



Web application for the Mentoring Academy program

Cassiano de Brito Andrade

Project work presented to the Escola Superior de Tecnologia e Gestão to obtain the Master's Degree in Information Systems under the dual diploma program with the Centro Federal de Educação Tecnológica de Minas Gerais.

Work carried out under the guidance of:

Prof. Doutor Paulo Alexandre Vara Alves (IPB)

Prof. Doutor José Eduardo Moreira Fernandes (IPB)

Prof. M.Sc. Flávio Roberto dos Santos Coutinho (CEFET-MG)

This project work does not include the criticisms and suggestions made by the Jury.

Bragança

February 2020



Web application for the Mentoring Academy program

Cassiano de Brito Andrade

Project work presented to the Escola Superior de Tecnologia e Gestão to obtain the Master's Degree in Information Systems under the dual diploma program with the Centro Federal de Educação Tecnológica de Minas Gerais.

Work carried out under the guidance of:

Prof. Doutor Paulo Alexandre Vara Alves (IPB)

Prof. Doutor José Eduardo Moreira Fernandes (IPB)

Prof. M.Sc. Flávio Roberto dos Santos Coutinho (CEFET-MG)

This project work does not include the criticisms and suggestions made by the Jury.

Bragança

February 2020

Abstract

Mentoring Academy is a program which aims to integrate students of Instituto Politécnico de Bragança in the academy through peer mentoring and peer tutoring. This program needed an application that permits its execution. Motivated by this demand, we aimed at proposing a web application that complies with the Mentoring Academy requirements. The system requirements were defined together with the program stakeholders. Based on the literature review, we opted for the technologies ASP.NET Core and MySQL, for the backend, and Angular 6, for the frontend. Subsequently, the system architecture was planned. SCRUM methodology was used in the application development, with biweekly meetings with the stakeholders to evaluate what was done and to define the efforts for the following fortnight. Eighteen features were developed and presented, including assigning mentors and tutors to new students, scheduling meetings, generating graphics with information about the students, notifications made by the system and importing new users. Finally, the system was deployed on a Linux server provided by the Escola Superior de Tecnologia e Gestão de Bragança, allowing it to be tested and later used by the Mentoring Academy users. Approximately 61% of the functional requirements defined in the system modeling were developed (44 out of 72) and some functionalities not previously defined were also implemented. The units tests developed for the web application proved promising in terms of components generation and HTTP request services, both exception-free. That said, the system provides students' integration, permitting them to have socio-academic inclusion and support in curricular units.

Keywords: web application; asp.net core; angular; mentoring; tutoring.

Resumo

Mentoring Academy é um programa que visa a integração de estudantes do ensino superior do Instituto Politécnico de Bragança por meio de mentoria e tutoria em pares. Esse programa necessitava de uma aplicação que viabilize sua execução. Motivado por essa demanda, o objetivo deste trabalho é propor uma aplicação web que atenda aos requisitos do Mentoring Academy. Em conjunto com os stakeholders do programa, os requisitos do sistema foram definidos. Com base na revisão bibliográfica, optou-se pelas tecnologias ASP.NET Core e Angular 6. O desenvolvimento da aplicação ocorreu nos moldes da metodologia SCRUM, com reuniões quinzenais com os stakeholders para avaliar o que foi feito e definir os esforços da quinzena seguinte. Dezoito funcionalidades foram desenvolvidas e apresentadas, dentre elas atribuição de mentores e tutores aos novos alunos, marcação de reuniões, geração de gráficos com informações sobre os alunos e importação de novos utilizadores. Por fim, o deploy do sistema ocorreu em um servidor Linux disponibilizado pela Escola Superior de Tecnologia e Gestão de Bragança, permitindo que ele fosse testado e posteriormente usado pelos utilizadores do Mentoring Academy. 44 dos 72 requisitos funcionais definidos na modelagem do sistema foram desenvolvidos. Também foram desenvolvidas algumas funcionalidades não definidas anteriormente. Os testes de unidade desenvolvidos para o aplicativo web mostraram-se promissores em termos de geração de componentes e serviços para requisições HTTP, ambos livres de exceções. Posto isso, o sistema fornece um meio de integração de estudantes, permitindo-lhes ter inclusão socio-académica e apoio em unidades curriculares.

Palavras-chave: aplicação web; asp.net core; angular; mentoria; tutoria.

Contents

1	Introduction	1
2	State of the Art	3
2.1	Peer mentoring and peer tutoring applications	3
2.2	Front-end frameworks	5
2.2.1	Angular 1	6
2.2.2	Angular 2	7
2.2.3	React	8
2.2.4	Vue	10
2.3	Back-end frameworks	11
2.3.1	ASP.NET and ASP.NET MVC	12
2.3.2	Ruby on Rails	13
2.3.3	Django	14
2.3.4	Laravel	15
2.4	Development tools and technologies definition	15
2.4.1	Front-end framework	16
2.4.2	Back-end framework	17
2.4.3	Other technologies	18
3	Analysis and Design	19
3.1	Mentoring Academy program	19
3.1.1	Peer mentoring	21

3.1.2	Peer tutoring	21
3.1.3	Problem	22
3.2	Development Methodology	23
3.3	Requirements Analysis	25
3.3.1	Requirements elicitation	25
3.3.2	Use case diagram	31
3.3.3	Class diagram	36
3.4	System architecture	39
4	Development of the Mentoring Academy web application	41
4.1	DevOps	41
4.2	Developed features	46
4.2.1	Request tutoring	49
4.2.2	Request mentoring	49
4.2.3	Apply for tutor role and remove application	50
4.2.4	Apply for mentor role and remove application	52
4.2.5	Send, read, update and delete emails	52
4.2.6	CRUD users and assign or remove user roles	54
4.2.7	CRUD meeting room	56
4.2.8	List meetings	58
4.2.9	Schedule a meeting	61
4.2.10	Update a meeting	62
4.2.11	Cancel a meeting	63
4.2.12	Reply a meeting invitation	64
4.2.13	Close a meeting	65
4.2.14	Assign mentor to mentee	66
4.2.15	Assign tutor to tutee	70
4.2.16	Graphs	71
4.2.17	System notification	73

4.2.18	Import new users	75
4.3	Unit tests	77
4.4	System deployment on a Linux server	80
4.4.1	Back-end publishing	80
4.4.2	Front-end publishing	80
4.4.3	MySQL configuration in the server	80
4.4.4	Database dump	81
4.4.5	ASP.NET Core prerequisites installation in the server	82
4.4.6	Deployment transfer to the server and execution of the ASP.NET Core application	82
4.4.7	Nginx installation and configuration	83
5	Conclusions	85
	References	88

List of Figures

2.1	Angular 1 two-way binding.	6
2.2	Angular 2 one-way binding.	8
2.3	Mounting and re-rendering components in React.	9
2.4	Vue.js data-driven concept.	10
2.5	Graph of the web framework popularity over the time based on GitHub and Stack Overflow scores.	12
3.1	Flowchart of the methodology adopted in the development of this work. . .	23
3.2	The Scrum method approach.	24
3.3	Sprint planning, duration and output.	25
3.4	Use case diagram focused on the common user persona.	32
3.5	Use case diagram focused on the administrator persona.	32
3.6	Use case diagram focused on the tutor and tutee personas.	33
3.7	Use case diagram focused on the mentor and mentee personas.	34
3.8	Use case diagram focused on the mentor, tutor and school coordinators personas.	35
3.9	Class diagram.	37
3.10	Proposed web application architecture for the Mentoring Academy system.	40
4.1	Azure DevOps home page which lists all the projects created.	42
4.2	Features created for the Mentoring Academy Azure DevOps project.	42
4.3	Sprint planning using the Azure DevOps.	43
4.4	Task card description of the Azure DevOps.	44

4.5	Branches created for the Mentoring Academy development in the Azure DevOps repository.	44
4.6	Organization of the commits in the Azure DevOps repository.	45
4.7	Sprints schedule for the Mentoring Academy web application development.	45
4.8	Screenshot of the Mentoring Academy landing page.	46
4.9	Screenshot of the Mentoring Academy internationalization functionality.	47
4.10	Screenshot of the Mentoring Academy home page after logging in.	48
4.11	Screenshot regarding the request tutoring functionality.	49
4.12	Screenshot regarding the request mentoring functionality.	50
4.13	Screenshot regarding the appliance for tutor role functionality.	51
4.14	Screenshot regarding the application removal for tutor role functionality.	51
4.15	Screenshot regarding the appliance and application removal for mentor role functionalities.	52
4.16	Screenshot regarding the sending a new email functionality.	53
4.17	Screenshot regarding the reading or deleting the emails sent functionalities.	54
4.18	Screenshot regarding the reading or deleting the emails received functionalities.	54
4.19	Screenshot regarding the users and roles update functionalities.	55
4.20	Screenshot regarding the activation and deactivation history of a user.	56
4.21	Screenshot regarding the meeting rooms listing functionality.	57
4.22	Screenshot regarding the meeting room creation functionality.	57
4.23	Screenshot regarding the meeting listing functionality.	58
4.24	Screenshot regarding the scheduled meetings report functionality.	59
4.25	Screenshot regarding a meeting evaluation functionality.	60
4.26	Screenshot regarding the close meeting functionality.	60
4.27	Screenshot regarding the schedule meeting functionality.	61
4.28	Screenshot regarding the schedule meeting functionality.	62
4.29	Screenshot regarding the meeting update functionality.	63
4.30	Screenshot regarding the meeting cancellation functionality.	64

4.31	Screenshot regarding the response of the meeting invitation functionality.	65
4.32	Screenshot regarding the meeting closure functionality.	66
4.33	Screenshot regarding the connection between mentors and mentees functionality.	67
4.34	Screenshot regarding the automatic connection between mentors and mentees functionality.	68
4.35	Screenshot regarding the assignment of mentees to a mentor manually.	68
4.36	Screenshot regarding the mentors assignment intensity table.	69
4.37	Screenshot regarding the summed up information about the mentees.	70
4.38	Screenshot regarding the connection between tutors and tutees functionality.	71
4.39	Screenshot regarding the graph about the assigned mentees.	72
4.40	Screenshot regarding the graph about the reunions per week.	72
4.41	Screenshot regarding the system notification about the meeting scheduling functionality.	73
4.42	Screenshot regarding the system notification about the meeting scheduling functionality.	73
4.43	Screenshot regarding the system notification about the meeting scheduling functionality.	74
4.44	Screenshot regarding the system notification about the meeting scheduling functionality.	74
4.45	Screenshot regarding the system notification about the meeting cancellation functionality.	75
4.46	Screenshot regarding the system notification about the user registration functionality.	75
4.47	Screenshot regarding the user importation functionality.	76
4.48	Unit testing of an Angular component.	77
4.49	Angular service which is responsible to make a http request to get all the languages registered.	78

4.50	Unit testing of the Angular service responsible to make a http request to get all the languages registered.	79
4.51	Running the unit tests developed for the Mentoring Academy web application.	79

Chapter 1

Introduction

Mentoring Academy program was created at the Polytechnic Institute of Bragança (IPB) to contribute to the integration of students entering higher education for the first time, as well as contributing to their academic and personal success [1].

This program counts on two coaching methods: (1) peer mentoring, which is composed of a mentor who shares experiences and helps a mentee to achieve a specific goal or entry effectively into a new environment, and (2) peer tutoring, which consists of tutors and tutees, who are also students, working together and teaching each other where the tutor helps the tutees in the learning process when they have questions.

The Mentoring Academy program was completely designed and, in order for it to be effectively launched and attend the IPB student's needs, it required an application that would permit its execution.

Motivated by this demand, we aimed at proposing a web application which complies with the Mentoring Academy requirements. Web applications deliver many benefits, for example: (1) easily accessible, since any computer or smartphone can be used to access it, (2) no installation required, since every computer or smartphone has a browser and (3) reachable by anyone anywhere in the world.

The methodology adopted for the development of this work is composed of four steps. The first step is about the system modeling, which contains the requirements elicitation,

class diagram and use case diagrams. The second step is responsible for the web application architecture definition which describes the interactions between user, client, web server, file system and database. The third step consists on the development of the Mentoring Academy web application. Finally, the fourth step describes the system deployment on a Linux server.

As a result, this work generated a web application for the Mentoring Academy program, which is a proposed solution for supporting the peer tutoring and peer mentoring coaching methods. The completion of this work permits the release of the Mentoring Academy program into the academic environment. That said, the system provides students' integration, permitting them to have socio-academic inclusion and support in curricular units.

This work is divided as follows: Chapter 2 addresses the state of the art regarding peer mentoring and tutoring applications, front-end frameworks, back-end frameworks and the development tools and technologies chosen for the web application development; Chapter 3 presents (a) the Mentoring Academy program, its main objectives and the problem which this work proposed to solve, (b) the methodology used for the development of this work, (c) the system specification, with requirements elicitation, employing use case diagrams and class diagram and (d) system architecture definition; Chapter 4 is about the development of the Mentoring Academy web application, how the methodology was used, the features and unit tests developed and the system deployment on a Linux server and, finally, Chapter 5 presents the conclusions of this work and possible future works.

Chapter 2

State of the Art

This chapter presents the state of the art regarding three different topics that will support the web application development for the Mentoring Academy program. Section 2.1 presents some recent works regarding peer mentoring and peer tutoring applications. Section 2.2 exposes the four most used front-end frameworks based on GitHub statistics and describes each one of them in terms of main goals, data binding and business applicability. Section 2.3 exhibits the five most popular back-end frameworks according to GitHub and Stack Overflow scores and a little description towards each one of them. Given the frameworks presented, Section 2.4 consists on the development tools and technologies definition. Therefore, the database, front-end framework and back-end framework were defined for the web application development and they are MySQL, Angular 6 and ASP.NET Core, respectively.

2.1 Peer mentoring and peer tutoring applications

Akobe, Popoola, Atayero, *et al.* [2] proposed an online peer tutoring application in which tutors are responsible for (1) accepting requests from tutees and in turn holding tutoring sessions for them and (2) uploading important documents to the platform for the tutees.

Spanorriga, Tsiotakis, and Jimoyiannis [3] designed and implemented an e-mentoring program for primary education teachers with low teaching experience. This application

had three main functionalities: (1) private chat between each mentor and mentee, (2) open-community space where any participant can contact one another and (3) closed group containing all the mentors so they can exchange experiences between them.

Wasilewski, Nonoyama, Dale, *et al.* [4] developed a peer mentoring training and web-based peer support program for ventilator-assisted individuals caregivers including information-sharing, peer-to-peer communication, and peer mentorship. The program includes (1) private chat, (2) a public discussion forum and (3) weekly moderated chats.

Clemmensen and Nørbjerg [5] developed a digital peer tutoring for shop floor workers. They aimed to develop capabilities among shop floor workers to use short videos to design and document solutions to operational and collaboration issues related to assistive technologies (collaborative robots), so others could watch their videos and learn from them

Thakare, Jadhav, Mane, *et al.* [6] designed the development of an online mentoring system to promote and encourage students to actively participate in the academic activities. They pointed out two features which are (1) the upload of an assignment by the mentor and (2) the upload of its answer by each mentee.

Evans and Moore [7] developed a web-based peer-tutoring system called Online Peer-Assisted Learning (OPAL) for problem-based undergraduate courses. OPAL included (1) questions in which if the students answered one correctly they would become tutors of that specific question, (2) registration of “tutoring ticket” by each tutor, containing his or her accessibility and readiness to teach the question, instruction history and duration of previous tutoring sessions, (3) video reflections from the tutors and tutees about a tutoring session and (4) answers from the students of a 5-point Likert scale about the tutoring sessions received.

Phiri, Meinel, and Suleman [8] proposed the design and implementation of a peer tutoring teaching platform aimed at facilitating the orchestration of tutor-led learning activities. This platform had two main features: (1) upload of documents for the tutees and (2) upload of assignments for the tutees.

Table 2.1 summarizes all the features encountered in the works cited above. These features will serve as a basis in Section 3.3.1 in order to define the web application requirements.

Table 2.1: Features encountered in the related works.

Code	Feature
F-01	Advisee can request session from a specific advisor.
F-02	Advisor can accept or refuse request from advisee for a session.
F-03	Advisor can upload files to the platform for the advisees.
F-04	Advisee can download files from the platform uploaded by the advisor.
F-05	Private chat between each advisor and advisee.
F-06	Open-community space where any participant can contact one another.
F-07	Closed group containing all the advisors so they can exchange experiences between them.
F-08	Weekly moderated chats.
F-09	Assignment upload by the advisor.
F-10	Assignment resolution from the advisee.
F-11	Advisoring guided per question.
F-12	Video upload by advisors and advisees containing feedback about a session.
F-13	Evaluation of sessions with a Likert scale.

2.2 Front-end frameworks

There are numerous front-end frameworks and libraries and Github’s usage statistics can reflect global front-end developers’ tendency on each front-end frameworks and libraries [9]. In order to find leading front-end frameworks and libraries under the industry standard, Xing, Huang, and Lai [9] collected usage data from Github, which is the largest Git-repository hosting service globally [9]. These data are exposed in Table 2.2.

Table 2.2: Github front-end frameworks usage statistics in May 2018.

	Angular 1	Angular 2	React	Vue
Downloads	1.4 million	2.6 million	9.2 million	1.5 million

Source: Xing, Huang, and Lai [9].

Therefore, the following sections will explore better each one of these frameworks so in Section 2.4, regarding development tools and technologies definition, the most suitable front-end framework for this work might be chosen.

2.2.1 Angular 1

Angular is a famous JavaScript ECMAScript 5 (ES5) based open source front-end web application framework and developed by Google in 2010 [10]. Angular's first development intention is assisting web designer in creating a persistent web form with more efficiency [10].

Angular 1's core concept is two-way data binding in web browsers reducing the back-end's data processing responsibility in web servers [10]. Figure 2.1 presents Angular 1 two-way binding.

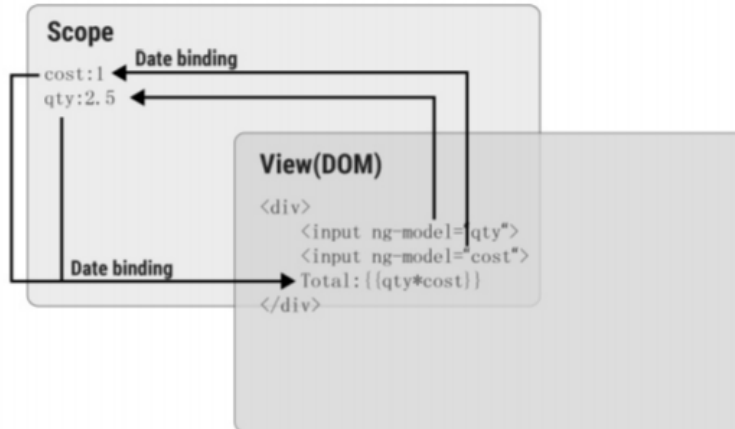


Figure 2.1: Angular 1 two-way binding.

Source: Omari, Erramdani, and Filali [11].

AngularJS is uncomplicated single-page application (SPA) framework with additional JavaScript codes that provides the application's data binding and responsive formula [12]. Angular 1 increases its usability, design visualization, user knowledge, and content development in designing the website or system wherein according to McLeod and Schell [13], the information is one of the five main types of resources [14].

AngularJS has main four disadvantages, which are (1) limitation in watchers (not more than 2000), (2) not built for mobile devices, (3) multiple ways to do the same things, which makes it hard for a developer to tell which way is the best and (4) two-way binding checks all the variables twice for updating which makes the user interface (UI) slow. That said, dynamic applications don't always perform that well in AngularJS; complex single-page applications (SPAs) can be laggy and inconvenient to use due to their size.

2.2.2 Angular 2

As the front-end history moves forward, Angular gradually transforms its position to fit into more development criteria, and web developers can develop more complex applications using Angular framework [9].

However, Angular 1 has many limitations due to its initial design model and is left behind other front-end frameworks in recent years [9]. In order to modernize Angular 1, its second version, called Angular 2, was released completely rewritten by Google development team in 2016 [15].

According to *Angular Official Blog AngularJS to Angular Concepts: Quick Reference* [16], Angular 2 brought four main changes: (1) Angular 2 discontinues template directive and controller and instead it uses a new module called "Component" to combine both parts; (2) Angular 2 adds event binding as one-way binding from view to component which is non-existent in Angular 1, like illustrated in Figure 2.3; (3) Angular 2 changes its language from "JavaScript Based" to "TypeScript Based" which is a strict syntactical superset of JavaScript developed by Microsoft; (4) Angular 2 uses zone.js instead of Scope to monitor interactive activities.

Therefore, Angular 2 team upgrades the concept that developers can use both one-way and two-way binding because choosing the hybrid method, in different circumstances, is the best solution to absorb one-way and two-way's advantages [9].

Consequently, Angular 2 provides the most suitable solution in data processing combined one-way and two-way binding [9]. Furthermore, its official technical support is

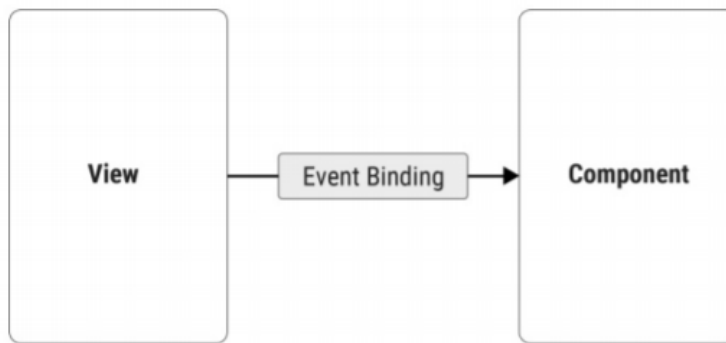


Figure 2.2: Angular 2 one-way binding.

Source: Xing, Huang, and Lai [9].

stable and reliable with the Google development team [9]. However, its volume is too enormous to play appropriate running performance due to its abundant functions, and its based language has limited communities [9]. Therefore, Angular 2 is suitable in large scale e-Business solutions which require complicated functions and sophisticated data processing method [9].

Angular 2 main disadvantages are (1) it is verbose and complex, (2) it has steep learning curve and (3) command-line interface (CLI) documentation lacks details. For instance, in Angular, one may need up to five files for a single component, have to inject dependencies and declare the component life-cycle interfaces.

2.2.3 React

In order to provide Facebook and Instagram website with better user experience, Facebook developed the React JavaScript library [17]. Due to React's powerful features, Facebook released it as an open source JavaScript ECMAScript 6 (ES6) based library to global developers and companies in 2013 [17]. Besides, Facebook also launched React Native to develop mobile applications with React under major mobile platforms such as iOS and Android in 2015 [18].

React's core engineering is creating a virtual Document Object Model (DOM) [18]. It creates a virtual DOM with one-way data binding, so when the user jumps to another

subpage, React creates an updated virtual DOM first then compares the difference between virtual DOM and displayed DOM [9]. After it summarizes the difference in each component, it will re-render the actual updated part, and the other uniform parts will not be re-rendered [9]. Figure 2.3 shows how the mounting and re-rendering happen in React.

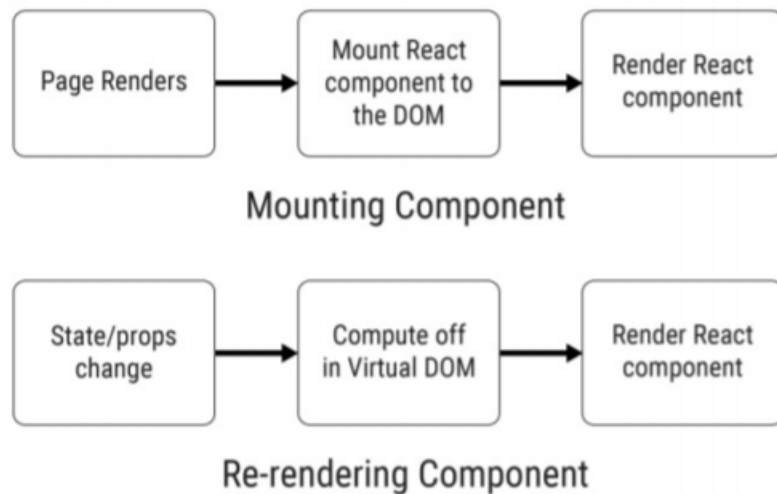


Figure 2.3: Mounting and re-rendering components in React.

Source: Eisenman [18].

React has high efficiency in rendering updated DOM and its powerful technical support and durable API allow developers to avoid update and migration concerns [9]. However, React is a JavaScript library and its volume is not the same as Angular 2 which means that React cannot provide comprehensive features and developers need to deploy by themselves [9]. Social media and communication applications usually require (1) more customized features and (2) functions with fast rendering speed, therefore they are the potential customers for React [9]. Therefore, React is used for dynamic web application development [12].

React is a technology that constantly changes, which makes the developers to relearn a new way of doing the same things. That said, it also prejudices the documentation, once the constant updates make the documentation remain poor.

2.2.4 Vue

Vue.js is a popular JavaScript ES6 based open source frameworks developed by Evan You in 2014 [19]. Vue's first development intention was to achieve responsive data binding and UI components with easy application programming interface (API) [19].

Although Vue is supposed to apply in a SPA, usually considered as limited functions and hard to adopt in commercial usage, the open source community provides strong third party supporting libraries and packages to support Vue to drive complex SPA with routing, state management and build tooling [19].

Figure 2.4 presents Vue.js data-driven concept, which illustrates: (1) View, which displays the website content, (2) View Model, which contains DOM listeners and data bindings and (3) Model, which represents real state content (plain JavaScript objects). Vue uses one-way binding with DOM listeners to achieve the result of two-way binding [9].

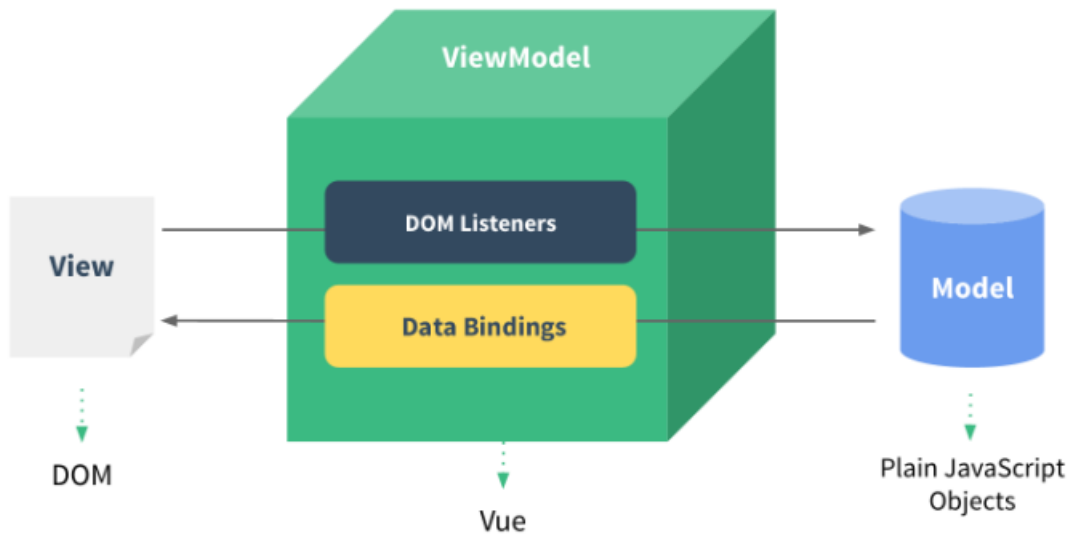


Figure 2.4: Vue.js data-driven concept.

Source: *Vue.js Official Blog Reactive Data Binding* [20].

Therefore, Vue provides both two-way and one-way binding option in data processing and its minimum volume occurs in the most efficient rendering and processing, compared

to Angular 2 and React [9]. Although Vue has its significant flexible superiority in front-end development, its technical support is not too reliable due to its development team scale limitation with unexpected official updated plans [9]. Furthermore, its smallest volume means that it includes the minimum features and considering various circumstances, Vue is suitable for small and medium web project which requires flexibility and simplified to develop with the fastest data processing speed [9].

Vue has two main disadvantages which are (1) closed community development, since it is not as popular as React or Angular and (2) lack of support for large projects, as of 2018, the development of the framework is not tied to corporate interests; therefore, any changes to the existing framework are mostly depended on the feedback of community members.

2.3 Back-end frameworks

HotFrameworks website can be adopted for researching the most popular frameworks nowadays, as it analyzes Stack Overflow and GitHub [21]. Therefore, Table 2.3 exhibits the five most popular frameworks according to *HotFrameworks Find your new favorite web framework* [22].

Table 2.3: Measuring web framework popularity based on GitHub and Stack Overflow scores.

Position	1	2	3	4	5
Framework	ASP.NET	Ruby on Rails	ASP.NET MVC	Django	Laravel

Source: *HotFrameworks Find your new favorite web framework* [22].

Furthermore, Figure 2.5 presents a graph of the web framework popularity over the time, from second quarter 2015 to first quarter of 2019, based on GitHub and Stack Overflow scores. It is possible to notice that ASP.NET is the irrefutable leader, having the maximum score during the whole period, while the other frameworks fluctuated over the time.

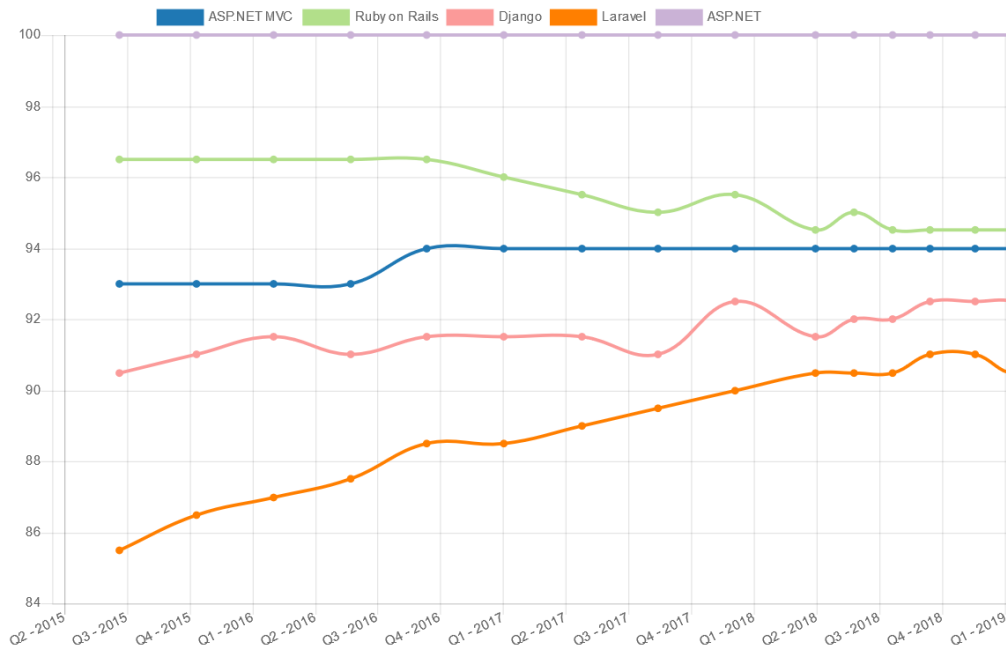


Figure 2.5: Graph of the web framework popularity over the time based on GitHub and Stack Overflow scores.

Source: *HotFrameworks Find your new favorite web framework* [22].

Therefore, the subsequent sections will explore better each one of these frameworks so in Section 2.4, regarding development tools and technologies definition, the most suitable back-end framework for this work might be chosen.

2.3.1 ASP.NET and ASP.NET MVC

ASP.NET (Active Server Pages .NET) is a web application framework that is developed by Microsoft and utilises the Common Language Runtime (CLR) [23]. The features provided are similar to Java EE, for example, Web Forms allows creating dynamic content, in

a similar fashion to JSP and JSF [24]. A number of components, notably ASP.NET Web API, ASP.NET AJAX, ASP.NET MVC provide various functions [25]. ASP.NET is typically run on Internet Information Services (IIS), which is not supported on GNU/Linux by Microsoft; thus, Kestrel must be used instead [26].

ASP.NET MVC 5 was released in 2013 [27]. ASP.NET MVC is a Web development framework from Microsoft that combines the effectiveness and tidiness of Model-View-Controller (MVC) architecture, the most up-to-date ideas and techniques from agile development, and the best parts of the existing ASP.NET platform [28]. It's a complete alternative to traditional ASP.NET Web Forms, delivering considerable advantages for all but the most trivial of Web development projects [28]. In terms of operating system, ASP.NET MVC is cross-platform.

The ASP.NET MVC Framework implements the MVC pattern and, in doing so, provides greatly improved separation of concerns. In fact, ASP.NET MVC implements a modern variant of the MVC pattern that is especially suitable for Web applications [28]. By embracing and adapting the MVC pattern, the ASP.NET MVC Framework provides strong competition to Ruby on Rails and similar platforms, and brings the MVC pattern into the mainstream of the .NET world [28].

ASP.NET, when compared to open source alternatives, is expensive, as you have expenses like SQL Server licenses, Visual Studio licenses, Windows server licenses. In addition, porting ASP application from one server to another is expensive, once the configuration settings in an ASP web application is obscure, and stored in the IIS metabase.

2.3.2 Ruby on Rails

The first open source release of Rails was in July 2004 with commit rights to the project following soon after [29]. Rails uses the Model-View-Controller (MVC) programming pattern to organize application programming so that it is easier to work with [29]. *Ruby on Rails Official Blog Ruby on Rails Guides (v5.0.0.1)* [30] provides numerous guides to working with Rails and getting started as a new developer to the framework.

Rails is based mainly on two principals: convention over configuration and don't repeat yourselves (DRY), resulting in higher productivity [31]. By applying MVC and working in tune with the HTTP protocol instead of against it, by promoting conventions instead of the need for configuration, and by integrating an object-relational mapping (ORM) tool into its core, Rails applications more or less fell into place without much effort [28]. In terms of operating system, Ruby on Rails is multi-platform.

Ruby on rails has four main disadvantages, which are (1) runtime speed, it is slow when compared to the runtime speed of NodeJS or GoLang, (2) boot speed, depending on the number of gem dependencies and files, it can take a significant amount of time to start, which can hinder developer performance, (3) documentation can be hard to find, particularly for the less popular gems and for libraries which make heavy use of mixins and (4) multi-threading, this means if one is not careful, requests will get queued up behind the active request and can introduce performance issues.

2.3.3 Django

Django was created in the fall of 2003 when programmers for the Lawrence Journal-World newspaper began using Python to build applications [29]. In the summer of 2005, the developers decided to release the framework as open source software and named it after the jazz guitarist Django Reinhardt [29]. Since 2008, the newly formed Django Software Foundation continued maintaining Django [32].

In terms of documentation, Django provides well documented starter guides on their site so developers can begin learning the framework and, aside from this, it is easier to obtain specific help related to the Python language itself [29]. In terms of operating system, Django is cross-platform.

Django has 3 main disadvantages, which are (1) it is monolithic, since it has a certain way to define and perform tasks, so one may need to spend more time on an unique project, (2) it is not for small projects, since all the functionalities of Django come with lots of code, therefore it takes server's processing and time, which poses some issues for low-end

websites which can run on even very little bandwidth, and (3) it uses regular expression for URLs, since Django uses RegEx to determine its URL routing models which makes the code bigger and makes twisted syntaxes.

2.3.4 Laravel

The Laravel framework relies on Hypertext Preprocessor (PHP) as its scripting (programming) language [33]. Laravel has an expressive syntax that provides solutions for development and facilitates general task in big web projects [34]. Laravel is a clear, simple, elegant, and well-documented framework that focuses on equipping and enabling developers [33]. It helps developers learn, start, and develop quickly [35]. The Laravel framework is useful when there are issues of time, security, and complexity [33].

According to Poudel [33], (1) Ubuntu 17.10 operating system is required to develop web applications based on Laravel which are written in the PHP programming and (2) Nginx (for deploy) and Apache (for development) are the web servers. Additionally, in terms of operating system, Laravel is cross-platform.

Laravel has five main disadvantages, which are (1) being a new framework, it is not as mature as many other frameworks, (2) the composer is not so strong as compared to npm (for node.js) or ruby gems and pip (python), (3) community support is not widespread compared to other platforms, (4) many methods included in the reverse routing process are complex and (5) it is not easy for legacy systems to get transferred to Laravel.

2.4 Development tools and technologies definition

In order to develop a web application, two frameworks are necessary: one for the front-end and another for the backend. The front-end framework permits to create the web application interface and all elements that make up a site's visual identity. The back-end framework is responsible for the web application gears, creating the code responsible for the website performance and requests. The front-end and back-end frameworks chosen

for the development of the Mentoring Academy system will be discussed in Sections 2.4.1 and 2.4.2.

2.4.1 Front-end framework

Stefankrause Developer Team Results for js web frameworks benchmark [36] performed a benchmark with some front-end frameworks, including Angular 1, Angular 2, React and Vue. Table 2.4 shows the results of this experiment. “Ready memory” means the quantity of memory usage after page load and “run memory” means the quantity of memory usage after adding 1000 rows, both in megabytes. Larger volume means the framework or library contains more features and functions, but it will spend more time loading [9].

Table 2.4: Front-end frameworks and library memory allocation performance with standard deviation in a benchmark.

Activity	Angular 1	Angular 2	React	Vue
ready memory	4.9 +/- 0.2 (1.5)	4.8 +/- 0.0 (1.4)	4.5 +/- 0.1 (1.3)	3.8 +/- 0.0 (1.1)
run memory	13.0 +/- 0.1 (3.2)	10.9 +/- 0.1 (2.7)	9.7 +/- 0.1 (2.4)	7.5 +/- 0.1 (1.9)

Source: *Stefankrause Developer Team Results for js web frameworks benchmark* [36].

After analyzing the comparison data between Angular 1, Angular 2, React and Vue in various aspects include data binding, language-based, technical support, volume and performance, Xing, Huang, and Lai [9] concluded that (1) Angular 2 contains the most comprehensive functions and features that suitable for enormous commercial projects, especially in e-Business, and (2) React and Vue are suitable to live stream, communication, blog and small or medium scale applications.

The differences from Angular 2 to Angular 6 are very small, once the main changes were made between Angular 1, also known as AngularJS, and the new Angular (2-6). After that, the differences between all Angular versions are insignificant to the general angular developer.

The main differences between them are (1) bundle size in Angular 6 is reduced and

load time of application is also improved, (2) Angular 2 has typescript 2.1 support and Angular 6 has latest typescript 2.7+ and (3) Angular 2 doesn't have support for native element and Angular 6 does.

Therefore, the front-end framework chosen for the development of the web application for this work is Angular 6.

2.4.2 Back-end framework

The Mentoring Academy web application shall be used for a very long time once it is released. Therefore, it would be appropriate to choose a back-end framework that is being used widely, meaning that it has a large amount of available information which helps on the development and maintenance processes. Thereby, according to *HotFrameworks Find your new favorite web framework* [22], ASP.NET would be that framework, once it has maximum scores on both GitHub and Stack Overflow in terms of demand.

However, the Mentoring Academy web application is supposed to be deployed on a Linux server and ASP.NET runs only on Windows. For that reason, it would be important to have the ASP.NET knowledge, background and maintenance potential associated with the capability of running on Linux.

ASP.NET Core is the next generation of ASP.NET [26], cross-platform .NET framework for building modern cloud-based web applications on Windows, Mac or Linux developed by Microsoft and community contributors [37].

As one of web application frameworks, ASP.NET Core can scaffold back-end services for web API, identity and database [38]. It also supports security features such as authentication, authorization, data protection, Secure Sockets Layer (SSL) enforcement, app secrets, anti-request forgery protection, and cross-origin resource sharing (CORS) management [39].

Furthermore, with the release of .NET Core in 2016 and, subsequently, the ASP.NET Core [40], Microsoft is supporting more operating systems [26]. Now, there is a first-party CLR implementation available on GNU/Linux, in addition to a modern rewrite of

ASP.NET and a new web server – Kestrel [41]. Hereupon, the back-end framework chosen for the development of the web application for this work is ASP.NET Core.

2.4.3 Other technologies

After choosing the front-end and back-end frameworks, it is time to define other technologies: (1) integrated development environment (IDE), (2) source control tool and (3) database. Firstly, as for the IDE, it was chosen Microsoft Visual Studio, an integrated Microsoft development environment for software development that is specifically dedicated to the .NET Framework and C# language.

For the development of this work, Azure DevOps Server was used for version control and project management. Secondly, git was used for source control, a distributed version control system used in software development and to track changes in any set of files. Thirdly, as for the database, it was chosen MySQL, an open source relational database management system mostly used in free applications to manage databases.

Chapter 3

Analysis and Design

This chapter is divided in four parts. Section 3.1 presents the Mentoring Academy program and the problem which this work proposed to solve. Section 3.2 exposes the development methodology adopted. Section 3.3 presents the requirements analysis, with requirements elicitation, use case diagrams and class diagram. Finally, Section 3.4 exhibits the system architecture.

3.1 Mentoring Academy program

Transitioning from high school to college can be a formidable challenge, especially for students who are the first in their family to attend college (first-generation) or are from low-income backgrounds [42].

An effective strategy is to learn from competent, relatable models, people who can draw on their own experiences to help first-generation students build a sense of self-efficacy and achievement [43] without making them feel stigmatized [42].

Two known coaching methods that can help novice students on their academic life are peer mentoring and peer tutoring.

On the one hand, as defined by Spanorriga, Tsiotakis, and Jimoyiannis [3], peer mentoring is considered as an intense interpersonal relationship and a process that brings together a senior individual (mentor) and a younger, less skilled or experienced person

(mentee), by sharing experiences and helping the latter to achieve a specific goal or entry effectively into a new environment.

On the other hand, peer tutoring is a unique teaching technique; it consists of two or more students working together and teaching each other rather than learning from a teacher's direct instruction [44]. A broad definition given in [2] stated that "peer tutoring involves two pupils: a tutor and a tutee. The students educate one another usually in minute groups. These gatherings are cautiously sorted out and monitored by an educator or coordinator in a classroom setting".

Peer tutoring has a lengthy history of success in classrooms [45], and it is an effective strategy which is successful in improving student academic performance [46]. Peer tutoring has also been said to improve student's self-esteem [47].

With this intention of helping novice students on their academic life, the Mentoring Academy program was created in the IPB to contribute to the integration of students entering higher education for the first time, as well as contributing to their academic and personal success [1].

Hereupon, this program has eight main objectives which are: (1) promote the adaptation of students to higher education and, in particular, to the School of Technology and Management of Bragança (ESTiG); (2) contribute to the student's socio-academic inclusion; (3) guide to the most appropriate choices, identifying the constraints and difficulties that may jeopardize academic success and facilitate the integration of students; (4) recognize and counter as early as possible, in the academic course, situations of academic failure; (5) identify students' needs to reach their full academic potential; (6) contribute to the promotion of quality teaching and academic development; (7) increase the bi-directional flow of information between school and students and (8) contribute to equal opportunities for students.

In order to achieve these objectives, the Mentoring Academy program counts on the two coaching methods: peer mentoring and peer tutoring. Sections 3.1.1 and 3.1.2 will present how the program designed each approach, and Section 3.1.3 will expose the problem regarding the Mentoring Academy program which this work proposed to solve.

3.1.1 Peer mentoring

Peer mentoring aimed at the socio-academic integration of the first-time student joining the Instituto Politécnico de Bragança, who will be accompanied by a mentor who will assist him/her in adapting to a new reality.

Therefore, it is destined to the first year students who will be assigned to a mentor who will help them to integrate into the school and the city and will accompany them since their enrollment. The mentor can (1) provide valuable information and answer to the questions the mentees may have about the school, the course and the social work services, (2) help in the integration into the social and cultural environment within the city and (3) advise about the challenges presented through the academic route and about possible need for academic support in some curricular units.

The program foresees that the mentor and the mentees will meet at least biweekly but, if the stakeholders deem it necessary, it can be more often. Timetables of the meetings must be articulated between both parts. In case someone is unable to attend the meeting, he/she must notify the concerned parties and schedule a new meeting.

Mentors should keep record of their meetings, containing the content discussed in each session, and this record shall be shared with the mentor team leader or the program coordinators. There will be at least two semiannual meetings between the mentors, the mentor team leader and the program coordinators to identify potential problems.

Throughout the year, there will be actions, aligned to the program's objectives, organized by the program coordinators, mentors or mentor team leader, which may be useful for the identified needs.

3.1.2 Peer tutoring

Peer tutoring aimed at supporting, accompanying and advising a tutor who will help students, from any year but especially those from the first year, to achieve academic success by overcoming their difficulties.

It will be developed with the voluntary support of students, active teachers, retired

teachers and other collaborators who meet the pedagogical conditions to tutor students who voluntarily applied for this initiative.

Tutors will support these students in overcoming their difficulties in the curricular units, advising on study methods and eventually assisting with pedagogical support in solving exercises, problems or other activities performed within the curricular unit which the student presents difficulties. Accordingly, these actions may contribute to (1) the students academic and social success, (2) increase their motivation and (3) reduce academic dropout.

The tutoring sessions are held at least once a week in small groups of minimum three and maximum six tutees, lasting from 60 to 90 minutes in the provided space. Whenever convenient, there may be the intervention of a senior tutor (an active or retired teacher or senior student). Attendance and punctuality are expected from both tutors and tutees. Whenever someone is unable to attend the scheduled session, he/she should notify in due time and schedule a new session.

Sessions are opened by the tutors and confirmed by the tutees. Tutors should keep a record of the sessions. Therefore, the tutors should give a report to the senior tutor about (1) the sessions and (2) the tutees anytime requested. Regular meetings will be held between tutors, senior tutors and the coordination team at least twice a semester to evaluate the program.

3.1.3 Problem

The Mentoring Academy program was completely designed and, in order for it to be effectively launched and attend the IPB students needs, it required an application that permits its execution. Motivated by this demand, we aimed at proposing a web application which complies the Mentoring Academy requirements.

For the purpose of developing a good web application for the Mentoring Academy program, the following section exposes the methodology adopted for the development of this work.

3.2 Development Methodology

The methodology used for the development of this work is composed of four steps, as presented in Figure 3.1.

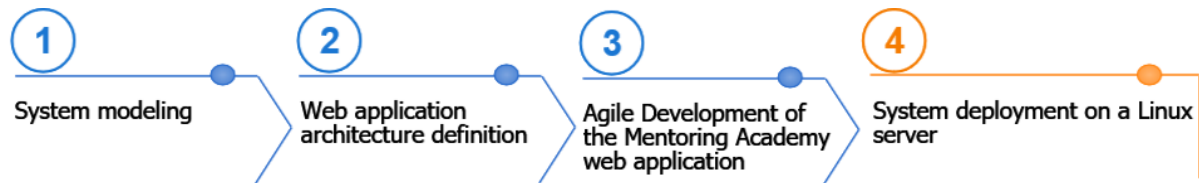


Figure 3.1: Flowchart of the methodology adopted in the development of this work.

The **first step** is about the system modeling, which contains the requirements elicitation, class diagram and use case diagrams. Altogether, 72 functional requirements were defined together with the Mentoring Academy stakeholders [1]. Then the class and use case diagrams were designed in order to attend all the functionalities defined.

The **second step** is responsible for the web application architecture definition which describes the interactions between user, client, web server, file system and database.

The **third step** consists on the development of the Mentoring Academy web application. As defined by Permana [48], agile software development is one of the methodologies for software development and the word agile means to be fast, lightweight, free-moving and alert. Agile is a word used to describe a process model concept which is different from the existing process model concepts [49].

In agile software development, the interactions and the individuals are more important than the process and the tools, a running software is more important than a complete documentation, collaboration with the clients is more important than a contract negotiation, and being responsive to changes is more important than following a plan [48].

Scrum was developed by Jeff Sutherland in 1993 and its goal is to become a development and management methodology that follows the principles of agile methodology [50].

Scrum is a software development process for small teams which consists of an initial planning phase and a series of short development phases, called sprints, for delivering

the product incrementally [51]. It is a framework where teams can (1) solve complex problems that are constantly changing and (2) produce high-value products creatively and productively [52].

Figure 3.2 describes the Scrum method using an iterative and incremental approach to improve predictability and risk control [52]. Basically the iteration is composed by the cycle (1) prioritization of tasks, (2) detailed requirements of the prioritized tasks, (3) design and analysis of the workforce during the iteration period, (4) implementation and developer testing of the tasks, (5) final testing based on the quality assurance and (6) deployment of the new functionalities.

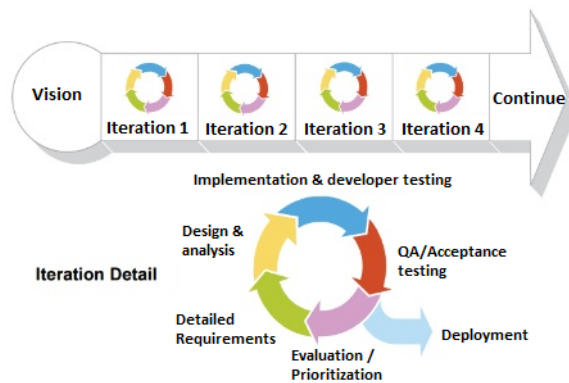


Figure 3.2: The Scrum method approach.

Source: Sutherland and Sutherland [53].

An iteration, or a sprint, happens usually every fifteen days and its backlog is a small part of the whole product backlog, as illustrated in Figure 3.3.

The development methodology used was Scrum, with biweekly sprints and a total of 11 sprints. Altogether, 44 out of 72 functional requirements previously defined were developed and some functionalities that weren't previously defined were also developed. Unit tests were developed using Angular 6 in order to ensure that Angular components and Hypertext Transfer Protocol (HTTP) request services of the application met its design and behaved as intended.

Finally, the **fourth step** describes the system deployment on a Linux server, which was composed of seven steps: (1) back-end publishing, (2) front-end publishing, (3) MySQL



Figure 3.3: Sprint planning, duration and output.

Source: Permana [48].

configuration in the server, (4) database dump, (5) ASP.NET Core prerequisites installation, (6) deployment transfer to the server and execution of the ASP.NET Core application and (7) Nginx installation and configuration.

3.3 Requirements Analysis

The web application system modeling in this work is divided into three parts: (1) requirements elicitation, (2) use case diagrams and (3) class diagram. Section 3.3.1 exhibits the nine identified personas and the functional requirements regarding each persona. Given the functional requirements, Section 3.3.2 presents the use case diagrams designed to comply all the requirements. Finally, Section 3.3.3 exhibits the class diagram representing how each element of the system will interact with each another.

3.3.1 Requirements elicitation

The requirements elicitation begins with the identification of all the web application personas. Therefore, nine personas were identified: (1) common user, (2) administrator, (3) tutor, (4) tutee, (5) mentor, (6) mentee, (7) school coordinator, (8) mentor coordinator and (9) tutor coordinator.

The first persona, the common user, is a student that can request tutoring and mentoring, apply himself/herself for tutor and mentor roles, participate in activities, lectures or

workshops and use the e-mail. Table 3.1 exhibits all the functional requirements regarding the common user persona.

Table 3.1: Functional requirements of the web application regarding the common user persona.

Code	Description
FR-01	A common user shall be able to request tutoring.
FR-02	A common user shall be able to request mentoring.
FR-03	A common user shall be able to apply himself/herself for tutor role.
FR-04	A common user shall be able to remove his/her application for tutor role.
FR-05	A common user shall be able to apply himself/herself for mentor role.
FR-06	A common user shall be able to remove his/her application for mentor role.
FR-07	A common user shall be able to update personal user information.
FR-08	A common user shall be able to reply presence in activity, lecture or workshop.
FR-09	A common user shall be able to send, read, update and delete emails.
FR-10	A common user shall be able to receive messages sent to email group.

The second persona, the administrator, is a person responsible for the settings adjustments concerning users, courses, schools, curricular units, school coordinators, email groups, meeting rooms and roles. Table 3.2 presents all the functional requirements regarding the administrator persona.

Table 3.2: Functional requirements of the web application regarding the administrator persona.

Code	Description
FR-11	An administrator shall be able to activate or deactivate users.
FR-12	An administrator shall be able to create, read, update and delete courses.
FR-13	An administrator shall be able to create, read, update and delete schools.
FR-14	An administrator shall be able to create, read, update and delete curricular units.
FR-15	An administrator shall be able to create, read, update and delete school coordinators.
FR-16	An administrator shall be able to create, read, update and delete email group type.
FR-17	An administrator shall be able to create, read, update and delete meeting room.
FR-18	An administrator shall be able to assign or remove user roles.

The third persona, the tutor, is responsible for managing his/her meetings with the tutees, replying meeting invitations sent by his/her coordinator and providing relevant material to the tutees. The fourth persona, the tutee, is responsible for replying meeting invitations sent by the tutor. Table 3.3 exhibits all the functional requirements regarding the tutor and tutee personas.

Table 3.3: Functional requirements of the web application regarding the tutor and tutee personas.

Code	Description
FR-19	The tutor shall be able to schedule a meeting with the tutees.
FR-20	The tutor shall be able to update a meeting with the tutees.
FR-21	The tutor shall be able to list the meetings scheduled by him/her.
FR-22	The tutor shall be able to cancel a meeting scheduled by him/her.
FR-23	The tutor shall be able to close a past meeting by submitting a report.
FR-24	The tutor shall be able to obtain a tutoring participation certificate.
FR-25	The tutor shall be able to reply a meeting invitation sent by the tutor or school coordinator.
FR-26	The tutor shall be able to provide material to the tutees.
FR-27	The tutee shall be able to reply a meeting invitation sent by the tutor.
FR-28	The tutee shall be able to report a meeting feedback.
FR-29	The tutee shall be able to close a meeting by submitting a report.
FR-30	The tutee shall be able to consult the material made available by the tutor.

The fifth persona, the mentor, is in charge of managing his/her meetings with the mentees, replying meeting invitations sent by his/her coordinator and providing relevant material to the mentees. The sixth persona, the mentee, is in charge of replying meeting invitations sent by the mentor. Table 3.4 presents all the functional requirements regarding the mentor and mentee personas.

The seventh and eighth personas, the school and mentor coordinators, share some common responsibilities, which are opening the mentoring matriculation, assigning a mentor to each mentee, manage the mentor role, issue the mentoring participation certificate for the mentors and schedule meetings with the mentors. Table 3.5 exhibits all the common functional requirements regarding both personas.

Table 3.4: Functional requirements of the web application regarding the mentor and mentee personas.

Code	Description
FR-31	The mentor shall be able to schedule a meeting with the mentees.
FR-32	The mentor shall be able to update a meeting with the mentees.
FR-33	The mentor shall be able to list the meetings scheduled by him/her.
FR-34	The mentor shall be able to cancel a meeting scheduled by him/her.
FR-35	The mentor shall be able to close a past meeting by submitting a report.
FR-36	The mentor shall be able to obtain a mentoring participation certificate.
FR-37	The mentor shall be able to reply a meeting invitation sent by the mentor or school coordinator.
FR-38	The mentor shall be able to provide material to the mentees.
FR-39	The mentee shall be able to reply a meeting invitation sent by the mentor.
FR-40	The mentee shall be able to report a meeting feedback.
FR-41	The mentee shall be able to close a meeting by submitting a report.
FR-42	The mentee shall be able to consult the material made available by the mentor.

Table 3.5: Functional requirements of the web application regarding the school and mentor coordinators personas.

Code	Description
FR-43	The school and mentor coordinators shall be able to open mentoring matriculation.
FR-44	The school and mentor coordinators shall be able to assign mentor to mentee manually.
FR-45	The school and mentor coordinators shall be able to assign or remove mentor role.
FR-46	The school and mentor coordinators shall be able to automatically assign mentor to mentee.
FR-47	The school and mentor coordinators shall be able to issue mentoring participation certificate.
FR-48	The school and mentor coordinators shall be able to list meetings between mentors and mentees.
FR-49	The school and mentor coordinators shall be able to schedule a meeting with mentors.

The school and tutor coordinators also share some common responsibilities, which are opening the tutoring matriculation, assigning a tutor to each tutee, manage the tutor role, issue the tutoring participation certificate for the tutors and schedule meetings with the tutors. Table 3.6 presents all the common functional requirements regarding the school and tutor coordinators personas.

Table 3.6: Functional requirements of the web application regarding the school and tutor coordinators personas.

Code	Description
FR-50	The school and tutor coordinators shall be able to open tutoring matriculation.
FR-51	The school and tutor coordinators shall be able to assign tutor to tutee manually.
FR-52	The school and tutor coordinators shall be able to assign or remove tutor role.
FR-53	The school and tutor coordinators shall be able to automatically assign tutor to tutee.
FR-54	The school and tutor coordinators shall be able to issue tutoring participation certificate.
FR-55	The school and tutor coordinators shall be able to list meetings between tutors and tutees.
FR-56	The school and tutor coordinators shall be able to schedule a meeting with tutors.

There are some functional requirements that concern the three coordinators, which are (1) send message to email group, (2) manage event and news, (3) issue certificate of attendance to activity, lecture and workshop, (4) manage activity, lecture and workshop and (5) update and list meetings scheduled by him/her. Table 3.7 exhibits all the common functional requirements regarding the school, mentor and tutor coordinators personas.

Table 3.7: Functional requirements of the web application regarding the school, mentor and tutor coordinators personas.

Code	Description
FR-57	The school, mentor and tutor coordinators shall be able to send message to email group.
FR-58	The school, mentor and tutor coordinators shall be able to create, read, update or delete event or news.
FR-59	The school, mentor and tutor coordinators shall be able to issue activity, lecture or workshop attendance certificate.
FR-60	The school, mentor and tutor coordinators shall be able to create, read, update or delete activity, lecture and workshop.
FR-61	The school, mentor and tutor coordinators shall be able to confirm presences in activity, lecture or workshop.
FR-62	The school, mentor and tutor coordinators shall be able to update meeting scheduled by him/her.
FR-63	The school, mentor and tutor coordinators shall be able to cancel meeting scheduled by him/her.
FR-64	The school, mentor and tutor coordinators shall be able to list meetings scheduled by him/her.
FR-65	The school, mentor and tutor coordinators shall be able to close meeting scheduled by him/her.

Table 3.8 presents the functional requirements regarding only the school coordinator persona, which are basically about the management of the tutor and mentor coordinator roles.

Table 3.8: Functional requirements of the web application regarding the school coordinator persona.

Code	Description
FR-66	The school coordinator shall be able to assign or remove the mentor coordinator role.
FR-67	The school coordinator shall be able to assign or remove the tutor coordinator role.

Finally, there are some functional requirements regarding the system, which is responsible for notifying the interested participants about (1) session scheduling, (2) session cancellation, (3) mentoring certificate issuance, (4) tutoring certificate issuance and (5) activity, lecture or workshop certificate issuance. Table 3.9 exhibits all the functional requirements regarding the system.

Table 3.9: Functional requirements of the web application regarding the system.

Code	Description
FR-68	The system shall be able to notify session scheduling to interested parties.
FR-69	The system shall be able to notify session cancellation to interested parties.
FR-70	The system shall be able to notify when a mentoring certificate is issued.
FR-71	The system shall be able to notify when a tutoring certificate is issued.
FR-72	The system shall be able to notify when an activity, lecture or workshop certificate is issued.

3.3.2 Use case diagram

The use case diagram was divided into seven parts for readability. Therefore, each part focus on up to two personas: (1) common user, (2) administrator, (3) tutor and tutee, (4) mentor and mentee, (5) mentor coordinator, (6) tutor coordinator and (7) school coordinator.

Figure 3.4 presents the first part which is focused on the common user persona. The functional requirements approached in this part of the diagram are the ones listed on Table 3.1 (from FR-01 to FR-10). As shown by the figure, in order to require mentoring the common user must be a first grader, whereas to apply for mentor role he/she must be at least at sophomore year.

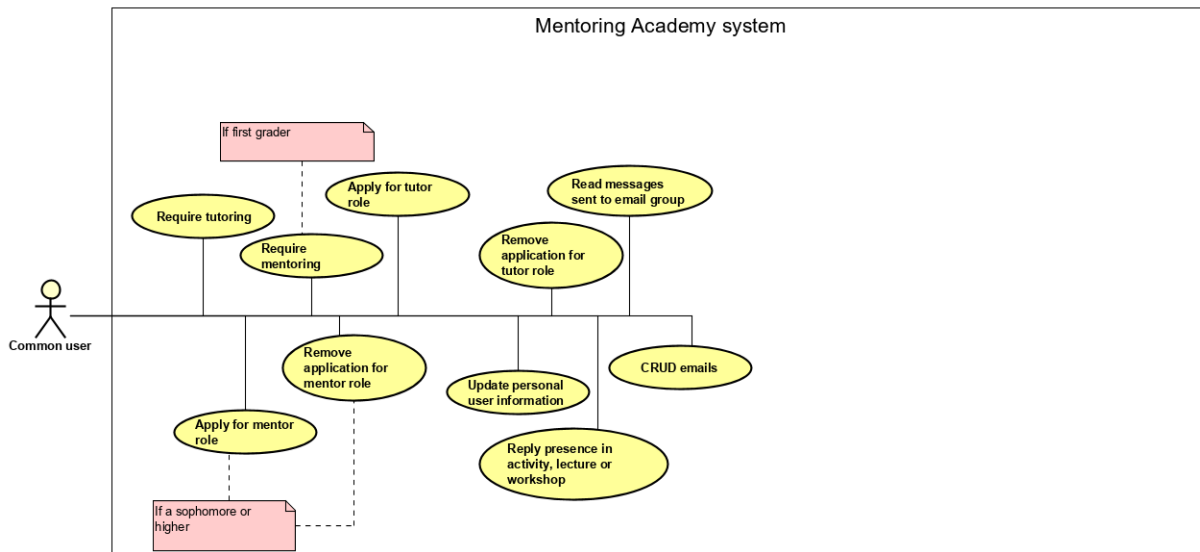


Figure 3.4: Use case diagram focused on the common user persona.

Figure 3.5 exhibits the second part which is focused on the administrator persona. The functional requirements approached in this part of the diagram are the ones listed on Table 3.2 (from FR-11 to FR-18). The administrator will be responsible for the essential create, read, update and delete (CRUD) operations and for the user roles updating.

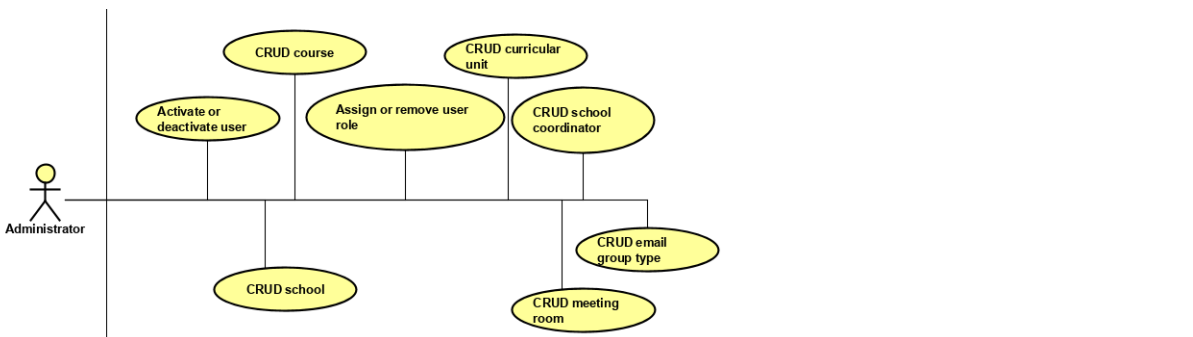


Figure 3.5: Use case diagram focused on the administrator persona.

Figure 3.6 presents the third part which is focused on the tutor and tutee personas. The functional requirements approached in this part of the diagram are the ones listed on Table 3.3 (from FR-19 to FR-30) and two from Table 3.9 regarding meeting scheduling (from FR-68 to FR-69). Whenever the tutor schedules or updates a meeting, the system is responsible for notifying the interested parties.

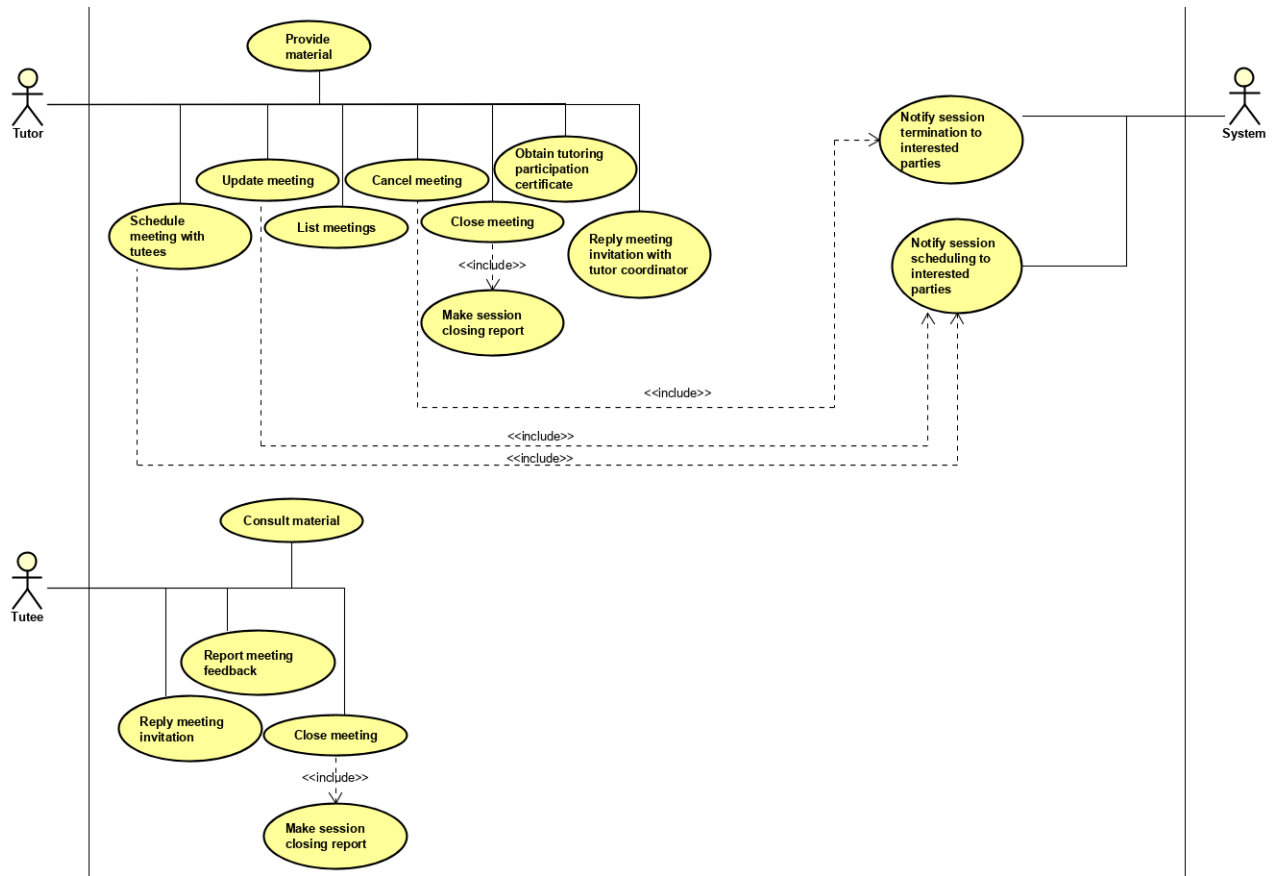


Figure 3.6: Use case diagram focused on the tutor and tutee personas.

Figure 3.7 exhibits the fourth part which is focused on the mentor and mentee personas. The functional requirements approached in this part of the diagram are the ones listed on Table 3.4 (from FR-31 to FR-42) and two from Table 3.9 regarding meeting scheduling (from FR-68 to FR-69). As well as the tutor, the system is responsible for notifying the interested parties about a meeting scheduling or updating by the mentor.

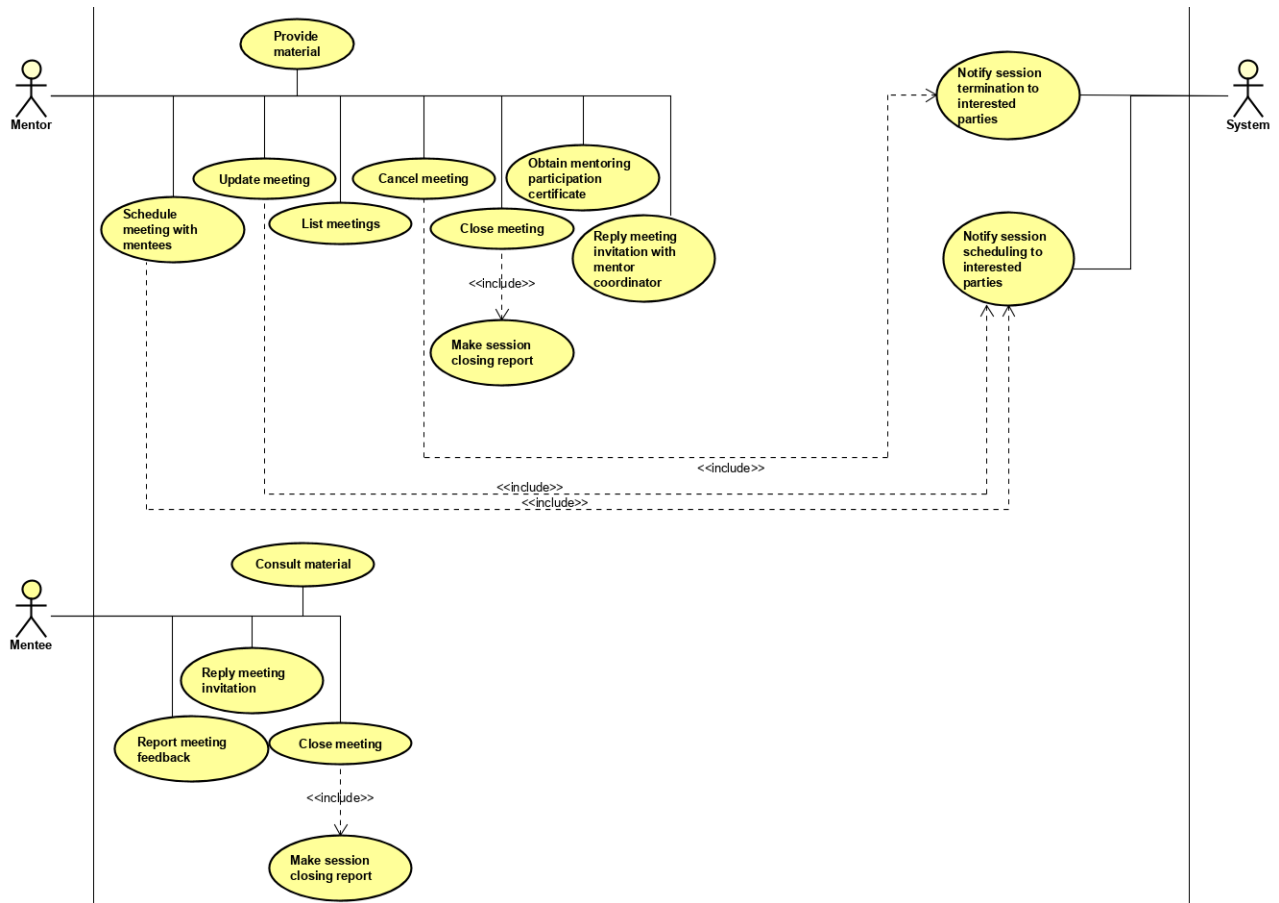


Figure 3.7: Use case diagram focused on the mentor and mentee personas.

Figure 3.8 presents the last three parts which regard the mentor, tutor and school coordinators personas. Regarding the mentor coordinator, the functional requirements approached in this part of the diagram are the ones listed on Tables 3.5 and 3.7 (from FR-43 to FR-49 and from FR-57 to FR-65) and four from Table 3.9 regarding meeting scheduling and certificate issuance (FR-68, FR-69, FR-70 and FR-72).

The CRUD operations on event or news regard the title, description, external links and attachments. Whenever the mentor coordinator schedules or updates a meeting, the system is responsible for notifying the interested parties.

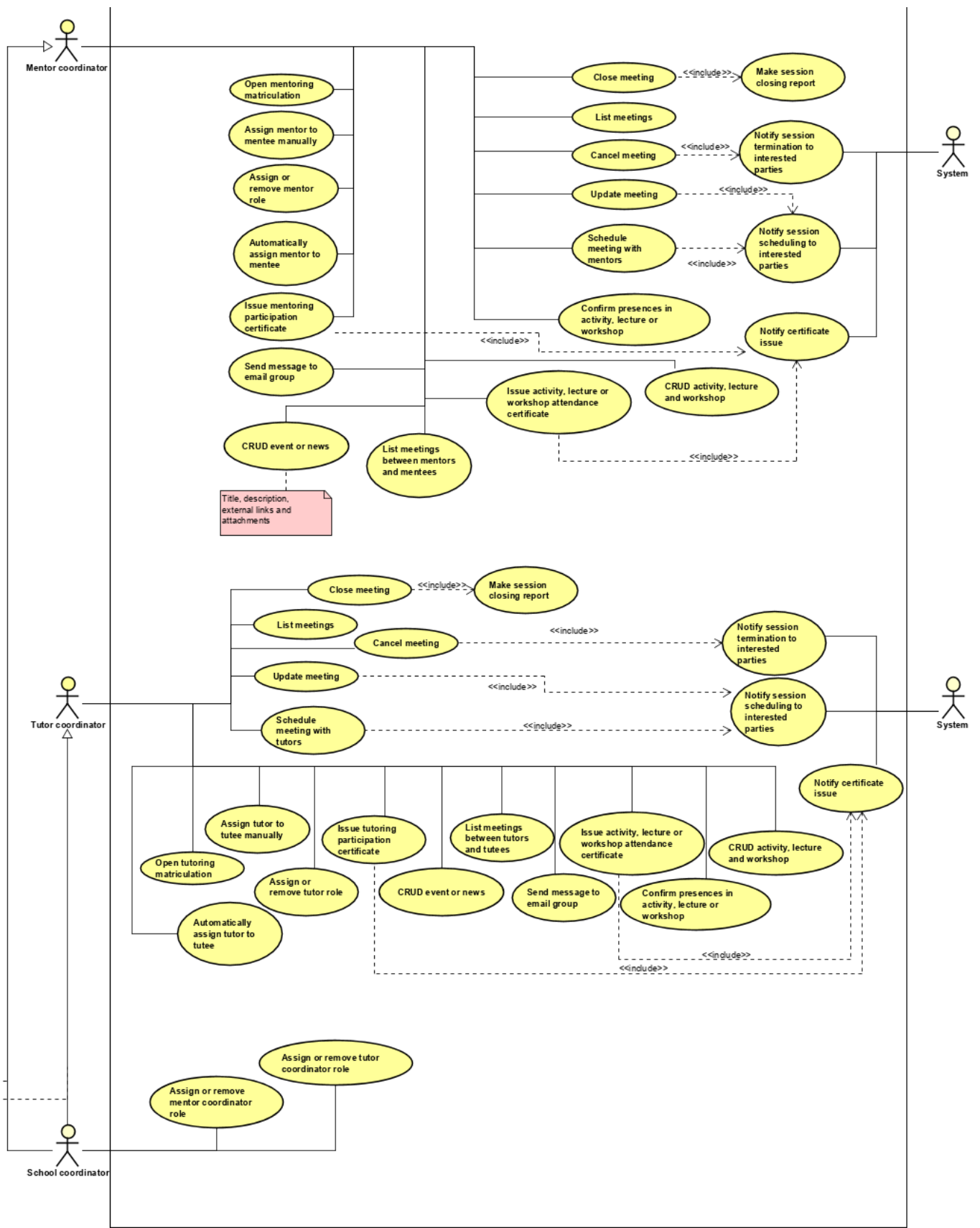


Figure 3.8: Use case diagram focused on the mentor, tutor and school coordinators personas.

Regarding the tutor coordinator, the functional requirements approached in this part of the diagram are the ones listed on Tables 3.6 and 3.7 (from FR-50 to FR-65) and four from Table 3.9 regarding meeting scheduling and certificate issuance (FR-68, FR-69, FR-71 and FR-72).

As well as for the mentor coordinator, the tutor coordinator is also responsible for the CRUD operations on event or news and the system is responsible for notifying the interested parties about a meeting scheduling or updating by him/her.

Regarding the school coordinator, the functional requirements approached in this part of the diagram are the ones listed on Table 3.8 (from FR-66 to FR-67). The school coordinator has the same powers as the mentor and tutor coordinators, inheriting all of their use cases. In addition, the school coordinator is responsible for assigning and removing the tutor and mentor coordinator roles.

3.3.3 Class diagram

The class diagram of the web application, presented by Figure 3.9, describes the structure of the system by showing the system's classes, their attributes and the relationships among objects.

The **AspnetUsers** is the class responsible to represent the users registered in the system. A user can be associated with multiple courses (**AspnetUserCourses**) and he/she can be an IPB employee (**Employees**) but not necessarily. Additionally, the user is associated with one school (**Schools**) and can speak multiple **Languages** registering each one through the **AspnetUserLanguages** class association.

CurricularUnits are associated to a course (**Courses**) and a course is associated to a school (**Schools**) and a degree (**Degrees**). These four classes and **Employees** have each one a correspondent class which is responsible for providing a translation for each object name into each available language (**Languages**) in the system.

The **AspnetUserRoles** class is the one accountable for containing the multiple roles an user might have. The roles are represented by the class **AspnetRoles** and each role

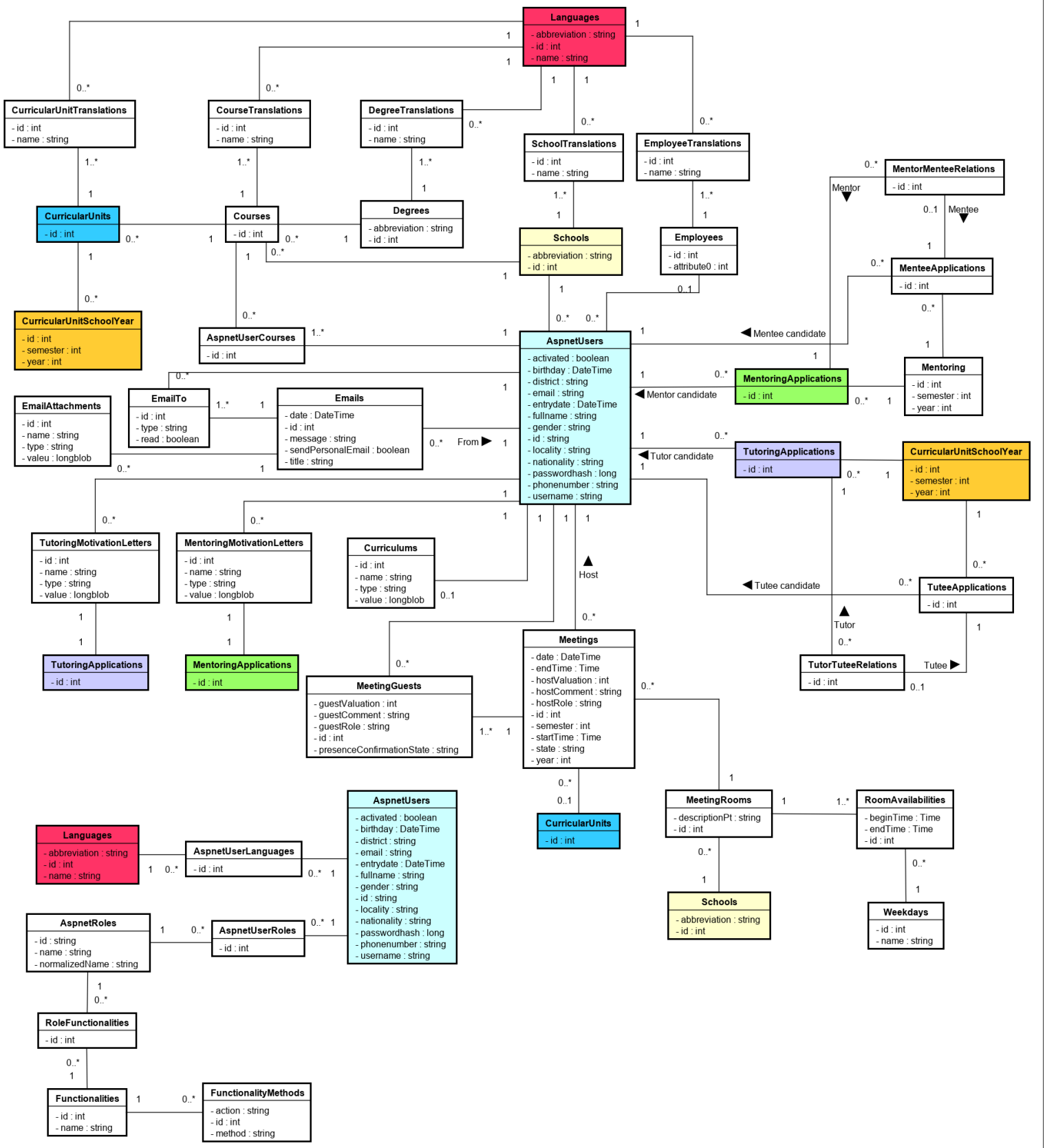


Figure 3.9: Class diagram.

might be associated with multiple `Functionalities` and therefore multiple methods (`FunctionalityMethods`). The classes `Functionalities` and `FunctionalityMethods` are the ones responsible to give an user authorization to access any application functionality.

An user can apply for mentoring and tutoring multiple times through the `MentoringApplications` and `TutoringApplications` classes. A mentoring application is necessarily associated to a semester and a year (`Mentoring`) whereas the tutoring is to a semester, a year and a curricular unit (`CurricularUnitSchoolYear` and `CurricularUnit`).

On the one hand, in order to apply for tutoring, the user must have a curriculum registered in the `Curriculums` class and a motivation letter in the `TutoringMotivationLetters` class. On the other hand, in order to apply for mentoring, the user must have just a motivation letter in the `MentoringMotivationLetters` class. A curriculum is unique per user, whilst the motivation letter is unique per application.

The mentees are registered through the `MenteeApplications` class and each mentee has a mentor through the `MentorMenteeRelations` class. Similarly, the tutees are registered through the `TuteeApplications` class and each tutee has a tutor through the `TutorTuteeRelations` class.

Scheduling a meeting requires the `Meetings` class which has all the core information about the meeting and `MeetingGuests` which contains the information about each guest's evaluation and comments. A meeting must have a room (`MeetingRooms`) which has start and end times according to the room availability registered in `RoomAvailabilities`.

An user can send multiple emails, involving the `Emails` class, and each email must have at least one remittee and can have multiple attachments, involving the `EmailTo` and `EmailAttachments` classes respectively.

3.4 System architecture

The web application architecture describes the interactions between applications, databases and middleware systems on the web. As any typical web application, there are two different codes (sub-programs) running side-by-side: (1) client-side code, which is the code that is in the browser and responds to user inputs and (2) server-side code, which is the code that is on the server and responds to the HTTP requests.

The model chosen for the Mentoring Academy web application components is the “one web server, one database”, which is the simplest as well as the least reliable web app component model. Such a model uses a single server as well as a single database.

The web application architecture chosen is the SPAs, which allows dynamic interactions by means of providing updated content to the current page instead of loading the entire page after any user interaction. SPAs are designed in a way so that they request for most necessary content and information.

Figure 3.10 illustrates the interactions between user, client, web server, file system and database. It begins when the user interacts with the interface over the browser starting a HTTP request. This request is sent to the web server, which resolves it consulting the file system and the database and replies a HTTP response.

This HTTP response may contain static files (HTML, CSS, JavaScript and images) or data from the database. The static files served by the file system are generated by the Angular’s deployment.

The authentication process is responsible for returning a JSON Web Token (JWT) containing the following data: (1) username, (2) user identification and (3) the roles the user possesses. Once received this token, every request made by the client shall contain this token as part of the authorization field.

In the server, this token will be used to identify who is the user responsible for the request and his/her roles. As explained in the Section 3.3.3, each role can grant access to multiple functionalities. Therefore, the server can check if the logged in user is authorized to access the requested function.

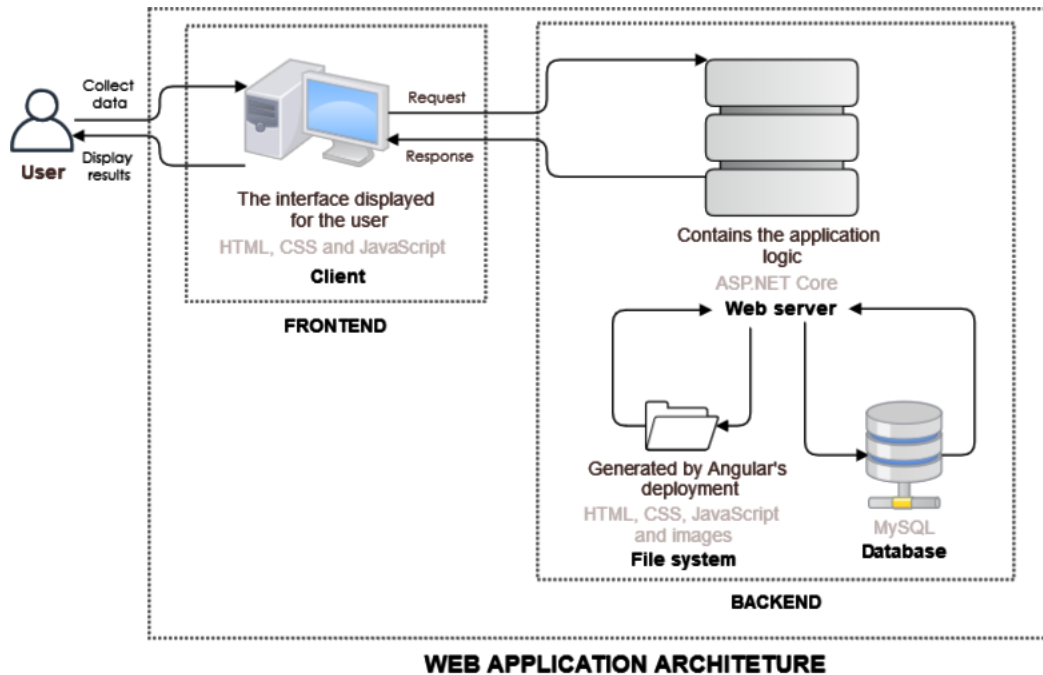


Figure 3.10: Proposed web application architecture for the Mentoring Academy system.

Consequently, each server function may contain two authorization methods: (1) token authorization and (2) role authorization. The first one will verify if the token is still valid and the second one will verify if the user can access the function based on his/her roles.

Additionally, the frontend shall be responsible for presenting on its interface only the functionalities which the logged in user can have access to, in order to avoid unauthorized requests.

For the web server, it was used the Entity Framework (EF) Core as an object-relational mapper, permitting to generate a model from the existing database any time the database is update. In addition, Language-Integrated Query (LINQ) was used to build queries in a simple way by its unified syntax.

Once defined the web application architecture, the next chapter presents the development process of the Mentoring Academy web application.

The following chapter presents the development of the Mentoring Academy system process, containing the methodology adopted, the features developed, the unit tests created and the deployment process description.

Chapter 4

Development of the Mentoring Academy web application

The development of the Mentoring Academy web application started on June 4th of 2019 and ended on January 7th of 2020, using Scrum as the development methodology. Altogether, eleven sprints were held lasting fifteen days each. Section 4.1 presents the Scrum methodology and how it was used for the development of the Mentoring Academy application, Section 4.2 exhibits all the developed features, Section 4.3 presents the unit tests developed for this work and Section 4.4 exhibits the system deployment process.

4.1 DevOps

The development of this work was made using the Azure DevOps online tool. It was used for (1) features mapping, (2) sprint planning and (3) version control. Figure 4.1 exhibits the project created there for the Mentoring Academy development.

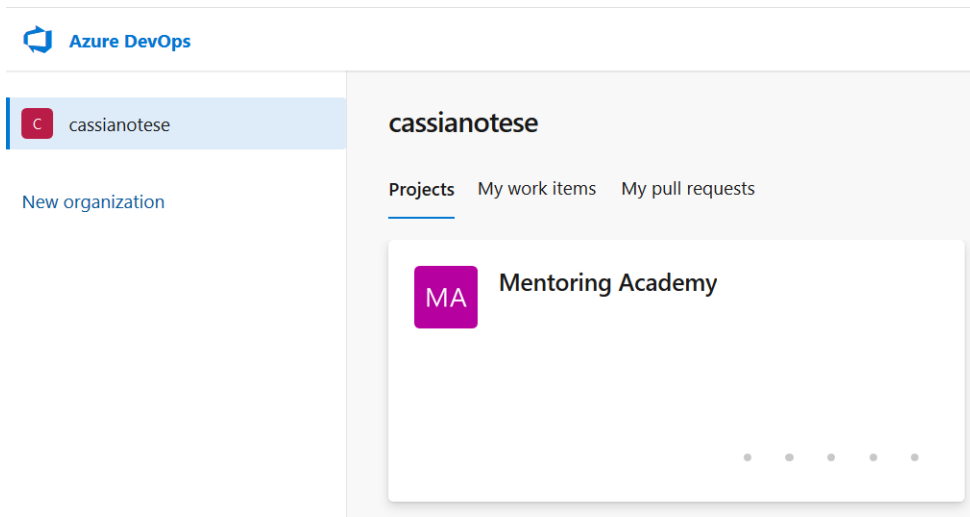


Figure 4.1: Azure DevOps home page which lists all the projects created.

Azure DevOps has a board in which we can register all the system's features, like shown by Figure 4.2. Once we define the tasks, we can link each task to the feature which it is associated to. In the figure, it is possible to observe each feature's name and its respective persona(s).

cassianotese / Mentoring Academy / Boards / Backlogs

Search

Mentoring Academy Team

Backlog Analytics + New Work Item View as Board Column Options

Order	Work Item Type	Title	Tags
1	Feature	Register users	Common user
2	Feature	Apply for tutor role	Common user
3	Feature	Request tutoring	Common user
4	Feature	CRUD email	Common user
5	Feature	Email group	Common user
6	Feature	Provide material	Mentor Tutor
7	Feature	Consult material	Mentee Tutee
8	Feature	Reply presence for workshop	Common user
9	Feature	Obtain participation certificate for workshop	Common user
10	Feature	Apply for mentor role	Common user
11	Feature	Schedule meeting with tutees	Tutor
12	Feature	Obtain tutoring participation certificate	Tutor
13	Feature	Update meeting with tutees	Tutor
14	Feature	List meetings with tutees	Tutor
15	Feature	Cancel meeting with tutees	Tutor
16	Feature	Close meeting with tutees	Tutor

Figure 4.2: Features created for the Mentoring Academy Azure DevOps project.

We planned each sprint using the board functionality. Figure 4.3 exhibits the sixth sprint planning, which occurred from October 9 to October 22. The blue card represents the “product backlog item”, which is associated to the feature, and the yellow card represents the “task”, which is associated to the blue one.

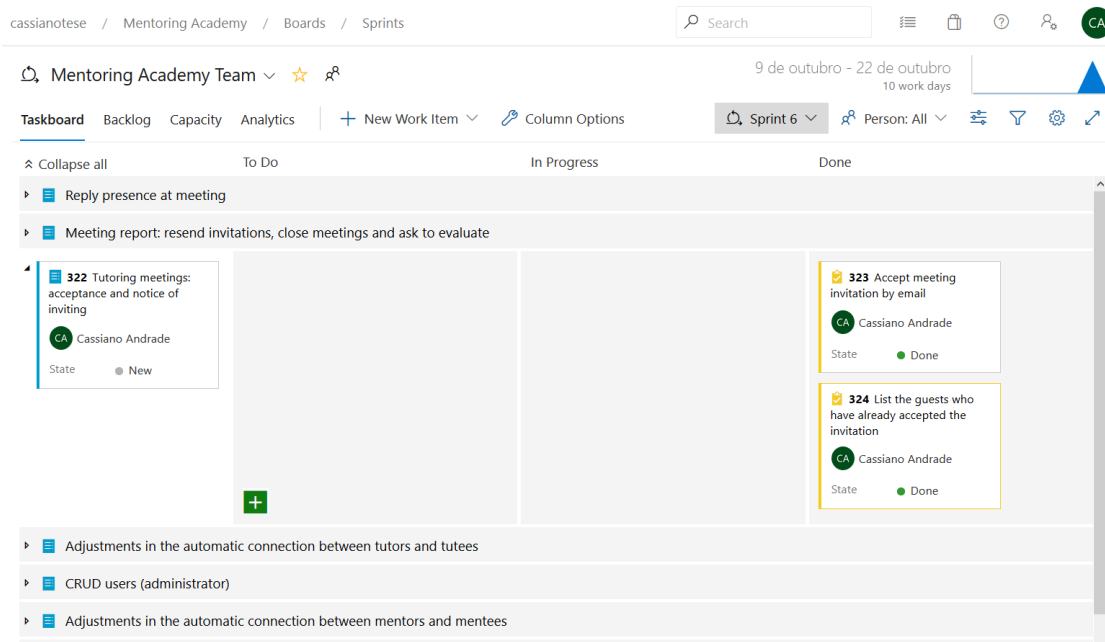


Figure 4.3: Sprint planning using the Azure DevOps.

Figure 4.4 presents the task card description of the Azure DevOps. On the right panel there are two sections: (1) development and (2) related work. Regarding the development section, it is possible to associate the task with a branch, so its commits can be registered in the task’s description. In terms of the related work section, it contains all the items which the card is associated to.

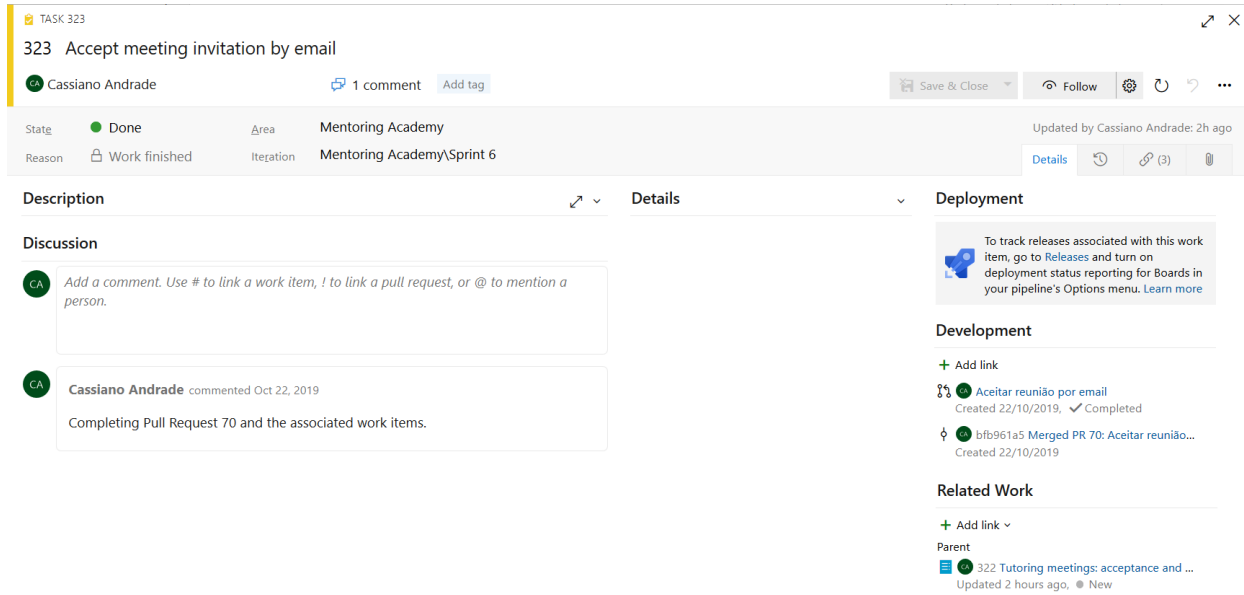


Figure 4.4: Task card description of the Azure DevOps.

Figure 4.5 exhibits the branches created for the Mentoring Academy development. The “dev” branch is used for development and testing. Once the code is ready, this branch is merged with the “master” branch, which contains the code tested and ready for deploy.

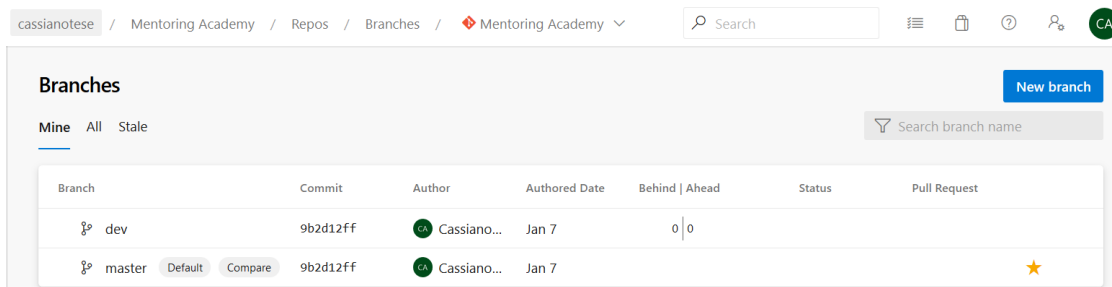


Figure 4.5: Branches created for the Mentoring Academy development in the Azure DevOps repository.

Figure 4.6 shows the commits made to the “master” branch. It is possible to observe on the left part that this branch is constantly being merged. This happens because the development occurs on another branch, therefore leaving both branches constantly in different stages of development.

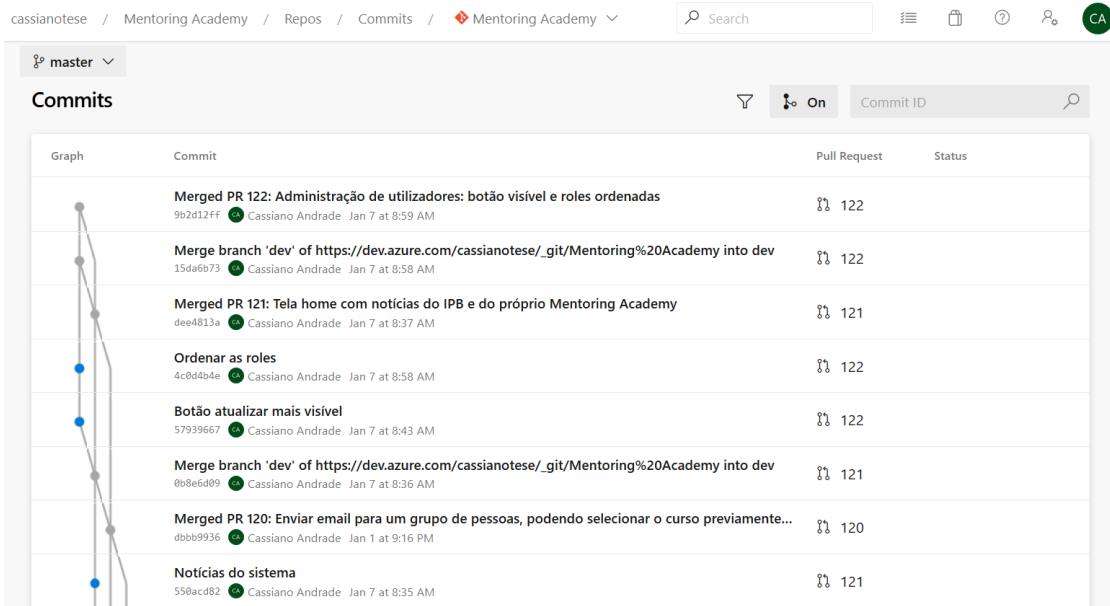


Figure 4.6: Organization of the commits in the Azure DevOps repository.

The development of the Mentoring Academy web application started on June 4th of 2019 and ended on January 7th of 2020, using Scrum as the development methodology. Altogether, 11 sprints were held lasting fifteen days each. Figure 4.7 presents the sprints scheduled for the Mentoring Academy web application development.

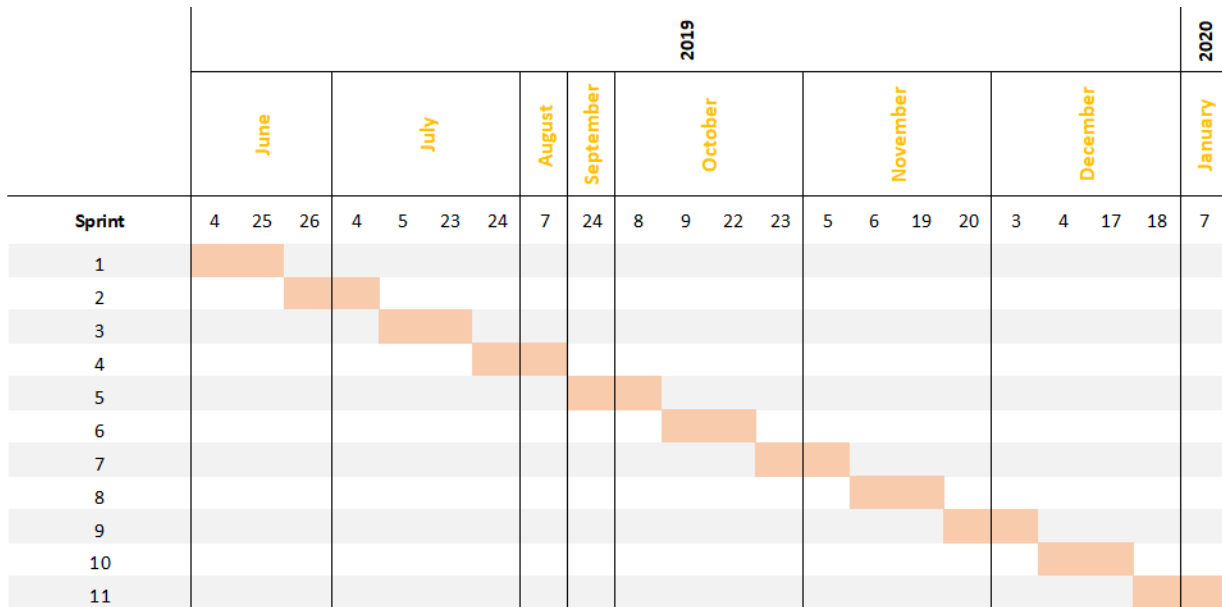


Figure 4.7: Sprints schedule for the Mentoring Academy web application development.

4.2 Developed features

The first part of the application developed was the landing page, like shown by Figure 4.8. The landing page contains 5 sections: (1) the program, which contains the main objectives of the Mentoring Academy program, (2) mentoring, which explains how the peer mentoring works, (3) tutoring, which explains how the peer tutoring works, (4) platform, which contains the system's main functionalities and (5) frequently asked questions.

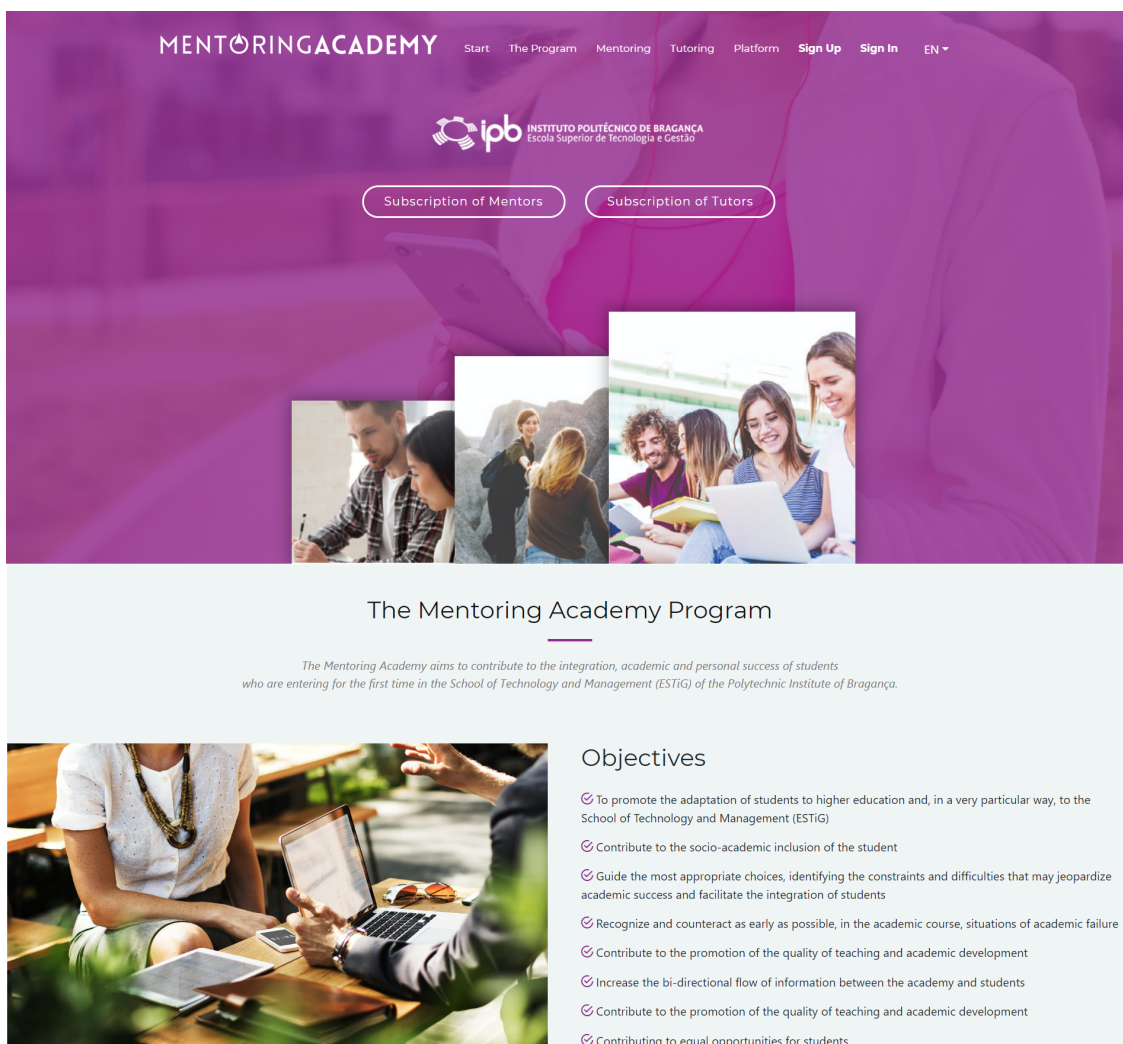


Figure 4.8: Screenshot of the Mentoring Academy landing page.

The system was developed with the internationalization functionality, which is responsible for translating the whole page into a desired language. For now, the system supports two languages, which are english and portuguese. Figure 4.9 shows the combo box responsible for the choice of language.

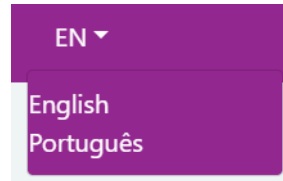
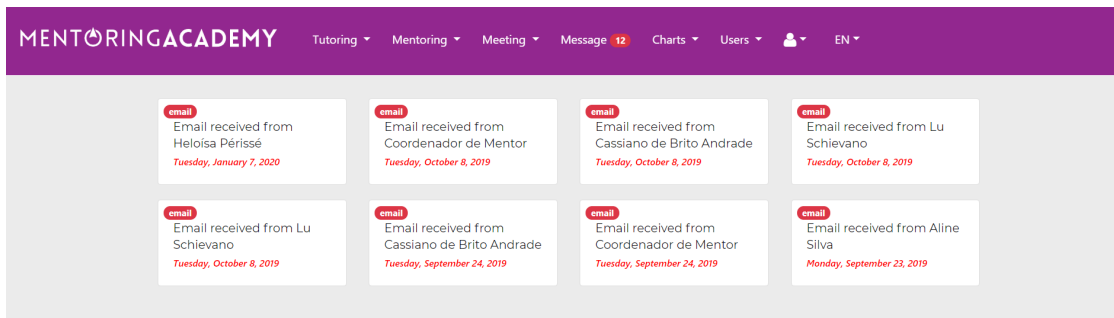


Figure 4.9: Screenshot of the Mentoring Academy internationalization functionality.

Once the user is logged in, he/she will have access to the home page, like shown by Figure 4.10. The user will have access to the functionalities which he/she has authorization to access in the upper menu. The home page contains two types of displayed information: (1) system notifications, which contains the most recent emails and/or scheduled meetings regarding the logged in user and (2) IPB news, containing the most recent news about IPB.



IPB News

Eleição da CT-IPB - LISTA DE CANDIDATURA

Relativamente ao processo eleitoral em curso, a Comissão Eleitoral, nos termos a que se refere a convocatória, informa que:

- 1 - Foi apresentada e aceite a lista de candidatura que se anexa;
- 2 - Podem os interessados apresentar reclamação até ao dia 6 de janeiro de 2020.

A documentação pode ser consultada em:

<http://www.ipb.pt/files/20200102sj80.pdf>

<http://www.ipb.pt/files/20200102nxt.pdf>

<http://www.ipb.pt/files/20200102mvm4.pdf>

<http://www.ipb.pt/files/202001021tq9.pdf>

Friday, January 3, 2020

Nota de Luto pela morte do aluno do IPB, Luís Giovani dos Santos Rodrigues

O Instituto Politécnico de Bragança, tendo tomado conhecimento, no dia de ontem, 31 de dezembro, da morte do seu aluno Luís Giovani dos Santos Rodrigues, do curso de Design de Jogos Digitais, natural da ilha do Fogo, em Cabo Verde, apresenta à família e amigos as suas mais sentidas condolências neste momento de dor e consternação.

As circunstâncias trágicas que determinaram a morte deste jovem, envolvem toda a comunidade IPB na mais profunda solidariedade com os que lhe são próximos, manifestando a nossa disponibilidade para apoiar a família e o nosso desejo de que todos os factos sejam cabal e inteiramente apurados.

Determina-se luto académico até à realização das exéquias fúnebres.

Wednesday, January 1, 2020

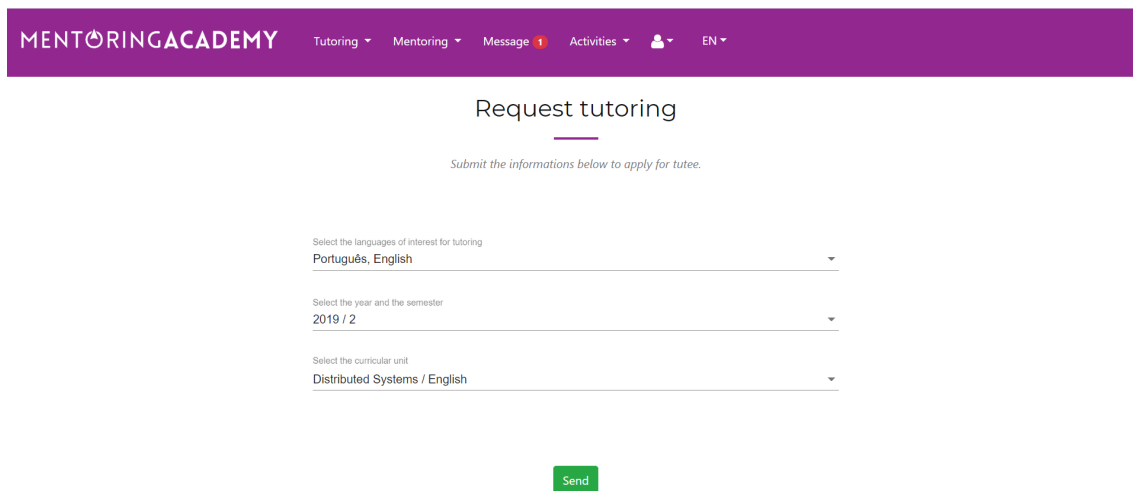
Figure 4.10: Screenshot of the Mentoring Academy home page after logging in.

In Section 3.3.1, 72 functional requirements were defined for the Mentoring Academy web application. The eleven sprints executed covered 44 functional requirements which were prioritized, leaving 28 functional requirements undone. The following subsections present the developed features and the functional requirements complied by each feature.

4.2.1 Request tutoring

Figure 4.11 presents the screenshot regarding the request tutoring functionality, which complies with the FR-01 (A common user shall be able to request tutoring).

In order to request tutoring, the student must select (1) the languages he/she can speak, (2) the current year and semester and (3) the curricular unit, then click on the “Send” button. If the student wants to remove his/her request, he/she can deselect the curricular unit and click on the button, then the request will be removed.



The screenshot shows the 'Request tutoring' form in the MENTORING ACADEMY interface. The form is titled 'Request tutoring' and includes a sub-header 'Submit the informations below to apply for tutee.' Below this, there are three dropdown menus: 'Select the languages of interest for tutoring' (with 'Português, English' selected), 'Select the year and the semester' (with '2019 / 2' selected), and 'Select the curricular unit' (with 'Distributed Systems / English' selected). A green 'Send' button is located at the bottom of the form.

Figure 4.11: Screenshot regarding the request tutoring functionality.

4.2.2 Request mentoring

Figure 4.12 exhibits the screenshot regarding the request mentoring functionality, which complies with the FR-02 (A common user shall be able to request mentoring).

The process of requesting mentoring is simpler, being necessary to select just the current year and semester then click on the “Send” button. To remove the request, the student can click on the “Remove” button that appears once the request has been made.

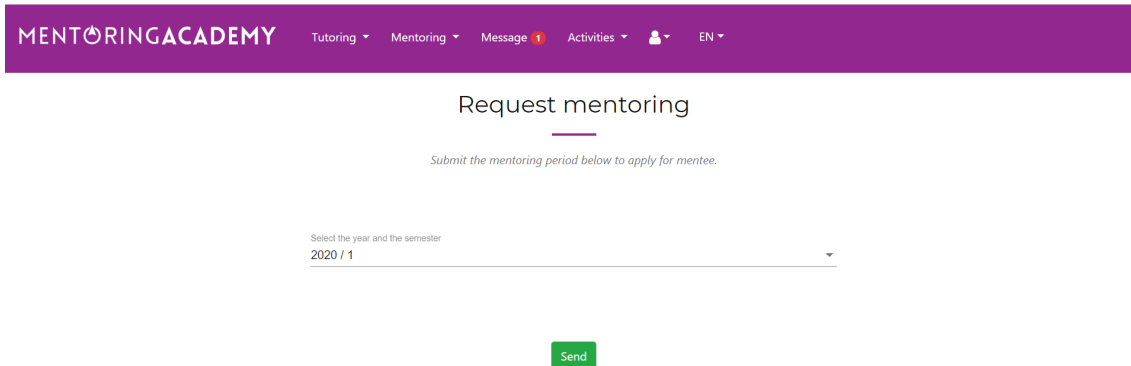


Figure 4.12: Screenshot regarding the request mentoring functionality.

4.2.3 Apply for tutor role and remove application

Figure 4.13 presents the screenshot about the appliance for tutor role functionality, which complies with the FR-03 (A common user shall be able to apply himself/herself for tutor role) and FR-04 (A common user shall be able to remove his/her application for tutor role).

In order to apply for tutor, the student must (1) select the languages he/she can speak, (2) select the current year and semester, (3) select the curricular unit, (4) upload the curriculum and (5) upload the motivation letter, then click on the “Confirm application” button.

The student can update the curriculum and the motivation letter at any time by clicking on the “Update” button, and he/she can also visualize them by clicking on the “View” button.

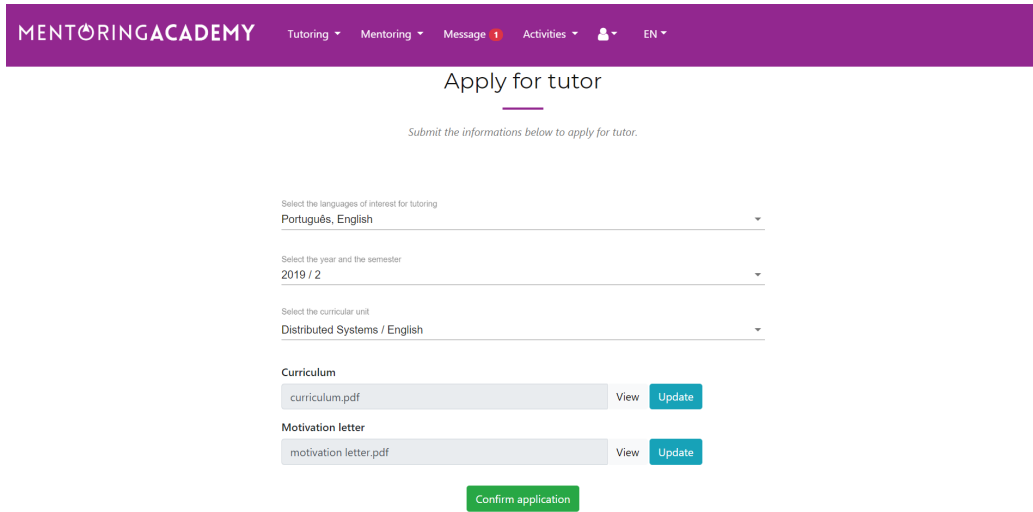


Figure 4.13: Screenshot regarding the appliance for tutor role functionality.

In case the student wants to remove the application, he/she can just deselect the curricular unit then click on the “Remove application” button, as shown in Figure 4.14.

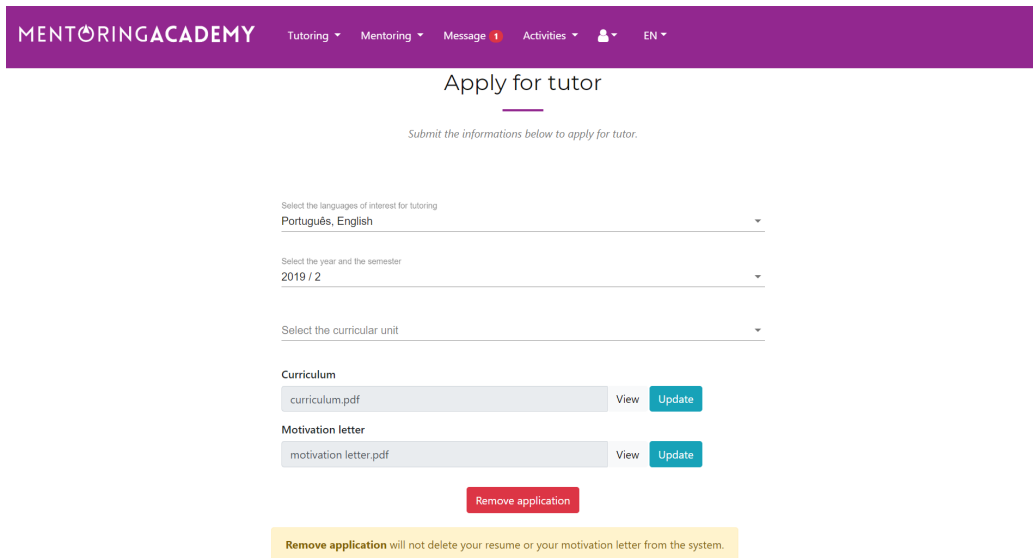


Figure 4.14: Screenshot regarding the application removal for tutor role functionality.

4.2.4 Apply for mentor role and remove application

Figure 4.15 exhibits the screenshot about the appliance for mentor role functionality, which complies with the FR-05 (A common user shall be able to apply himself/herself for mentor role) and FR-06 (A common user shall be able to remove his/her application for mentor role).

In order to apply for mentor, the student must (1) select the current year and semester, (2) upload the curriculum and (3) upload the motivation letter. Once the application has been made, the student can update it by any time by clicking on the “Confirm application” button or remove it by clicking on the “Remove application” button.

The screenshot shows the 'Apply for mentor' page in the MENTORINGACADEMY system. The page has a purple header with the logo and navigation links: Tutoring, Mentoring, Message (with a notification icon), Activities, a user profile icon, and EN. The main heading is 'Apply for mentor' with a subtitle 'Submit the informations below to apply for mentor.' Below this is a dropdown menu for 'Select the year and the semester' currently set to '2020 / 1'. There are two file upload sections: 'Curriculum' with a file named 'curriculum.pdf' and 'Motivation letter' with a file named 'motivation letter.pdf'. Each file has 'View' and 'Update' buttons. At the bottom, there are two buttons: 'Confirm application' (green) and 'Remove application' (red). A yellow warning box at the bottom states: 'Remove application will not delete your resume or your motivation letter from the system.'

Figure 4.15: Screenshot regarding the appliance and application removal for mentor role functionalities.

4.2.5 Send, read, update and delete emails

The following figures present the screenshots about the email functionalities, which complies with the FR-09 (A common user shall be able to send, read, update and delete emails), FR-10 (A common user shall be able to receive messages sent to email group) and FR-57 (The school, mentor and tutor coordinators shall be able to send message to email group).

Figure 4.16 shows three tabs: (1) new message, which is select in the figure, (2) messages sent and (3) messages received. In order to send a new message, the user can optionally filter the recipients by selecting courses or entry years, then the user must select the recipient(s), check or not the “Send to recipients personal email” checkbox, define the email’s title, upload attachments if needed, write the message then click on the “Send” button.

In the Mentoring Academy web application, the notification of a new email appears as a red number in the toolbar. In Figure 4.16, it is shown that the logged in user has 11 unread emails.

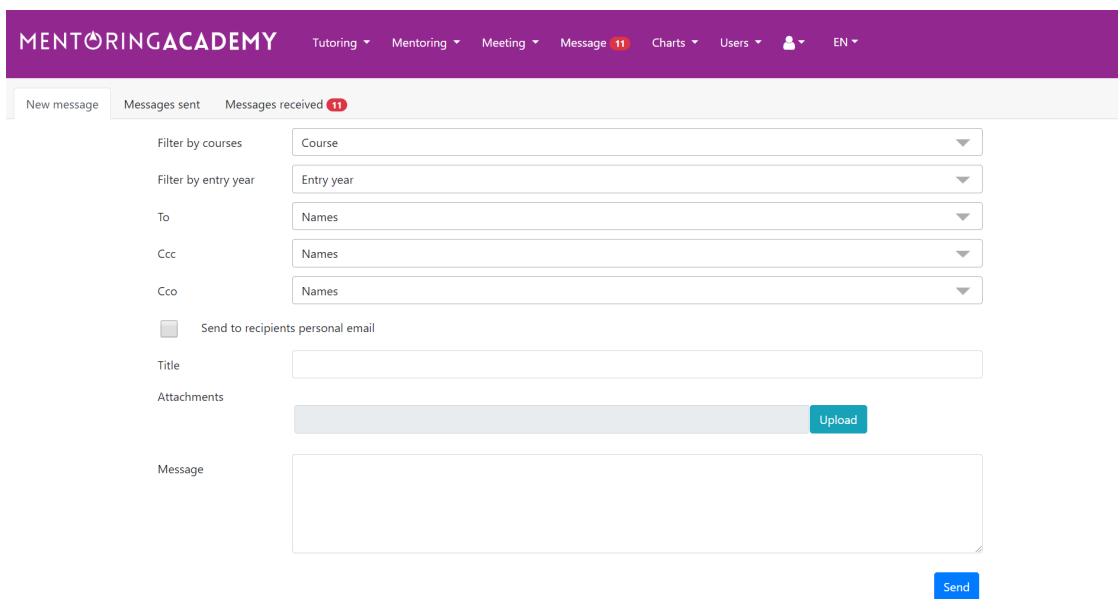


Figure 4.16: Screenshot regarding the sending a new email functionality.

Figure 4.17 shows the content of the second tab “Messages sent” when selected, where it is possible to see all the emails sent by the logged in user. In the figure, it is shown 2 emails and that it is possible to delete each one by clicking on the trash icon on the left, or read them just by clicking on the email.

Figure 4.18 shows the content of the third tab “Messages received” when selected, where it is possible to see all the emails received by the logged in user. It is possible to delete a email by clicking on the trash icon on the left or read a email by clicking

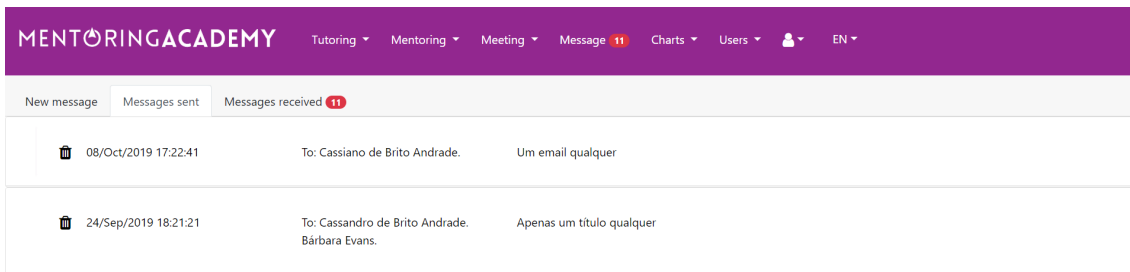


Figure 4.17: Screenshot regarding the reading or deleting the emails sent functionalities.

on the email. In the figure, the content of the second email is shown, which contains 3 attachments and a short message.

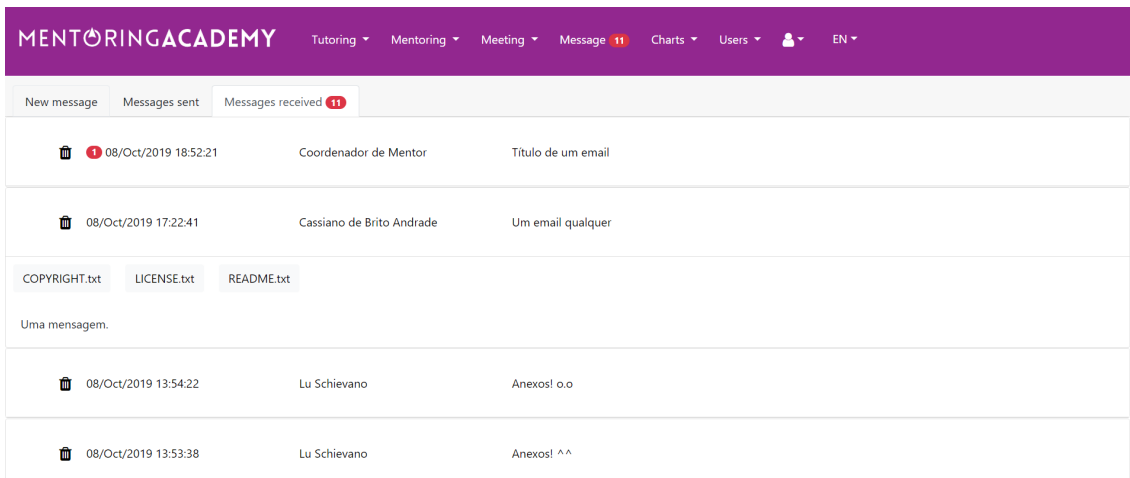


Figure 4.18: Screenshot regarding the reading or deleting the emails received functionalities.

4.2.6 CRUD users and assign or remove user roles

Figure 4.19 exhibits the screenshot about the users and roles update functionalities, which complies with the FR-11 (An administrator shall be able to activate or deactivate users) and FR-18 (An administrator shall be able to assign or remove user roles).

The administrator can filter the users by typing information about their names, emails or phone numbers or by selecting if he/she wants activated or deactivated users. For each user, the administrator can update his/her roles by selecting the roles in the combo box then click on the “Update” button in the column’s title.

MENTORINGACADEMY Meeting Message 11 Charts Users EN

User Administration

Filter by name, email or phone number Filter by enabled or disabled users

Full name	Email	Roles	Update	Operation	Activation history
Abdula Jacov	deletar5@gmail.com	Mentee, Student		Activate	Activation history
Adriana Esteves	deletar14@gmail.com	Student		Deactivate	Activation history
Alessandra Rodrigues	deletar8@gmail.com	Student		Deactivate	Activation history
Aline Silva	cassiano.b.andrade@gmail.com	Employee, Mentee, Stu...		Deactivate	Activation history
Alisson DILAurentis	cassiano.b.andrade@gmail.com	Mentee, Student, Tutee...		Deactivate	Activation history
Andr� Priester	cassiano.b.andrade@gmail.com	Mentee, Mentor, Student		Deactivate	Activation history
Andressa Urach	cassiano.b.andrade@gmail.com	Mentor, Student		Deactivate	Activation history
B�rbara Evans	cassiano.b.andrade@gmail.com	Student		Activate	Activation history
Bonnie Bennet	deletar20@gmail.com	Student		Deactivate	Activation history
Bruno Galla�o	deletar15@gmail.com	Mentee, Student		Deactivate	Activation history

Items per page: 10 1 - 10 of 57

Figure 4.19: Screenshot regarding the users and roles update functionalities.

The administrator can activate or deactivate a user by clicking on the “Activate” or “Deactivate” buttons, respectively. Each activation or deactivation process requires a comment by the administrator about the motives that led him/her to make this operation. Figure 4.20 shows the activation history of a user, obtained by clicking on the button “Activation history” in the last column.

The first column shows the administrator’s full name, the second column exhibits the operation name, the third column shows the reason and the last column the operation date.

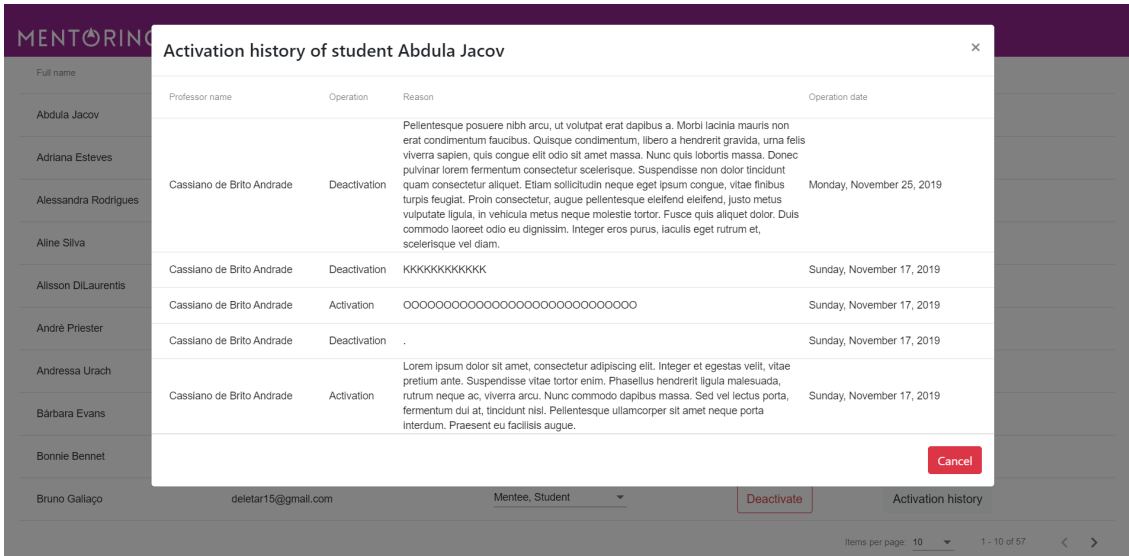


Figure 4.20: Screenshot regarding the activation and deactivation history of a user.

4.2.7 CRUD meeting room

The following figures present screenshots about the CRUD operations regarding the meeting rooms, which complies with the FR-17 (An administrator shall be able to create, read, update and delete meeting room).

Figure 4.21 shows all the meeting rooms created in a table and a button “Create” responsible for starting the creation of a new meeting room. For each meeting room in the table, it is possible to update it by clicking on the pencil icon or to delete it by clicking on the trash icon.

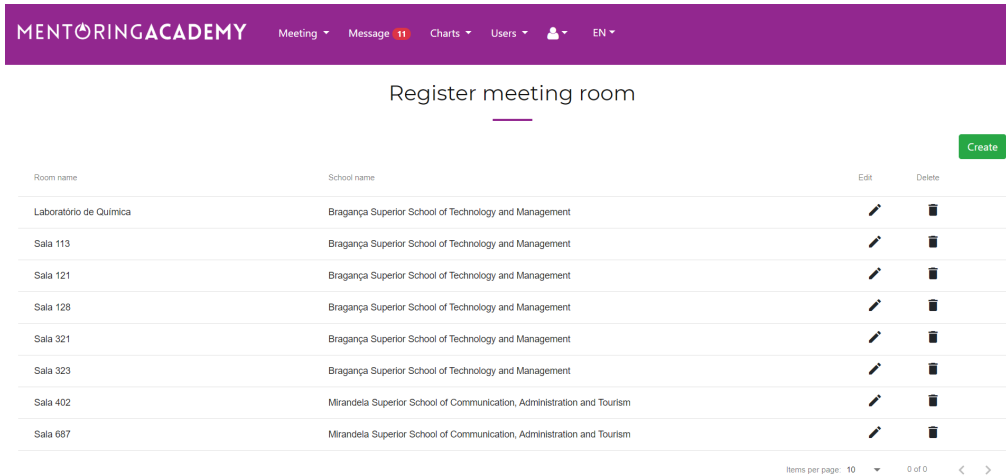


Figure 4.21: Screenshot regarding the meeting rooms listing functionality.

Figure 4.22 shows the pop-up that appears after clicking the “Create” button. In order to create a new room, it is necessary to type the room’s name, select the school where it is located, then add its availability times for meetings, selecting for each availability the weekdays and the begin and end times.

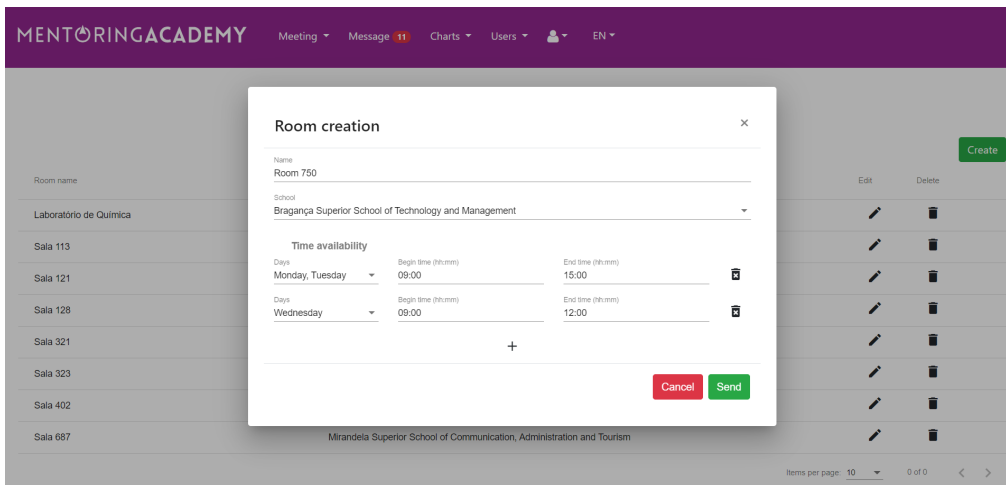


Figure 4.22: Screenshot regarding the meeting room creation functionality.

4.2.8 List meetings

Figure 4.23 presents the screenshot about the meeting listing functionality, which complies with the FR-21 (The tutor shall be able to list the meetings scheduled by him/her), FR-33 (The mentor shall be able to list the meetings scheduled by him/her) and FR-64 (The school, mentor and tutor coordinators shall be able to list meetings scheduled by him/her).

In the figure, it is shown a “Create or update meeting” button, which is responsible to create, update or cancel a meeting, and a table with all the meetings scheduled by the logged in user. For each meeting in the table, it is possible to see the guests who accepted, declined or didn't reply the invitation.

Date	Start time	End time	Guests answer
Wednesday, December 18, 2019	06:30:00	07:30:00	<input type="checkbox"/> Mentorado 2 (Not answered)
Wednesday, December 18, 2019	07:30:00	08:30:00	_____
Tuesday, December 17, 2019	05:00:00	06:00:00	_____
Monday, December 16, 2019	05:00:00	06:00:00	_____
Wednesday, December 11, 2019	06:30:00	07:30:00	_____
Tuesday, December 10, 2019	05:00:00	06:00:00	_____
Tuesday, November 26, 2019	06:00:00	07:00:00	_____

Figure 4.23: Screenshot regarding the meeting listing functionality.

Figure 4.24 exhibits the screenshot about the scheduled meetings report functionality, which complies with the FR-48 (The school and mentor coordinators shall be able to list meetings between mentors and mentees) and FR-55 (The school and tutor coordinators shall be able to list meetings between tutors and tutees).

In order to receive the scheduled meetings report of a tutor or mentor, the coordinator must select his/her name in the first box and choose the year in which the meetings were held in the second box. After that, all the meetings of the selected host in the given year will be shown in the table below.

The third box allows the coordinator to do some operations over the meetings, which are: (1) ask to evaluate the meeting, in case of a meeting that was already held but yet not evaluated by its guests or host, (2) resend invitations, in case of a meeting that wasn't held yet and has invitations unread by its guests and (3) close meetings, in case of a meeting that was held, not evaluated by all its guests or host and yet the coordinator decides to close it. The second combo box shows all the meetings so the coordinator can select one or more meetings to apply the selected operation on them.

MENTORING ACADEMY Tutoring ▾ Meeting ▾ Message Activities ▾ Charts ▾ 👤 ▾ EN ▾

Scheduled Meetings Report

Select the host to list the meetings scheduled by him/her.

Choose the host
Only meetings regarding the chosen host will appear.

Choose the host
Cassiano de Brito Andrade ▾

Choose the year
Only meetings for the chosen year will appear.

Year
2019 ▾

Old meetings
Resend meeting invitations to those who have not yet answered or close them.

Operation ▾

Meetings ▾

Apply

More information	Date	Qty. of guests	Qty. without answering	Qty. without evaluating	State	Close meeting
+	Dec 18, 2019 06:30:00	1	1	1	Waiting closure	⊗
+	Dec 18, 2019 07:30:00	1	1	1	Waiting closure	⊗
+	Dec 17, 2019 05:00:00	1	1	1	Waiting closure	⊗
+	Dec 16, 2019 05:00:00	1	1	1	Waiting closure	⊗
+	Dec 11, 2019 06:30:00	1	1	1	Waiting closure	⊗
+	Dec 10, 2019 05:00:00	1	1	1	Waiting closure	⊗
+	Nov 26, 2019 06:00:00	1	1	1	Waiting closure	⊗
+	Nov 20, 2019 06:30:00	1	1	1	Waiting closure	⊗
+	Nov 20, 2019 10:00:00	1	0	1	Waiting closure	⊗

Items per page: 10 ▾ 1 - 9 of 9 < >

Figure 4.24: Screenshot regarding the scheduled meetings report functionality.

The plus icon in the beginning of each line of the table is responsible for opening a pop-up like shown in Figure 4.25. This pop-up will show all the guests invited for the meeting, the presence confirmation status (present, not present or unanswered) and the evaluation if it were made.

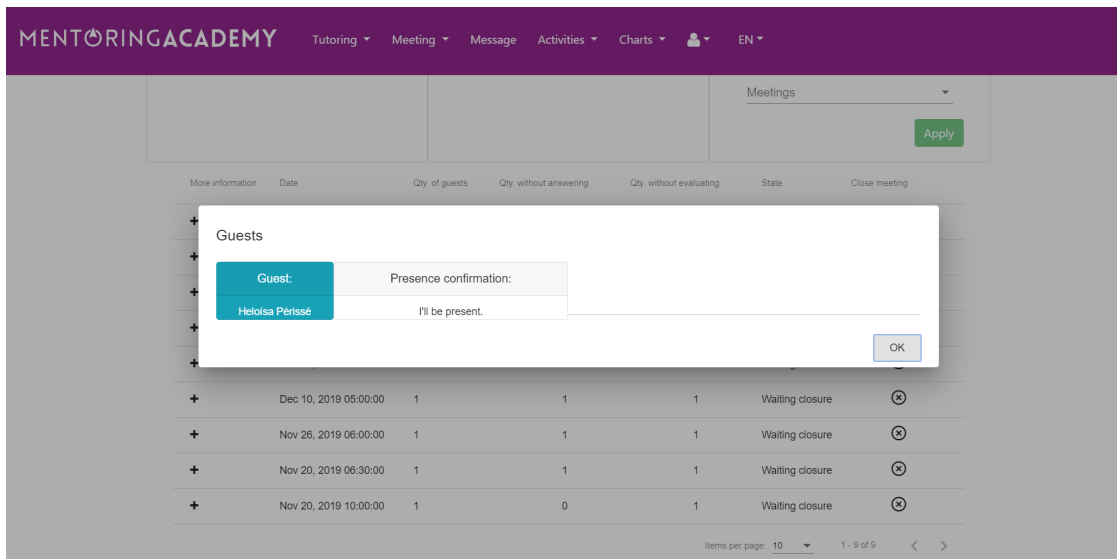


Figure 4.25: Screenshot regarding a meeting evaluation functionality.

The x icon in the end of each line of the table is responsible for closing the meeting, like shown in Figure 4.26. The coordinator can close a meeting even knowing that there are pending evaluations.

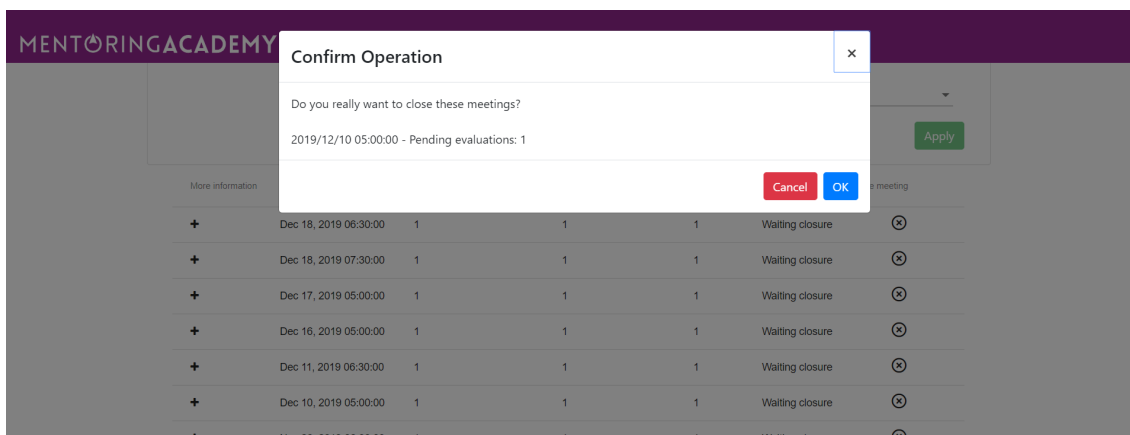


Figure 4.26: Screenshot regarding the close meeting functionality.

4.2.9 Schedule a meeting

The following figures present the screenshots about the schedule meeting functionality, which complies with the FR-19 (The tutor shall be able to schedule a meeting with the tutees), FR-31 (The mentor shall be able to schedule a meeting with the mentees), FR-49 (The school and mentor coordinators shall be able to schedule a meeting with mentors) and FR-56 (The school and tutor coordinators shall be able to schedule a meeting with tutors).

Figure 4.27 shows the pop-up that appears once the user click the “Create or update meeting” button. The pop-up offers three operations: (1) create meeting, (2) update meeting and (3) cancel meeting. This section will show the “Create meeting” functionality.

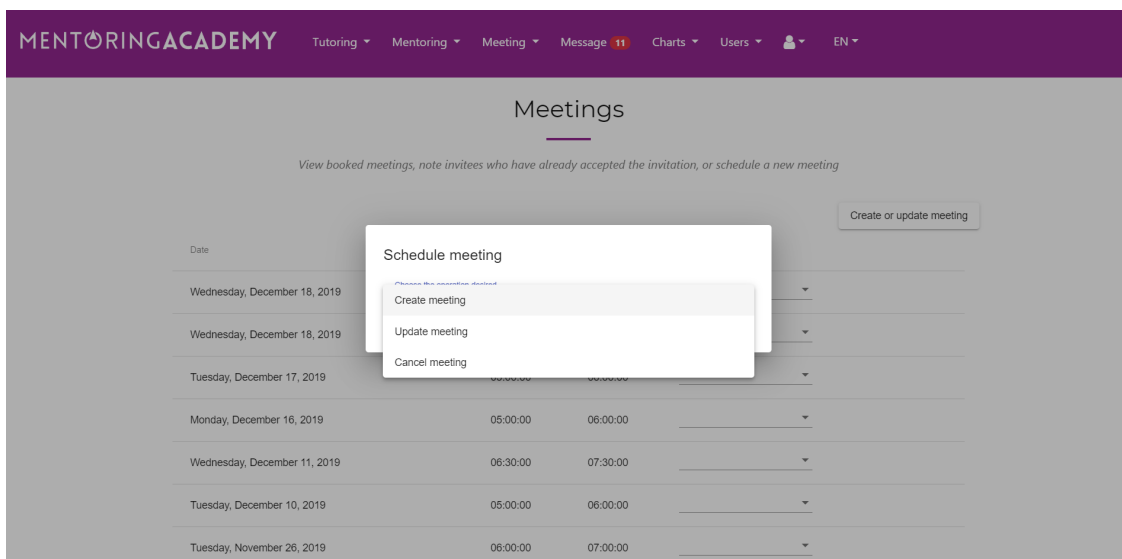


Figure 4.27: Screenshot regarding the schedule meeting functionality.

Figure 4.28 shows the pop-up that appears once the “Create meeting” option is selected. In order to schedule a meeting, the user must (1) select his/her role, (2) choose the school in where the meeting will be held, (3) select the room, (4) choose the date of the meeting, (5) select the current year and semester, (6) choose the time to begin and finish the meeting according to the room’s availability, (7) select the curricular unit (if applicable) and (8) select the guests. Finally, the user can hit the “Send” button to schedule the meeting.

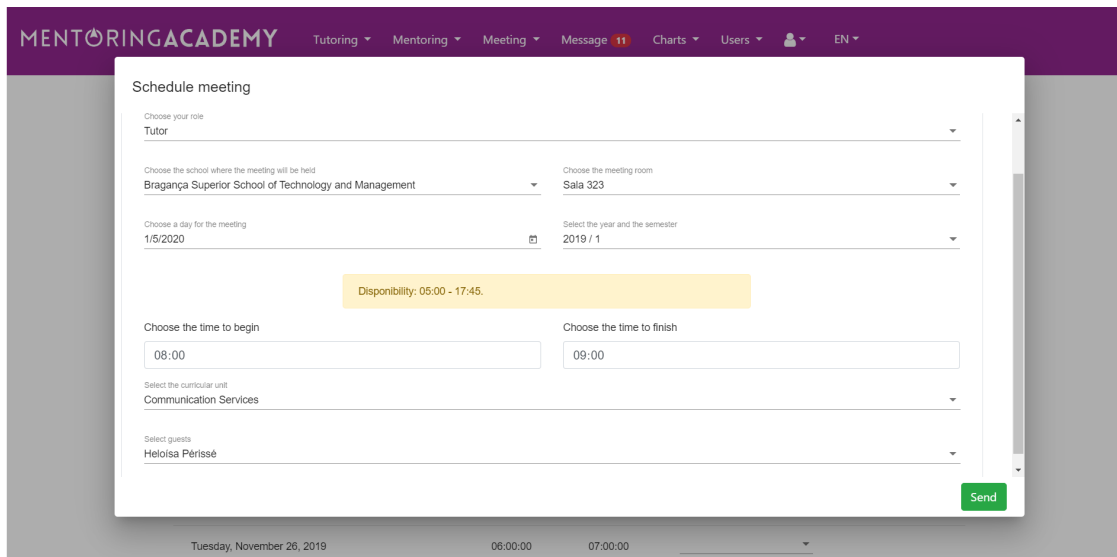


Figure 4.28: Screenshot regarding the schedule meeting functionality.

4.2.10 Update a meeting

Figure 4.29 exhibits the screenshot about the meeting update functionality, which complies with the FR-20 (The tutor shall be able to update a meeting with the tutees), FR-32 (The mentor shall be able to update a meeting with the mentees) and FR-62 (The school, mentor and tutor coordinators shall be able to update meeting scheduled by him/her).

In order to update a meeting, it is necessary to click on the “Create or update meeting” button then select the “Update meeting” option in the pop-up. Afterwards, the user has to select the meeting which he/she wants to update, change the information he/she needs to change then click on the “Send” button.

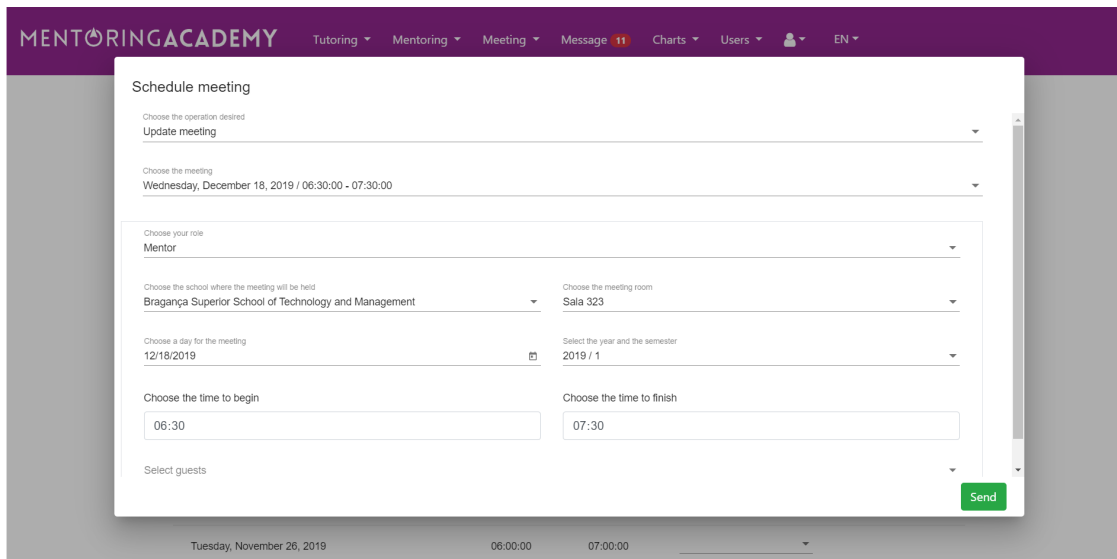


Figure 4.29: Screenshot regarding the meeting update functionality.

4.2.11 Cancel a meeting

Figure 4.30 presents the screenshot about the meeting cancellation functionality, which complies with the FR-22 (The tutor shall be able to cancel a meeting scheduled by him/her), FR-34 (The mentor shall be able to cancel a meeting scheduled by him/her) and FR-63 (The school, mentor and tutor coordinators shall be able to cancel meeting scheduled by him/her).

In order to cancel a meeting, it is necessary to click on the “Create or update meeting” button then select the “Cancel meeting” option in the pop-up. Afterwards, the user has to select the meeting which he/she wants to cancel. The information about the meeting will be presented so the user can make sure that is the right meeting he/she wants to cancel, then click on the “Cancel” button.

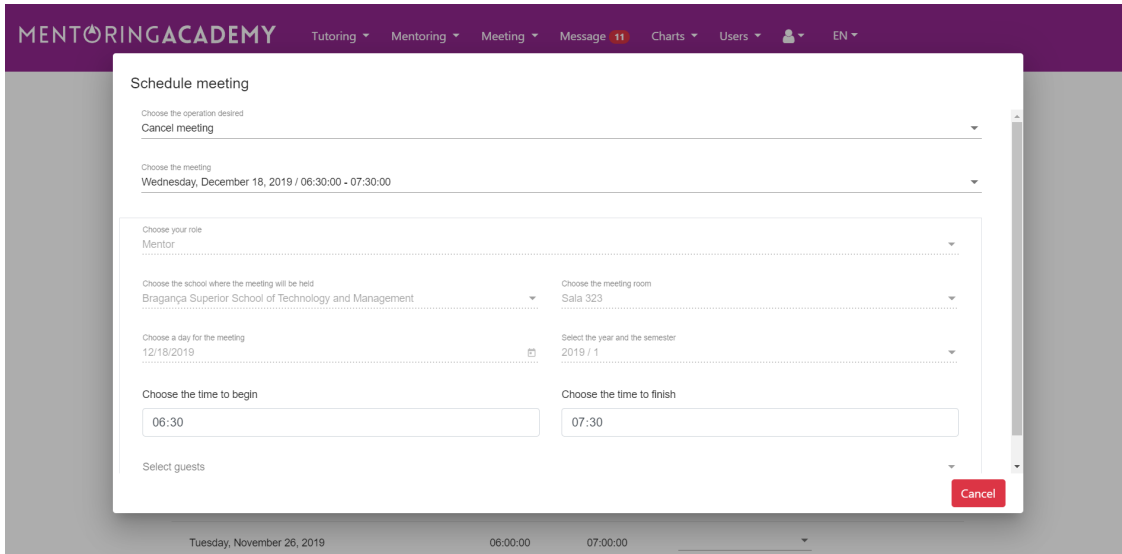


Figure 4.30: Screenshot regarding the meeting cancellation functionality.

4.2.12 Reply a meeting invitation

Figure 4.31 exhibits the screenshot about the response of the meeting invitation functionality, which complies with the FR-25 (The tutor shall be able to reply a meeting invitation sent by the tutor or school coordinator), FR-27 (The tutee shall be able to reply a meeting invitation sent by the tutor), FR-37 (The mentor shall be able to reply a meeting invitation sent by the mentor or school coordinator) and FR-39 (The mentee shall be able to reply a meeting invitation sent by the mentor).

The screen shows a table with all the meetings which the logged in user was invited to attend to. The columns present the information in this order: (1) the meeting date and time, (2) the meeting host, (3) the start and (4) end times of the meeting, (5) the meeting status, which could be canceled, scheduled or closed, (6) the presence response combo box, from which the user can reply the meeting invitation with “I’ll be present” or “I’ll be absent”, and (7) the “evaluate meeting” button, which is responsible of giving the user the possibility to evaluate the meeting which he/she attended to.

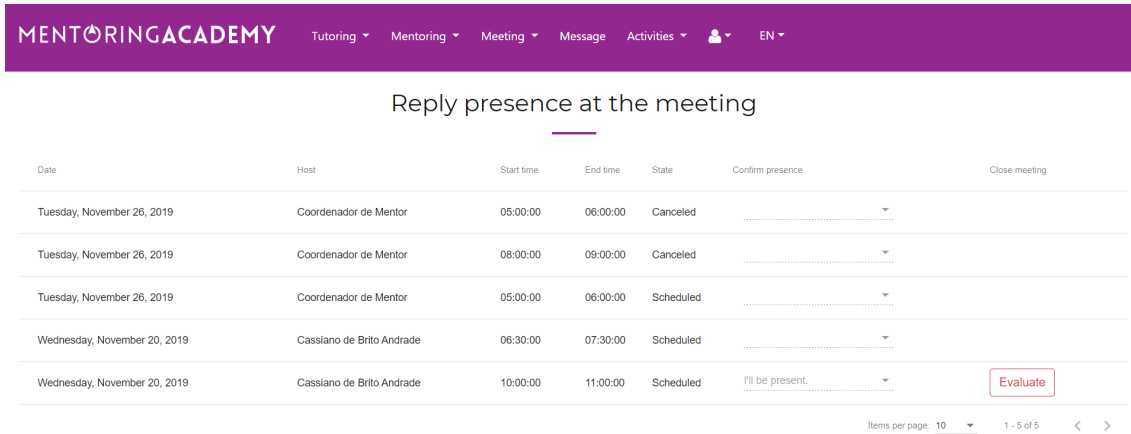


Figure 4.31: Screenshot regarding the response of the meeting invitation functionality.

4.2.13 Close a meeting

Figure 4.32 presents the screenshot about the meeting closure functionality, which complies with the FR-23 (The tutor shall be able to close a past meeting by submitting a report), FR-28 (The tutee shall be able to report a meeting feedback), FR-29 (The tutee shall be able to close a meeting by submitting a report), FR-35 (The mentor shall be able to close a past meeting by submitting a report), FR-40 (The mentee shall be able to report a meeting feedback), FR-41 (The mentee shall be able to close a meeting by submitting a report) and FR-65 (The school, mentor and tutor coordinators shall be able to close meeting scheduled by him/her).

In order to close a meeting, the user needs to click on the “Evaluate” button of a meeting which he/she attended to, then the pop-up will appear like shown in the figure. The users needs to fill two fields: (1) a comment about the meeting, which is textual, and (2) a rate from 1 (unhelpful) to 5 (very useful) which he/she can select in the combo box. Afterwards, the user can just click on the “Send” button to finalize the evaluation.

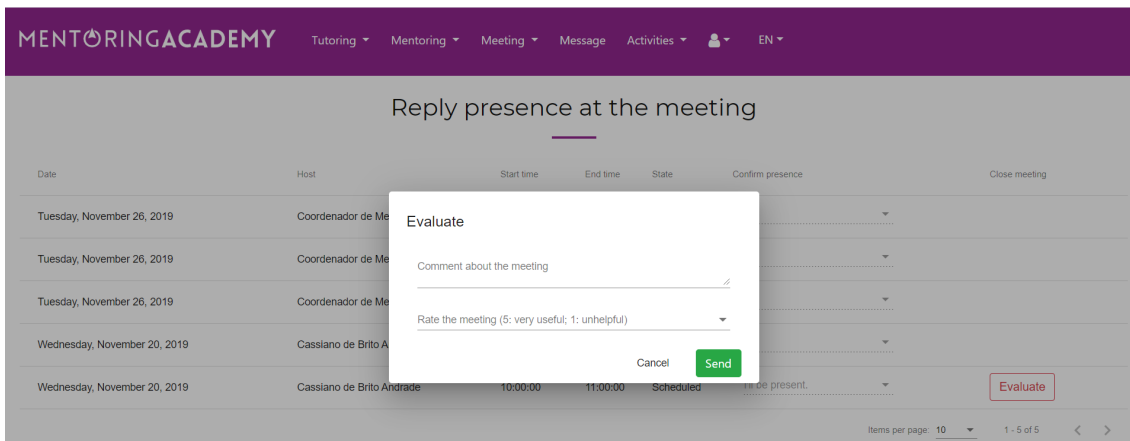


Figure 4.32: Screenshot regarding the meeting closure functionality.

4.2.14 Assign mentor to mentee

The following figures exhibit the screenshots about the mentor and mentee connection functionality, which complies with the FR-44 (The school and mentor coordinators shall be able to assign mentor to mentee manually) and FR-46 (The school and mentor coordinators shall be able to automatically assign mentor to mentee).

Figure 4.33 shows the page which is responsible for the manual and automatic connection between mentors and mentees made by the school and mentor coordinators. The first part of the page is composed of three boxes: (1) the first box is responsible for the year, semester and course selection, which will filter all shown mentors and mentees; (2) the second box is responsible for sorting the mentors, which offers the options “Name” and “Quantity of assignments”; (3) the third box is responsible for the automatic connection between mentors and mentees.

The second part of the page displays three important information: quantity of mentors, quantity of mentees and quantity of mentees that are not assigned to any mentor. Below it, there is a combo box which contains three view types: (1) connection by mentor, which is the selected one in the figure, (2) mentor table and (3) mentee table. The first view, shown by the figure, contains each mentor in a box, with their personal information (gender, country and city) and quantity of mentees selected for each one. The combo

box on each mentor's box contains the mentees which the coordinator can manual or automatic assign them to the mentors.

MENTORING ACADEMY Mentoring Meeting Message Activities Charts EN

Connect mentors and mentees

Year, semester and course
The mentors and mentees from the selected period will appear.
Select the year and the semester: 2019 / 1
Select the courses: Information systems

Mentors sorting
Sort the mentors below according to the selected criteria.
Order by mentors according to: Name
Sort

Automatic connection
Connect mentors and mentees according to the criteria below.
Select the connection criteria: 1 District, region or state
Mentees per mentor: 2 Tolerance: 0
 Ignore district, region and state of foreign countries
Connect Clean connections

QUANTITY OF MENTORS: 5 QUANTITY OF MENTEES: 8 MENTEES NOT SELECTED: 8

View type: Connections by me...

Andressa Urach Other / Portugal / Bragança 0 mentees selected	Select the mentees for the mentor
Cassiano de Brito Andrade Other / Portugal / Bragança 0 mentees selected	Select the mentees for the mentor
Mentor 1 Masculine / Brazil / Aruana 0 mentees selected	Select the mentees for the mentor
Mentor 2 Masculine / Brazil / Belo Horizonte 0 mentees selected	Select the mentees for the mentor
Pedro Silva Masculine / Portugal / Bragança 0 mentees selected	Select the mentees for the mentor

Send

Figure 4.33: Screenshot regarding the connection between mentors and mentees functionality.

The automatic connection offers three rules, which are (1) district, region or state, (2) country and (3) spoken language, like shown by Figure 4.34. The coordinator can select how many rules he/she wants, in any order he/she wants. The number that appears in the beginning of the rule represents the order of which the user selected the rule, so when the user clicks on the “Connect” button, the system will automatically connect mentors

and mentees following these rule orders.

At first, the system respects a maximum quantity of mentees per mentor, which is shown in the third box. If the coordinator needs a bigger quantity of mentees per mentor, he/she can put how much more he/she needs to add in the “Tolerance” field. In case the coordinator wants to apply the “District, region or state” rule only to portuguese students, he/she can check the “Ignore district, region or state of foreign countries” checkbox. If the coordinator needs to restart the connection, he/she can click on the “Clean connection” button to undo all the selections.

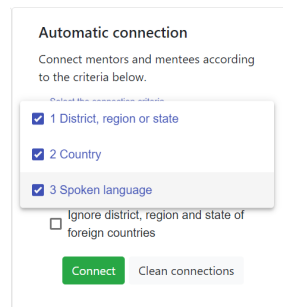


Figure 4.34: Screenshot regarding the automatic connection between mentors and mentees functionality.

Figure 4.35 shows what appears once the user clicks on the combo box contained inside the mentor’s box. The combo box contains all the mentees who are selected for that mentor or those who weren’t selected yet for any mentor. For each mentee, it is exhibited (1) name, (2) gender, (3) country and (4) city.



Figure 4.35: Screenshot regarding the assignment of mentees to a mentor manually.

Figure 4.36 shows the “Mentor table” view type, which is responsible for showing a table of which each line represents a mentor with their personal information, the quantity of mentees assigned to him/her and an assignment intensity bars which represents

graphically the quantity of mentees that mentor has: four bars represent that the mentor is approximately the one with more mentees assigned to him/her and one bar represents that the mentor is approximately the one with less mentees assigned to him/her. The coordinator can export this information to an Excel file by clicking on the “Export” button shown beside the view type combo box.

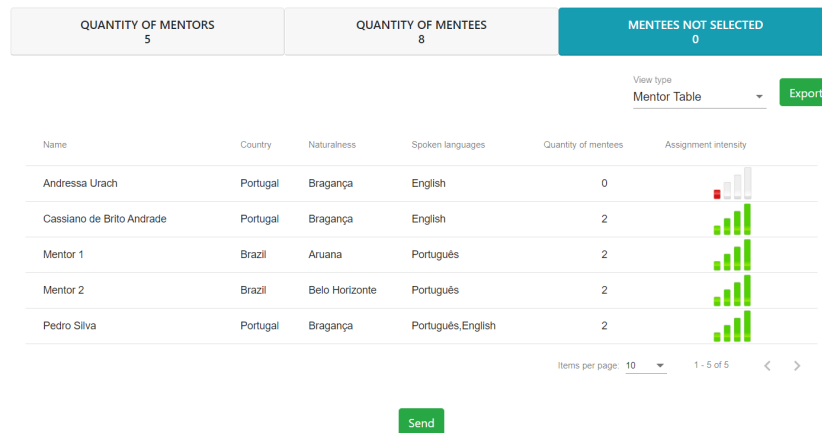


Figure 4.36: Screenshot regarding the mentors assignment intensity table.

Figure 4.37 shows the “Mentee table” view type, which is responsible for showing a table of which each line represents a mentee with their personal information and the tutor to whom he/she is assigned to. This table can also be exported to Excel by clicking on the “Export” button.

Name	Country	Naturalness	Spoken languages	Mentor name
Lorena Lopes	Portugal	Bragança	Português,English	Pedro Silva
Ingrid Guimarães	Japan	Ono	English	Cassiano de Brito Andrade
Luciano Hulk	Portugal	Bragança	Português,English	Pedro Silva
Mentorado 5	Brazil	Belo Horizonte	Português	Mentor 1
Mentorado 3	Brazil	Belo Horizonte	Português	Mentor 2
Mentorado 6	Brazil	Belo Horizonte	Português,English	Mentor 1
Helôisa Périssé	China	Sanya	Português,English	Cassiano de Brito Andrade
Mentorado 4	Brazil	Belo Horizonte	Português	Mentor 2

View type: Mentee table Export

Items per page: 10 1 - 8 of 8 < >

Send

Figure 4.37: Screenshot regarding the summed up information about the mentees.

4.2.15 Assign tutor to tutee

Figure 4.38 presents the screenshot about the connection between tutors and tutees functionality, which complies with the FR-51 (The school and tutor coordinators shall be able to assign tutor to tutee manually) and FR-53 (The school and tutor coordinators shall be able to automatically assign tutor to tutee).

This page is similar to the one presented in the subsection 4.2.14. The only difference is that this one is destined to the tutors and tutees, therefore in the first box the school or tutor coordinator has to select the curricular unit to filter the shown tutors and tutees. All the rest is similar to the explanation given in the previous section.

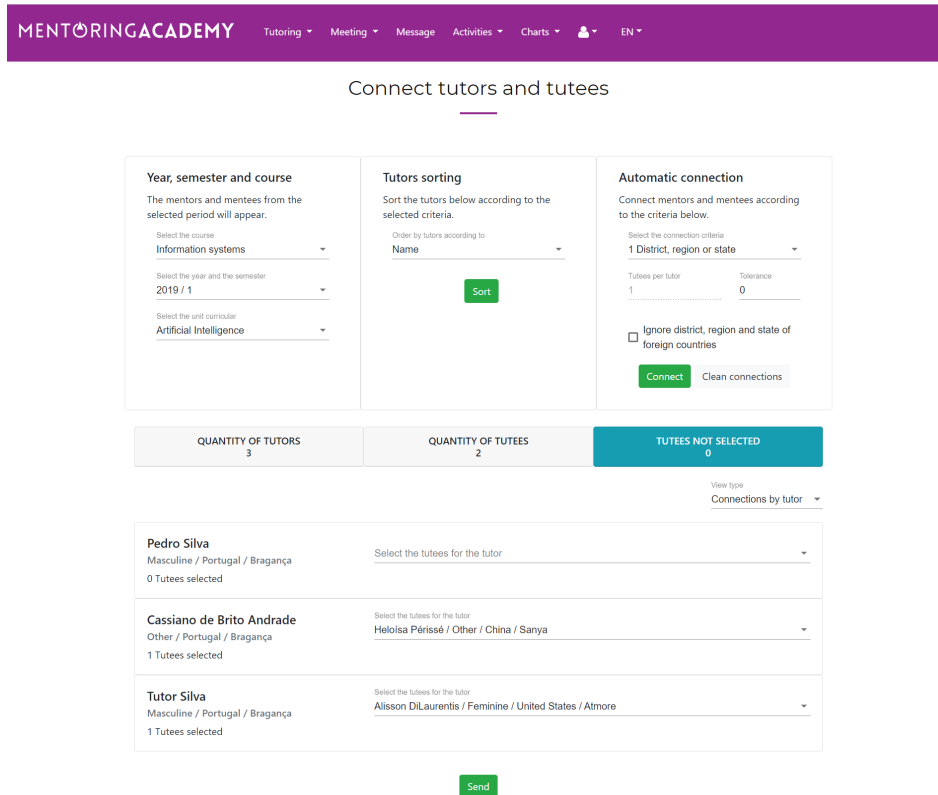


Figure 4.38: Screenshot regarding the connection between tutors and tutees functionality.

4.2.16 Graphs

It was developed for the Mentoring Academy system two graphs to help coordinators keep track of the program's status more easily. The first graph is about the assigned mentees, like shown in Figure 4.39. The coordinator can filter the mentees by their countries and the year of when they made part of the mentoring program, and the graph will show, for each country, the quantity of mentees not assigned to any mentor yet in a red bar and the quantity of mentees that were assigned to a mentor in a blue bar.

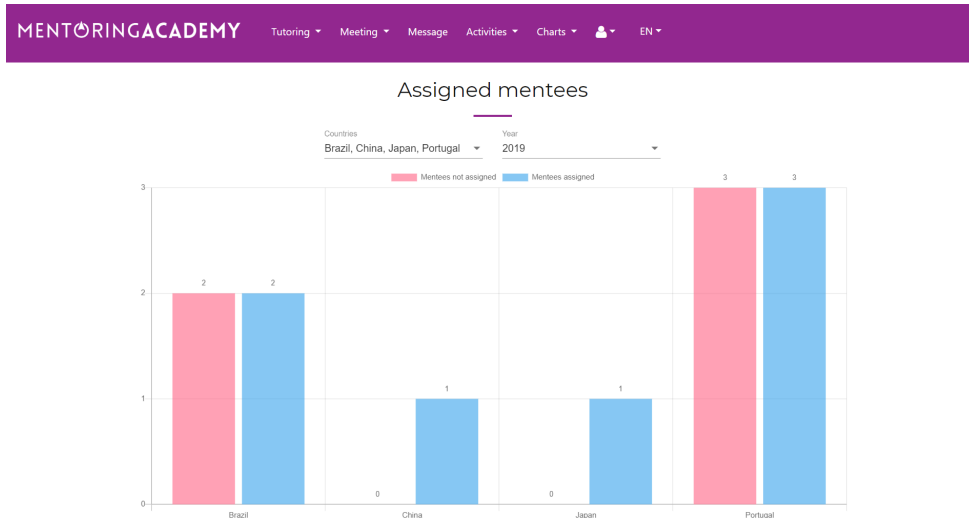


Figure 4.39: Screenshot regarding the graph about the assigned mentees.

Figure 4.40 presents the reunions per week graph, which can be used to see the quantity of meetings scheduled by mentors or tutors.

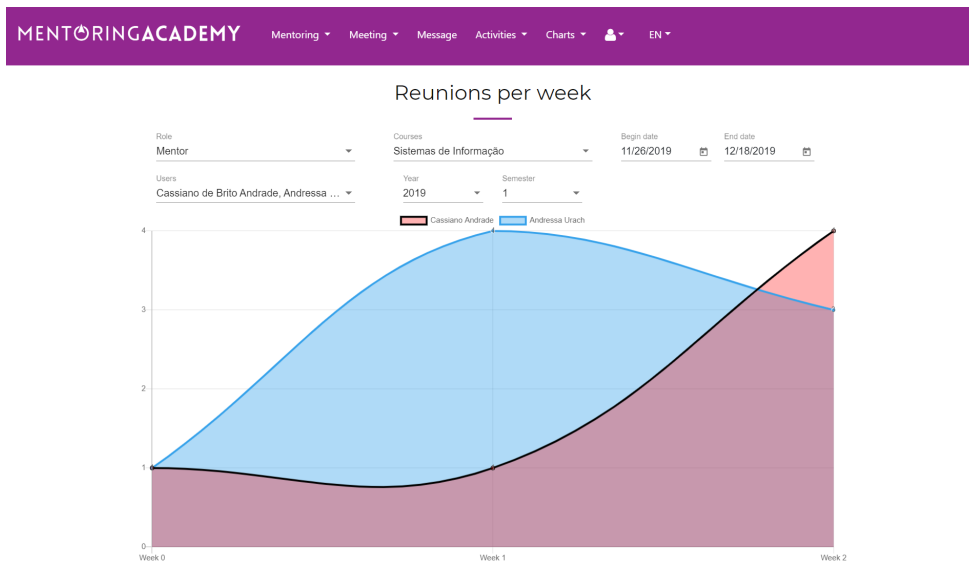


Figure 4.40: Screenshot regarding the graph about the reunions per week.

The coordinator can filter information by selecting the curricular unit, the start and end dates, the mentors or tutors, the year and the semester, then the graph will show the quantity of meetings the mentor or tutor had over the weeks.

4.2.17 System notification

The following figures exhibit the screenshots about system notifications regarding meeting scheduling and cancelling, which complies with the FR-68 (The system shall be able to notify session scheduling to interested parties) and FR-69 (The system shall be able to notify session cancellation to interested parties).

Figure 4.41 shows a screenshot of the email sent to a user's personal email about a meeting scheduled by the "Coordenador de Mentor" on the day "January 1st of 2020" at 9 o'clock.

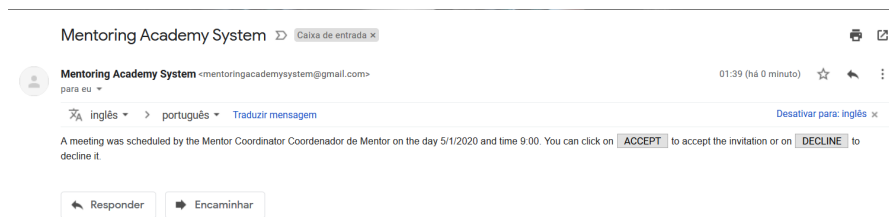


Figure 4.41: Screenshot regarding the system notification about the meeting scheduling functionality.

If the user clicks on the "ACCEPT" button, he/she will be redirected to the page shown by the Figure 4.42, which confirms that he/she accepted the invitation successfully.

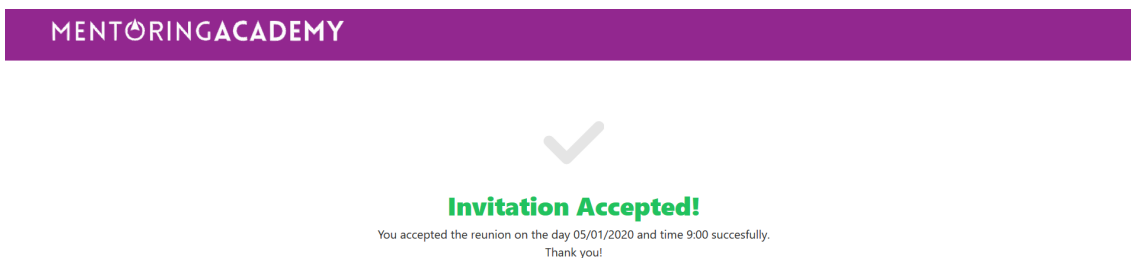


Figure 4.42: Screenshot regarding the system notification about the meeting scheduling functionality.

If the user clicks on the “DECLINE” button, he/she will be redirected to the page shown by the Figure 4.43, which confirms that he/she declined the invitation successfully.

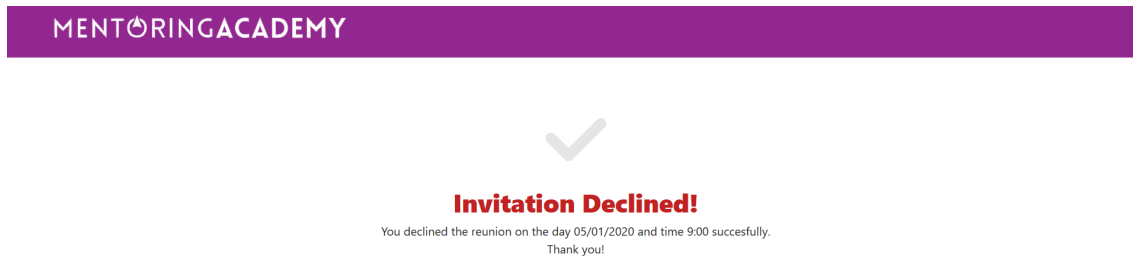


Figure 4.43: Screenshot regarding the system notification about the meeting scheduling functionality.

The user also receives a Short Message Service (SMS) notification in his/her personal cellphone number, which is sent using Twilio, a cloud communications platform as a service (CPaaS), like shown by Figure 4.44.

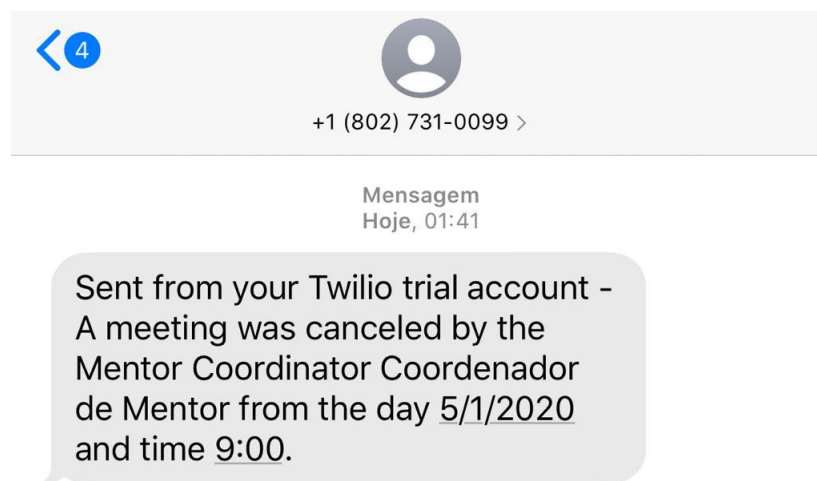


Figure 4.44: Screenshot regarding the system notification about the meeting scheduling functionality.

If the meeting’s host cancels the meeting, all of its guests will receive a notification by email and SMS like shown by Figure 4.45.

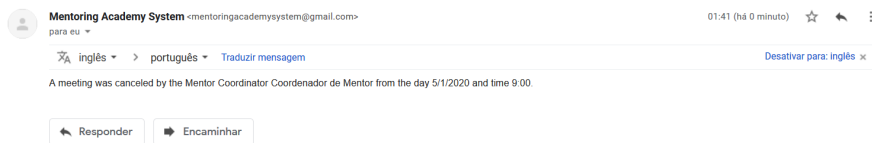


Figure 4.45: Screenshot regarding the system notification about the meeting cancellation functionality.

Figure 4.46 shows the system notification about the user registration on the system functionality. This email is sent once the user is registered into the Mentoring Academy web application using the importation functionality, further explained in subsection 4.2.18. This email contains information about the program and the user's login and password automatically generated.

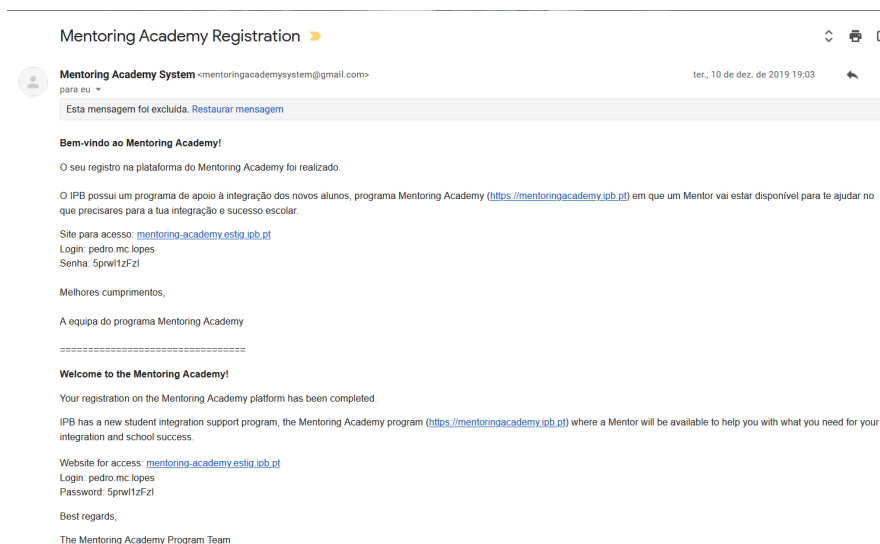


Figure 4.46: Screenshot regarding the system notification about the user registration functionality.

4.2.18 Import new users

Figure 4.47 presents the user importation functionality, which loads the new students recently enrolled in the IPB and still not registered in the Mentoring Academy web application. This importation can be made only by the administrator.

In the table below, it is shown all the users imported information, which are: (1) student code, (2) name, (3) gender, (4) city, (5) district, (6) nationality, (7) email, (8) phone number, (9) entry date, (10) school and (11) course. The user can manually change any of these information if desired. Additionally, the user must inform these new imported users roles and their first study year and semester.

Once everything is set, the administrator can import these new users by clicking on the “Send” button. Each one of the imported users will receive an email informing their username and password, like shown by Figure 4.46.

The screenshot shows the 'Import users' page in the MENTORING ACADEMY system. The page has a purple header with the logo and navigation menus. Below the header, the title 'Import users' is centered. A green button labeled 'Load new users' is positioned below the title. Underneath, there are three dropdown menus for 'Roles', 'First year of studies', and 'First semester of studies'. Two checkboxes are present: 'Ignore locality and district of foreign students' and 'Send e-mail to all users'. A table with 8 columns (Student code, Name, Gender, Locality, District, Nationality, Email, Phone number) displays 10 rows of user data. At the bottom of the table, there is a 'Send' button and a pagination indicator showing 'Items per page 10' and '1 - 10 of 169'.

Student code	Name	Gender	Locality	District	Nationality	Email	Phone number
43223	André Dialudi Sadi	Masculino	Bragança	Bragança	Angola	andre.ds.sadi@alunos.ipb.pt	
45538	Pratiksha Sharma	Feminino	São Teotónio	Évora	Portugal	pratiksha.sp.sharma@alunos	
44265	Arliton César Vaz de B	Masculino	Bragança	Bragança	Cabo Verde	arilton.cv.barros@alunos.ipb	
40603	Crisosto Mané Injai	Masculino	Bragança	Bragança	Guiné-Bissau	crisosto.ml.injai@alunos.ipb	
43351	Cássia Kanco Afonso	Feminino	Bragança	Bragança	Angola	cassia.ka.afonso@alunos.ip	
43385	Alexandre Edson Calul	Masculino	Bragança	Bragança	Angola	alexandre.ec.papa@alunos	
42991	Emanuel de Jesus Del	Masculino	Bragança	Bragança	Cabo Verde	emanuel.tj.santos@alunos.i	
43382	Deo Graças Massudi Tr	Masculino	Bragança	Bragança	Angola	deo.gm.toco@alunos.ipb.pt	
43345	Carlos Barbosa da Silv	Masculino	Bragança	Bragança	Brasil	carlos.bd.lima@alunos.ipb.p	
44296	Valdino Fernandes Ma	Masculino	Bragança	Estrangeiro	Cabo Verde	valdino.fm.mascarenhas@al	

Figure 4.47: Screenshot regarding the user importation functionality.

4.3 Unit tests

In computer programming, unit testing is a software testing method by which individual units of source code are tested to determine whether they are fit for use. Defects revealed by a unit test are easy to locate and relatively easy to repair.

Using unit testing allows us to test a much larger portion of the system code than that done manually. It is much easier to exercise all possible paths by unit testing than by manual testing.

That said, for this work we developed tests for two different units: (1) Angular components and (2) Angular services.

Figure 4.48 exemplifies one of the Angular's component unit test. The main objective of this type of unit test is to generate the respective component to see if there is any exception due its creation. If any problem pops up, the test will fail.

```
import { async, ComponentFixture, TestBed } from '@angular/core/testing';
import { AppModule } from 'src/app/app.module';
import { Style } from 'src/shared/models/style';
import { LanguageComponent } from './language.component';

describe('LanguageComponent', () => {
  let component: LanguageComponent
  let fixture: ComponentFixture<LanguageComponent>
  let languageDropdownStyle: Style = new Style("#92278F", "white")

  beforeEach(async(() => {
    TestBed.configureTestingModule({
      imports: [
        AppModule
      ]
    })
    .compileComponents()
  }))

  beforeEach(() => {
    fixture = TestBed.createComponent(LanguageComponent)
    component = fixture.componentInstance
    component.style = languageDropdownStyle
    fixture.detectChanges()
  });

  it('should create', () => {
    expect(component).toBeTruthy()
  })
})
```

Figure 4.48: Unit testing of an Angular component.

Figure 4.49 exhibits a service used to make HTTP requests regarding the Language controller. Figure 4.50 shows the unit test developed to test this service. The objective of this test is to check if the function `getData()` is running correctly, feeding it with a mock object. Therefore, no real HTTP request is made in the testing process.

```
import { HttpClient, HttpHeaders } from '@angular/common/http';
import { Injectable } from '@angular/core';

@Injectable({
  providedIn: 'root'
})
export class LanguageService {
  url = 'api/language'

  constructor(private http: HttpClient) { }

  getData() {
    return this.http.get(this.url, {
      headers: new HttpHeaders({
        'Authorization': "Bearer " + localStorage.getItem("jwt"),
        'Content-type': 'application/json'
      })
    })
  }
}
```

Figure 4.49: Angular service which is responsible to make a http request to get all the languages registered.

In order to run these tests, it was used Karma (version 4.0.1) as the task runner, which uses a configuration file in order to set the startup file, the reporters, the testing framework, the browser among other things. And, as the testing framework, it was used Jasmine (version 2.99.0).


```

import { HttpClientTestingModule, HttpTestingController } from '@angular/common/http/testing';
import { TestBed, TestBed } from '@angular/core/testing';
import { LanguageService } from './language.service';

describe('LanguageService', () => {
  let injector: TestBed
  let service: LanguageService
  let httpMock: HttpTestingController

  beforeEach(() => {
    TestBed.configureTestingModule({
      imports: [HttpClientTestingModule],
      providers: [LanguageService]
    })
    injector = TestBed.injector()
    service = injector.get(LanguageService)
    httpMock = injector.get(HttpTestingController)
  })

  describe('#getLanguages', () => {
    it('should return an Observable<Language[]>', () => {
      const mock = [
        { "id": 1, "name": "Português" },
        { "id": 2, "name": "English" },
      ]

      service.getData().subscribe((res) => {
        expect(res).toEqual(mock)
      })

      const mockReq = httpMock.expectOne(service.url)
      expect(mockReq.cancelled).toBeFalsy()
      expect(mockReq.request.responseType).toEqual('json')

      mockReq.flush(mock)
    })
  })
})

```

Figure 4.50: Unit testing of the Angular service responsible to make a http request to get all the languages registered.

It was developed 33 tests, like shown by Figure 4.51, and 33 tests ran successfully and no test failed.

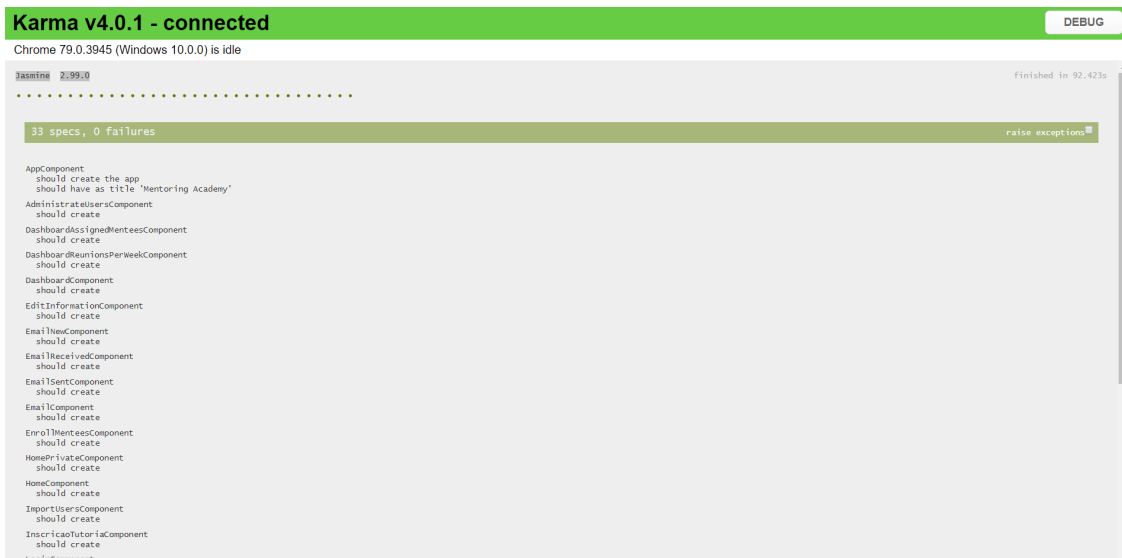


Figure 4.51: Running the unit tests developed for the Mentoring Academy web application.

4.4 System deployment on a Linux server

The system deployment occurred on a Ubuntu server 18.04 and it was divided into six parts: (1) back-end publishing, (2) front-end publishing, (3) MySQL configuration in the server, (4) database dump, (5) ASP.NET Core prerequisites installation in the server, (6) deployment transfer to the server and execution of the ASP.NET Core application and (7) Nginx installation and configuration. Each one of these steps were detailed in the sections below.

4.4.1 Back-end publishing

In order to publish the back-end code, it is needed to run the following command on the prompt from the development environment. This command packages the app into the directory “bin/release/netcoreapp2.1/publish”.

```
> dotnet publish --configuration Release
```

4.4.2 Front-end publishing

On this step, it is necessary to build Angular application in production mode. To do so, we need to run the command below. This command shall generate dist folder with output files.

```
> ng build --prod
```

Since we want to set Angular application to be served as default route of ASP.NET Core application, we copied all files from dist folder and pasted them in “bin/release/netcoreapp2.1/publish/wwwroot” ASP.NET Core folder.

4.4.3 MySQL configuration in the server

To install MySQL on Ubuntu 18.04, it is necessary to update the packages indexes from the server and then install the default package, as in the command described below.

```
$ sudo apt-get update
$ sudo apt-get install mysql-server
```

In order to ASP.NET Core application execute successfully, it is required to make MySQL case insensitive. With that purpose, we need to edit the following configuration file.

```
$ sudo nano /etc/mysql/my.cnf
```

Once the edition has started, we can add the following lines in the end of the file.

```
[mysqld]
lower_case_table_names = 1
```

After editing and saving the configuration file, we need to restart MySQL so the new configuration can take root, as described next.

```
$ sudo /etc/init.d/mysql restart
```

Now that MySQL is configured correctly, we need to create the database “ementoring” so we can move on to the database dump step.

```
$ /usr/bin/mysql -u root -p
$ mysql> CREATE DATABASE ementoring;
$ mysql> exit
```

4.4.4 Database dump

To successfully transfer the database from local development environment to the server, the first step is to create a dump, as shown below.

```
> mysqldump -u root -p ementoring > ementoring.sql
```

The second step is to transfer the file to the server. The method used in the command below is transferring the file through SSH connection.

```
> scp -r ./ementoring.sql a40272@193.136.195.28~/
```

The last step is to restore the dump building and populating the database “ementoring”.

```
$ mysql -u root -p ementoring < ./ementoring.sql
```

4.4.5 ASP.NET Core prerequisites installation in the server

Before installing .NET, it is mandatory to register the Microsoft key, to register the product repository and to install required dependencies. To do so, we can run the following commands.

```
$ wget -q https://packages.microsoft.com/config/ubuntu/16.04/packages-microsoft-prod.deb -O packages-microsoft-prod.deb
$ sudo dpkg -i packages-microsoft-prod.deb
```

Then we can update the products available for installation then install the .NET Core SDK, as described in the following commands.

```
$ sudo apt-get update
$ sudo apt-get install apt-transport-https
$ sudo apt-get update
$ sudo apt-get install dotnet-sdk-2.1
```

4.4.6 Deployment transfer to the server and execution of the ASP.NET Core application

The whole environment is ready for the application. So, we can finally transfer the whole publishing information to the server, as shown next.

```
> scp -r ./bin/release/netcoreapp2.1/publish/*
a40272@193.136.195.28:~/
```

Moreover, we can execute the application by running the following command, which is divided into two parts: (1) “nohup”, which is responsible for making the process independent from the terminal, so it will keep running after the terminal is closed and (2) “dotnet ementoring.dll”, which is responsible for executing the ASP.NET Core application.

```
$ nohup dotnet ementoring.dll
```

At this point, the application is only accessible on the localhost. In order to make it available on the Internet, we need to install and configure Nginx, as described in the posterior section.

4.4.7 Nginx installation and configuration

The following command is responsible for installing nginx.

```
$ sudo apt-get install nginx
```

To configure Nginx as a reverse proxy to forward requests to the ASP.NET Core app, we need to modify “/etc/nginx/sites-available/default”. So we have to open it in a text editor, like shown next.

```
$ sudo vim /etc/nginx/sites-available/default
```

Then we can replace its contents with the following text.

```
server {
    listen 80;

    location / {
        proxy_pass http://localhost:5000;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection keep-alive;
        proxy_set_header Host $host;
        proxy_cache_bypass $http_upgrade;
    }
}
```

```
}  
}
```

Finally, we shall reload nginx configuration by running the following command, then the application might be accessible through ASP.NET Core REST API.

```
$ sudo nginx -s reload
```

Chapter 5

Conclusions

Mentoring Academy program was created in the IPB to contribute to the integration of students entering higher education for the first time, as well as contributing to their academic and personal success. The Mentoring Academy program was completely designed and, in order for it to be effectively launched and attend the IPB students needs, it required an application that permitted its execution.

Motivated by this demand, we aimed at proposing a web application that complies with the Mentoring Academy requirements. The methodology adopted was divided as follows: (1) system modeling, (2) web application architecture definition, (3) development of the Mentoring Academy web application and (4) system deployment on a Linux server.

System modeling was the first step of the methodology and it was responsible for molding the system through requirements elicitation, use case diagrams and class diagram. Altogether, 9 personas of the Mentoring Academy were identified and 72 functional requirements were defined.

With this information, use case diagrams were designed, describing each persona's capacities to the system. Finally, the class diagram was created to describe the structure of the system by showing the system's classes, their attributes and the relationships among objects.

Thereafter, the web application architecture was planned. The model chosen for the web application was "one web server, one database", which is the simplest as well as the

least reliable web application component model. The architecture chosen was SPA, which allows dynamic interactions by means of providing updated content to the current page instead of loading the entire page after any user interaction. JWT was the method chosen for the authentication process.

JWT contains the following data: (1) username, (2) user identification and (3) the roles the user possesses. Once received this token, every request made by the client shall contain this token as part of the authorization field. In the server, this token will be used to identify who is the user responsible for the request and his/her roles. Consequently, each server function may contain two authorization methods: (1) token authorization and (2) role authorization. The first one will verify if the token is still valid and the second one will verify if the user can access the function based on his/her roles.

In terms of development, it was used Scrum as the development methodology, holding altogether eleven sprints lasting fifteen days each. Azure DevOps was used for features mapping, sprint planning and version control, therefore each task was associated to a branch, so its commits could be registered in the task's description.

Unit tests were developed regarding Angular components and Angular services in order to test individual units of source code before deploying them. At the end of the 11 sprints, 44 out of the 72 functional requirements planned were developed, tested and deployed successfully.

System deployment on a Linux server was the last step of the methodology. It was divided into 7 parts: (1) back-end publishing, (2) front-end publishing, (3) MySQL configuration in the server, (4) database dump, (5) ASP.NET Core prerequisites installation in the server, (6) deployment transfer to the server and execution of the ASP.NET Core application and (7) Nginx installation and configuration. After these steps were concluded, the Mentoring Academy web application was successfully running over the Linux server and available for access over the Internet.

As a result, this work generated a web application for the Mentoring Academy program, which is a proposed solution for supporting peer tutoring and peer mentoring coaching methods. The completion of this work permits the Mentoring Academy program to

be released into the academic environment.

Hereupon, the system provides students' integration, permitting them to have socio-academic inclusion and support in curricular units. That said, the program now can help the transitioning from high school to college of students, especially those who are the first in their family to attend college (first-generation) or are from low-income backgrounds.

The units tests developed for the web application proved promising in terms of (1) components generation and (2) HTTP request services, both exception-free. In addition, approximately 61% of the functional requirements defined in the system modeling were developed (44 out of 72) and some functionalities not previously defined were also developed, like (a) mentor, mentee, tutor and tutee tables with summed information and export to Excel functionality, (b) graphs about assigned mentees and reunions per week, (c) meeting invitation acceptance and decline through the personal email notification and (d) import new users recently enrolled in the IPB.

Therefore, this works left 28 functional requirements undone. Thus, as future works, these functional requirements could be developed and tested. In addition, there are some studies that could be made on the system regarding data mining over the peer mentoring and tutoring accomplishments in terms of (1) successful match up between "mentor and mentees" and "tutor and tutees" and (2) academic success after participating in the Mentoring Academy program.

The relevance of these studies is due to two reasons. On the one hand, a successful match up is important especially when dealing with students from all over the world attending the same school, because there are multiple cultural variables that could influence their academic success. On the other hand, accompany their academic success is important to see if the approach adopted by the Mentoring Academy program is correct as well as to measure the program results in the academic environment.

References

- [1] *Mentoring Academy Official Website the mentoring academy program*, <https://mentoringacademy.ipb.pt/>, Dec. 2019 (cit. on pp. 1, 20, 23).
- [2] D. Akobe, S. Popoola, A. Atayero, O. Oseni, and S. Misra, “A web framework for online peer tutoring application in a smart campus”, *International Conference on Computational Science and Its Applications*, Jun. 2019 (cit. on pp. 3, 20).
- [3] C. Spanorriga, P. Tsiotakis, and A. Jimoyiannis, “E-mentoring and novice teachers’ professional development: Program design and critical success factors”, *EdMedia and Innovate Learning*, 2018 (cit. on pp. 3, 19).
- [4] M. Wasilewski, M. Nonoyama, C. Dale, D. McKim, J. Road, D. Leasa, R. Goldstein, and L. Rose, “Development of a web-based peer support program for family caregivers of ventilator-assisted individuals living in the community: Protocol for a pilot randomized controlled trial”, *JMIR Research Protocols*, Feb. 2019 (cit. on p. 4).
- [5] T. Clemmensen and J. Nørbjerg, “digital peer-tutoring’. early results from a field evaluation of a ‘ux at work’ enhancing learning format”, *Cardiff Ubiquity Press*, 2019 (cit. on p. 4).
- [6] S. Thakare, S. Jadhav, I. Mane, S. Pawar, and A. Kulkarni, “Online mentoring system (an online mentor-student system)”, *International Journal of Engineering Trends and Technology*, Jan. 2019 (cit. on p. 4).

- [7] M. Evans and J. Moore, “Peer tutoring with the aid of the internet”, *British Journal of Educational Technology*, Jan. 2013 (cit. on p. 4).
- [8] L. Phiri, C. Meinel, and H. Suleman, “Peer tutoring orchestration: Streamlined technology-driven orchestration for peer tutoring”, *Proceedings of 9th International Conference on Computer Supported Education*, Apr. 2017 (cit. on p. 4).
- [9] Y. Xing, J. Huang, and Y. Lai, “Research and analysis of the front-end frameworks and libraries in e-business development”, *Proceedings of the 2019 11th International Conference on Computer and Automation Engineering*, Feb. 2019 (cit. on pp. 5, 7–11, 16).
- [10] M. Ramos, M. T. Valente, and R. Terra, “Angularjs performance: A survey study”, *IEEE Software*, Mar. 2018 (cit. on p. 6).
- [11] M. Omari, M. Erramdani, and S. Filali, “Getting model of mvvm pattern from uml models”, *Proceedings of the International Conference on Industrial Engineering and Operations Management*, Apr. 2017 (cit. on p. 6).
- [12] J. S. Sabao and A. M. Lacorte, “Cloud-based real-time bulletin board posting using angularjs framework”, *International journal of simulation: systems, science technology*, 2019 (cit. on pp. 6, 9).
- [13] R. McLeod and G. Schell, *Management Information Systems*, 9th. Upper Saddle River, New Jersey, USA: Pearson/Prentice Hall, 2004, ISBN: 9780131408357 (cit. on p. 6).
- [14] —, *Management Information Systems*, 10th. Upper Saddle River, New Jersey, USA: Pearson/Prentice Hall, 2007, ISBN: 9780131889187 (cit. on p. 6).
- [15] Z. Chen, *Html5 hybrid mobile application: Building mobile applications using web technologies with ionic*, Kokkola, Central Ostrobothnia, Finland, 2018 (cit. on p. 7).
- [16] *Angular Official Blog angularjs to angular concepts: Quick reference*, <https://angular.io/guide/ajs-quick-reference>, Feb. 2019 (cit. on p. 7).

- [17] C. Gackenheim, “Introducing flux: An application architecture for react”, *Introduction to React*, 2015 (cit. on p. 8).
- [18] B. Eisenman, *Learning React Native: Building Native Mobile Apps with JavaScript*, 2nd. Champaign, Illinois, USA: O’Reilly Media Inc, 2015, ISBN: 9781491989142 (cit. on pp. 8, 9).
- [19] O. Pastushenko, “Uxgraph-vue.js library with predefined d3 graphs”, Apr. 2017 (cit. on p. 10).
- [20] *Vue.js Official Blog reactive data binding*, <https://v1.vuejs.org/guide/overview.html>, Feb. 2019 (cit. on p. 10).
- [21] N. H. da Silva and M. Carvalho, *Análise das tecnologias requisitadas em vagas de desenvolvedor web de acordo com a localidade das empresas*, João Pessoa, Paraíba, BR, Nov. 2017 (cit. on p. 11).
- [22] *HotFrameworks find your new favorite web framework*, <https://hotframeworks.com/>, Jan. 2019 (cit. on pp. 11, 12, 17).
- [23] *Microsoft Official Blog common language runtime (clr) overview*, <https://docs.microsoft.com/en-us/dotnet/standard/clr>, Feb. 2019 (cit. on p. 12).
- [24] *Microsoft Official Blog visão geral sobre o asp.net core mvc*, <https://docs.microsoft.com/pt-br/aspnet/core/mvc/overview?view=aspnetcore-3.1>, Feb. 2019 (cit. on p. 13).
- [25] *GitHub asp.net*, <https://github.com/aspnet>, Feb. 2019 (cit. on p. 13).
- [26] K. Kronis and M. Uhanova, “Performance comparison of java ee and asp.net core technologies for web api development”, *Applied Computer Systems*, May 2018 (cit. on pp. 13, 17).
- [27] X. Mao, *Comparison between symfony, asp.net mvc, and node.js express for web development*, Fargo, North Dakota, USA, 2018 (cit. on p. 13).
- [28] A. Freeman, *Pro ASP.NET MVC 4*, 4th. New York, New York, EUA: Apress, 2012, ISBN: 978-1-4302-4236-9 (cit. on pp. 13, 14).

- [29] T. Crawford and T. Hussain, “A comparison of server side scripting technologies”, *SERP’17 - The 15th Int’l Conf on Software Engineering Research and Practice*, May 2017 (cit. on pp. 13, 14).
- [30] *Ruby on Rails Official Blog ruby on rails guides (v5.0.0.1)*, <https://guides.rubyonrails.org/v5.0.0.1/>, Feb. 2019 (cit. on p. 13).
- [31] T. Muhammed, R. Mehmood, E. Abozinadah, and S. Sharaf, “Selecweb: A software tool for automatic selection of web frameworks”, *EAI/Springer Innovations in Communication and Computing*, Jun. 2019 (cit. on p. 14).
- [32] A. Holovaty and J. Kaplan-Moss, *The Definitive Guide to Django*, 2nd. New York, New York, EUA: Apress, 2009, ISBN: 978-1-4302-1937-8 (cit. on p. 14).
- [33] A. Poudel, *A comparative study of project management system web applications built on asp.net core and laravel mvc frameworks*, Setor Cloud, Minnesota, USA, Dec. 2018 (cit. on p. 15).
- [34] L. Alfat, A. Triwiyatno, and R. R. Isnanto, “Sentinel web: Implementation of laravel framework in web-based temperature and humidity monitoring system”, *Second International Conference on Information Technology, Computer, and Electrical Engineering*, 2015 (cit. on p. 15).
- [35] M. Stauffer, *Laravel: Up and running: A framework for building modern PHP Apps*, 1st. Sebastopol, California, USA: O’Reilly Media, Dec. 2016, ISBN: 1491936088 (cit. on p. 15).
- [36] *Stefankrause Developer Team results for js web frameworks benchmark*, <https://www.stefankrause.net/js-frameworks-benchmark6/webdriver-ts-results/table.html>, Dec. 2019 (cit. on p. 16).
- [37] *GitHub asp.net core*, <https://github.com/aspnet/AspNetCore>, Feb. 2019 (cit. on p. 17).
- [38] Z. Liu and B. Gupta, “Study of secured full stack web development”, *EPiC Series in Computing*, 2019 (cit. on p. 17).

- [39] *Microsoft Official Blog overview of asp.net core security*, <https://docs.microsoft.com/en-us/aspnet/core/security/?view=aspnetcore-2.1>, Feb. 2019 (cit. on p. 17).
- [40] *Microsoft Official Blog announcing asp.net core 1.0*, <https://devblogs.microsoft.com/aspnet/announcing-asp-net-core-1-0/>, Feb. 2019 (cit. on p. 17).
- [41] *Microsoft Official Blog introduction to asp.net core*, <https://docs.microsoft.com/en-us/aspnet/core/?view=aspnetcore-3.1>, Feb. 2019 (cit. on p. 18).
- [42] S. Plaskett, D. Bali, M. Nakkula, and J. Harris, “Peer mentoring to support first-generation low-income college students”, *Phi Delta Kappan*, Mar. 2018 (cit. on p. 19).
- [43] M. Jacobi, “Mentoring and undergraduate academic success: A literature review”, *Review of Educational Research*, 1991 (cit. on p. 19).
- [44] E. Plotnick, “Trends in educational technology”, *ERIC Digest*, 1996 (cit. on p. 20).
- [45] C. Olmscheid, *The effectiveness of peer tutoring in the elementary grades*, Apr. 1999 (cit. on p. 20).
- [46] H. Pigott, J. Fantuzzo, and P. Clement, “The effects of reciprocal peer tutoring and group contingencies on the academic performance of elementary school children”, *Journal of Applied Behavior Analysis*, 1986 (cit. on p. 20).
- [47] D. Miller, K. Topping, and A. Thurston, “Peer tutoring in reading: The effects of role and organization on two dimensions of self-esteem”, *British Journal of Educational Psychology*, Sep. 2010 (cit. on p. 20).
- [48] P. A. G. Permana, “Scrum method implementation in a software development project management”, *International Journal of Advanced Computer Science and Applications*, 2015 (cit. on pp. 23, 25).
- [49] R. Martin, *Agile software development : principles, patterns, and practices*, 1st. Upper Saddle River, New Jersey, USA: Pearson, 2003, ISBN: 0135974445 (cit. on p. 23).

- [50] A. Pham and P.-V. Pham, *Scrum in action Agile software project management and development*, 1st. Boston, Massachusetts, USA: Course Technology PTR, 2011, ISBN: 143545913X (cit. on p. 23).
- [51] L. Rising and N. Janoff, “The scrum software development process for small teams”, *Institute of Electrical and Electronics Engineers*, 2000 (cit. on p. 24).
- [52] R. Nabila, S. Oktaviana, and A. Hidayati, “The development of vocational high school information system using angularjs and scrum”, *Journal of Physics: Conference Series*, 2019 (cit. on p. 24).
- [53] J. Sutherland and K. Sutherland, *The scrum guides*, <https://www.scrumguides.org/docs/scrumguide>, Jul. 2016 (cit. on p. 24).