TÉCNICO LISBOA

IST Computer Science and Engineering Doctoral Degree Information Management System (DD-IMS)

Hugo Alexandre Resende da Rocha

Thesis to obtain the Master of Science Degree in

Information Systems and Computer Engineering

Supervisor: Prof. Helena Isabel de Jesus Galhardas

Examination Committee

Chairperson: Prof. Francisco António Chaves Saraiva de Melo
Supervisor: Prof. Helena Isabel de Jesus Galhardas
Member of the Committee: Prof. Fernando Henrique Côrte-Real Mira da Silva

January 2021

Acknowledgments

First, I would like to thank my family and friends. This thesis would not have been possible without their caring and support.

I would like to thank my thesis supervisor, Prof. Helena Galhardas, for giving me the opportunity to be part of this project and for the support given during this thesis development.

I would also like to thank the project stakeholders (students, faculty members, Computer Science and Engineering administrative staff, Post-Graduate Area administrative staff), that had a crucial role during the requirements gathering phase.

I would also like to thank all the Computer Science and Engineering PhD students, faculty members, and administrative staff members who participated in the DD-IMS usability validation, for their help and valuable feedback to improve the system.

Finally, I would like to thank the IST IT Services, in particular, Prof. Fernando Mira da Silva and Eng. Jorge Goulart, whose technical support about the Fenix API was fundamental for the implementation of some of the DD-IMS modules.

Resumo

Nos dias de hoje, um vasto número de operações relacionadas com a gestão de informação em instituições de Ensino Superior são tratados por sistemas de informação. No Instituto Superior Técnico (IST), desenvolveu-se o Fenix, um sistema de gestão de informação académica, que tem a capacidade de gerir informação de modo a dar suporte a todos os processos relacionados com os graus académicos da Universidade. Contudo, o sistema Fenix apresenta limitações ao nível da gestão de informação relacionada com os graus de Doutoramento. De modo a colmatar estas limitações, desenhou-se e implementou-se um Sistema de Gestão de Informação para os Doutoramentos (DD-IMS), que gere a informação relativa aos alunos do Doutoramento em Engenharia Informática e de Computadores (DEIC). A arquitetura do DD-IMS consiste numa arquitetura cliente-servidor, composta por três camadas que estão separadas fisicamente: (i) uma base de dados; (ii) um serviço de um servidor web; (iii) uma aplicação cliente. O DD-IMS comunica com o sistema Fenix de modo a tirar vantagem da informação que lá está armazenada. O sistema foi validado em termos de funcionalidade, usabilidade e performance. Os utilizadores demonstraram um elevado nível de satisfação com a usabilidade do DD-IMS. Em termos de performance, os resultados obtidos mostram que, com o aumento do número de utilizadores a utilizarem o sistema em simultâneo, o DD-IMS apresentou um crescimento controlado a nível da taxa de erros e do tempo médio de resposta, mantendo assim uma performance aceitável quando usado por um elevado número de utilizadores em simultâneo.

Palavras-chave: Sistema de Gestão de Informação; Gestão de Alunos de Doutoramento; Sistemas de Gestão para o Ensino Superior; Engenharia de Software

Abstract

As a result of the development of modern Information and Communication Technologies (ICT), the academic world has entered the information society era. Nowadays, many operations regarding Higher Education data management are handled by ICT Systems. At Instituto Superior Técnico (IST), the Fenix system, an Academic Information Management System capable of managing data and supporting all processes related to the university degrees, was developed. However, the Fenix system has limitations in terms of managing data regarding Doctoral Degrees. We designed and implemented a Doctoral Degree Information Management System (DD-IMS) that manages data concerning the IST Computer and Science Engineering (CSE) Department PhD students. The architecture for the DD-IMS is a client-server architecture composed of three tiers that are physically separated (three-tier architecture): (i) a database to store the relevant data about PhD processes, (ii) a web server to contain the business logic, and (iii) a client, whose functionality is accessible via a web browser. DD-IMS communicates with the Fenix system to take advantage of the data that is already stored there. We conducted an experimental validation to evaluate the DD-IMS at three levels: functionality, usability, and performance. Users showed a high level of satisfaction with the DD-IMS usability. Regarding performance, the results obtained show that, with the increase of concurrent users, the DD-IMS had a sustained growth in terms of error rate and average response time, maintaining an acceptable performance with a large number of concurrent users.

Keywords: Information Management System; PhD Students Management; Higher Education Management Systems; Software Engineering

Contents

Acknowledgments . . . iii
Resumo . . . v
Abstract . . . vii
List of Figures . . . xv
List of Tables . . . xvii

1 Introduction **1**
1.1 Main Processes of an IST Doctoral Program . . . 1
1.2 Problem . . . 3
1.3 Proposed Solution . . . 4
1.4 Contributions . . . 4
1.5 Outline . . . 5

2 Background **6**
2.1 Application . . . 6
2.2 Registration . . . 8
2.3 Supervision . . . 8
2.4 Study Plan . . . 8
2.5 Thesis Proposal . . . 9
2.6 Thesis . . . 10

3 Requirements **13**
3.1 Fenix Limitations . . . 13
3.2 Requirements . . . 15

4 Related Work **19**
4.1 Higher Education Information Management Systems . . . 19
4.1.1 FenixEdu . . . 20
4.1.2 Manipal University Jaipur - Online Faculty Information System . . . 20
4.1.3 University Study-Oriented System . . . 21
4.1.4 HISinOne . . . 22
4.1.5 Fedena . . . 22

4.1.6 Salesforce for Education . . . 23
4.1.7 Banner by Ellucian . . . 23
4.1.8 Discussion . . . 24
4.2 Web Application Development Frameworks . . . 25
4.2.1 Django . . . 26
4.2.2 Flask . . . 27
4.2.3 Express.js . . . 27
4.2.4 Ruby on Rails . . . 27
4.2.5 Spring Boot . . . 28
4.2.6 Laravel . . . 28
4.2.7 Discussion . . . 28

5 Computer Science and Engineering Doctoral Degree Information Management System 30
5.1 Architecture . . . 32
5.1.1 Client . . . 33
5.1.2 Web Server . . . 33
5.1.3 Database . . . 34
5.2 User Roles and Permissions . . . 34
5.3 Evaluation of Application Module . . . 35
5.3.1 Use Cases . . . 36
5.3.2 Class Diagram . . . 37
5.3.3 Relational Model . . . 38
5.3.4 User Interface . . . 40
5.4 Registration Module . . . 41
5.4.1 User Interface . . . 42
5.5 Study Plan Module . . . 43
5.5.1 User Interface . . . 44
5.6 Supervision Module . . . 45
5.6.1 User Interface . . . 47
5.7 Curricular Plan Module . . . 48
5.7.1 User Interface . . . 48
5.8 Thesis Proposal (CAT) Module . . . 48
5.8.1 User Interface . . . 50
5.9 Thesis Module . . . 51
5.9.1 User Interface . . . 52
5.10 Statistics Module . . . 53
5.10.1 User Interface . . . 54
5.11 Doctoral Program Management Module . . . 55
5.11.1 User Interface . . . 55

5.12 Fenix Connection . . . 56
5.13 Implementation Details . . . 57
5.13.1 Common Functionalities . . . 58
5.13.2 Deployment . . . 58

6 Experimental Validation **60**
6.1 Functionality . . . 60
6.2 Usability . . . 61
6.2.1 Formative Evaluation . . . 61
6.2.2 Summative Evaluation . . . 63
6.2.3 Results . . . 66
6.3 Performance . . . 75
6.3.1 Measures . . . 75
6.3.2 Planning . . . 75
6.3.3 Results . . . 76

7 Final Remarks **80**
7.1 Summary . . . 80
7.2 Future Work . . . 82

Bibliography **87**

A Functionality: UML Use Cases **88**
A.1 Registration . . . 88
A.2 Study Plan . . . 89
A.3 Supervision . . . 89
A.4 Curricular Plan . . . 90
A.5 Thesis Proposal (CAT) . . . 91
A.6 Thesis . . . 92
A.7 Statistics . . . 92
A.8 Doctoral Program Management . . . 93

B Database: UML Class Diagrams and Relational Model **94**
B.1 Class Diagram . . . 94
B.1.1 Users . . . 94
B.1.2 Registration . . . 95
B.1.3 Study Plan . . . 97
B.1.4 Supervision . . . 100
B.1.5 Curricular Plan . . . 101
B.1.6 Thesis Proposal (CAT) . . . 102
B.1.7 Thesis . . . 104

B.2 Relational Model . . . 106
B.2.1 Users . . . 106
B.2.2 Registration . . . 110
B.2.3 Study Plan . . . 111
B.2.4 Supervision . . . 113
B.2.5 Thesis Proposal (CAT) . . . 114
B.2.6 Thesis . . . 116

C Developer Guide **118**
C.1 Setup . . . 118
C.2 Project Structure . . . 119
C.2.1 Adding a new module to DD-IMS . . . 123
C.3 Deployment . . . 123
C.3.1 Update the system . . . 128
C.3.2 Create data backups / Migrate data from one database to another . . . 128
C.3.3 Logs . . . 129
C.4 Unit Tests . . . 129

D Usability Evaluation - Tasks **130**
D.1 Student Tasks . . . 130
D.2 Supervisor Tasks . . . 132
D.3 Coordinator Tasks . . . 133
D.4 CC3C Member Tasks . . . 134
D.5 CCP Member Tasks . . . 134

E Usability Evaluation - Satisfaction Questionnaire **136**

List of Figures

1.1 Main data flows involved in the IST Doctoral Program. 3

2.1 Student's academic path in the CSE Doctoral Program. 7

5.1 Integration of the DD-IMS with the main data flows involved in the CSE Doctoral Degree. . 31
5.2 Architecture of the System. 32
5.3 Layered architecture of each module of the DD-IMS 33
5.4 Evaluation of Applications Process. 36
5.5 Use Cases Diagram for Evaluation of Applications Module. 37
5.6 Class Diagram for Evaluation of Applications module. 38
5.7 Screenshot of the main page of Evaluation of Applications tab. 41
5.8 Registration Process. 42
5.9 Screenshot of the Registration form wizard. 42
5.10 Study Plan document in PDF format. 44
5.11 Study Plan Process. 44
5.12 Screenshot of the Study Plan form wizard. 45
5.13 Supervision document in PDF format. 46
5.14 Supervision Process. 46
5.15 Screenshot of the Supervision Team form. 48
5.16 Screenshot of the Study Plan left side menu option, containing the student's curricular plan. 49
5.17 Thesis Proposal (CAT) Jury Process. 49
5.18 Thesis Proposal (CAT) Defense Process. 50
5.19 Screenshot of the CC3C Feedback left side menu option, inside the thesis proposal jury page. 51
5.20 Thesis Jury Process. 52
5.21 Thesis Grade Process. 52
5.22 Screenshot of the Thesis left side menu option, inside the student page. 53
5.23 Screenshot of the UI designed for the Statistics module. 54
5.24 Screenshot of the main page of Management tab. 56
5.25 Deployment Architecture. 59

6.1 a) Number of users that concluded each of the Student tasks; b) Total number of errors committed in each of the Student tasks. 67
6.2 Average, minimum and maximum execution times for each of the Student tasks. 68
6.3 a) Number of users that concluded each of the Supervisor tasks; b) Total number of errors committed in each of the Supervisor tasks. 68
6.4 Average, minimum and maximum execution times for each of the Supervisor tasks. 69
6.5 a) Number of users that concluded each of the Staff tasks; b) Total number of errors committed in each of the Staff tasks. 69
6.6 Average, minimum and maximum execution times for each of the Staff tasks. 70
6.7 a) Number of users that concluded each of the Coordinator tasks; b) Total number of errors committed in each of the Coordinator tasks. 70
6.8 Average, minimum and maximum execution times for each of the Coordinator tasks. . . . 71
6.9 a) Number of users that concluded each of the CC3C members tasks; b) Total number of errors committed in each of the CC3C members tasks; c) Average, minimum and maximum execution times for each of the CC3C members tasks. 71
6.10 a) Number of users that concluded each of the CCP members tasks; b) Total number of errors committed in each of the CCP members tasks; c) Average, minimum and maximum execution times for each of the CCP members tasks. 71
6.11 a) Users age distribution; b) Users gender distribution. 72
6.12 Prediction of the modules that will be used more often by users. 72
6.13 SUS scores distribution. 73
6.14 Error rate with variable number of actions. 77
6.15 Error rate with variable number of users. 77
6.16 Average response time with variable number of actions. 77
6.17 Average response time with variable number of users. 78
6.18 Throughput with variable number of actions. 78
6.19 Throughput with variable number of users. 78

A.1 Use Cases Diagram for Registration Module. 88
A.2 Use Cases Diagram for Study Plan Module. 89
A.3 Use Cases Diagram for Supervision Module. 90
A.4 Use Cases Diagram for Curricular Plan Module. 91
A.5 Use Cases Diagram for Thesis Proposal (CAT) Module. 91
A.6 Use Cases Diagram for Thesis Module. 92
A.7 Use Cases Diagram for Statistics Module. 93
A.8 Use Cases Diagram for Doctoral Program Management Module. 93

B.1 User Class Diagram. 95
B.2 Class Diagram for Registration module. 96
B.3 Class Diagram for Study Plan module. 99

B.4 Class Diagram for Supervision module. 100
B.5 Class Diagram for Curricular Plan module. 102
B.6 Class Diagram for Thesis Proposal module. 103
B.7 Class Diagram for Thesis module. 105

List of Tables

4.1 Comparison of Information Management Systems that were built from scratch for Higher Education institutions (described in Sections 4.1.1 to 4.1.4). 24
4.2 Comparison of ERPs for Higher Education institutions (described in Sections 4.1.5 to 4.1.7). 25
4.3 Comparison of web application frameworks. 29

5.1 Association between user roles and permission groups. 35
5.2 Helper Functions and Classes 58

Chapter 1

Introduction

As a result of the development of modern Information and Communication Technologies, the academic world has entered the information society era. Nowadays, many operations regarding data management in Higher Education institutions (e.g., enrolling in courses or exams, publishing grades, etc.) that were not performed via IT Systems in the past, are now handled by these systems. The basic idea behind all these systems is to enable entering the relevant data, storing it, and then displaying it as required by different types of users, namely students, teachers, coordinators, and administrative staff.

Instituto Superior Técnico[1] (IST) offers a broad set of degrees that cover the most important scientific areas related to Engineering, Science, Technology, and Architecture. Information systems are required to manage data that supports all processes related to those degrees. Thus, the Fenix system[2] was developed to be a comprehensive academic Information Management System. Regarding Bachelor and Master degrees, it provides several functionalities that enable students, teachers, degree coordinators, and administrative staff to effectively manage data. In what concerns data about PhD processes regarding courses and students, Fenix also gives support but in a limited way.

1.1 Main Processes of an IST Doctoral Program

Doctoral Programs involve a set of milestones that must be achieved by the students, from their application to the thesis defense, which is the final step in a Doctoral Program. The result is a considerable amount of data exchanged between students, coordinators, supervisors, and administrative staff.

Currently, the Fenix system gives support to the management of IST Doctoral processes for students, coordinators, supervisors, departments administrative staff, Post-Graduation Area[3] (PGA) administrative staff, members of the Computer Science and Engineering[4] (CSE) Department Coordinating Committee for the 3rd cycle[5] (CC3C), and members of the CSE *Pedagogical/Scientific Council* (CCP from the Portuguese *Conselho Científico-Pedagógico*).

[1] https://tecnico.ulisboa.pt/
[2] https://fenix.tecnico.ulisboa.pt/
[3] https://posgraduacao.tecnico.ulisboa.pt/en/o-nucleo-de-pos-graduacao/
[4] https://fenix.tecnico.ulisboa.pt/departamentos/dei/o-dei
[5] https://fenix.tecnico.ulisboa.pt/departamentos/dei/organizacao

Students have two views (inside a tab called Student) of the data regarding their Doctoral Program process: (i) the Curriculum (*Currículo*) view where the students can visualize the courses they have already concluded and the courses they are enrolled in the current semester; and (ii) the Doctorates (*Doutoramentos*) view where a student can visualize her process and the associated documents, in PDF format, submitted during her application. The Doctoral Program *coordinator* has two tabs that allow her to visualize the data regarding a PhD student: one called Coordinator that contains the PhD processes (*Processos de Doutoramento*) view where documents, in PDF format, regarding the application and student progress can be visualized; another one called Coordinator Portal (*Portal do Coordenador*) where the courses where the student is enrolled in or has taken can be visualized. *Supervisors* only have a tab called Teaching (*Docência*), where they can visualize the doctoral processes of students they are supervising. They also have access to all their students' documents in PDF format analogously to the Coordinator. Regarding the *CSE department administrative staff*, they have access to three tabs that contain information about PhD students: (i) Coordinator, (ii) Academic Administration (*Administração Académica*), and (iii) Academic Services (*Secretaria Académica*). The Coordinator tab gives access to the PhD Processes view, where documents, in PDF format, regarding the application and student progress can be visualized. The Academic Administration tab allows administrative staff to visualize an aggregated view of data and statistics regarding students (e.g., students per degree, students per course). The Academic Services tab gives access to the students' personal information (e.g., birth date, address, etc.). Regarding the *PGA administrative staff*, three tabs are available: (i) Coordinator, which contains the PhD processes view, where documents, in PDF format, regarding the application and student progress can be visualized and the Enrollment (*Inscrições*) view, where the courses where the student is enrolled in or has taken can be visualized; (ii) Academic Administration (also available for the CSE department administrative staff), where some statistics regarding students can be visualized; and (iii) Academic Services tab, that gives access to the students' personal information (e.g., birth date, address, etc.). The *CC3C members* also have a tab called Coordinator Portal, as it happens with the coordinator, that gives access to the courses where the students are enrolled in or have taken. They also have the Teaching tab, where they can visualize not only their students' processes as supervisors but also other students' processes as a CC3C member.

Figure 1.1 shows the main data flows involved in IST Doctoral Programs. Departments[6] send the original version of PDF documents (in paper format) and forms (e.g., study plan, jury proposals, etc.) related to Doctoral Programs processes to the PGA for its validation and approval (it is the responsibility of the PGA coordinator to submit PhD documents to the IST Scientific Council for approval). The approved documents are digitally stored in PDF format in the Fenix system, to be further visualized by coordinators, supervisors, students, or members of the administrative staff when requested. For example, when a student chooses her study plan (i.e., the courses she wants to enroll), it must be analyzed and approved (signed) by the student's supervisor(s), by the department coordinator, and then sent to the PGA that submits it to the Scientific Council. This process is not automatically supported, as it is done by hand. After this sequence of approvals, the document is stored in the Fenix system, in the

[6] In the figure, we only illustrate the CSE department but it is valid for all the other IST departments

student's PhD processes section. Another example is the thesis defense process, where the documents submitted by the students (e.g., thesis dissertation, student's CV, thesis submission form, etc.) and the thesis defense jury proposal go through the coordinator, supervisor(s), PGA, and Scientific Council, to be validated and approved. The Fenix system has the capability of sending the following alerts to the CSE Doctoral Degree actors: (i) a candidate has submitted her application; (ii) request constitution of the Thesis Proposal defense and Thesis defense jury; (iii) a student has submitted her thesis; and (iv) the Thesis Proposal defense and Thesis defense was scheduled. Moreover, the CSE department administrative staff uses several spreadsheets, as a complement to the Fenix system, to store data regarding the PhD students.

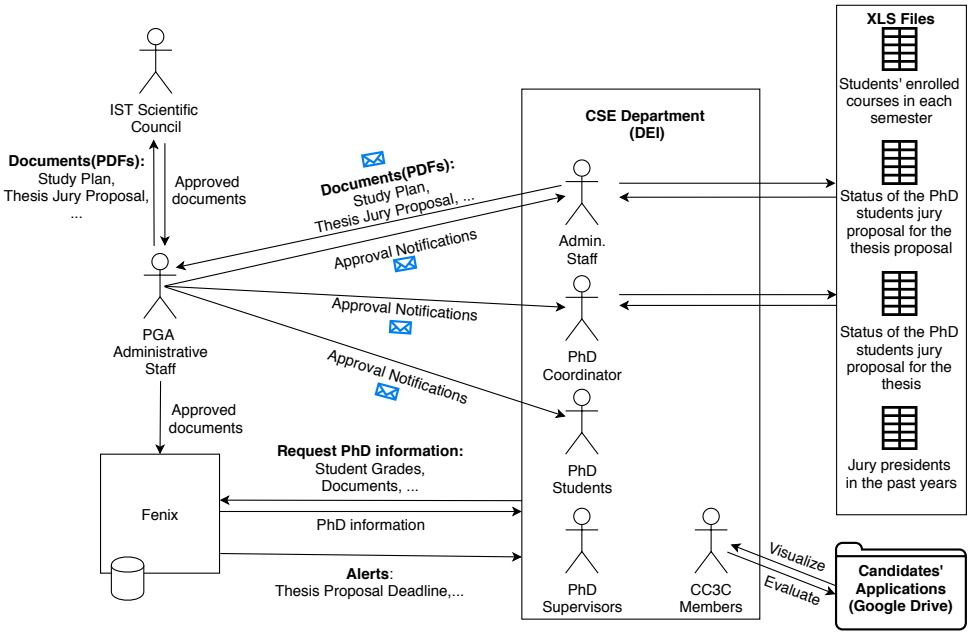


Figure 1.1: Main data flows involved in the IST Doctoral Program.

1.2 Problem

The Fenix system has limitations in terms of managing data regarding Doctoral Degrees. First, the existence of multiple views available to the coordinator, administrative staff, CC3C members, and students makes it very hard to manage and visualize data due to the existence of multiple points of access to students' data. Moreover, it is very difficult and sometimes impossible to obtain statistics about the degree (e.g., how many students are enrolled, how many have defended their Thesis Proposal in a given period, etc.). In particular, this feature is critical for coordinators that are responsible for periodically producing reports with statistics regarding the Doctoral Degree activity. In some cases, the results retrieved through the various views may even be inconsistent. For example, when PGA administrative staff access the Fenix Academic Services and Academic Administration views, different results may be retrieved regarding the list of active PhD students (the number of listed students differs in these two views). Furthermore, there are some types of data that are not accessible for certain user types (e.g.,

a supervisor does not have access to her students' curricular plan, which contains the courses in which the students have enrolled in and the grades obtained in those courses; students do not know how much time is left for important deadlines such as the thesis proposal submission).

Second, at the level of the CSE Doctoral Degree, there is an emergent need to have a software system that has the capability of providing answers to a set of questions that are typically posed by students, supervisors, coordinator, CC3C members, and department administrative staff (described in more detail in Chapter 3) that, currently, cannot be answered (e.g., how much time, in average, do students take to conclude their Doctoral Program; for a given supervisor, who are her students, etc.).

1.3 Proposed Solution

Due to the limitations presented in Section 1.2, the IST CSE department decided that it was crucial to have an Information System, called IST CSE Doctoral Degree Information Management System (DD-IMS[7]), that can efficiently and effectively support the storage, management, and visualization of data concerning the IST CSE Doctoral Program students. The system is expected to be a single point of access to PhD students' data in the CSE department. It communicates with the Fenix system to take advantage of the data that can be retrieved through the Fenix API[8].

The actors involved in this Information System are CSE students, CSE students' supervisors, the CSE Doctoral Degree coordinator, the CSE department administrative staff, the members of the CSE CC3C, the members of the CSE CCP, and the members of the CSE Scientific Committee. Along with the PGA administrative staff, those are the project stakeholders. The PGA staff does not use the system directly, so it does not belong to the set of actors.

1.4 Contributions

Having in mind the objectives defined at the beginning of this thesis, the principal contributions of this thesis are:

- **Requirements gathering** through interviews with the different stakeholders: students, supervisors, CSE department administrative staff, coordinator, CC3C members, and PGA administrative staff. A document with the gathered requirements was published as a technical report[11].
- **Design and implementation of a relational database** that stores all the relevant data concerning active[9] CSE PhD students in a structured way, thus enabling to monitor each student's progress and to obtain statistics about the behavior of the PhD Doctoral program.
- **Design and implementation of the DD-IMS**, a system that manages the information regarding CSE PhD students. The DD-IMS is composed of the following modules: (i) *Evaluation of Appli-*

[7] Throughout this document, we will refer to the IST CSE Doctoral Degree Information Management System as DD-IMS

[8] http://fenixedu.org/dev/api/

[9] A student is considered to be active since the moment she is admitted in the Doctoral Degree until the moment when her thesis is defended and approved.

cations, that supports the evaluation of the candidates' applications; (ii) *Registration*, supporting the students' registration; (iii) *Study Plan*, that supports the students' study plan management; (iv) *Supervision*, that supports the students' supervision team management; (v) *Curricular Plan*, that supports the students' data regarding their enrolled courses and the courses that they have taken; (vi) *Thesis Proposal*, supporting the Thesis Proposal management, i.e., Thesis Proposal jury proposal and Thesis Proposal defense minutes; (vii) *Thesis*, supporting the thesis defense management, i.e., jury proposal; (viii) *Statistics*, that supports the statistics regarding CSE PhD students; and (iv) *Doctoral Program Management*, that supports the management of the Doctoral Program Faculty members, CC3C, CCP and Scientific Committee members, staff members, and coordinator. DD-IMS has a web application, accessible to IST users, which provides a user interface so that users can visualize and manage the data regarding the CSE PhD students' processes. Each of the system's actors has access to a specific set of views so that they can use the functionalities of the modules described above.

We conducted an experimental validation to evaluate the DD-IMS at three levels: functionality, usability, and performance. Regarding *functionality*, the behavior of the system modules has to comply with the requirements gathered from the different actors. The 609 unit tests, that were created for the functionality validation, run without errors. *Usability* validated the graphical user interface of the DD-IMS. Usability tests were performed with 29 future users of the system. In general, users showed a high level of satisfaction with the DD-IMS usability. We obtained an average System Usability Scale (SUS) score of 88.19 (on a scale from 0 to 100). For *performance* evaluation, loading tests were performed to analyze the system response when used by a large number of users simultaneously. The results obtained show that the DD-IMS had a sustained growth in terms of error rate and average response time, maintaining an acceptable performance with a large number of concurrent users. After observing the behavior of the DD-IMS in response to a significant number of concurrent users (considerably larger than the potential number of active users involved in the CSE Doctoral Degree which is around 200/300 users), we can conclude that the DD-IMS meets the performance and reliability needs of an Information Management System for the CSE Doctoral Degree students.

1.5 Outline

The document is organized as follows. Chapter 2 describes the main processes of the IST CSE Doctoral Degree. Chapter 3 presents the system requirements. Chapter 4 details the related work, focusing on similar systems that already exist and several frameworks for web application development. Chapter 5 presents the design and implementation details of the DD-IMS. Chapter 6 describes the experimental validation that we conducted and the results obtained. Finally, Chapter 7 presents a summary of the thesis and its results, and the future work to be developed in the DD-IMS.

Chapter 2

Background

This chapter describes the main processes involved in the IST CSE Doctoral Degree, as stated in the PGA procedures[1] and in the internal CSE PhD Program Regulation (in revision). Moreover, the forms mentioned in this chapter can be accessed either through the PGA Forms web page[2], or the CSE PhD Documents and Forms web page[3]. Although the focus is on the IST CSE Doctoral Degree, most of the processes are common to other IST Doctoral Degrees.

The diagram in Figure 2.1 shows the students' academic progress path through the CSE Doctoral Program and identifies the actors involved in each stage of the Doctoral Program. There are four main candidate/possible student stages of the CSE Doctoral Program:

1. Applying to the Doctoral Program
2. Completing 30 ECTS credits of courses
3. Setting up the Thesis Advisory Committee (CAT from the Portuguese *Comissão de Acompanhamento de Tese*) and defending a *Thesis Proposal* evaluated by that committee (corresponds to the PhD Thesis Proposal course, 30 ECTS)
4. Setting up the jury and passing a *thesis defense* (PhD Thesis, 180 ECTS)

2.1 Application

For a student who wants to take the CSE Doctoral Degree, the first thing to do is to apply. The *application* can be performed in three ways: (i) through an online form; (ii) in person at PGA; or (iii) via e-mail. For each applicant, all the documents submitted will be kept in the Fenix system. After an application is submitted, the Fenix system assigns a process number to the candidate and generates the application fee payment. When the payment is performed, the PGA administrative staff submits the application process to the Fenix system to be evaluated by the CSE CC3C members. Once the deadline for applying for

[1] https://aqai.tecnico.ulisboa.pt/files/sites/27/22-2-area-de-pos-graduacao-fev-2019-pdf-1-39mb.pdf
[2] https://posgraduacao.tecnico.ulisboa.pt/en/programas-doutorais/formularios/
[3] https://fenix.tecnico.ulisboa.pt/cursos/deic/curricular-plan-20172018

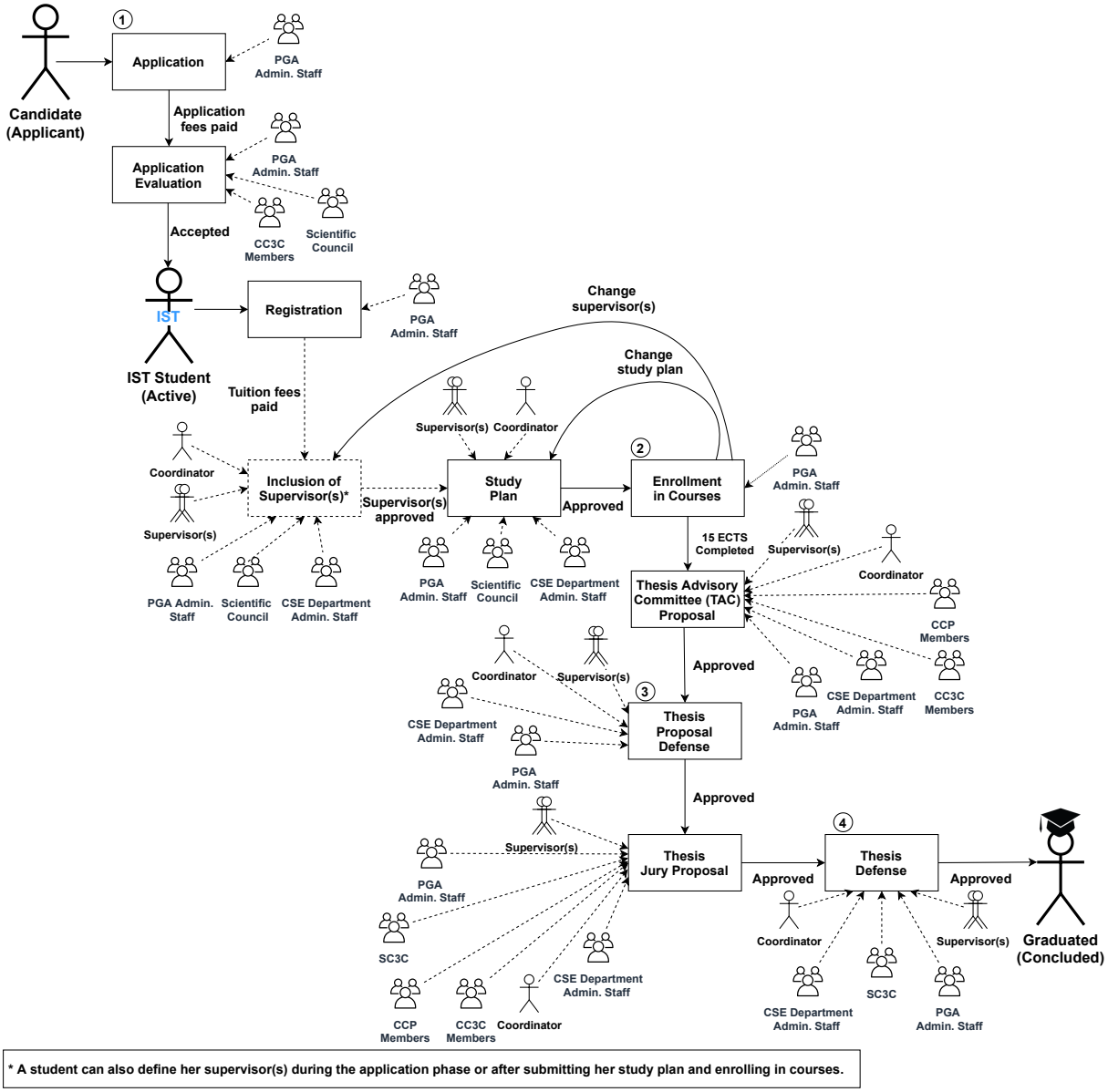


Figure 2.1: Student's academic path in the CSE Doctoral Program.

the program closes, the CSE department administrative staff extracts, from Fenix, the PDF documents submitted by each candidate and places them in a Google Drive directory. This directory is shared by the members of the CSE CC3C, which analyze all applications and approve or decline them. The result of this decision is sent by the CSE department in an official document (in Portuguese, it is called *Ofício ao Candidato*) to the PGA. The PGA checks if this document satisfies the University of Lisbon and IST legal regulations and validates it for further deliberation of the IST Scientific Council. If the document does not comply with the legislation, it will be returned to the CSE department to be rectified. After being approved by the IST Scientific Council, the PGA inserts the application decision into Fenix and sends an email to the candidate informing her about that decision. If the application is accepted, the candidate becomes an IST student.

2.2 Registration

Once a candidate is accepted, she can proceed to the *registration* phase. In this phase, the registration is formalized in the Fenix system. This formalization comprises the generation of an IST student number (if the person is new at IST) and the payment of tuition fees. The student must contact the IST IT Services Office[4] (DSI from the Portuguese *Direção de Serviços Informáticos)* to have access to the Fenix system.

2.3 Supervision

A Doctoral student must have scientific supervision by at least one professor, post-doctoral researcher or expert in the area of the thesis. When the supervisor is not an IST faculty member, the student is obliged to have at least one co-supervisor who is an IST faculty member or a post-doctoral researcher. Regarding the CSE Doctoral Degree, it is obligatory to have a supervisor from the CSE department. The maximum number of supervisors that a student can have is three. If the student's supervision team is already defined during the application phase, a letter of acceptance from the supervisor(s) must be submitted along with the other application documents requested. If not, the student must fulfill an inclusion of supervisor/co-supervisor form, which can be obtained from the PGA forms web page, after being accepted to the program. After being signed by the supervisor(s), this document is delivered to the CSE department and after being approved and signed by the coordinator, the CSE department sends the *inclusion of supervisor* form to the PGA via e-mail. Then, the PGA validates and sends it to the IST Scientific Council for further deliberation. After having been approved by the Scientific Council, it is sent to the PGA Academic Registry section. This area is responsible for inserting the student's supervisors in the Fenix system and notifying the student that her scientific supervision is approved.

During her Doctoral Program, a student is allowed to change her supervision team. A requirement (available at PGA web page) must be fulfilled and delivered in the CSE department, before being sent to the PGA. Like the study plan and supervisor inclusion forms, this document must be signed by the student, supervisor(s), and coordinator. The document is first validated by the PGA and then approved by the IST Scientific Council. After being approved, the PGA Academic Registry section updates the information in Fenix and notifies the actors involved in the student's Doctoral process i.e., student, coordinator, supervisor(s), and CSE department administrative staff.

2.4 Study Plan

Regarding a student's *study plan*, a form available at the CSE Doctoral Program web page (Documents and Forms) must be fulfilled and delivered to the CSE department, properly signed by the student and supervisor(s). In this document, the student states which courses she intends to enroll. CSE Doctoral Degree students must take a course called *Research Topics* (9 ECTS). In addition to this mandatory

[4] https://si.tecnico.ulisboa.pt/en/

course, students must choose 21 ECTS from a set of courses, which includes *Interdisciplinary Research*, *Outreach and Teaching Skills*, and other Optional Courses that belong to the CSE Doctoral Program curriculum[5]. Moreover, students can choose courses from other IST Doctoral Programs (up to 12 ECTS) or other faculties from the University of Lisbon (up to 6 ECTS). If a student has completed doctoral coursework (or equivalent) in other programs, she can request to the CSE Doctoral Program coordinator to have credit equivalences. It is also possible for a student to obtain credits (maximum of 4,5 ECTS for the CSE Doctoral Program) from prior education or professional training. Additionally, students who have a background other than Computer Science or Computer Engineering may have to take *propaedeutic courses*[6] (this is determined by the CC3C during the application evaluation). The list of propaedeutic courses must also be indicated in the study plan. After being approved and signed by the coordinator, the CSE department sends the original version of the study plan to the PGA, in paper format. Then, the PGA validates and sends it to the IST Scientific Council for further deliberation, and to the PGA Academic Registry section after having the IST Scientific Council approval. The study plan is then inserted into the Fenix system and the students are notified of the study plan approval. After having the study plan approved, the student must enroll in the courses. If the student's *study plan* is already approved within the courses enrollment period, she can enroll in courses online, via Fenix. If this does not happen, students can enroll in courses by sending an e-mail to the PGA asking for that. Before the beginning of each semester, all Doctoral students are informed by the PGA via e-mail about the courses enrollment period.

During her Doctoral Program, a student is allowed to change her study plan. A requirement (available at PGA web page) must be fulfilled and delivered in the CSE department, before being sent to the PGA. Like the study plan and supervision forms, this document must be signed by the student, supervisor(s), and coordinator. The document is first validated by the PGA and then approved by the IST Scientific Council. After being approved, the PGA Academic Registry section updates the information in Fenix and notifies the actors involved in the student's Doctoral process i.e., student, coordinator, supervisor(s), and CSE department administrative staff.

2.5 Thesis Proposal

Doctoral students in IST must present their *Thesis Proposal* within 24 months of the Doctoral Program start. When this 2-year deadline is close (about three months before the deadline), the Fenix system notifies the student, supervisors, coordinator, and the PGA administrative staff about the need of submitting the PhD Thesis Proposal and constituting its jury. However, the student can request an extension of the thesis proposal deadline. This requirement must have a valid justification and it must be approved by the supervisor and the Doctoral Degree coordination. Moreover, students only can submit their Thesis Proposal when they have at least 15 ECTS.

It is expected that the Thesis Proposal jury is sent by the supervisor to the coordinator or the CSE

[5] https://fenix.tecnico.ulisboa.pt/cursos/deic/curriculo
[6] https://en.wikipedia.org/wiki/Propaedeutics

department administrative staff. Again, the fulfillment of a form, available at the CSE Doctoral Program web page, is necessary for this process. It contains the following information:

- Full name, category, and affiliation of all jury members
- Identification of the supervisor, co-supervisors (if any) and the jury president
- Justification for choosing those jury members
- In addition, the following information must be supplied: (i) Thesis proposal summary; and (ii) CV of non-IST jury members

The CSE department administrative staff stores the jury proposal in a Google Drive directory. The CC3C members are then warned by e-mail to analyze and vote about this jury constitution in eight working days. Then, the proposal is put in a Google Drive directory shared by the CSE CCP. If there are no negative evaluations in 5 working days, the proposal is approved and the jury proposal form is sent to the PGA (via Fenix or e-mail). PGA will then verify if the jury constitution complies with IST Doctoral Programs general regulation. The PGA Logistics and Archive section will submit the jury proposal in the Fenix system and send a copy of the approved document to the student. Regarding the public presentation of the Thesis Proposal defense, the supervisor(s) is/are responsible for its scheduling and then for informing the CSE department administrative staff about the date. The CSE department administrative staff will be responsible for the room reservation and public announcement on the web. After the thesis proposal defense, the thesis proposal minutes, whose template is also available at the CSE Doctoral Program web page (Documents and Forms), are produced and signed by all jury members. The supervisor sends the minutes to the CSE department so that it can also be signed by the coordinator and then submitted to the PGA. After being approved by the PGA, the minutes are sent to the PGA Logistics and Archive section, which is responsible for submitting the document in Fenix and sending a copy of the document to the student via e-mail.

2.6 Thesis

A PhD student must deliver her PhD *thesis* in the PGA on a minimum of three years and a maximum of five years since the start of her Doctoral Program. However, the student can request an early delivery of the thesis (less than 3 years) or an extension of the thesis (more than 5 years). This requirement must have a valid justification and it must be approved by the supervisor and the Doctoral Degree coordination. Regarding the thesis submission, the following preconditions must be satisfied: (i) students must have concluded with success the courses defined in their study plan (30 ECTS); (ii) tuition fees payment must be regularized; and (iii) the thesis proposal defense minutes must be approved (30 ECTS from the Thesis Proposal course).

To submit her PhD thesis (*requerer provas* in Portuguese), the student needs to deliver several documents to the PGA: (i) thesis submission form; (ii) student's CV; (iii) copies of her *thesis dissertation*; (iv) a statement of all supervisors, in which they declare that they are aware that the thesis will be

delivered to the PGA; (v) a statement from the student declaring that she is presenting a new and original work; and (vi) a Reproduction Release Rights for Copyrighted and Proprietary Rights Material. After the thesis submission is concluded, all these documents are inserted into the Fenix system. Then, the process is sent to the PGA Academic Examinations section, and the Fenix system sends an alert to the Doctoral Program coordinator and CC3C members, asking to send the constitution of the *thesis defense* jury.

The Doctoral Program coordination must send the *thesis defense jury proposal* to the PGA within 30 days after the thesis submission. In this jury proposal form (available at the CSE Doctoral Program web page), two *rapporteurs* must be nominated to read and give an opinion about the thesis. The jury president should be the IST Scientific Council president but typically he delegates this task to a member of the CSE Scientific Committee (must be a full professor). The pool of full professors from this committee is maintained by the CSE department administrative staff. The thesis defense jury proposal is performed by the supervisor(s) and needs to be approved by the Doctoral Program coordinator, CC3C, and CCP, following an evaluation process similar to the Thesis Proposal evaluation. After the jury is appointed and submitted in Fenix, it must be validated by the PGA and approved by the IST Scientific Council. Then, the student and jury members are notified about the jury constitution. The rapporteurs' opinion shall be transmitted to the PGA, coordinator, and jury president via e-mail, in 40 days. The jury members should meet and decide, in 20 days after receiving the rapporteurs' opinion, if the thesis is ready to be defended. If it is an in-person meeting, the PGA is informed of the meeting's date and local, and a PGA member will be present in this meeting. If the jury members meet electronically, the jury president must forward the results of the meeting to the PGA. All members of the jury are also notified about this meeting via e-mail. This e-mail has the rapporteurs' report attached. During the first meeting, the jury must take into account the opinion of all its members and the rapporteurs. The students receive the result of the meeting via e-mail. There are three possible outcomes:

- The submitted version of the thesis is accepted and the jury schedules the public defense.
- The thesis is accepted but a new version that includes the jury corrections and recommendations must be submitted in the following 20 days. The public defense is also scheduled by the jury.
- The submitted version of the thesis is rejected. The student receives the jury recommendations and can submit a new version of the thesis in 120 days after the jury deliberation. If the student submits a new version of the thesis, there will be a second meeting with all the jury members.

After the thesis defense, the Doctoral Degree is given to the student that has been approved in the thesis defense. The PGA elaborates the minutes related to the *jury meetings* that have taken place before the defense and related to the thesis defense.

A Doctoral Program student passes through various status (*Applicant*, *Active*, *Concluded*). At any point, it is also possible that a student suspends (*Suspended*) or even withdraws (*Abandoned*) her Doctoral Program. When a student wants to interrupt (temporarily or definitely) her Doctoral Program, a request must be sent to the PGA. The PGA coordinator validates the student's request, analyzes

the process, and submits for approval of the IST Scientific Council and the Governing Board. When approved, it is forwarded to the PGA Academic Registry section, which will update the student status in Fenix and will also inform the student and her supervisor(s). In case of suspension, there is an interruption of the time counting for the thesis delivery deadline but the tuition fees payment cannot be interrupted. Finally, the student's process can also be canceled by the school if there are irregularities in the process. After being canceled, a process cannot be reactivated.

Chapter 3

Requirements

In order to understand the stakeholders needs, which is important when defining the solution for implementing the DD-IMS, we gathered the requirements that it should meet through a set of interviews. We decided to interview at least one person from each type of stakeholder so that we could understand the limitations of the current solution (Fenix system) and the requirements of the new system. Regarding the Doctoral Degree Coordinator, we met with the current coordinator and three former coordinators. Regarding PhD supervisors, we met with six supervisors from the different scientific areas[1] of the CSE department: one from Architecture and Operating Systems, two from Computer Graphics and Multimedia, one from Artificial Intelligence, one from Programming Methodology and Technology, and one from Information Systems. Regarding students, we interviewed five students in different stages of their Doctoral Program, including students at the beginning of the Doctoral Program, students that already have concluded the coursework part but did not submit the thesis proposal yet, students with the thesis proposal submitted but not presented yet, and students with the thesis proposal defended. Regarding the CSE department administrative staff, we met with the CSE department administrative staff coordinator and the person responsible for the Doctoral Programs in the CSE department administrative staff. Regarding the CSE department CC3C and CCP members, we met with two members of the CC3C, which also belong to the CCP. We also met with the PGA coordinator. The requirements are documented in [11].

In this chapter, we summarize the Fenix limitations identified by the stakeholders, as well as their requirements for the DD-IMS.

3.1 Fenix Limitations

After meeting with PhD students, PhD coordinators, PhD supervisor, members of the CSE Department and PGA administrative staff, and CC3C members, we identified the following limitations of the Fenix system for managing data regarding CSE PhD students:

1. It provides a **limited set of statistics** about the Doctoral Program students to the coordinators

[1] https://fenix.tecnico.ulisboa.pt/departamentos/dei/areas-cientificas

and administrative staff. These statistics are particularly important when producing a report about the CSE Doctoral Degree to the Agency for Assessment and Accreditation of Higher Education (A3ES, from the Portuguese *Agência de Avaliação e Acreditação do Ensino Superior*)

2. The existence of **multiple views** available to the coordinator and administrative staff makes it very difficult to manage data and to obtain statistics about the degree (e.g., how many students are enrolled, how many have defended their Thesis Proposal in a given period, etc.), particularly for coordinators that are responsible for producing reports with that kind of analysis. Moreover, for the PGA administrative staff, different results can be retrieved from the Academic Services and Academic Administration views (e.g., the number of active PhD students differs in these two views)

3. A supervisor cannot keep track of their **students' curricular plan** (she does not have access to the courses in which students have been enrolled nor to the grades obtained in those courses)

4. A supervisor can only visualize their **students' study plan through a PDF file**. Therefore, there is no direct link for getting information about each of the courses indicated in the study plan. In what regards students, they cannot visualize their study plan (they do not even have access to the study plan PDF file as they can only visualize the documents that were submitted during their application phase)

5. Both Supervisors and students cannot directly obtain how much time is left for the **thesis proposal submission deadline**

6. Fenix does not provide **statistics regarding the students' status** (concluded, active, suspended, etc.)

7. Fenix does not provide functionalities for **visualizing and evaluating the candidates' applications** (in the CSE department, the candidates' application documents are downloaded and placed in a Google Drive directory, which is shared among the CC3C members)

8. For the CSE administrative staff, Fenix provides **incomplete information about the personal data of students** that did not take any Bachelor and Master degree at IST (e.g., the phone number is not provided)

9. For CC3C members, the Fenix system **does not distinguish the students** that the faculty member is supervising from the students whose information the faculty member has access to as a member of the CC3C

10. Several **different technologies are being used in ad-hoc fashion** to support the students Doctoral Program process (i.e., email, Google Drive)

Although it is not a limitation of the Fenix system, students cannot submit their thesis proposal in Fenix (this could be done through the Thesis Proposal course web page but it was never decided to use the course web page for the students' thesis proposal submission as this web page is only intended for providing guidelines and documentation to the PhD students).

To address the limitations mentioned above, the CSE department administrative staff uses four spreadsheets to store data regarding the PhD students. These spreadsheets contain information about: (i) the Doctoral Degree jury presidents in the past years, (ii) the students' enrolled courses in each semester (based on the students' study plan), (iii) the status of the PhD students' jury proposal for the thesis proposal defense, and (iv) the status of the PhD students' jury proposal for the thesis defense.

3.2 Requirements

During the meetings with PhD students, PhD coordinators, PhD supervisor, members of the CSE Department and PGA administrative staff, and CC3C members, we also identified the following requirements for the DD-IMS (there is no relation between the requirements identifier, R-x.y, and the limitations stated above):

- **R-1**: Regarding the current and past PhD coordinators, the functionalities that they considered useful to have are as follows:
  - **R-1.1**: For a given student:
    - **R-1.1.1**: Get the list of courses in which she is enrolled
    - **R-1.1.2**: Get the list of courses concluded with success
    - **R-1.1.3**: Get her supervisors
    - **R-1.1.4**: Check if she has already concluded the thesis proposal
    - **R-1.1.5**: Get the thesis submission deadline
    - **R-1.1.6**: Get the courses contained in the student's study plan
    - **R-1.1.7**: Get her grades obtained in courses
    - **R-1.1.8**: Get her final grade after completing the thesis
    - **R-1.1.9**: Get her scientific production (list of papers)
    - **R-1.1.10**: Get the thesis title, the year of conclusion of the thesis, and its classification (rejected, approved or approved with distinction)
  - **R-1.2**: For a given supervisor:
    - **R-1.2.1**: Get her research unit (where she develops her research work)
    - **R-1.2.2**: Get her students
  - **R-1.3**: Visualize the following generic statistics:
    - **R-1.3.1**: Get the average time that students take to finish their PhD program
    - **R-1.3.2**: Get the number of suspended students and abandoned students
    - **R-1.3.3**: Get the number of registered foreign students
  - **R-1.4**: Visualize the following statistics:
    - **R-1.4.1**: Get the total number of students enrolled per year or curricular year

        - **R-1.4.2**: Get the number of students registered in a given time interval, and a percentage of how many of those students have already concluded their Doctoral Program
        - **R-1.4.3**: Get the number of candidates and accepted candidates per year
        - **R-1.4.4**: Get the number of graduated students per year and the number of years taken to finish the degree
        - **R-1.4.5**: Get the number of students that did not have defended their Thesis Proposal after 24 months since the start date of their PhD
        - **R-1.4.6**: Get the number of students that did not have defined their supervision after 6 months since the start date of their PhD
        - **R-1.4.7**: Get the number of students that did not have defended their Thesis after 48 months since the start date of their PhD and did not require any extension for the Thesis Defense deadline
    - **R-1.5**: The system should generate alerts (shown in the application when users log in and also sent via e-mail) in the following conditions:
        - **R-1.5.1**: Students that did not have defended their Thesis Proposal after 24 months since the start date of their PhD. The alert is sent to students as well as to their supervisors and the coordinator.
        - **R-1.5.2**: Students that did not have defined their supervision after 6 months since the start date of their PhD. The alert is sent to students and the coordinator.
        - **R-1.5.3**: Students that did not have defended their Thesis after 48 months since the start date of their PhD and did not require any extension for the Thesis Defense deadline. The alert is sent to students as well as to their supervisors and the coordinator.
- **R-2**: The functionalities that the PhD supervisors considered useful to have are the following ones:
    - **R-2.1**: Get a student's study plan
    - **R-2.2**: Follow her students' progress (which courses has she concluded and which grade did she obtained). A graphical timeline that shows the student's progress could be interesting
    - **R-2.3**: Check how much time is left for the thesis proposal submission deadline
    - **R-2.4**: Get information about the student's study plan changes and supervisor changes
    - **R-2.5**: Get the students' grades in previous degrees (if obtained at IST)
    - **R-2.6**: Get the CSE Doctoral Degree students that do not have supervisors yet as well as their scientific interests
    - **R-2.7**: Have a direct link to the web page of the courses in which her students are enrolled
    - **R-2.8**: Be able to fulfill PhD forms (e.g., Thesis Proposal jury, Thesis jury) online
    - **R-2.9**: Have a feature for helping to schedule dates for Thesis Proposal and Thesis
- **R-3**: Regarding PhD students, the following functionalities are considered useful to have:

  - **R-3.1**: Get the study plan
  - **R-3.2**: Check how much time is left for the thesis proposal deadline
  - **R-3.3**: Be notified about seminars related to the student's research area, where the student can subscribe to topics in which she has interest
  - **R-3.4**: Have access to a forum for PhD students where they can post or get information about the Doctoral Degree
  - **R-3.5**: Present, in a timeline, the students' progress in their Doctoral Degree
  - **R-3.6**: Have indications on the thesis writing procedures (this could also be done by providing to the students a link for the CSE Doctoral Degree web page with templates and rules for writing the thesis document)
  - **R-3.7**: Be able to fulfill PhD forms (e.g., study plan, supervisor inclusion, etc.) online

- **R-4**: Regarding the CSE department administrative staff, the functionalities considered useful to be implemented by the DD-IMS are as follows:
  - **R-4.1**: Get the most attended courses in the Doctoral Program (courses with more enrolled students)
  - **R-4.2**: Get the students' status which can be one of the following: concluded, active, suspended, abandoned, registered but not enrolled in any curricular unit in the current semester, with academic part concluded, and without grade in a course
  - **R-4.3**: Get the number of students for each of the possible students' status
  - **R-4.4**: Submit and get the minutes regarding the thesis proposal and the thesis defense
  - **R-4.5**: Get the CC3C members' opinion about the candidate applications
  - **R-4.6**: Have a complete view of the students' personal and academic data and be able to validate students' academic data such as research unit and doctoral program
  - **R-4.7**: Be able to automatically generate PhD documents, in PDF format (e.g., study plan, Thesis Proposal jury, etc.) and ask the system to send the document via e-mail to the PGA

- **R-5**: Regarding the CC3C members, the functionalities considered useful are the following ones:
  - **R-5.1**: Separate the students that the faculty member is supervising from the other PhD students
  - **R-5.2**: Get relevant information and documents regarding the candidates' application (i.e., CV, certificates of the degrees awarded, list of courses taken and respective grades, statement of purpose, letters of recommendation)
  - **R-5.3**: Evaluate the applications (approve or decline the applications) through an automatic system

The PGA administrative staff is not an actor in the system as they do not interact directly with the system. In fact, it was considered that a direct communication with the new system does not facilitate the PGA administrative staff work as the PGA administrative staff would still need to interact with the system of each department. Unlike what happens with the CSE department members, the DD-IMS cannot be a single point of access for the PGA administrative in which regards the PhD processes. For this reason, no functionalities were implemented for the PGA administrative staff role in the system to be developed.

Another requirement of the system is that the application to be developed is a web-based application, accessible via browser. This allows users to access the system from any operating system and without needing to install the application.

Chapter 4

Related Work

This chapter presents the most relevant works concerning: (i) Higher Education Information Management Systems whose purpose is similar to the system we propose to develop in this thesis (Section 4.1); and (ii) web application development frameworks which are software packages that are used to develop web applications (Section 4.2).

4.1 Higher Education Information Management Systems

In this section, we describe software systems with a similar purpose to the Information System that we developed. These systems can be divided into two types of systems: (i) Enterprise Resource Planning (ERP) systems that can be adapted and extended to create adequate software for an organization-specific goal; and (ii) software systems built from scratch to satisfy the needs of a specific organization. While some universities invest in ERP systems to integrate all functional areas of the organization, other universities use their IT human resources to build in-house developed systems that integrate all those areas. In Higher Education institutions, the services typically supported by information systems are the following ones [5]: Student Lifecycle Management (SLM), Learning Management Systems (LMS), Human Resources (HR), Finance, Library services, Student Information, Content Management System (CMS), and the University Portal. In which concerns our project, the services we want to provide are PhD SLM and Student Information. At IST, most of the other services are supported by the Fenix system or other software products (e.g., SAP[1] for Finance).

Sections 4.1.1 to 4.1.4 describe software systems built from scratch: FenixEdu[2], Manipal University Jaipur - Online Faculty Information System (MUJ-OFIS), University Study-Oriented System (USOS), and HISinOne. Sections 4.1.5 to 4.1.7 describe ERP systems that have a general-purpose and are configurable for Higher Education: Fedena, Salesforce for Education, and Banner by Ellucian. We choose to refer to the Fedena system because it has an open source version. The remaining systems were chosen because they are considered by G2[3], a business software review site, as two of the highest-rated ERP

[1] https://www.sap.com/index.html
[2] http://fenixedu.org/
[3] https://www.g2.com/categories/higher-education-student-information-systems

systems for Higher Education Institutions, in terms of market presence and clients' satisfaction. Then, in Section 4.1.8, we make a brief analysis of the systems described and discuss whether they can be used in the development of our system.

4.1.1 FenixEdu

FenixEdu project started in 2002 at IST. It consists of an open source system that provides a wide set of academic and management functionalities to the IST academic community. Some of the modules provided by FenixEdu are Human Resources Information, Students Admission, Fee Management, Students Enrollment Management, and Examination Management. Since the system is open source and the code is organized in modules, it is possible to extend its functionality and create new modules. The license used is *GNU Lesser General Public License (LGPL) v3*, which enables commercial, private and patent use, distribution and modification of the system (changes need to be documented). The system documentation is publicly available[4] but no support is provided if any problem occurs. The FenixEdu framework was used in the development of the Fenix system, already mentioned in this document.

The FenixEdu software is written in Java[5]. The FenixEdu applications run on top of the MySQL[6] or MariaDB[7] database management systems (DBMS), using the InnoDB engine.

FenixEdu makes available an open API [2], which can be used for retrieving data from IST. The endpoints provided by this API give access to data related to students, degrees, courses, and other services related to IST (e.g., the menu of the campus canteen). There are two types of endpoints: public (can be accessed by everyone) and private (require user authentication). This API allows users to grant access to their private data (private endpoints) through the OAuth2[8] authorization protocol. The big advantage of this API is that it allows the academic community (i.e., students, teachers, third-party software vendors) to integrate IST academic data with their software, without being restricted to a particular technology.

4.1.2 Manipal University Jaipur - Online Faculty Information System

Manipal University Jaipur - Online Faculty Information System (MUJ-OFIS) [3] is an application developed to manage data about faculty members in India. Its purpose is to record faculty accomplishments, including research, publications, teaching, awards, and service activities for each calendar year. The data entered is used to generate a variety of reports such as the Annual Report, Departmental Report, and individual CVs following templates. The functionalities of this system allow faculty members to keep their profile updated, easily manage and download their reports, and keep work history related to former students, for example. Regarding administrative staff, it is possible to visualize faculty and departmental activity reports and generate documents. Nothing is said about the management of Doctoral students in particular. So, it is unknown whether this system manages Doctoral Degrees information.

[4] https://fenix-framework.github.io/index.html
[5] https://www.oracle.com/java/
[6] https://www.mysql.com/
[7] https://mariadb.org/
[8] https://oauth.net/2/

The MUJ-OFIS client application is compatible with Google Chrome, Opera, and Safari but has certain compatibility issues with Mozilla and Internet Explorer. It can be accessed by any type of device (desktop or mobile).

Regarding web development frameworks, Laravel[9] was used in the development of MUJ-OFIS. The technologies used for the front-end were HTML[10], CSS[11], Bootstrap Framework[12], JavaScript[13], PHP[14], AJAX[15], and JQuery[16]. In terms of back-end, the system is compatible with the MySQL database. Apache[17] and WAMP[18] were used for server and software stack, respectively.

4.1.3 University Study-Oriented System

University Study-Oriented System (USOS)[10] is a student management information system, owned by the University Centre for Informatization (MUCI) consortium[19], for Higher Education institutions in Poland. Like Fenix, this system is a case of an in-house developed software system. It comprises an Oracle central database (named USOS database), a desktop application for the university administrative staff, developed in Oracle technology, and several web-based applications for handling the student and teacher processes (e.g., enrollment in courses, grades).

The universities that use USOS (around 40) do not differ in what concerns their general procedures since they follow the regulation. However, some procedures may vary from university to university which requires different solutions for each institution. So, being able to quickly develop new software solutions or adjustments is an important requirement of the system.

USOS provides an API[20] which contains a set of endpoints that give access to the university data. Despite being publicly available, a special administration key that gives unrestricted access to the data is needed (administrators can run any API endpoint with any user). Without that administration key, the applications need to ask users to log in and grant access to their data. It is also possible to access the API anonymously but with a limited subset of methods available.

The web applications used to be developed using technologies like JavaScript, PHP, Smartly, and Ajax. Around 2013, the USOS designers proposed a new architecture of the system. Moreover, they decided to implement a new framework, Django USOS, to share some functionalities among the various applications and support rapid application development. This framework can be used as a basis for new applications that need to access the USOS central Oracle database. It is written in Python[21] and based on the Django framework[22]. Besides having direct access to the USOS database, Django USOS also

[9] https://laravel.com
[10] https://developer.mozilla.org/en-US/docs/Web/HTML
[11] https://developer.mozilla.org/en-US/docs/Web/CSS
[12] https://getbootstrap.com/
[13] https://www.javascript.com/
[14] https://www.php.net/
[15] https://developer.mozilla.org/en-US/docs/Web/Guide/AJAX
[16] https://jquery.com/
[17] https://httpd.apache.org/
[18] http://www.wampserver.com/en/
[19] http://muci.edu.pl/
[20] http://apps.usos.edu.pl/developers/api/
[21] https://www.python.org/
[22] https://djangoproject.com

provides a set of services to the client applications such as mailing service, search mechanism, authentication based on Central Authentication Service (CAS)[23], data models for converting Oracle tables into Django objects, libraries that interact with USOS API, etc.

Django USOS was tested as the software development framework of an application responsible for the archive of bachelor/master thesis. Nothing is said about the management of Doctoral students in particular. So, it is unknown whether this system can be the basis of a Doctoral Students Information Management System.

4.1.4 HISinOne

HISinOne [4][7] is a project developed by Hochschul-Informations-System (HIS)[24] in Germany, whose main goal is to create an integrated management system for universities. This includes all processes of the student life cycle and the management of human/financial resources (ERP). Some of the components that were already implemented were related to the student lifecycle management (from admission to alumni), going through the creation of the student's study plan and enrollment in courses and exams.

Nowadays, this system is deployed in several German Higher Education institutions. One of these is the University of Freiburg[25]. The migration to this new system is not completed yet. While some study programs already migrated to this platform, others are still using the old system[26]. In which regards Doctoral Programs, the system mentions that the 'graduate' role is still in development, so it is possible that it may not fully support Doctoral Programs and its processes.

The technology basis used for the development of this system was the Java Enterprise Edition (J2EE) platform[27]. The Hibernate[28] framework was used for mapping an object-oriented domain model to a relational database model. JavaServer Faces[29] (JSF) was used for building user interfaces for the web application. HISinOne is compatible with the PostgreSQL 9.1 DBMS[30].

4.1.5 Fedena

Fedena is an ERP system for educational institutions developed by Foradian Technologies[31]. Analyzed in [8], this system offers a huge range of functionalities for Higher Education institutions, in particular, student management, which is the most desired functionality in our context. Some of the features provided are student information management (i.e., access to current and archived student records, using various filters) and institution data management (i.e., creation of customized forms that allow the insertion of data in the system). With flexible pricing, Fedena provides different types of solutions, allowing customers to choose between more affordable solutions and more complete solutions, with more mod-

[23] https://www.apereo.org/projects/cas
[24] https://www.his.de/
[25] https://campus.uni-freiburg.de/
[26] https://www.studium.uni-freiburg.de/en/logins
[27] https://www.oracle.com/java/technologies/java-ee-glance.html
[28] https://hibernate.org/
[29] https://javaee.github.io/javaserverfaces-spec/
[30] https://www.postgresql.org/
[31] https://www.foradian.com/

ules and better support. There is also an open source version of this system, Project Fedena[32]. When comparing this version to the paid versions, the open source version not only lacks support from the vendor but also many features, including inventory, custom reports, and registration. Moreover, the fact that the basic version of the system is open source code and free of charge makes it very customizable. This means that developers can build their student information system customized modules or plugins to modify and extend the basic functionality without changing the basic source code. This feature makes it possible to adapt Fedena to the specific processes of each institution.

The technology used in the Fedena development was Ruby on Rails[33], a web framework written in Ruby[34]. The system is compatible with MySQL. Fedena is a cloud-hosted platform. Its client application is web-based, so users can access it via browser. It also has a mobile application for both iOS and Android operating systems.

4.1.6 Salesforce for Education

Salesforce for Education is a product designed by Salesforce[35] that provides a cloud platform supporting student management, admissions, and other Higher Education administrative support functions. It provides the system documentation and online support in case of problems.

Regarding the technologies used, Apex[36] is the language that Salesforce created for developing applications in their platform. Apex follows the basic modern standards for an object-oriented programming language and is visually and programmatically very similar to Java. Visualforce[37] is the language used to create user interfaces. This language uses <> </> component pairs to define layout components and define attributes. As stated before, Salesforce for Education is a cloud-based platform. Its client application is web-based, so users can access the application via browser.

4.1.7 Banner by Ellucian

Banner is an ERP system developed by Ellucian[38]. Banner is a cloud-based software provider for Higher Education institutions. This platform works as a campus-wide option to manage all aspects of those institutions, including functionalities as student management, human resources, financials, and more. Regarding student management, which is the most desired functionality in our context, some of the features provided by Banner are registration, advising, administration tools, and key reporting. It provides the system documentation and online support in case of problems.

Regarding the technologies used, Banner is written in Java and it is compatible with the Oracle DBMS. Banner by Ellucian is a cloud-hosted platform. Its client application is web-based and users can access the application via browser.

[32] https://projectfedena.org/
[33] https://rubyonrails.org/
[34] http://www.ruby-lang.org/
[35] https://www.salesforce.org/
[36] https://developer.salesforce.com/docs/atlas.en-us.apexcode.meta/apexcode
[37] https://developer.salesforce.com/docs/atlas.en-us.pages.meta/pages
[38] https://www.ellucian.com/

4.1.8 Discussion

A summary of the systems analyzed in this section is presented in Tables 4.1 and 4.2. To compare the analyzed systems, we collect the following characteristics for each one of those systems: (i) *Open Source*: indicates if the system is open source; (ii) *Development Team*: the entity responsible for system development; (iii) *Compatible Databases*: the databases that are compatible with the system; (iv) *API*: indicates if the system provides an Application Programming Interface (API) (only present in Table 4.1 because the ERP systems do not provide an API); (v) *Extensible*: indicates if the system functionality can be extended; (vi) *Browser Client Application*: indicates if the client can be accessed via a browser; (vii) *Programming Language*: the programming language used in system development; and (viii) *Supports PhD Degree*: indicates if the system supports the management of PhD Degrees (only applicable to systems built from scratch).

Table 4.1: Comparison of Information Management Systems that were built from scratch for Higher Education institutions (described in Sections 4.1.1 to 4.1.4).

Systems/ Characteristics	**FenixEdu**	**MUJ-OFIS**	**USOS**	**HISinOne**
Open Source?	Yes (LGPL 3.0)	No	No	No
Development Team	IST	Manipal University	MUCI Consortium	HIS
Compatible Databases	MySQL Maria DB	MySQL	Oracle Database	Postgres
API?	Yes	No	Yes	No
Extensible?	Yes	No	Yes	Yes
Browser Client Application?	Yes	Yes[1]	Yes	Yes
Programming Language	Java	PHP (Laravel)	Python (Django)	Java (J2EE)
Supports PhD Degree?	Limited	Unknown	Unknown	In Development

[1] Compatible with Google Chrome, Opera, and Safari but has certain compatibility issues with Mozilla and Internet Explorer.

After analyzing these systems, we concluded that there were two possible solutions to implement the DD-IMS: to use an already existent ERP system available in the market or to build a new system from scratch. Another possibility was to improve the Fenix, as its software is extensible and open source, but was not considered due to IT resources limitations in the FenixEdu team.

Using an already existing ERP system has the advantage of being possible to use the available functionalities of that system. On the other hand, this may not be easy if we need to change or extend the functionality of the system. Adapting the system architecture to the requirements and modules specified for the project may be difficult if the architecture is not the most appropriate one for the implementation of one or more functionalities. Moreover, most of these ERP systems offer a wide range of functionalities that cover all areas of Higher Education institutions. Normally, these systems manage not only the students' lifecycle (which is part of the academic area implemented in the Fenix system) but also financial

Table 4.2: Comparison of ERPs for Higher Education institutions (described in Sections 4.1.5 to 4.1.7).

Systems/ Characteristics	Fedena	Salesforce for Education	Banner by Ellucian
Open Source?	Yes[1] (LGPL 2.1)	No	No
Development Team	Foradian Technologies	Salesforce	Ellucian
Compatible Databases	MySQL	—[2]	Oracle Database
Extensible?	Yes	Yes	Yes
Browser Client Application?	Yes	Yes	Yes
Programming Language	Ruby (Ruby on Rails)	Apex	Java

[1] Only the base version is open source.
[2] Unknown

and human resources areas (not supported by Fenix but supported by other software systems at IST). The use of such a system would cause the duplication of those functionalities. Besides, using an ERP system implies its acquisition, which has a financial cost associated. With this in mind, only an open source system can be considered for use in our system implementation and Project Fedena is the only system that meets this requirement.

Concerning the solution of building a new system from scratch, the big advantage is that it is possible to create a solution that is tailored to the system requirements and whose architecture is adapted to all functionalities of the system. In contrast, it is not possible to use the functionalities already implemented in existing systems. However, the cost of implementing all the functionalities can be mitigated by using web application frameworks or libraries that facilitate the development.

After analyzing the advantages and disadvantages of both solutions, it was decided to build a new system from scratch. Despite not using any of the systems described, the Fenix API, a component referred to in this section, was used for the benefit of the new system. With this API, it is possible to communicate with the Fenix system, thus avoiding the storage of data that is already stored in the Fenix system.

4.2 Web Application Development Frameworks

According to the discussion presented in Section 4.1, the most adequate solution for the problem stated in Section 1.2 is to develop an Information Management System from scratch. Therefore, in this section, we analyze some of the most popular web application development frameworks according to the Hot-Frameworks[39] ranking. The frameworks analyzed were: Django, Flask[40], Express.js[41], Ruby on Rails, Spring Boot[42], and Laravel.

[39] https://hotframeworks.com/
[40] https://palletsprojects.com/p/flask/
[41] https://expressjs.com
[42] https://spring.io/

When selecting a web application development framework, the most important factors to take into account are ease-of-use, performance, and productivity.

The *ease-of-use* of a framework is measured by some aspects as to how easy it is to learn the underlying programming language, the quality of its documentation, and the size and activity of its community. The framework *performance* is usually not the most relevant factor while analyzing web frameworks because even frameworks with relatively slow run times are more than ”good enough” for mid-sized sites running on moderate hardware. For performance, we took account of (i) TechEmpower Framework Benchmark[43], a platform that compares the performance of web application frameworks executing tasks such as JSON serialization, database access, and server-side template composition; and (ii) a benchmark that assesses the most popular frameworks and technologies[44], while performing specific tasks with different levels of complexity such as database access or Fibonacci numbers computation. A framework *productivity* refers to how quickly can new features be created once developers are familiar with the framework, and includes the effort to write and maintain code. It also depends on documentation, community, compatible databases, and programming experience. Another factor related to the productivity of a framework is the number of tools and libraries provided by the framework. Some web frameworks include tools and libraries that address every problem their developers can think ”by default” (*batteries included* approach), while more lightweight frameworks expect web developers to pick and choose a solution from separate libraries (*get it yourself* approach).

4.2.1 Django

Django is a high-level framework written for Python, created in 2005. The primary goal of Django is to facilitate the creation of complex database-driven websites. It takes care of much of the problems of web development, so developers can focus on writing their applications without needing to reinvent the wheel. Its extensive, updated and detailed documentation and its large and active community make this framework easy to learn and to use.

Django follows the Model-View-Controller[9] (MVC) pattern, but with a curious peculiarity: in Django, the controller is called ”view” and the view is called ”template”.

Regarding productivity, Django follows a *batteries included* approach as it provides almost everything most developers might want due to a huge number of libraries and tools. Regarding databases, Django officially supports PostgreSQL, MySQL, Oracle, and SQLite[45] but there are also many database backends provided by third parties (e.g., Microsoft SQL Server). Moreover, it is based on the Python language that is known for being easy to read and to maintain.

In terms of performance, Django is not the best option due to Python’s low performance in comparison with other languages, like JavaScript.

[43] https://www.techempower.com/benchmarks/#section=data-r18
[44] https://medium.com/@mihaigeorge.c/web-rest-api-benchmark-on-a-real-life-application-ebb743a5d7a3
[45] https://www.sqlite.org/index.html

4.2.2 Flask

Flask is a lightweight framework for Python, created in 2010. Unlike Django, Flask can be considered a ”micro-framework” because of its minimalism (i.e., does not have tools nor libraries). Flask suggests some tools or libraries in its documentation but does not enforce any dependencies or project layout. This characteristic gives developers a lot of flexibility in how they develop their web apps. It is up to the developers to choose the tools and libraries they want to use (*i.e., get it yourself* approach). Moreover, Flask is also known for having good documentation and an active community, but not as extensive as Django.

In terms of performance, Flask is not the best option (just like Django) due to Python's low performance.

4.2.3 Express.js

Express.js is a framework for JavaScript, created in 2010, and it is one of the most famous frameworks for Node.js[46] environment. Express.js is extremely popular, partially because of its simplicity and ease-of-use. It is well documented and has a big community with a lot of tutorials and guides.

Regarding productivity, Express.js is a minimalist and flexible framework. It does not incorporate every component developers may need (*get it yourself* approach), but many third-party libraries and tools can be used. When using a minimalist framework like this (or Flask, for example), developers can face difficulties in finding out which is the best component for their purpose. Regarding databases, Express.js does not support databases natively. However, it is possible to connect a database to an Express.js application by installing Node.js modules that work as database drivers. So, these database drivers make it possible for developers to use databases like MySQL, Oracle, SQLite, PostgreSQL, MongoDB, and many more.

Express.js is one of the frameworks with the highest performance because it is resource-efficient as Node.js uses lightweight multitasking within a single thread rather than creating new processes when new web requests are received. As it runs on a single-threaded environment, Express.js is not the best option for heavy computation applications because heavy operations will block further incoming requests.

4.2.4 Ruby on Rails

Ruby on Rails (or Rails) is a web framework written for the Ruby programming language. Created in 2004, Rails adopts a design philosophy that is very similar to Django: it encourages the MVC pattern as well. Rails divides the code into three sub frameworks: Active Record, Action View, and Action Controller [1]. Rails has a massive community but lacks proper documentation.

Rails comes with many useful *gems* which are packages that extend the functionalities of the application (*batteries included*). These are helpful for developing applications faster and more efficiently.

[46] https://nodejs.org

A disadvantage of Rails is its runtime speed, which is very slow when compared with other frameworks (for instance, Express.js or Spring Boot).

4.2.5 Spring Boot

Spring Boot is a framework created in 2003 for server-side web development in Java. Spring Boot is built on top of Spring, which follows the inversion of control (IOC) pattern. IOC is a principle that enables a framework to take control of the flow of a program and make calls to our custom code, in contrast with traditional programming where our code takes control of the flow of the program. Frameworks use abstractions with additional behavior so that we can add specific behavior to the program, by extending the classes of the framework. This pattern decouples the task's execution from its implementation, which eases switching between different implementations and makes programs more modular. Spring Boot is a setup of Spring with *batteries included* as it auto-configures the web application server with the services that are necessary for the web application. This facilitates the developers' work and makes it more efficient. Spring Boot can be integrated with Hibernate, which supports various databases such as MySQL, PostgreSQL, Oracle, SQL Server, among others.

Spring Boot can be used for small problems but it is more appropriate for large-scale applications that use a cloud approach. Usually, multiple applications run in parallel communicating with each other; while some provide user interaction, others perform backend work (e.g., accessing databases or other services).

In terms of performance, Spring Boot has a good performance when compared to some of the other frameworks analyzed.

4.2.6 Laravel

Laravel is a web application framework for PHP, created in 2011. Laravel is considered the most popular and used framework written in PHP language. Moreover, there is extensive documentation for learning about Laravel and modern PHP.

Laravel simplifies the production process and takes much of the pain out of web app developers. It is well known for its clean and elegant PHP code. Laravel possesses a decent amount of packages that could extend its basic functionality (*batteries included*) and simplify tasks such as authentication, routing, and queues. Regarding databases, Laravel supports four database management systems: MySQL, PostgreSQL, SQLite, and Microsoft SQL Server.

In terms of performance, Laravel is the slowest framework when compared to the other frameworks analyzed in this section.

4.2.7 Discussion

A summary of the framework analysis performed in this section is presented in Table 4.3. The characteristics used to compare the web frameworks are the following: programming language in which the

framework is written, the year in which the framework was created, its ease-of-use, performance, and productivity.

Table 4.3: Comparison of web application frameworks.

	Programming Language	Created in	Ease-of-Use	Performance	Productivity
Django	Python	2005	Extensive and detailed documentation Huge community	Low	Highly customizable Easy to read and to maintain
Flask	Python	2010	Good documentation Active community	Low	Flexible Minimalist
Express.js	JavaScript	2010	Good documentation Lots of tutorials and guides	High	Flexible Minimalist
Ruby on Rails	Ruby	2004	Huge community Lacks proper documentation	Very slow runtime speed	Huge amount of third-party add-ons
Spring Boot	Java	2003	Good documentation with guides and tutorials	Good	Simplifies and makes it more efficient the developers' work
Laravel	PHP	2011	Extensive documentation	Very low	Clean code Simplifies tasks related to web development

Spring Boot, Ruby on Rails, and Django are the oldest frameworks, so these frameworks are expected to be more mature than the others. Regarding ease-of-use, all frameworks provide good documentation, except Ruby on Rails. It is worth highlighting the extensive and detailed documentation of Django, with the support of a huge community. The preference for a customizable framework, instead of a lightweight framework without incorporated tools or libraries, excludes Flask or Express.js. This type of light framework is more appropriate for basic apps that do not require database operations. In terms of programming languages, Python is easier to maintain than others like Java or PHP, which favors Django. In conclusion, even with lower performance than the other frameworks analyzed, Django was considered the most appropriate solution for the type of web-based software application to be developed in this thesis.

Chapter 5

Computer Science and Engineering Doctoral Degree Information Management System

This section describes the design and implementation of the solution to the problems of the Fenix system listed in Section 1.2: the DD-IMS. DD-IMS is a software application developed to manage the data regarding the CSE Doctoral Degree in a way that simplifies the job of all stakeholders involved, and facilitates the collection of data used to obtain statistics related to the degree. The system communicates with the FenixEdu API, taking advantage of the data that is already stored in the Fenix system.

Figure 5.1 shows how the new Information System can be integrated into the current processes and data flows involved in the CSE Doctoral Degree. This figure can be seen as an evolution of Figure 1.1, presented in Chapter 1. The differences between these two figures are the following: (i) with the DD-IMS, the CSE department users only need to communicate with this system, instead of sending information to the PGA administrative staff by e-mail and interact with the Fenix system; (ii) the new system accesses the Fenix system to retrieve data that is stored there; and (iii) to satisfy the PGA needs, the new system is capable of generating documents in PDF format (e.g., study plan form, thesis defense jury proposal, etc.) and sending them to the PGA by e-mail.

In order to meet the requirements described in Chapter 3, the system to be developed has the following objectives:

1. **To manage the evaluation of applications of candidates**: Store and manage the data and documents submitted by candidates, namely information related to previous academic degrees, CV, motivation letter, and recommendation letters
2. **To manage PhD students' personal and academic data**: Access the Fenix system to get personal and academic data about PhD students, namely gender, birth date, and PhD start date, and allow students to complement the data that cannot be obtained from Fenix, namely identity number, research unit, financing sources, etc.

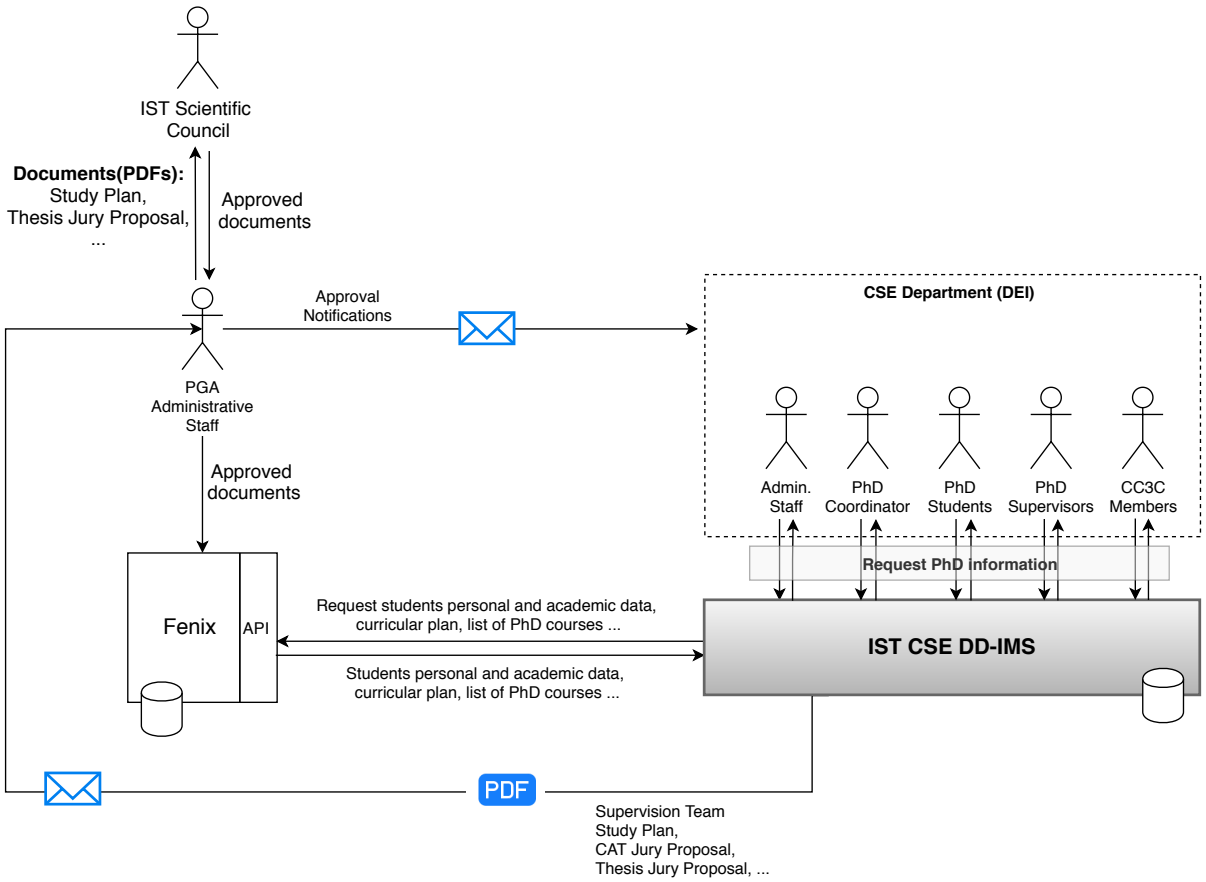


Figure 5.1: Integration of the DD-IMS with the main data flows involved in the CSE Doctoral Degree.

3. **To manage PhD students' progress**: Manage the students' progress during the Doctoral Program, namely their study plan, courses taken (accessible via Fenix system), thesis proposal jury and defense, and thesis jury and defense

4. **To inform students and supervisors**: Remind students of their deadlines and allow supervisors to follow their students' progress

5. **To retrieve statistics**: Return statistics related to students (e.g., being able to get the number of active students in an academic year, get the number of foreign students enrolled in the last 5 years, get the average time that students graduated in 2018/2019 took to finish their PhD, etc.)

Section 5.1 presents the system architecture and its main components. The different types of users and their permissions over the system are specified in Section 5.2. Section 5.3 details the design and implementation of the Evaluation of Applications module. Section 5.4 details the design and implementation of the Registration module. Section 5.5 details the design and implementation of the Study Plan module. Section 5.6 details the design and implementation of the Supervision module. Section 5.7 details the design and implementation of the Curricular Plan module. Section 5.8 details the design and implementation of the Thesis Proposal (CAT) module. Section 5.9 details the design and implementation of the Thesis module. Section 5.10 details the design and implementation of the Statistics module. Section 5.11 details the design and implementation of the Doctoral Program Management module. Section

5.12 describes the communication with Fenix, which is established through the FenixEdu API. Finally, the implementation decisions and the technologies used are presented in Section 5.13.

5.1 Architecture

The architecture for the DD-IMS is a client-server architecture composed of three tiers that are physically separated (three-tier architecture): (i) a client that allows the user to access the application through a web browser; (ii) a web server that encloses the application logic; (iii) a relational database that stores the data about PhD students and their progress. The architecture of the developed system is represented in Figure 5.2. This figure also shows the typical communication between the three main components of the architecture and the FenixEdu API, which can be defined as follows:

1. The user requests a specific action to the *web application*.

2. An *HTTP request* is sent from the client (through a web browser) to the web server.

3. The *web server* receives the request and sends a request to the database or to the FenixEdu API to obtain the necessary data for answering the client request.

4. The *database* or the *FenixEdu API* sends the response to the web server. In the case of a read request, data related to the request is sent as the response. If it is a write request (this type of request is only posed to the database), the response is the confirmation that the data was written with success.

5. The web server receives the response from the database or the FenixEdu API and returns the response to the user, with the requested data, or with the information that the action was successful.

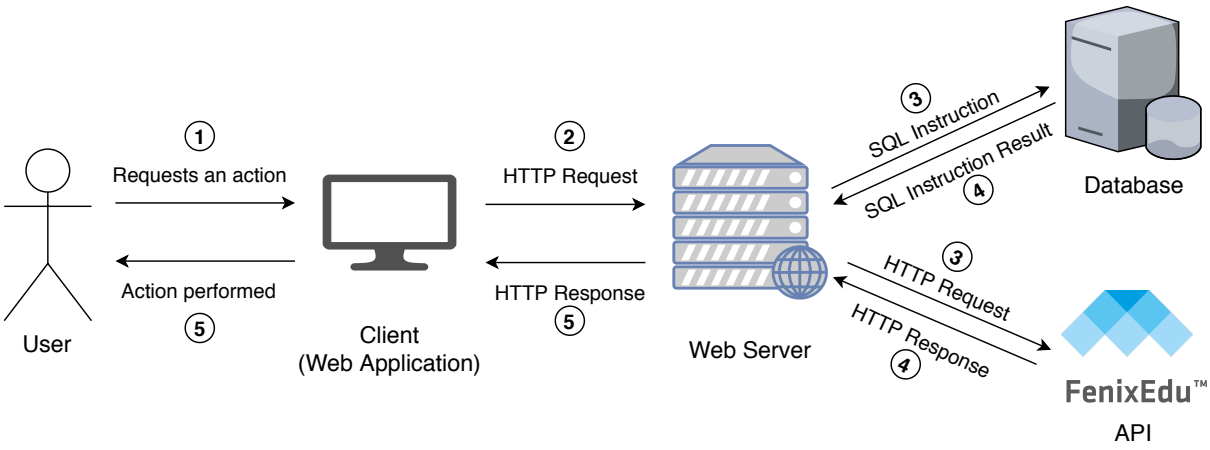


Figure 5.2: Architecture of the System.

Sections 5.1.1, 5.1.2 and 5.1.3 detail the three main components of this architecture.

5.1.1 Client

In this solution, clients are web browsers. These send HTTP requests to the web server. This architectural decision allows users to access the system from any operating system. Users simply need to navigate through their web browser to access the system. The only requirement for the users is having a device with a browser installed and a network connection.

For this system, the client simply displays the information provided by the web server, so it is a thin client. We took this decision to minimize the amount of information processed by the client so that it is possible for users to access the system with less resource consumption on their devices (i.e. memory, disk, computing power).

5.1.2 Web Server

The web server component contains the business logic of the system. DD-IMS modules' can be divided into three layers: Model, View, and Template. The **Model** layer is the interface that contains everything related to the database access. The **View** layer is the middle layer between the Model layer and the Template layer that contains the business logic. This layer uses the business logic to decide what is pulled from the database through the Model and passed to the Template, and it also gets information from the client side and uses the business logic to update the database via the Model layer. Finally, the **Template** layer is responsible for presenting the data that comes from the View layer. This layer controls what should be displayed and how it should be displayed to the users. The layered architecture of a DD-IMS module is presented in Figure 5.3.

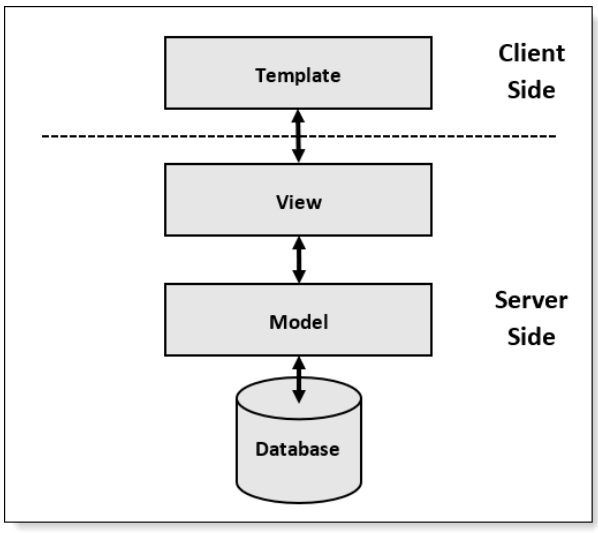


Figure 5.3: Layered architecture of each module of the DD-IMS

Regarding its functionality, the web server can be divided into nine distinct modules: (i) Evaluation of Applications; (ii) Registration; (iii) Study Plan; (iv) Supervision; (v) Curricular Plan; (vi) Thesis Proposal (CAT); (vii) Thesis; (viii) Statistics; and (ix) Doctoral Program Management. For each one of these modules, more details will be given below in this chapter.

5.1.3 Database

The database component is responsible for persistently storing the system data in a structured way and accessible via queries, unlike in the Fenix system where the information related to students' study plan, supervision team, thesis proposal jury and defense minutes, and thesis jury is displayed only through documents. This data is accessed and modified by the web server. We opted to use a relational database to store persistent data. This type of database was chosen because of its ability to combine data from multiple tables and group the values of multiple rows to a single value (aggregation functions as average, count, sum, etc).

Before implementing the physical model, we designed class diagrams with the data models and the relational models for all system's modules, except for those that do not require any data be stored or only need to access data from Fenix or data that belongs to other modules.

5.2 User Roles and Permissions

The DD-IMS is accessed by users with three types of roles: Student, Staff, and Faculty (the users class diagram and the resulting relational model can be visualized in Appendix B). These roles must have different permissions in the system. To handle this issue, user roles permission groups (each permission group gives access to a set of actions). In DD-IMS, each one of the three types of user roles is associated with one or more permission groups, as shown in table 5.1.

Each permission group gives access to specific functionalities of the DD-IMS. The **Student Required** permission group gives access to functionalities related to the academic path of a student in her Doctoral Program: Registration, Study Plan, Supervision, Thesis Proposal, and Thesis. The **Supervisor Required** permission group gives access to the user's supervised students. Both **Coordinator Required** and **Staff Required** permission groups give access to all Doctoral Program students processes and statistics related to Doctoral Program students and candidates. With one of these permissions, users can also visualize and manage information regarding the Doctoral Program faculty members, CC3C members, coordinator, and courses. Also, with the **Coordinator Required**, **Staff Required**, and **CC3C Required** permissions, users can manage the data regarding the evaluation of the applications to the Doctoral Program. The functionalities related to the evaluation of juries, i.e., being able to evaluate both thesis proposal and thesis jury proposals, require one of the following permissions: **Staff Required**, **Coordinator Required**, **CC3C Required**, or **CCP Required**. In addition to these functionalities, all types of users are informed about the pending actions (Alerts) that they must perform in the application.

Regarding authentication, all the system users from IST can log in the DD-IMS through their IST account without needing to create a new account in the DD-IMS.

Table 5.1: Association between user roles and permission groups.

User Role	Permission Groups
Student	**Student Required** - all students belong to this permission group (e.g. allows students to edit supervision, study plan, etc.)
Staff	**Staff Required** - assigned to administrative staff members (e.g. allows to change the Doctoral Program coordinator, CC3C and CCP members, etc.)
Faculty	**Supervisor Required** - assigned to all faculty members (e.g. allows supervisors to visualize their students' processes) **Coordinator Required** - assigned to the faculty member who is the coordinator of a Doctoral Program (e.g. allows a Doctoral Program coordinator to visualize the processes of all students enrolled in that Doctoral Program) **CC3C Required** - assigned to faculty members that are members of the CSE CC3C (e.g. allows a CC3C member to give feedback about applications, as well as to give feedback about thesis proposal and thesis juries) **CCP Required** - assigned to faculty members that are members of the CSE CCP (e.g. allows a CCP member to give feedback about thesis proposal and thesis juries) **Scientific Committee Required** - assigned to faculty members that are members of the CSE Scientific Committee (e.g. allows a Scientific Committee member to visualize a thesis jury)

5.3 Evaluation of Application Module

The Evaluation of Applications module is responsible for the evaluation of the candidates' applications. This section describes this module, including the corresponding BPMN diagram, use cases UML diagram, UML class diagram, and relational model. In this chapter, the use case diagrams, class diagrams, and relational models will only be presented for this module (the remaining ones are available in Appendix A and B).

The Evaluation of Applications business process is represented in Figure 5.4. This process begins with the administrative staff inserting the information about a candidate application to the CSE Doctoral Degree in the system[1]. Then, CC3C members submit their feedback about the application. Based on this feedback, the final decision is taken and submitted by the coordinator. When the applications final decision is already known, the administrative staff (or the coordinator) can create a letter with the applications and generate, in PDF format, the application results, indicating if the applications were approved or declined. If there is something wrong in the letter created (e.g. an application is missing or an application result is incorrect), the administrative staff or the coordinator can delete that letter and create a new one. Then, the coordinator must sign and upload a new version of the document before it is sent by the system to the PGA via e-mail. When all application results are sent to the PGA, the administrative staff can close the applications phase, by archiving all the applications (these are kept in

[1] Ideally, the application information could be obtained through the FenixEdu.

the database as closed). Then, there will be no applications shown on the *Evaluation of Applications* tab until the next application stage starts and the administrative staff inserts the new applications.

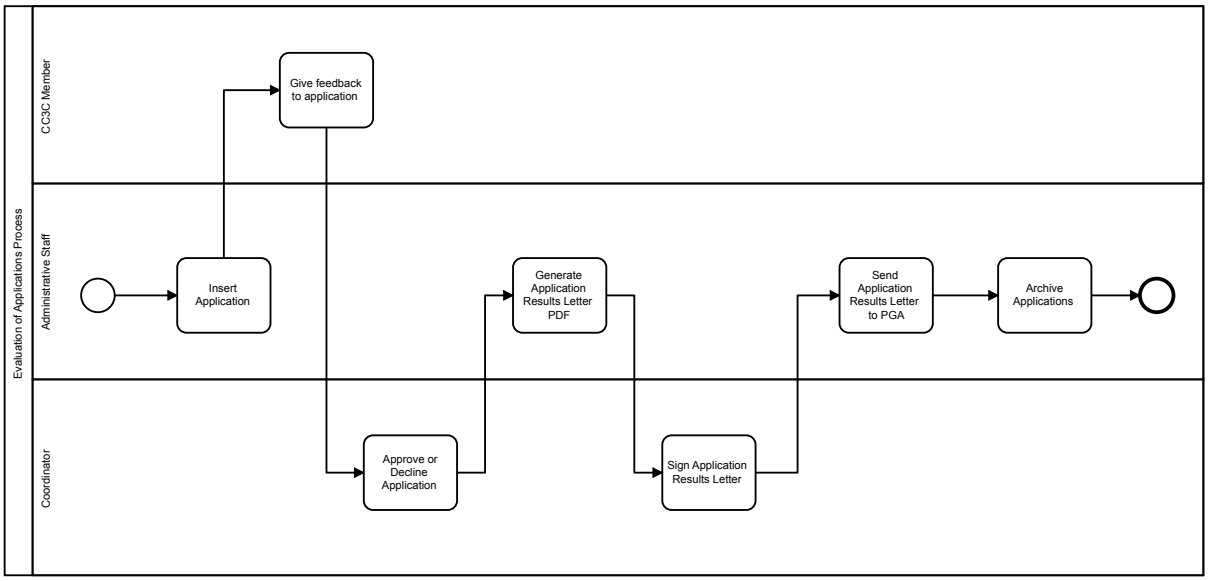


Figure 5.4: Evaluation of Applications Process.

5.3.1 Use Cases

The DD-IMS modules functionality was modeled using UML[2] use cases diagrams. UML use cases diagrams were designed with the diagrams[3] tool. Figure 5.5 shows the diagram that represents the use cases involved in the Evaluation of Applications module. The actors involved in this module are administrative staff, CC3C members, and the coordinator. This module allows the administrative staff to insert an application (*Insert Application*), by entering the candidate information (process ID, name, previous degree), the documents submitted (CV, motivation letter, etc), and the supervision team, if already identified. The administrative staff can also manage application processes (*Edit Application*, *Delete Application*), visualize the applications (*Visualize Application*) and the feedback given by CC3C members (*Visualize CC3C member feedback*), create a results letter which is a document that contains the results of the evaluation of applications to the CSE Doctoral Program (*Create application results letter*), generate it in PDF format (*Generate results letter in PDF format*), and then send the letter to PGA (the system delivers the e-mail automatically) (*Send results letter to PGA*). Finally, the administrative staff can archive applications (*Archive applications*). CC3C members can visualize the applications and the feedback given by the other CC3C members, and submit their opinion about the candidates (*Give feedback to application*). The coordinator can visualize the application information and the feedback given by CC3C members, manage application processes, submit the application result (*Submit application result*), create a results letter, generate it in PDF format, and then upload a new version with her signature (*Upload signed results letter*).

[2]https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-use-case-diagram/
[3]https://app.diagrams.net/

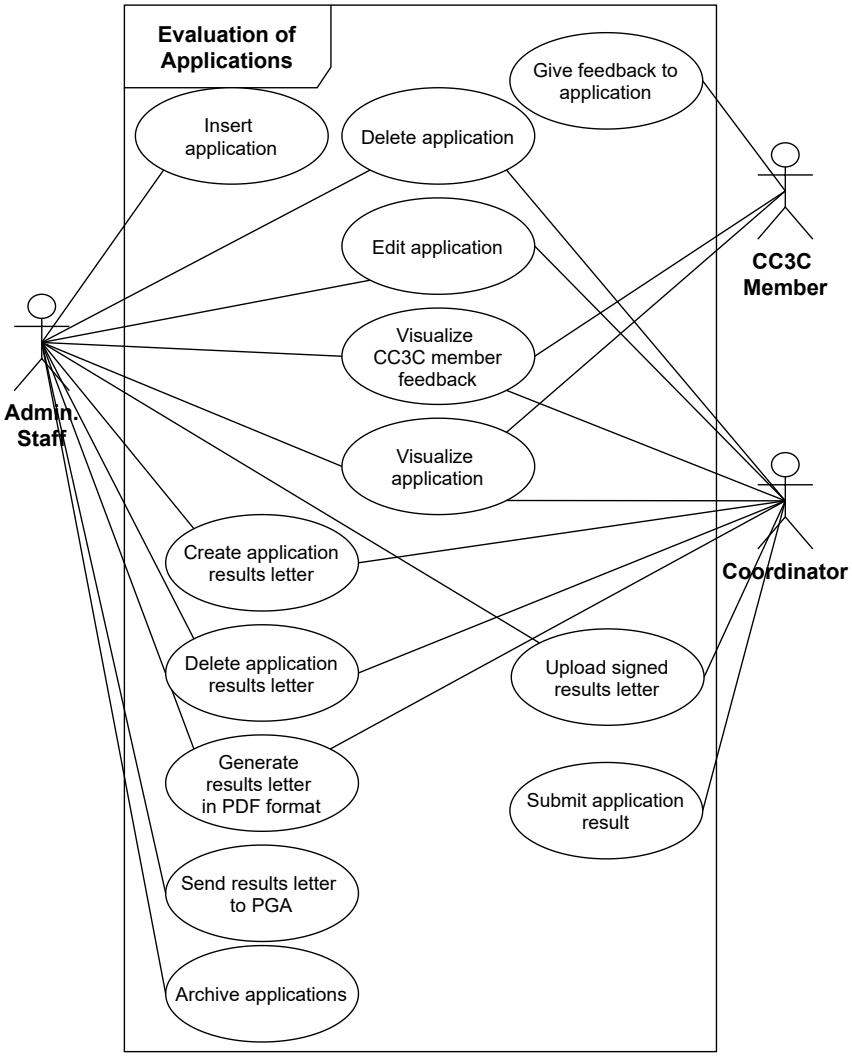


Figure 5.5: Use Cases Diagram for Evaluation of Applications Module.

5.3.2 Class Diagram

The database design was modeled using UML[4] class diagrams. UML class diagrams were designed with the diagrams[5] tool. Figure 5.6 shows the class diagram for the Evaluation of Applications module. The data models created for this module are: (i) **Application** model, which represents a candidate's application; (ii) **Recommendation Letter** model, which represents a recommendation letter document that is associated with one application; (iii) **Previous Degree Certificate** model, which represents a previous degree certificate document that is associated with one application; (iv) **Candidate Supervision** model, which is associated with an application, represents the supervision team of the candidate; (v) **Candidate Faculty Supervisor** model, represents the faculty supervisors that belong to a candidate's supervision team; (vi) **Candidate Non Faculty Supervisor** model, represents the non faculty supervisors that belong to a candidate's supervision team; (vii) **CC3C Evaluation** model, which represents the association between a CC3C member and an application, storing the feedback given by the CC3C member; and (viii) **Application Results Letter** model, that represents a results letter and must be associated with

[4] http://agilemodeling.com/artifacts/classDiagram.htm
[5] https://app.diagrams.net/

one or more applications.

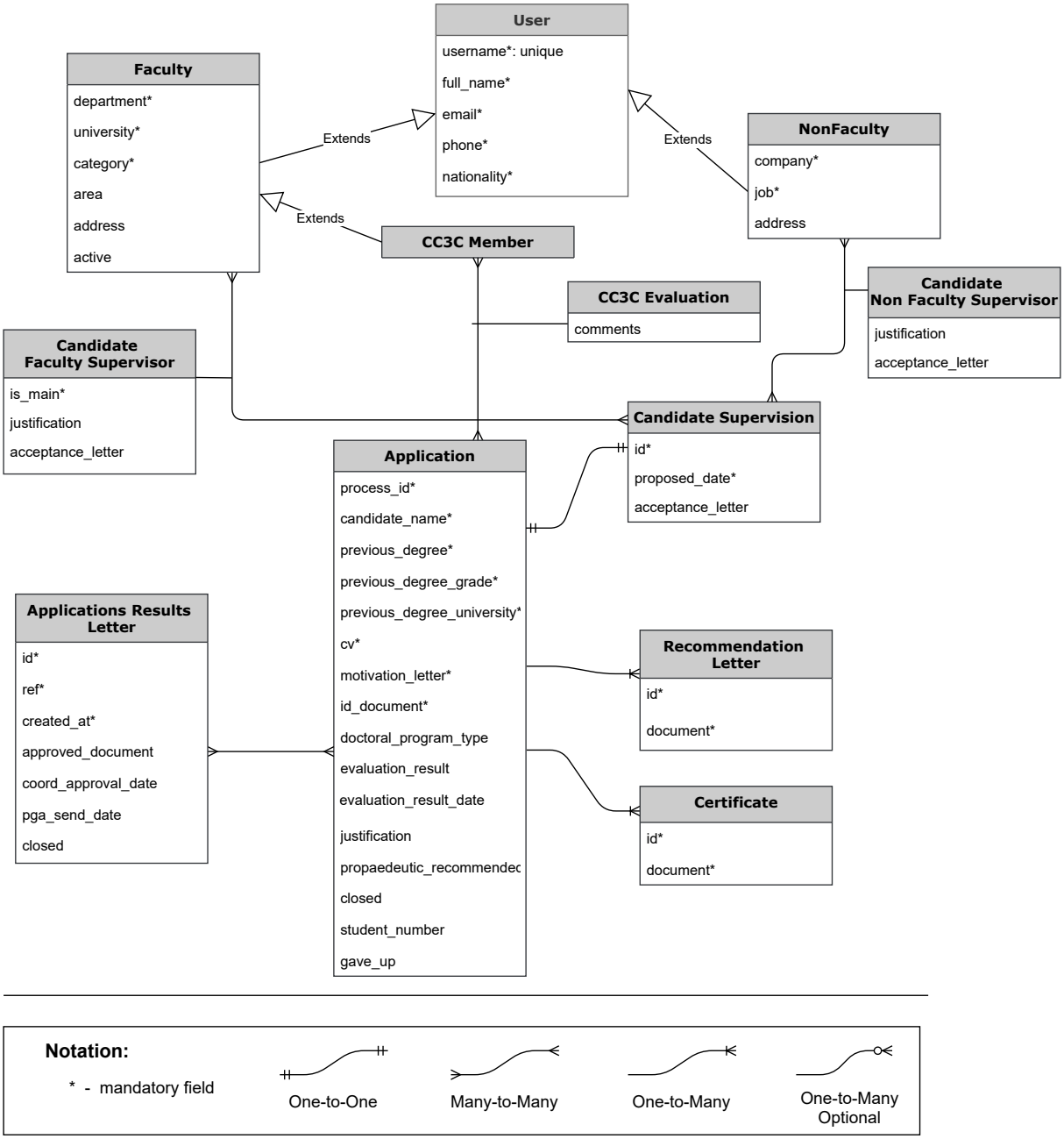


Figure 5.6: Class Diagram for Evaluation of Applications module.

5.3.3 Relational Model

The relational model was obtained from the class diagram described above, by applying conversion rules[6]. The relational model presented below is an optimized version of the original relational model. This optimized version reduces the number of tables and is closer to the concrete implementation of the database than the original one.

To represent the relational models, the notation used was the following (Primary Key is underlined and Foreign Key is represented as FK):

$table_1(\underline{attr_1}, attr_2, \ldots)$

$table_2(\underline{attr_3}, attr_4, \ldots)$

$attr_3$: FK($table_1$)

The integrity constraints and null constraints of the relational model are represented below each relation.

Application(process_id, candidate_name, previous_degree, previous_degree_grade, previous_degree_university, cv, motivation_letter, id_document, doctoral_program_type, evaluation_result, evaluation_result_date, justification, propaedeutic_recommended, closed, student_number, gave_up)

- candidate_name: NOT NULL
- previous_degree: NOT NULL
- previous_degree_grade: NOT NULL
- previous_degree_university: NOT NULL
- cv: NOT NULL
- motivation_letter: NOT NULL
- id_document: NOT NULL
- **Integrity Constraint**: The value of the attribute *evaluation_result* can take one of the following values:
  1. "Approved"
  2. "Approved Conditionally"
  3. "Approved with Propaedeutic Courses"
  4. "Declined"
  5. "Will be re-evaluated in the next application period"

RecommendationLetter(id, application_id, document):

- application_id: FK(Application)
- document: NOT NULL

Certificate(id, application_id, document):

- application_id: FK(Application)
- document: NOT NULL

ApplicationResultsLetter(id, ref, created_at, approved_document, coord_approval_date, pga_send_date, closed):

- ref: NOT NULL

- created_at: NOT NULL
- **Integrity Constraint**: The value of the attribute coord_approval_date must be earlier than the value of the attribute pga_send_date

application_applicationresultsletter(application_id, letter_id):

- application_id: FK(Application)
- letter_id: FK(ApplicationResultsLetter)

CC3C_Evaluation(username, process_id, comments):

- username: FK(CC3C_Member)
- process_id: FK(Application)

CandidateSupervision(id, application_id, proposed_date, acceptance_letter):

- application_id: FK(Application)
- proposed_date: NOT NULL

CandidateFacultySupervisor(supervisor_username, supervision_id, is_main, justification, acceptance_letter):

- supervisor_username: FK(Faculty)
- supervision_id: FK(CandidateSupervision)
- is_main: NOT NULL

CandidateNonFacultySupervisor(supervisor_username, supervision_id, justification, acceptance_letter):

- supervisor_username: FK(NonFaculty)
- supervision_id: FK(CandidateSupervision)

5.3.4 User Interface

To have access to this module functionalities, the users involved in this module (administrative staff, CC3C, and coordinator) have access to a tab called ”Evaluation of Application”. On the main page of this tab, shown in Figure 5.7, a table with the current applications in evaluation is shown. On this page, the administrative staff members visualize a button called ”Insert New Application”, which will redirect them to a page where they can insert the candidate’s information and supervision team (if already defined). Also, both administrative staff and coordinator visualize a dropdown button called ”Results Letters”. When clicked, this button allows users to visualize and perform actions on the application results letters already created (i.e. Generate PDF, Upload Document Signed, Send to PGA, Delete) and to create a new results letter.

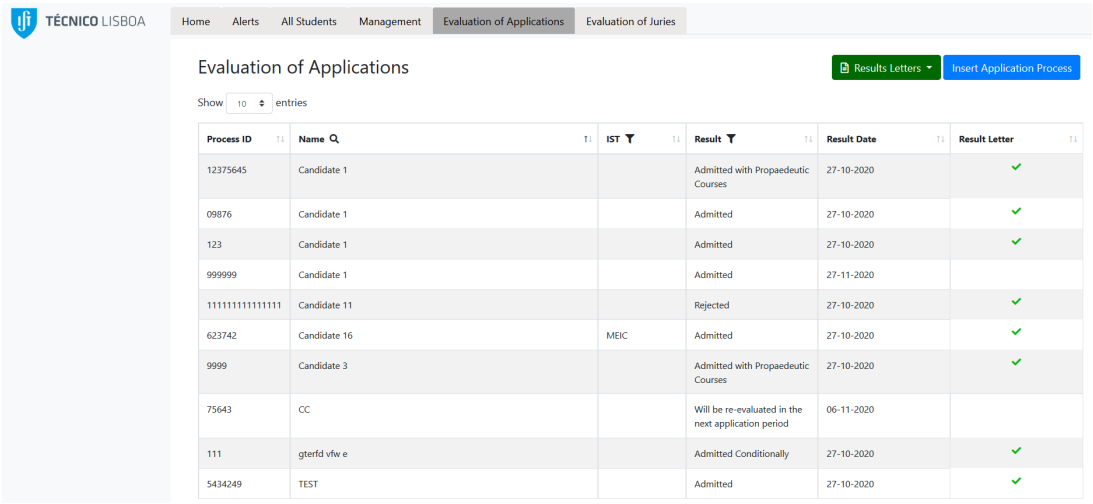


Figure 5.7: Screenshot of the main page of Evaluation of Applications tab.

In addition to the main page mentioned, users also have access to a page where they can visualize the information related to a specific application process. On this page, the information displayed is divided into 3 left side menu options: (i) Candidate, which displays the candidate's information and allows the download of the application documents submitted (i.e. CV, Motivation Letter, Recommendation Letters, Previous Degrees Certificates); (ii) Supervision, which lists the candidate's supervision team (if already defined); and (iii) CC3C Feedback, which allows CC3C members to submit their feedback about the application and to visualize the feedback submitted by the others.

5.4 Registration Module

The Registration module is responsible for the students' registration in the DD-IMS. The actors involved in this module are students, administrative staff, supervisors, and the coordinator.

The Registration business process is represented in Figure 5.8. After being enrolled in the CSE Doctoral Program and registered in Fenix, the students (i.e., candidates whose applications were approved) can register in DD-IMS. To register in the system, students start to insert their personal information and academic data, related to their current Doctoral Program and their previous degrees. After being registered, students can visualize and edit their academic and personal data. This data can also be visualized by the student's supervisors, the CSE Doctoral Program coordinator, and the CSE administrative staff. The system allows the administrative staff to validate the students' data and edit it if some error is found. Concerning the academic data, one of the characteristics presented is student status. A student can have one of four possible statuses: (i) Active; (ii) Suspended; (iii) Abandoned; and (iv) Concluded. The administrative staff is able to change the student status from Active to Abandoned/Suspended or from Suspended to Active/Abandoned. The administrative staff is also able to add a thesis deadline extension for the students that requested an extension, indicated the number of days extended.

The use case diagram related to the Registration module is presented in appendix A. The class diagram and relational model of this module are presented in Appendix B.

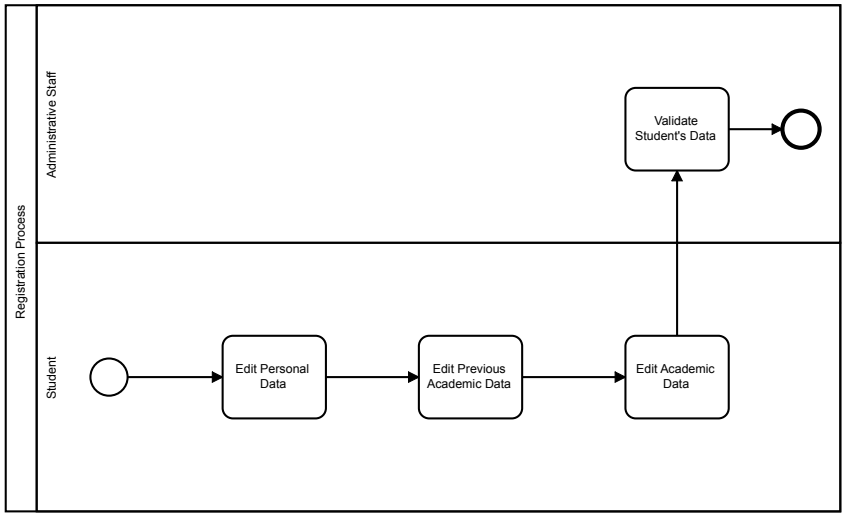


Figure 5.8: Registration Process.

5.4.1 User Interface

Students have access to a tab called ”Registration”, where they can register in the DD-IMS, by inserting their personal and academic information, and visualize the data inserted by them if already registered. This data can always be edited. When registering, a form wizard (multi-step form) is presented to students, as shown in Figure 5.9. This wizard is divided into 4 steps: Personal Information, Previous Degrees, Academic Information, and Financing Sources. This division was made to avoid the visualization of too much information at the same time. Some of the form fields are automatically obtained from Fenix is displayed and cannot be modified (e.g. Name, Email).

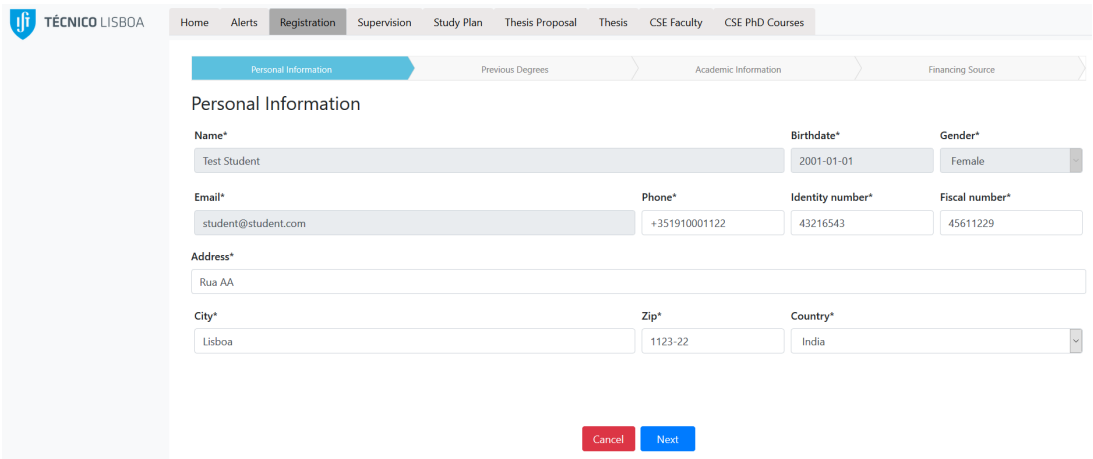


Figure 5.9: Screenshot of the Registration form wizard.

The administrative staff and the coordinator are also able to visualize the students' personal and academic information inside a tab called ”All Students”. On this tab, a table with the active Doctoral Program students is shown. This table displays the student's name, supervisor name, IST ID, status (i.e. Active, Abandoned, Suspended, Concluded), PhD start date, and PhD end date (presents the deadline if the student has not concluded her PhD yet). By clicking on a row of this students table, users are redirected to a page that contains the student information regarding her Doctoral Program process,

which includes not only the 3 left side menu options related to the Registration module (i.e. Personal Information, Previous Degrees, Academic Information) but also other left side menu options related to other modules.

Similarly, the students' supervisors have access to the same information in a tab called "My Students" but, instead of having access to all active students, supervisors only have access to their supervised students.

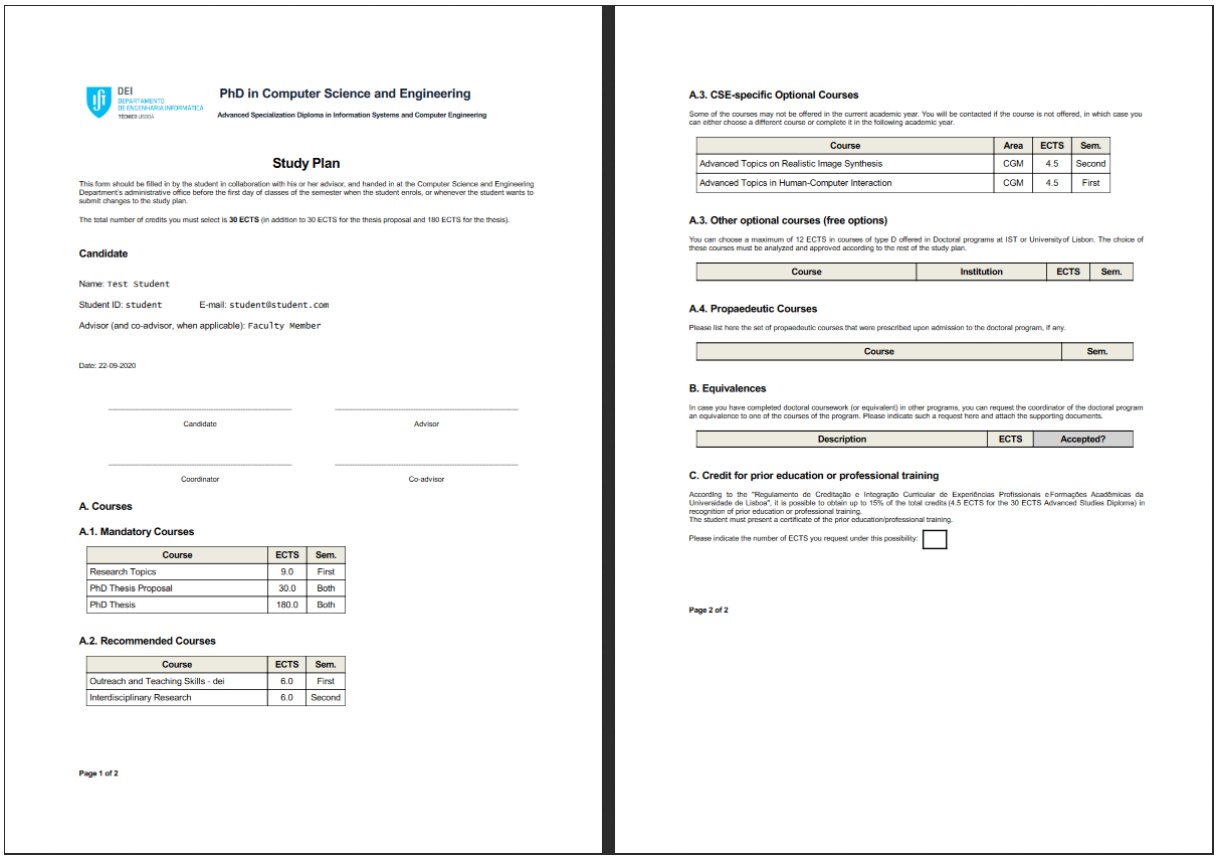
5.5 Study Plan Module

The Study Plan module is responsible for managing the students' study plan. The actors involved in this module are students, administrative staff, supervisors, and the coordinator.

This module allows registered students to create their study plan. To ensure that all the students currently enrolled in the CSE Doctoral Program can register in DD-IMS and submit their study plan, the system supports not only the current curriculum but also older curriculums, namely 2015/16, 2016/17, 2017/18, in which some students have defined their study plan. If the study plan submitted is valid, the system allows the student's supervisors (if already defined) to approve it. The coordinator approves the student's supervision team after the study plan is approved by one of the supervisors or if the supervision is not defined yet. In the last case, the coordinator approves not only as a coordinator but also as a temporary supervisor. If the study plan is declined by a supervisor or a coordinator (a justification must be given when declining a study plan), the student receives a notification and must insert a new one. If not, the administrative staff can generate a PDF document that contains the student's study plan and then upload a new version of that document signed by all the actors involved. An example of a study plan document is shown in Figure 5.10. Then, the system sends this document to the PGA by e-mail. The system allows the administrative staff to visualize the e-mail content and then confirm the submission. To finish the study plan approval, the administrative staff submits the date of the Scientific Council approval when an e-mail from PGA with that information is received. Before the approvals workflow is concluded, both the administrative staff members and the coordinator can revert the workflow to a previous stage (e.g. the coordinator can approve a study plan and then revert her decision, going back to the stage where the supervisors have approved the study plan). The Study Plan business process is represented in Figure 5.11.

The system also allows students to modify their approved study plan. The approval workflow is similar to the one presented above. In this case, the PDF document generated only shows the study plan modifications (added and removed courses) instead of displaying the entire study plan.

Besides these functionalities, all the four types of actors allowed to access this module can visualize the student's study plan and the status of the approval workflow. The use case diagram related to the Study Plan module is presented in appendix A. The class diagram and relational model of this module are presented in Appendix B.



PhD in Computer Science and Engineering

Advanced Specialization Diploma in Information Systems and Computer Engineering

Study Plan

This form should be filled in by the student in collaboration with his or her advisor, and handed in at the Computer Science and Engineering Department's administrative office before the first day of classes of the semester when the student enrols, or whenever the student wants to submit changes to the study plan.

The total number of credits you must select is **30 ECTS** (in addition to 30 ECTS for the thesis proposal and 180 ECTS for the thesis).

Candidate

Name: Test Student

Student ID: student E-mail: student@student.com

Advisor (and co-advisor, when applicable): Faculty Member

Date: 22-09-2020

Candidate Advisor

Coordinator Co-advisor

A. Courses

A.1. Mandatory Courses

Course	ECTS	Sem.
Research Topics	9.0	First
PhD Thesis Proposal	30.0	Both
PhD Thesis	180.0	Both

A.2. Recommended Courses

Course	ECTS	Sem.
Outreach and Teaching Skills - dei	6.0	First
Interdisciplinary Research	6.0	Second

Page 1 of 2

A.3. CSE-specific Optional Courses

Some of the courses may not be offered in the current academic year. You will be contacted if the course is not offered, in which case you can either choose a different course or complete it in the following academic year.

Course	Area	ECTS	Sem.
Advanced Topics on Realistic Image Synthesis	CGM	4.5	Second
Advanced Topics in Human-Computer Interaction	CGM	4.5	First

A.3. Other optional courses (free options)

You can choose a maximum of 12 ECTS in courses of type D offered in Doctoral programs at IST or University of Lisbon. The choice of these courses must be analyzed and approved according to the rest of the study plan.

Course	Institution	ECTS	Sem.

A.4. Propaedeutic Courses

Please list here the set of propaedeutic courses that were prescribed upon admission to the doctoral program, if any.

Course	Sem.

B. Equivalences

In case you have completed doctoral coursework (or equivalent) in other programs, you can request the coordinator of the doctoral program an equivalence to one of the courses of the program. Please indicate such a request here and attach the supporting documents.

Description	ECTS	Accepted?

C. Credit for prior education or professional training

According to the "Regulamento de Creditação e Integração Curricular de Experiências Profissionais e Formações Académicas da Universidade de Lisboa", it is possible to obtain up to 15% of the total credits (4.5 ECTS for the 30 ECTS Advanced Studies Diploma) in recognition of prior education or professional training.
The student must present a certificate of the prior education/professional training.

Please indicate the number of ECTS you request under this possibility:

Page 2 of 2

Figure 5.10: Study Plan document in PDF format.

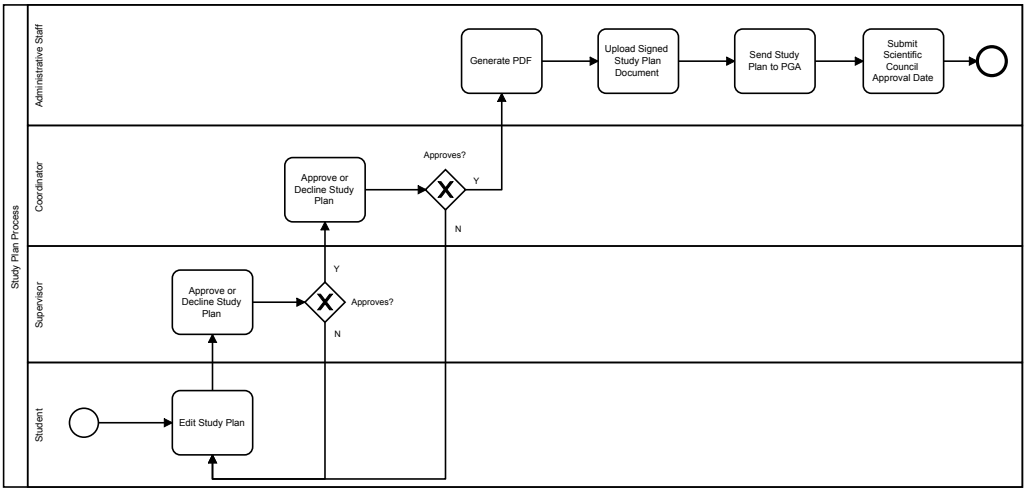


Figure 5.11: Study Plan Process.

5.5.1 User Interface

Students have access to a tab called ”Study Plan”, where they can submit their study plan, choosing the list of courses that they want to enroll. As it happens in the Registration module, a form wizard is presented to students, as shown in Figure 5.12. This wizard is divided into 7 or 8 steps: CMU Courses (Only shown if the student's Doctoral Program is under the CMU Portugal Partnership[6]), Mandatory Courses,

[6]https://www.cmuportugal.org/

Recommended Courses, Optional Courses, Free Courses, Propaedeutic Courses, Prior Credits, and Summary where they can visualize the courses selected before.

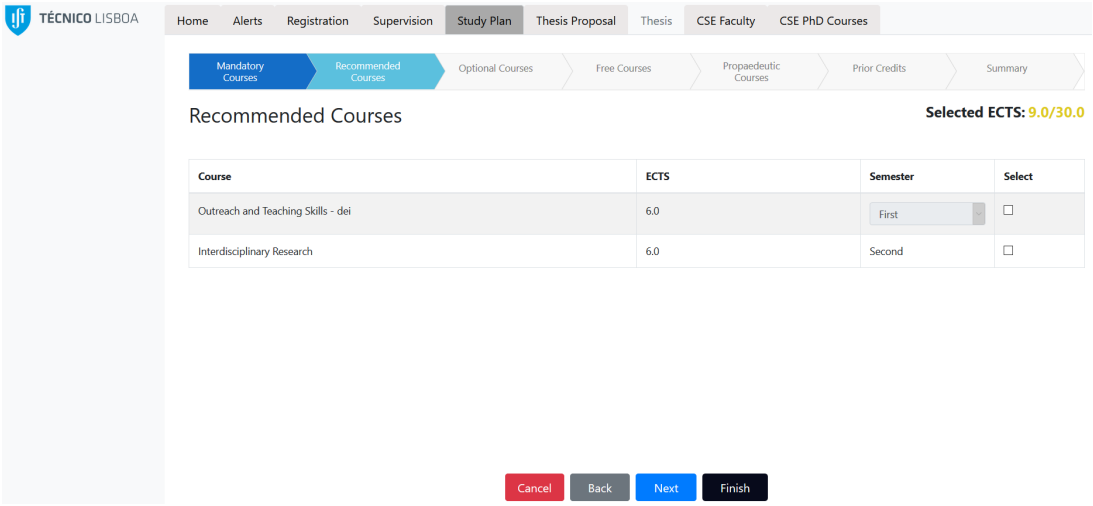


Figure 5.12: Screenshot of the Study Plan form wizard.

As in the Registration module, the administrative staff and the coordinator are also able to visualize the students' study plan inside "All Students" tab. The study plan and its approvals workflow are displayed on the student's process page, in a left side menu option called "Study Plan". Here the coordinator can approve or decline the study plan and the administrative staff is able to generate it in PDF format, upload a version of the document signed by the intervenients (i.e. student, supervisors, and coordinator), send it to PGA, or submit the date of the IST Scientific Council approval.

Similarly, the students' supervisors have access to the same information on the "Study Plan" left side menu option of the student's process page, inside "My Students" tab. Here they can approve or decline the study plan.

5.6 Supervision Module

The Supervision module is responsible for managing the students' supervision team. The actors involved in this module are students, administrative staff, supervisors, and the coordinator.

This module allows registered students to insert their supervision team. If the supervision submitted is valid (cannot have more than 3 supervisors and at least one of the supervisors must belong to the IST CSE department), the system allows the selected supervisors to approve it. When one of the supervisors approves the supervision team, the coordinator can approve the student's supervision. If someone declines the supervision, the student receives a notification and must insert a new supervision team. If not, the administrative staff can generate a PDF document that contains the student's supervision team and then upload a new version of that document signed by all the actors involved. An example of a supervision team inclusion document is shown in Figure 5.13. Then, the administrative staff can send this document to the PGA by e-mail. The system allows the administrative staff members to visualize the e-mail content and then confirm the submission. To finish the supervision approval, the administrative

staff can submit the date of the Scientific Council approval after receiving an e-mail from the PGA with that information. Before the approvals workflow is concluded, both the administrative staff members and the coordinator can revert the workflow to a previous stage (e.g. the coordinator can approve a supervision team and then revert her decision, going back to the stage where the supervisors have approved the supervision team). The Supervision inclusion business process is represented in Figure 5.14.

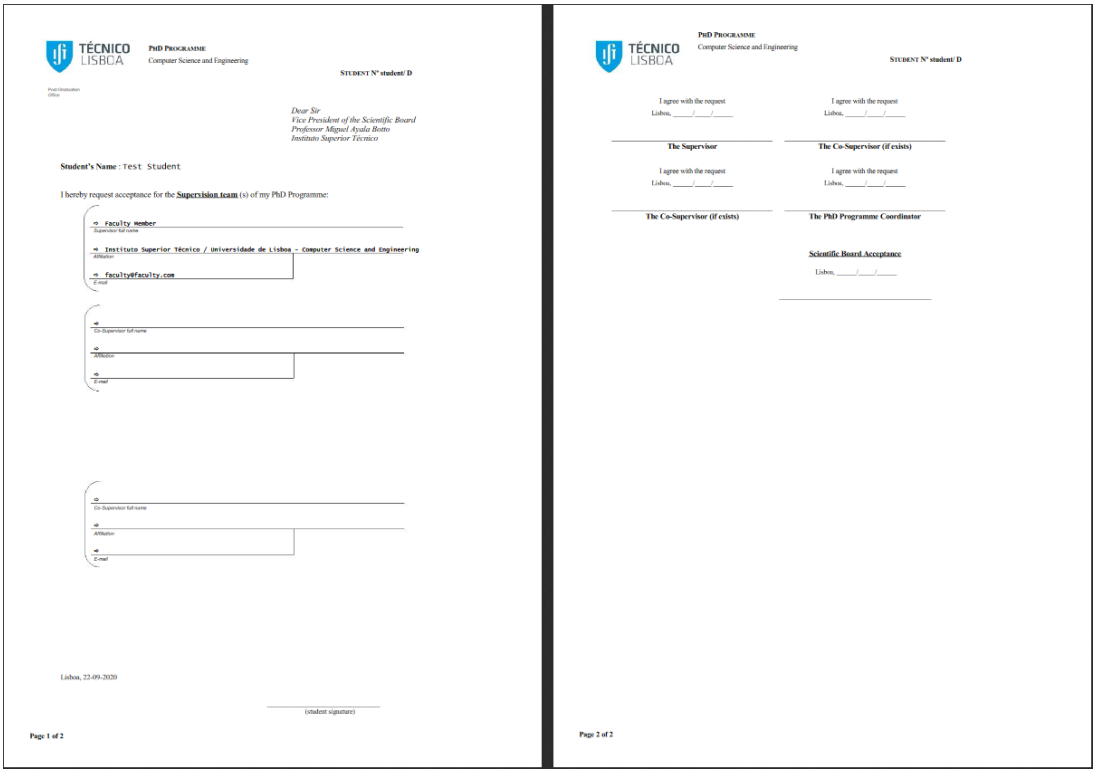


Figure 5.13: Supervision document in PDF format.

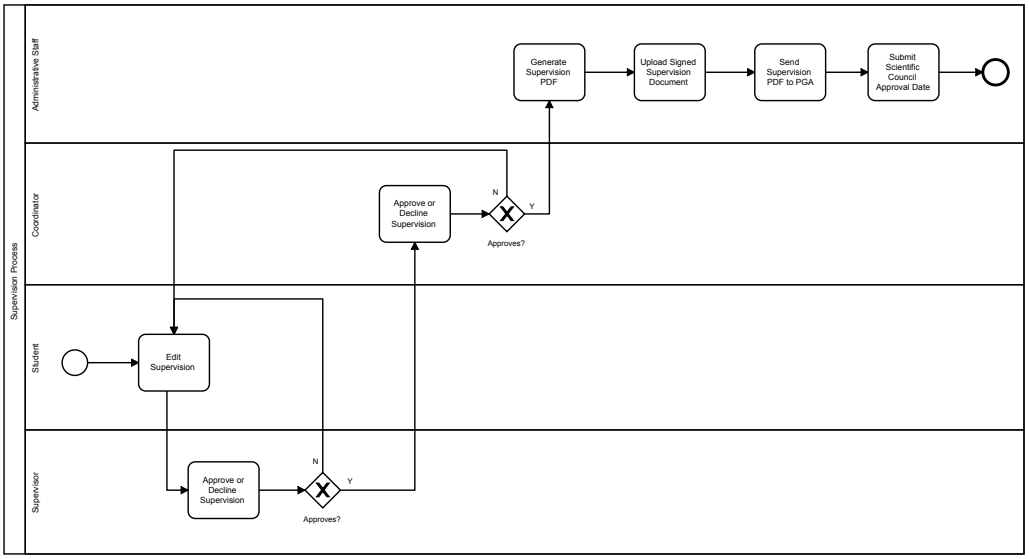


Figure 5.14: Supervision Process.

The system also allows the students to modify their approved supervision. The approval workflow

in the system is similar to the one presented above. The only difference between these two processes is the PDF documents generated. While in the supervision creation process a single PDF document is generated with all the supervisors included, in the supervision modification process a PDF document is generated for each supervisor modification. The document generated depends on the type of modification:

- Supervisor inclusion, when a new supervisor is added to the supervision team
- Supervisor removal, when a supervisor from the previous supervision team is removed
- Supervisor change, when a supervisor from the previous supervision team is replaced by a new supervisor
- Co-Supervisor inclusion, when a new co-supervisor is added to the supervision team
- Co-Supervisor removal, when a co-supervisor from the previous supervision team is removed
- Co-Supervisor change, when a co-supervisor from the previous supervision team is replaced by a new co-supervisor

Besides these functionalities, all four types of actors allowed to access this module can visualize the student's supervision and the state of the approval workflow. The use case diagram related to the Supervision module is presented in appendix A. The class diagram and relational model of this module are presented in Appendix B.

5.6.1 User Interface

Students have access to a tab called ”Supervision”, where they can submit their supervision team. The form provided, which is shown in Figure 5.15 allows students to select their Faculty Supervisors, Faculty Co-Supervisors, and Non-Faculty Supervisors. For the Faculty Supervisors and Co-Supervisors, students can choose the faculty name from a select box, where they can find all the CSE Department faculty members. An option named ”Other” is also present to allow students to choose faculty members from other departments or universities. When this option is selected, other additional fields appear in the form (i.e. Full Name, Department, and University).

As in other modules, the administrative staff and the coordinator are also able to visualize the students' study plan inside ”All Students” tab. The supervision and its approvals workflow are displayed on the student's process page, in a left side menu option called ”Supervision”. Here the coordinator and administrative staff can perform the same actions as the ones described for the Study Plan module.

Similarly, the students' supervisors have access to the same information on the ”Supervision” left side menu option of the student's process page, inside ”My Students” tab. Here they can approve or decline the supervision.

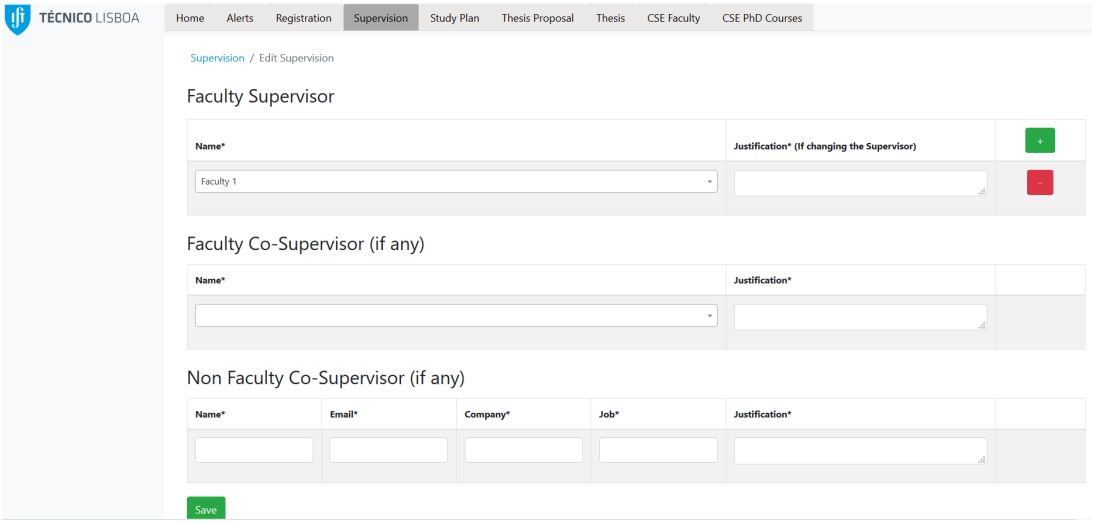


Figure 5.15: Screenshot of the Supervision Team form.

5.7 Curricular Plan Module

The Curricular Plan module is responsible for providing the students' data regarding their enrolled courses and the courses that they have taken. This module allows coordinators, supervisors, administrative staff members, and the students themselves to visualize the courses that they have taken (with the corresponding grade and enrollment year and semester) or are enrolled in the current semester. The use case diagram related to the Curricular Plan module is presented in appendix A. The class diagram of this module is presented in Appendix B.

5.7.1 User Interface

Students have access to their curricular plan information inside the "Study Plan" tab. Administrative staff, the coordinator, and supervisors also have access to this information, in the "Study Plan" left side menu option of the student's process page, as shown in Figure 5.16. For each of the courses selected in the study plan, the enrollment period (if the student has already enrolled in the course) and the correspondent grade (if the student has already been evaluated) are displayed.

5.8 Thesis Proposal (CAT) Module

The Thesis Proposal module is responsible for the thesis proposal management, i.e., thesis proposal jury proposal and thesis proposal defense minutes. The actors involved in this module are students, administrative staff, supervisors, the coordinator, CC3C members, and CCP members.

This module allows supervisors to submit their students' thesis proposal jury proposals. Supervisors are requested to insert the thesis proposal information (thesis title and abstract) and the list of jury members as well as a justification for their inclusion. If the jury proposal submitted is valid (one of the jury members must be selected as president), the system allows the CC3C and CCP, in this order, to give feedback about it. Based on this feedback, the final decision (approve/decline) is taken and

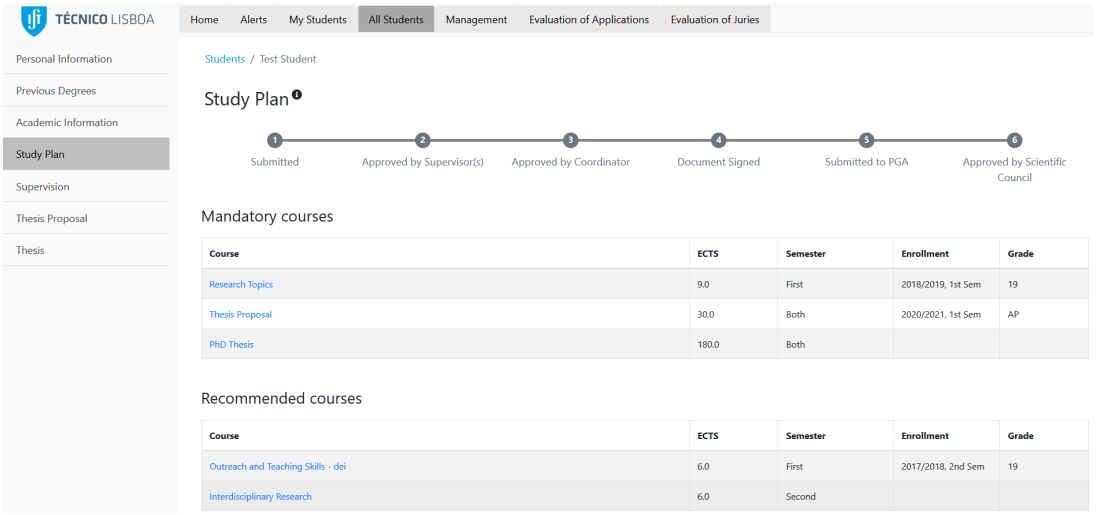


Figure 5.16: Screenshot of the Study Plan left side menu option, containing the student's curricular plan.

submitted by the coordinator and the administrative staff. Both the coordinator and the administrative staff can revert the decisions submitted. If the thesis proposal jury is declined by the CC3C or CCP, the supervisor receives a notification and must edit the jury proposal. If not, the administrative staff can generate a PDF document that contains the student's thesis proposal title and abstract, and the list of jury members with a justification attached, and then upload a new version of the document signed by the coordinator. Then, the administrative staff sends this document to the PGA by e-mail. The system allows the administrative to visualize the e-mail content and then confirm the submission. To finish the thesis proposal jury approval, the administrative staff can submit the date of the Scientific Council approval after receiving an e-mail from the PGA with that information. The Thesis Proposal Jury business process is represented in Figure 5.17.

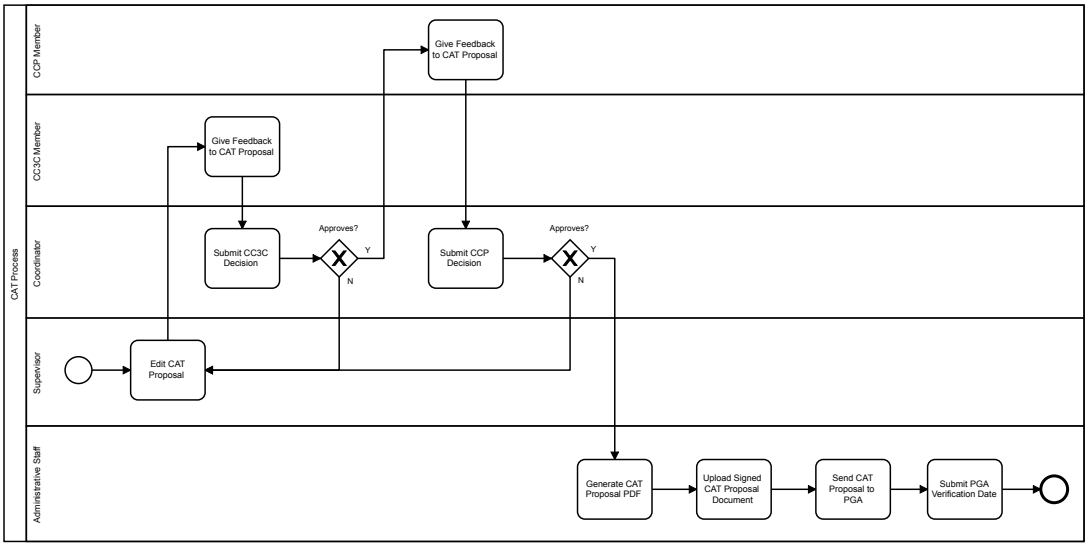


Figure 5.17: Thesis Proposal (CAT) Jury Process.

When the jury proposal approval workflow is concluded, the student's supervisor can schedule the thesis proposal defense date. After the defense takes place, the supervisor has the opportunity to

submit the thesis proposal defense minutes in the system, indicating the thesis proposal grade and the criteria used to assign that grade. Then, the supervisor can generate a PDF document that contains the thesis proposal defense minutes. After that, a new version of this document can be uploaded first by the supervisor (with the jury members' signature) and then by the coordinator (with her signature). The administrative staff can send the document to the PGA by e-mail. To finish the thesis proposal defense minutes approval, the administrative staff must submit the date of the Scientific Council approval. The Thesis Proposal Defense business process is represented in Figure 5.18.

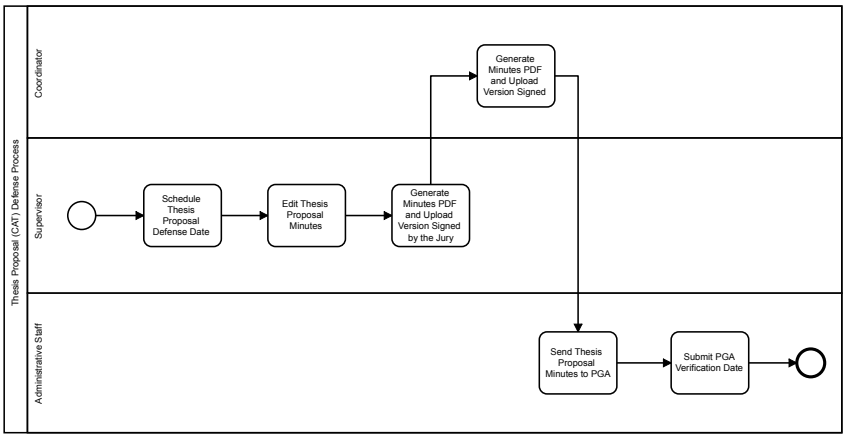


Figure 5.18: Thesis Proposal (CAT) Defense Process.

The use case diagram related to the Thesis Proposal module is presented in appendix A. The class diagram and relational model of this module are presented in Appendix B.

5.8.1 User Interface

The actors involved in the thesis proposal jury evaluation process (administrative staff, the coordinator, CC3C members, and CCP members) have access to a tab called ”Evaluation of Juries”. This tab is divided into 2 left side menu options: Thesis Proposal and Thesis. For the Thesis Proposal part, the main page contains a table with the thesis proposal juries submitted. Each one of these juries can be visualized in more detail by clicking in the correspondent table row. Inside the thesis proposal jury page, there are 3 left side menu options: (i) Jury, which contains the jury members, specifying who is the president of the jury and who are the supervisors; (ii) CC3C Feedback (presented in Figure 5.19), where a table showing the feedback given by the CC3C members is displayed and the coordinator can approve or decline the jury after analyzing the feedback received; and (iii) CCP Feedback, where a table showing the feedback given by the CCP members is displayed and the coordinator can approve or decline the jury after analyzing the feedback received.

Students have access to a tab called ”Thesis Proposal”, where they can visualize information regarding their thesis proposal. This information is divided into 3 left side menu options: (i) ”Thesis Proposal”, which shows the thesis title and abstract; (ii) Jury, where the jury members can be visualized; and (iii) ”Defense”, which displays the information regarding the thesis proposal defense, including the grade, comments, and the criteria used during the evaluation which is composed of 5 items: Problem descrip-

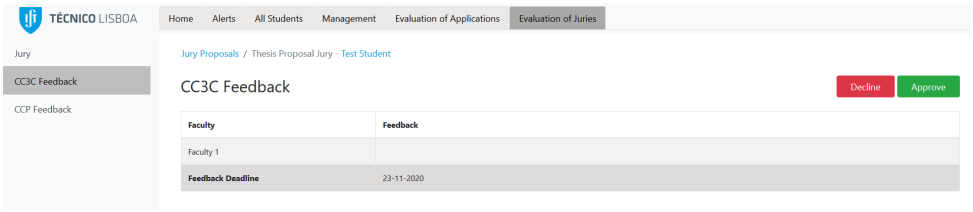


Figure 5.19: Screenshot of the CC3C Feedback left side menu option, inside the thesis proposal jury page.

tion and innovation potential, Quality of the state-of-the-art survey, Proposed approach and methodology, Planning of activities, and Oral Presentation and Discussion (are graded from Fail to Outstanding).

As in other modules, the administrative staff and the coordinator are also able to visualize the students' thesis proposal information inside "All Students" tab. The details about the thesis proposal are shown on the student's process page, in a left side menu option called "Thesis Proposal". This option contains 2 sub-options, "Jury" and "Defense", which display information about the thesis proposal jury and defense, respectively.

Similarly, the students' supervisors have access to the same information on the "Thesis Proposal" left side menu option and "Jury" and "Defense" sub-options of the student's process page, inside "My Students" tab. On these sub-options, they can submit the thesis proposal jury and the thesis proposal defense minutes.

5.9 Thesis Module

The Thesis module is responsible for the management of information concerning the PhD thesis, i.e., thesis jury proposal and thesis defense minutes. The actors involved in this module are students, administrative staff, supervisors, coordinator, CC3C members, CCP members, and Scientific Committee members.

This module allows supervisors to submit their students' thesis jury proposals. Supervisors are requested to insert the thesis proposal information (thesis title and abstract) and the list of jury members. If the jury proposal submitted is valid (two of the jury members must be selected as rapporteurs), the system allows the CC3C and CCP, in this order, to give feedback about it. Based on this feedback, the final decision (approve/decline) is taken and submitted by the coordinator. Both the coordinator and the administrative staff can revert the decisions submitted. If the thesis jury is declined by the CC3C or CCP, the supervisor receives a notification and must edit the jury proposal. If not, the administrative staff can appoint the president of the jury from the list of full professors eligible to be jury presidents (CSE Scientific Committee). After that, the administrative staff can generate two PDF documents: (i) a document that contains the student's thesis title and abstract, and the list of jury members with a justification attached; and (ii) a document that indicates who was the full professor selected as president of the jury. After that, a new version of the documents signed by the coordinator can be uploaded. Then, the administrative staff can send the documents to the PGA by e-mail. The system allows the administrative to visualize the e-mail content and then confirm the submission. To finish the thesis jury

approval, the administrative staff can submit the date of the Scientific Council approval after receiving an e-mail from the PGA with that information. The Thesis Jury business process is represented in Figure 5.20.

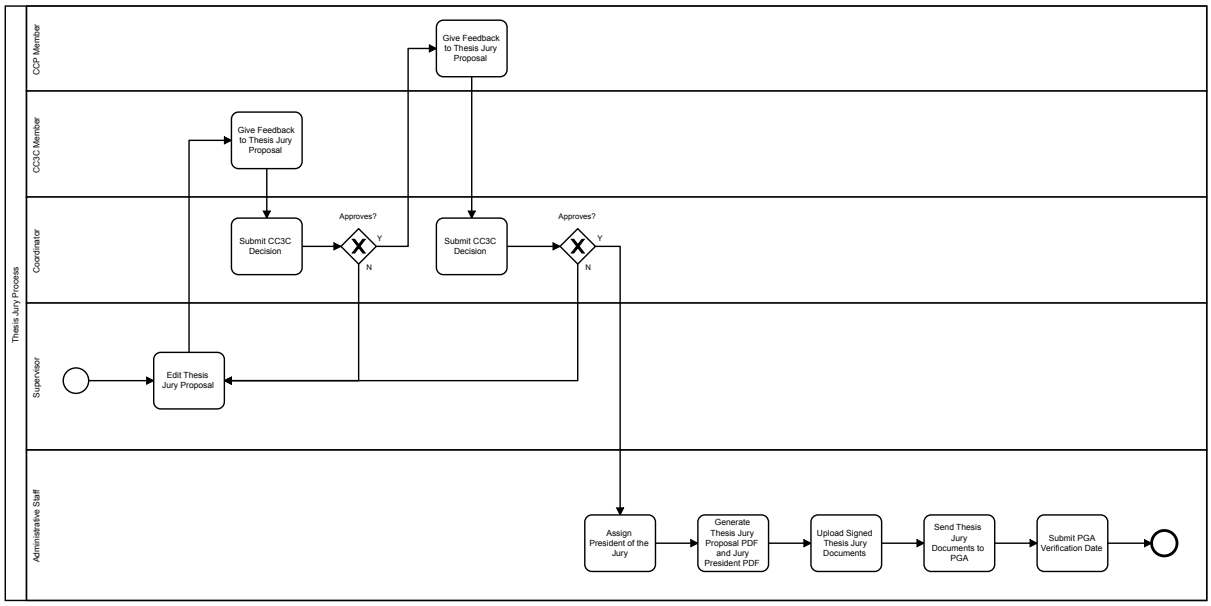


Figure 5.20: Thesis Jury Process.

When the jury proposal approval workflow is concluded, the administrative staff can submit the thesis defense date. After the defense, the thesis defense minutes are filled in by a PGA member. Therefore, there are no functionalities related to minutes submission on DD-IMS. The only information that is saved in the system is the student thesis grade, which can be inserted in DD-IMS by the administrative staff. If the student's thesis was approved, the student's status is changed from Active to Concluded in the system. If failed, the supervisor can schedule a new defense date. The Thesis Defense business process is represented in Figure 5.21.

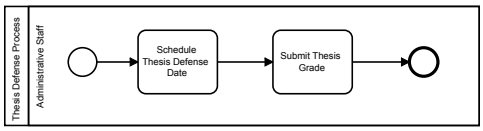


Figure 5.21: Thesis Grade Process.

The use case diagram related to the Thesis module is presented in appendix A. The class diagram and relational model of this module are presented in Appendix B.

5.9.1 User Interface

As in the Thesis Proposal module, the actors involved in the thesis jury evaluation process have access to the "Evaluation of Juries" tab. For the Thesis part, the main page contains a table with the thesis juries submitted. Each one of these juries can be visualized in more detail by clicking in the correspondent table row. Inside the thesis jury page, there are 3 left side menu options: (i) Jury, which contains the

jury members, specifying who is the president of the jury, who are the rapporteurs, and who are the supervisors; (ii) CC3C Feedback, where a table showing the feedback given by the CC3C members is displayed and the coordinator can approve or decline the jury after analyzing the feedback received; and (iii) CCP Feedback, where a table showing the feedback given by the CCP members is displayed and the coordinator can approve or decline the jury after analyzing the feedback received.

Students have access to a tab called ”Thesis”, where they can visualize information regarding their thesis. This information is divided into 3 left side menu options: (i) ”Thesis”, which shows the thesis title and abstract; (ii) Jury, where the jury members can be visualized; and (iii) ”Defense”, which displays the information regarding the thesis defense date and grade.

As in other modules, the administrative staff and the coordinator are also able to visualize the students' thesis information inside ”All Students” tab. The details about the thesis are shown on the student's process page, in a left side menu option called ”Thesis”, shown in Figure 5.22. This left side menu option contains 2 sub-options, ”Jury” and ”Defense”, which display information about the thesis jury and defense, respectively. After the student defends her thesis, the administrative staff is able to submit the thesis grade, by clicking on a button called ”Submit Grade” inside the ”Defense” sub-option.

Similarly, the students' supervisors have access to the same information on the ”Thesis” left side menu option and ”Jury” and ”Defense” sub-options of the student's process page, inside ”My Students” tab. On the ”Jury” sub-option, they can submit the thesis jury.

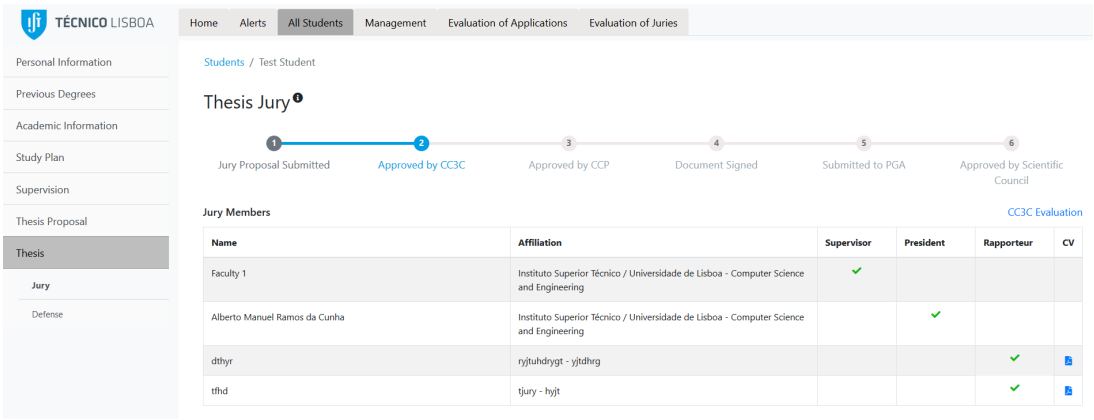


Figure 5.22: Screenshot of the Thesis left side menu option, inside the student page.

5.10 Statistics Module

The Statistics module is responsible for providing statistics about the Doctoral Program. The actors involved in this module are the coordinator and the CSE department administrative staff. These users can visualize indicators related to students, candidates, thesis, and thesis proposals. These indicators were collected during the requirements gathering phase, described in Chapter 3. The following indicators are provided by DD-IMS:

- **Number of students**. For this indicator, the following filters can be used: (i) Academic year; (ii)

Time Interval; (iii) Status (i.e. Active, Concluded, Suspended, or Abandoned); and (iv) Foreign (i.e. filter only foreign students or only Portuguese students)

- **Average time that graduated students took to finish their CSE Doctoral Program**. For this indicator, the following filters can be used: (i) Academic year; (ii) Time Interval; and (iii) Foreign
- **Number of candidates**. For this indicator, the following filters can be used: (i) Academic year; (ii) Time Interval; (iii) Academic semester (i.e. Winter or Summer); and (iv) Application status (e.g. Approved, Rejected, Approved Conditionally, etc.)
- **Number of thesis proposals defended**. For this indicator, the following filters can be used: (i) Academic year; (ii) Time Interval; (iii) Grade (i.e. Approved or Failed); and (iv) Foreign
- **Number of theses defended**. For this indicator, the following filters can be used: (i) Academic year; (ii) Time Interval; (iii) Grade (i.e. Approved, Approved with Distinction, Approved with Distinction and Praise, or Rejected); and (iv) Foreign

The use case diagram related to the Statistics module is presented in Appendix A.

5.10.1 User Interface

The statistics about the Doctoral Program are accessible by the coordinator and the administrative staff in the ”All Students” tab, more exactly in the ”Statistics” left side menu option. Inside this left side menu option, users can select 1 of the 4 sub-options that represent each type of statistics.

Regarding the display of the statistics, the solution found was to display the indicators below the group of filters available, which differ for each type of statistics, and above the list of students/candidates that correspond to the filters. An example of the UI designed for the statistics module is shown in Figure 5.23. In this example, the indicators refer to the students that have concluded their Doctoral Program during the academic year of 2020/21.

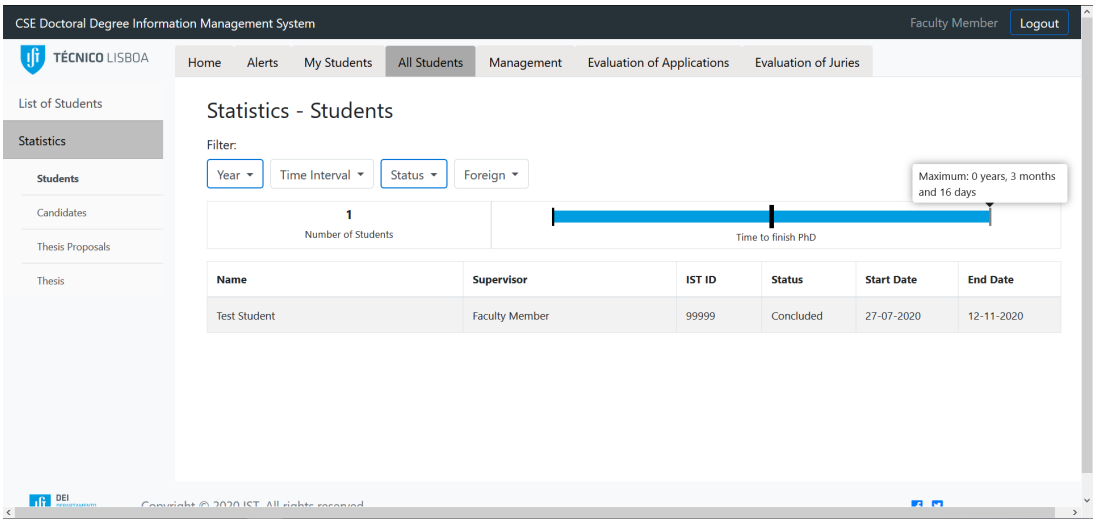


Figure 5.23: Screenshot of the UI designed for the Statistics module.

5.11 Doctoral Program Management Module

The Doctoral Program Management module is responsible for the CSE Doctoral Program management. The actors involved in this module are administrative staff and coordinator. These users can visualize and manage information regarding the CSE Doctoral Program faculty members, CC3C and CCP members, Scientific Committee members, coordinator, courses offered, and synchronization with Fenix.

Regarding faculty, the system allows the users to visualize the list of CSE Doctoral Program faculty members, insert more faculty members, edit faculty member's data, and remove a faculty member from DD-IMS. This can be carried out in 2 ways: (i) upload a CSV file (in a specific schema) with the list of CSE faculty members; or (ii) insert manually a faculty member by filling in the form asked. The CSE faculty members must be inserted into the system so that the students can select their IST CSE supervisors from a list.

The system also allows the users to visualize and manage the CSE Doctoral Program coordination, which includes coordinator, CC3C members, CCP members, Scientific Committee members (the members of this committee must be full professors), and administrative staff members. From the list of IST CSE faculty members, the users can select the coordinator (only the administrative staff has permission) and CC3C, CCP, Scientific Committee, and administrative staff members.

Regarding courses, the CSE Doctoral Program courses can be visualized and managed as well. The system allows the users to edit the information related to each course (scientific area, course type). The users can also load the current CSE Doctoral Degree curriculum (list of CSE Doctoral Degree courses) from Fenix.

Finally, the CSE Doctoral Program coordinator can synchronize the students' curricular plan with Fenix to obtain the information used in the Curricular Plan module.

The use case diagram related to the Doctoral Program Management module is presented in Appendix A.

5.11.1 User Interface

To have access to this module functionalities, the users involved in this module (administrative staff and coordinator) have access to a tab called "Management". On the main page of this tab, shown in Figure 5.24, the information displayed is divided into 4 left side menu options: (i) "Coordination" (the selected left side menu option on the screenshot presented in Figure 5.24); (ii) "Faculty"; (iii) "Courses"; and (iv) "Fenix Synchronization".

The "Coordination" left side menu option allows users to visualize tables with the Doctoral Program coordinator, CC3C and CCP members, Scientific Committee members, and the administrative staff members. The Scientific Committee members table also shows the number of juries presided by each of the full professors present in that list. On this menu option, users have access to a button called "Edit", which redirects them to a page that contains an editable view of the CSE Doctoral Program Coordination. This view allows users to change the coordinator, add new members, or remove current members from the CC3C, CCP, Scientific Committee, or administrative staff tables.

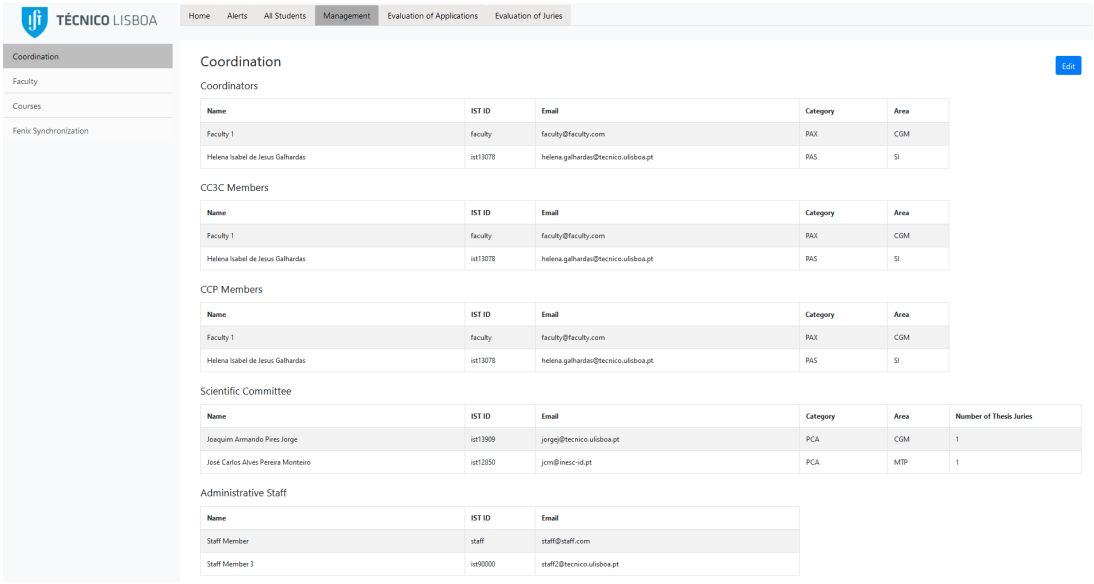


Figure 5.24: Screenshot of the main page of Management tab.

The ”Faculty” left side menu option displays a table with the CSE Department faculty members and a button called ”Insert Faculty”, which is used to insert a new faculty member manually or to insert several new faculty members from an uploaded file. Each row of the CSE Department faculty members table represents a faculty member and it is clickable so that users can navigate to the faculty page in DD-IMS. On this page, the faculty member information is shown and the administrative staff can edit this information or delete the faculty member from DD-IMS.

The ”Courses” left side menu option displays a table with the CSE Doctoral Program courses. On this menu option, 2 buttons are shown: (i) ”Load”, which functionality is loading the current CSE Doctoral Degree curriculum (list of CSE Doctoral Degree courses) from Fenix; and (ii) ”Edit”, which redirects users to a page that contains an editable view of the CSE Doctoral Program courses, where they can edit the courses type (i.e. Mandatory, Recommended, Optional) and scientific area (i.e. CGM, IA, SI, ASO, MTP).

The ”Fenix Synchronization” left side menu option displays the date of the last synchronization and presents to the coordinator a button called ”Sync”, which functionality is to synchronize with Fenix.

5.12 Fenix Connection

The DD-IMS connects with Fenix, taking advantage of the data that is already stored there. This communication is established through the FenixEdu API, provided by FenixEdu, which has two types of endpoints: public (can be accessed by everyone) and private (require user authentication).

To obtain the data already stored in Fenix, the DD-IMS communicates with the following endpoints of the Fenix API:

- **Students’ personal data (/person)**: Obtain the student’s personal data such as IST number, full name, gender, birth date, and email. This endpoint is private so only the student herself can

access this data.

- **Students' degrees (/person/curriculum)**: Obtain the student's current degree information (Doctoral Program start date). This endpoint is private so only the student herself can access this data.

- **Degree courses (/degrees/{id}/curriculum)**: Obtain the list of Doctoral Program courses and its related data (semester, ECTS). The academic semester must be provided to the request. This endpoint is public so everyone can access this data.

- **Students' enrolled courses (/phdThesisProcesses?username={IST-ID})**: Obtain the student's enrolled courses in a given semester and the corresponding grades (if already submitted in the Fenix system). This endpoint is private and only the Doctoral Program coordinator can access this data.

5.13 Implementation Details

The DD-IMS user interface was developed using the most known technologies for producing web content, supported by most of the browsers, namely HTML, CSS, and JavaScript. Regarding the web server, Django was the web framework used for implementing this system (as mentioned in Section 4.2). More information about the setup and structure of the DD-IMS software, as well as the instructions on how to add a new module to the DD-IMS, are presented in Appendix C.2.

Regarding the database component, having in mind that Django natively supports four of the most popular DBMS according to the DB-Engines ranking[7] (MySQL, PostgreSQL, Oracle, and SQLite) and all of them are relational DBMS, we opted to choose one of these four options instead of using a database that is not natively supported by Django. Oracle was excluded because it is not open source. SQLite is more limited than the others, especially in high-volume websites, as it does not support concurrency as well as MySQL and PostgreSQL. For these reasons, the Django documentation does not recommend to run and migrate SQLite in a production environment. So, there were two remaining systems: MySQL and PostgreSQL. We opted to use the PostgreSQL Relational DBMS for setting up and hosting the database as it is considered by the Django documentation as the most capable of all the databases here in terms of schema support while MySQL lacks support for transactions regarding schema modification operations, meaning that it is impossible to roll back to an earlier point if a migration fails (the changes must be unpicked manually). It also has reasonably small limits on name lengths for columns, tables, and indexes.

In terms of authentication, the IST CAS[8] is used to authenticate IST users in the DD-IMS. The IST CAS allows any user with a Técnico ID to authenticate in the system.

[7] https://db-engines.com/en/ranking
[8] https://id.tecnico.ulisboa.pt

5.13.1 Common Functionalities

The common functionalities are contained in a file called *utils.py*, that contains a set of helper functions and classes, which are described in table 5.2.

Table 5.2: Helper Functions and Classes

Helper Functions/Classes	Description
reverse_query_string	A function that builds a URL with a query string.
any_permission_required	A decorator which checks if a user has any of the specified permissions. It is useful for endpoints that require the user to belong to one of many permission groups (e.g. coordinator, administrative staff, CC3C and CCP members can visualize a candidate process). The decorator provided by Django library, permission_required cannot be used in this case as it takes only a single permission.
Date-related functions	Functions that receive dates and perform actions on those dates and are commonly used on different modules of the DD-IMS (i.e., compare two dates, extract the academic year or the academic semester from a given date, get all the academic years since a given start date until now).
Classes for generic forms	Forms that are commonly used on different modules of the DD-IMS: - Email form, used when a user edits an e-mail before sending it to PGA; - Date Range form, used when a user selects a start date and an end date for something; - Date form, used when a user selects a date without a specific time; - Datetime form, used when a user selects a date with a specific time; - Upload form, used when a user uploads a file.

Besides the functions stated in table 5.2, *utils.py* contains one more function, *render_template_as_pdf*, that is capable of generating PDF documents. The generation of PDF documents is one of the system requirements and is present in most of the DD-IMS modules. To generate the PDF documents, we need to create HTML templates with the information that should be displayed in the PDF document. The tools used in the conversion of these HTML templates to PDF format were: (i) *wkhtmltopdf*[9], an open source command line tool that renders HTML into PDF; and (ii) *pdfkit*[10], a python wrapper for the *wkhtmltopdf* utility.

To avoid the duplication of some HTML code, we also created a base HTML template and several objects representing HTML tables (contained in *html_table.py* file) and their descendants such as table headers or table rows, that can be parameterized to be reusable for all different types of tables.

5.13.2 Deployment

The DD-IMS web server is currently running on a virtual machine of the CSE/IST RNL[11] (in Portuguese, it is called *Rede das "Novas" Licenciaturas*), which provides IT Services to IST members and is responsible for the maintenance and management of the Computer Science laboratories. The DD-IMS is accessible at this URL: `https://phdisvm.rnl.tecnico.ulisboa.pt/`.

The deployment architecture is presented in figure 5.25. Besides Django, 3 other components are

[9] https://wkhtmltopdf.org/
[10] https://github.com/JazzCore/python-pdfkit
[11] https://rnl.tecnico.ulisboa.pt/

involved in this process: NGINX[12], Gunicorn[13], and Supervisor[14]. NGINX acts as a reverse proxy, a server that sits in front of web servers and forwards client requests (i.e. Web Browsers) to those web servers. It receives all client requests to the server and decides if the requested information is a static resource that it can serve by itself or if it is something more complicated, that must be passed to Gunicorn. The NGINX is configured with HTTPS certificates so that it only accepts requests via HTTPS. If an HTTP request is sent, NGINX first redirects the user to the HTTPS, and then handle the request. To generate these certificates, we used Let's Encrpyt[15], a certificate authority that provides certificates for Transport Layer Security (TLS), free of charge. The certificates generated are valid for 90 days and its renewal is automatic. Gunicorn is a Python Web Server Gateway Interface (WSGI) HTTP Server, responsible for executing the Django code behind a proxy server. Depending on the number of processors the server has, it can spawn multiple workers to process multiple requests in parallel. Finally, Supervisor is a process control system and it supervises Gunicorn and Django to make sure everything is running normally. For example, if Gunicorn crashes, Supervisor automatically restarts it.

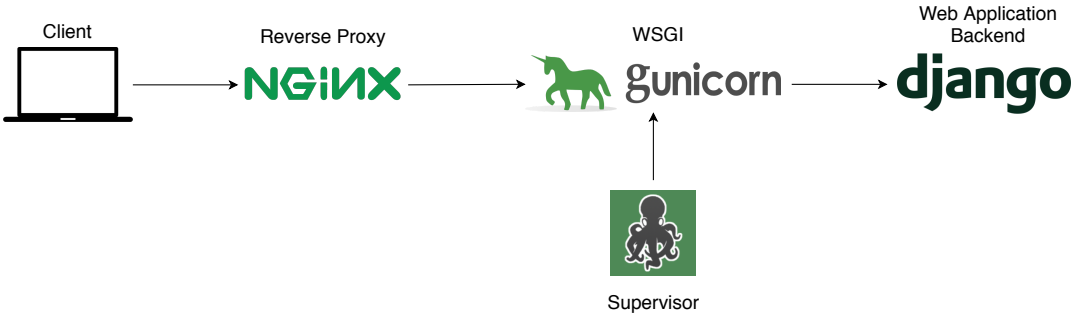


Figure 5.25: Deployment Architecture.

The DD-IMS database is deployed on the IST PostgreSQL database server[16]. The IST IT Services make available a PostgreSQL database server to all users registered in Fenix. The database is accessible via the Sigma cluster[17], a Unix Shell Service.

The instructions on how to deploy the DD-IMS web server are presented in Appendix C.3.

[12]https://nginx.org/
[13]https://gunicorn.org/
[14]http://supervisord.org/
[15]https://letsencrypt.org/
[16]https://suporte.dsi.tecnico.ulisboa.pt/manual-do-utilizador/base-de-dados-postgresql
[17]https://suporte.dsi.tecnico.ulisboa.pt/manual-do-utilizador/acesso-ao-cluster-sigma

Chapter 6

Experimental Validation

This chapter describes the experimental evaluation that we performed to validate the DD-IMS and the results that we obtained. The system was evaluated at three levels: functionality (Section 6.1), usability (Section 6.2), and performance (Section 6.3).

6.1 Functionality

Regarding *functionality*, the behavior of the system modules has to comply with the requirements gathered from the different stakeholders. To ensure that the functionalities of the modules implemented had the same behavior as expected in the use cases identified in the requirements document [11], unit tests were developed. The test suite covered the functionalities of the modules implemented, including errors and alternative scenarios.

For unit testing, we used the library provided by Django, *unittest*[1], which is a Python standard library module.

The test suite created contains 609 tests. During the test process, the following errors were found:

- For some of the alerts that are shown to users, the text displayed is not correct;
- The current coordinator should not be able to change the Doctoral Program coordinator. Only the administrative staff should be able to modify the Doctoral Program coordinator;
- When visualizing a student's study plan, the institution of the free courses chosen by the students is not displayed correctly;
- The system should not allow students to insert the same faculty member more than once in their supervision team;
- When inserting a new candidate's application to the Doctoral Program, the IST student number field should not be mandatory (the candidates that come from other universities do not have a student number when they submit their application).

[1] https://docs.python.org/3/library/unittest.html

All the errors stated above were already fixed and all the tests run without errors. The instructions on how to run the unit test and add new unit tests to the test suite are presented in Appendix C.4.

6.2 Usability

We performed usability tests with the future users of the system, namely: PhD students, supervisors, coordinator, members of the CSE administrative staff, members of the CC3C, and members of the CCP. The goal of these tests was to validate the DD-IMS graphical user interface, namely in what concerns the performance of the users when using the system as well as their satisfaction. This evaluation was divided into two steps: formative evaluation and summative evaluation detailed in Sections 6.2.1 and 6.2.2 respectively.

6.2.1 Formative Evaluation

This type of evaluation was performed iteratively during the implementation phase, to identify aspects that cause usability problems. After each iteration of this evaluation, the aspects to improve were listed and a new version of the graphical user interface was developed to improve the system's usability. This formative evaluation was performed in two iterations. The first iteration was focused on receiving feedback about the user interface of the modules that were implemented at the time, while the second iteration was focused on receiving feedback not only about the user interface but also about the functionalities of the modules.

In the first iteration of the formative evaluation, users gave feedback about the user interface of Registration, Study Plan, and Supervision modules. 7 students (already enrolled in the CSE Doctoral Program) registered in DD-IMS and defined their study plan and supervision, giving feedback about these modules. Moreover, the students' supervisors, as well as the CSE Doctoral Program coordinator, accessed the DD-IMS to approve the students' study plans and the supervisions. Regarding students, the feedback obtained is listed below:

- **FF-1**: The DD-IMS is unreadable on mobile devices. Some modifications should be done, such as shorten the text and change the headers.
- **FF-2**: Some students do not develop their PhD work in any research unit, so the research unit should not be mandatory when students insert their information.
- **FF-3**: When students register in DD-IMS, there should be a wizard progress bar, which allows students to understand how much is left to complete their Registration.
- **FF-4**: When defining a study plan, the system only accepts the submission if the sum of ECTS is equal to 30. In some cases, the courses selected may slightly exceed 30 ECTS.
- **FF-5**: Students should be able to review the courses selected before submitting the study plan.

- **FF-6**: When the students are choosing the free courses to include in their study plan, the DD-IMS should autocomplete the name of the courses, at least for the courses from Doctoral Programs that are more related to the CSE Doctoral Program (e.g., PhD Programme in Electrical and Computer Engineering).
- **FF-7**: In addition to the wizard progress bar, when defining the study plan, students should also visualize the current number of ECTS selected to have an idea if they need to add more courses or to remove any of the selected courses.
- **FF-8**: When visualizing the study plan, the courses should have a link to the course page on the IST website.
- **FF-9**: When selecting a supervisor from the CSE Department faculty combo box, there should exist a search bar, which allows students to select the supervisor faster.

Regarding faculty members, the feedback obtained is listed below:

- **FF-10**: Display the students' progress in a timeline or a dashboard. Have a color code for the student status. With this, it will be possible to distinguish students in different phases of their Doctoral Program (e.g., distinguish students with thesis proposal defended from students that did not complete the academic part yet).
- **FF-11**: The approval workflow displayed when visualizing students' supervision and study plan is not clear enough. Besides that, it should be displayed on top of the page instead of being on the right to reduce the space used, which will be useful on screens with lower resolution.
- **FF-12**: Support study plans in the context of dual degrees (e.g. CMU Portugal) and study plans defined when old curricular plans were used.
- **FF-13**: A supervisor should not be able to approve a student's study plan before approving her supervision.
- **FF-14**: The left sidebar that contains the menu options should be improved because it is very sparse and redundant with the tabs on top of the page

The feedback items **FF-2**, **FF-3**, **FF-4**, **FF-5**, **FF-6**, **FF-7**, **FF-8**, **FF-9**, **FF-11**, **FF-12**, **FF-13**, and **FF-14** were integrated in DD-IMS. Regarding **FF-1**, some modifications were made but more improvements need to be done for the DD-IMS to be fully legible on mobile devices. The feedback item **FF-10** was not integrated yet as it will require some time to implement, which consequently will delay the implementation of the other modules.

In the second iteration of the formative evaluation, the modules developed so far were tested in order to: (i) find bugs, ii) find improvements to be applied to the UI, and (iii) confirm the functionalities implemented. The following modules were tested: Evaluation of Applications, Registration, Supervision, Study Plan, Thesis Proposal, Thesis, and Management.

Regarding the Evaluation of Applications, the module functionalities were tested by the CSE Doctoral Program coordinator, 2 members of the CSE administrative staff, and 8 CC3C members. The 15 applications for the CSE Doctoral Program in the first semester of 2019/20 were inserted in DD-IMS. In total, the administrative staff inserted 15 applications in the system.

From that set of applications, the 12 candidates whose application was approved were asked to register in DD-IMS, define their study plan and supervision. The supervisors and the coordinator approved the students' supervision and study plan. In addition to that, the following steps of the approval workflow were also performed in DD-IMS by the administrative staff: (i) the study plan/supervision documents were generated and a signed version of the documents was uploaded; (ii) the documents were sent to PGA (this action was simulated because, at that time, the Simple Mail Transfer Protocol (SMTP) server that will be used by DD-IMS to send mails automatically was not defined); and (iii) the Scientific Council approval date was submitted to the system.

For testing the Thesis Proposal and Thesis modules, 1 student was selected for each of these modules. The students selected had submitted the thesis proposal jury and thesis jury, respectively, at the time of the second phase of the formative evaluation.

6.2.2 Summative Evaluation

This type of evaluation was performed after the conclusion of the DD-IMS implementation, to evaluate the success of the final product. We evaluated the system quantitatively and qualitatively.

Quantitative Measures

The quantitative evaluation focused on how the users performed a set of tasks assigned to them. As the system has multiple user groups, six types of tasks were defined: Student tasks, Supervisor tasks, Staff tasks, Coordinator tasks, CC3C member tasks, and CCP member tasks. User testing was divided into two phases: (i) introductory phase; and (ii) tasks execution phase. During the introduction phase, the users were introduced to the system. They could also explore the system and ask questions on how to use it. During the tasks execution phase, the users were asked to execute the set of tasks assigned to them. For the execution of these tasks, the users used dummy data, which was loaded into the system, to ensure that users were working with the same data. This set of tasks was accessible by the users via a guide (in PDF format) that was previously given to them. This guide also contained an introduction to the DD-IMS and a brief explanation about how the evaluation was performed. The following measures were taken during the execution of the tasks:

- **"Did the user complete her task?"** - indicates if the user was able to conclude the task;
- **"How much time was taken to conclude the task?"** - corresponds to the time, in seconds, that the user took to finish the task;
- **"How many errors were made by the user?"** - corresponds to the number of mistakes made during the task execution. We considered that a user makes a mistake when she diverts from the

task execution path or when she uses improperly an element of the graphical user interface (e.g., inserting a date in incorrect format).

Qualitative Measures

After finishing the execution of tasks, users gave their feedback about the DD-IMS, by answering a questionnaire[2]. The content of the questionnaire is presented in Appendix E. This questionnaire was used to qualitatively measure the system usability and is divided into three parts:

1. **Personal Information, Type of Tasks Executed and Planned Utilization:** In this section, users were asked about their personal information, what type of tasks did they execute, and which modules of the DD-IMS they expect to use more often. The users that executed the Student tasks were also asked about the curricular plan in which they defined their study plan, which allowed us to understand if there were any problems or limitations regarding the DD-IMS support for older curriculums in the study plan creation.
2. **System Usability Scale (SUS):** Based on the SUS[3] questionnaire, it consists of a 10 item questionnaire with five response options for users (from Strongly agree to Strongly disagree). With the users' answers, it is possible to calculate the SUS score, a number, between 0 and 100, that represents a measure of the system usability. To calculate this number, each item score must be multiplied by 2.5 to convert the original scores from 0-40 to 0-100.
3. **Additional feedback:** Besides the 10 items answered by the users, users were free to include further comments related to the system.

We also used the *Think-Aloud* [4] method during the execution of the tasks. This testing method consists of giving users the opportunity of expressing their feelings about the system user interface while interacting with it. Unlike the tasks used for obtaining quantitative measures when users were asked to follow strictly the script, in this phase, the users had the freedom to decide which actions they want to perform (e.g. Students were asked to submit their own personal and academic information, unlike in one of the previous tasks where they were asked to submit the data specified in the script).

The questionnaire and the *Think-Aloud* method, along with the quantitative metrics used, had a key role in the DD-IMS evaluation, indicating the users level of satisfaction and the difficulties that they have found during the execution of the tasks.

Planning

The final evaluation of the system usability with a larger sample of users was preceded by pilot-tests, which consisted of the execution of tasks to validate the evaluation procedure to be followed in the final evaluation. Six users participated in these pilot-tests, one for each type of tasks, to ensure that all types of tasks assigned to each user group were validated.

[2]Available at: `https://tinyurl.com/DD-IMS-Evaluation`
[3]https://www.usability.gov/how-to-and-tools/methods/system-usability-scale.html
[4]https://www.nngroup.com/articles/thinking-aloud-the-1-usability-tool/

For the final evaluation, we selected a set of users that did not have any experience with DD-IMS before the evaluation but, at the same time, have familiarity with the context and vocabulary of the tasks. A total of 29 users participated in the DD-IMS usability evaluation, having the following number of users for each type of tasks:

- 10 students;
- 10 supervisors;
- 2 administrative staff members;
- 3 previous CSE Doctoral Degree coordinators;
- 2 previous CSE CC3C members;
- 2 previous CSE CCP members.

The DD-IMS evaluation sessions were performed via Zoom. Each session was expected to have a duration of approximately 40 minutes. Before starting each session, users were asked to access the DD-IMS website (https://phdisvm.rnl.tecnico.ulisboa.pt/) and log in with their Fenix account. In the first 10 minutes, users were introduced to DD-IMS and were free to interact or make questions about it. Then, the execution of the tasks lasted about 20 minutes. During the evaluation, users indicated when they were ready to start a task as well as when they consider having finished a task. Moreover, users were allowed to ask questions after the end of a task and before the start of the next task. Before ending the session, users had the last 10 minutes to answer the questionnaire. For the session, a laptop and a network connection were required.

The list of tasks assigned to the administrative staff members is presented below. The tasks assigned to the remaining user groups are available in Appendix D.

1) **Evaluation of Applications**

   a) For candidate 1, find the previous degree grade.

   b) Edit the Candidate 1 application with the following changes: (i) Previous Degree Grade: 18; (ii) Doctoral Program Type: CMU Portugal.

   c) Edit the candidate 1 supervision team: Add ”Faculty 1” as a Faculty Co-Supervisor without acceptance letter and with the following justification ”This is a test”.

   d) Send the application results letter with reference ”Test Letter 1” to Post-Graduate Area (PGA).

2) **Students Information**

   a) Find the student Y PhD start date and student number.

   b) Change the student Y status from ”Active” to ”Suspended”.

3) **Study Plan**

   a) For student X, generate study plan document.

4) **Supervision**

   a) For the student X supervision, submit the Scientific Council approval date: ”16/11/2020”.

5) **Statistics**

   a) Find the number of active students during the academic year of 2018/19.

6) **Management**

   a) Find the IST ID of the CSE Doctoral Degree coordinator.

   b) Insert a new faculty member with the following information:

      - **Full Name**: Test Faculty
      - **Email**: facultytest@tecnico.ulisboa.pt
      - **IST Number**: ist00000
      - **Area**: CGM
      - **Category**: PCA

   c) Find the number of ECTS and course type of the course ” Interdisciplinary Research”.

   d) Change the professional category of faculty Y to ”PAS”.

7) **Thesis Proposal**

   a) For student A thesis proposal jury, find the CCP approval date.

   b) For student A, generate the thesis proposal jury proposal PDF.

8) **Thesis**

   a) For student B, send the thesis jury proposal to PGA.

6.2.3 Results

This section presents the results of the DD-IMS summative evaluation. It is divided into two parts: the first one analyzes the results of the quantitative evaluation for each type of task; the second one analyzes the results of the qualitative evaluation.

Quantitative evaluation

Figures 6.1 and 6.2 present the results obtained for the Student tasks, which are described in Appendix D.

In Figure 6.1, we can observe the number of users that concluded each of the tasks and the total number of errors committed. In addition, the following conclusions can be taken:

- In Task 1-a), some students forgot to insert the grades of previous degrees and conclusion year on the first try. Another error occurred when the phone number was inserted in an invalid format. Some of the students did not feel comfortable with the need of inserting the country code (e.g. +351) in the phone number field.

- In Task 1-b), some of the students found it difficult to discover in which tab their personal information and academic information was available. One of the students did not even conclude this task, in which students were asked to find their PhD start date. The name of the tab (”Registration”) was the main cause of those difficulties. To address this problem, we decided that the tab name is ”Registration” until the student registers in DD-IMS. From that moment onwards, the tab name will change to ”My Information”.
- In Task 2-a), one of the students did not conclude the task because he did not submit the study plan by clicking on the ”Submit”, displayed in the last step of the Study Plan form wizard.
- In Task 3-a), some of the students did not understand well the task wording and tried to insert at first the non-faculty co-supervisors as faculty co-supervisors.
- The users were able to perform all the other tasks without any difficulty.

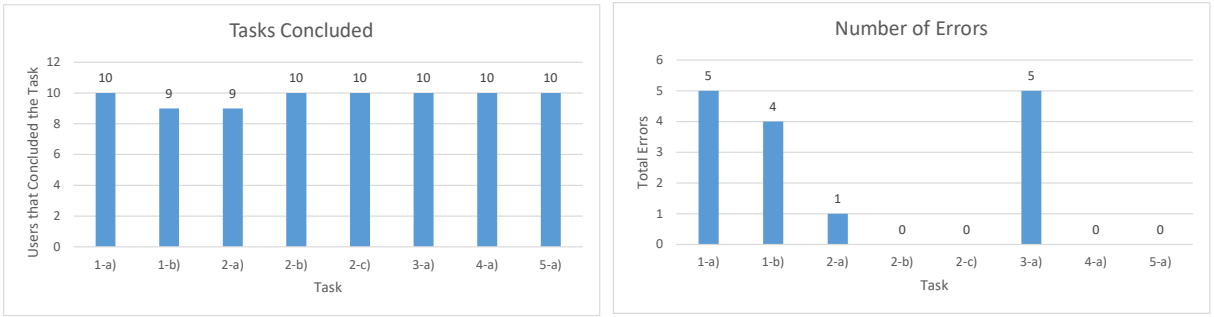


Figure 6.1: a) Number of users that concluded each of the Student tasks; b) Total number of errors committed in each of the Student tasks.

Figure 6.2 presents the average execution time for each task. As expected, tasks 1-a), 2-a), and 3-a), were the ones that took more time to complete. In these tasks, students had to insert the personal and academic information stated in the students' script, submit the study plan described in the script, and submit the supervision team listed in the script, respectively, so it was expected that they would need more time to insert the data. The time variation is also higher for these 3 tasks. One of the main reasons for this variation is the fact that the users had different typing speeds. Also, some of the users did not have a second screen available or the printed guide to visualize the guide while executing the tasks in the web browser.

Figures 6.3 and 6.4 present the results obtained for the Supervisor tasks, which are described in Appendix D.

In Figure 6.3, we can observe the number of users that concluded each of the tasks and the total number of errors committed. All the users have successfully completed all the tasks. Regarding the number of errors made, the following conclusions can be taken:

- In Task 4-a), some of the supervisors tried to submit the thesis proposal jury without specifying the president of the jury. After reading again the description of the task, they found out who was the president and concluded the task with success.

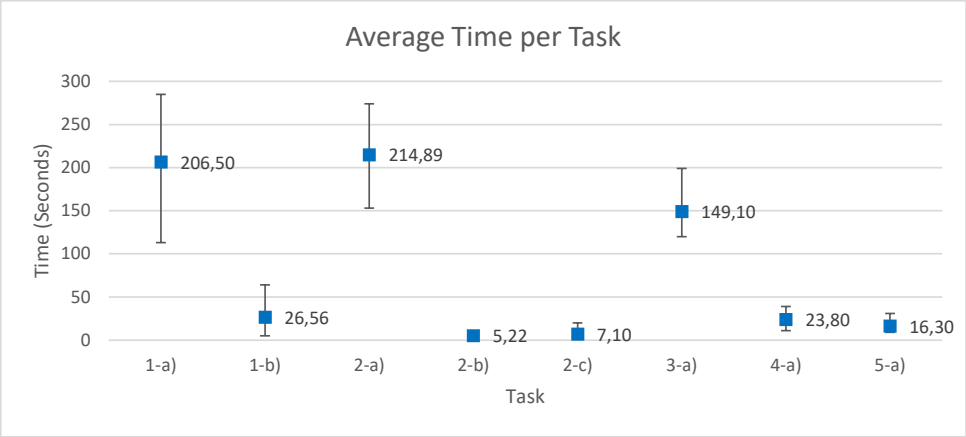


Figure 6.2: Average, minimum and maximum execution times for each of the Student tasks.

- In Task 4-b), some of the supervisors did not find the place to submit the thesis proposal defense minutes on the first attempt. One of the supervisors first browsed the page that allows to edit the thesis proposal (the title and abstract could be modified) instead of moving to the page where the thesis proposal minutes could be submitted.

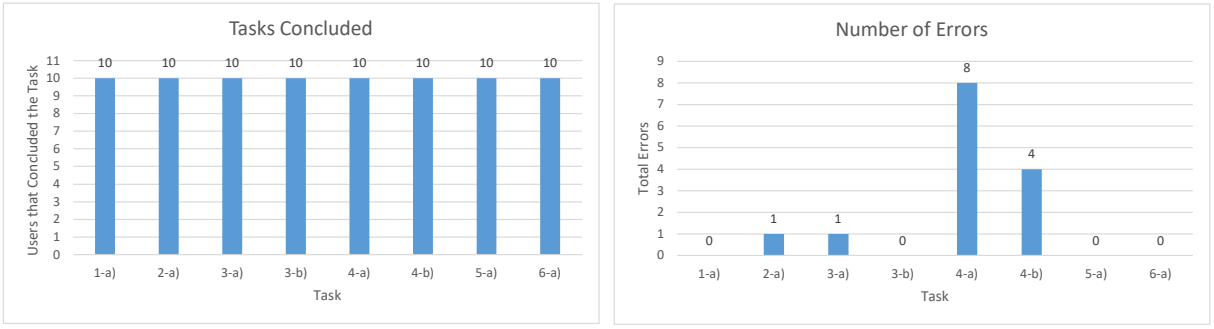


Figure 6.3: a) Number of users that concluded each of the Supervisor tasks; b) Total number of errors committed in each of the Supervisor tasks.

Figure 6.4 presents the average execution time for each task. As expected, tasks 4-a) and 4-b) were the ones that took more time to complete. In these tasks, supervisors had to insert data regarding the thesis proposal jury and defense, respectively, so it was expected that they would need more time to insert the data. The time variation is also higher for these 2 tasks. The main reason for this variation is the fact that some users faced some difficulties during the tasks execution while others concluded the tasks without making any errors.

Figures 6.5 and 6.6 present the results obtained for the Staff tasks, which are described in Appendix D.

In Figure 6.5, we can observe the number of users that concluded each of the tasks and the total number of errors committed. The following conclusions can be taken:

- In Task 1-c), one of the users faced some difficulties in discovering how to edit the candidate's supervision team.

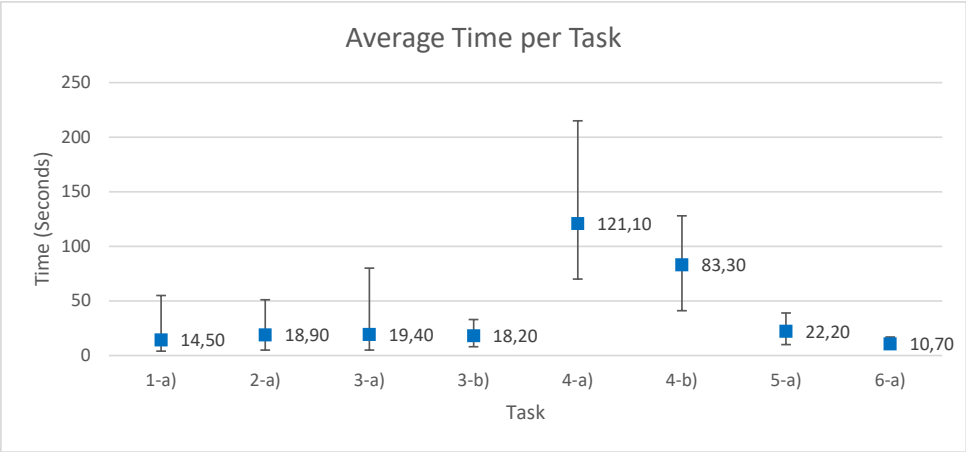


Figure 6.4: Average, minimum and maximum execution times for each of the Supervisor tasks.

- In Task 5-a), one of the users made some errors while using the statistics filters (i.e. Filtered by foreign students but that filter was not mentioned in the task description).
- In Task 6-d), in which the users must edit the information of a faculty member, one of the users did not conclude the task because she did not find out how to reach the page where that information can be edited.

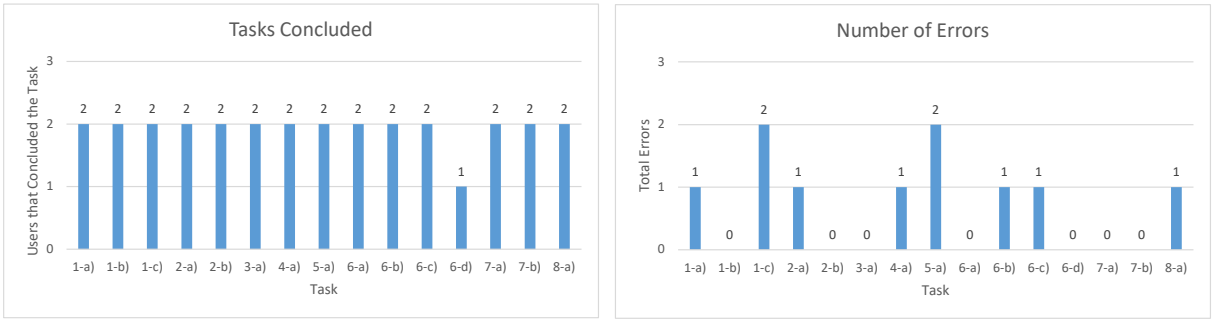


Figure 6.5: a) Number of users that concluded each of the Staff tasks; b) Total number of errors committed in each of the Staff tasks.

Figures 6.7 and 6.8 present the results obtained for the Coordinator tasks, which are described in Appendix D.

In Figure 6.7, we can observe the number of users that concluded each of the tasks and the total number of errors committed. The following conclusions can be taken:

- In Task 4-a), the users faced some difficulties when accessing the statistics. They found it difficult to understand that the ”Time Interval” filter was an alternative to the ”Year” and that it was not required to use this filter when obtaining the number of candidates admitted on the winter semester of 2020/21 (placing the value ”2020/21” on the ”Year” filter was enough). Also, one of the users found difficulties when trying to navigate from the students statistics to the candidates statistics.
- In Task 7-a), in which the users must browse the thesis defense jury information to submit the decision of the CCP evaluation (in ”Evaluation of Juries tab”), the users first did not switch from the

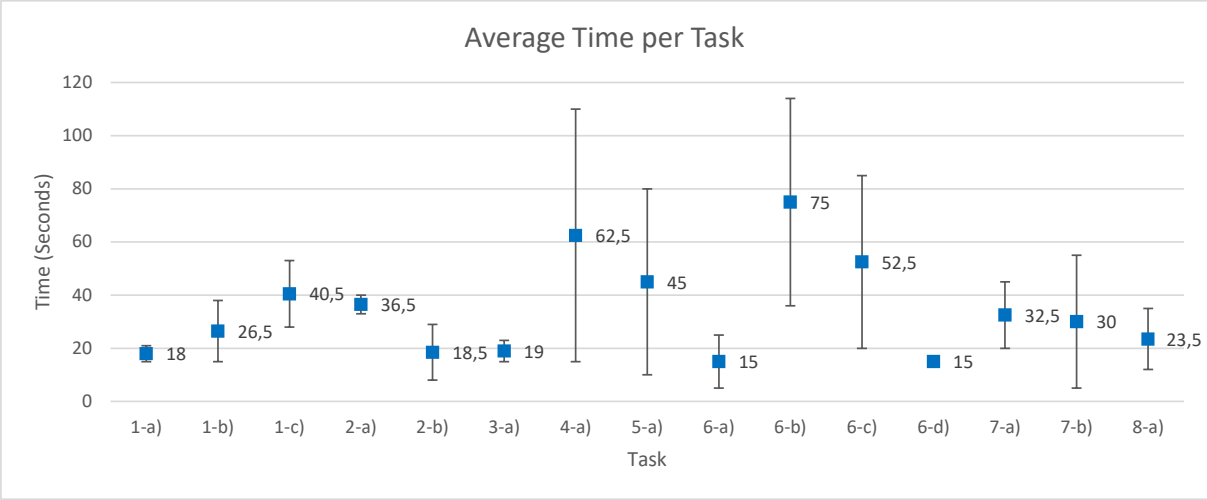


Figure 6.6: Average, minimum and maximum execution times for each of the Staff tasks.

”Thesis Proposal” left side menu option to the ”Thesis” left side menu option.

- In Task 8-a), one of the users first misunderstood the task description and inserted a wrong evaluation result for the application (selected the ”Approved” option instead of selecting ”Approved with Propaedeutic Courses”). Another error made by users occurred when they first tried to submit the application result by editing the candidate's information. After discovering the error, the users browsed the CC3C Evaluation left side menu option of the application page and then submitted the evaluation result.
- In Task 8-b), one of the users forgot to insert the reference of the application results letter. Another user found some difficulties to find the button that allows users to create the application results letter.

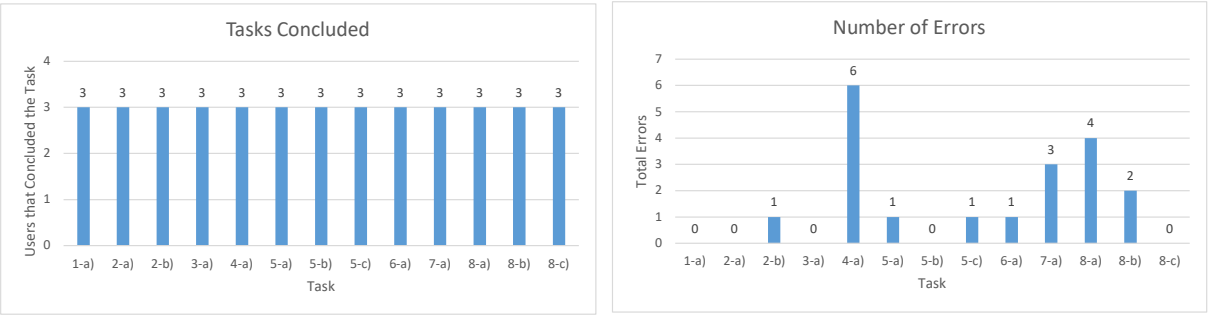


Figure 6.7: a) Number of users that concluded each of the Coordinator tasks; b) Total number of errors committed in each of the Coordinator tasks.

Figure 6.9 presents the results obtained for the CC3C members tasks, which are described in Appendix D. All users have successfully concluded their assigned tasks and did not make any kind of errors. Regarding the task execution times, the values were all between 10 and 40 seconds which means that the users did not find any major difficulties while executing the tasks.

Figure 6.10 presents the results obtained for the CCP members tasks, which are described in Appendix D. All users have successfully concluded their assigned tasks. In Task 2-a), in which the users

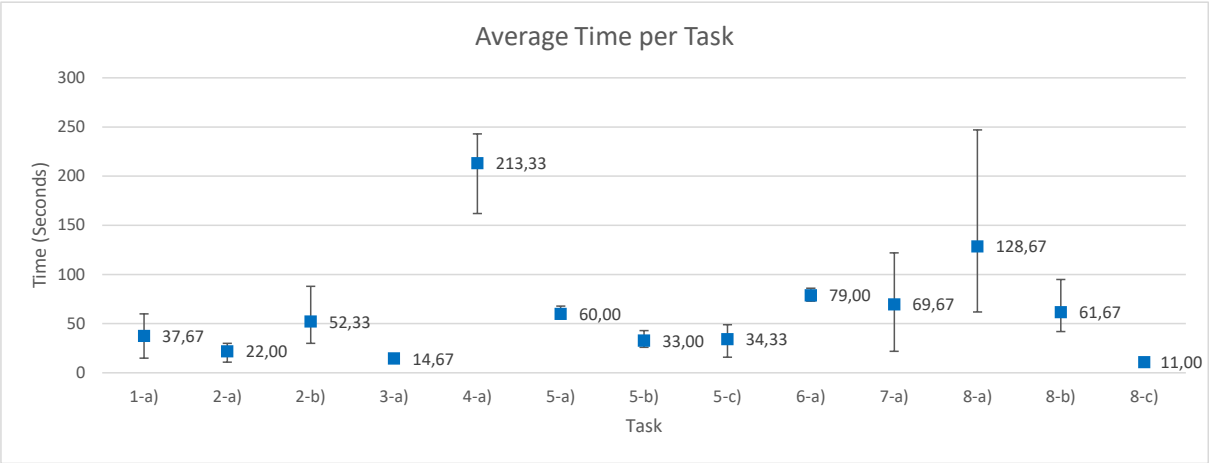


Figure 6.8: Average, minimum and maximum execution times for each of the Coordinator tasks.

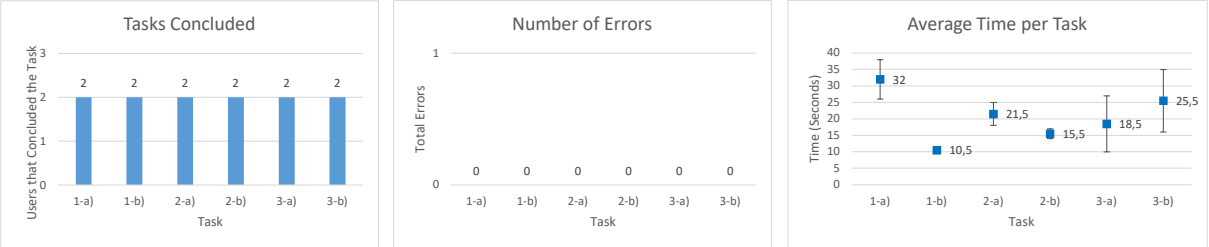


Figure 6.9: a) Number of users that concluded each of the CC3C members tasks; b) Total number of errors committed in each of the CC3C members tasks; c) Average, minimum and maximum execution times for each of the CC3C members tasks.

must browse the thesis defense jury information to find the rapporteurs (in ”Evaluation of Juries tab”), one of the users first did not switch from the ”Thesis Proposal” left side menu option to the ”Thesis” left side menu option. Regarding the task execution times, task 2-a) had the highest range of values (highest difference between the minimum and maximum) due to the error made by one of the users. Besides that, the values show that the users did not find other major difficulties while executing the tasks.

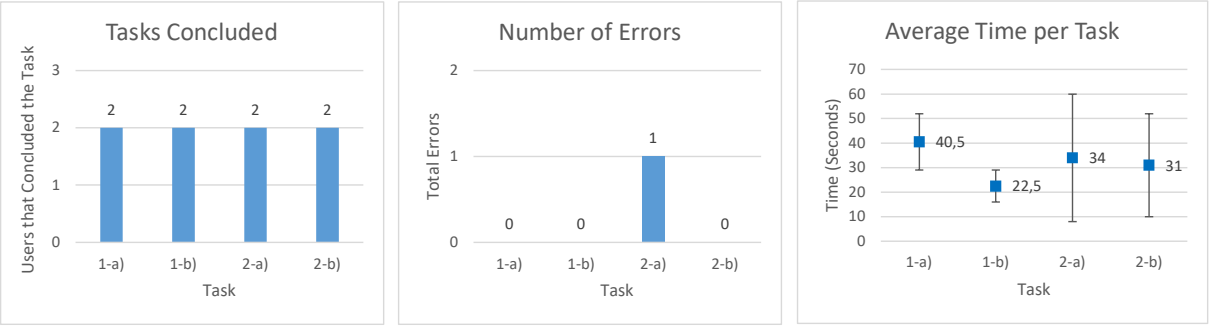


Figure 6.10: a) Number of users that concluded each of the CCP members tasks; b) Total number of errors committed in each of the CCP members tasks; c) Average, minimum and maximum execution times for each of the CCP members tasks.

Qualitative evaluation

Figures 6.11, 6.12, and 6.13 present the results of the questionnaire answered by the users at the end of the evaluation sessions.

In Figure 6.11, the age and gender distribution are presented. In which regards the age distribution, the chart 6.11.a) shows a large variation in the ages of the users that participated in the evaluation sessions. On the opposite way, the gender distribution chart shows a huge imbalance between the number of male and female users. This is due to the fact that the CSE department population (concerning students and faculty members) mainly belong to the male gender.

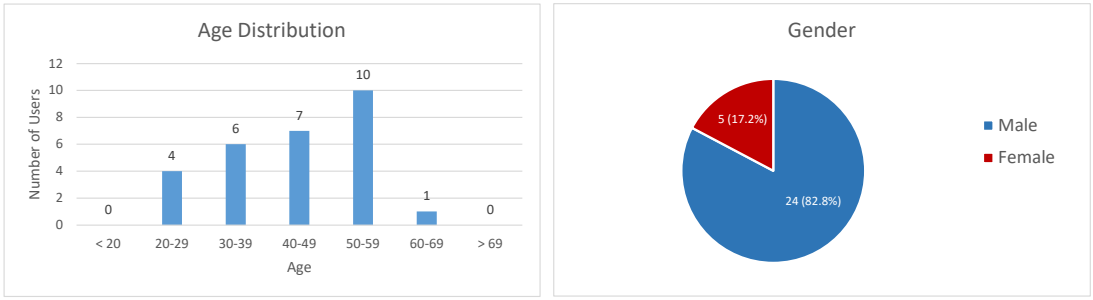


Figure 6.11: a) Users age distribution; b) Users gender distribution.

Figure 6.12 shows a prediction of the modules that will be used more often by DD-IMS users. The modules that are expected to be more used are Study Plan, Supervision, Curricular Plan, Thesis Proposal, and Thesis. The Evaluation of Applications, Statistics, and Management modules are not accessible to all types of users, which resulted in a lower frequency for these modules. However, the administrative staff members indicated that the Management module will be one of the most used by them. Regarding the previous coordinators, they expect that the Statistics module will be one of the most used by a coordinator.

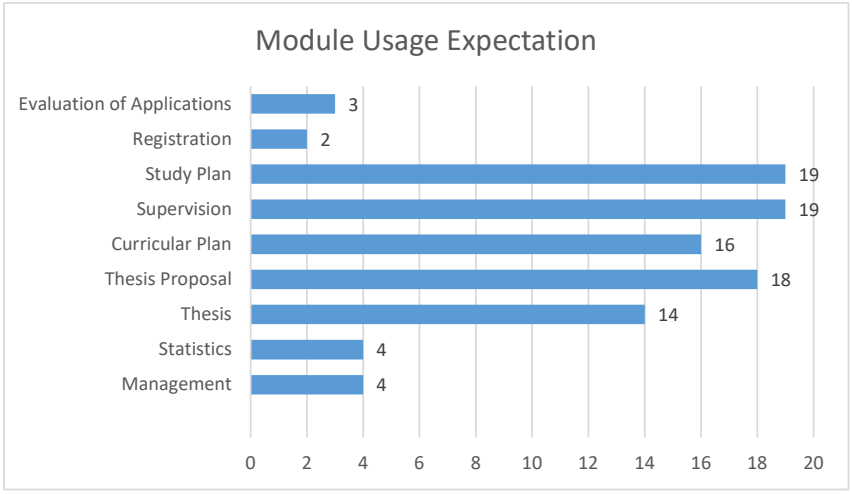


Figure 6.12: Prediction of the modules that will be used more often by users.

Based on the questionnaire answers given by the 29 users that participated in the DD-IMS usability evaluation, we converted each user's answers in a score from 0 to 100 (SUS score). After analyzing this

data, we obtained an average SUS score of 88.19, with a standard deviation of 10.54 and a median of 90. This is interpreted as a very good evaluation result, which means that users are quite satisfied with the DD-IMS usability. Analyzing the SUS scale percentile ranking[5], which tells how well a system SUS score compares to other systems, we can conclude that the DD-IMS is well above the average (68), receiving an 'A' grade, and its SUS score (88.19) is close to the 100th percentile. Figure 6.13 presents the distribution of the SUS scores given by the users.

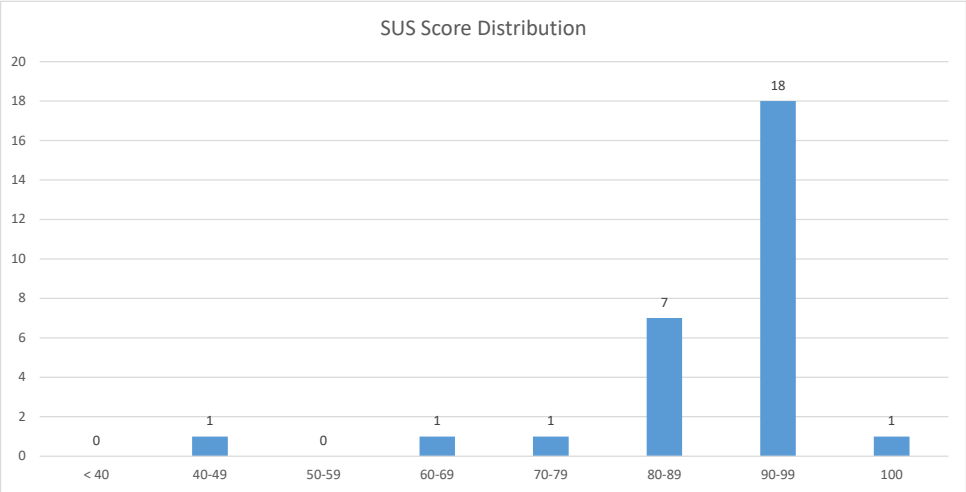


Figure 6.13: SUS scores distribution.

During the *Think-Aloud* phase of the evaluation sessions, users also suggested some modifications to be implemented on the DD-IMS:

1. **Students suggestions**:

   (a) Allow students to indicate their phone number without the country code. To support international phone numbers, the DD-IMS should display these two elements separately. To select the country code, users could choose the country from a list of flags.

   (b) When students finish their registration in the DD-IMS, change the name of the "Registration" to "My Information".

   (c) When fulfilling the supervision team information, the fields for entering the Non-Faculty Supervisors data are too small.

   (d) Allow students to insert only one previous degree (some students only have a 5-year bachelor's degree or an integrated master's programme).

   (e) For students that have multiple e-mails associated with their Fenix account, allow them to choose which of the e-mails they want to place in their personal information.

   (f) When students select the optional courses they want to include in their study plan, add a search icon to the Course Name column so that students can find the courses more easily.

[5]https://measuringu.com/interpret-sus-score/

(g) The select box with search (uses the JQuery plugin Select2[6]) should be improved (e.g. When a student types their supervisor's first and last names without typing the middle names, the supervisor disappears from the list of options of the select box).

2. **Supervisors suggestions**:

   (a) When submitting a jury proposal, the button that submits the jury should be named ”Submit” instead of ”Save”.

   (b) Add a ”Cancel” button which allows users to undo the modifications done in a jury proposal.

   (c) When declining a student's study plan or supervision, the DD-IMS should provide to the users a set of default justifications so that the users can select one of them if appropriate.

3. **Administrative Staff members suggestions**:

   (a) Order the alerts by date.

   (b) The DD-IMS should inform the user if other users are browsing on the same page as the user.

   (c) Place the statistics in a new tab.

   (d) When visualizing a student's process, display the deadlines for the thesis proposal and thesis submissions (if not submitted yet).

   (e) Display an alert to the administrative staff when students are close to their deadlines for the thesis proposal or thesis submission.

4. **Coordinators suggestions**:

   (a) When submitting the evaluation of an application, change the button name from ”Submit Result” to ”Submit Decision”.

   (b) Change the name of some of the tabs (i.e. change ”Evaluation of Juries” to ”Review of Jury Proposals” and ”Evaluation of Applications” to ”Candidates”)

5. **CC3C members suggestions**

   (a) Display the alerts due date for the alerts that have a deadline.

   (b) When a CC3C member submits her feedback about a jury proposal or an application, the success message, which informs that the feedback was submitted with success, should appear near the text submitted and not at the top of the screen.

6. **CCP members suggestions**

   (a) Display the alerts due date for the alerts that have a deadline.

[6] https://select2.org/

6.3 Performance

We performed load tests to analyze the system response when used by a large number of users simultaneously.

The tool used for performance testing was JMeter[7]. It is an open source and platform-independent application, written in Java. It was designed to simulate real-user behaviors (using the system functionalities) and measure the performance of web applications. JMeter can be used to simulate a heavy load on a server or group of servers to test its strength or to analyze its overall performance under different load types.

6.3.1 Measures

The system performance was evaluated by the following three measures: (i) **error rate** measures (in percentage) how frequently are internal server errors (or HTTP 5xx codes[8]) returned to clients; (ii) **response time** measures (in ms) how much time, on average, do clients wait for system response since they send their request; and (iii) **throughput** measures (in requests per second) how many requests the system can handle simultaneously.

6.3.2 Planning

The load tests were executed on a virtual machine of the CSE/IST RNL, similar to the one in which the DD-IMS is running in production, with the following specifications:

- Operating System: Debian stable
- RAM: 1 GB
- Processor: Intel Core (4 MB Cache, 2.4 GHz)
- Number of CPU Cores: 1
- Disk Space: 10GB

The load testing focused mostly on student actions. We made this decision because in general, the actions performed by the students are the heaviest ones, as they require more data processing (e.g., register in the system by inserting their personal and academic data, submit study plan, etc.). Furthermore, it is expectable that the number of students using the system simultaneously is higher than the number of faculty members or administrative staff members doing that. For example, when the students are notified about the results of the applications to a Doctoral Program, it is expectable that most students will perform a significant number of actions regarding their personal and academic information, as well as their supervision and study plan.

[7] https://jmeter.apache.org/
[8] https://developer.mozilla.org/pt-PT/docs/Web/HTTP/Status

Before running the tests, we prepared the system by inserting 400 virtual users with the Student role. The 20 courses from the current curriculum of the CSE Doctoral Program and the 83 faculty members of the CSE department were also loaded into the system. As none of the virtual users had accounts in Fenix, they could not authenticate with Fenix. Each one of them had a unique username and password so that they could log in the system via an admin page.

The load tests were divided into 9 scenarios, which made it possible to evaluate the system behavior variation from two different perspectives:

1. Testing the system behavior with a **variable number of users** performing a set of actions (50 actions were performed by 25, 50, 100, 200, 400, and 800 users)

2. Testing the system behavior with a **variable number of actions** performed by each user (50 users performed 25, 50, 100, 200, 400, and 800 actions)

To simulate users behavior more realistically, we added a delay (Think Time) of 2 seconds between each user's action. The users executed the following actions: (i) Login; (ii) Visualize Alerts; (iii) Visualize Personal and Academic Information; (iv) Edit Personal and Academic Information; (v) Visualize Study Plan; (vi) Submit Study Plan; (vii) Visualize Supervision Team; (viii) Submit Supervision Team; (ix) Visualize list of CSE PhD Courses; (x) Visualize list of CSE Faculty Members; and (xi) Logout. Except for the Login and Logout actions, which were executed only once at the beginning and at the end of a scenario respectively, each of the remaining actions was executed a random number of times and in random order by each user.

6.3.3 Results

This section presents the results obtained with the evaluation of the DD-IMS performance. For each one of the three measures analyzed, we present two graphics: one with a variable number of actions and the other with a variable number of users.

Error Rate

The graphics shown in Figures 6.14 and 6.15 present the results obtained regarding the error rate measure.

In Figure 6.14, the error rate maintains a constant percentage of 0 percent. In Figure 6.15, the error rate remains equal or very close to zero, until the number of users is 100. Starting from there, there is linear growth which means that the system has a sustained increase in the error rate. With 800 users, the error rate reaches 4 percent.

Average Response Time

The graphics shown in Figures 6.16 and 6.17 present the results obtained regarding the average response time measure.

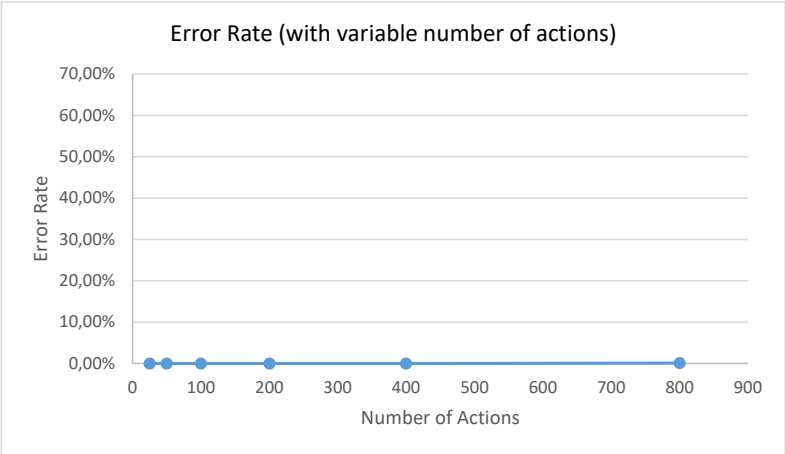


Figure 6.14: Error rate with variable number of actions.

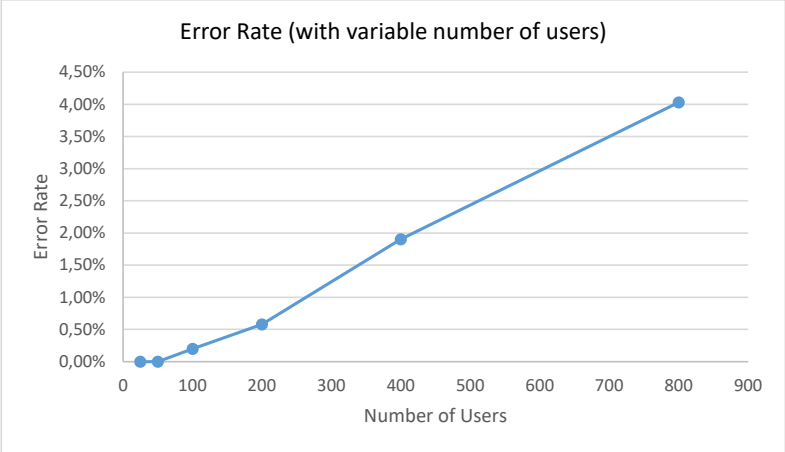


Figure 6.15: Error rate with variable number of users.

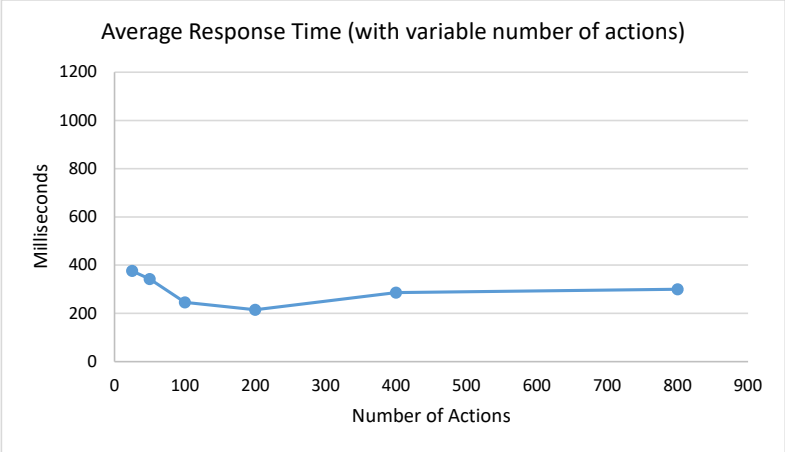


Figure 6.16: Average response time with variable number of actions.

In Figure 6.16, there is little variation over the average response time. In Figure 6.17, the average response time grows linearly with the number of concurrent users.

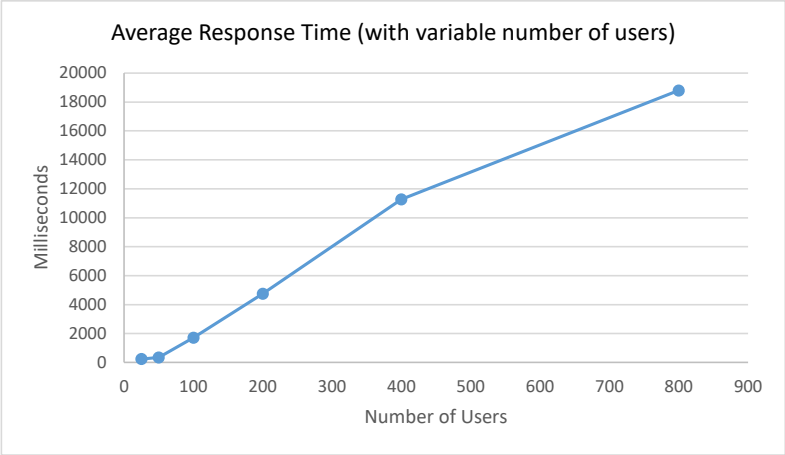


Figure 6.17: Average response time with variable number of users.

Throughput

The graphics shown in Figures 6.18 and 6.19 present the results obtained regarding the throughput measure.

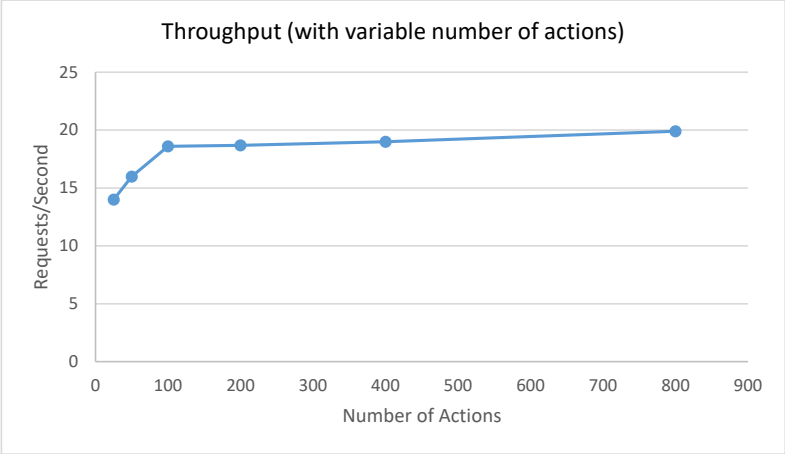


Figure 6.18: Throughput with variable number of actions.

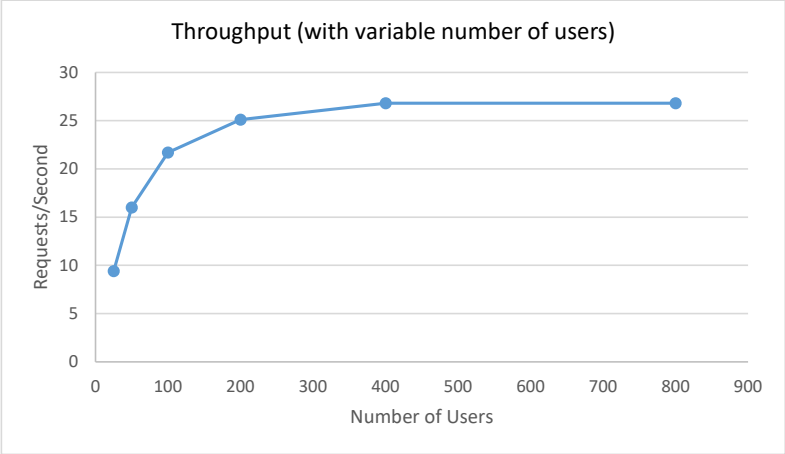


Figure 6.19: Throughput with variable number of users.

In Figure 6.18, the throughput has a significant growth between 25 and 100 users and then stabilizes with values between 18,5 and 20 requests per second. In Figure 6.19, the throughput increases logarithmically from less than 10 requests per second (with 25 users) to approximately 27 requests per second (with 800 users). This is due to the fact that with only 25 users sending requests with a think time of 2 seconds, the system is not handling as many requests as it can handle.

Chapter 7

Final Remarks

This chapter presents the conclusions of this thesis, in particular, we summarize the main results in Section 7.1 and we list the limitations and improvements suggested by users during the system validation as well as new functionalities that need to be added in Section 7.2.

7.1 Summary

This document presents the work performed on this thesis. The main contributions of this thesis were: (i) the requirements gathering through interviews with the different stakeholders; (ii) the design and implementation of a relational database that stores all the relevant data concerning active CSE Doctoral Degree students; and (iii) the design and implementation of the DD-IMS, a system that manages the information regarding CSE PhD students.

In Chapter 2, the main processes of the IST CSE Doctoral Degree are presented. These processes are based on the PGA procedures for all the IST Doctoral Degrees. Chapter 3 presents the Fenix limitations and the requirements gathered during the interviews with the DD-IMS stakeholders.

In Chapter 4, two types of software systems with a similar purpose to DD-IMS were described: (i) systems that were built from scratch; and (ii) ERP systems for Higher Education Institutions. Based on this analysis, we decided that the best solution for implementing the DD-IMS was to build a system from scratch. This chapter also analyzes several frameworks for web application development. We decided to use the Django framework for the DD-IMS implementation.

Chapter 5 describes the details about the design and implementation of the DD-IMS. The architecture of the DD-IMS is composed of three tiers: (i) a client that allows the user to access the application through a web browser; (ii) a web server that encloses the application logic; (iii) a relational database that stores the data about PhD students and their progress. Regarding its functionality, the DD-IMS web server is divided into nine distinct modules: (i) Evaluation of Applications; (ii) Registration; (iii) Study Plan; (iv) Supervision; (v) Curricular Plan; (vi) Thesis Proposal (CAT); (vii) Thesis; (viii) Statistics; and (ix) Doctoral Program Management. The DD-IMS communicates with the Fenix system to authenticate the users and to take advantage of the data that can be retrieved through the Fenix API, more specifically,

the students' personal and academic data (including information about their enrolled courses), and the list of the CSE Doctoral Degree courses. The DD-IMS is currently running on a virtual machine of the CSE/IST RNL and its database is deployed on the IST PostgreSQL database server.

Chapter 6 describes the experimental validation that we conducted for the DD-IMS, at three levels: functionality, usability, and performance. Regarding functionality, 609 tests were created for the functionality validation using the *unittest* library. All the created tests run without errors. Usability tests were performed with 29 future users of the system. Each user was assigned a set of specific tasks and each task had its success criteria. Finally, the users gave their feedback by completing a satisfaction questionnaire. This questionnaire included a set of questions with five response options for users, which allowed us to calculate the SUS score. We obtained an average SUS score of 88.19. With these results, we can conclude that, in general, users showed a high level of satisfaction with the DD-IMS usability. Finally, for *performance* evaluation, loading tests were performed to analyze the system response when used by a large number of users simultaneously, using the *JMeter* tool. The system performance was evaluated by the following measures: response time, throughput, and error rate. The results obtained show that the DD-IMS had a sustained growth in terms of error rate and average response time, maintaining an acceptable performance with a large number of concurrent users. After observing the behavior of the DD-IMS in response to a significant number of concurrent users (considerably larger than the potential number of active users involved in the CSE Doctoral Degree which is around 200/300 users), we can conclude that the DD-IMS meets the performance and reliability needs of an Information Management System for the CSE Doctoral Degree students.

The following main results were achieved by this thesis:

- A technical report[11] with the requirements gathered from the interviews with the different system stakeholders: students, supervisors, CSE department administrative staff, coordinator, CC3C members, and PGA administrative staff. This document also contains the diagrams (i.e. Use Cases Diagram, Class Diagram with the Data Models, Relational Model) that were produced during the Design phase of the DD-IMS, which are shown in Appendix A and B.

- A relational database that stores all the relevant data concerning CSE PhD students in a structured way, thus enabling to monitor each student's progress and to obtain statistics about the behavior of the PhD Doctoral program.

- The design and implementation of DD-IMS, a system that manages the information regarding CSE PhD students. DD-IMS encloses a web application, accessible to IST users, which provides a user interface so that users can visualize and manage the data regarding the CSE PhD students' processes. The DD-IMS has already entered into production. At this time, 78 CSE Doctoral Degree students had registered in DD-IMS, among the 89 students listed in the Fenix system as being registered (*matriculados*, in Portuguese) and active.

- A developer manual, available in Appendix C, that describes the setup and the structure of the DD-IMS software and gives instructions for future developers on how to use the DD-IMS, including

adding new modules, deploying the system, updating the system functionalities, creating database backups, migrating data from one database to another, accessing the server logs, running the unit tests, and adding more tests to the DD-IMS test suite.

7.2 Future Work

The first version of the DD-IMS is in production and is already being used within the CSE department. If the decision is to adopt it as the official tool within the department to manage the CSE doctoral program data, it must be maintained and improved, fixing problems detected by its users. Currently, DD-IMS has the following limitations:

1. Currently, the DD-IMS is still not sending e-mails to the users. The SMTP server that will be used to send mails automatically is not completely configured yet.

2. When requesting the students' curricular plan to the Fenix API, the enrollments in courses outside IST are not being returned (in Fenix, are called *Opção de Substituição* in Portuguese). To address this limitation, the DD-IMS should contain a new functionality, which allows the administrative staff or the coordinator to insert manually the information regarding the students' enrollment in a course, including semester, year, and grade. Moreover, the API endpoint can be modified (after this, the information obtained may not be complete) or become obsolete. This new functionality will also be very important to prevent this situation.

In addition, it is still necessary to integrate the following suggestions given by the users, grouped by user type:

1. **Students**:

   - Allow students to indicate their phone number without the country code. To support international phone numbers, the DD-IMS should display these two elements separately. To select the country code, users could choose the country from a list of flags.
   - For students that have multiple e-mails associated with their Fenix account, allow them to choose which of the e-mails they want to place in their personal information.

2. **Supervisors**:

   - Add a "Cancel" button which allows users to undo the modifications done in a jury proposal.
   - When declining a student's study plan or supervision, the DD-IMS should provide to the users a set of default justifications so that the users can select one of them if appropriate.

3. **Administrative Staff**:

   - Order the alerts by date.

- The DD-IMS should inform the user if other users are browsing on the same page as the user (e.g. with this functionality, an administrative staff member can realize that other administrative staff members are browsing the same page as her, and possibly will perform the same action as she is intending to perform).
- Place the statistics in a new tab.
- When visualizing a student's process, display the deadlines for the thesis proposal and thesis submissions (if not submitted yet).
- Display an alert to the administrative staff when students are close to their deadlines for the thesis proposal or thesis submission.

4. **Coordinators**:

   - When submitting the evaluation of an application, change the button name from "Submit Result" to "Submit Decision".
   - Change the name of some of the tabs (i.e. change "Evaluation of Juries" to "Review of Jury Proposals" and "Evaluation of Applications" to "Candidates").
   - When visualizing the list of PhD students, add a column that displays the students' thesis proposal defense date or the thesis proposal deadline if the student has not defended her thesis proposal yet. The date should be displayed in different colors, depending on the thesis proposal status: (i) Black, if the student did not defend the thesis proposal yet, but she is still within the deadline; (ii) Green, if the student already defended her thesis proposal; (iii) Red, if the deadline has expired; and (iv) Yellow, if the deadline has expired but a jury proposal was already submitted.

5. **CC3C/CCP members**:

   - Display the alerts due date for the alerts that have a deadline.
   - When a CC3C member submits her feedback about a jury proposal or an application, the success message, which informs that the feedback was submitted with success, should appear near the text submitted and not at the top of the page.

Besides the feedback given during the DD-IMS evaluation sessions, there are also other functionalities that need to be implemented in the future:

- For the students that have already concluded the academic part of the PhD, the system should automatically retrieve from Fenix the information about the students' study plan and curricular plan (enrolled courses and grades) without asking the students to submit their study plan.
- Facilitate the visualization of the students' progress, especially for supervisors. When visualizing a list of their students, the users should be able to understand the progress of each student

without needing to access her process in particular (i.e., clicking in a student to get full information about her). This could be done through the use of badges[1]. Each badge would symbolize the achievement of a PhD milestone (e.g. study plan defined, academic part concluded, thesis proposal defended, thesis defended).

- Make the DD-IMS fully legible for mobile devices.
- The administrative staff should be capable of inserting the PhD process of students that already have concluded their CSE Doctoral Degree, which includes the functionalities of inserting the students' personal and academic information, study plan, supervision, thesis proposal jury and defense minutes, and thesis jury and defense result.
- The DD-IMS should support the management of the students' progress report. Students submit annually a report that contains their work progress so that the thesis proposal jury members can follow the progress of the students' research work until they submit the thesis. The thesis proposal jury members will analyze the report and give feedback about the student's progress. The DD-IMS should allow students to submit their progress report. The president of the thesis proposal jury should be able to submit the feedback given by the thesis proposal jury.
- The DD-IMS should present new types of statistics to the users: (i) Number of active supervisors (supervisors that have PhD students at the moment); (ii) Number of students that did not submit their thesis proposal and already exceeded the deadline (24 months); and (iii) Number of students that did not submit their thesis and already exceeded the deadline (5 years or more if the student has requested any deadline extension).
- The DD-IMS should support the visualization of rapporteurs' opinion about the thesis, in particular by the supervisors. As the rapporteurs may not belong to IST and therefore cannot access the DD-IMS, the president of the jury should be capable of inserting the rapporteurs' opinion.
- The DD-IMS should support the submission of an equivalence to the thesis proposal, for the students that defend their thesis proposal outside the IST (e.g. CMU students can defend their thesis proposal only at CMU, without needing to defend their thesis proposal in IST).
- Allow the renaming of the committees (i.e. CC3C, CCP) mentioned in the DD-IMS.
- The DD-IMS should be able to support other IST Doctoral Degrees, adapting the study plan and supervision rules (e.g. the maximum number of ECTS, maximum number of supervisors). This could be done by having a configuration layer, with JSON files, where the supervision and study plan rules are defined.
- Display more suggestive and user-friendly error pages so that the user understands what went wrong while browsing the DD-IMS Web Application (e.g. When a student that is not enrolled or

[1] In this context, badges are symbols that represent milestones or the progress achieved (https://www.creativebloq.com/ux/ui-design-pattern-tips-achievements-badges-111413424)

has not been enrolled in the CSE Doctoral Degree, inform the student, in a user-friendly way, that she does not have access to the DD-IMS because she is not a CSE Doctoral Degree student).

- Make the DD-IMS configurable through parameters inserted in the Management tab of the DD-IMS Web Application (e.g. rename the committee names: CC3C, CCP, Scientific Committee).

Bibliography

[1] M. Bächle and P. Kirchberg. Ruby on rails. *IEEE software*, 24(6):105–108, 2007.

[2] R. Barata, S. Silva, D. Martinho, L. Cruz, and L. Guerra e Silva. Open APIs in Information Systems for Higher Education. *European University Information Systems (EUNIS) 2014 Congress*, 2014.

[3] R. Bhatnagar, A. Kumar, and S. Gupta. Role of Information Systems in an University Setup – A Case Study. *International Journal of Computer Science and Electronics Engineering (IJCSEE) , ISSN 2320–4028 (Online)*, 4:151–156, 2016.

[4] D. Biella, U. Blotevogel, M. Ney, and S. Radermacher. Student management with HISinOne–A status report. *European University Information Systems (EUNIS) 2013 Congress*, 2013.

[5] B. Chaushi and Z. Dika. Higher Education Information Systems: An Overview of the Latest Trends and Issues. *Annual International Meeting of Alb-Science Institute*, 2013.

[6] R. Elmasri and S. B. Navathe. *Fundamentals of Database Systems*. Pearson, 7th edition, 2015. ISBN 0133970779.

[7] U. Hübner[1], I. Duda, M. Merz, D. Natusch, and H.-D. Weckmann. HISinOne-Development and early adoption partnerships. *European University Information Systems (EUNIS) 2008 Congress*, 2008.

[8] A. Kulkarni, N. Hegde, M. Sharma, A. A. Kulkarni, N. Hegde, and M. Sharma. Educational ERP systems in the market–a comparative study. *International journal of innovative research science in technology*, 1(8):84–91, 2015.

[9] A. Leff and J. T. Rayfield. Web-application development using the model/view/controller design pattern. In *IEEE International Enterprise Distributed Object Computing Conference*, pages 118–127. IEEE, 2001.

[10] J. Mincer-Daszkiewicz. Framework for rapid in-house development of web applications for higher education institutions in Poland. *European University Information Systems (EUNIS) 2013 Congress*, 2013.

[11] H. Rocha and H. Galhardas. IST CSE Doctoral Degree Information Management System: Requirements Specification. Technical Report 1, INESC-ID, Jan. 2021.

Appendix A

Functionality: UML Use Cases

This appendix contains the UML Use Cases diagrams that were produced during the Design phase of the DD-IMS. These diagrams present the functionality of each of the DD-IMS modules. UML use cases diagrams were designed with the diagrams[1] tool.

A.1 Registration

Figure A.1 shows the diagram that represents the registration use cases.

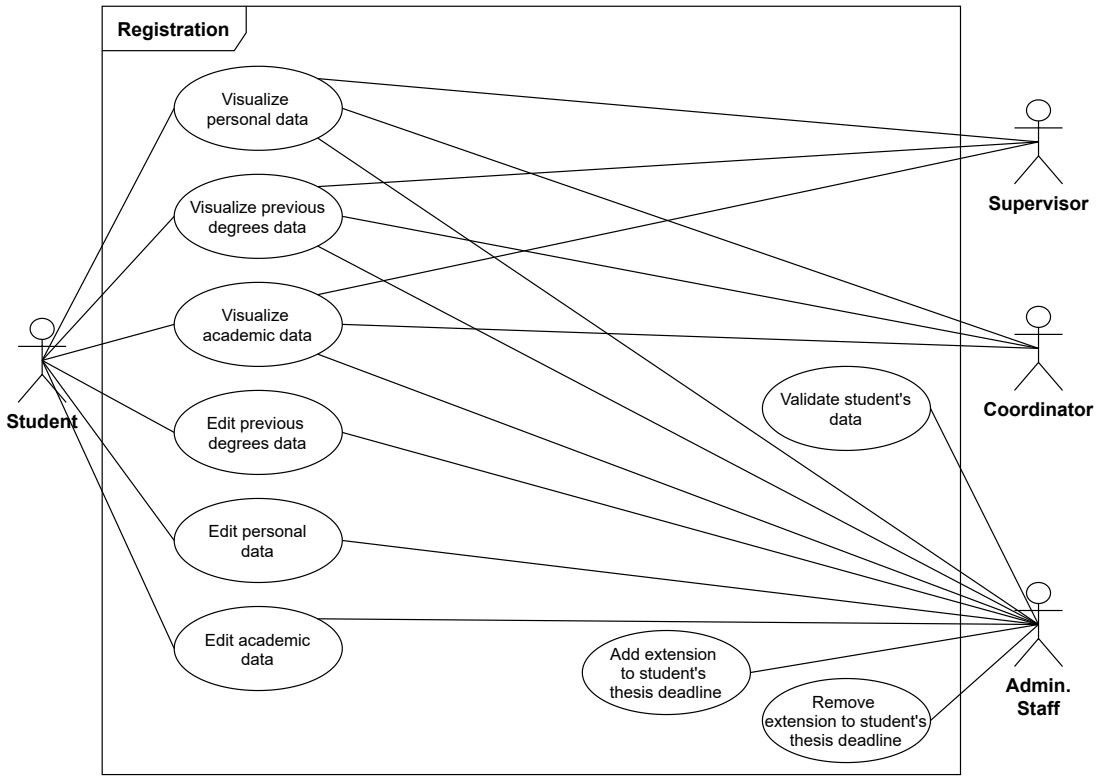


Figure A.1: Use Cases Diagram for Registration Module.

[1] https://app.diagrams.net/

A.2 Study Plan

Figure A.2 shows the diagram that represents the study plan use cases.

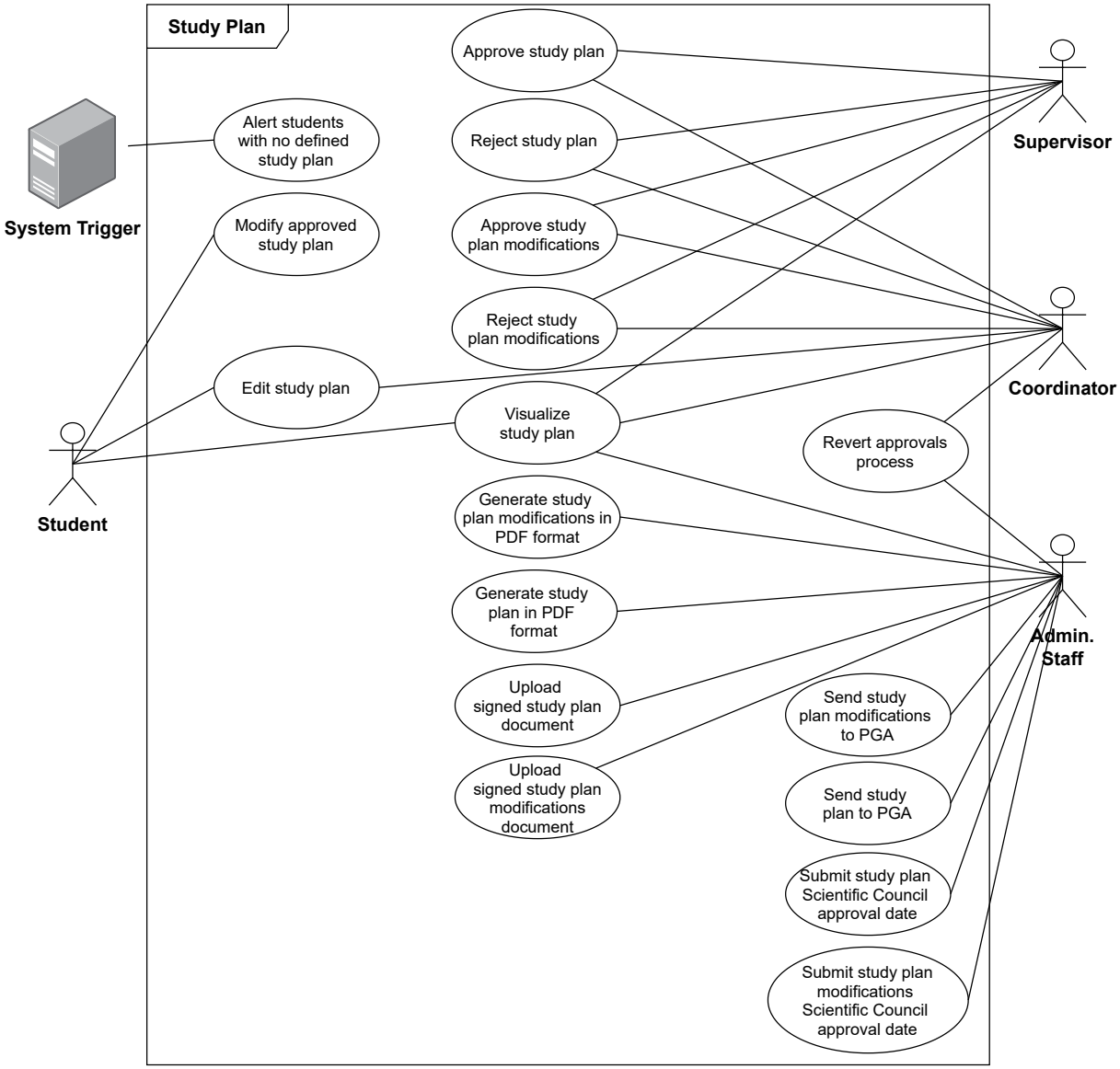


Figure A.2: Use Cases Diagram for Study Plan Module.

A.3 Supervision

Figure A.3 shows the diagram that represents the supervision use cases.

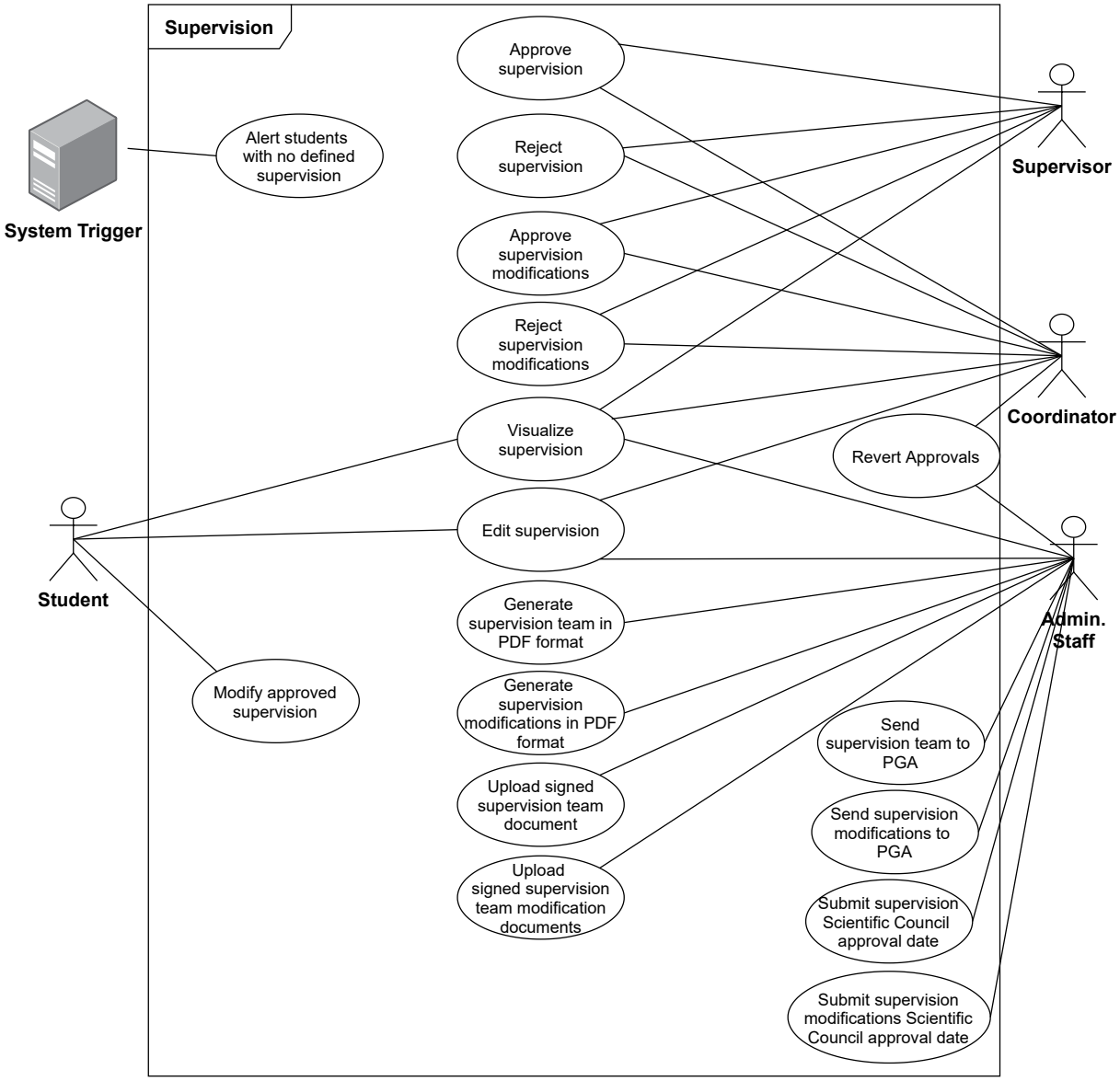


Figure A.3: Use Cases Diagram for Supervision Module.

A.4 Curricular Plan

Figure A.4 shows the diagram that represents the use case related to the Curricular Plan module.

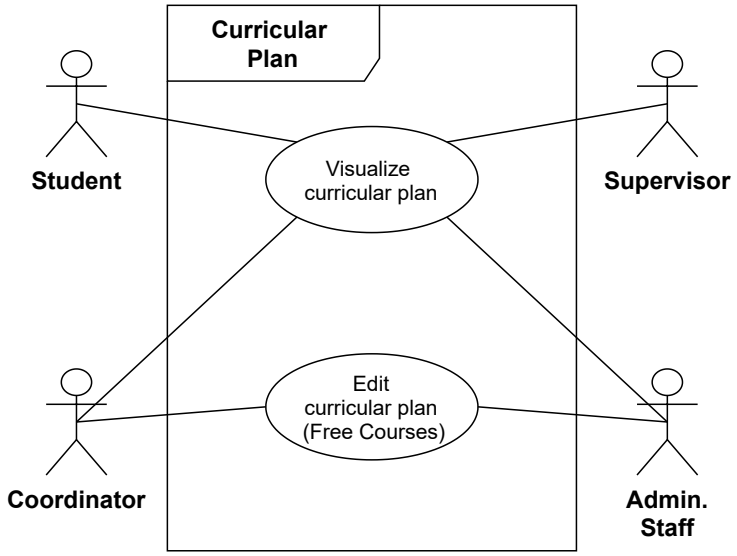


Figure A.4: Use Cases Diagram for Curricular Plan Module.

A.5 Thesis Proposal (CAT)

Figure A.5 shows the diagram that represents the use cases related to the Thesis Proposal module.

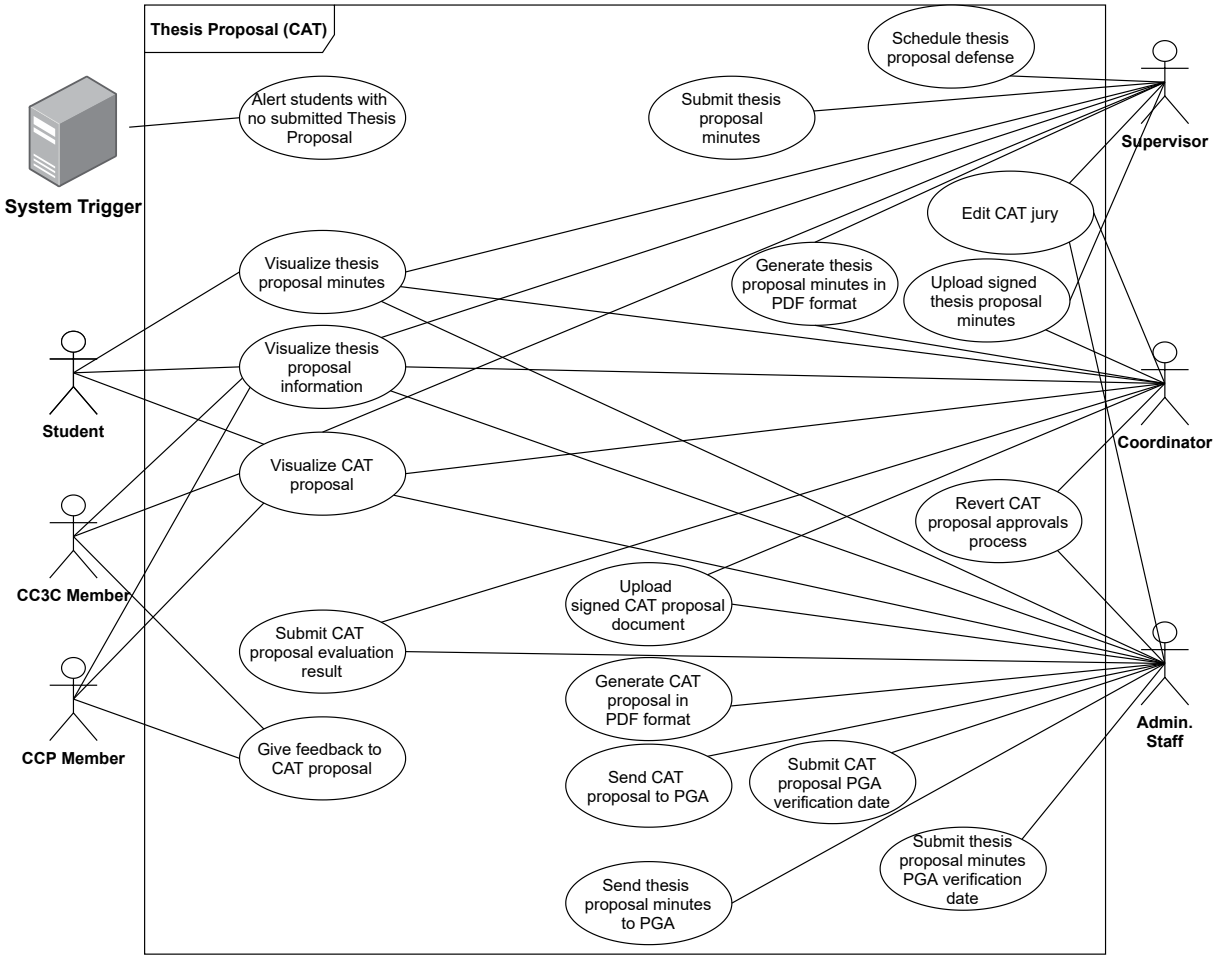


Figure A.5: Use Cases Diagram for Thesis Proposal (CAT) Module.

A.6 Thesis

Figure A.6 shows the diagram that represents the use cases related to the Thesis module.

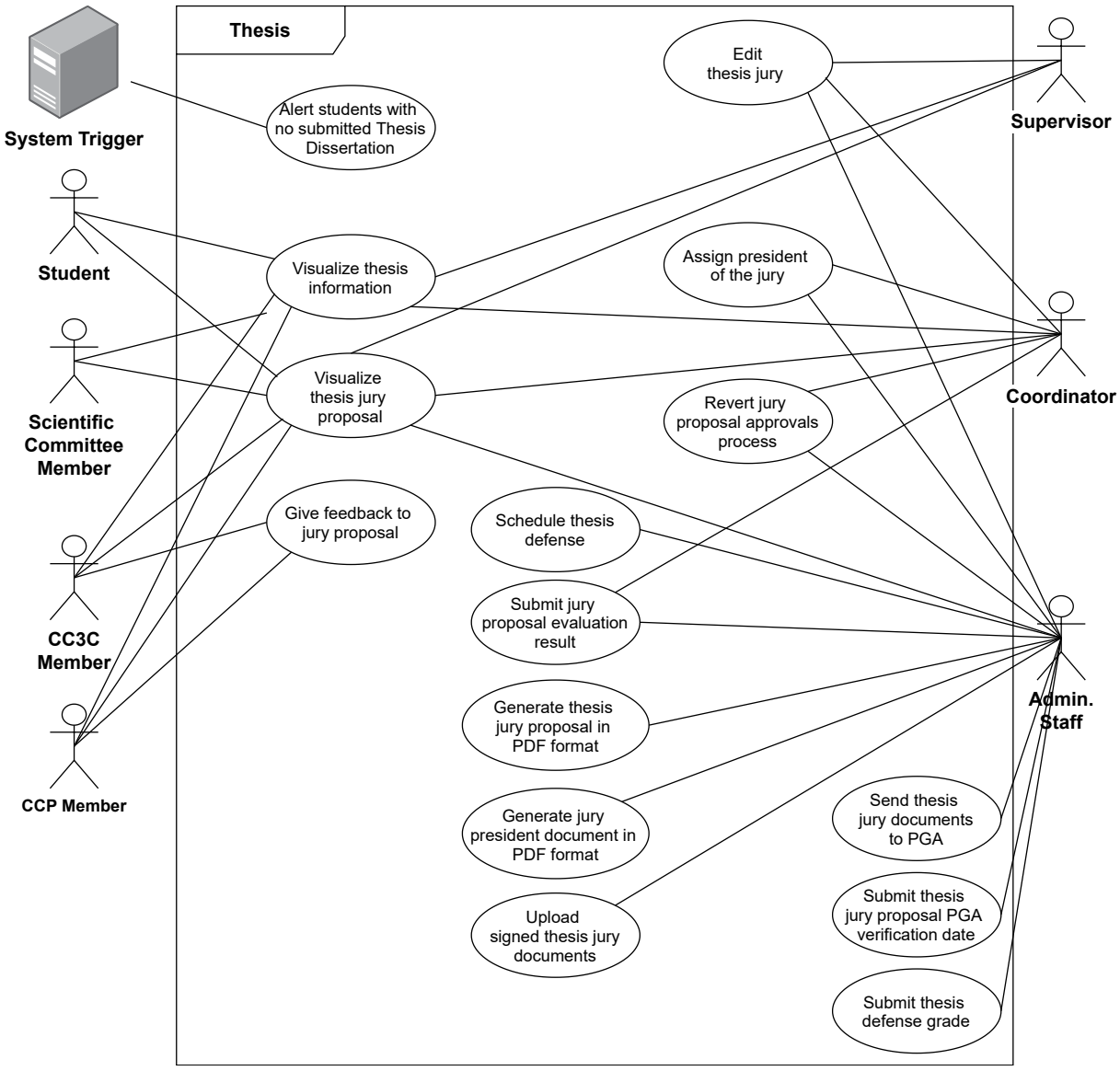


Figure A.6: Use Cases Diagram for Thesis Module.

A.7 Statistics

Figure A.7 shows the diagram that represents the use cases related to the Statistics module.

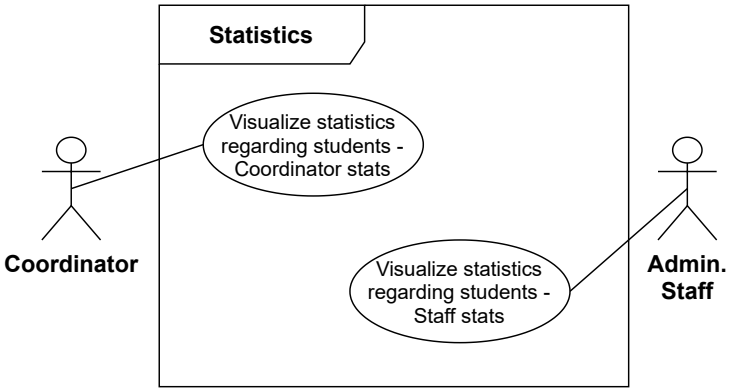


Figure A.7: Use Cases Diagram for Statistics Module.

A.8 Doctoral Program Management

Figure A.8 shows the diagram that represents the use cases related to the Doctoral Program Management module.

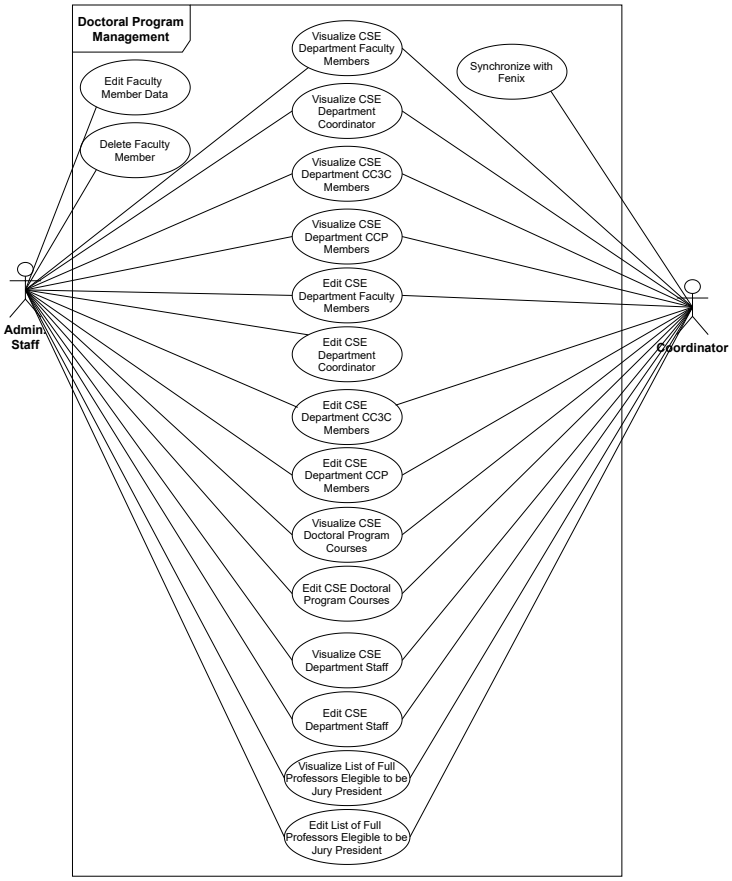


Figure A.8: Use Cases Diagram for Doctoral Program Management Module.

Appendix B

Database: UML Class Diagrams and Relational Model

This appendix contains the class diagrams (in Section B.1) that represent the Data Models, and the Relational Model (in Section B.2), produced during the Design phase of the DD-IMS.

B.1 Class Diagram

This section contains the UML Class diagrams that were produced during the Design phase of the DD-IMS. The notation used in the class diagrams is presented at the bottom of the class diagram. UML class diagrams were designed with the diagrams[1] tool.

B.1.1 Users

Before showing the class diagram for each module, the UML class diagram with the different types of users is shown in Figure B.1. All classes in this diagram extend from the User class, which contains the attributes that are common to all types of users.

For the users class diagram, we identified the following integrity constraints:

- **CD-IC1**: *Extends* relationship between *User* and its subclasses is mandatory.
- **CD-IC2**: A *Coordinator* must also be a *CC3C Member* and a *CCP Member*.

[1] https://app.diagrams.net/

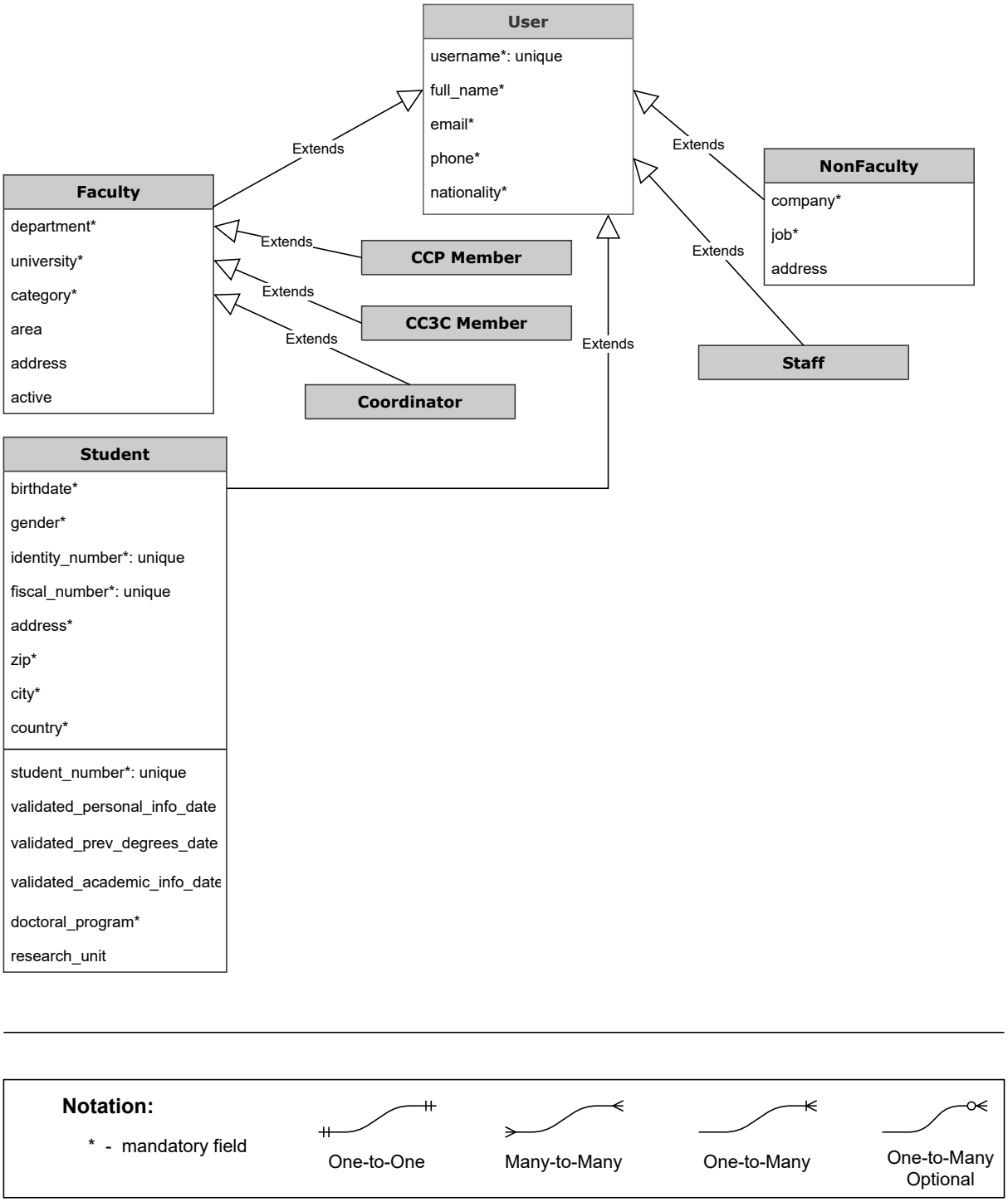


Figure B.1: User Class Diagram.

B.1.2 Registration

Figure B.2 shows the class diagram for the Registration module.

For this module, we identified the following integrity constraints:

- **CD-IC3**: The attribute *status* from *Registered Student* can take the following values:

  1. ”Active”

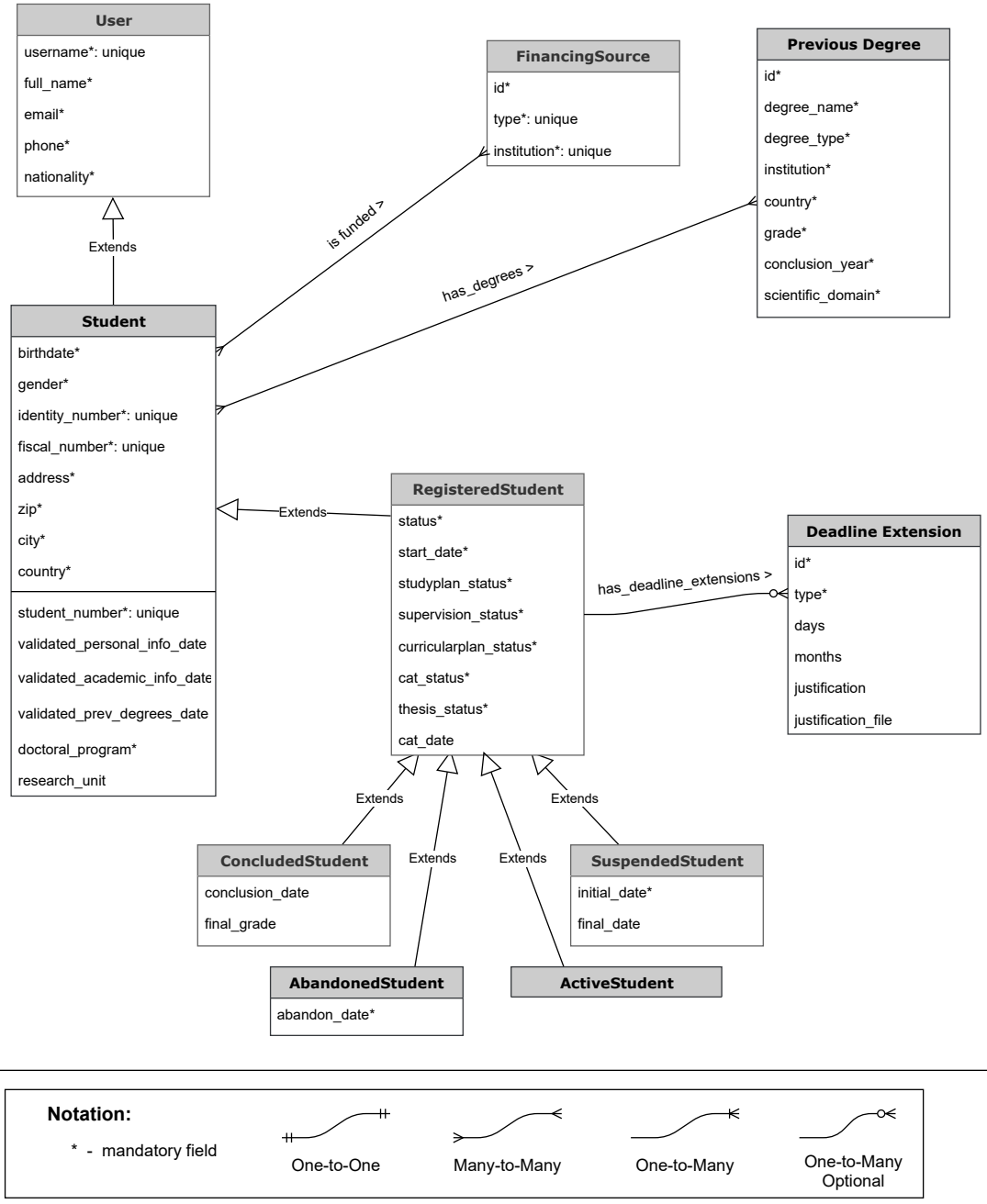


Figure B.2: Class Diagram for Registration module.

2. ”Abandoned”
3. ”Concluded”
4. ”Suspended”

- **CD-IC4**: *Extends* relationship between *Registered Student* and its subclasses is mandatory and disjoint.
- **CD-IC5**: The attribute *fs_type* from *Financing Source* can take the values ”Company”, ”Academic” or ”State” (Fundação para a Ciência e Tecnologia)[2].

[2]https://www.fct.pt/

- **CD-IC6**: The attribute *type* from *Deadline Extension* can take the values ”3 Months Extension”, ”2 Months Extension due to Lockdown”, ”1 Year Extension with Justification”, or ”Other”.

B.1.3 Study Plan

Figure B.3 shows the class diagram for the Study Plan module.

For this module, we identified the following integrity constraints:

- **CD-IC7**: The value of the *credits* attribute from *Course* for *Propaedeutic Courses* is always 0.
- **CD-IC8**: *Extends* relationship between *Course* and its subclasses is mandatory and disjoint.
- **CD-IC9**: The attribute *type* from *Prior Credits* can take the values ”Replacement”, ”Equivalence”, ”Exemption” or ”Training”.
- **CD-IC10**: The sum of *credits* in the relationships between *Study Plan* and *Course*, and *Study Plan* and *Prior Credits* must be at least the number of *total_credits*.
- **CD-IC11**: The sum of *credits* of *Free Course* for a given *Study Plan* is at most 12 and the *Free Course* must be given in a *university* that belongs to the University of Lisbon
- **CD-IC12**: The sum of *credits* of *Prior Credits* instances, for the same *Study Plan*, which has the attribute *type* with value ”Training” in the relationship between *Study Plan* and *Prior Credits* is at most 15% of the total sum of *credits* (4.5 ECTS).
- **CD-IC13**: The value of *total_credits* attribute from *Study Plan* must be at least 30.
- **CD-IC14**: The value of the attribute *coord_approval_date* from *StudyPlan* must be earlier than the value of the attribute *pga_send_date*.
- **CD-IC15**: The value of the attribute *pga_send_date* from *StudyPlan* must be earlier than the value of the attribute *sc_approval_date*.
- **CD-IC16**: A *Student* should be associated with, at least, one *StudyPlan* after 3 months since the start date of their PhD. If not, the system sends an alert to the student, her supervisor(s), and the coordinator.
- **CD-IC17**: The attribute *studyplan_status* from *Registered Student* can take the following values:
  1. ”Not Submitted”
  2. ”Submitted”
  3. ”Approved by Supervisor(s)”
  4. ”Rejected by Supervisor(s)”
  5. ”Approved by Coordinator”
  6. ”Rejected by Coordinator”

7. ”Submitted to PGA”
8. ”Approved by Scientific Council”
9. ”Requested Modification”
10. ”Modification Approved by Supervisor(s)”
11. ”Modification Rejected by Supervisor(s)”
12. ”Modification Approved by Coordinator”
13. ”Modification Rejected by Coordinator”
14. ”Modification Submitted to PGA”

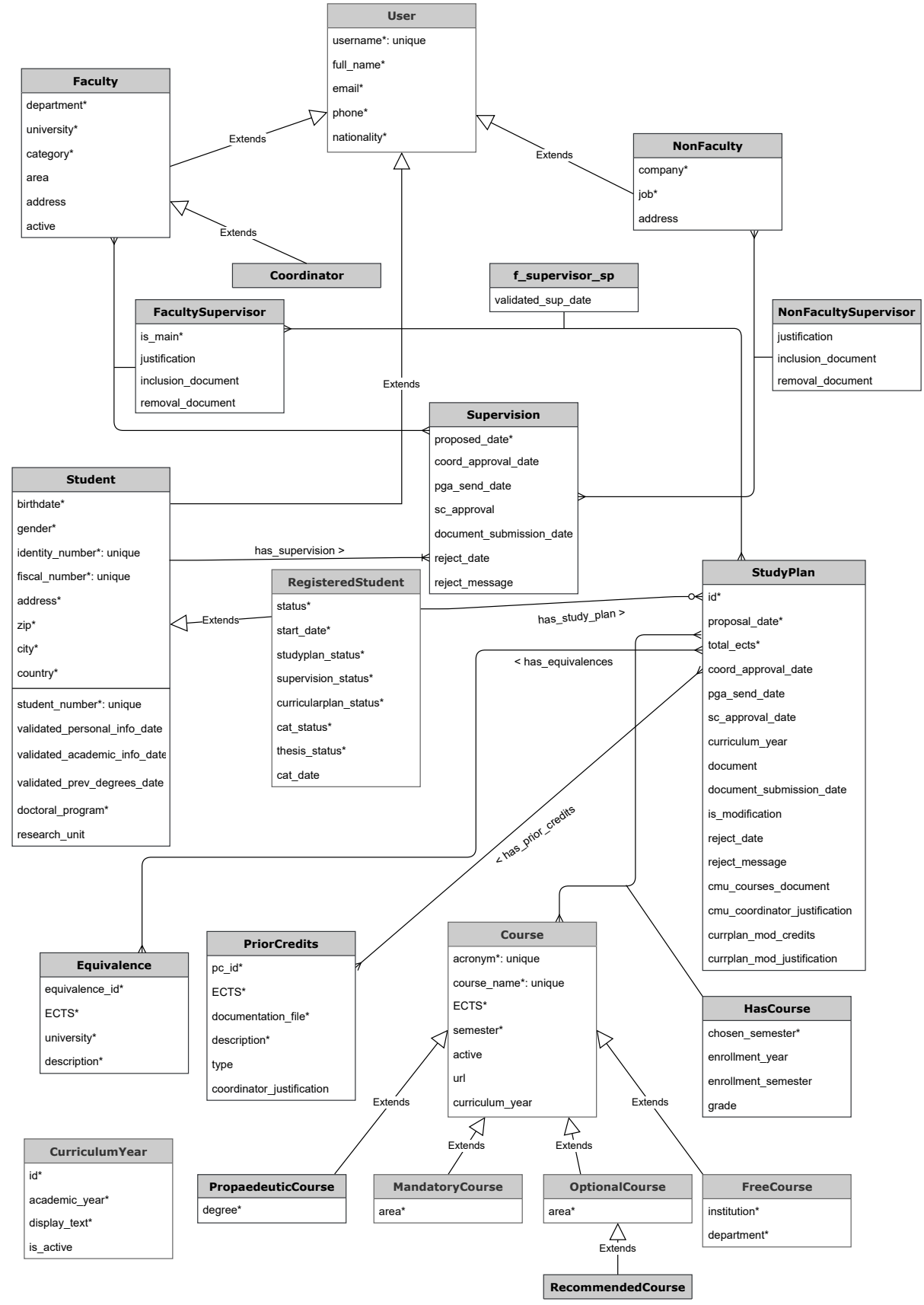


Figure B.3: Class Diagram for Study Plan module.

B.1.4 Supervision

Figure B.4 shows the class diagram for the Supervision module.

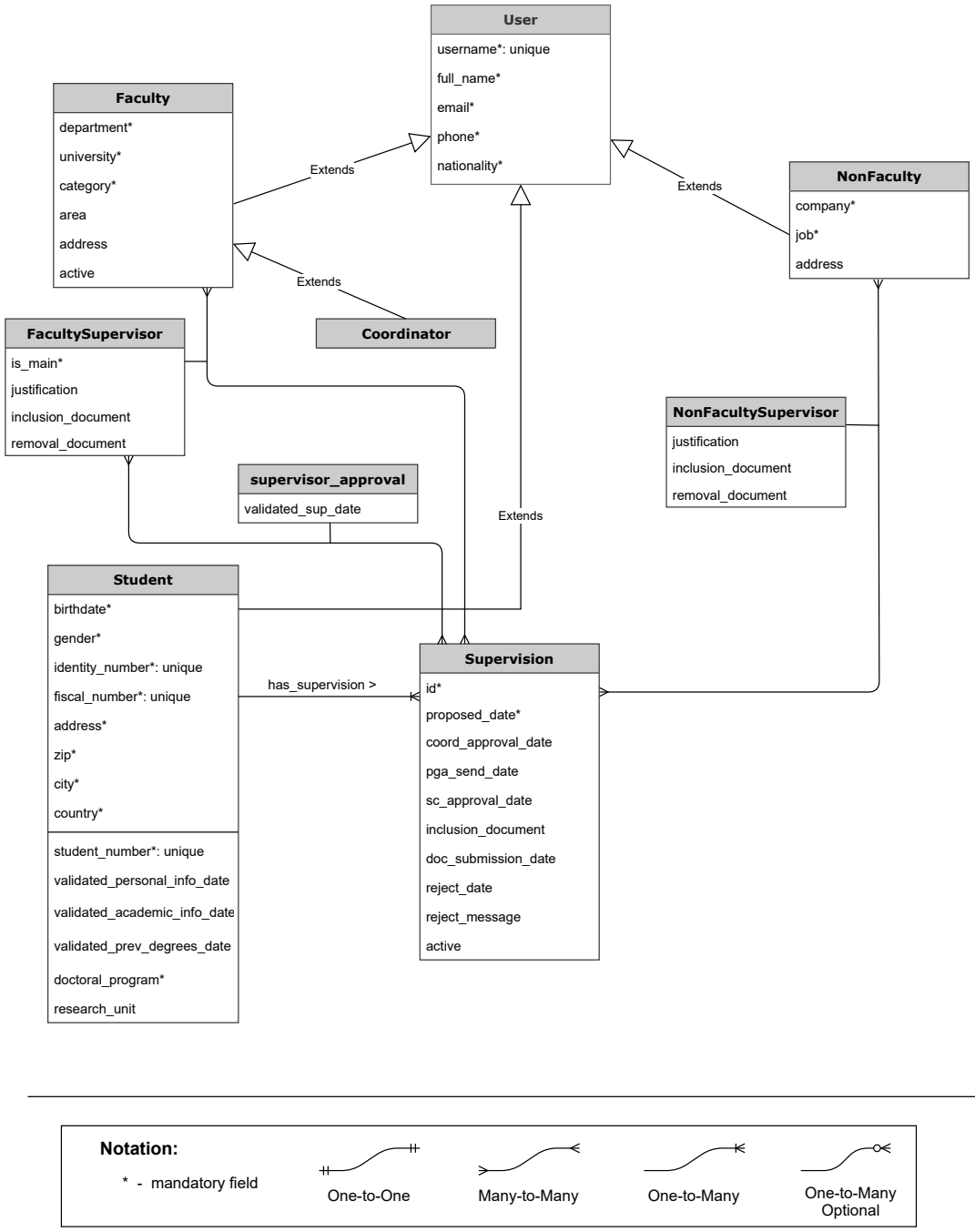


Figure B.4: Class Diagram for Supervision module.

For this module, we identified the following integrity constraints:

- **CD-IC18**: A *Student* must have at least one *FacultySupervisor* that belongs to the IST CSE Department.
- **CD-IC19**: A *Student* must have at least one *FacultySupervisor*.
- **CD-IC20**: A *Student* must have a maximum of three *FacultySupervisors* and *NonFacultySupervi-*

sors.

- **CD-IC21**: The value of the attribute *coord_approval_date* from *Supervision* must be earlier than the value of the attribute *pga_send_date*.
- **CD-IC22**: The value of the attribute *pga_send_date* from *Supervision* must be earlier than the value of the attribute *sc_approval_date*.
- **CD-IC23**: A *Student* should be associated with, at least, one *Supervision* after 1 year since the start date of their PhD. If not, the system sends an alert to the student and the coordinator.
- **CD-IC24**: The attribute *supervision_status* from *Registered Student* can take the following values:
  1. ”Not Submitted”
  2. ”Submitted”
  3. ”Approved by Supervisor(s)”
  4. ”Rejected by Supervisor(s)”
  5. ”Approved by Coordinator”
  6. ”Rejected by Coordinator”
  7. ”Submitted to PGA”
  8. ”Approved by Scientific Council”
  9. ”Requested Modification”
  10. ”Modification Approved by Supervisor(s)”
  11. ”Modification Rejected by Supervisor(s)”
  12. ”Modification Approved by Coordinator”
  13. ”Modification Rejected by Coordinator”
  14. ”Modification Submitted to PGA”

B.1.5 Curricular Plan

Figure B.5 shows the class diagram for the Curricular Plan module.

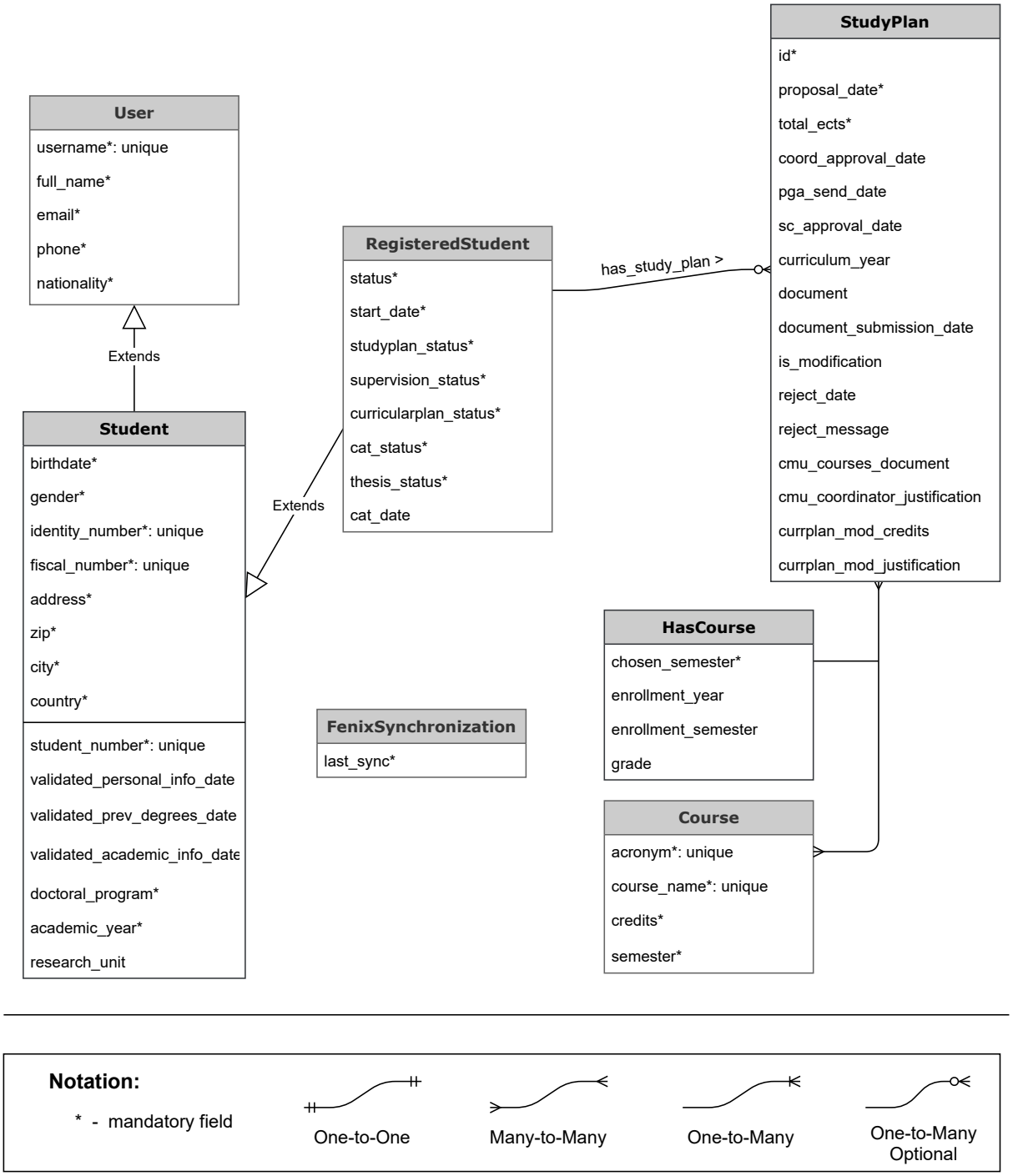


Figure B.5: Class Diagram for Curricular Plan module.

B.1.6 Thesis Proposal (CAT)

Figure B.6 shows the class diagram for the Thesis Proposal module.

For this module, we identified the following integrity constraints:

- **CD-IC25**: All the *Supervisors* must be part of the *Thesis Proposal Jury*.
- **CD-IC26**: The *thesis_proposal_status* attribute from *Student* must take one of the following values:
  1. ”Not Submitted”;

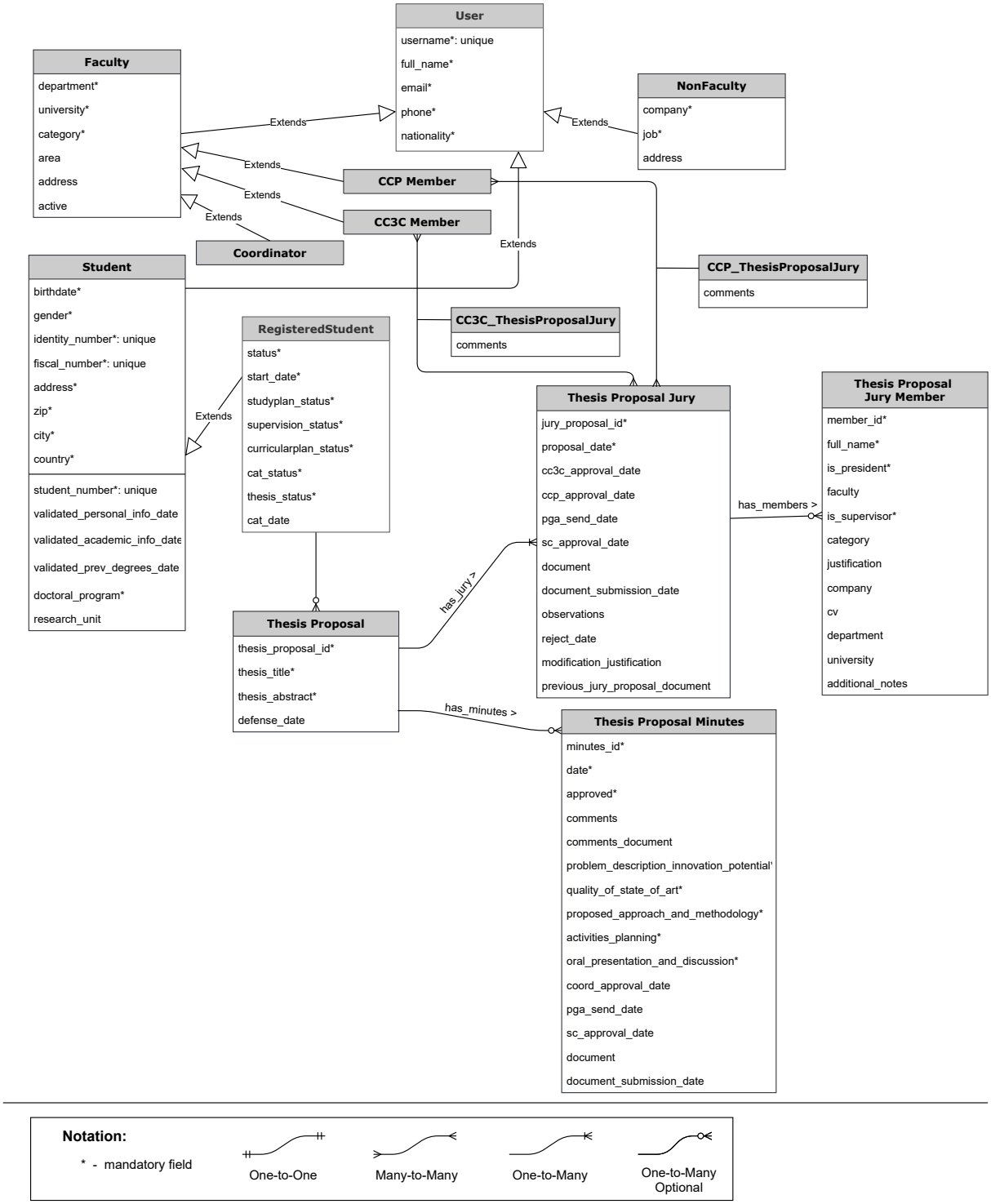


Figure B.6: Class Diagram for Thesis Proposal module.

2. ”Jury Proposal Submitted”;
3. ”Jury Proposal Approved by CC3C”;
4. ”Jury Proposal Rejected by CC3C”;
5. ”Jury Proposal Approved by CCP”;
6. ”Jury Proposal Rejected by CCP”;

7. ”Jury Proposal Submitted to PGA”;
8. ”Jury Proposal Approved by Scientific Council”;
9. ”Defense Scheduled”;
10. ”Minutes Submitted”;
11. ”Minutes Signed by Coordinator”;
12. ”Minutes Submitted to PGA”;
13. ”Minutes Approved by Scientific Council”.

- **CD-IC27**: The *grade* attribute from *Thesis Proposal Minutes* must take one of the following values:
  - ”Approved”;
  - ”Failed”.
- **CD-IC28**: The *problem_description_innovation_potential*, *quality_of_state_of_art*, *proposed_approach_and_methodology*, *activities_planning*, and *oral_presentation_and_discussion* attributes from *Thesis Proposal Minutes* must take one of the following values:
  - ”Fail”;
  - ”Fair”;
  - ”Good”;
  - ”Very Good”;
  - ”Outstanding”.
- **CD-IC29**: The value of the attribute *cc3c_approval_date* from *Thesis Proposal Jury* must be earlier than the value of the attribute *ccp_approval_date*.
- **CD-IC30**: The value of the attribute *thesis_proposal_status* from *Student* should not be equal to ”Not Submitted” after 24 months since the start date of their PhD. If that happens, the system sends an alert to the student, her supervisor(s), and the coordinator.

B.1.7 Thesis

Figure B.7 shows the class diagram for the Thesis module.

For this module, we identified the following integrity constraints:

- **CD-IC31**: The *thesis_status* attribute from *Student* must take one of the following values:
  1. ”Not Submitted”;
  2. ”Jury Proposal Submitted”;
  3. ”Jury Proposal Approved by CC3C”;
  4. ”Jury Proposal Rejected by CC3C”;

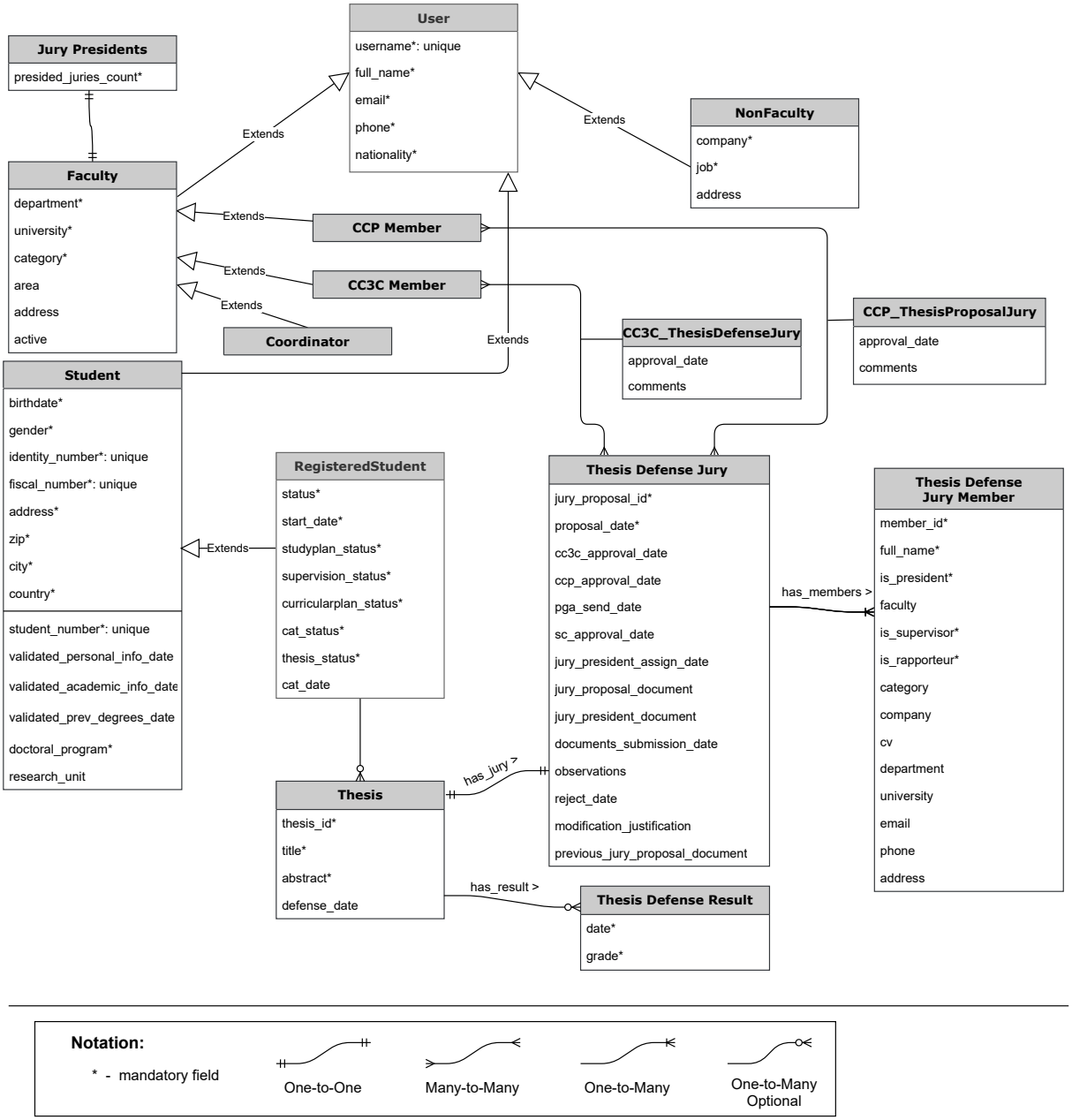


Figure B.7: Class Diagram for Thesis module.

5. ”Jury Proposal Approved by CCP”;
6. ”Jury Proposal Rejected by CCP”;
7. ”Jury President Assigned”;
8. ”Jury Proposal Submitted to PGA”;
9. ”Jury Proposal Approved by Scientific Council”;
10. ”Defense Scheduled”;
11. ”Defended”.

- **CD-IC32**: The *grade* attribute from *Thesis Defense Result* must take one of the following values:
  - ”Rejected”;

  - ”Approved”;
  - ”Approved with Distinction”;
  - ”Approved with Distinction and Praise”.

- **CD-IC33**: The value of the attribute *cc3c_approval_date* from *Thesis Defense Jury* must be earlier than the value of the attribute *ccp_approval_date*.
- **CD-IC34**: The value of the attribute *thesis_status* from *Student* should not be equal to ”Not Submitted” after 4 years since the start date of their PhD. If that happens and the student did not require any extension for the thesis defense deadline, the system sends an alert to the student, her supervisor(s), and the coordinator.
- **CD-IC35**: The attribute *presided_juries_count* from *Jury Presidents* has a default value of 0

B.2 Relational Model

This section contains the relational model that was produced during the Design phase of the DD-IMS. The relational model was obtained from the class diagram described above, by applying conversion rules. The relational model presented below is an optimized version of the original relational model. This optimized version reduces the number of tables and is closer to the concrete implementation of the database than the original one.

To represent the relational models, the notation used was the following (Primary Key is underlined and Foreign Key is represented as FK):

$table_1(\underline{attr_1}, attr_2, \ldots)$
$table_2(\underline{attr_3}, attr_4, \ldots)$
$attr_3$: FK($table_1$)

The integrity constraints and null constraints of the relational model are represented below each relation. There are some integrity constraints from the class diagrams that are directly propagated to the relational model. Other integrity constraints of the relational model have to be included because the expression power of the relational model is lower than the expression power of the UML class diagrams.

B.2.1 Users

User(username, full_name, email, phone, nationality):

- full_name: NOT NULL
- email: NOT NULL
- phone: NOT NULL
- nationality: NOT NULL

- **R-IC-1**: each username value must exist in Faculty or Student or Non-Faculty Supervisor or Staff

Staff(username):

- username: FK(User)

Faculty(username, department, university, category, area, address, active, is_coordinator, is_cc3c_member, is_ccp_member):

- username: FK(User)
- **R-IC-2**: if a Faculty tuple has is_coordinator attribute with value ”True”, the is_cc3c_member and is_ccp_member attributes must also have value ”True”

Student(username, birthdate, gender, identity_number, fiscal_number, address, zip, city, country, student_number, validated_personal_info_date, validated_prev_degrees_date, validated_academic_info_date, doctoral_program, research_unit, status, start_date, studyplan_status, supervision_status, curricularplan_status, cat_status, thesis_status, cat_date):

- username: FK(User)
- unique(identity_number)
- unique(fiscal_number)
- unique(student_number)
- birthdate: NOT NULL
- gender: NOT NULL
- identity_number: NOT NULL
- fiscal_number: NOT NULL
- address: NOT NULL
- zip: NOT NULL
- city: NOT NULL
- country: NOT NULL
- student_number: NOT NULL
- doctoral_program: NOT NULL
- status: NOT NULL
- start_date: NOT NULL
- studyplan_status: NOT NULL

- supervision_status: NOT NULL
- curricularplan_status: NOT NULL
- cat_status: NOT NULL
- thesis_status: NOT NULL
- **R-IC-3**: each username value must exist in Active or Suspended or Abandoned or Concluded. The attribute status from Student can take one of those values ('Active', 'Suspended', 'Abandoned' or 'Concluded')
- **R-IC-4**: the same username value cannot exist in Active, Suspended, Abandoned, and Concluded simultaneously
- **R-IC-5**: the attributes studyplan_status and supervision_status from Student can take the values:
    1. "Not Submitted"
    2. "Submitted"
    3. "Approved by Supervisor(s)"
    4. "Rejected by Supervisor(s)"
    5. "Approved by Coordinator"
    6. "Rejected by Coordinator"
    7. "Submitted to PGA"
    8. "Approved by Scientific Council"
    9. "Requested Modification"
    10. "Modification Approved by Supervisor(s)"
    11. "Modification Rejected by Supervisor(s)"
    12. "Modification Approved by Coordinator"
    13. "Modification Rejected by Coordinator"
    14. "Modification Submitted to PGA"
- **R-IC-6**: the attribute cat_status from Student can take the values:
    1. "Not Submitted";
    2. "Jury Proposal Submitted";
    3. "Jury Proposal Approved by CC3C";
    4. "Jury Proposal Rejected by CC3C";
    5. "Jury Proposal Approved by CCP";
    6. "Jury Proposal Rejected by CCP";

7. ”Jury Proposal Submitted to PGA”;
8. ”Jury Proposal Approved by Scientific Council”;
9. ”Defense Scheduled”;
10. ”Minutes Submitted”;
11. ”Minutes Signed by Coordinator”;
12. ”Minutes Submitted to PGA”;
13. ”Minutes Approved by Scientific Council”.

- **R-IC-7**: the attribute thesis_status from Student can take the values:

1. ”Not Submitted”;
2. ”Jury Proposal Submitted”;
3. ”Jury Proposal Approved by CC3C”;
4. ”Jury Proposal Rejected by CC3C”;
5. ”Jury Proposal Approved by CCP”;
6. ”Jury Proposal Rejected by CCP”;
7. ”Jury President Assigned”;
8. ”Jury Proposal Submitted to PGA”;
9. ”Jury Proposal Approved by Scientific Council”;
10. ”Defense Scheduled”;
11. ”Defended”.

- **R-IC-8**: The value of the attribute *cat_status* from *Student* should not be equal to ”Not Submitted” after 24 months since the start date of their PhD. If that happens, the system sends an alert to the student.

- **R-IC-9**: The value of the attribute *thesis_status* from *Student* should not be equal to ”Not Submitted” after 4 years since the start date of their PhD. If that happens and the student did not require any extension for the thesis defense deadline, the system sends an alert to the student.

NonFaculty(username, address, company, job):

- username: FK(User)
- company: NOT NULL
- job: NOT NULL

B.2.2 Registration

ConcludedStudent(username, conclusion_date, final_grade):

- username: FK(Student)

ActiveStudent(username):

- username: FK(Student)

SuspendedStudent(username, initial_date, final_date):

- username: FK(Student)
- initial_date: NOT NULL

AbandonedStudent(username, abandon_date):

- username: FK(Student)
- abandon_date: NOT NULL

FinancingSource(id, type, institution):

- **R-IC-10**: the attribute type can take the values ”Company”, ”Academic” or ”State”

PreviousDegree(id, degree_name, degree_type, institution, country, grade, conclusion_year, scientific_domain):

- degree_name: NOT NULL
- degree_type: NOT NULL
- institution: NOT NULL
- country: NOT NULL
- grade: NOT NULL
- conclusion_year: NOT NULL
- scientific_domain: NOT NULL

DeadlineExtension(id, student, type, months, days, justification, justification_file):

- student: FK(Student)
- type: NOT NULL
- days: NOT NULL
- **R-IC-11**: the attribute type can take the values ”3 Months Extension”, ”2 Months Extension due to Lockdown”, ”1 Year Extension with Justification”, or ”Other”

has_degrees(username, degree_name):

- username: FK(Student)
- degree_name: FK(PreviousDegree)

is_funded(institution, type, username):

- institution, type: FK(FinancingSource)
- username: FK(Student)

B.2.3 Study Plan

StudyPlan(id, proposal_date, total_ects, coord_approval_date, pga_send_date, sc_approval_date, curriculum_year, document, document_submission_date, is_modification, reject_date, reject_message, username, cmu_courses_document, cmu_coordinator_justification, curricularplan_modification_credits, curricularplan_modification_credits_justification):

- username: FK(Student)
- proposal_date: NOT NULL
- total_credits: NOT NULL
- **R-IC-12**: the value of total_ects of Study Plan must be at least 30
- **R-IC-13**: The value of the attribute coord_approval_date must be earlier than the value of the attribute pga_send_date
- **R-IC-14**: The value of the attribute pga_send_date must be earlier than the value of the attribute sc_approval_date
- **R-IC-15**: A *Student* should be associated with, at least, one *StudyPlan* after 3 months since the start date of their PhD. If not, the system sends an alert to the student, her supervisor(s), and the coordinator.

CurriculumYear(id, academic_year, display_text, is_active):

- academic_year: NOT NULL
- display_text: NOT NULL

Course(acronym, course_name, ECTS, semester, active, url, curriculum_year):

- unique(course_name)
- course_name: NOT NULL
- ECTS: NOT NULL

- semester: NOT NULL
- **R-IC-16**: each acronym value must exist in PropaedeuticCourse or MandatoryCourse or FreeCourse or OptionalCourse
- **R-IC-17**: the same acronym value cannot exist in Propaedeutic, Mandatory, Free and Optional simultaneously

PropaedeuticCourse(course_name, degree):

- course_name: FK(Course)
- degree: NOT NULL
- **R-IC-18**: the value of credits for a propaedeutic course is 0

MandatoryCourse(course_name, area):

- course_name: FK(Course)
- area: NOT NULL

FreeCourse(course_name, institution, department):

- course_name: FK(Course)
- institution: NOT NULL
- department: NOT NULL

OptionalCourse(course_name, area, is_recommended):

- course_name: FK(Course)

RecommendedCourse(course_name)

- course_name: FK(OptionalCourse)

PriorCredits(pc_id, ECTS, documentation_file, description, type, coordinator_justification):

- ECTS: NOT NULL
- documentation_file: NOT NULL
- description: NOT NULL
- **R-IC-19**: the attribute type can take the values ”Replacement”, ”Equivalence”, ”Exemption” or ”Training”
- **R-IC-20**: a student cannot have more than 4.5 ECTS from Prior Credits of type ”Training”

Equivalence(equivalence_id, ECTS, university, description):

- ECTS: NOT NULL

- university: NOT NULL
- description: NOT NULL

FenixSynchronization(id, last_sync):

- last_sync: NOT NULL

has_course(sp_id, course_name, chosen_semester, enrollment_year, enrollment_semester, grade):

- sp_id: FK(StudyPlan)
- course_name: FK(Course)
- chosen_semester: NOT NULL
- **R-IC-21**: The value of the attribute chosen_semester must be equal to the value of the attribute semester in Course (except if the attribute semester of Course has value ”Both”)

has_prior_credits(sp_id, pc_id):

- sp_id: FK(StudyPlan)
- pc_id: FK(PriorCredits)

f_Supervisor_sp(supervisor_username, student_username, study_plan, validated_sup_date):

- supervisor_username, student_username: FK(FacultySupervisor)
- study_plan: FK(StudyPlan)

B.2.4 Supervision

Supervision(id, proposal_date, coord_approval_date, pga_send_date, sc_approval_date, inclusion_doc, doc_submission_date, reject_date, reject_message, active, student_username):

- student_username: FK(Student)
- proposal_date: NOT NULL
- **R-IC-22**: The value of the attribute coord_approval_date must be earlier than the value of the attribute pga_send_date
- **R-IC-23**: The value of the attribute pga_send_date must be earlier than the value of the attribute sc_approval_date
- **R-IC-24**: A *Student* should be associated with, at least, one *Supervision* after 1 year since the start date of their PhD. If not, the system sends an alert to the student and the coordinator.

FacultySupervisor(supervisor_username, supervision_id, is_main, justification, inclusion_document, removal_document):

- supervisor_username: FK(Faculty)
- supervision_id: FK(Supervision)
- is_main: NOT NULL

NonFacultySupervisor(supervisor_username, supervision_id, justification, inclusion_document, removal_document):

- supervisor_username: FK(NonFaculty)
- supervision_id: FK(Supervision)

supervisor_approval(supervisor_username, supervisor_supervision_id, supervision_id, validated_sup_date):

- supervisor_username, supervisor_supervision_id: FK(FacultySupervisor)
- supervision_id: FK(Supervision)

B.2.5 Thesis Proposal (CAT)

ThesisProposal(thesis_proposal_id, thesis_title, thesis_abstract, defense_date, username):

- username: FK(Student)
- thesis_title: NOT NULL
- thesis_abstract: NOT NULL

ThesisProposalJury(juryproposal_id, proposal_date, cc3c_approval_date, ccp_approval_date, pga_send_date, sc_approval_date, document, document_submission_date, observations, reject_date, modification_justification, previous_jury_proposal_document, thesis_proposal):

- thesis_proposal: FK(ThesisProposal)
- proposal_date: NOT NULL
- **R-IC-25**: The value of the attribute cc3c_approval_date must be earlier than the value of the attribute ccp_approval_date

ThesisProposalMinutes(minutes_id, date, approved, comments, comments_document, problem_description_and_innovation_potential, quality_of_state_of_art, proposed_approach_and_methodology, activities_planning, oral_presentation_and_discussion, coord_approval_date, pga_send_date, sc_approval_date, document, document_submission_date, thesis_proposal):

- thesis_proposal: FK(ThesisProposal)
- date: NOT NULL
- approved: NOT NULL

- problem_description_and_innovation_potential: NOT NULL
- quality_of_state_of_art: NOT NULL
- proposed_approach_and_methodology: NOT NULL
- activities_planning: NOT NULL
- oral_presentation_and_discussion: NOT NULL
- **R-IC-26**: The value of the attributes *problem_description_innovation_potential*, *quality_of_state_of_art*, *proposed_approach_and_methodology*, *activities_planning*, and *oral_presentation_and_discussion* must take one of the following values:
  - ”Fail”;
  - ”Fair”;
  - ”Good”;
  - ”Very Good”;
  - ”Outstanding”.

ThesisProposalJuryMember(member_id, juryproposal_id, full_name, faculty, category, justification, is_president, is_supervisor, company, cv, department, university, additional_notes):

- juryproposal_id: FK(ThesisProposalJury)
- faculty: FK(Faculty)
- full_name: NOT NULL
- is_president: NOT NULL
- is_supervisor: NOT NULL

CC3C_ThesisProposalJury(juryproposal_id, username, comments):

- juryproposal_id: FK(ThesisProposalJury)
- username: FK(CC3C_Member)

CCP_ThesisProposalJury(juryproposal_id, username, comments):

- juryproposal_id: FK(ThesisProposalJury)
- username: FK(CCP_Member)

B.2.6 Thesis

Thesis(thesis_id, title, abstract, defense_date, username):

- username: FK(Student)
- title: NOT NULL
- abstract: NOT NULL

ThesisDefenseJury(juryproposal_id, proposal_date, cc3c_approval_date, ccp_approval_date, pga_send_date, sc_approval_date, jury_president_assign_date, jury_proposal_document, jury_president_document, documents_submission_date, observations, reject_date, modification_justification, previous_jury_proposal_document, thesis):

- thesis: FK(Thesis)
- proposal_date: NOT NULL
- **R-IC-27**: The value of the attribute cc3c_approval_date must be earlier than the value of the attribute ccp_approval_date

ThesisDefenseJuryMember(member_id, juryproposal_id, full_name, faculty, category, is_rapporteur, is_president, is_supervisor, company, cv, department, university, email, phone, address):

- juryproposal_id: FK(ThesisDefenseJury)
- faculty: FK(Faculty)
- full_name: NOT NULL
- is_president: NOT NULL
- is_supervisor: NOT NULL
- is_rapporteur: NOT NULL

CC3C_ThesisDefenseJury(juryproposal_id, username, comments):

- juryproposal_id: FK(ThesisDefenseJury)
- username: FK(CC3C_Member)

CCP_ThesisDefenseJury(juryproposal_id, username, comments):

- juryproposal_id: FK(ThesisDefenseJury)
- username: FK(CCP_Member)

ThesisDefenseResult(result_id, date, grade, thesis):

- thesis: FK(Thesis)

- date: NOT NULL
- grade: NOT NULL
- **R-IC-28**: The value of the attribute *grade* must take one of the following values:
  - ”Rejected”;
  - ”Approved”;
  - ”Approved with Distinction”;
  - ”Approved with Distinction and Praise”.

JuryPresidents(id, faculty, presided_juries_count):

- faculty: FK(Faculty)
- **R-IC-29**: The attribute *presided_juries_count* has a default value of 0

Appendix C

Developer Guide

This appendix describes the DD-IMS software, as well as the instructions on how to use it, namely:

- The DD-IMS setup (Section C.1);
- The structure of the DD-IMS software (Section C.2);
- How to add a new module to DD-IMS (Section C.2.1);
- How to deploy the system (Section C.3);
- How to update the system (Section C.3.1);
- How to create a backup of the database and how to migrate data from one database to another (Section C.3.2);
- How to access the server logs (Section C.3.3);
- How to run DD-IMS unit tests (Section C.4);
- How to add more tests to the DD-IMS test suite (Section C.4).

C.1 Setup

This section describes the setup of the DD-IMS, including the software used, information about the git repository, and the deployment.

The DD-IMS is developed in Django, a Python web framework. The Database Management System used for setting up and hosting the database is PostgreSQL. The DD-IMS communicates with the Fenix system to take advantage of the data that can be retrieved through the Fenix API[1].

The source code of the DD-IMS project is available at the RNL Gitea[2] in the DEI/PhDMS repository. To access the RNL Gitea, it is obligatory to be connected to the IST network or to be using the IST VPN service.

[1] http://fenixedu.org/dev/api/
[2] https://ark.rnl.tecnico.ulisboa.pt/

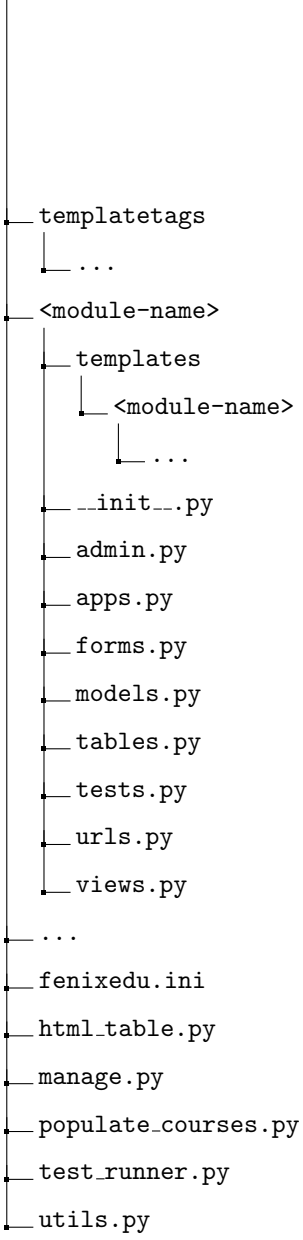
Regarding deployment, the DD-IMS web server is currently running on a virtual machine of the RNL, accessible at this URL: `https://phdisvm.rnl.tecnico.ulisboa.pt/`. The DD-IMS database is deployed on the IST PostgreSQL database server[3]. The IST IT Services make available a PostgreSQL database server to all users registered in Fenix. Currently, the database server is associated with my student account (ist194058).

C.2 Project Structure

This section describes the structure of the DD-IMS project. The DD-IMS software is organized in the following Python modules and files:

```
/PhDMS
├── .gitignore
├── requirements.txt
└── deic
    ├── deic
    │   ├── __init__.py
    │   ├── settings
    │   │   ├── __init__.py
    │   │   ├── common.py
    │   │   ├── dev.py
    │   │   └── prod.py
    │   ├── urls.py
    │   └── wsgi.py
    ├── media
    │   └── ...
    ├── static
    │   ├── css
    │   │   └── ...
    │   ├── img
    │   │   └── ...
    │   └── js
    │       └── ...
    └── templates
        ├── emails
        │   └── ...
        ├── includes
        │   └── ...
        └── ...
```

[3]https://suporte.dsi.tecnico.ulisboa.pt/manual-do-utilizador/base-de-dados-postgresql



The **PhDMS** folder is the root of the repository. It contains the **.gitignore** file which specifies the files that should not be committed to the repository (e.g. IDE files, configuration files that must be kept secret), the **requirements.txt** file which lists the python packages that are required to run the Django project, and the **deic** folder. The **deic** folder is a container for the Django project. The sub-folders and files of this folder will be described in the paragraphs below.

The inner **deic** folder contains the following files/sub-folders:

- **__init__.py**: this type of files are empty files that tell Python that the folder is a Python package.
- **settings**[4]: this folder contains all the project's configuration. The production and development settings are placed in two different files: **dev.py** and **prod.py**, respectively. The **common.py** file contains the settings that are common for the two environments. Finally, the **__init__.py** file contains information that must be kept secret (e.g. database password), so it is not committed to the repository
- **urls.py**[5]: the main URL file. This file is responsible for mapping the Web Application URL paths to Python functions. This file points to each module **urls.py** file so that the entire list of URLs does

[4]https://docs.djangoproject.com/en/3.1/topics/settings/
[5]https://docs.djangoproject.com/en/3.1/topics/http/urls/

not need to be specified in this file (avoids the main URL file from getting too big to read).

- **wsgi.py**[6]: this file is a gateway interface used for deployment.

The **media** folder contains the media files[7] uploaded to the system. The **static** folder contains the website static files (additional files, such as images, JavaScript files or CSS files).

The **templates** folder contains the HTML templates that are used by more than one of the project modules (e.g. footer and navbar of the DD-IMS webpage are the same for all views). This folder also contains the body for each type of e-mail that is sent from the DD-IMS. The **templatetags** folder contains custom template tags and filters[8]. The goal of these tags and filters is to extend the Django template engine[9] functionalities.

For each module, identified by ¡module-name¿, the following files/sub-folders are contained in the corresponding folder:

- **templates**: this folder contains the HTML templates that are used only by the module views (a view is explained in **views.py**)
- **__init__.py**: empty file.
- **admin.py**: this file allows developers to register the database models (explained in **models.py**) with the Django Admin, an administrative tool that allows the system administrators to quickly add, delete, or edit any database model from a web interface (accessible via /admin path).
- **apps.py**: the configuration file of the module.
- **forms.py**: this file contains the forms created.
- **models.py**: this file contains the database models. The model classes defined in this file are translated automatically by Django into database tables.
- **tables.py**: this file contains auxiliary functions that create the data (headers and row values) of the tables that are displayed by the UI.
- **tests.py**: this file contains the unit tests. If there is a large amount of code for the unit tests, this file should be split into multiple files. In that case, a **tests** folder should be created, and the **__init__.py** file should be added inside, as well as the separated test files.
- **urls.py**: this file is responsible for mapping the routes of the module.
- **views.py**[10]: this file contains the functions that handle the HTTP requests sent from users and return an HTTP response, thus they contain the business logic necessary to return the response. If there is a large amount of code for the views, this file should be split into multiple files. In that case, a **views** folder should be created, and the **__init__.py** file should be added inside, as well as the separated view files.

[6] https://docs.djangoproject.com/en/3.1/howto/deployment/wsgi/
[7] https://docs.djangoproject.com/en/3.1/topics/files/
[8] https://docs.djangoproject.com/en/3.1/ref/templates/builtins/
[9] https://docs.djangoproject.com/en/3.1/topics/templates/
[10] https://docs.djangoproject.com/en/3.1/topics/http/views/

The DD-IMS project contains the following modules:

- **alerts**: contains the functionalities offered by the Alerts tab;
- **application_eval**: contains the functionalities offered by the Evaluation of Applications tab;
- **coordinator**: contains the functionalities offered by the coordinator's All Students tab;
- **cse_info**: contains the functionalities offered by the CSE PhD Courses and CSE Faculty tabs;
- **django_fenix_auth**: contains the functionalities regarding the authentication via IST Central Authentication Service (CAS) and the access to the Fenix API endpoints;
- **enroll**: contains the functionalities offered by the Registration tab;
- **juries**: contains the functionalities offered by the Evaluation of Juries tab;
- **management**: contains the functionalities offered by the Management tab;
- **staff**: contains the functionalities offered by the administrative staff members' All Students tab;
- **studyplan**: contains the functionalities offered by the Study Plan tab;
- **supervision**: contains the functionalities offered by the Supervision tab;
- **supervisor**: contains the functionalities offered by the My Students tab;
- **thesis_defense**: contains the functionalities offered by the Thesis tab;
- **thesis_proposal**: contains the functionalities offered by the Thesis Proposal tab;
- **users**: contains the database models and the functionalities offered by the Home tab.

The outer **deic** folder also contains the following files:

- **fenixedu.ini**: this file contains the FenixEdu configuration (i.e. Client ID, Client Secret, Redirect Uri). This file is not committed to the repository as it contains information that must be kept secret.
- **html_table.py**: this contains the classes that represent the different elements of an HTML table. These classes are instantiated when the table data is created (on **tables.py**).
- **manage.py**: a command-line utility that is used to run management commands related to the project (e.g. run tests, create migrations).
- **populate_courses.py**: this file contains the code for loading the CSE courses list from Fenix. It can also be executed as a script (passing the semester as an argument).
- **text_runner.py**: this file contains the configuration for the unit tests execution.
- **utils.py**: this file contains a set of helper functions and classes.

C.2.1 Adding a new module to DD-IMS

A new module can be added to DD-IMS with execution of the following command (in the Django project root, **deic**/):

```
$ django-admin startapp <module-name>
```

This command creates a new folder with the module name. Inside this folder, some of the files mentioned above are automatically generated: **admin.py**, **apps.py**, **models.py**, **tests.py**, and **views.py**. The other files/folders must be created manually.

After creating the module, it is necessary to configure the project to use the new module. This is done in the **common.py** file inside **deic**/**settings**/, by adding the module name to the *INSTALLED_APPS* list.

C.3 Deployment

This section describes how to deploy DD-IMS to a production server. The following steps must be executed:

1. Connect to the remote server via ssh:

```
> ssh root@<remote-server>
```

2. Update the server and install the tools that will be required for the deployment:

   - **Update the system**:

```
$ sudo apt-get update
$ sudo apt-get -y upgrade
```

   - **Install Python**:

```
$ sudo add-apt-repository ppa:deadsnakes/ppa
$ sudo apt-get update
$ sudo apt-get install python3.7
```

   - **Install NGINX**[11]:

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

   - **Install Virtualenv**:

```
$ wget https://bootstrap.pypa.io/get-pip.py
$ sudo python3.7 get-pip.py
$ sudo pip3.7 install virtualenv
```

   - **Download deb**[12] **file from** `https://github.com/wkhtmltopdf/wkhtmltopdf/releases` **(version 0.12.5) and install wkhtmltopdf**.

[11] https://nginx.org/en/
[12] https://fileinfo.com/extension/deb

3. Create an application user named ”phdis” and enter a password. Some extra information will be requested (i.e. full name, phone), but these extra fields can be set as default by pressing ENTER. After creating the user, add the user to the sudoers list:

```
$ adduser phdis
$ gpasswd –a phdis sudo
```

4. Switch to the application user and print the path of the current folder to check if the current folder is correct:

```
$ sudo su – phdis
$ pwd
  /home/phdis
```

5. Clone the git repository (before this, Git[13] must be installed on the virtual machine):

```
$ git clone <repository>
```

6. Start a virtual environment:

```
$ virtualenv venv –p python3.7
```

7. Initialize the virtual environment:

```
$ source venv/bin/activate
```

8. Install the requirements:

```
$ pip install –r PhDMS/requirements.txt
```

9. Add these two extra libraries:

```
$ pip install gunicorn
$ pip install psycopg2
```

10. Move to the Django project root folder:

```
$ cd PhDMS/deic
```

11. Create the FenixEdu configuration file, **fenixedu.ini**, with the following structure:

```
[fenixedu]
client_id=<CLIENT_ID>
client_secret=<CLIENT_SECRET>
redirect_uri=<REDIRECT_URI>
logout_url=<LOGOUT_URL>

[DEFAULT]
```

[13] https://git-scm.com/downloads

```
base_url=https://fenix.tecnico.ulisboa.pt/
login_url=https://fenix.tecnico.ulisboa.pt/oauth/userdialog?client_id={}&
    redirect_uri={}
access_token_url=https://fenix.tecnico.ulisboa.pt/oauth/access_token
api_endpoint=api/fenix/
api_version=1
```

12. Create the **__init__.py** file inside /**deic**/**settings**/ folder. Inside this file, import the production settings and insert the information that must be kept secret (e.g. database password), as shown in the example below:

```
from __future__ import absolute_import
from .prod import *  # change to .dev when you are in development environment

##### DJANGO SECRETS
SECRET_KEY = <DJANGO_APPLICATION_SECRET_KEY>
DATABASES['default']['PASSWORD'] = <DATABASE_PASSWORD>
```

13. Migrate the DD-IMS database[14]:

```
$ python manage.py migrate
```

14. Collect the static files[15] (this command copies all the static assets to an external folder where NGINX can serve the files for users):

```
$ python manage.py collectstatic
```

15. Configure Gunicorn[16]:

    - Create a script file named **gunicorn_start** inside /**home**/**phdis** (this script will start the application server):

```
#!/bin/bash

NAME="PhD_Information_System"
DIR=/home/phdis/PhDMS/deic
USER=phdis
GROUP=phdis
WORKERS=5
BIND=unix:/home/phdis/run/gunicorn.sock
DJANGO_SETTINGS_MODULE=deic.settings
DJANGO_WSGI_MODULE=deic.wsgi
LOG_LEVEL=DEBUG

export LC_ALL=en_US.UTF8
```

[14] https://docs.djangoproject.com/en/3.1/topics/migrations/
[15] https://docs.djangoproject.com/en/3.1/howto/static-files/
[16] https://gunicorn.org/

```
cd $DIR
source ../../venv/bin/activate

export DJANGO_SETTINGS_MODULE=$DJANGO_SETTINGS_MODULE
export PYTHONPATH=$DIR:$PYTHONPATH

exec ../../venv/bin/gunicorn ${DJANGO_WSGI_MODULE}:application \
--name $NAME \
--workers $WORKERS \
--user=$USER \
--group=$GROUP \
--bind=$BIND \
--log-level=$LOG_LEVEL \
--log-file=-
```

- Make the file executable:

```
$ chmod u+x gunicorn_start
```

- Create two empty folders, one for the socket file and one to store the logs:

```
$ mkdir run logs
```

16. Configure Supervisor[17]:

- Create an empty log file inside /**home**/**phdis**/**logs**/:

```
$ touch logs/gunicorn.log
```

- Create the Supervisor file:

```
$ sudo vi /etc/supervisor/conf.d/phdis.conf

[program:phdis]
command=/home/phdis/gunicorn_start
user=phdis
autostart=true
autorestart=true
redirect_stderr=true
stdout_logfile=/home/phdis/logs/gunicorn.log
environment=LANG=en_CA.UTF-8,LC_ALL=en_CA.UTF-8,LC_LANG=en_CA.UTF-8
```

- Save the file and reload the Supervisor process:

```
$ sudo supervisorctl reread
$ sudo supervisorctl update
```

- Check if everything is ok with the Supervisor process:

```
$ sudo supervisorctl status phdis
  phdis            RUNNING   pid 7739, uptime 0:00:05
```

[17] http://supervisord.org/

17. Configure NGINX:

- Create the NGINX configuration file named **phdis** inside /**etc**/**nginx**/**sites-available**/:

```
upstream app_server {
    server unix:/home/phdis/run/gunicorn.sock fail_timeout=0;
}

server {
    server_name phdisvm.rnl.tecnico.ulisboa.pt;  # here can also be the
        IP address of the server

    keepalive_timeout 5;
    client_max_body_size 4G;

    access_log /home/phdis/logs/nginx-access.log;
    error_log /home/phdis/logs/nginx-error.log;

    location /static/ {
        alias /home/phdis/staticfiles/;
    }

    # checks for static file, if not found proxy to app
    location / {
        try_files $uri @proxy_to_app;
    }

    location @proxy_to_app {
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header Host $http_host;
        proxy_redirect off;
        proxy_pass http://app_server;
    }
}
```

- Create a link of the **phdis** file to the **sites-enabled** folder:

```
$ sudo ln -s /etc/nginx/sites-available/phdis /etc/nginx/sites-enabled/
    phdis
```

- Remove the default NGINX website:

```
$ sudo rm /etc/nginx/sites-enabled/default
```

- Restart NGINX service:

```
$ sudo service nginx restart
```

After following these steps, the DD-IMS website should already be accessible via browser. The website can be protected with an HTTPS certificate. The certificate can be generated using Let's Encrypt[18],

[18] https://letsencrypt.org/

a certificate authority run that provides X.509 certificates for Transport Layer Security (TLS) encryption at no charge. The certificates are valid for 90 days and are automatically renewed.

C.3.1 Update the system

The production server must be updated when there are changes regarding the database schema or the DD-IMS functionalities. This section contains the instructions on how to perform an update.

Assuming that the system administrator has already initialized the virtual environment (venv) and the last version of the source code was already pulled from the repository, the following commands must be executed to update the DD-IMS in the production server:

- Switch to the application user and move to the Django project root:

```
$ sudo su - phdis
$ cd PhDMS/deic
```

- Collect the static files for NGINX (only needs to be executed if the static files have been modified):

```
$ python manage.py collectstatic
```

- Update the database schema and apply the database migration process (only need to be executed if the database models have been modified):

```
$ python manage.py makemigrations
$ python manage.py migrate --run-syncdb
```

- Restart Supervisor and NGINX services:

```
$ sudo service supervisor restart
$ sudo service nginx restart
```

C.3.2 Create data backups / Migrate data from one database to another

This section contains the instructions on how to create data backups and migrate data from one database to another. It is crucial to create data backups periodically to keep data secure and protect the DD-IMS against data loss, which can be caused by a database server crash. Regarding the data migration from one database to another, there can be many reasons for data migrations, including data volume growth, performance requirements, or database maintenance.

To change the DD-IMS database server and migrate the data to a new database, the following steps must be followed:

- Export all data in the database to a JSON file. This command[19] should be used not only to migrate the data to a new database but also to create backups of the database.

[19] https://docs.djangoproject.com/en/3.1/ref/django-admin/#dumpdata

```
$ python manage.py dumpdata --natural-foreign --exclude auth.permission --exclude
    admin.logentry --exclude contenttypes --exclude sessions --indent 4 > <
    filename>.json
```

- Modify the database in project settings in **dev.py** and the database password in **__init__.py**.
- Load the data into the new database[20]:

```
$ python manage.py migrate
$ python manage.py loaddata <filename>.json
```

C.3.3 Logs

The system logs are accessible in the virtual machine in **/home/phdis/logs/** folder. Here, four log files can be accessed: (i) **nginx-access.log** and **nginx-error.log**, the nginx logs; (ii) **gunicorn.log**, the gunicorn logs; and (iii) **debug.log**, the Django logs which allows developers to visualize user actions or possible bugs.

In what concerns the Django logs, the logging[21] module is used. The logging configurations must be set on the project settings.

C.4 Unit Tests

To run the DD-IMS unit tests[22], the following command is used:

```
$ python manage.py test
```

It is also possible to run only the unit tests of a module, with the following command:

```
$ python manage.py test <module-name>
```

To add more tests to the DD-IMS, it is necessary to create a new tests file inside the **tests/** folder of the module that will be tested and then write the tests code.

[20] https://docs.djangoproject.com/en/3.1/ref/django-admin/#loaddata
[21] https://docs.python.org/3/library/logging.html
[22] https://docs.djangoproject.com/en/3.1/topics/testing/overview/

Appendix D

Usability Evaluation - Tasks

This appendix contains the tasks that were assigned to each type of user during the DD-IMS evaluation sessions. The following types of tasks are listed below in this chapter:

- Student Tasks (Section D.1);
- Supervisor Tasks (Section D.2);
- Coordinator Tasks (Section D.3);
- CC3C Member Tasks (Section D.4);
- CCP Member Tasks (Section D.5);

D.1 Student Tasks

1) **Registration**

   a) Register in the application with the following information:

   - **Phone**: +351930001122
   - **Identity Number**: 43215678
   - **Fiscal Number**: 45344444
   - **Address**: Rua BB
   - **City**: Lisboa
   - **Zip Code**: 1234-111
   - **Country**: Portugal
   - **Previous Degree 1**: Licenciatura em Engenharia Eletrotécnica, a Bachelor Degree (3 years) on Faculdade de Ciencias in Portugal, concluded in 2014 with grade 17. Scientific Domain is Computer Science

- **Previous Degree 2**: Mestrado em Engenharia Informática e de Computadores, a Master Degree on Instituto Superior Técnico in Portugal, concluded in 2018 with grade 16. Scientific Domain is Computer Science
- **Student Number**: 99999
- **Doctoral Program Type**: CSE Doctoral Degree
- **Research Unit**: INESC-ID
- **Financing Source**: State type from Fundação para a Ciência e a Tecnologia

b) Find the start date of your PhD.

2) **Study Plan**

a) Submit your study plan (your study plan academic year is 2019/20) with the following courses:

- **Outreach and Teaching Skills - dei** - Recommended Course in the first semester
- **Interdisciplinary Research** - Recommended Course
- **Boolean Constraints and Optimization** - Optional Course
- **Test Free Course (TFC)** - a Free Course from department DEEC in Instituto Superior Tecnico, with 4.5 ECTS, given in the Second Semester
- **Test Prop Course (TPC)**: a Propaedeutic Course from the degree Mestrado em Engenharia Informática e de Computadores, given in the First Semester

b) In your study plan, find the number of ECTS of the course ”Research Topics”.

c) Access the webpage of the ”Research Topics” course.

3) **Supervision**

a) Submit your supervision team:

- **Barbara Duarte** - your Faculty Supervisor
- **Non Faculty Test 1** - one of your Non Faculty Co-Supervisors from Company ”Test Company” with job ”Project Manager”, whose email is nfsup1@gmail.com. Your justification is the following: ”Because test 1.”
- **Non Faculty Test 2** - the other Non Faculty Co-Supervisor from Company ”Another Test Company” with job ”Software Engineer”, whose email is nfsuptest2@yahoo.com. Your justification is the following: ”Because test 2.”

4) **CSE Faculty**

a) Find the scientific area of faculty member ”José Alberto Rodrigues Pereira Sardinha”.

5) **CSE Courses**

a) Find the number of ECTS and semester of the course ”High Performance Computing”.

6) **Thinking Aloud phase** - in this phase, we ask you to wait a few seconds so that we can undo the data that you inserted during the previous tasks. After that, please insert your correct information related to your personal and academic information, supervision team, and study plan. During this phase, we ask you to use the system while continuously thinking out loud, giving your feedback as you move through the DD-IMS user interface.

D.2 Supervisor Tasks

1) **Students Information**

   a) For student Jonny Anthony, find his student number.

2) **Supervision**

   a) For student Jonny Anthony, approve the supervision team.

3) **Study Plan**

   a) For student Jonny Anthony, find the optional courses selected in the student's study plan.

   b) For student Jonny Anthony, decline the study plan with the following justification: "Decline study plan test".

4) **Thesis Proposal (CAT)**

   a) For student Emmanuella Mcbride, submit the thesis proposal jury with the following information:

      - **Thesis Title**: Test title
      - **Thesis Abstract**: This is not a thesis. It is a test.
      - **Jury Member and President of the Jury**: Faculty member Barbara Duarte from IST CSE Department, with the following justification: "She is a dummy user used for testing."

   b) For student Marcos Miranda, submit the thesis proposal minutes with the following information:

      - **Result**: Approved
      - **Comments**: Overall, a very good work.
      - **Problem description and innovation potential**: Outstanding
      - **Quality of state of art survey**: Very Good
      - **Proposed approach and methodology**: Outstanding
      - **Activities planning**: Good
      - **Oral presentation and discussion**: Very Good

5) **CSE Faculty**

   a) Find the scientific area of faculty member "José Alberto Rodrigues Pereira Sardinha".

6) **CSE Courses**

   a) Find the number of ECTS and semester of the course ”High Performance Computing”.

7) **Thinking Aloud phase** - please verify your real students' supervision and study plan and approve them if everything is correct, and submit your students' thesis proposal (CAT) jury. During this phase, we ask you to use the system while continuously thinking out loud, giving your feedback as you move through the DD-IMS user interface.

D.3 Coordinator Tasks

1) **Students Information**

   a) Find the student Jonny Anthony's Master Degree grade.

2) **Study Plan**

   a) For student Jonny Anthony, find the number of prior credits requested and download the prior credits documentation file.

   b) In student Jonny Anthony's study plan, do the following actions: (i) add 6 ECTS as equivalence credits obtained from Test University, with description ”Test Equivalences”; (ii) Set the prior credits for previous training to 3.0 ECTS; and (iii) Approve the study plan.

3) **Supervision**

   a) Decline the student Jonny Anthony's supervision with the following justification: ”Declined because Testing”.

4) **Statistics**

   a) Find the following statistics: (i) Number of candidates admitted on the Winter semester of 2020/21; (ii) Number of students that concluded their Doctoral Program in 2019/20; and (iii) Number of foreign students active in 2020/21.

5) **Management**

   a) Add faculty Barbara Duarte to the CCP members list.

   b) Find the total number of faculty members whose scientific area is ”IA”.

   c) Set the scientific area of the course ”Advanced Topics in Cybersecurity” to ASO.

6) **Thesis Proposal Jury (CAT)**

   a) For student Emmanuella Mcbride, submit the final decision of the CC3C evaluation about the jury: Decline with the following justification ”This is a test”.

7) **Thesis Jury**

a) For student Marcos Miranda, submit the final decision of the CCP evaluation: Approve.

8) **Evaluation of Applications**

a) For candidate Alberto Castro, submit the evaluation result ”Approved with Propaedeutic Courses” with the justification ”Admitted during testing.” and the following list of propaedeutic courses recommended: (i) Artificial Intelligence; (ii) Computer Graphics.

b) Create the official document to be sent to Post-Graduate Area (PGA) with reference ”Test Letter”, that contains the application results of Daniela Rosario, Alberto Castro, and Aiysha Cleveland.

c) Generate the PDF of the document created in 8-b).

D.4 CC3C Member Tasks

1) **Evaluation of Applications**

a) For candidate Daniela Rosario’s application, find who are her supervisors.

b) Give the following feedback about the candidate Daniela Rosario’s application: ”This is a test”.

2) **Thesis Proposal Jury (CAT)**

a) Find the president of student Jonny Anthony’s thesis proposal jury.

b) Give the following feedback about the student Jonny Anthony’s thesis proposal jury: ”This is a test”.

3) **Thesis Jury**

a) For student Emmanuella Mcbride’s thesis defense jury, find the feedback given by the CC3C member Barbara Duarte.

b) Give the following feedback about the student Emmanuella Mcbride’s thesis defense jury: ”This is a test”.

D.5 CCP Member Tasks

1) **Thesis Proposal Jury (CAT)**

a) In the student Jonny Anthony’s thesis proposal jury, find the feedback given by CCP Member Barbara Duarte.

b) Give the following feedback about the student Jonny Anthony’s thesis proposal jury: ”This is a test”.

2) **Thesis Jury**

a) Find the rapporteurs of the student Emmanuella Mcbride's thesis defense jury.

b) Give the following feedback about the student Emmanuella Mcbride's thesis defense jury: "This is a test".

Appendix E

Usability Evaluation - Satisfaction Questionnaire

This appendix presents the satisfaction questionnaire used on the DD-IMS usability validation. The questionnaire is shown below.

Questionnaire on the usability of the CSE Doctoral Degree Information Management System (DD-IMS)

Thank you for taking part in the evaluation of the CSE Doctoral Degree Information Management System.
Before concluding this session, we still need you to answer the following questions.
* Required

1. Gender *

Mark only one oval.

- ◯ Female
- ◯ Male
- ◯ Prefer not to say
- ◯ Other: _____

2. Age

_____

3. Which task did you execute in DD-IMS usability evaluation? *

Mark only one oval.

- ◯ Student tasks *Skip to question 4*
- ◯ Supervisor tasks *Skip to question 6*
- ◯ Coordinator tasks *Skip to question 7*
- ◯ Administrative Staff tasks *Skip to question 8*
- ◯ CC3C Member tasks *Skip to question 9*
- ◯ CCP Member tasks *Skip to question 10*

4. Which modules of the DD-IMS will you use more often? *

Check all that apply.

Check all that apply.

- [ ] Registration in the Application
- [ ] Supervision
- [ ] Study Plan
- [ ] Curricular Plan
- [ ] Thesis Proposal Jury and Defense
- [ ] Thesis Jury

5. In which curricular year did you define your study plan? *

Mark only one oval.

- ◯ 2015/16
- ◯ 2016/17
- ◯ 2017/18
- ◯ 2018/19 or Later

Skip to question 11

6. Which modules of the DD-IMS will you use more often? *

Check all that apply.

Check all that apply.

- [ ] Supervision
- [ ] Study Plan
- [ ] Curricular Plan
- [ ] Thesis Proposal Jury and Defense
- [ ] Thesis Jury

Skip to question 11

7. Which modules of the DD-IMS will you use more often? *

Check all that apply.

Check all that apply.

- [ ] Supervision
- [ ] Study Plan
- [ ] Curricular Plan
- [ ] Thesis Proposal Jury and Defense
- [ ] Thesis Jury
- [ ] Statistics
- [ ] Management
- [ ] Evaluation of Applications

Skip to question 11

8. Which modules of the DD-IMS will you use more often? *

Check all that apply.

Check all that apply.

- [ ] Supervision
- [ ] Study Plan
- [ ] Curricular Plan
- [ ] Thesis Proposal Jury and Defense
- [ ] Thesis Jury
- [ ] Statistics
- [ ] Management
- [ ] Evaluation of Applications

Skip to question 11

9. Which modules of the DD-IMS will you use more often? *

Check all that apply.

Check all that apply.

- [ ] Supervision
- [ ] Study Plan
- [ ] Curricular Plan
- [ ] Thesis Proposal Jury and Defense
- [ ] Thesis Jury
- [ ] Statistics
- [ ] Management
- [ ] Evaluation of Applications

Skip to question 11

10. Which modules of the DD-IMS will you use more often? *

Check all that apply.

Check all that apply.

- [ ] Registration in the Application
- [ ] Supervision
- [ ] Study Plan
- [ ] Curricular Plan
- [ ] Thesis Proposal Jury and Defense
- [ ] Thesis Jury
- [ ] Statistics
- [ ] Management

Skip to question 11

A series of 10 sentences will be provided below. For each one, please indicate your level of agreement.

All items should be checked. If you feel that none of them cannot respond to a particular item, please mark the centre point of the scale (3).

11. I think that I would like to use this system frequently. *

Mark only one oval.

Mark only one oval.

	1	2	3	4	5	
Strongly Disagree						Strongly Agree

12. I found the system unnecessarily complex. *

Mark only one oval.

Mark only one oval.

	1	2	3	4	5	
Strongly Disagree						Strongly Agree

13. I thought the system was easy to use. *

Mark only one oval.

Mark only one oval.

	1	2	3	4	5	
Strongly Disagree						Strongly Agree

14. I think that I would need the support of a technical person to be able to use this system. *

Mark only one oval.

Mark only one oval.

	1	2	3	4	5	
Strongly Disagree						Strongly Agree

15. I found the various functionalities in this system were well integrated. *

Mark only one oval.

Mark only one oval.

	1	2	3	4	5	
Strongly Disagree						Strongly Agree

16. I thought there was too much inconsistency in this system. *

Mark only one oval.

Mark only one oval.

	1	2	3	4	5	
Strongly Disagree						Strongly Agree

17. I would imagine that most people would learn to use this system very quickly. *

Mark only one oval.

Mark only one oval.

	1	2	3	4	5	
Strongly Disagree						Strongly Agree

18. I found the system very cumbersome to use. *

Mark only one oval.

Mark only one oval.

	1	2	3	4	5	
Strongly Disagree						Strongly Agree

19. I felt very confident using the system. *

Mark only one oval.

Mark only one oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

20. I needed to learn a lot of things before I could get going with this system. *

Mark only one oval.

Mark only one oval.

	1	2	3	4	5	
Strongly Disagree	◯	◯	◯	◯	◯	Strongly Agree

Additional Comments

If you have any additional comments you'd like to share, please use the following text box.

21.