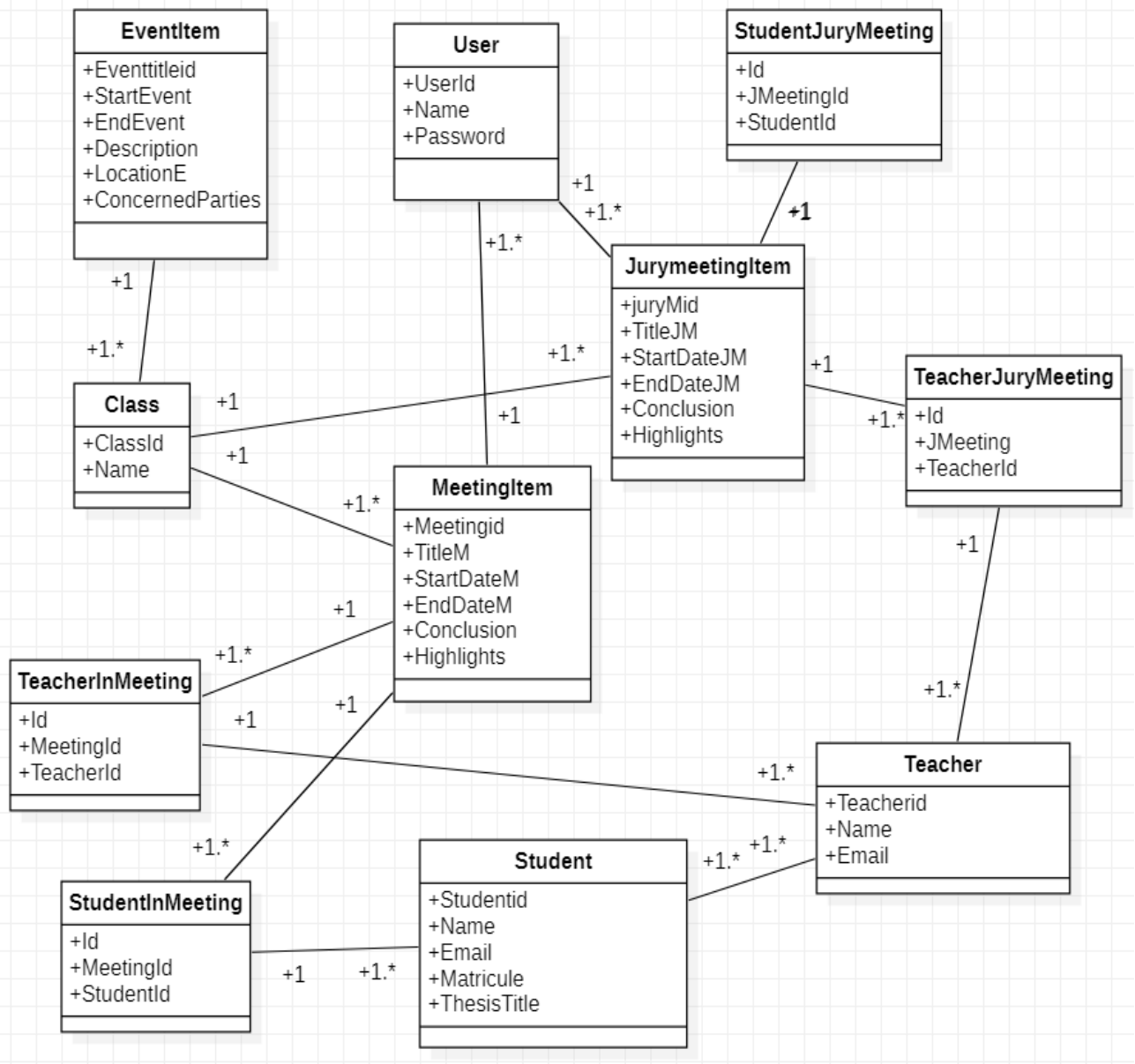


Figure 3.4: Class Diagram

3.5 Detailed Conception

3.5.1 Sequence Diagram «Authentication»

The user accesses his account via the password and the username thus authenticating himself to the server. This data would have been recorded in the table titled user and that allows the controller to perform a read operation on this table and finding it for the user in question before allowing access or denying it.

The system sequence diagram of the "Authenticate" is described by the following figure 3.5:

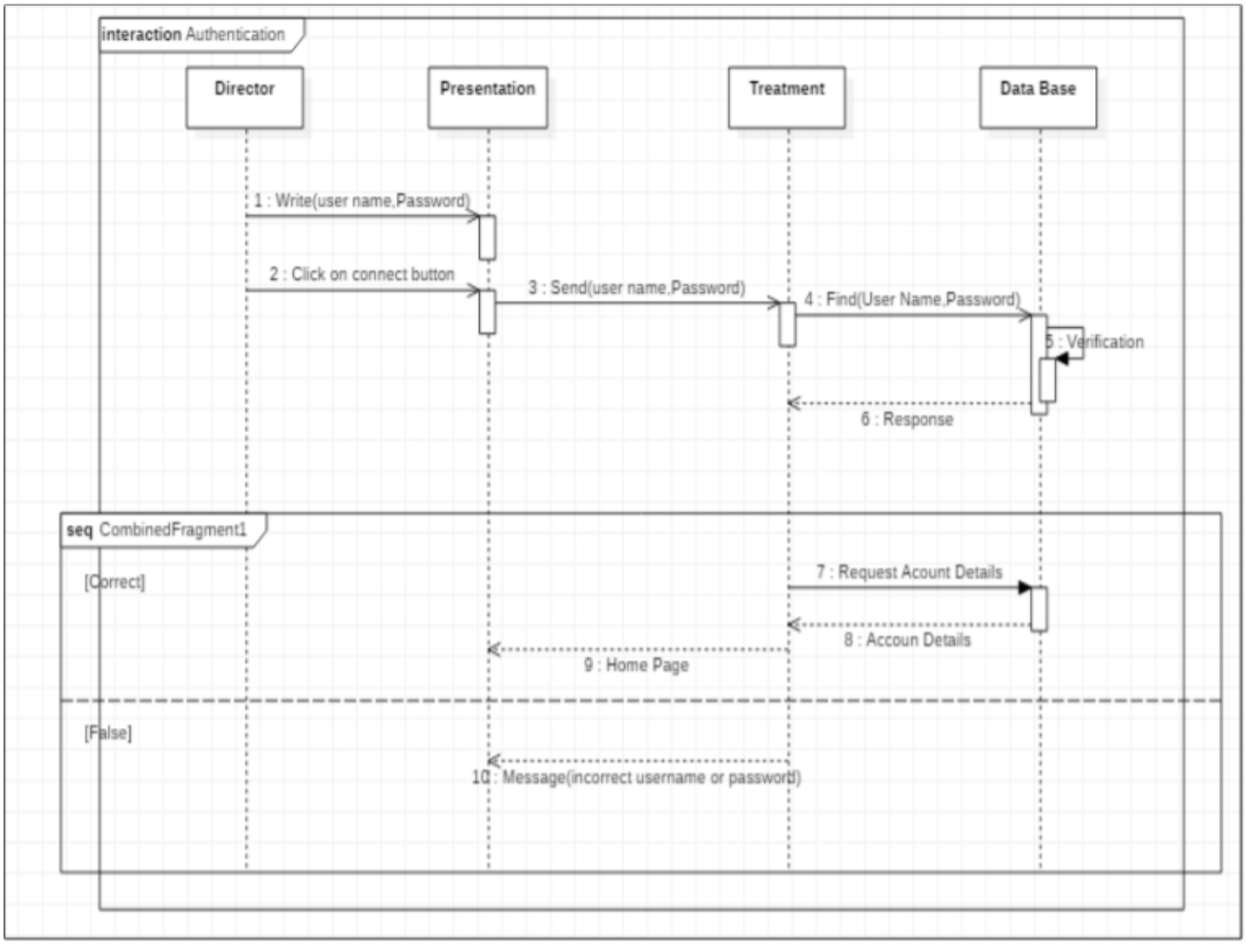


Figure 3.5: Sequence Diagram «Authentication»

3.5.2 Sequence Diagram «Manage Meeting»

This figure 3.6 shows how the director, after the authentication, has the ability to add a new meeting, delete an existing one or modify it at any time.

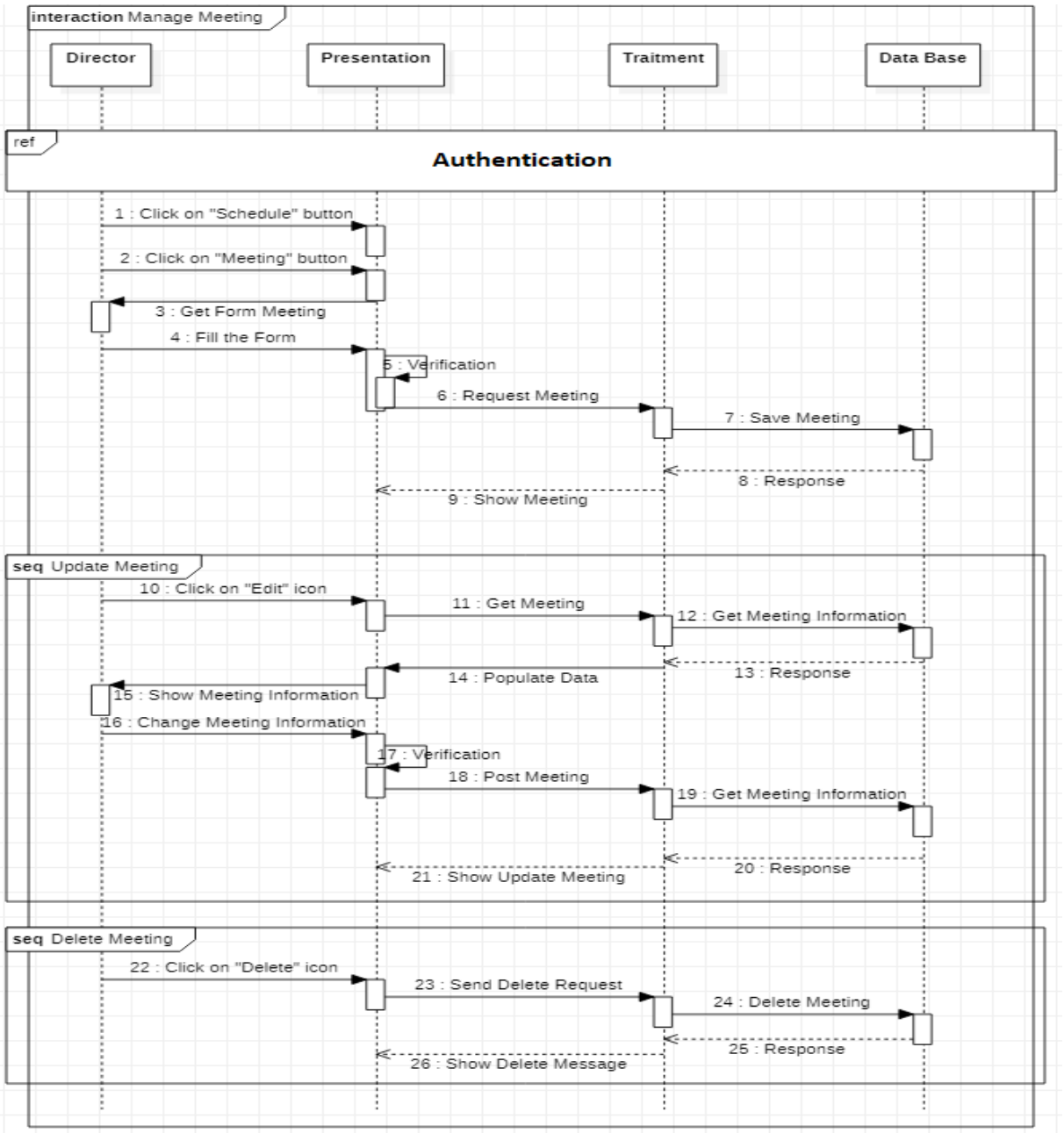


Figure 3.6: Sequence Diagram «Manage Meeting»

3.5.3 Sequence Diagram «Send E-mail»

In this figure 3.7, we can see that when a meeting is created by the director, an e-mail with the title and details is automatically sent to all participants informing them of the time, place and contents desired by the user.

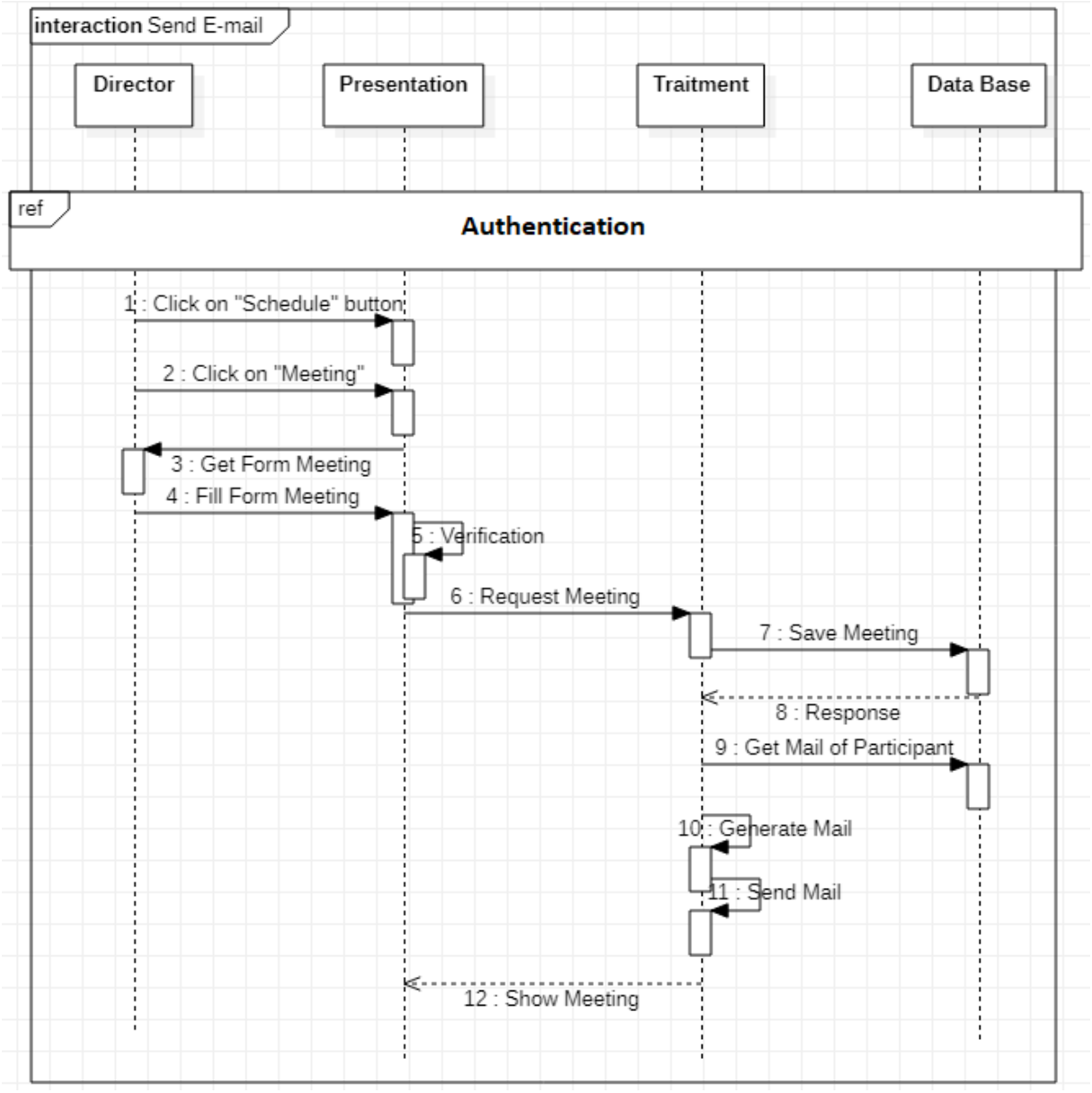


Figure 3.7: Sequence Diagram «Send E-mail»

3.5.4 Sequence Diagram «Generate PV»

This functionality allows the user to generate a PDF document containing the participants as well as the minutes of the meeting, the highlights and a conclusion. This PDF is sent to the involved parties as a reference.

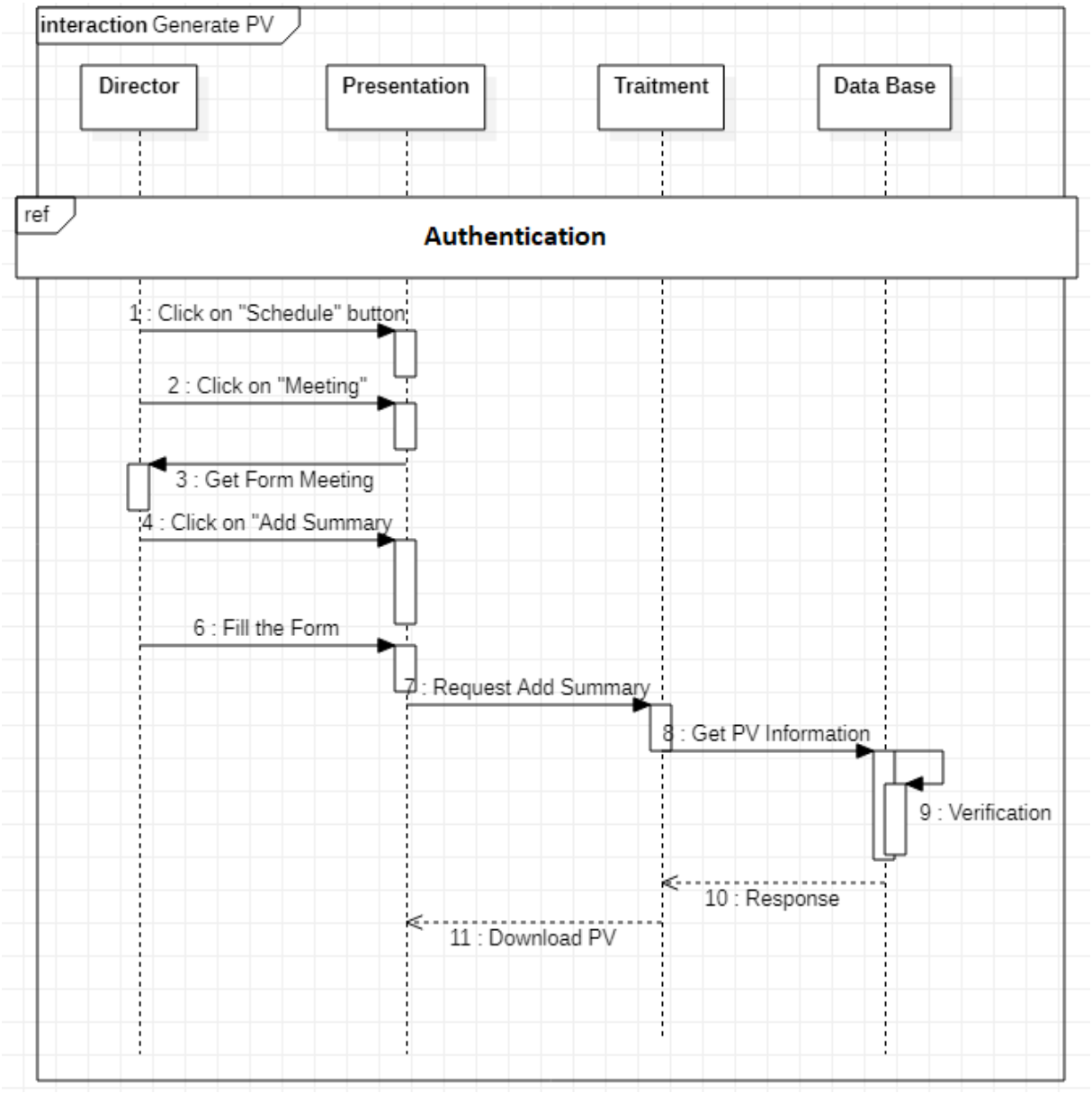


Figure 3.8: Sequence Diagram «Generate PV»

3.6 The web site map

The following web site map (Figure 3.9) represents the links between the graphical interfaces starting from the login page that leads to the home page and all the accessible interfaces that branch from there.

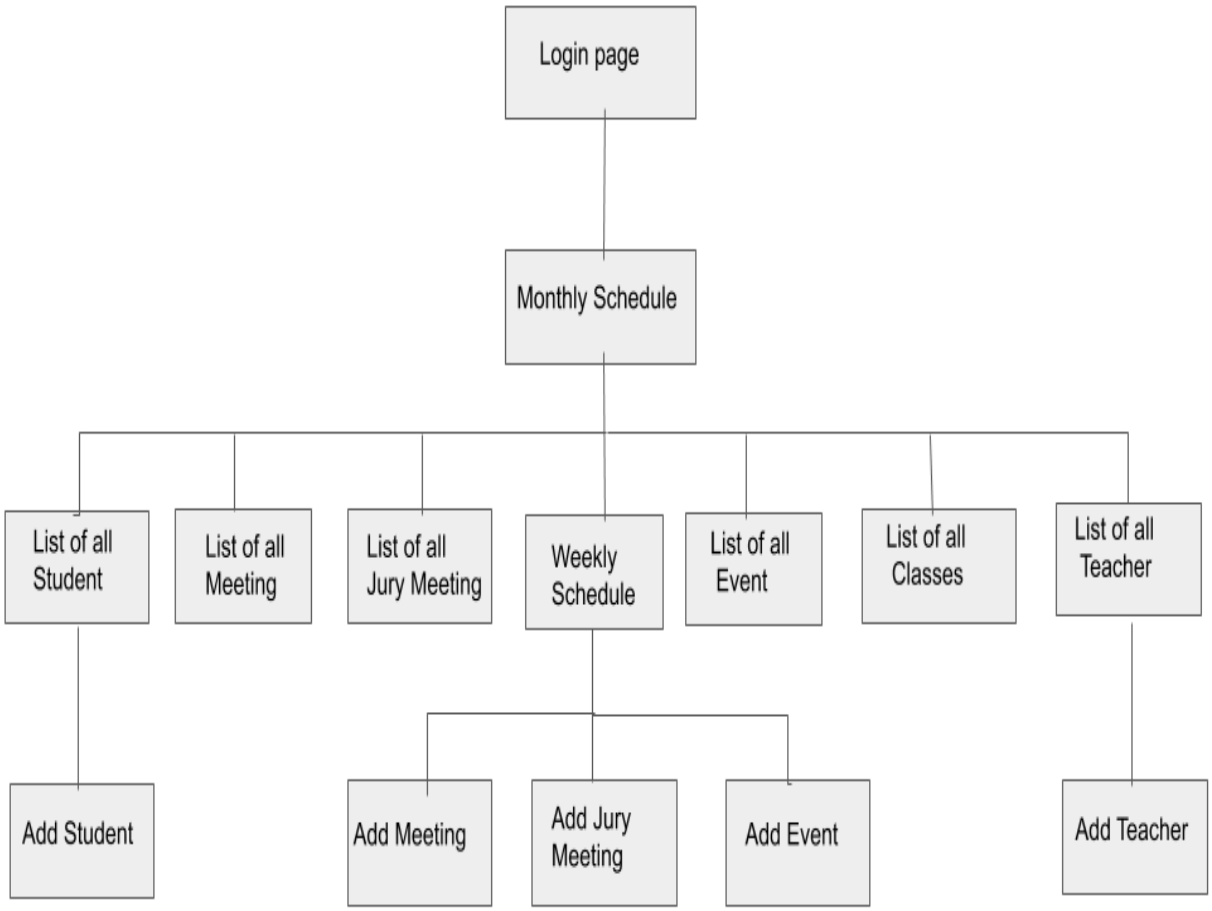


Figure 3.9: The web site map

3.7 Database Schema

To delve into deeper details than in the general diagram that one create the following database schema using the primary and secondary keys to manage mySQL server.

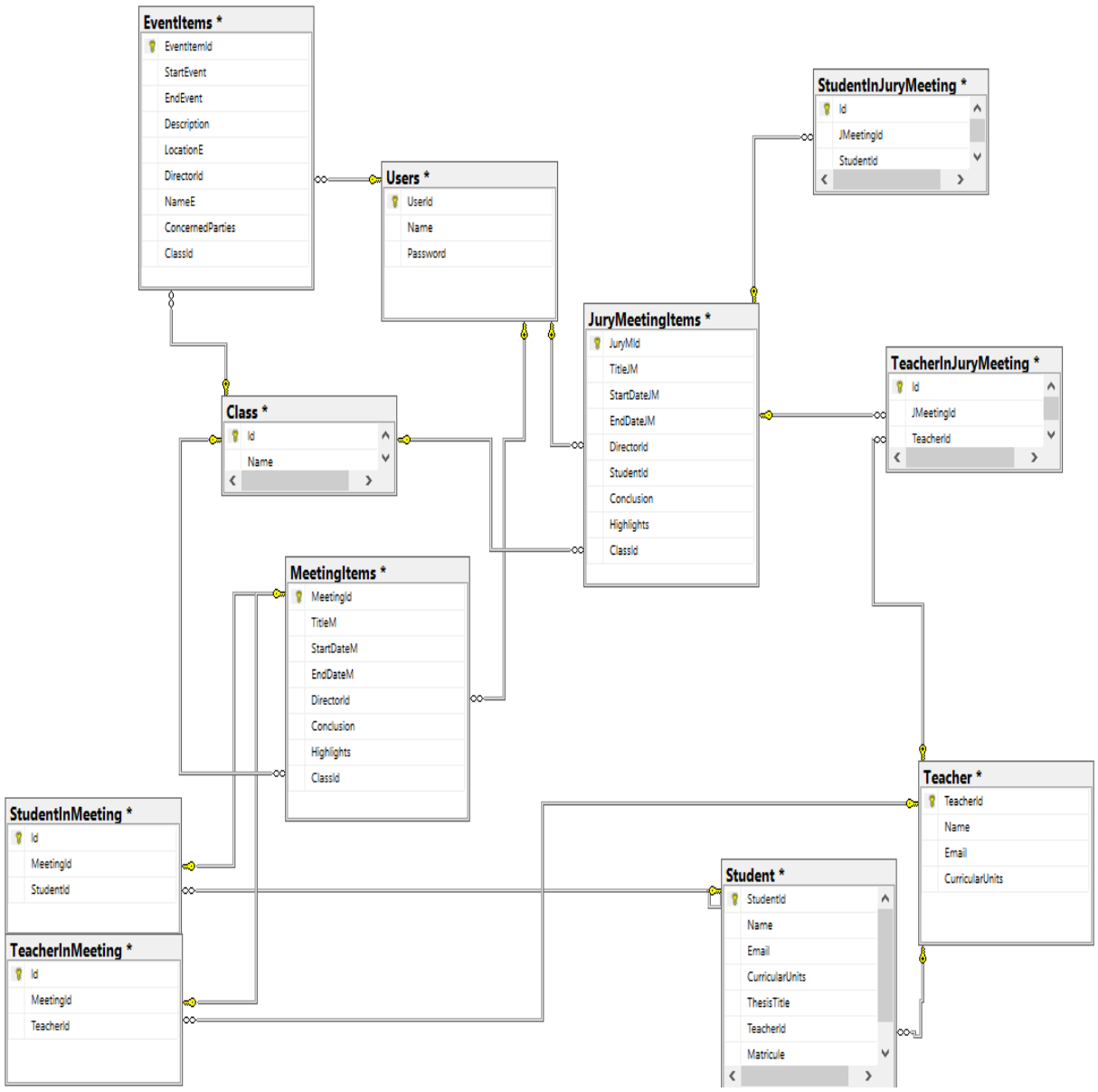


Figure 3.10: Data Base Schema

Conclusion

In this chapter, we specified the functional and non-functional requirements of the solution, that one have also explored the way they will interact with the solution through the global and detailed conception which gives a clear idea of the overall functioning of the solution. The chapter concluded with the web site map which shows how fluid and easy the user experience will be and database schema.

Chapter 4

Technologies and Development Tools

This chapter will explore the adopted technologies along with the suggested frameworks of the work concerning our project that will allow us to accomplish our goal as well as the overall structure of the solution and explore the details of the followed architecture.

4.1 Front-end Framework

In order to choose the appropriate front-end technology for web frameworks, it is essential to assess to the existing options .

4.1.1 Angular

Angular is a platform and framework for building and designing sophisticated single-page client applications using HTML and TypeScript. Since it is written in TypeScript, it implements core and optional functionality as a set of TypeScript libraries that is import into apps. In order to use the Angular framework, one should be familiarized with Javascript, HTML, CSS. The architecture of an Angular application relies on some fundamental concepts. The basic building blocks of the Angular framework are Angular components that are organized into NgModules. These components use services, which provide specific functionality not directly related to views. Service providers can be

injected into components as dependencies, making the final code modular, reusable and efficient. Finally, Angular JS lets us extend the HTML vocabulary of the application which results in a readable, expressive and quick to develop environment[3].

4.1.2 Vue

Vue is a progressive framework for building user interface which is designed from the ground up to be incrementally adoptable unlike other monolithic frameworks. The core library is focused on the view layer only, and it is simpler to pick up and integrate with other libraries or existing projects. On another level, the Vue framework is also capable of powering sophisticated Single-Page Applications when it is used in combination with modern tooling and supporting libraries. As a progressive JavaScript Framework, Vue focuses on building user interfaces. It is smaller and light weight when compared to other more complex front-end frameworks like Angular JS, and other kinds of javascript frameworks. Being designed from the ground up to be incrementally adoptable makes it very easy to integrate into an existing project. In addition it is very simple to use and easy to get started. It is important to know that it can do anything with the user interface, as it doesn't care about the model or database[4].

4.1.3 React

React JS is one of the most popular Javascript frameworks and here are a few reasons: IT is an open source JavaScript library which is used for building user interfaces, define simple views for each state of your application, and when your data changes, React will optimally update just the components that need it. It allows developers to create intuitive, interactive and engaging applications with minimum coding and the best rendering performance. Basically it gives developers the ability to work with a virtual browser that is more friendly than the real browser. React also uses one-way data binding, It's easier to debug self-contained components of large React JS apps. Finally the React JS applications are easy to test[5].

the below figure 4.1 present the Mounting and re-rendering components in React.

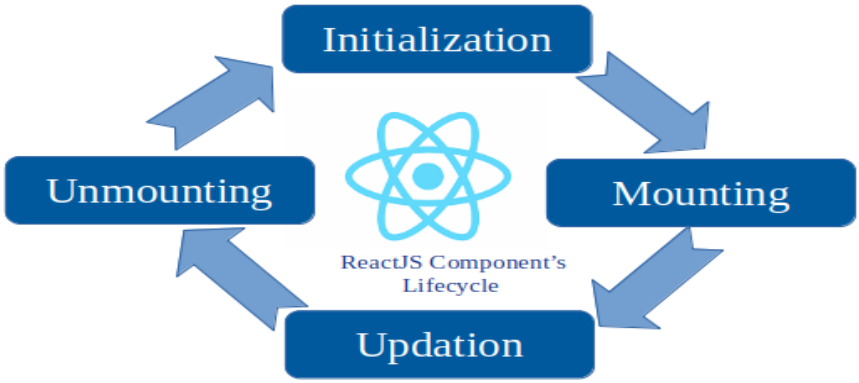


Figure 4.1: Mounting and re-rendering components in React.
[6]

4.2 Back-end Framework

After seeing the available choices for the front-end, we must now move on to choose the appropriate alternative for the back-end, while many solutions offer the same services, a small detail can guide the choice when we are keeping in mind the features and specific needs of the final project.

4.2.1 Laravel

Laravel is a free and open source PHP web application framework which focuses on the simplifying of simple tasks used in the majority of projects such as authentication, routing, sessions. It is intended for the development of web applications following the model–view–controller (MVC) architectural pattern and based on Symfony. Some of the features of Laravel are a modular packaging system with a dedicated dependency manager, different ways for accessing relational databases, utilities that aid in application deployment and maintenance. Another useful feature is the fact that Laravel reuses the

components from other framework in developing web application. It includes name spaces and interfaces, thus helping to organize and manage resources.[7]

4.2.2 Django

Django is a free and open source web application framework written in Python. A framework is nothing more than a collection of modules that make development easier. They are grouped together, and allow us to create applications or websites from an existing source, instead of from scratch. As "A high-level Python Web framework", it encourages rapid development and pragmatic design. Django offers a big collection of modules that one can use in the own projects. Primarily, frameworks exist to save developers a lot of wasted time and headaches and Django is no different. But it is also flexible and extensible, allowing developers to augment the template language as needed. The main goal of the Django framework is to allow developers to focus on components of the application that are new instead of spending time on already developed components, thus enabling users to focus on developing components needed for their application [8].

4.2.3 Asp.Net

Asp.Net is a developer platform made up of tools, programming languages, and libraries for building many different types of applications. its base platform provides components that apply to all different types of apps. ASP.NET is a web application framework developed and marketed by Microsoft [9] to allow programmers to build dynamic web sites. It allows us to use a full featured programming language such as C or VB.NET to build web applications easily. ASP.NET also provides a programming model, a comprehensive software infrastructure and various services required to build up robust web applications for PC, as well as mobile devices. Being part of Microsoft .Net platform, ASP.NET applications are compiled codes, written using the extensible and reusable components or objects present in .Net framework. These codes can use the entire hierarchy of classes in

.Net framework. These features are used to produce interactive, data-driven web applications over the internet with a large number of controls such as text boxes, buttons, and labels for assembling, configuring, and manipulating code to create HTML pages.

4.3 Technologies Choices

With all the technologies in the world changing on a daily basis, it is hard to devote time to learning a new framework, especially when that framework could ultimately become a dead end, so we are feeling a little bit lost in the framework jungle. Choosing the right framework upfront is critical as a huge amount of effort is required to switch later. By the suggestion of the coordinators, we have settled down on frameworks that are as efficient as up to date with the current technology. We have also highly taken into consideration the best implementation procedures in order for the framework to ensure future ease of maintainability and upgrades.

4.3.1 Front-end choice

After examining the available front-end possibilities, we settled on using React JS for the mentioned reasons, however using it is not enough so we needed to add the Redux library to create a more sophisticated schedule solution. Redux library is an open source and is the more commonly used with libraries like React or Angular for building user interfaces, indeed, redux can help in writing applications that behave consistently as well as run in different environments (client, server and native). The Redux architecture allows the user to log changes as well as "time-travel debugging", and even send complete error reports to a server and can be a predictable state container for JavaScript applications. One of the features that make Redux useful for the project is that it automatically implements complex performance optimizations, thus ensuring that a component only re-renders when the data it needs changes.

4.3.2 Back-end choice

For the choice of back-end technology, we settled after careful study of the existent options on the ASP.NET solution and specifically ASP.NET core 3. Indeed, ASP.NET Core is a Free open Source framework for Linux, Windows and macOS operating systems. Finally the choice is guided by the following advantages:

- Cross platform.
- Unified (It offers a single unified component like .NET Standard library for all platforms with the same code, same languages and same tools).
- High performance (Packages reduce the request pipe line and improve the application performance).
- Lightweight (It allows the deploy libraries and components that are needed in set of package.
- Hosting (.Net Core application can be hosted on multiple Web servers such as Apache, Docker...).
- Testability - Unit testing is very easy in .Net core applications.

4.3.3 Other Technologies

In addition to the two main front-end and back-end technologies, we need to use some other technologies for the solution to run smoothly and to guarantee fluid communication between the different components.

4.3.3.1 Microservices

Microservice architecture is a method of developing software applications as a suite of modular and separate services, in which each service runs a unique process and communicates through a light and well-defined mechanism to achieve a business objective.

First, software built as microservices can, by definition, be divided into multi-component services, So that each of these services can be deployed, adjusted, and then redeployed independently without compromising the integrity of an application. As a result, you will only need to change one or more separate services instead of having to redeploy entire applications Second, the microservices style is generally organized around the capabilities and priorities of the business. Third, microservices are designed to cope with failure. we chose this architecture thanks to its multiple characteristics[10]:

- Microservice architecture gives developers the freedom to develop and deploy independently.
- Code for different services can be written in different languages.
- Easy integration and automatic deployment.
- Developers can use the latest technology.
- When a modification is required in a certain part of the application, only the related service can be modified and redeployed: there is no need to modify and redeploy the entire application.

Figure 2.4 illustrates the micro service architecture[11].

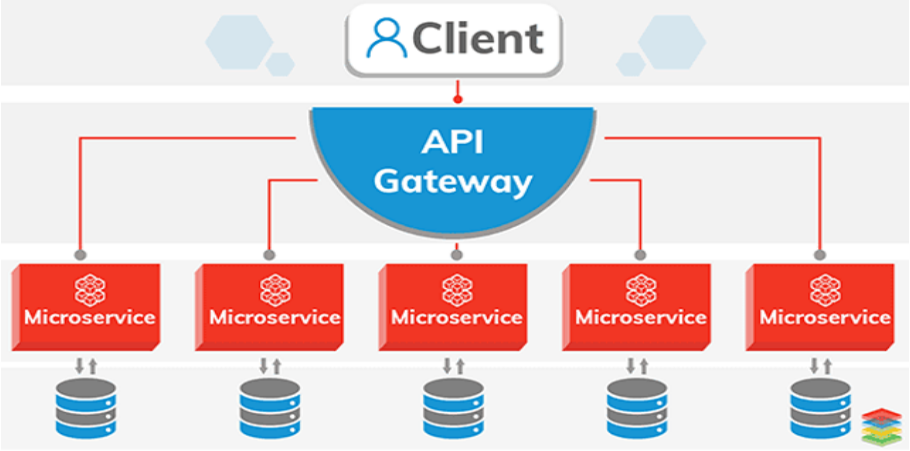


Figure 4.2: Micro Services Architecture

4.3.3.2 SQL Server Management Studio

SQL Server Management Studio (SSMS) is an advanced development and integrated environment to manage all SQL infrastructure. that one use SSMS to access administer, manage configure and develop any components of SQL Server, SQL Data Warehouse and Azure SQL Database. SSMS is widely used because of the following advantages [12]:

- Various add-in options.
- Better metadata memory management.
- A low-memory load path for column store tables.
- The advanced security features: Always Encrypted (user update encrypted data without having to decrypt it first).

4.3.3.3 ASP.Net Core Identity

For the subscription system, Identity ASP.NET allows the addition of functionality to the app connection. Via this system, any user has the capacity to register and thus creating a new account or login through an already existing one. Furthermore, he can access the app via an external connection provider if the user has accounts in Facebook, Google, Microsoft, Twitter, etc... User names, passwords and all profile data can be stored by configuring the ASP.NET core identity to use an SQL server database. Of course the option remains for the user to use his own persistent storage[13].

4.3.3.4 Entity Framework

Entity Framework is an Open Source product of Microsoft, it is independent to the .NET release cycle.Entity Framework works with all relational database that has a valid Entity Framework provider, it allows to insert, delete and update command generation. Entity Framework will execute the relevant query from database and then materialize results into instances.

Figure 4.3 present the architecture of entity framework.

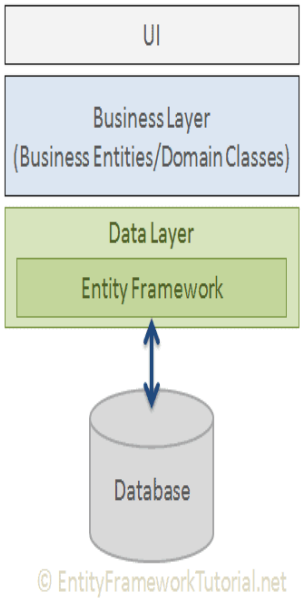


Figure 4.3: Architecture of Entity Framework
[14]

4.3.3.5 StarUml

It is an open source tool allow us to visualize the essence of the ideas and software designs. It allow us to quickly and effortlessly build diagrams that create a clear understanding among teams.

4.4 General Structure

The web application architecture describes the interactions between applications, databases and middle ware systems on the web and is the reason why multiple applications can work simultaneously. In any typical web application, there are two different codes (sub-programs) running in parallel: Firstly is the client-side code, which is the code contained in the browser and which responds to user inputs and secondly we have the server-side

code, which is the code contained on the server and responds to the HTTP requests from the browser to send back it files to be executed. The next part will explain the system architecture chosen.

4.5 Architecture Used

The used architecture for that solution will be the MVC model that is composed of three layers as shown in figure 4.1 :

Model: component that corresponds to all the data-related logic that the user works with. This can represent either the data that is being transferred between the View and Controller components.

View: is the user interface.

Controller: act as an interface between Model and View components to process all the business logic and incoming requests, manipulate data using the Model component and interact with the Views to render the final output.

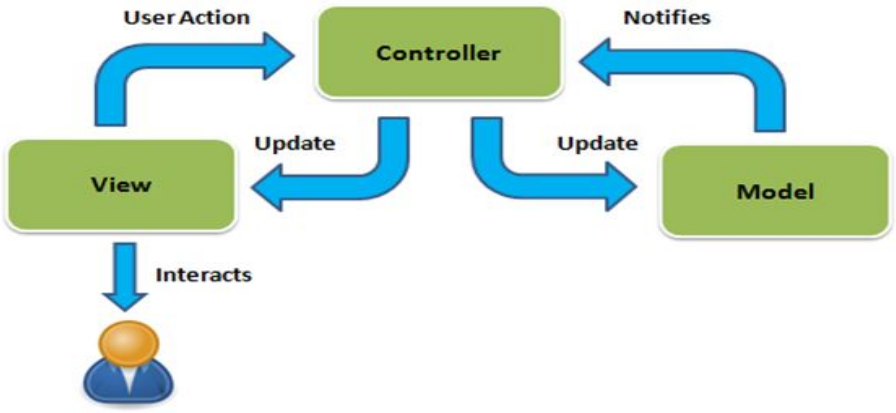


Figure 4.4: MVC Model

[15]

Conclusion

In this chapter, one chosen the technologies which will best suit with our project while taking into account our primary user who are the Master Directors. Then we have explained the general structure of the solution and the further explored the followed architecture. The following chapter will go through the steps of the implementation of our solution.

Chapter 5

Development of iMaster.Schedule web application

5.1 Introduction

After building a clear idea of the project through the previous chapters, that one finally moved on to create the solution using the mentioned technologies and following the decided upon structure. In this chapter we will expose the features of this realised product through the various interfaces of the front-end as well as the back-end.

5.2 Interfaces of the Front-end

the front-end choices manifest themselves finally as graphical interfaces that the user can interact with, here are the interfaces that the users will face when using the solution.

5.2.1 Interface of Authentication

The following figure 5.1 represents the interface for the Authentication page:

Figure 5.1: Interface of Authentication

As can be seen in the image, the authentication process is simple, the user needs to fill two boxes with name and password respectively and the system will get the Token from the API to access to the home page. However, in case the user does not have a previous account he can press the register button in order to be registered as a new user.

5.2.2 Interface of schedule

The monthly schedule appears in the home page and will be the first thing that the user sees after authentication. This schedule will first show the current month but the user has the ability to switch to other months via the arrow buttons. The days which contain items such as events or meetings will be highlighted with the appropriate color code or a unique colour mixture. The next figure 5.2 is a screenshot of the homepage as seen by the director in which he can consult his schedule:

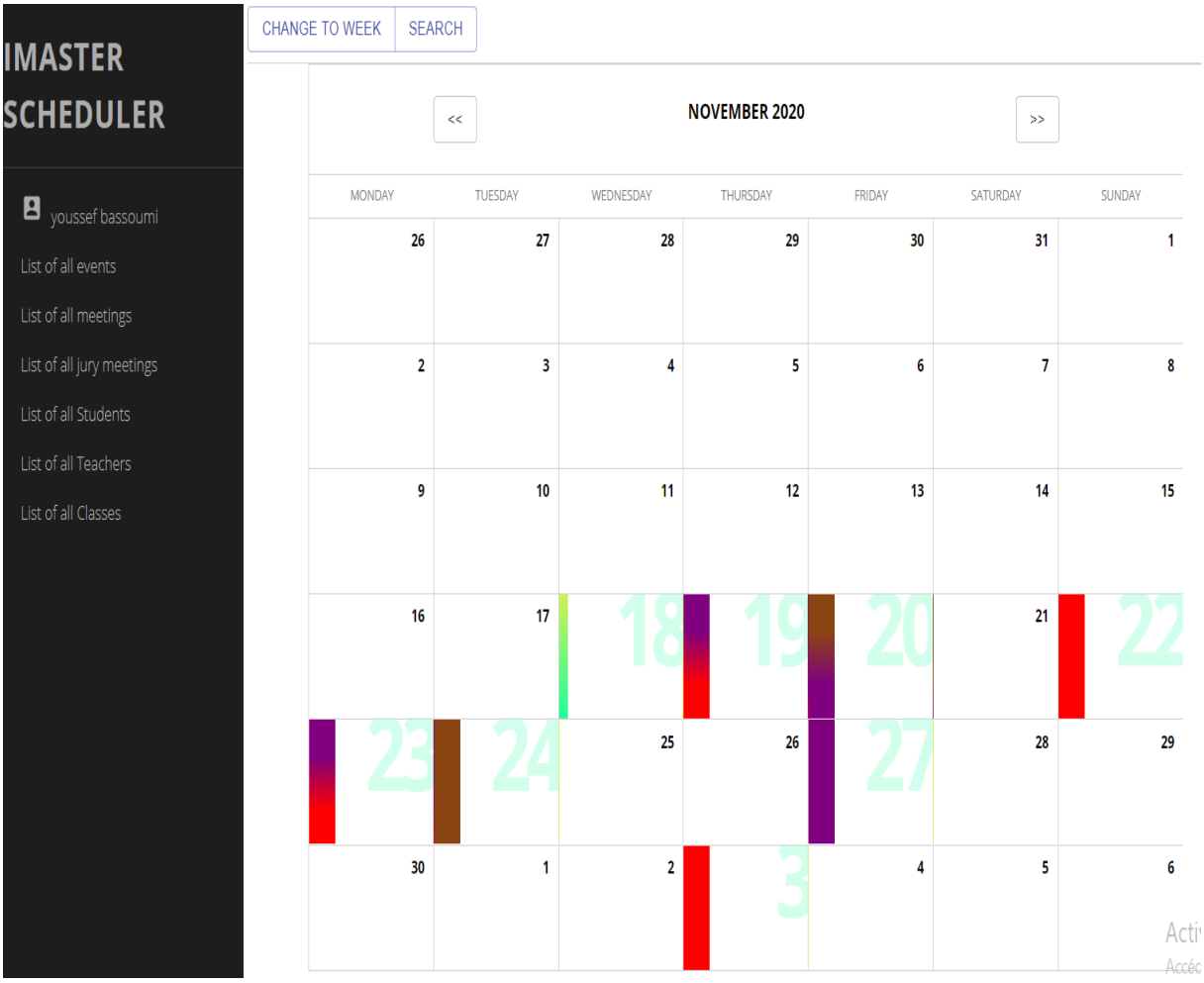
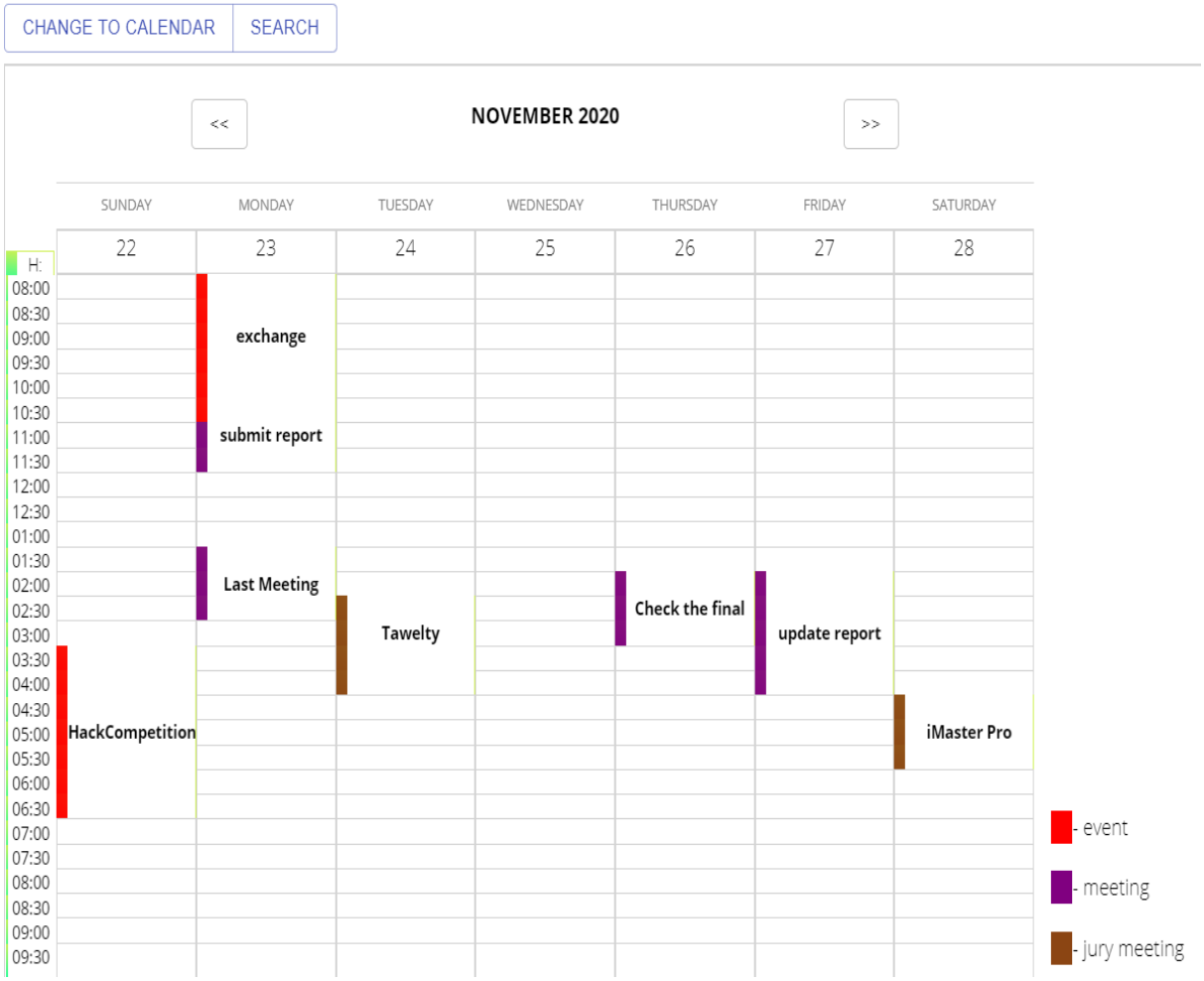


Figure 5.2: Interface of the schedule

5.2.3 Interface of the weekly schedule

By pressing the change to week button, the user has the ability to consult the weekly schedule interface which will feature more details than the monthly one. By moving to this interface, that one can now see the hours as well as the days of a single week. Furthermore the user can see the title of the item next to a color code bar which depends on the type and will be purple if it is a meeting, brown if it is a jury meeting and red if it is an event. The master director can access his further details by seeing his weekly schedule as represented in this figure 5.3:



3

Figure 5.3: Interface of the weekly schedule

5.2.4 Interface of list of all meeting

By clicking on the list of all meetings button, the user can access the exhaustive list of all scheduled meetings. They will appear in a table that will show the meeting ID, the title, as well as the starting and end times. From this table, the user can in addition consult, modify or cancel the meetings. Another feature is the fact that the user can consult the PDF file of post-meeting summary if it exists directly from this list. This interface represents the list of all meetings which the director can consult as seen in the figure 5.4 :

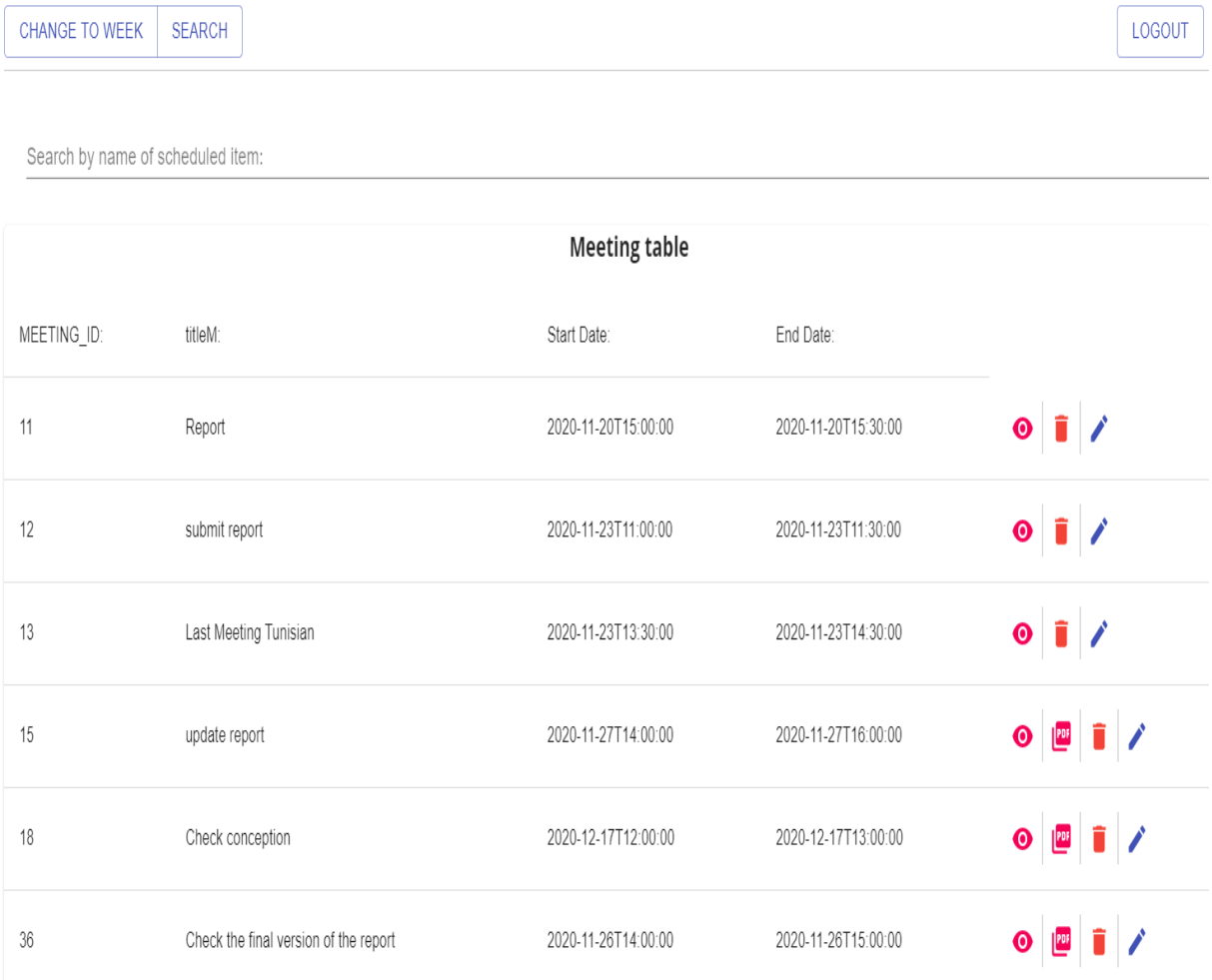


Figure 5.4: Interface of list of all meeting

5.2.5 Interface of add meeting

By pressing add meeting, the user can access the add meeting interface, there he can set all the details of the desired meeting. Firstly we can set the time and duration of the meeting by defining the start and finish times. That one can also add the title of the meeting as well as the location . After that, that one can move on to choosing the participants which can be either students or professors. The participants might need to consult some documents before the meeting as a preparation which is why the choose file button was added which the user can click to attach any type of file. To add a new meeting, the master director will fill the form that this next figure 5.5 lays out:

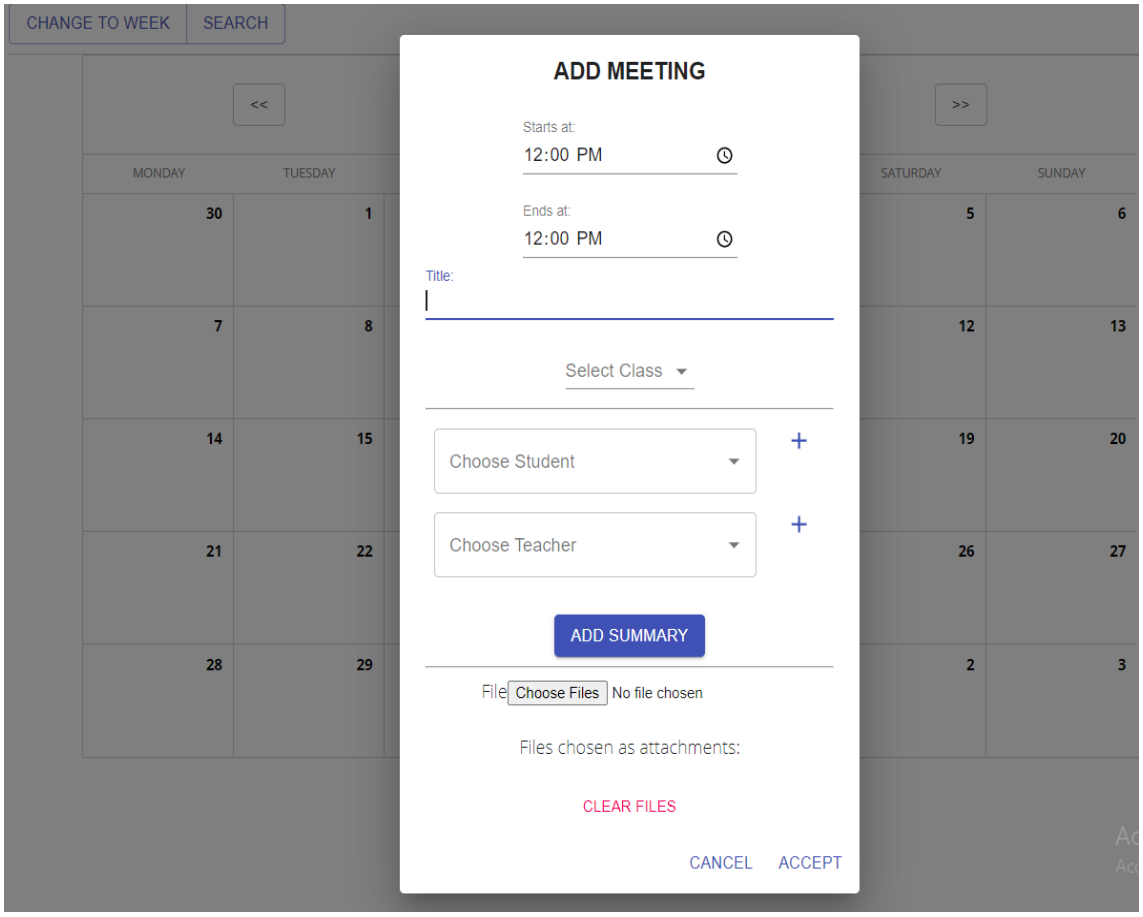


Figure 5.5: Interface of add meeting

5.2.6 Interface of the post meeting "PV"

After a meeting, the user can generate a pdf document which will contain the a list of the involved parties as well as the highlights and conclusions of the meeting. In the following figure 5.6, in that one can see an example of a post meeting PDF file.

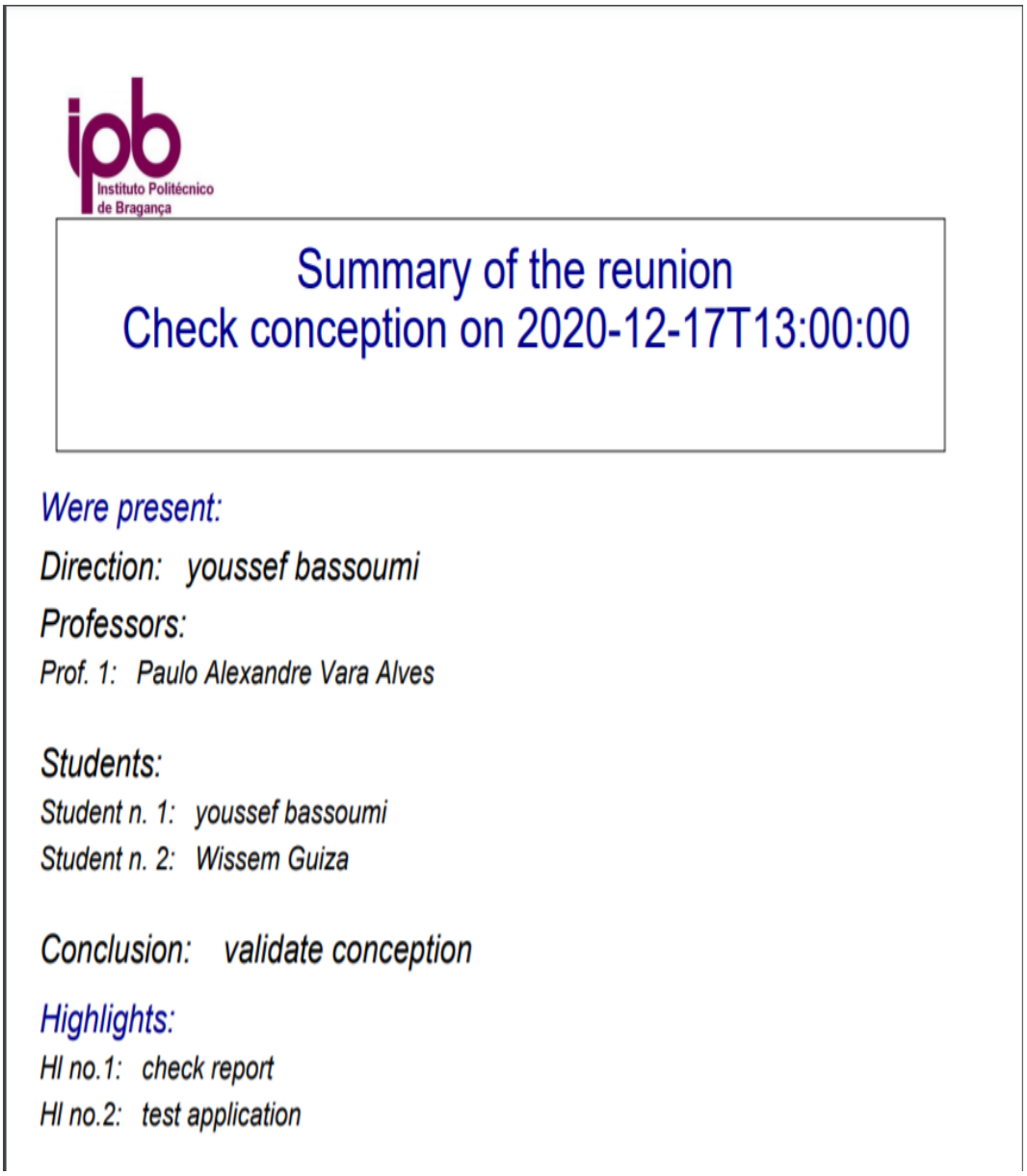


Figure 5.6: Interface of the post meeting "PV"

5.2.7 Interface of Add Event

To be able to add an event, the user has to click on the desired day in his schedule and choose the add event option which will reveal the represented form. The form has to be filled with the starting and ending times as well as the location of the event and its name. After that the user can choose if only students or teachers or both are concerned. Furthermore, by clicking on choose files the user can attach any relevant documents to be consulted before the meeting. Besides creating meetings, the master director can also create events as shown in the following image :

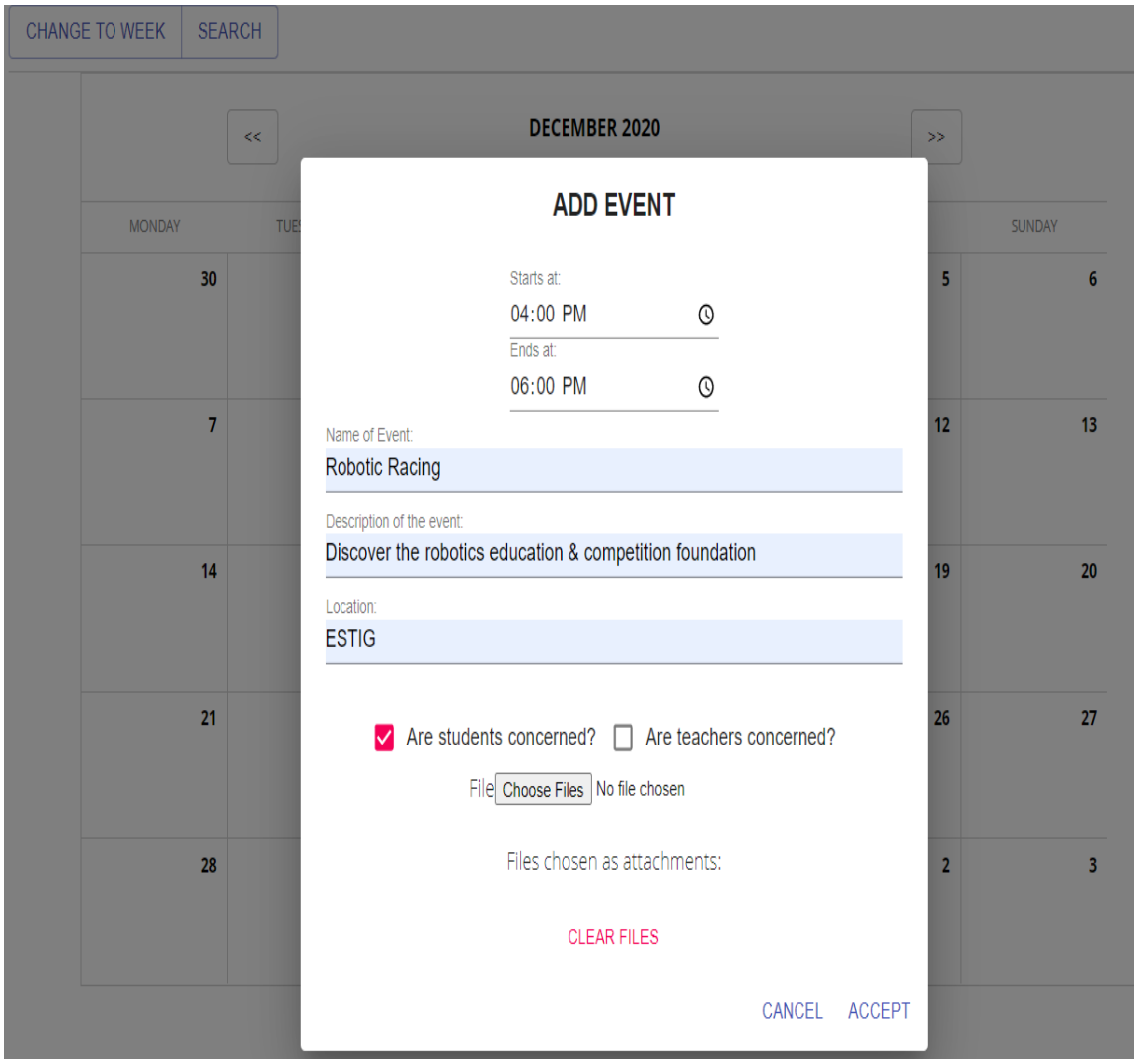


Figure 5.7: Interface of Add Event

5.2.8 Interface of the E-mail

The following figure 5.8 shows us an example of an e-mail received by an involved party, that one can see that the mail contains the meeting's informations such as invited participants, title and date. That one can also see that the mail has two files attached which were added by the director to be used as relevant documentation before the meeting.

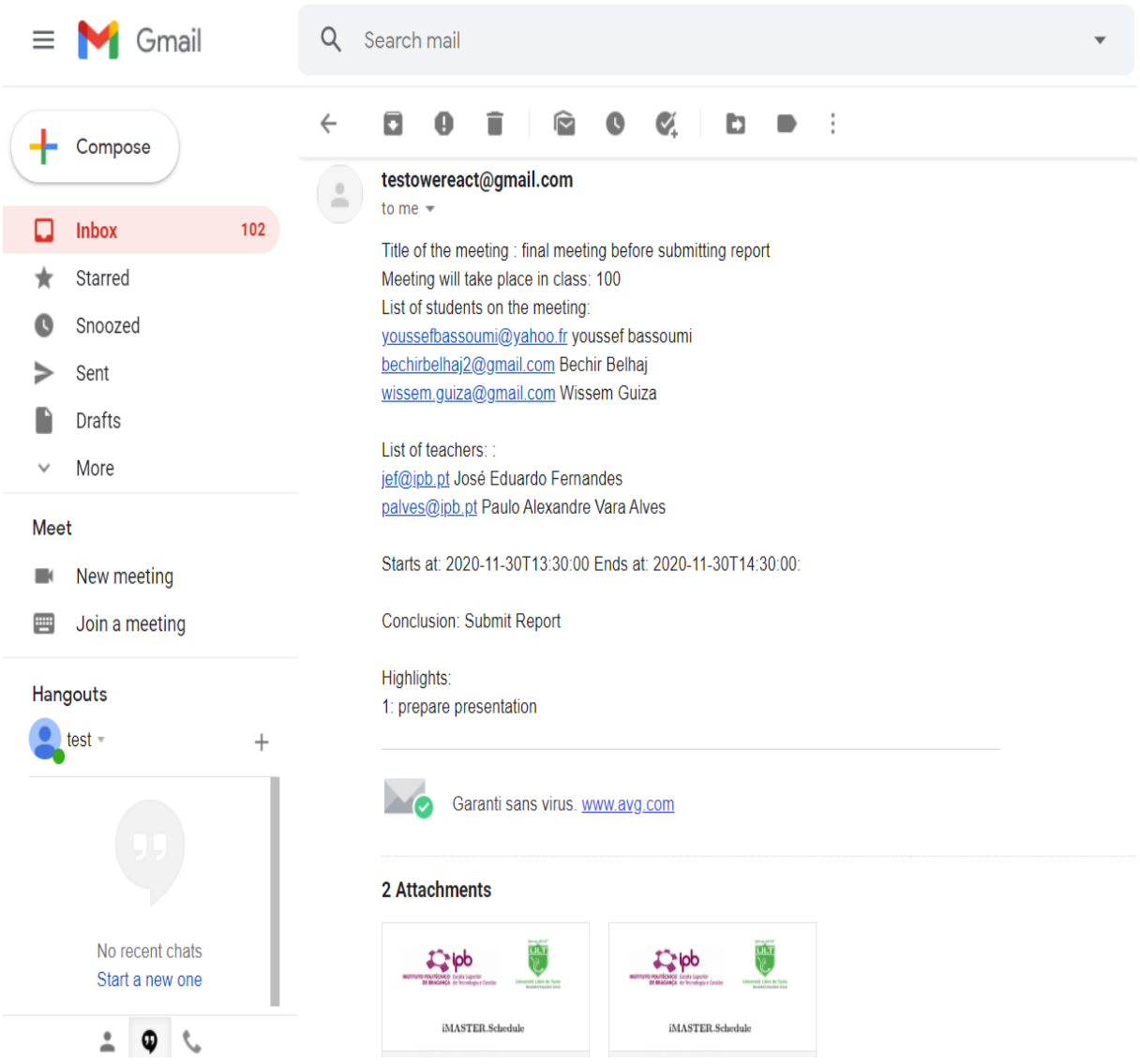


Figure 5.8: Interface of the E-mail

5.2.9 Interface of Add Student

By clicking on add student, the user has the ability to add a student or a group of students to the database. In order to do this, the user will fill the form with the name, matricule, and email of the students. He can also add the curricular units and specify if the student is in Master 1 or Master 2. If the student is in Master 2, the user has further informations to add such as the thesis title and the supervising professor. Alternatively if we have a group of students to add we can click on choose file to upload an excel file containing the relevant informations by using the XLSX library.

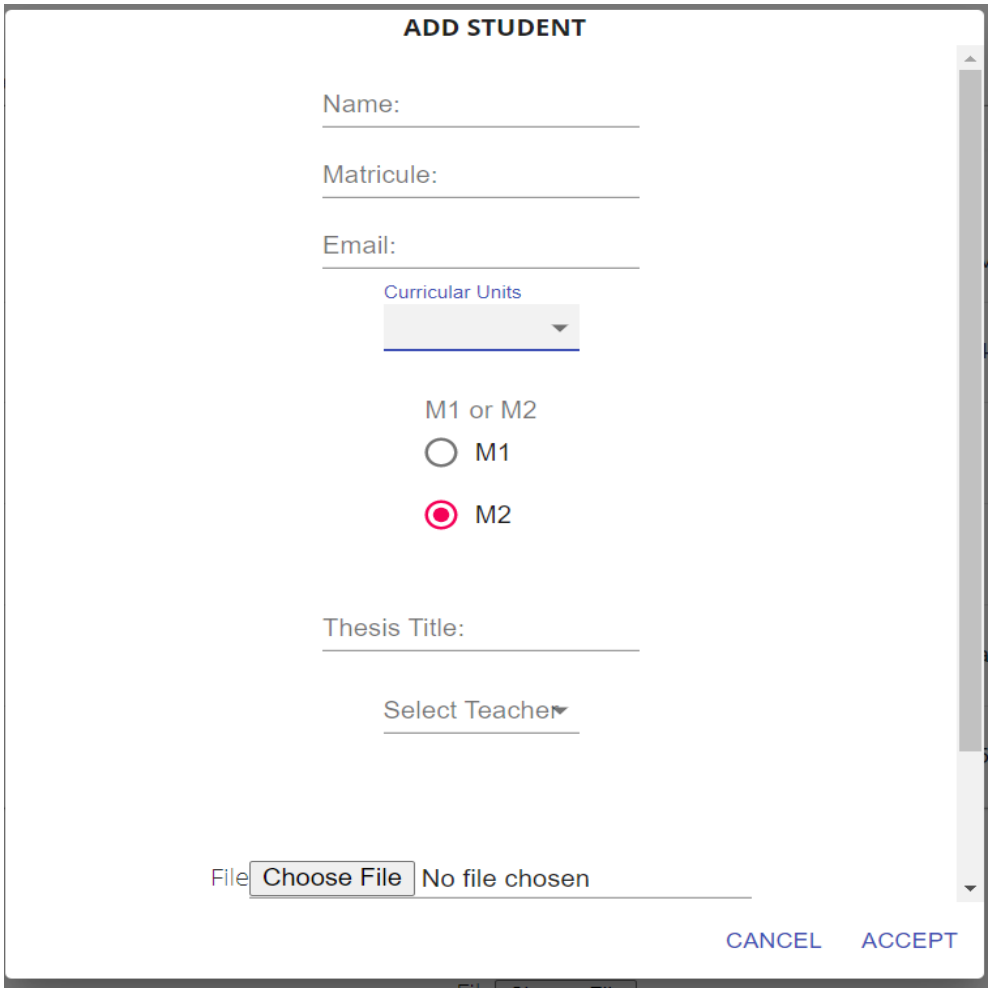


Figure 5.9: Interface of Add Student

5.2.10 Interface of choosing curricular Units

Another feature within the add student form is the option of adding the studied curricular units in the form of a list with checkable boxes which is the figure 5.10 below:

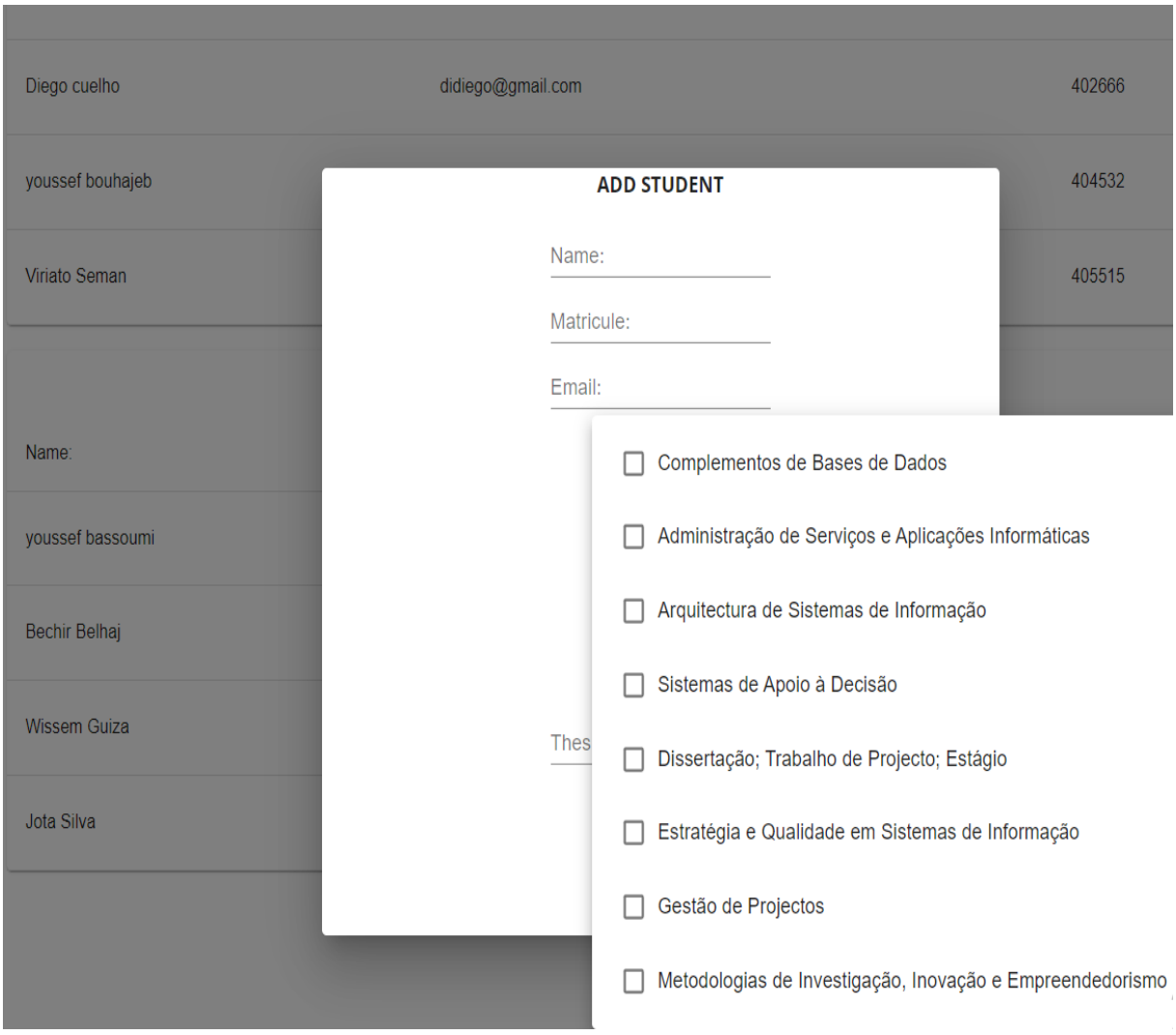


Figure 5.10: Interface of choosing curricular Units

5.2.11 Interface of The List of all student

By clicking on the list of all student button, the user can access the exhaustive list of all students. They will appear in two table devided by their level and that will show the name, email, and student number. This interface represents the list of all student which the director can consult as seen in the figure:

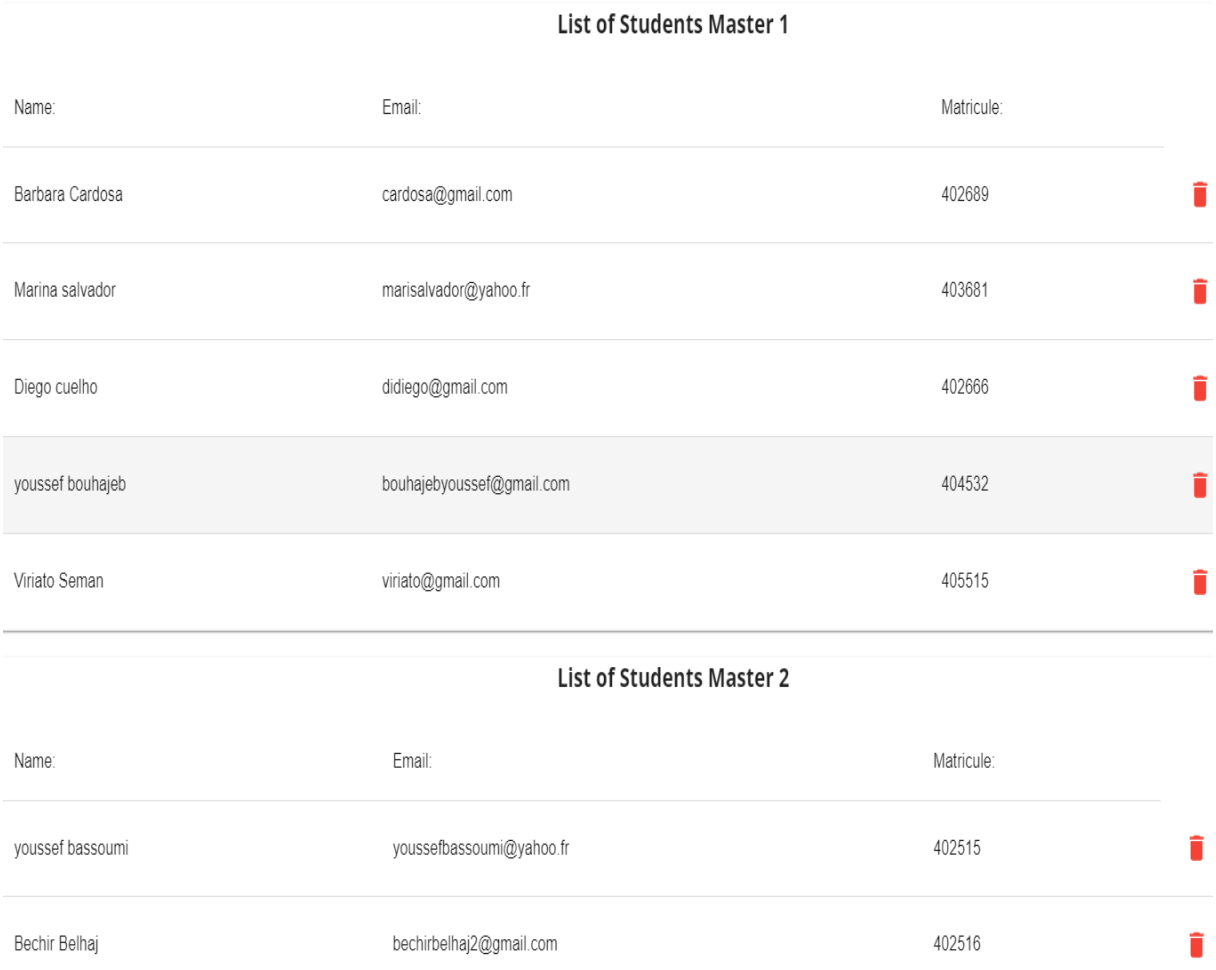


Figure 5.11: Interface of the List of all student

5.2.12 Interface of Student Information

By clicking on the student, the user can access to all the information, that will show the name, email, student number, also his thesis title and the name of his supervisor and the list of all the currucular units that he is studying. This interface represents the student information which the director can consult as seen in the figure:

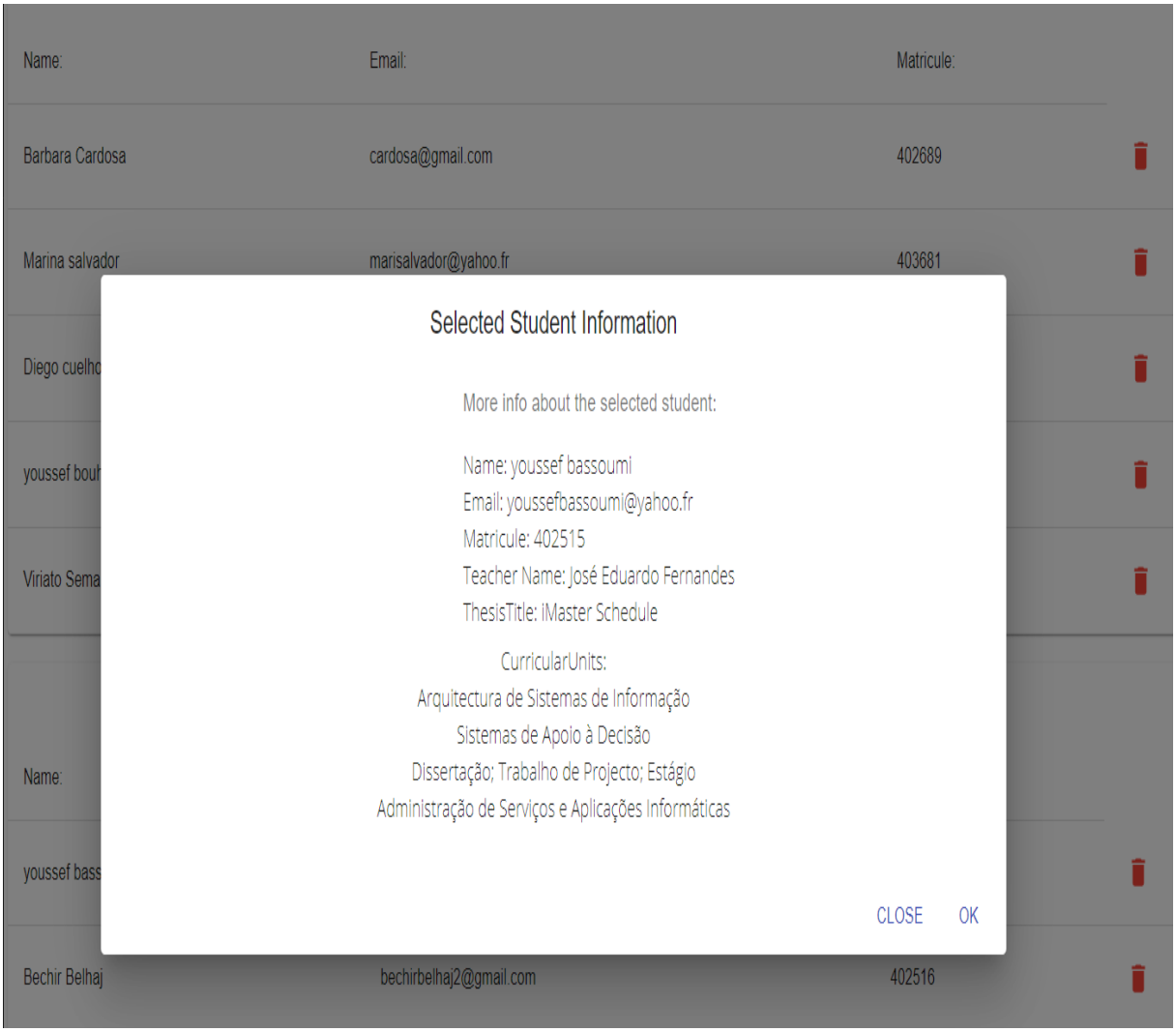


Figure 5.12: Interface of Student Information

Conclusion

In this chapter, one presenting the result of the solution through various interfaces and have briefly describing the contents of each interface to explain the main characteristics of each functionality as perceived by the user, indeed, one seeing all the functionalities offered by the solution both from front-end as well as back-end perspectives. All the interfaces remain modifiable thanks to the used technologies which offers and great potential for improvements in the future and leaves an open window for the addition of further functionalities.

Chapter 6

Conclusion

During this project, it was identified a need within the services proposed by the university to master director and studied the existing solutions to explore their functionalities and assess which one should be in the solution. Following that, one identified the main actor and his interactions with the solution and moved on to detail the structure and an architecture for the project which allowed us to then delve into a detailed exposition of the functionalities performed by "iMaster.Schedule".

After realising the project by going through the mentioned steps, one ended up with a solution that allows us to give the user access to the desired functionalities. Indeed the solution allows the master director to access the application securely through the authentication functionality, create and consult the three activities: events, meetings and jury meetings via his schedule, invite remotely any number of participants in the database, create a post-meeting detailing the meetings as well as attach relevant documents to be sent as references. Likewise, the user can add new students and professors to in order to keep his database always updated. The passive functionality of monitoring the human resources within the department is also guaranteed through the fact that the user can check the students and professor lists and consult further details about them such as curricular units, e-mails and even names of thesis projects.

Overall this project was an opportunity to learn a multitude of technologies as well as a lot about the inner functioning of the host university. This work can be further improved in a multitude of ways, like adding other actors to manage their schedule like students and teachers. Also adding the possibility for the user to make his meeting by a video call and the used technologies allow for such improvements in the future.

These improvements to the platform will make it extendable to reach, not just the master directors, but the students and professors as well which can have their own accounts and receive the invitations directly inside the platform. The current digitalization of most processes within universities is still an ongoing process since the used technologies always evolve and innovative and ingenious ways are permanently developed, that is why the proposed solution has a considerable scale up potential and can set a new and more efficient management process for time and human resources within the different departments.

Bibliography

[1] *Ipbvirtual*, https://virtual.ipb.pt/portal/site/~a42705/tool/5e0582f5-fb48-4219-a1b0-1d37f707fb71.

[2] *Google calender*, https://calendar.google.com/calendar/u/0/r.

[3] *Angular*, https://angular.io/guide/architecture.

[4] H. X. J Song's M Zhang, *Design and implementation of a vue.js-based college teaching system*, https://www.researchgate.net/publication/334468164_Design_and_Implementation_of_a_Vuejs-Based_College_Teaching_System, Jul. 2019.

[5] S. Shah, *React js— architecture + features + folder structure + design pattern*, https://saurabhshah23.medium.com/react-js-architecture-features-folder-structure-design-pattern-70b7b9103f22, Aug. 2020.

[6] A.Nigam, *Mounting and re-rendering components in react*, https://www.freecodecamp.org/news/how-to-understand-a-components-lifecycle-methods-in-reactjs-e1a609840630/, Mar. 2019.

[7] devmedia.com, *Laravel*, https://www.devmedia.com.br/laravel-tutorial/33173.

[8] developer.mozilla, *Django*, https://developer.mozilla.org/fr/docs/Learn/Server-side/Django/Introduction, Apr. 2020.

[9] *Tutorialspoint*, https://www.tutorialspoint.com/asp.net/index.htm, Jul. 2019.

[10] *Microservices*, https://skelia.com/articles/5-major-benefits-microservice-architecture//, Mar. 2018.

[11] xenonstack, *Micro services architecture*, https://www.xenonstack.com/insights/microservices/, Dec. 2018.

[12] R. Pijacek, *Sql server management studio*, https://skelia.com/articles/5-major-benefits-microservice-architecture, Jan. 2019.

[13] *Introduction to identity on asp.net core*, https://docs.microsoft.com/en-us/aspnet/core/security/authentication/identity?view=aspnetcore-5.0&tabs=visual-studio, 2020.

[14] Entityframeworktutorial, *Architecture of entity framework*, https://www.entityframeworktutorial.net/what-is-entityframework.aspx.

[15] thedotnetguide, *Mvc model*, https://thedotnetguide.com/mvc-design-pattern/.