A Database for CMU Portugal

Maria Inês de Magalhães Torres Queiroz de Morais

Thesis to obtain the Master of Science Degree in

Computer Science and Engineering

Supervisors: Prof. Maria Inês Camarate de Campos Lynce de Faria
Dr. Sílvia Manuela Azevedo de Castro

Examination Committee

Chairperson: Prof. Alberto Abad Gareta
Supervisor: Prof. Maria Inês Camarate de Campos Lynce de Faria
Member of the Committee: Prof. Helena Isabel De Jesus Galhardas

June 2023

Declaration
I declare that this document is an original work of my own authorship and that it fulfills all the requirements of the Code of Conduct and Good Practices of the Universidade de Lisboa.

Acknowledgments

I would like to thank my parents for their friendship, patience, encouragement, and caring, for always being there, and without whom this project would not be possible.

I would like to thank Prof. Inês Lynce and Dr. Sílvia Castro for giving me the opportunity to do my dissertation in an area of my interest. I would also like to thank them for their guidance, orientation, and support throughout this process.

I would also like to thank João Fumega from the CMU Portugal team.

I would also like to acknowledge Prof. Helena Galhardas for her clarification of some project-related uncertainties and for her insightful feedback.

Special thanks to João Antunes for being there through the good and the bad over all these years.

Finally, I would like to thank my friends and colleagues who had the kindness to help me in times of need.

This work was supported by national funds through Fundação para a Ciência e a Tecnologia (Portuguese Foundation for Science and Technology), under project UIDB/50021/2020, and through the ”Carnegie Mellon Portugal Program”.

Abstract

From the perspective of business owners or managers, the challenges of data storage and organization can be overwhelming. With the rapid advancement of technology and the increasing reliance on digital information, businesses need to efficiently handle large volumes of data. With this evolution, it makes sense to invest in a data management system. Concerning this topic, the basis for our research problem focuses on creating a database for Carnegie Mellon Portugal Program (CMU Portugal) and a user interface so users can insert, alter, view, and remove records from the database without being forced to use SQL queries. At the moment, CMU Portugal's data sources consist of various Excel tables and CMU Portugal's website. As such, the process of developing a database and user interface for CMU Portugal requires the development of the database model, the profiling, cleaning, and migration of the current CMU Portugal's data to the database. For this process, we used SQL to build the database, MySQL as the database management system, the Pentaho Data Integration tool to profile, clean, and migrate the data, and HTML, CSS, and Javascript to build the user interface, along with PHP as the server-side scripting language to interact with the database. The implementation of this system is further described in this document. Additionally, the system was evaluated based on the quality of the data after its migration to the database, the performance of the database, and the user-friendliness of the user interface. Conclusions and considerations for future work are taken at the end.

Keywords

CMU Portugal; Database Design; Data Profiling; Data Cleaning; Data Migration; User interface development.

Resumo

Do ponto de vista de proprietários ou gestores de negócios, os desafios de armazenamento e organização de dados e a crescente dependência de informação na forma digital, leva à necessidade de lidar eficazmente com grandes volumes de dados, sendo que, faz sentido investir num sistema de gestão de dados. Posto isto, o nosso projeto centra-se na criação de uma base de dados para a Carnegie Mellon Portugal Program (CMU Portugal) e de uma interface para que os utilizadores possam interagir com a base de dados. De momento, as fontes de dados da CMU Portugal consistem em várias tabelas Excel e no website da CMU Portugal. Como tal, o nosso projeto requer o desenvolvimento de um modelo da base de dados, a análise, a limpeza e a migração dos dados atuais da CMU Portugal para a base de dados e a elaboração da interface do utilizador. Para este processo, utilizámos SQL para construir a base de dados, MySQL como sistema de gestão de base de dados, a ferramenta Pentaho Data Integration para analisar, limpar e migrar os dados, e HTML, CSS e Javascript para construir a interface do utilizador, juntamente com PHP como a linguagem para interagir com a base de dados. A implementação deste sistema é descrita mais pormenorizadamente no presente documento. Além disso, o sistema foi avaliado com base na qualidade migração dos dados para a base de dados, no desempenho da mesma e na facilidade de utilização da interface do utilizador. No final, são apresentadas conclusões e considerações para o trabalho futuro.

Palavras Chave

CMU Portugal; Concepção de Bases de Dados; Análise de Dados; Limpeza de Dados; Migração de Dados; Interface para o utilizador.

Contents

1 Introduction 1

1.1 Motivation . . . 1

1.2 Problem definition . . . 2

1.3 Contributions . . . 2

1.4 Document Organization . . . 2

2 About CMU Portugal 5

2.1 Talent development . . . 6

2.2 Knowledge creation . . . 7

2.3 Innovation and entrepreneurship . . . 7

2.4 Communication and outreach . . . 8

3 CMU Portugal's data and sources 9

3.1 CMU Portugal Data Overview . . . 9

3.1.1 Students . . . 10

3.1.2 Visitors . . . 11

3.1.3 Faculty and Researchers . . . 11

3.1.4 Projects . . . 11

3.2 Data Sources . . . 12

3.2.1 Excel Files . . . 12

3.2.2 CMU Portugal Website . . . 16

3.3 Data quality problems . . . 17

4 Related Work 19

4.1 Database Management System . . . 19

4.1.1 PostgreSQL . . . 20

4.1.2 Oracle Database . . . 20

4.1.3 MySQL . . . 21

4.1.4 Discussion . . . 21

4.2 Data Analysis and Integration . . . 22

4.2.1 Data Profiling . . . 22
4.2.2 ETL . . . 22
4.2.3 Tools Discussion . . . 23
4.3 Database user interface . . . 24
4.3.1 PHP . . . 24
4.3.2 Java Server Pages . . . 25
4.3.3 Python and the Django Framework . . . 25
4.3.4 Discussion . . . 25

5 Database and User Interface 27
5.1 Requirements Gathering . . . 27
5.2 Database Model . . . 29
5.2.1 Entity-Relationship Model . . . 30
5.2.2 Relational Model . . . 36
5.3 Data Profiling, Cleaning, and Migration . . . 40
5.3.1 Data Profiling . . . 40
5.3.2 Data Cleaning . . . 47
5.3.3 Data Migration . . . 48
5.4 User interface . . . 51
5.4.1 Search students . . . 51
5.4.2 Add Student . . . 52
5.4.3 Update Student . . . 54
5.4.4 Remove Student . . . 54

6 Work Evaluation 55
6.1 Quality Assurance in the Data Migration . . . 55
6.2 Database Performance . . . 57
6.3 User testing . . . 59
6.3.1 Results . . . 60

7 Conclusion 63
7.1 Summary and Conclusions . . . 63
7.2 System Limitations and Future Work . . . 65

Bibliography 66

A SQL code of the database 69

B Database performance's queries 79

C User Testing **85**

C.1 User tasks 85

C.2 Questionnaire 86

C.3 Provided documents 88

List of Figures

2.1 Organization of the 3rd Phase of Carnegie Mellon Portugal Program (CMU Portugal) . . . 6

3.1 Organization of CMU Portugal's data 10

5.1 Entity-Relationship Model of CMU Portugal's database 31

5.2 Process of profiling CMU Portugal's data 40

5.3 Completeness analysis of the "Ph.D. Students" table 41

5.4 Result of the completeness analysis on the table "Ph.D. Students" 41

5.5 Transformation to detect approximate duplicates in the "Ph.D. Students" table 42

5.6 Transformation to detect approximate duplicates in the columns of the table "Ph.D. Students" 42

5.7 Result of the approximate duplicate detection in the column "Program in Carnegie Mellon University (CMU)" of the table "Ph.D. Students" 43

5.8 Result of the approximate duplicate detection in the column "Department in CMU" of the table "Ph.D. Students" 43

5.9 Transformation to filter distinct study programs in Portugal or nationality areas from the table "Ph.D. Students" 44

5.10 Result of the transformation to filter distinct study programs in Portugal from the table "Ph.D. Students" 45

5.11 Result of the transformation to filter distinct nationality areas of students from the table "Ph.D. Students" 45

5.12 Transformation to detect students' advisors in the "Ph.D. Students" table that are not present in the "Faculty" table 45

5.13 Approximate duplicates between of students' advisors' names in the "Ph.D. Students" and "Faculty" tables 46

5.14 Transformation to help clean approximate duplicates of students' advisors' names between the "Ph.D. Students" and "Faculty" tables 47

5.15 Transformation to prepare the "Ph.D. Students" table before migration 48

5.16 Pentaho Data Integration (PDI) job to run the transformations needed to migrate CMU Portugal's data to the database 48

5.17 Home page of the CMU Portugal database user interface 51

5.18 Cropped image of the CMU Portugal's students list 52

5.19 Cropped image of the "Add Student" page of the user interface 53

5.20 Cropped image of the "Remove Student" page of the user interface 54

6.1 Transformation to ensure the data completeness and semantical correctness in the "person" database table 56

List of Tables

3.1 Description of the source Excel files 12

4.1 Summary of the pros and cons of the Database Management Systems (DBMSs) 21

4.2 Key characteristics of Talend and Pentaho 23

6.1 Duration and records retrieved for each query 58

6.2 Duration in minutes of the time taken for each user to complete each task 61

6.3 Average points and percentage given by users to the user interface based on the heuristics 61

Listings

A.1 SQL code of the database for CMU Portugal 69
B.1 Database performance's queries 79

Acronyms

CMU	Carnegie Mellon University
CMU Portugal	Carnegie Mellon Portugal Program
Co-PI	Co-Principal Investigator
CRUP	Conselho de Reitores das Universidades Portuguesas
DBMS	Database Management System
EBP	Early Bird Project
ERI	Entrepreneurial Research Initiative
ERP	Exploratory Research Project
FCT	Fundação para a Ciência e a Tecnologia
ICT	Information and Communication Technologies
JSP	Java Server Pages
LP	Lead Project
LSCRP	Large-Scale Collaborative Research Project
PDI	Pentaho Data Integration
PI	Principal Investigator

1

Introduction

Contents

1.1 Motivation 1

1.2 Problem definition 2

1.3 Contributions 2

1.4 Document Organization 2

1.1 Motivation

More and more, the process of modifying and transforming data is becoming an essential topic in the technology sector. Businesses, companies, and institutions produce large amounts of data that need to be stored and managed in a flexible way. Conversely, databases allow for data storage quickly and easily and their usage in a wide variety of everyday tasks. Database management systems can become crucial in the daily processes of organizations since records can be created, updated, and retrieved quickly with these systems. At an early stage, organizations may turn to Excel files and similar documents to organize their data. However, this solution is not viable in the long term. This is the case with the Carnegie Mellon Portugal Program (CMU Portugal), an international partnership program that has grown over the years to include an ever-expanding network of members as well as educational and mobility initiatives. This expansion has made it hard for CMU Portugal to store all of this new data simply and consistently with the help of Excel files alone.

1.2 Problem definition

We were faced with the challenge of developing a database and database user interface for CMU Portugal. CMU Portugal stores its data in multiple Excel files and has, additionally, a website where the data is supplied by other Excel files or handwritten inputs. Given this high number of different sources to respond to different needs, data quality issues arise. Taking this into consideration, our goal is to, first of all, organize CMU Portugal's data in a clear and concise manner through a database model. Secondly, analyze CMU Portugal's data, outline and correct its quality issues. Finally, we aim to create an SQL database exempt from the data problems mentioned before, that ultimately integrates the various data sources in order to fulfill both its users' needs and act as an integrated source for the website, as well as, a database user interface.

1.3 Contributions

The contributions of the thesis "A database for CMU Portugal" and its interface are:

- The database's Entity-Relationship and Relational model.
- The SQL implementation of the database's code.
- The profiling, cleaning, and migration of the data related to the CMU Portugal's students and institutions from the various existent sources to the database.
- The development of a user interface in which the user can view, filter, modify, and delete student records.

1.4 Document Organization

This document is structured in six additional parts. In the following section 2, we give an overview of the CMU Portugal Program, its structure, main goals, and tools to achieve its mission. In section 3, a background of CMU Portugal's data, its sources, and quality issues are presented. In section 4 we analyze and compare the tools necessary for our work. Section 5 lays out the new CMU Portugal database and its interface, from the structure of the database model to the specifics of the data analysis, cleaning, and migration, as well as the layout of the user interface. The section 6 presents the work's evaluation, and in the end, a small conclusion sums up the work done as well as the system's limitations and future work in section 7.

2

About CMU Portugal

Contents

2.1 Talent development 6

2.2 Knowledge creation 7

2.3 Innovation and entrepreneurship 7

2.4 Communication and outreach 8

CMU Portugal is an international partnership between Carnegie Mellon University (CMU) in Pittsburgh, Pennsylvania, United States, and various Portuguese universities, research institutions, and corporations. This partnership provides a platform for education, research, and innovation, focusing its activities primarily in the area of Information and Communication Technologies (ICT). According to CMU Portugal's annual report [1], the program's activities are supported by the Fundação para a Ciência e a Tecnologia (FCT), sponsored by the Conselho de Reitores das Universidades Portuguesas (CRUP), and co-financed by industry partners and CMU.

The mission of CMU Portugal is, by contributing to an innovative community in close connection with state-of-the-art research, advanced graduate education, and highly innovative companies, to take Portugal to the front lines of research and technological development in the area of ICT.

The program began in 2006 and has been split into three phases so far: the first phase between

2006 and 2012, the second phase between 2012 and 2017, and the third phase that started in 2018 and should end by the year 2030. In order to accomplish its mission, CMU Portugal's third phase of the program includes several instruments, which are described in the following sections and in fig. 2.1.

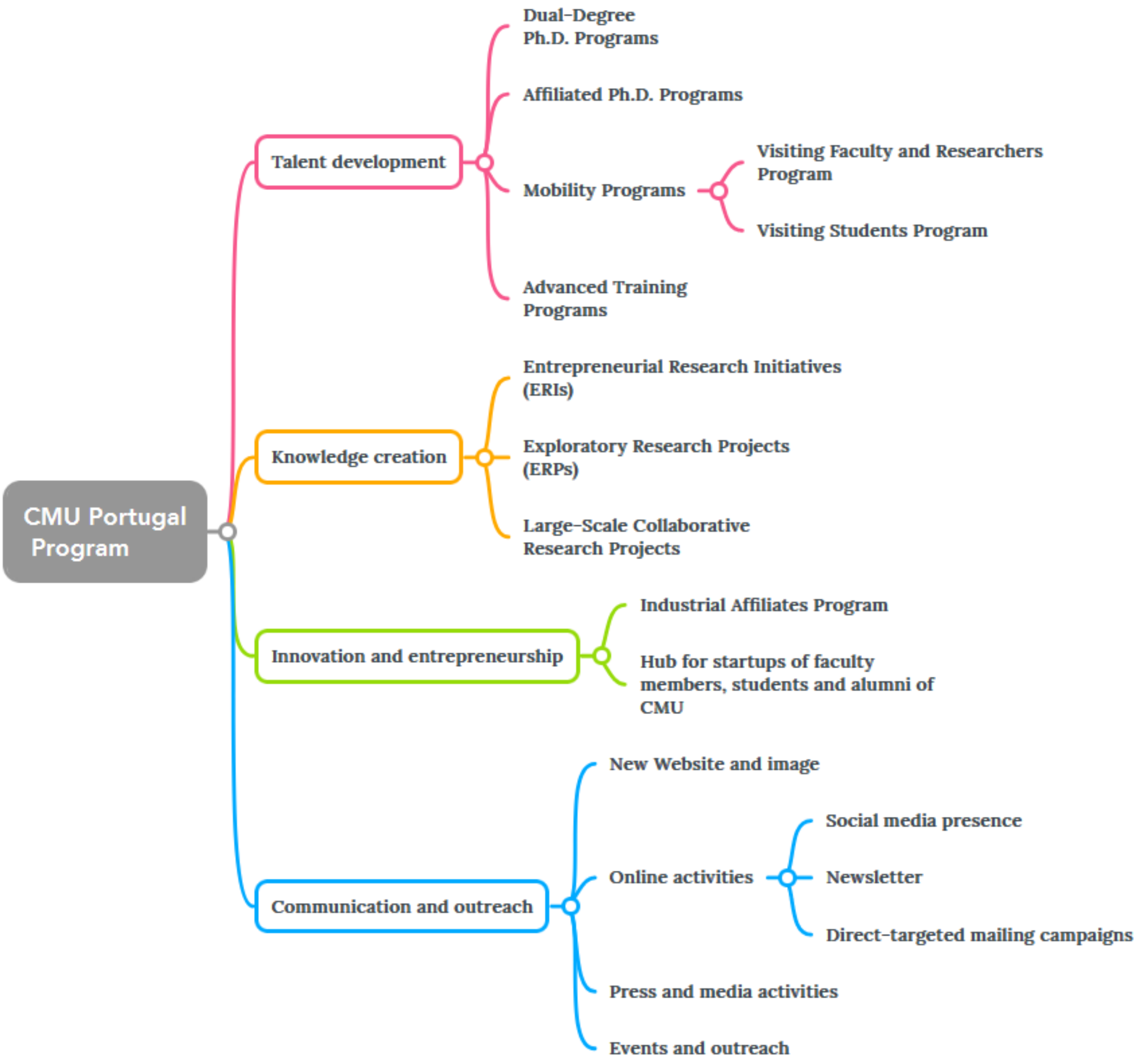

CMU Portugal Program
Talent development
Dual-Degree Ph.D. Programs
Affiliated Ph.D. Programs
Mobility Programs
Visiting Faculty and Researchers Program
Visiting Students Program
Advanced Training Programs
Knowledge creation
Entrepreneurial Research Initiatives (ERIs)
Exploratory Research Projects (ERPs)
Large-Scale Collaborative Research Projects
Innovation and entrepreneurship
Industrial Affiliates Program
Hub for startups of faculty members, students and alumni of CMU
Communication and outreach
New Website and image
Online activities
Social media presence
Newsletter
Direct-targeted mailing campaigns
Press and media activities
Events and outreach

Figure 2.1: Organization of the 3rd Phase of CMU Portugal

2.1 Talent development

As it is possible to see in Fig. 2.1 the initiative of Talent development is divided into three programs:

1. *Dual-Degree Ph.D. Programs* that provide students with education in two universities, at CMU and, at a Portuguese university. Students finish with two Ph.D. degrees, one from each university.

2. *Affiliated Ph.D. Programs* which are fully hosted by a Portuguese university and include a research period at CMU of up to one year. Students finish with a Ph.D. degree awarded by the host Portuguese university.

3. *Mobility Programs* consist of the Visiting Faculty and Researchers Program and the Visiting Students Program. The first program entails the visit of faculty members and researchers to the corresponding host department at CMU, where a collaboration between both parties takes place in research, co-teaching, and other academic activities. The Visiting Students Program enables students to attend courses, participate in research projects and immerse themselves in the CMU community.

4. *Advanced Training Programs* that consist of short and intensive educational programs for professionals in ICT areas.

2.2 Knowledge creation

The Knowledge creation initiative is made up of the following ventures (see Fig. 2.1):

1. *Entrepreneurial Research Initiatives (ERIs)* are science, engineering, management, and policy projects that comprise research teams from two Portuguese universities, one from CMU, and at least one partner company.

2. *Exploratory Research Projects (ERPs)* aim to promote new initiatives with high impact potential that encourage Portugal's international competitiveness and innovation capacity in strategic ICT emerging areas of interest to CMU Portugal.

3. *Large-Scale Collaborative Research Projects (LSCRPs)* are projects with a higher duration that should involve industrial research and experimental development activities to create new products, services, processes, and systems.

2.3 Innovation and entrepreneurship

CMU Portugal wants to foster collaboration between industry and academia, by working on establishing a close relationship with the Portuguese industry through the companies that are part of its Industrial Affiliates Program. The goal is to promote an innovative ecosystem in Portugal and position it as a hub for global innovation and technology.

Besides that, CMU Portugal acts as a hub for faculty members, students, and alumni to launch their entrepreneurial initiatives.

2.4 Communication and outreach

In order to improve its communication and outreach, CMU Portugal has developed a new website and a new image, namely a new CMU Portugal logotype and several other secondary logotypes. Besides that, it has invested in online activities such as a better social media presence, the launch of a newsletter, and the creation of direct-targeted mailing campaigns. CMU Portugal has launched press releases to the Portuguese media as well as organized several events targeting different audiences to showcase the outcomes and benefits of the program.

3

CMU Portugal's data and sources

Contents

3.1 CMU Portugal Data Overview 9

3.2 Data Sources 12

3.3 Data quality problems 17

As previously mentioned, the CMU Portugal's data comes from multiple sources, such as various Excel files and handwritten inputs on their website, which leads to data quality issues. Such problems include inconsistent nomenclatures, approximate duplicate entries in the Excel files, empty Excel table entries, discrepancies between different Excel files, and data that is presented on the website but not in the Excel files.

Considering this, the purpose of this section is to give an overview of the CMU Portugal's data, its data sources, as well as its data quality issues.

3.1 CMU Portugal Data Overview

CMU Portugal's data is divided into four main topics, as it is possible to see in fig. 3.1. The data includes the students that participate in the educational programs described in section 2.1, the visitors

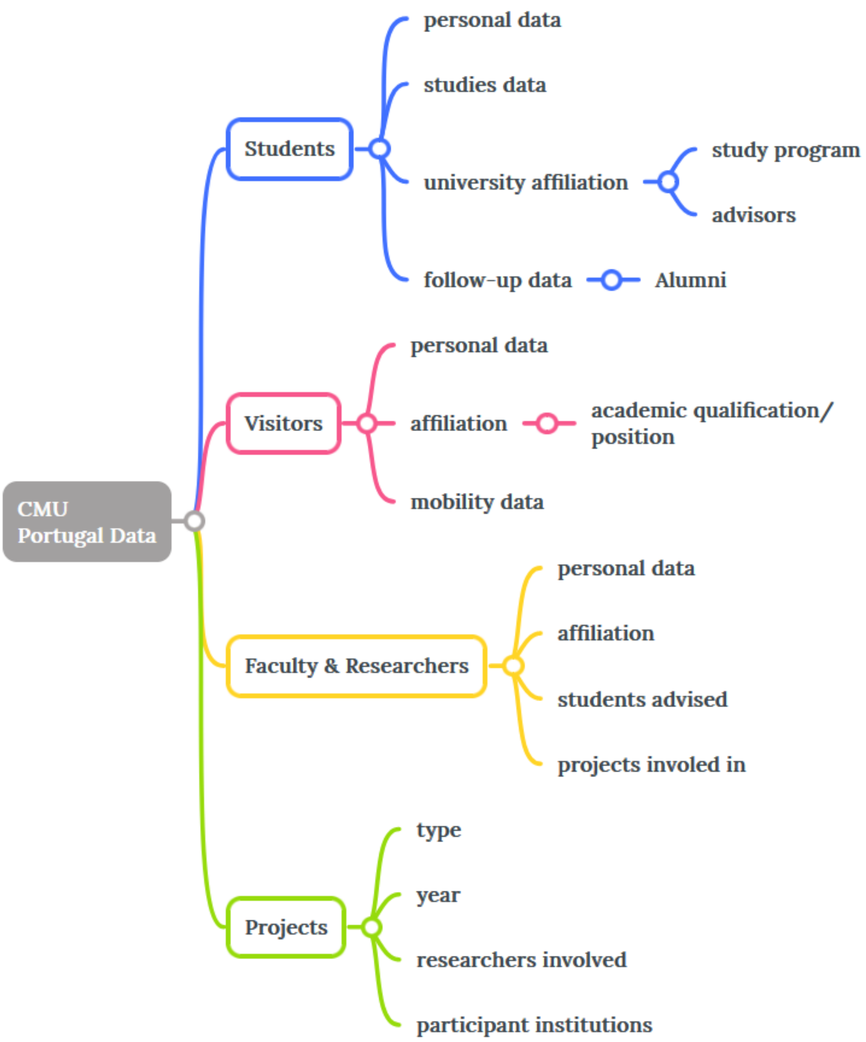


Figure 3.1: Organization of CMU Portugal's data

that participate in the mobility programs also described in section 2.1, faculty members and researchers working under CMU Portugal and the project initiatives fostered by the program as described in section 2.2.

3.1.1 Students

The students' personal data mostly concerns their names, genders, emails, nationalities, and online profiles such as Google Scholar, LinkedIn, and others. In terms of the general data kept about their studies under CMU Portugal, records about the students' research area, degree type (Dual Degree Ph.D. or Affiliated Ph.D.), status and dates of enrollment, expected graduation, and graduation are kept, as well as the number of enrollments. A student's status refers to whether the student is still active, has withdrawn from the program, or has transitioned to being an alumni. Since students under CMU Portugal have contact with CMU and another partner Portuguese university, a key element of data kept by CMU Portugal are the doctorate program(s) the students are enrolled in, as well as the name of the school and university in Portugal and the department and school name in CMU they are affiliated with. Doctorate students with the degree type "Dual Degree" are enrolled in two study programs, one at a Portuguese university and one at CMU, while students with the degree type "Affiliated" are only enrolled in a Portuguese university, as it is described in Section 2.1. However, all students, regardless of degree type, are associated with a CMU department. Besides that, the names of the advisors at the respective universities are stored, as well as data related to the student's thesis. Finally, CMU Portugal keeps follow-up information on its alumni, such as their current position, location, the position's institution name and type, as well as the starting date. The institution type refers to whether the institution is a university, a governmental agency, or an industrial institution.

3.1.2 Visitors

The visitors are students and faculty members coming from Portuguese universities that participate in CMU Portugal's mobility programs (see Section 2.1). Just as with students participating in CMU Portugal's educational programs, the visitors' personal data is stored and consists of their names, genders, and emails. The names of the schools and universities where the visitors come from correspond to their affiliation data. Additionally, for the student visitors, the highest academic qualification obtained and, for the faculty visitors, their position under the university they come from are kept. Consequently, data records of the names of the hosts and host departments at CMU that receive the visitors are generated. Lastly, the visitors' mobility data consists of the mobility's start and end dates, as well as the length of stay.

3.1.3 Faculty and Researchers

The faculty and researchers' personal data consists just like the visitors of their names, genders, and emails. The faculty members' and researchers' affiliation(s) can be the name of the department, school, and university they currently work under, the research lab they research under, or both. Other important data involves the names of the students they advise and the name and type of project they work on or have worked under.

3.1.4 Projects

Projects can be of the type ERI, ERP, or LSCRP, as described in section 2.2. Additionally, data on previous research initiatives, such as the Early Bird Projects (EBPs) and Lead Projects (LPs) have also been stored. The project's year identifies the year the project initiative was launched. Important data on the project concerns the researchers involved and their positions in the project. In general, all projects are made up of at least one Principal Investigator (PI) from a Portuguese institution and another PI from the CMU. Moreover, data on the institutions that participate in the project and their role in them, e.g. principal contractor, promoter, participant, etc., are also important records. Since projects are usually made up of one Portuguese PI and PI from CMU, it is also the norm that at least one Portuguese institution and one CMU department are involved in a project.

3.2 Data Sources

3.2.1 Excel Files

The majority of CMU Portugal's data comes from various Excel files. Overall, four Excel files with a total of six worksheets were provided. These are described in Table 3.1. The tables in these files typically have a large number of columns, some of which correspond to redundant data. The columns maintained in these tables will be further examined in the following subsections.

Table 3.1: Description of the source Excel files

File Name	Worksheet Name	Description	Nr. of Columns	Nr. of Rows
Ph.D. Students-Alumni	Ph.D. Students	Data on Doctorate Students	47	175
Ph.D. Students-Alumni	AlumniPositions	Data on alumni work positions	19	85
Faculty	Faculty	Data on researchers and faculty members	44	469
Visitors	Visiting Faculty	Data on faculty participants of Mobility Programs	31	88
Visitors	Visiting Students	Data on students participants in mobility programs	31	88
Projects	Projects	Data on project initiatives	14	89

3.2.1.A Ph.D.Students Table

The doctorate students' table contains columns with personal data on the students, such as:

- Full Name, Short Name, Email, Gender, Nationality, Nationality Continent

Students may have more than one email and more than one nationality, which are separated by commas in the respective columns.

The columns that include the students' academic data are:

- Degree Type, Research Area, Student Status, Enrollment Date, Enrollment Year, Academic Enrollment Year, Withdraw Year, Graduation Year, Expected Graduation Year, Scholarship number;

In terms of data on a student's enrollments in CMU and a Portuguese university, the following columns arise:

- **CMU**: Program in CMU, Program in CMU Abbreviation, School in CMU, Department in CMU ;

An example of a row containing these columns looks like this: "Human-Computer Interaction; HCI; School of Computer Science; Human-Computer Interaction Institute".

- **PT**: Program in PT, Program in PT Abbreviation, School in Portugal, School in PT Abbreviation, University in Portugal, University in PT Abbreviation

These columns combine to form the following example row: "Engenharia Informática e de Computadores; EIC; Instituto Superior Técnico; IST; Universidade de Lisboa; UL".

However, while studying at a Portuguese university, students may be hosted by another institution, such as a Research Center where they do their research. Therefore, the following columns exist:

- Host Institution in Portugal, Host Institution in Portugal Abbreviation;

Other columns include:

- CMU advisor(s), PT advisor(s), Thesis title, Thesis Link, CMU Portugal Research Project;

Students may have more than one advisor in CMU and more than one advisor in a Portuguese university, whose names are separated by commas in the respective columns. The "CMU Portugal Research Project" column refers to the project(s) inside CMU Portugal the student participates in.

Finally, these columns store data on the students' online profiles:

- Webpage, Research Gate, DBLP, Google Scholar,LinkedIn ,Orcid;

The columns in this table that are considered redundant or unnecessary are:

- *PhD Id*, *Years for Ph.D.*, *Current Year*, *Graduation Date*, *First Name*, *Last Name*, *CMU advisor Count*, *PT advisor Count*;

The PhD Id is not necessary, since a different internal numerical id is created for the database. “Years for Ph.D.” can be easily calculated by subtracting the graduation year with the enrollment year. “Current Year” is similar to the “Student Status” column and thus considered redundant. The “Graduation Date” has inconsistent date formats and was deemed unnecessary by the CMU Portugal’s team, as did the remainder columns presented here.

3.2.1.B AlumniPositions Table

This table contains a lot of redundant columns that have already been provided in the “Ph.D.Students” table, such as:

- *PhD Id*, *Student Short Name*, *Research Area*, *Nationality*, *Nationality Continent*, *Academic Enrollment Year*, *Enrollment Year*, *Graduation Year*, *Nr. of Ph.D. Years*, *Student Status*;

Important columns that make up this table are:

- Position, University/ Company/ Other, Start Date Position, Type of Institution, Country, Company Website, Date Verified, Source;

”Position” refers to the position name of the alumni, “University/ Company/ Other” to the employer’s organization name, and “Type of Institution” to the organization type. The “Company Website” is the link on the company’s website where the position of the alumni is shown. Finally, “Date Verified” details the year the position of the alumni was last verified and, ”Source” is the source of the data stored of the alumni’s work positions.

3.2.1.C Faculty Table

The following columns store the faculty members’ and researchers’ personal data:

- Database, Name, Email 1, Email 2;

”Database” describes the CMU Portugal’s members’ type, e.g.: “Faculty CMU” or “Researcher PT”, among others.

The columns that keep the faculty members’ and researchers’ affiliation with Universities or Research Centers are:

- Affiliation, Institution, Department , Research Center, Research Center 2;

The column “Affiliation” refers to the type of affiliation that a faculty member or researcher has, such as whether they are connected to the “Academic” or “Industry” field. “Institution” is the name of the organization, whether it be a company or a university, that they work for.

The remainder of the columns in this table correspond to repeated data. These include the project names and types that the faculty and researchers work on, which are listed in the "Projects" table, the visiting faculty and students that they have hosted at CMU also described in the "Visiting Faculty" and "Visiting Students" table and finally, the doctoral students that they have advised, which are listed in the "Ph.D. Students" table.

3.2.1.D Visiting Faculty and Visiting Students table

The "Visiting Faculty" and "Visiting Students" tables contain very similar columns. As with other tables, the personal data of the visitors is gathered, in this case by the columns:

- Name, Gender, Email;

Additionally, the columns "Academic Qualification", e.g., "Bachelor" or "Master", for visiting students, and "Work Position" for visiting faculty are maintained.

The visitors' mobility data includes the columns:

- Enrollment Year, Start Date, End Date;

Visiting students have the supplementary column "Sponsor in Portugal".

The following columns reflect data on the visitor's origin institution:

- Faculty, Faculty Abbreviation, University;

Additionally, the "Visiting Faculty" table has the columns:

- Research Lab Abbreviation, Research Lab;

These consist of research centers in Portugal, where these visitors conduct their research.

The hosts names and department in CMU that receive the visitors are described by the ensuing columns:

- Host name, Host department, Host School;

Some of the remaining columns of this table are considered redundant, such as, for instance, columns like "*Length of Stay*" which can be calculated, while the remaining others were disregarded at the advice of CMU Portugal's team.

3.2.1.E Projects Table

The "Projects" table contains general data on the projects contemplated by the columns:

- Project Type, Project Year, Project Reference;

The names and positions of the participant researchers are kept in the columns:

- Portugal Principal Investigator, Portugal Co-Principal Investigator, CMU Principal Investigator, CMU Co-Principal Investigator;

It is not guaranteed that all projects have a Portuguese Co-Principal Investigator (Co-PI) or a Co-PI from CMU, but there must always be a Portuguese and CMU PI present.

In terms of the institutions participating in the project, the following columns arise:

- Proponent Institution, Participating Institutions, Participating Organizations, Research Unit, CMU Department 1, CMU Department 2;

There is always a proponent institution and a CMU department present in the project, the other institutions are not always part of all projects.

Redundant columns in this table include the emails of the participant researchers since these are already stored in the "Faculty" table.

3.2.2 CMU Portugal Website

CMU Portugal's website is developed in WordPress, and as previously stated, it is fed by Excel files and handwritten inputs. These Excel files are formatted in a way where only the necessary data for the website is retained from the original Excel files, shown in Table 3.1. Usually, the formatted Excel files are created when there is a need to perform large uploads of data onto the website. Otherwise, new data is inserted via a form plugin that is supported by WordPress.

In the back end of WordPress, this data is stored in WordPress' own centralized SQL database, which uses MySQL as its database management system [2]. Among the data kept in WordPress's centralized database are [3]:

- Posts, pages, and other content.
- Organizational information such as categories and tags.
- User data and comments.
- Site-wide settings.
- Plugin and theme-related data.

With this in mind, the CMU Portugal's data is scattered across some of these tables, making the WordPress database not usable for the intended purpose. However, through a plugin supported by WordPress, it is possible to download the data in CSV format referring to the students, faculty, researchers, and projects stored on the website.

The necessary columns to extract from the Website that are not in the Excel files detailed in Table 3.1 are:

- **Student's data**: Introduction text, Research topics;
- **Project's data**: Start Date, End date;

The doctoral students' introduction text refers to a brief text on the website's profile of the student that gives insight into the student's motivation and background. Apart from that, some projects have a start and end date documented on the website that is not present in the original Excel files.

3.3 Data quality problems

As previously stated, the data sources from CMU Portugal have some data quality issues. After profiling the data the issues outlined below arose. CMU Portugal's data profiling is further detailed in Section 5.3.1.

1. Empty excel table entries: Some Excel tables have important unfilled entries that must be completed before migrating the data into the new database. The main identified cases of this type are:
    - Students without email.
    - Students without a study program in Portugal.
    - Faculty members and researchers without email.
    - Faculty members and researchers without affiliation and an associated institution.
    - Alumni without work position start date.
2. Approximate duplicates in tables' columns: This happens mostly due to mistakes in inputing the data. One of these cases happens in the "Ph.D. Students" table in the column "Department in CMU" which stores the department in CMU the doctorate students are affiliated with. This occurs, for example, with the CMU department "Human-Computer Interaction Institute" which also appears as "Human-computer interaction Institute" with the difference being the "c" in "Computer" and "i" in "Interaction" not being capitalized.
3. Duplicate entries in the "Faculty" table: There are duplicate entries of some faculty members and researchers that contain the same name and represent the same person but have distinct data from each other. This typically happens when old and new data are not merged, leaving two rows of the same person with old and new data in each.

4. Nomenclature Inconsistencies: There are variations in the definitions of several terms and columns, resulting in the occurrence of non-normalized conventions that can cause some confusion. These happen in the “Ph.D.Students” and “Faculty” tables, namely:

   - The column “Department” in the “Faculty” table is supposed to store the department a faculty member or researcher is associated with. However, this column is often misused to store the school or research center these people are affiliated with. For instance, this column may contain “Departamento de Engenharia Informática”, “ISR Lisboa” or “Heinz College”, with the second and third examples not being departments but a research center and a school at CMU, respectively.
   - The column “Program in Portugal” of the “Ph.D. Students” table which stores the study program of the doctorate students during their stay in Portugal. The same program can have different names, for example, the doctorate program “Engenharia Informática e de Computadores” at Instituto Superior Técnico either appears with that name and the abbreviation “EIC” or as “Programa Doutoral em Engenharia Informática e de Computadores” with no abbreviation given. Nevertheless, these two entries represent the same program.
   - The abbreviation of the institution name “Instituto Superior Técnico” is abbreviated in the “Ph.D.Students” table to “IST” and in the “Faculty” table to “Técnico”.

5. Inconsistency between excel tables caused mostly by not up-to-date information. These inconsistencies take place mainly when the names of faculty members and researchers in the “Faculty” table are used in other tables. This happens in the “Ph.D.Students” table in the columns of the advisors, in the “Projects” table columns that detail the names of the researchers that participate in these projects, and in the “Visiting Faculty” and “Visiting Students” columns presenting the names of the hosts at CMU. The main inconsistencies are:

   - Incomplete names of faculty members or researchers in different tables than the “Faculty” table that don’t match with their complete names in the “Faculty” table and vice versa.
   - Names of faculty members and researchers in different tables than the “Faculty” table that do not exist in the “Faculty” table.

6. Missing data from the Excel files that is contained in the website. As previously described in Section 3.2.2 the students’ data “Introduction text” and “Research topics”, and the project’s data “Start Date” and “End date” need to be extracted from the website to complete the data coming from the original Excel files.

4

Related Work

Contents

4.1 Database Management System 19

4.2 Data Analysis and Integration 22

4.3 Database user interface 24

This chapter analyzes the software tools necessary to accomplish the goals described in Section 1.2. First, database management systems necessary to build the database for CMU Portugal are studied. Data profiling and ETL tools to analyze, clean, and migrate the CMU Portugal data are then explored, followed by an examination of tools to build a user interface for the database.

4.1 Database Management System

In accordance with Abraham Silberschatz and Sudarshan [4] a Database Management System (DBMS) is a collection of interconnected data and a set of software systems that make it possible to define, create, maintain, and regulate access to this data. Additionally, and according to Mostafa et al. [5], it also provides access to multiple concurrent users while maintaining data integrity. The collection of interconnected data is the database, which is organized in a structured format and typically contains important information of an organization. A DBMS's main objective is to offer a simple and effective method for storing and retrieving large amounts of data from a database. For this reason, it normally includes the tools and interfaces to create, query, and modify such databases. Some of Database Management Systems (DBMSs) essential characteristics include:

1. Data definition: enables the specification of a database's structure, i.e., the definition of its tables, fields, and the relationships between them.

2. Data manipulation: provides the ability to insert, update, and delete data from the database.

3. Data retrieval: allows for high-speed retrieval of data from the database using queries that can be conducted through various interfaces.

4. Data security: ensures the safety of the stored data, be it from system crashes or unauthorized access attempts. To this end, it possesses security components that let its users manage who has access to the database.

Since information is so important in most organizations, various examples of DBMSs exist, such as PostgreSQL, Oracle, and MySQL. These DBMSs are further explored below.

4.1.1 PostgreSQL

According to PostgreSQL documentation [6], PostgreSQL has as its foundation POSTGRES, Version 4.2, and is an object-relational database management system. It was created by the Berkeley Computer Science Department of the University of California in the 1980s. Among its most notable features are the fact that PostgreSQL is open-source, has great scalability, i.e., can handle substantially large databases, and its extensibility. PostgreSQL's extensibility means it includes a vast number of built-in functions and operators that can be extended by its users, allowing them to define their own data types, functions, and procedural languages.

4.1.2 Oracle Database

The Oracle database is a relational database management system created in the late 1970s by the Oracle Corporation, according to the Oracle Database documentation [7]. It is widely used for enterprise-level data management, has great scalability, high availability, and has since become one of the most popular DBMS [8]. However, it is not open-source and has a steep learning curve.

4.1.3 MySQL

MySQL is a popular open-source relational database management system that was first released in 1995 and is now owned by Oracle Corporation. According to Poljak et al. [9], MySQL provides ease of use with simple syntax and mild complexity, high performance, and is open-source. However, it is not as scalable as the other DBMSs presented before, making it a good option for the development of small, web-based solutions with a small volume of data. It also does not fully comply with SQL standards, providing no support for some standard SQL features.

4.1.4 Discussion

Table 4.1: Summary of the pros and cons of the DBMSs

	PostgreSQL	**Oracle**	**MySQL**
Open-source	✓	✗	✓
Supported on Windows	✓	✓	✓
Supported by WordPress	✗	✗	✓
Data Types Supported	15	10	15
Learning Curve	Mild	Hard	Mild
Supported programming languages	C, C++, Java, PHP, Python,	C, C#, C++, Java, JavaScript, PHP, Python	C, C#, C++ Java, PHP Python
Scalabilty	✓✓✓	✓✓✓	✓✓

In table 4.1 a summary of the advantages and disadvantages of each of the above-mentioned DBMS is presented. It is important that the chosen DBMS is open-source and supported by Windows since it is the environment where the project will be developed. A key characteristic the DBMS must fulfill is to be supported by WordPress, considering that a goal of the new CMU Portugal database is to be an integrated data source that can serve the users' needs and the website. As it is stated in Section3.2.2 the DBMS of WordPress is MySQL. This makes MySQL the logical DBMS to choose from. However, the fact that it has a mild learning curve and supports many data types and programming languages also makes it a valuable candidate. The Oracle Database makes for a bad contender since it is not open-source and has a hard learning curve. PostgreSQL, on the other hand, is a good contender against MySQL since it has better scalability and since MySQL does not provide support for some standard SQL features, as mentioned in 4.1.3. Nonetheless, the CMU Portugal database does not need a lot of scalability at this point and probably not in the near future because it does not yet reach the tens of thousands of records. With all this in mind, it makes sense to choose MySQL as the DBMS of the CMU Portugal database, for the reason that it is open source, can be developed in the Windows environment, and most importantly, is supported by WordPress. It is fundamental that this new database be able to be integrated into the CMU Portugal website. Besides that, it is a good solution for the development of a small application with a modest amount of data.

4.2 Data Analysis and Integration

In this section, the goal is to analyze the tools that can help with the process of analyzing and understanding the quality problems of the CMU Portugal data and ease the process of integrating and migrating this data. The aim is to find the tool that is most complete in solving these two issues. To achieve this, it will be necessary to make use of data profiling and ETL tools. It is possible that some tools overlap between data profiling and ETL, but in essence, data profiling tools are primarily used to understand the structure, content, and quality of data, while ETL tools are more focused on transfering data from one place to another.

4.2.1 Data Profiling

In accordance to Naumann [10] data profiling is the process of examining the data available in an existing data source and collecting statistics and information about the quality of the data in order to identify potential issues or areas for improvement. To do this, data profiling tools encompass a vast array of methods to analyze data sets and produce metadata, such as, for instance, the number of null values and distinct values in a column, its data type, or the most frequent patterns of its values, as well as detect approximate properties of the data set. The key characteristics of these tools are [11]:

1. Providing the ability to examine data from several sources and formats, such as databases, flat files, or web services

2. Assist in determining the data's level of completeness, correctness, accuracy, consistency, and validity.

3. Enabling to map relationships between different data elements, such as tables or columns in a database

4. Provide dashboards and visuals to help users comprehend the data.

4.2.2 ETL

Extract, Transform, Load is referred to as ETL. It consists of an operation used in data integration that aims to extract data from a variety of sources, transform it into a common format, and load it into a target database or data warehouse [12]. The extract step consists of moving data, typically in its raw form, from various sources, such as databases, files, or web services. The transform step includes the transformation of data in such a way that it meets the target system's requirements. This usually means the need to perform data mapping, data cleansing, and data normalization. The load step concerns loading the data into the target system, which may involve the creation of tables, indexes, or other database objects.

Additionally, ETL tools often have a graphical user interface that enables them to drag and drop components onto a visual canvas in order to construct ETL workflows. Connectors, data transformations, data mapping, and data quality checks are among some of these components.

4.2.3 Tools Discussion

Table 4.2: Key characteristics of Talend and Pentaho

	Talend	**Pentaho**
Open-source	✓	✓
Data validation	✓	✓
Duplicate Detection	✓	✓
Value format change	✓	✓
Complex transformation	✓	✓
Join multiple sources	✓	✓
GUI	tough to grasp	modernized, easy to understand
Data Integration	✓✓✓	✓✓✓
Speed	✓	✓✓✓

In terms of data profiling tools, there are a variety of software tools like pandas, the Pentaho Data Integration plug-in DataCleaner and the IBM InfoSphere Information Analyzer. Pandas is a popular Python library for data manipulation and analysis. While pandas is not a data profiling tool per se, it does have some features that can be used for data profiling and can be useful for getting a basic understanding of the data. However, it makes for a bad contender since it is not as complete and does not offer the same level of sophistication as the other tools. Some ETL tools that are also suitable for data profiling are Talend Open Studio and Pentaho Data Integration (PDI). IBM DataStage is an ETL tool that is also a product within IBM InfoSphere. However, the products within IBM InfoSphere are not open-source, providing only a time-limited free trial for IBM DataStage. Since there are other great open-source options and there is no need to utilize a paid tool for the project, IBM InfoSphere also makes for a bad contender. A look is taken at these open-source tools, considering Pentaho as the junction of

Pentaho Data Integration and its plug-in DataCleaner.

In table table 4.2 [11] [13] [14] a summary of the key characteristics of these tools is provided. As it is possible to see in the table table 4.2, both Talend and Pentaho fulfill almost all the same qualities. Pentaho is somewhat faster and has a slightly more intuitive GUI than Talend. Concluding, both tools would be suitable for the development of the project, both being very complete and meeting the necessary requirements. The main deciding factors, in the end, came down to Pentaho being faster and to the lack of experience with the Talend tool. This way, Pentaho became the most viable solution, cutting time in the process of adapting to another tool and allowing for the time spared to be spent developing other features of the project.

4.3 Database user interface

The database user interface is the component of a database application that enables users to interact with the database. This component offers a graphical interface that allows the entering, viewing, and modification of the data, as well as running queries and generating reports. A well-designed user interface contributes to an easy user interaction with the database and increases the application's overall usability and effectiveness. With this in mind and following Sridevi [15] guidelines, the aim is to create an intuitive, responsive, and easy-to-use interface with simple navigation. Additionally, it should include validation checks to guarantee that the data that is being entered is valid and meets business rules or requirements, as well as include features that allow for search, sorting, and filtering options.

User interfaces can be developed using a variety of technologies and tools, such as HTML, CSS, and JavaScript for the front end of the user interaction and application programs for processing requests at the application server, such as Java Server Pages, or scripting languages such as PHP and Python.

4.3.1 PHP

PHP is a scripting language widely used for server-side scripting that can be combined with HTML [4]. Some of the key strengths of PHP include being open-source, being known for its ease of use and low learning curve, having a large community, being well-maintained, and having many open-source libraries. Additionally, it supports many popular databases, including MySQL, PostgreSQL, Oracle, and SQLite. Besides that, it is flexible and scalable. However, PHP is dynamically typed, which means it can be very error-tolerant, making it possible to develop highly inefficient solutions for simple problems.

4.3.2 Java Server Pages

Java Server Pages (JSP) is a scripting language that allows developers to embed Java code directly into HTML pages, as well as enabling the mix between static HTML with dynamically generated HTML. JSP is generally considered to have good performance, and according to Trent et al. [16], JSP tends to perform better than PHP under high loads. However, it has a higher learning curve. Overall, JSP is a powerful and flexible technology that enables programmers to construct dynamic and interactive web pages using Java as the server-side scripting language. It is a popular option for enterprise web development, especially for applications that require significant business logic and database connectivity.

4.3.3 Python and the Django Framework

Python is a high-level programming language that may be used for a variety of applications, such as server-side scripting. Additionally, Python is a powerful tool that is simple to understand and pick up on and has a large support base with a variety of libraries for numerous usages. However, Python is not routinely used to create server-side scripts with back-end developers, instead, relying on frameworks such as Django to perform most of the labor-intensive tasks. The Django framework for Python is a popular open-source web application framework with good security features and scalability, which makes it a good fit for large-scale, complex web applications. Additionally, Django supports PostgreSQL, MySQL, Oracle, and SQLite databases. However, this comes at the price of a moderate to steep learning curve, according to Ghimire [17].

4.3.4 Discussion

It was decided to use HTML, CSS, and Javascript to develop the database user interface. Almost every computer today comes pre-installed with a web browser, and HTML is independent of the operating system or browser. Consequently, a web-based program can be used on any computer that has internet access. Besides that, the alternative option of building a client-server architecture would most likely require the installation of application-specific software on client machines [4]. With this in mind, it is possible to conclude that a web application built in HTML and paired with CSS and Javascript can result in a sophisticated user interface supported by most web browsers, which makes it a suitable choice for the project.

PHP was chosen as the scripting language to take requests on the server. This is because the other options are better suited to develop complex web applications that require significant business logic and database connectivity. Since this is not the case, the option of an open-source scripting language with a simpler learning curve that is flexible and scalable made for a good fit. Besides that, PHP is widely used, suitable for a wide range of applications, from small to large-scale, and is powerful and versatile.

5

Database and User Interface

Contents

5.1 Requirements Gathering 27
5.2 Database Model 29
5.3 Data Profiling, Cleaning, and Migration 40
5.4 User interface 51

In this chapter, we go over the necessary steps to develop the CMU Portugal database and its user interface. We start with the requirements gathered, followed by the database's Entity-Relationship and relational models. Besides that, a section detailing the process of CMU Portugal's data profiling, cleaning, and migration into the database is presented. Finally, the last section outlines the user interface's development and features.

5.1 Requirements Gathering

The requirements gathering for the system to be developed consists of the main searches carried out in the CMU Portugal's various data sources. The goal is to identify the most important ones by interviewing members of the CMU Portugal team so that our integrated database can satisfy all of these demands.

In terms of visualization of the data, the following key searches arise:

1. Show the name, research area, degree type, and status of all students.
2. Show the name of all students, their study program in Portugal, their school in Portugal abbreviated, their study program in CMU, their school in CMU, and their department in CMU.

   It should be noted that regardless of degree type, all students should be displayed. However, students with an Affiliated degree type are not associated with a CMU program or school, they are just affiliated with a CMU department. Only Dual-degree students are associated with schools and programs in both CMU and Portugal.
3. Show the names, and the projects all of the faculty members and researchers of CMU Portugal are involved in.
4. Show the names, and students supervised of all faculty members and researchers of CMU Portugal.

In terms of the website, the requirements are:

5. Show the name, school in Portugal abbreviated, research area, and enrollment year of all active students.
6. Filter the first search by name.
7. Filter the first search by degree type.
8. Filter the first search by school abbreviation in Portugal.
9. Filter the first search by research area.
10. Filter the first search by an interval of enrollment years.
11. Show a specific student's introduction text, research topics, enrollment year, research area, school in Portugal abbreviated, CMU department, thesis title, supervisor in Portugal, supervisor in CMU, and online profiles.
12. All the above mentioned searches but for students with the status of "Alumni".
13. Show the names, university affiliations, department or school affiliations, and research center affiliations of all faculty members and researchers of CMU Portugal.

    It is important to note that faculty members and researchers can be associated with universities, their schools and departments, a research center, or both.
14. Filter the above search by affiliation.

Relatively to analytical searches the CMU Portugal team needs to perform in order to generate reports, the following come to light:

15. Show the number of students with the degree type "Dual Degree" grouped by their gender.
16. Show the number of students with the degree type "Dual Degree" grouped by their nationality area.
17. Show the number of students with the degree type "Dual Degree" grouped by their department at CMU.
18. Show the number of students with the degree type "Dual Degree" grouped by their university in Portugal.
19. Show the number of students with the degree type "Dual Degree" grouped by their enrollment year and current status, with the status being equal to "Student", "Alumni, "Withdraw".
20. Show all of the above-mentioned searches for students with the degree type "Affiliated".
21. Show the number of students with the degree type "Dual Degree" grouped by their enrollment year and study program at CMU.
22. Show the number of students with the degree type "Affiliated" grouped by their enrollment year and research area.
23. Show the first four searches, but for all students with the status "Alumni".
24. Number of visitors grouped by the year of their mobility and their university in Portugal.
25. Number of visitors grouped by the year of their mobility and their host department at CMU.
26. Number of visitors grouped by gender.

By acknowledging all these requirements, the aim is for them to be met by the new integrated CMU Portugal database. In the following sections, it is shown how the database model is thought out to adhere to these necessities.

5.2 Database Model

A database model is a conceptual representation of how data is organized and structured within a database system. In order to facilitate effective information storage, retrieval, and manipulation, it defines the logical structure and relationships between the various data elements. For this database, an Entity-Relationship and a Relational model were elaborated. In the following sections, the developed Entity-Relationship and Relational models of the database are presented.

5.2.1 Entity-Relationship Model

The developed Entity-Relationship model is presented in fig. 5.1. A description of this model is provided below. Firstly, we start off by detailing the entities, which are real-world objects or concepts that can be uniquely identified, and the attributes associated with them. Secondly, we outline the relationships, which represent associations between entities that describe how they are related to each other.

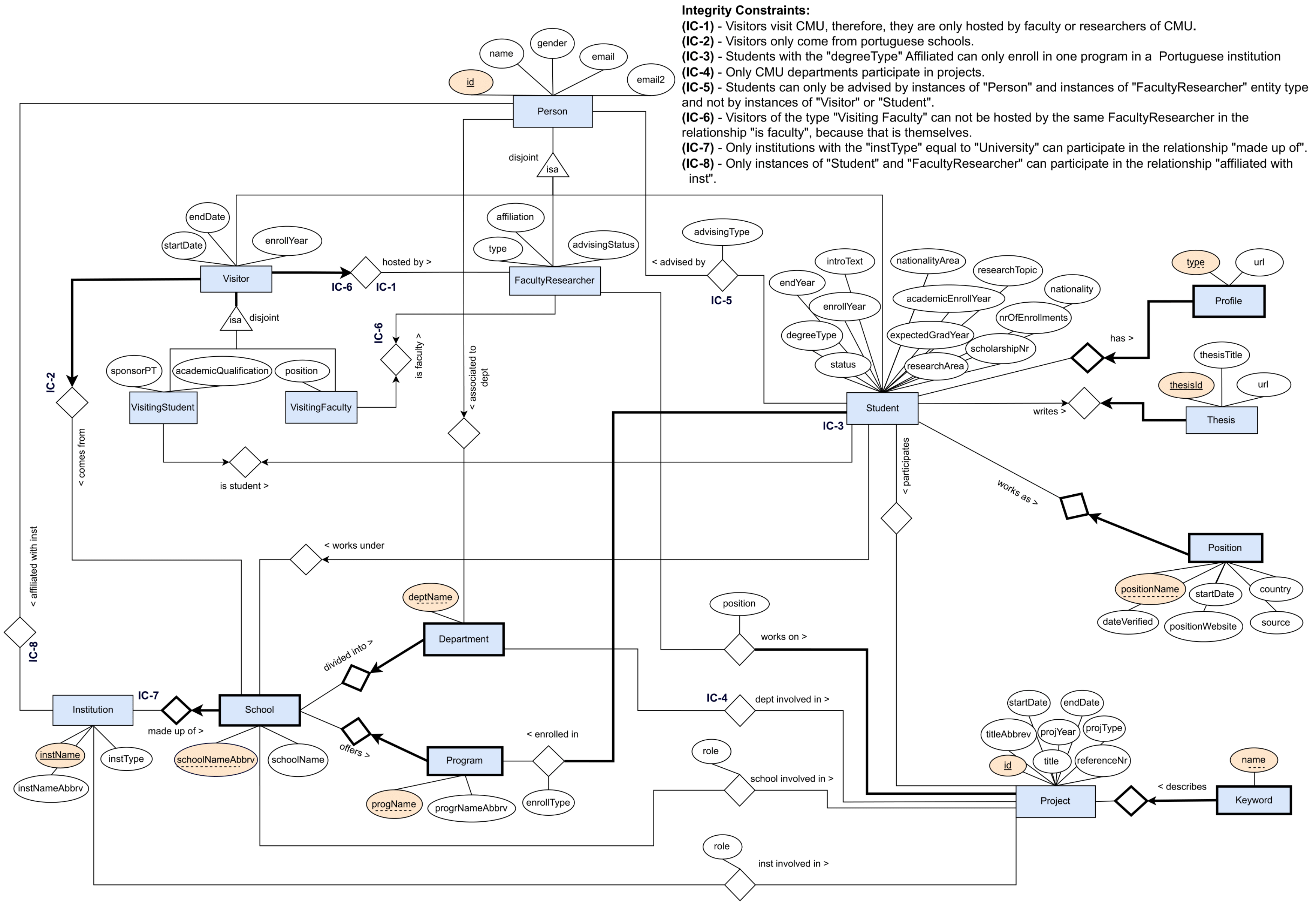
5.2.1.A Entities and attributes

- Person: is a generalization of the entities "Student", "FacultyResearcher" and "Visitor". The attributes of "Person" keep the overall personal data of the lower-level entities. Besides that, the attribute "id" is an internal id created to identify each "Person". In this case, these entities inherit all the attributes and relationships of the higher-level entity they are linked to. Since the constraint on the generalization is not total, instances of Person exist. This is for the case of students' supervisors that no longer exist as CMU Portugal's faculty or researchers.

- Student: an entity that represents students enrolled in CMU Portugal's Dual Degree or Affiliated doctoral programs. This entity's attributes correspond to the student's academic data, and the part of the personal data detailed in Section 3.2.1.A that corresponds to their nationalities. The attribute "endYear" refers to the year when a student left the program, either because they graduated or withdrew from it. There was a possibility of generalizing the entity student, with its lower levels being the degree types the student can have, "Dual Degree" and "Affiliated" and the student's statuses, "Active," "Withdrawn," or "Alumni". However, given the requirements gathered, it was important to have columns that show a student's degree type and status, which would not have been possible with the generalization mentioned above. Given this, this data was stored in the "Student" attributes.

- FacultyResearcher: this entity depicts the faculty members and researchers at CMU Portugal that supervise students during their Ph.D. and/or participate in projects included in CMU Portugal's project initiatives. The attribute "type" refers to the type given by CMU Portugal to a faculty member or researcher, e.g., "Faculty PT", "Researcher CMU", "Faculty and Researcher PT", etc. The attribute "advisingStatus" designates whether a faculty member is currently supervising a student or not. Finally, the attribute "affiliation" specifies whether the affiliation of the "FacultyResearcher" is to an academic or industrial institution.

- Visitor: portrays students and faculty members of Portuguese universities that participate in the CMU Portugal's mobility programs described in Section 2.1. The attributes of "Visitor" are the start and end dates of the mobility program as well as the enrollment year in this same program.



Integrity Constraints:
(IC-1) - Visitors visit CMU, therefore, they are only hosted by faculty or researchers of CMU**.**
(IC-2) - Visitors only come from portuguese schools.
(IC-3) - Students with the "degreeType" Affiliated can only enroll in one program in a Portuguese institution
(IC-4) - Only CMU departments participate in projects.
(IC-5) - Students can only be advised by instances of "Person" and instances of "FacultyResearcher" entity type and not by instances of "Visitor" or "Student".
(IC-6) - Visitors of the type "Visiting Faculty" can not be hosted by the same FacultyResearcher in the relationship "is faculty", because that is themselves.
(IC-7) - Only institutions with the "instType" equal to "University" can participate in the relationship "made up of".
(IC-8) - Only instances of "Student" and "FacultyResearcher" can participate in the relationship "affiliated with inst".

Figure 5.1: Entity-Relationship Model of CMU Portugal's database

- Visiting Student: specialization of “Visitor” which represents only the students that participate in CMU Portugal’s mobility programs. The attributes specific to these students are their sponsor in Portugal and academic qualification, meaning the highest academic qualification obtained by these visiting students.
- Visiting Faculty: specialization of “Visitor” which represents only the faculty members that participate in CMU Portugal’s mobility programs. The attribute of “Visiting Faculty” corresponds to their work position.
- Profile: is a weak entity of “Student” that consists of the online profiles of students enrolled in CMU Portugal’s educational programs. The attribute “type” refers to the type of the online profile, e.g., “Google Scholar”, “Orcid”, “LinkedIn”, etc. The attribute “url” corresponds to the link to access this profile.
- Thesis: this entity represents the theses that students of CMU Portugal write during their doctorate. This entity has an id, to distinguish it from others so that it is possible to look at students’ theses without it being forcefully associated with a specific student. The rest of the entity’s attributes are the thesis title and the “url” which link can be used to access the thesis.
- Position: a weak entity of “Student” that stands for the work positions of students that have finished their Ph.D. and have become alumni of CMU Portugal. The attributes of this entity are the name of the work position, the starting date, and the country where it is located. Besides that, “dateVerified” saves the date when the work position of the alumni was last checked, “source”, the origin of this information, and “positionWebsite”, the link where it is possible to find the alumni’s position within the company website.
- Institution: this entity represents institutions with whom students, faculty members, researchers, visitors, and project initiatives of CMU Portugal are affiliated. Therefore, these institutions can be of the type “University”, “Research Center” or “Company” which is detailed by the attribute “instType”. The remaining attributes are the institution’s name and abbreviated name.
- School: weak entity of “Institution”, represents the schools of a university, such as, for instance, “Instituto Superior Técnico” which is a school of “Universidade de Lisboa”. The attributes are the name and abbreviation of the school.
- Department: weak entity of “School”, refers to the departments that exist in a school, such as, for example, “Departamento de Engenharia Informática”, which is a department in the school “Instituto Superior Técnico”. The entity’s attribute is the name of the department.
- Program: is a weak entity of “School”, that describes the Ph.D. programs of a school. The entity’s attributes are the name and abbreviation of the program.

- Project: depicts the project initiatives fostered by CMU Portugal. The entity's attributes consist of the project's year, its start and end dates, as well as the title and the abbreviated title of the project. Besides that, the project's reference number, its type (see Section 2.2), and an id to distinguish between projects are also part of the attributes.

- Keyword: a weak entity of "Project", describes the keywords of a project, such as for instance the keywords "Risk Management", "Fraud control", "Revenue Assurance". This entity's attribute is the name of the keyword.

5.2.1.B Relationships

We begin by describing the relationships in which the entity "Student" is involved. After that, we move on to the relationships that involve "FacultyResearcher", then "Visitor", then "Person" and finally "Project".

- enrolled in: this relationship between "Student" and "Program" represents the study program(s) the Ph.D. students are enrolled in. Doctorate students may be enrolled in more than one program since students with the degree type "Dual Degree" are enrolled in a program at a Portuguese university and at CMU. On the other hand, students with the degree type "Affiliated" only enroll in a Ph.D. program at a Portuguese university (see Section 2.1). Students must be enrolled in at least one study program. Since "Program" is a weak entity of "School" which is a weak entity of "University", the relationship "enrolled in" provides the student's study program along with the school and university it is enrolled in. The relationship's attribute "enrollType" defines whether this enrollment is in a Portuguese university or in CMU.

- advised by: while pursuing their doctorate, students have advisors who are faculty members or researchers of CMU Portugal. The relationship "advised by" between "Student" and "Person" depicts this case. The relationship is between "Student" and "Person" and not between "Student" and "FacultyResearcher" since there are older students whose advisors are no longer part of the faculty members and researchers of CMU Portugal and thus cannot be represented as instances of "FacultyResearcher". However, this relationship can only happen between instances of "Student" and instances of "Person" and "FacultyResearchers" and not between instances of "Student" and instances of "Visitor" or "Student", as the integrity constraint (IC-5) details. As it is depicted in this relationship, students may have many advisors, while these advisors may also have many students they supervise. This relationship's attribute "advisingType" refers to whether the supervision between student and advisor takes place in a Portuguese university or at CMU.

- writes: this relationship between "Student" and "Thesis" portrays the data about the thesis the Ph.D. student has developed. Students write only one thesis, which is not mandatory since there are students who withdraw from the program or are still at the beginning of their studies. A thesis must be obligatorily written by a student and only one student.

- is student: relationship between “Student” and “VisitingStudent”, represents the case when “visiting” students that participate in CMU Portugal’s mobility programs later enroll and become students of CMU Portugal’s doctorate programs. This is a one-to-one relationship, meaning a “visiting” student can only correspond to a Ph.D. student and vice versa since they are the same person.
- participates: is a relationship between “Student” and “Project”. This relationship exists because Ph.D. students of CMU Portugal can participate in CMU Portugal’s ongoing project initiatives. As such, a doctorate student may participate in more than one project, and a project may have various Ph.D. students.
- works under: this relationship between “Student” and “School” designates the schools belonging to universities, a former student, i.e. an alumnus, may work at. Alumni can only have one working position at a time, so if they work at a school, it can only be at that school. On the other hand, schools may have many CMU Portugal’s alumni working there.
- works on: relationship between “FacultyResearcher” and “Project” that depicts the project initiatives of CMU Portugal on which members of the faculty and researchers work. The relationship’s attribute “position” states the position of the faculty member or researcher under that same project, such as, for instance “PI” or “Co-PI”. Faculty members and researchers may be working on various projects, and a project may also have many of these people working on it. However, a project must have at least one faculty member or researcher participating in it.
- is faculty: just as with the “is student” relationship, the “is faculty” relationship represents “visiting” faculty members that participate in CMU Portugal’s mobility programs and, later on, become faculty members or researchers at CMU Portugal. This is again a one-to-one relationship, with a “visiting” faculty only corresponding to a faculty member or researcher at CMU Portugal and vice versa.
- hosted by: participants of mobility programs, i.e. “Visitors”, are hosted by faculty members or researchers at CMU. This relationship between “Visitor” and “FacultyResearcher” portrays the faculty members and researchers that are hosts to the visitors coming to CMU. These hosts are always affiliated to CMU since this is where visitors are hosted. Visitors must obligatorily have one host and a maximum of one host at CMU. On the other hand, members of the faculty and researchers at CMU can be hosts to many visitors.
- comes from: is a relationship between “Visitor” and “School” that depicts the schools belonging to Portuguese universities, where the participants of mobility programs originate from. The participants must be affiliated with at least and a maximum of one Portuguese school, while Portuguese

schools can have many students or members of faculty that participate in CMU Portugal's mobility programs.

- associated to dept: relationship between "Person" and "Department". This relationship is inherited by "Student", "FacultyResearcher" and "Visitor". All doctorate students are associated with a department at CMU. This happens either because their study program is under that department, in the case of "Dual Degree Ph.D." students, or because their research period was carried out under that CMU department, in the case of "Affiliated Ph.D." students (see Section 2.1). These students are affiliated with a maximum of one department. Faculty members and researchers of CMU Portugal working in the academic field are also affiliated with the department at the school and university they work for. It is not mandatory for all faculty members and researchers of CMU Portugal to be associated with a department, but if they are, they can only be associated with a maximum of one department. "Visitors" are also associated with a CMU department since their participation in a mobility program consists of them being hosted by a CMU department. They are hosted by a maximum of one CMU department.

- affiliated with inst: As with the previous relationship, this relationship between "Person" and "Institution" is inherited by "Student", "FacultyResearcher" and "Visitor". However, only instances of "Student" and "FacultyResearcher" participate in this relationship, as stated in the integrity constraint IC-8. Doctorate students can be hosted by institutions such as research centers, where they conduct their research during their Ph.D. This is one case of how students participate in this relationship. Another case is of former students who have now become alumni and who work in an organization, represented by "Institution". The previously described relationship "works under", only covers the case of alumni working in schools belonging to universities, but in which it is important to also keep the link to the school under the university they affiliate with. The case of "FacultyResearcher" is similar. This relationship describes the affiliation between faculty members and researchers of CMU Portugal that do not work in the academic field but in companies, with "Institution" representing those companies.

- dept involved in: relationship between "Project" and "Department", that depicts the CMU departments that participate in the project initiatives fostered by CMU Portugal. A CMU department may be involved in various projects, and a project normally has one to two CMU departments. Only departments of CMU participate in this relationship, as stated in the integrity restraint IC-4.

- school involved in: As in the previous relationship, this relationship between "Project" and "School" portrays the schools that are involved in CMU Portugal's projects. The relationship's attribute "role" describes the role this school plays in the project, for instance, the school may be a "Proponent Institution", a "Participant Institution" or a "Research Unit". The projects can have many participating schools, and the schools may also participate in various CMU Portugal's projects.

- inst involved in: this relationship between "Project" and "Institution" is similar to the previous ones as it depicts the institutions involved in CMU Portugal's projects. As previously described, "Institution" can represent a university, organization, research center, or company. The relationship's attribute "role" stands for the same as in the relationship "school involved in", the role the institution takes in the CMU Portugal's project. Once again, projects may have many participating institutions, and an institution may also participate in many projects.

5.2.2 Relational Model

The relational model is presented below, in which the primary keys of each table are underlined, the foreign keys are represented by "FK" and the integrity restraints by "IC".

```
Person (id, name, gender, email, email2)
    IC: id can only appear in FacultyResearcher or Student or Visitor, but not on more
    than one simultaneously.

Student (id, researchArea,  degreeType, status, enrollYear, academicEnrollYear,
    expectedGradYear, endYear, nrOfEnrollments, nationality, nationalityArea,
    scholarshipNr, introText, researchTopic)
    id: FK(Person)
    IC: When Person entity is deleted so is the Student with the same id.

Profile (id, type, url)
    id: FK(Student)
    IC: When Student is deleted so is/are the Profiles with the same id.

Position (id, positionName, startDate, country, dateVerified, source, positionWebsite)
    id: FK(Student)
    IC: When Student is deleted so is the Position with the same id.

FacultyResearcher(id, type, affiliation, advisingStatus)
    id: FK(Person)
    IC: When Person is deleted so is the Student with the same id.

Visitor (id, enrollYear, startDate, endDate)
```

```
    id: FK(Person)
    IC:
     - When Person is deleted so is the Visitor with the same id.
     - The endDate must be a later date than the startDate.

VisitingStudent (id, sponsorPT, academicQualification)
    id: FK(Visitor)
    IC: When Visitor entity is deleted so is the VisitingStudent with the same id.

VisitingFaculty (id, position)
    id: FK(Visitor)
    IC: When Visitor entity is deleted so is the VisitingFaculty with the same id.

Thesis (thesisId, thesisTitle, url)

Institution (instName, instNameAbbrv, instType)

School (instName, schoolNameAbbrv, schoolName)
    instName: FK(Institution)
    IC: When Institution is deleted so is the School that belongs to the Institution
    with the same name.

Department (instName, schoolNameAbbrv, deptName)
    instName, schoolNameAbbrv: FK(School)
    IC: When the School is deleted so is the Department that belongs to the school
    with the same name.

Program (instName, schoolNameAbbrv, progName, progNameAbbrv)
    instName, schoolNameAbbrv: FK(School)
    IC: When the School is deleted so is the Program that belongs to the school
    with the same name.

Project (id, projType, projYear, title, titleAbbrv, startDate, endDate, referenceNr)
    IC: The endDate must be a later date than the startDate
```

```
Keyword (id, name)
    id: FK(Project)
    IC: When the Project entity is deleted so is the Keyword entity.

AssociatedToDept (id, instName, schoolNameAbbrv, deptName)
    id: FK(Person)
    instName, schoolNameAbbrv, deptName: FK(Department)

AffiliatedWithInst (id, instName, affiliationType)
    id: FK(Person)
    instName: FK(Institution)

Writes (id, thesisId)
    id: FK(Student)
    thesisId: FK(Thesis)
    IC: All thesisId in Thesis must participate in writes.

AdvisedBy (Student.id, Person.id, advisingType)
    Student.id: FK(Student)
    Person.id: FK(Person)

EnrolledIn (id, instName, schoolNameAbbrv, progName, enrollType)
    id: FK(Student)
    instName, schoolNameAbbrv, progName: FK(Program)
    IC: All ids in Student must participate in enrolledIn.

WorksUnder (id, schoolNameAbbrv, instName)
    id: FK(Student)
    instName, schoolNameAbbrv: FK(School)

Participates (Student.id, Project.id)
    Student.id: FK(Student)
    Project.id: FK(Project)

WorksOn (FacultyResearcher.id, Project.id, position)
```

```
    FacultyResearcher.id: FK(FacultyResearcher)
    Project.id: FK(Project)
    IC: All ids in Project must participate in worksOn.

HostedBy (Visitor.id, FacultyResearcher.id)
    Visitor.id: FK(Visitor)
    FacultyResearcher.id: FK(FacultyResearcher)
    IC: All ids in Visitor must participate in hostedBy.

ComesFrom (id, instName, schoolNameAbbrv)
    id: FK(Visitor)
    instName, schoolNameAbbrv: FK(School)
    IC: All ids in Visitor must participate in comesFrom.

IsStudent (Student.id ,VisitingStudent.id)
    Student.id: FK(Student)
    VisitingStudent.id: FK(VisitingStudent)

IsFaculty (FacultyReasearcher.id, VisitingFaculty.id)
    FacultyReasearcher.id: FK(FacultyReasearcher)
    VisitingFaculty.id: FK(VisitingFaculty)

DeptInvolvedIn (instName, schoolNameAbbrv, deptName, id)
    instName, schoolNameAbbrv, deptName: FK(Department)
    id: FK(Project)

SchoolInvolvedIn (instName, schoolNameAbbrv, id, role)
    instName, schoolNameAbbrv: FK(School)
    id: FK(Project)

InstInvolvedIn (instName, id, role)
    instName: FK(University)
    id: FK(Project)
```

The code developed to create the database for CMU Portugal is available in Appendix A.

5.3 Data Profiling, Cleaning, and Migration

In this section, we go over the techniques used for profiling, cleaning, and migrating CMU Portugal's data. As previously stated in Subsection 4.2.3, the tools that will be utilized to accomplish this process are Pentaho Data Integration and its plug-in DataCleaner. From this point on in the project, due to time constraints and in line with the CMU Portugal team's preference, it was agreed to profile, clean, and migrate only the data related to the doctorate students. In terms of Excel files, this translates to the profiling, cleaning and migration of the data in the "Ph.D.Students" table and a part of the "Faculty" table since the Ph.D. students have supervisors that belong to the "Faculty" table. In terms of the database model, this meant migrating data to the tables "person", "student", "profile", "thesis", "institution", "school", "program" "department", "advisedBy", "enrolledIn", "writes", "associatedToDept", and"affiliatedWithInst".

5.3.1 Data Profiling

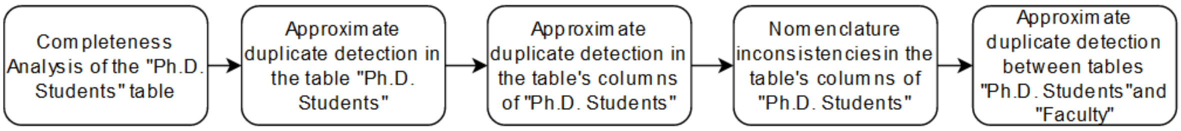

Completeness Analysis of the "Ph.D. Students" table
Approximate duplicate detection in the table "Ph.D. Students"
Approximate duplicate detection in the table's columns of "Ph.D. Students"
Nomenclature inconsistencies in the table's columns of "Ph.D. Students"
Approximate duplicate detection between tables "Ph.D. Students"and "Faculty"

Figure 5.2: Process of profiling CMU Portugal's data

The developed process of data profiling, displayed by fig. 5.2, consists of five steps. Firstly, a completeness analysis on the "Ph.D. Students" table is performed to find the columns with empty entries. Secondly, an approximate duplicate detection analysis is performed on this same table to check if there are any duplicate entries of doctoral students. Thirdly, an approximate duplicate detection in the columns of the "Ph.D. students" table is carried out to discover errors and typing mistakes, such as "Human-computer interaction Institute", in which the "i" in "interaction" and "c" in "computer" should be capitalized. After that, a detection of nomenclature issues is performed to discover non-standardized terms in the columns of this table, such as, for example, "Engenharia Electrotécnica e Computadores" and "Programa Doutoral em Engenharia Electrotécnica e de Computadores" which are the same program but designated in two different ways. Finally, an approximate duplicate detection analysis is performed between the "Ph.D. students" and "Faculty" tables to identify if the names of the Ph.D. students' advisors are consistent with the names of the faculty members and researchers' names kept in the "Faculty" table.

5.3.1.A Completeness Analysis

For the completeness analysis, the Pentaho Data Integration plug-in Data Cleaner is used. Firstly, the completeness analyzer tool of Data Cleaner is applied to the "Ph.D. Students" table, as can be seen in

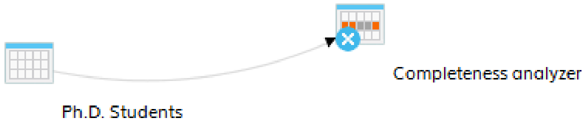
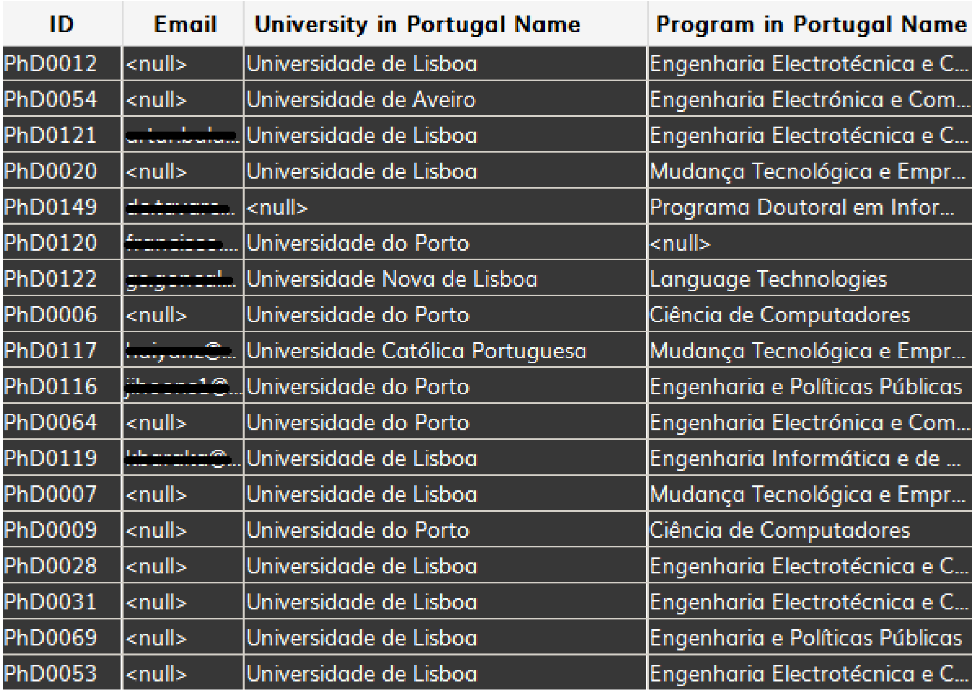


Figure 5.3: Completeness analysis of the “Ph.D. Students” table

fig. 5.3. Secondly, we select all columns in this table that should not be incomplete. The columns of this table that should not be empty are a student’s name, gender, email, nationality, degree type, status, research area, enrollment year, and the program(s), school(s), universities, and department(s) they are affiliated with. After selecting only the fields that interest us, we get the result that is presented in fig. 5.4. From this figure, it is possible to conclude that there are some students without an email, one without the name of the university where they are enrolled in Portugal, and another without a study program in Portugal.



ID	Email	University in Portugal Name	Program in Portugal Name
PhD0012	<null>	Universidade de Lisboa	Engenharia Electrotécnica e C...
PhD0054	<null>	Universidade de Aveiro	Engenharia Electrónica e Com...
PhD0121	[redacted]	Universidade de Lisboa	Engenharia Electrotécnica e C...
PhD0020	<null>	Universidade de Lisboa	Mudança Tecnológica e Empr...
PhD0149	[redacted]	<null>	Programa Doutoral em Infor...
PhD0120	[redacted]	Universidade do Porto	<null>
PhD0122	[redacted]	Universidade Nova de Lisboa	Language Technologies
PhD0006	<null>	Universidade do Porto	Ciência de Computadores
PhD0117	[redacted]	Universidade Católica Portuguesa	Mudança Tecnológica e Empr...
PhD0116	[redacted]	Universidade do Porto	Engenharia e Políticas Públicas
PhD0064	<null>	Universidade do Porto	Engenharia Electrónica e Com...
PhD0119	[redacted]	Universidade de Lisboa	Engenharia Informática e de ...
PhD0007	<null>	Universidade de Lisboa	Mudança Tecnológica e Empr...
PhD0009	<null>	Universidade do Porto	Ciência de Computadores
PhD0028	<null>	Universidade de Lisboa	Engenharia Electrotécnica e C...
PhD0031	<null>	Universidade de Lisboa	Engenharia Electrotécnica e C...
PhD0069	<null>	Universidade de Lisboa	Engenharia e Políticas Públicas
PhD0053	<null>	Universidade de Lisboa	Engenharia Electrotécnica e C...

Figure 5.4: Result of the completeness analysis on the table “Ph.D. Students”

5.3.1.B Detection of approximate duplicates in the “Ph.D. Students” table

In order to detect duplicates in the “Ph.D. Students” table, the tool Pentaho Data Integration is used, employing the transformation shown in fig. 5.5. This transformation consists of, firstly, choosing and

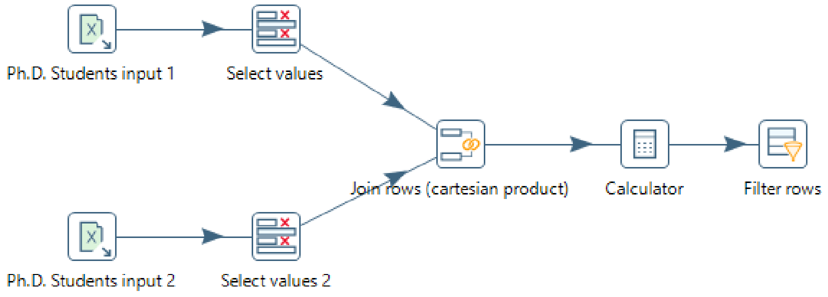


Figure 5.5: Transformation to detect approximate duplicates in the “Ph.D. Students” table

configuring its inputs “Ph.D. Students input 1” and “Ph.D. Students input 2”. In this case, the inputs are the Excel table “Ph.D. Students” and an exact copy of that same table. Secondly, “Select values” and “Select values 2” are used to select the columns from each table we want to view and compare. In this case, the selection is the same in both steps, since the goal is to compare the same columns with one another to find duplicates. Given this, we select columns “Id” and “Name” in both of the “Select values” and rename them to “id1”, “name1” and “id2”, “name2” to be able to distinguish between them. Following this, the columns coming from both inputs are joined in the “Join rows” step by using the cartesian product with the following condition: “id1 $<$ id2”. This condition is used since there is no need to compare a student’s name with its same instance. We then use the calculator step to create a new field called “similarity” which consists of the Levenshtein distance between the fields “name1” and “name2”. Finally, the step “Filter rows” is used to apply a threshold to the similarity. By the end of this transformation, the “Filter rows” step was unable to produce results after applying a variety of appropriate thresholds. This means that there are no duplicate entries in the table “Ph.D. Students”, i.e., there are no repeated students.

5.3.1.C Approximate duplicate detection in the table’s columns of “Ph.D. Students”

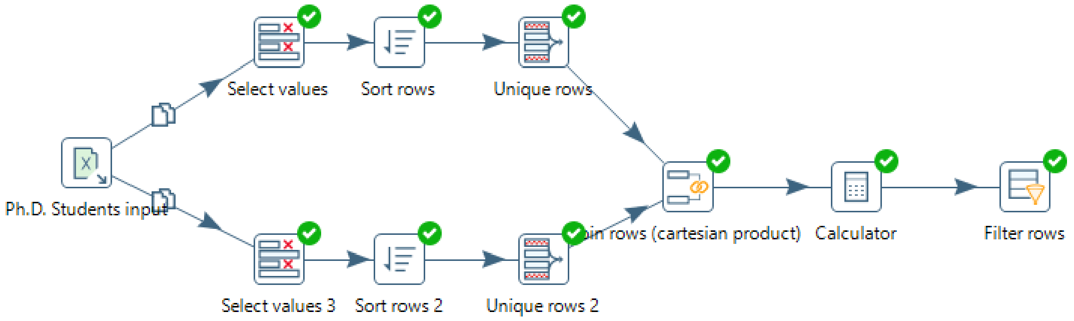
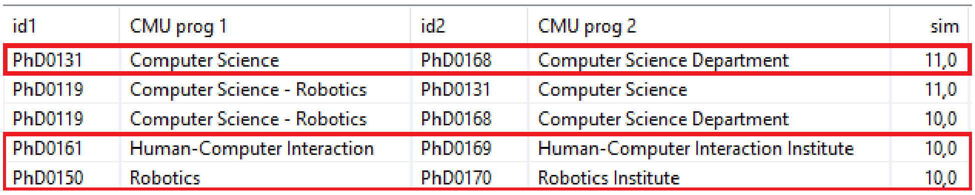


Figure 5.6: Transformation to detect approximate duplicates in the columns of the table “Ph.D. Students”

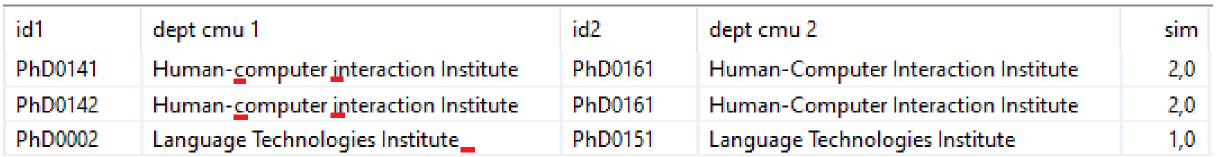
By detecting approximate duplicates in the “Ph.D. Students” table columns, we aim to discover typing errors and other mistakes. The transformation used is depicted in fig.5.6 and was employed to detect approximate duplicates in the columns that keep the student’s status, degree types, nationalities, research areas, schools in Portugal, university in Portugal, study program in CMU, school in CMU, and department in CMU. The transformation shown in fig. 5.6 is very similar to the one shown in fig. 5.5. The only difference is the added steps “Sort Rows” and “Unique rows”. These steps were added since, for example, when comparing CMU study programs, a limited number of doctorate programs appear repeated throughout almost two hundred students. The goal is to get rid of repeated values. By applying this transformation to the columns mentioned above, the results showed there are no approximate duplicates in the columns that keep the student’s status, degree types, nationalities, research areas, schools in Portugal, universities in Portugal, and schools in CMU. The results for the columns that keep the study programs in CMU are shown in 5.7 and for columns that keep the departments in CMU are shown in 5.8. In fig. 5.7, it is possible to see in the entries in red that CMU departments were inserted instead of CMU study programs by mistake.



id1	CMU prog 1	id2	CMU prog 2	sim
PhD0131	Computer Science	PhD0168	Computer Science Department	11,0
PhD0119	Computer Science - Robotics	PhD0131	Computer Science	11,0
PhD0119	Computer Science - Robotics	PhD0168	Computer Science Department	10,0
PhD0161	Human-Computer Interaction	PhD0169	Human-Computer Interaction Institute	10,0
PhD0150	Robotics	PhD0170	Robotics Institute	10,0

Figure 5.7: Result of the approximate duplicate detection in the column “Program in CMU” of the table “Ph.D. Students”

In fig. 5.8 we can identify typing mistakes in which some letters are not capitalized correctly. Besides that, in the last line of the table, a different type of quality issue can be observed, which is the existence of a blank space at the end of the term “Language Technologies Institute”. This leads us to conclude that there are probably more black spaces inserted by accident in the table “Ph.D. Students”.



id1	dept cmu 1	id2	dept cmu 2	sim
PhD0141	Human-computer interaction Institute	PhD0161	Human-Computer Interaction Institute	2,0
PhD0142	Human-computer interaction Institute	PhD0161	Human-Computer Interaction Institute	2,0
PhD0002	Language Technologies Institute	PhD0151	Language Technologies Institute	1,0

Figure 5.8: Result of the approximate duplicate detection in the column “Department in CMU” of the table “Ph.D. Students”

5.3.1.D Detection of Nomenclature inconsistencies in the “Ph.D. Students” table’s column

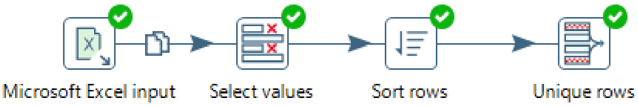
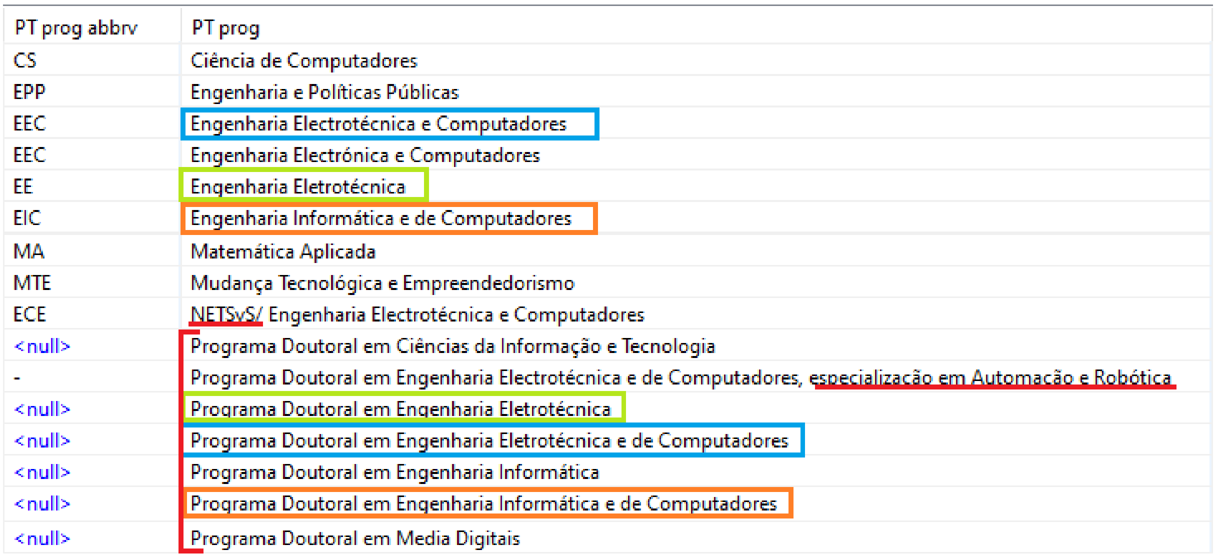


Figure 5.9: Transformation to filter distinct study programs in Portugal or nationality areas from the table “Ph.D. Students”

In order to detect cases of nomenclature inconsistencies, approximate duplicate detection was not used. This is because in this case, variations in the terminologies used are so dissimilar that they cannot be considered approximate duplicates nor detected with similarity measures. The similarity threshold applied to encounter these inconsistencies would have to be so high that other approximate duplicates that are of no interest would not be filtered out. Taking this into consideration, the transformation in fig. 5.9 is used to detect nomenclature inconsistencies in the column that keeps the students’ study programs in Portugal and the column that stores the student’s nationalities’ area. Either the column that stores the students’ study programs in Portugal or the area of the student’s nationalities is selected in the “Select values” step. The “Sort Rows” and “Unique Rows” steps are used to filter only unique rows containing distinct study programs in Portugal or distinct student’s nationalities’ areas. Although this transformation does not allow us to capture mistakes in these columns as clearly as the transformations shown before, it is still a good option that filters these columns in a way that helps us analyze mistakes in them.

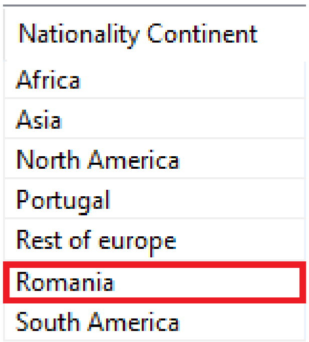
In fig. 5.10 it is possible to see the result of this transformation when applied to the column that keeps the students’ study programs in Portugal. We observe that there are at least three programs that are the same but designated with different names, such as, for example “Engenharia Electrotécnica e Computadores” and “Programa Doutoral em Engenharia Electrotécnica e de Computadores”. These should both be designated “Engenharia Electrotécnica e Computadores”. Underlined in red are other mistakes in this column, and overall, all programs starting with “Programa Doutoral” should have that part removed from their name. This is because we know these programs are doctoral programs based on the degree type of the student.

In fig. 5.11 we can observe the results of the transformation when applied to the column containing the areas corresponding to the student’s nationalities. The term ”Romania” is underlined in red since it does not correspond to an area of a student’s nationality but to a nationality. Its area should be ”Rest of Europe”. The term ”Portugal” is correct as a nationality area since the CMU Portugal’s team classifies students with Portuguese nationality as coming from the nationality area ”Portugal” and students coming from any other country in Europe with the nationality area ”Rest of Europe”. Students coming from other continents have the continent name assigned as their nationality area.



PT prog abbrv	PT prog
CS	Ciência de Computadores
EPP	Engenharia e Políticas Públicas
EEC	Engenharia Electrotécnica e Computadores
EEC	Engenharia Electrónica e Computadores
EE	Engenharia Eletrotécnica
EIC	Engenharia Informática e de Computadores
MA	Matemática Aplicada
MTE	Mudança Tecnológica e Empreendedorismo
ECE	NETSyS/ Engenharia Electrotécnica e Computadores
<null>	Programa Doutoral em Ciências da Informação e Tecnologia
-	Programa Doutoral em Engenharia Electrotécnica e de Computadores, especialização em Automação e Robótica
<null>	Programa Doutoral em Engenharia Eletrotécnica
<null>	Programa Doutoral em Engenharia Eletrotécnica e de Computadores
<null>	Programa Doutoral em Engenharia Informática
<null>	Programa Doutoral em Engenharia Informática e de Computadores
<null>	Programa Doutoral em Media Digitais

Figure 5.10: Result of the transformation to filter distinct study programs in Portugal from the table "Ph.D. Students"



Nationality Continent
Africa
Asia
North America
Portugal
Rest of europe
Romania
South America

Figure 5.11: Result of the transformation to filter distinct nationality areas of students from the table "Ph.D. Students"

5.3.1.E Approximate duplicate detection between the tables "Ph.D. Students" and "Faculty"

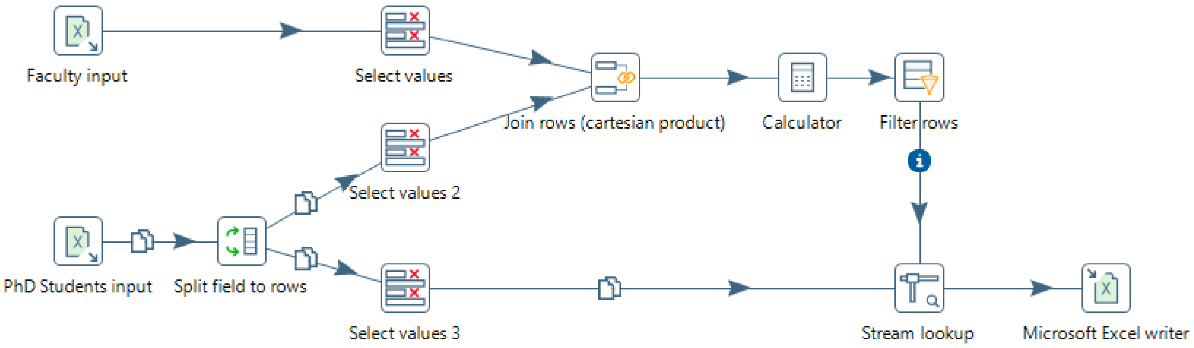
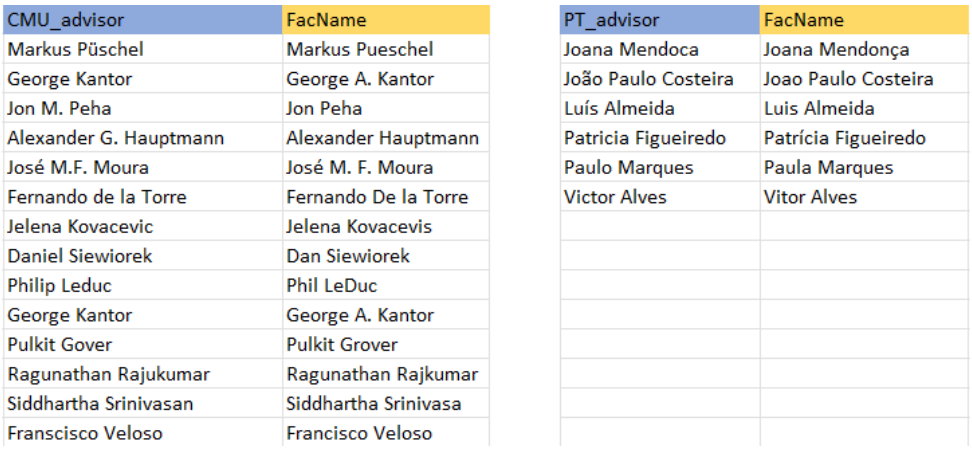


Figure 5.12: Transformation to detect students' advisors in the "Ph.D. Students" table that are not present in the "Faculty" table

The approximate duplicate detection between the tables “Ph.D. Students” and “Faculty” is performed in order to confirm if the names of the Ph.D. students’ advisors are equal to the names of these advisors in the “Faculty” table since the students’ advisors are almost always part of the faculty members and researchers of CMU Portugal. The only exception is advisors of former students who may no longer be part of CMU Portugal. In order to find the approximate duplicates, a transformation similar to the one in fig. 5.5 is performed. The difference is that as inputs, the tables “Ph.D. Students” and “Faculty” are chosen. In the case of the input table “Ph.D Students” there is a need to split fields into rows since students with more than one advisor have them in a single column separated with commas. The columns selected to be compared are the column that keeps the names of the faculty members and researchers in the “Faculty” table and the column of the “Ph.D. Students” table that keeps the names of the students’ advisors after splitting them into rows. This transformation is performed for the students’ advisors in Portugal and for the ones in CMU. Additionally, another transformation, depicted in fig. 5.12, is used to detect the student’s advisors in the table “Ph.D. Students” that are not present in the table “Faculty”. The “Stream lookup” step is used to look up the fields coming from the “Select values 3” step, which are the students’ advisors’ names, and find the ones that do not match with the fields coming from the “Filter rows” step. The fields coming from the “Filter rows” step are the names of faculty members and researchers in the “Faculty” table that match fully or approximately with the students’ advisors’ names in the “Ph.D. Students” table.

In fig. 5.13 it is possible to see the distinct students’ advisors’ approximate duplicates detected between the tables “Ph.D. Students” and “Faculty”. In blue are the columns that come from the “Ph.D. Students” table, and in yellow are the ones from the “Faculty” table. Additionally, forty students’ advisors’ names were not found in the “Faculty” table, some of them due to being written so differently in each table that these were filtered out by the similarity threshold.



CMU_advisor	FacName
Markus Püschel	Markus Pueschel
George Kantor	George A. Kantor
Jon M. Peha	Jon Peha
Alexander G. Hauptmann	Alexander Hauptmann
José M.F. Moura	José M. F. Moura
Fernando de la Torre	Fernando De la Torre
Jelena Kovacevic	Jelena Kovacevis
Daniel Siewiorek	Dan Siewiorek
Philip Leduc	Phil LeDuc
George Kantor	George A. Kantor
Pulkit Gover	Pulkit Grover
Ragunathan Rajukumar	Ragunathan Rajkumar
Siddhartha Srinivasan	Siddhartha Srinivasa
Franscisco Veloso	Francisco Veloso

PT_advisor	FacName
Joana Mendoca	Joana Mendonça
João Paulo Costeira	Joao Paulo Costeira
Luís Almeida	Luis Almeida
Patricia Figueiredo	Patrícia Figueiredo
Paulo Marques	Paula Marques
Victor Alves	Vitor Alves

Figure 5.13: Approximate duplicates between of students’ advisors’ names in the “Ph.D. Students” and “Faculty” tables

5.3.2 Data Cleaning

The data cleaning process was handled by employing a combination of direct alterations in the Excel file tables and the development of transformations with the Pentaho Data Integration tool. Besides that, communication with the CMU Portugal team was essential in order to know how to proceed with certain cases.

In the case of incomplete excel table entries, these were presented to the CMU Portugal team, which tried to fill in as many of the void table entries as possible.

Regarding the inconsistencies in the columns that keep the CMU study programs, the CMU departments, Portuguese study programs, and the student's nationality areas detailed in the Subsections 5.3.1.C and 5.3.1.D, direct alterations in the "Ph.D. Students" Excel table were done through Excel's find and replace feature. This solution was chosen since the number of instances requiring manual intervention was low, making it more efficient to handle them directly.

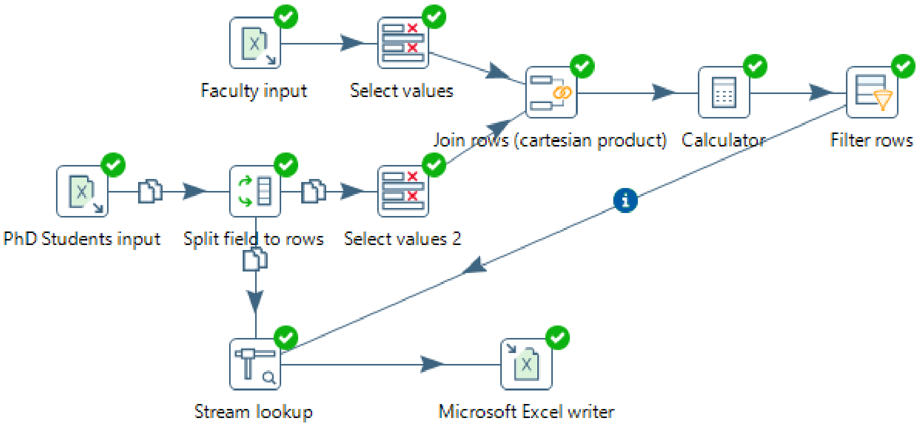


Figure 5.14: Transformation to help clean approximate duplicates of students' advisors' names between the "Ph.D. Students" and "Faculty" tables

In order to tackle the issues of approximate duplicates between the students' advisors' names in the "Ph.D. Students" and "Faculty" tables, the transformation depicted in fig. 5.14 was developed. This transformation detects the approximate duplicates between the two tables, as has been explained before. At the end of the transformation, the "Stream lookup" step takes all the records from the "Ph.D. Students" table and adds a new column, coming from the "Filter rows" step, which is the corresponding name of the students' advisors in the "Faculty" table. This happens for every row in which the students' advisors' names in the "Ph.D. Students" table do not match fully with the supposed advisors' names in the "Faculty" table. This extra column enables us to correct these inconsistencies in a more efficient way. However, there is still the issue of the students whose advisors' names are not found in the "Faculty" table. To solve this issue, it was necessary to search through the "Faculty" Excel table to find some of these names since they were written so differently that they were filtered out in the transformations that detect these inconsistencies. The help of the CMU Portugal team was also necessary to ensure the match between these names was done correctly. Besides that, the students' advisors' names that were not present in the "Faculty" table at all were filled in by the CMU Portugal team.

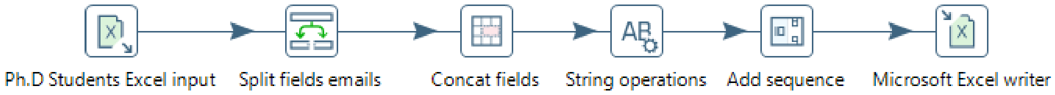


Figure 5.15: Transformation to prepare the "Ph.D. Students" table before migration

After cleaning the data, we further prepared it before migrating it to the database. For this, the transformation in fig. 5.15 was applied to the "Ph.D. Students" Excel table. We started off by splitting the column with the students' emails into two columns, since beforehand the emails of the students were stored all in one column, separated by commas. After that, in the "Concat fields" step, we merged the columns "Graduation date" and "Withdraw date" into one column named "End year". The "String operations" step was used to trim all the columns so there are no blank spaces at the beginning or end of the terms in the columns' fields. Finally, with the "Add sequence" step, we created a new internal sequential id of the type integer for every student.

5.3.3 Data Migration

Once again, the Pentaho Data Integration tool was used to migrate the data. Various transformations were developed to migrate the "Ph.D. Students" table's data into several tables of the database. To execute all these necessary transformations in a sequence of steps in a logical order, the job depicted in fig. 5.16 was developed. This job executes the specified transformations in order to migrate the data to the database with a single click. In fig. 5.16 the boxes with four green arrows represent a specific transformation like the ones shown in previous pictures, such as, for example, fig. 5.15.

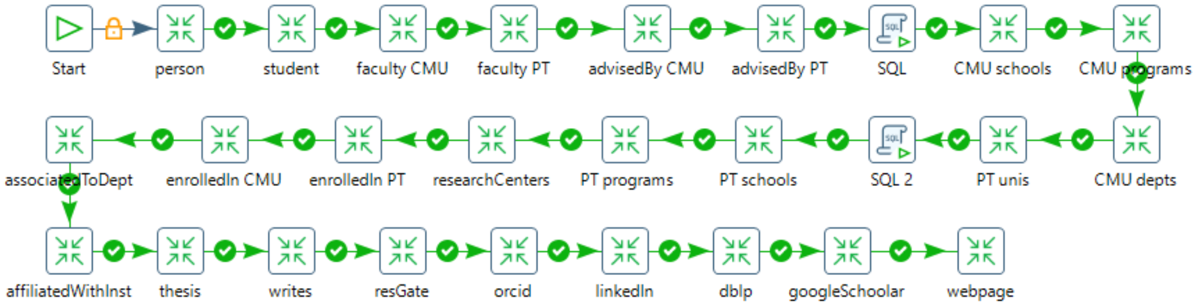


Figure 5.16: PDI job to run the transformations needed to migrate CMU Portugal's data to the database

First, the migration of doctorate students to the "Person" table takes place. This is done by having the "Ph.D. Students" table as input, selecting the columns with the names, genders, emails, and new ids of the students, and mapping these columns to the appropriate fields in the database with the “Table output” step. After that, on the “Student” step, the data related to the “Student” table of the database is migrated. In order to achieve this, a similar transformation as the one in the “person” step is performed. The difference is that there are two inputs: the “Ph.D. students” table and another Excel table with the students’ introduction text and research topics coming from the website. These two inputs are joined with the “merge join” step, with the condition that the names coming from one input match the ones coming from the other input. Afterward, the columns necessary for the “Student” database table are selected, and the records are inserted in this same table. The steps “faculty CMU” and “faculty PT” represent the transformation in which the students’ advisors are migrated to the “Person” database table. Since there is a column for the advisors at Portuguese universities and another for the advisors at CMU, two different but similar transformations are needed. Both these transformations have as input the “Ph.D Students” and “Faculty” Excel tables. The names of the students’ advisors are taken from the “Ph.D Students” table and singled out with the “Unique rows step”, then these names are joined with the data of the “Faculty” Excel table, in order to get their emails. The names and emails of the advisors are selected, a new internal sequential id of the type integer is created, and these records are inserted in the “Person” database table. It was decided to only migrate faculty members and researchers who are advisors to doctorate students and migrate them to the “Person” database table instead of the “FacultyResearcher” table. This is because the “Faculty” Excel table also has various data quality issues that have not been resolved, thus, in order to avoid data quality issues in the database, this decision was made. In the steps “advisedBy CMU” and “advisedBy PT”, transformations to migrate data to the “advisedBy” database table are performed. These transformations are very similar. They have as input the “Ph.D. Students” table and the “Person” database table. These inputs are joined to get the matching advisors’ and students’ ids. Before inserting these records into the database table, we need to add a new column that describes the “advisingType” of the “advisedBy” table, i.e., if the advisor is from a Portuguese institution or CMU. This is the difference between the transformations “advisedBy CMU” and “advisedBy PT”, in the first one, we use the step “Add constants” to insert a column with the term “CMU” and in the latter, the term “PT”. The following “SQL” step inserts CMU in the “institution” database table with the type “University”. The step “CMU schools” refers to the transformation to migrate the schools of CMU to the “School” database table. This transformation consists of selecting the column that keeps the schools in CMU and getting only distinct schools by using the “Unique rows” step. Afterward, a new column with the term “Carnegie Mellon University” is inserted for every school and these records are migrated to the “School” database table. The transformations corresponding to the “CMU programs” and “CMU depts” steps are the migrations of the CMU programs and departments into the “Program” and “Department” database tables, respectively. These transformations are very similar to the one in the step “CMU schools”, with the only difference being that we need to select not only the department names, program names, and program abbreviations but also the schools and university associated with them. The transformation in the step “PT unis” consists of selecting the column with the names of the Portuguese universities, using “Unique Rows” to select only distinct universities, and adding a column through “Add constant” with the institution type, which is the term “University” for all rows. Finally, we use “Table output” to insert these records in the “Institution” database table. The “SQL 2” step served to add the Portuguese universities’ abbreviations to the “Institution” database table. The transformations of the steps “PT schools” and “PT programs” are equal to the “CMU schools” and “CMU Programs” transformations, with the difference of not needing to add a new column. These transformations migrate the Portuguese schools with their respective university affiliations into the “School” database table and the study programs in Portuguese schools associated with Portuguese universities into the “Program” database table. The transformation in the “researchCenters” step is the same as the one in the “PT unis” step, in which the records are migrated to the “Institution” database table, with the difference of the added column containing the term “Research Center” in all rows. The “enrolledIn PT” and “enrolledIn CMU” steps consist of the transformations to migrate data to the “enrolledIn” database table. The transformations are similar, in that the students’ ids, study programs, schools, and universities they are enrolled in are selected. A new column is added with “Add constants” to fill out the enroll type of the “enrolledIn” database table, which describes if the enrollment is in a Portuguese university or CMU. After that, the records are inserted in this same database table. The steps “associatedToDept” and “affiliatedWithInst” are the transformations that respectively migrate the students’ affiliation to CMU departments and to any institutions, such as the research centers where they conduct their research during their Ph.D. These transformations are very simple, where the columns, such as the student’s id and their department at CMU or research host institution, are selected and respectively inserted into the “associatedToDept” or “affiliatedWithInst” database table. The “thesis” step refers to the transformation that migrates students’ theses to the “Thesis” database table. The students’ theses and theses’ links are selected, empty values are filtered out, and an internal id of the type integer is created for every thesis before inserting it into the “Thesis” database table. The “writes” step consists of the transformation to migrate the data into the “writes” database table. In this transformation, there are two inputs: the “Ph.D. Students” Excel table and the “Thesis” database table. These are joined on the theses’ titles, next, the matching students’ and the theses’ ids are selected and inserted into the “writes” database table. Finally, the remaining steps include the migration of the students’ online profiles into the “Profile” table. These profiles are each stored in different Excel table columns, with the name of the profile being the column name and the links to the students’ profiles being in the rows of the respective column. The individual transformations to migrate the students’ different profiles are similar. The students’ ids and links of a specific profile are selected, the rows with empty profile links are filtered out, and a new column is added with “Add constants” to add the profile type, such as “LinkedIn”, “GoogleScholar”, etc., to the “Profile” database.

5.4 User interface

A user interface was created so that the database's users can easily utilize it without needing to resort to SQL queries. With this interface, users can list, create, modify, and remove students. Besides that, it is possible to apply filters to search for or view only certain students.

The system's logic consists of a layered architecture in which the presentation layer deals with presenting information to the users and letting them interact with this information. The layer underneath the presentation layer is composed of a merge between the business layer, which handles aspects related to accomplishing functional requirements, and the data layer, which is responsible for operating data and the database. There is not a lot of business logic in this system since its main objective is to let users manipulate data from the database. However, there are some logical verifications that need to be made before using SQL queries on the database, such as, for instance, if a user tries to insert a student enrolled in a school that is not already in the database, it is necessary to insert this new instance of school with an associated university instead of just using one that already exists. Besides that, when inserting students, since some data is not mandatory to fill in, it is also necessary to check for null values in order not to enter these in the database, among other things. Apart from that, when it comes to handling the database, a connection is established with the database, which is then used to execute SQL queries in this database and return the results.

In the subsections below, we provide an outline of the interface's functionalities.

5.4.1 Search students

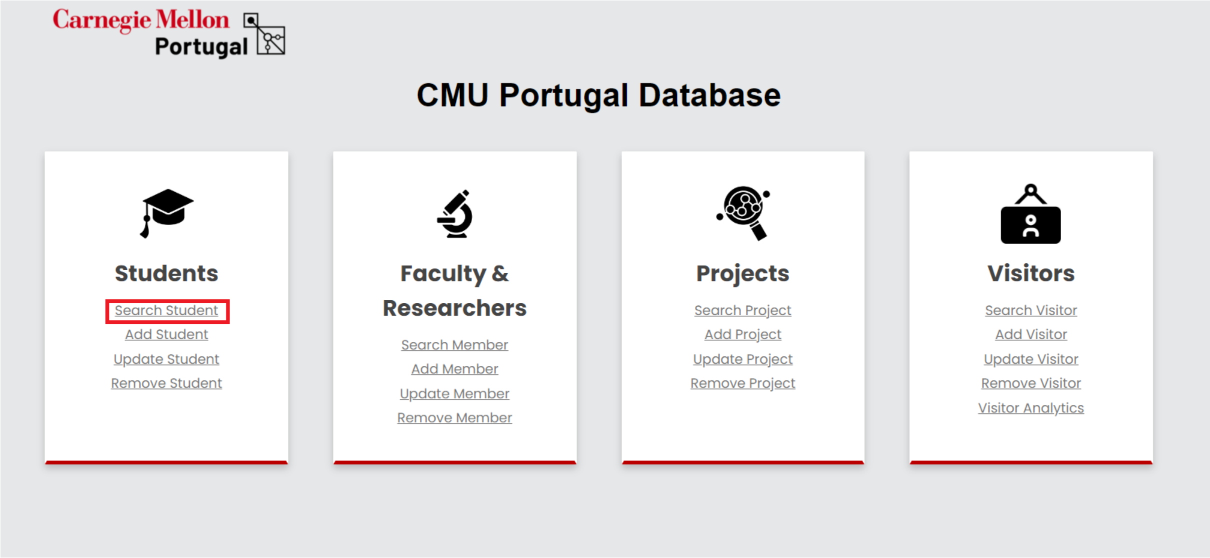
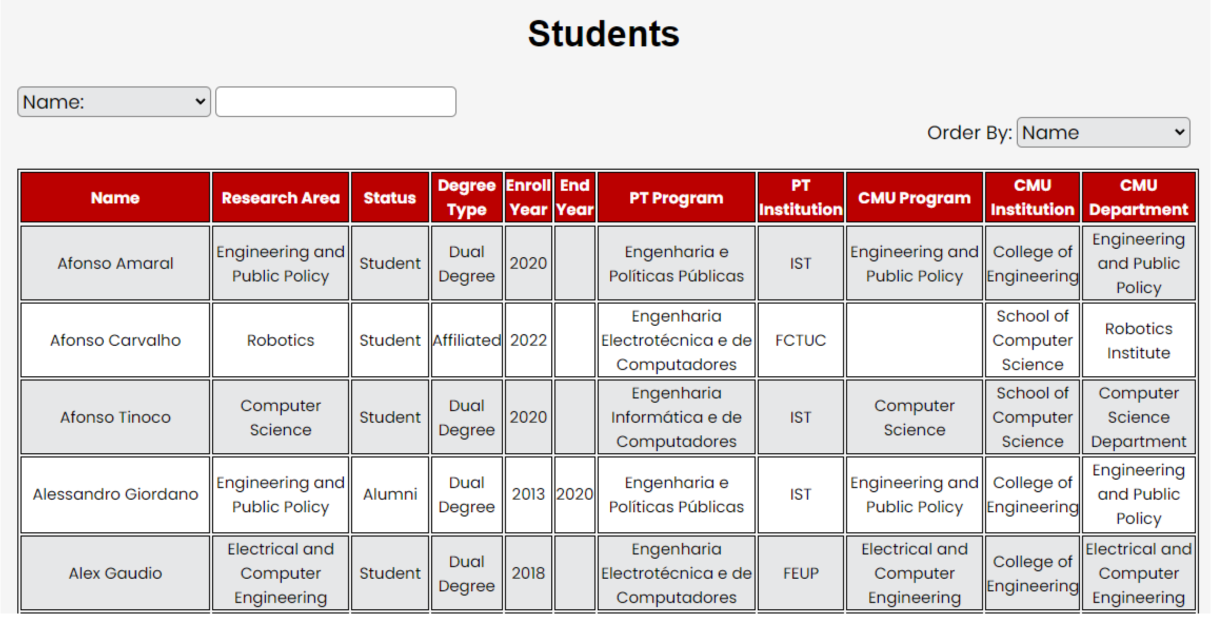


Figure 5.17: Home page of the CMU Portugal database user interface

Upon entering the user interface, the user sees the contents of the fig. 5.17. So far, only the functionalities concerning the students have been implemented. By clicking on “Select Students”, underlined in red in fig. 5.17, they are presented with a list of all doctorate students, along with their research area, status, degree type, enrollment, and end year, and their affiliations to Portuguese universities and CMU as can be seen in the cropped image of this list in fig. 5.18.



Students

Name: ▾

Order By: Name ▾

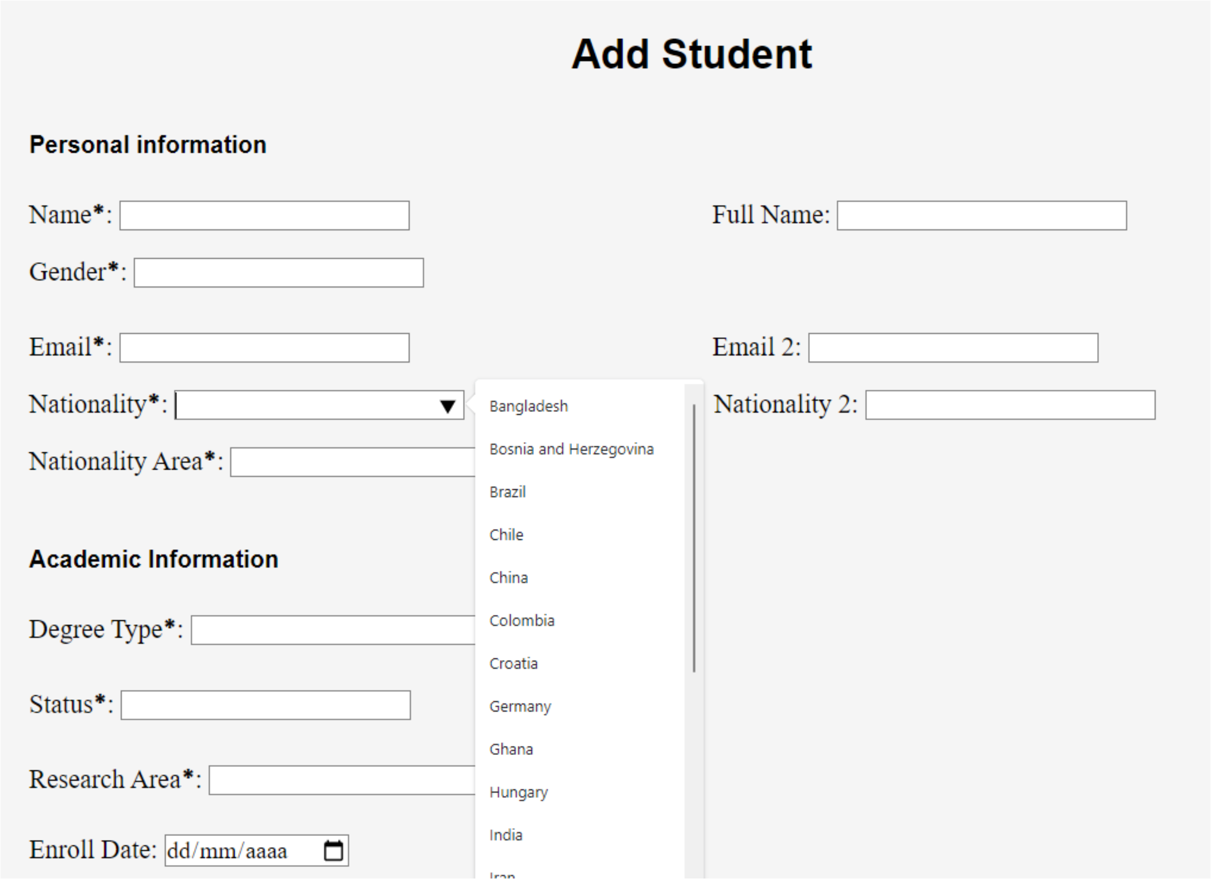
Name	Research Area	Status	Degree Type	Enroll Year	End Year	PT Program	PT Institution	CMU Program	CMU Institution	CMU Department
Afonso Amaral	Engineering and Public Policy	Student	Dual Degree	2020		Engenharia e Políticas Públicas	IST	Engineering and Public Policy	College of Engineering	Engineering and Public Policy
Afonso Carvalho	Robotics	Student	Affiliated	2022		Engenharia Electrotécnica e de Computadores	FCTUC		School of Computer Science	Robotics Institute
Afonso Tinoco	Computer Science	Student	Dual Degree	2020		Engenharia Informática e de Computadores	IST	Computer Science	School of Computer Science	Computer Science Department
Alessandro Giordano	Engineering and Public Policy	Alumni	Dual Degree	2013	2020	Engenharia e Políticas Públicas	IST	Engineering and Public Policy	College of Engineering	Engineering and Public Policy
Alex Gaudio	Electrical and Computer Engineering	Student	Dual Degree	2018		Engenharia Electrotécnica e de Computadores	FEUP	Electrical and Computer Engineering	College of Engineering	Electrical and Computer Engineering

Figure 5.18: Cropped image of the CMU Portugal’s students list

As we can see in fig. 5.18 at the top of the image, below the title “Students” and above the table, there are two filtering options. The one on the left side can filter the table by all of its columns. The columns can be selected on the drop-down menu, with the default column being “Name”, as seen in fig. 5.18. The white input box serves the purpose of writing the value the user wants to filter in that column, for example, a student name, or a certain research area, etc. The filter on the left lets us order the table by the students’ names or by year of enrollment. By clicking on a student’s row, we are redirected to another page with that student’s profile. Besides the already displayed information, this profile page contains additional information on the students, such as their email(s), nationalities(s), number of enrollments, advisors, research institution if applicable, information on their thesis, and online profiles.

5.4.2 Add Student

On the home page of the user interface (see fig.5.17) by clicking on “Add Student” we are redirected to another page that contains a large form that can be used to insert a new student. A part of this form can be seen in fig. 5.19.



Add Student

Personal information

Name*:

Full Name:

Gender*:

Email*:

Email 2:

Nationality*:

Nationality 2:

Nationality Area*:

Academic Information

Degree Type*:

Status*:

Research Area*:

Enroll Date: dd/mm/aaaa

Figure 5.19: Cropped image of the “Add Student” page of the user interface

This form contains forty-three fields, with eighteen of them being mandatory to fill in. At the bottom, there is a submit button that cannot be seen in fig. 5.19. Functions were developed so that when a user is writing in an input field, a drop-down menu with values already in the database for that field appears, as can be seen in fig. 5.19 for the “Nationality” field. This was done for all possible fields. Drop-down menus with options like this allow users to make fewer mistakes when inserting data since they can just select an option as opposed to having to constantly spell out every word that needs to be inserted. As we have seen in Section 5.3 a lot of CMU Portugal’s data quality issues stemmed from typing mistakes and non-adherence to certain term conventions. This problem is diminished by the presented solution. Besides that, on submitting the form, if there are any empty mandatory fields, the form does not get submitted, and a pop-up shows up next to the empty field to inform the user that the field needs to be filled in. When submitting the form, the user is asked before proceeding to confirm if they are certain they want to submit the form or go back and edit it again. Once the form is successfully submitted, a message appears to inform the user that the student has been inserted in the database, and a button to go back to the home page emerges.

5.4.3 Update Student

When users make mistakes inserting students or simply want to update a student's data, the option "Update Student" on the home page can be used. When clicking it, we are redirected to another page with a form equal to the one depicted in fig. 5.19 and with the same fields. There the user must provide the name of the student they wish to change and are instructed to fill in only the fields they wish to alter. In this form, drop-down menus are also provided for all possible fields so that users can select out of the options instead of needing to type in the fields. When submitting this form, the user is also asked before proceeding if they want to go forward or re-edit the form. Once a viable student name to alter and the student's alterations are provided, the user may click on the submit button, which updates the selected student's fields in the database.

5.4.4 Remove Student

By clicking on the "Remove Student" option on the home page of the user interface, the user is allowed to delete a student of their choice from the database. A cropped image of this page can be seen in fig. 5.20. The user is presented with a field to input the student they wish to remove, which they can either type in or choose from the available options. It is not possible to submit without providing a student's name. After submitting, if the user provided a student's name that is not in the database, a message appears informing the user that the student was not removed due to the input having the wrong student name. Otherwise, the student is successfully deleted from the database.

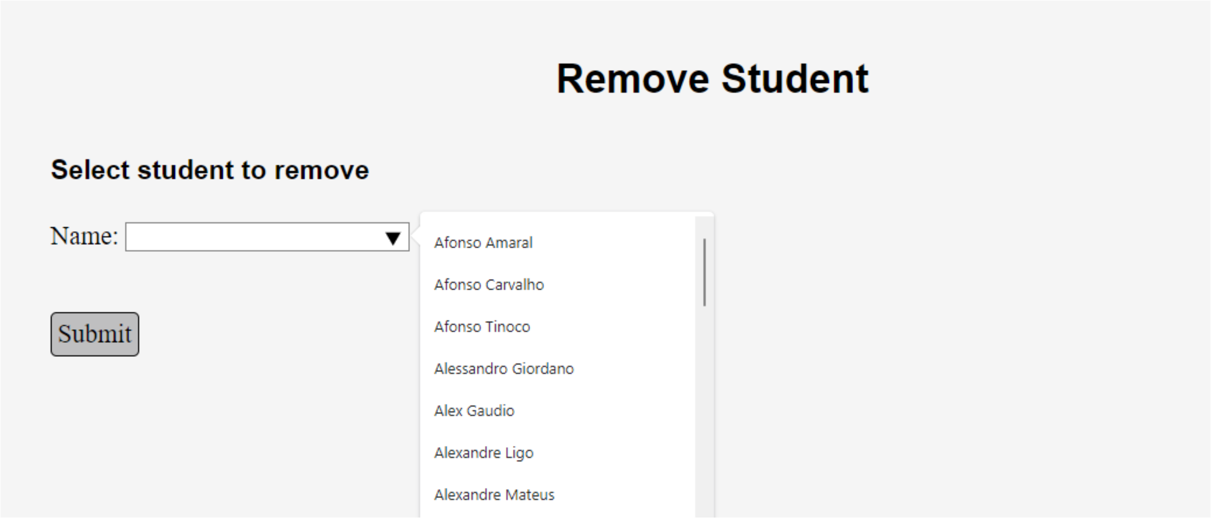


Figure 5.20: Cropped image of the "Remove Student" page of the user interface

6

Work Evaluation

Contents

6.1 Quality Assurance in the Data Migration 55

6.2 Database Performance 57

6.3 User testing 59

This chapter focuses on assessing and analyzing the quality of various aspects related to the implementation of the database system and its user interface. In this chapter, we analyze three key points: quality assurance in the data migration from the source Excel files of CMU Portugal into the database, the performance of this database, and finally, we test our user interface with users to get their feedback and understand if the platform is intuitive and adequately built.

6.1 Quality Assurance in the Data Migration

When migrating data, it is important to take into account the risk of compromising its quality. Data migration often involves moving large volumes of data from one system to another, for this reason, oversights can result in missing or incomplete data during the transfer. Inaccurate data mapping, which involves matching data fields between the source and target systems, can also result in incorrect data associations, data type mismatches, or loss of data integrity. For this reason and in accordance with Matthes et al. [18] the most typical data migration risks are:

- Completeness risk: the risk of losing one or more business objects during the migration of the data and the occurrence of additional business objects appearing in the target application that were absent in the source.
- Semantics risk: the risk of a business object that has been migrated to change semantics, for example, a student whose status is “Alumni” changes to “Withdraw”.
- Data corruption risk: the risk that the target application’s data model is not reflected in the transferred data. An example could be an institution other than a “University” such as a research center or company, that appears with schools, departments, or programs associated with it. This should not be possible, but it is also not prevented by the database.

In order to assure that there are no faults in the migrated data, we use testing techniques for data migration. In terms of the data corruption risks mentioned above, these have already been mitigated by the data profiling and cleaning methods used, described in Section 5.3. In the case of completeness and semantic risks, we use data validation techniques to ensure the completeness and semantical correctness of the migrated data. Such data validation techniques include completeness and type correspondence tests [19], which consist of looking at numerous business objects by automating the comparison of these objects coming from the source files and the target database. In order to perform these tests, Pentaho Data Integration is used, in which transformations are created to compare the records in the database to the records in the “Ph.D. Students table” to verify no records were lost or changed.

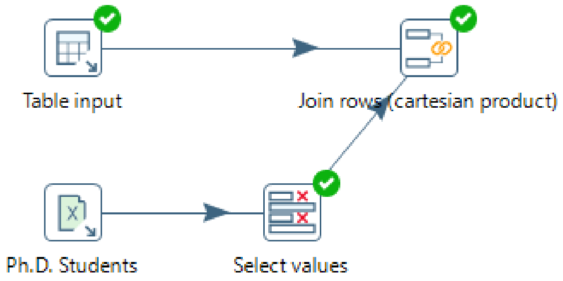


Figure 6.1: Transformation to ensure the data completeness and semantical correctness in the “person” database table

First, the transformation depicted in fig. 6.1, is used to compare the records in the “person” database table with the ones from the “Ph.D. Students” table. In the “Table input” step, we select all records from the “person” database table. The “Select values” step selects the students’ ids, names, full names, genders, and emails from the “Ph.D. Students” Excel table, which are the attributes of the “person” database table. Finally, the “Join rows” step is used to join both these inputs on the condition that the ids, names, full names, genders, and emails coming from the “Ph.D. Students” Excel table must match the ones coming from the “person” database table. After that, the “Join rows” step is previewed, which shows us all the rows that have matched the condition. When previewing the “Join rows” step we are able to see that all 174 students are present in this table, which means the data in the “person” database table has been successfully migrated without loss or alterations to the data. A transformation similar to this one was performed for the “student” database, which also successfully outputted all the students. Two different transformations were also performed to confirm the successful data migration to the “enrolledIn” table, one for the enrollments in Portuguese universities and another for enrollments in CMU, which did not point out any mistakes. These transformations also consisted of a direct comparison between the specified columns in the “Ph.D. Students” Excel table and the respective attributes in the database table, observing if the number of outputted rows is correct. Two transformations following the same logic were also needed to test the data in the “advisedBy” database table because students have advisors from Portuguese universities and CMU. No quality issues were found. Two similar transformations were executed to test the data quality in the tables “associatedToDept” and “affiliatedWithInst”, with no data problems found. Lastly, a similar transformations as the one performed for the “person” database table were also performed for the “writes” and “profile” database tables, with no mistakes identified.

In conclusion, with these tests, we are able to verify that no faults appeared in the data due to the data migration process. As such, we guarantee the data quality of the database is up to the same standards as the source data after the profiling and cleaning processes.

6.2 Database Performance

According to Han et al. [20], when benchmarking datasets, the volume of the data, the velocity with which it can be retrieved and generated, and the variety of its diversity in data types, structures, and sources are what define the data. Taking this into account, the data in our database, which performance we wish to test, does not have a lot of variety since it has only one structure and one source, which are the database’s structure and the database itself, respectively. However, with an analysis of the performance of the developed database regarding the characteristics of velocity and volume of data, we wish to ensure that the database responds promptly when users interact with it. In order to achieve this, we developed a group of queries, identifying relevant workloads, that portrayed the typical behavioral needs the database would need to respond to [20]. These queries are presented in Appendix B. The group of chosen queries is divided into three categories:

1. Queries necessary for the user interface to run as intended

2. Analytical queries described in Section 5.1 that are used by the CMU Portugal team to run annual

reports

3. Queries necessary for the CMU Portugal's website to function

Table 6.1: Duration and records retrieved for each query

	Query duration in seconds	**Records retrieved**
SQL query 1	0.00887300	1914
SQL query 2	0.00080600	870
SQL query 3	0.00052400	1392
SQL query 4	0.00009350	16
SQL query 5	0.00053625	10
SQL query 6	0.00062375	84
SQL query 7	0.00123975	201
SQL query 8	0.00081100	201
SQL query 9	0.00077425	16
SQL query 10	0.00053425	4
SQL query 11	0.00068350	12
SQL query 12	0.00040800	30
SQL query 13	0.00061375	232
SQL query 14	0.00063175	148
SQL query 15	0.00093050	87
Average	0.00120555	348

In table 6.1, queries one to five are queries used in the user interface, with the first three being chosen purposefully since these retrieve the higher amount of records. The following seven queries, queries six to twelve, represent analytical queries applied to the students, described in Section 5.1. These queries may not return a large number of records, but they all involve retrieving the count of a specific group of students grouped by, for example, their degree type, enrollment year, status, study program, and more, or a combination of various of these attributes. When choosing this subset of queries, we mostly tried to select the ones that involved grouping students based on a large number of attributes, and additionally, we prioritized queries that involved grouping the largest pool of students. The last three queries, thirteen through fifteen, are queries related to the functionality of CMU Portugal's website. Although the developed database is not currently serving as a data source for the website, we decided to test some queries that would potentially support the website's functioning and at the same time retrieve the most records.

After selecting the group of queries to test, we ran each of them three times and got the average time in seconds it took for each of these queries to execute. These results can be seen in table 6.1. We also calculated the average response time of the database based on this group of queries, which is 0.00119706 seconds. Taking all of this into consideration, the results obtained in table 6.1 show us the database response time for the average query is adequate, being in the millisecond range. This ensures that the users, who are the major executors of these queries, won't be kept waiting or have slow interactions with the user interface, which is the main goal.

6.3 User testing

In order to evaluate the developed user interface, we decided to have users test our system. In order to achieve this, we developed a set of four tasks the users should follow that cover all of the implemented functionalities of the user interface. The first task involves exploring the "Search Student" functionality, the second task the "Add Student" functionality, the third the "Update Student" functionality, and the fourth the "Remove Student" functionality. These tasks are further described in Section C.1. We timed the duration for each user to complete each task as an evaluation measure. For the interaction of users with the interface, sensitive information of CMU Portugal's students that should not be disclosed to the public was omitted from the platform. After that, we asked the users to fill out a two-part questionnaire, in which the first part consists of their demographic data. The second part consists of user ratings of the user interface based on a set of heuristics and the detailing of issues in the platform, along with their rating of these issues on a severity scale [21]. The users were handed a description of these heuristics and of the severity scale to answer the questionnaire, which are described below and also detailed in section C.3. The questionnaire is detailed in section C.2.

The heuristics chosen for the users to evaluate the user interface are based on the article by Dowding and Merrill [22]. From the ten heuristics presented in the article, we chose six that we considered to be the most applicable in the context of our user interface, which are the following:

1. Visibility of system status: The system should always update the user on what is happening by providing suitable feedback in a timely manner. Important characteristics are that every screen has a title or header to describe its contents, a consistent stylistic treatment across the system is applied, a clear indication of the current location exists, and the menu-naming terminology is consistent within the users' task domain.

2. Match between system and the real world: The system should speak the user's language, with words, phrases, and concepts familiar to the user, rather than system-oriented terms. It should follow real-world conventions, making information appear in a natural and logical order. Important characteristics are that section headings and subsections in each screen are ordered in a logical

way, there is a natural sequence to the menu choices for a data item, and that selected colors correspond to common expectations about color codes.

3. User control and freedom: Users should be free to select and sequence tasks, when appropriate, rather than having the system do this for them. Important characteristics are that there are clear exits on each document screen, all screens are accessible across the system, there is an ”undo” function, and users can easily move forward and backward.

4. Recognition rather than recall: The user should not have to remember information from one part of the dialogue to the next. Instructions for the use of the system should be visible or easily retrievable whenever appropriate. Important characteristics are that prompts, cues, and messages are placed where the user is likely to be looking on the screen, white space are used to create symmetry and lead the user in the appropriate direction, have items grouped into logical zones, and headings to distinguish between zones.

5. Consistency and standards: Users should not have to wonder whether different words, situations, or actions mean the same thing. Important characteristics are that formatting standards are followed consistently on all screens, each window has a title, there are no more than four to seven colors, and names are consistent, both within each tab and across the system, in grammatical style and terminology.

6. Aesthetic and minimalist design: Information that is unnecessary or rarely used shouldn’t be included in dialogues. Important characteristics are that only information essential to decision-making is displayed on the screen, large objects and bold fonts are used to distinguish sections, field labels are brief, familiar, and descriptive, and the visual layout is well designed.

The severity scale consists of a scale going from one to four that rates whether an issue found in the user interface needs to be addressed or not. This scale can be described as follows:

- **0**: I completely disagree that this is a usability issue.
- **1**: Cosmetic problem, needs to be rectified only if more time is available to complete the project.
- **2**: Minor usability problem, fixing this should be given low priority.
- **3**: Major usability problem that is important to fix should be given high priority.
- **4**: Usability catastrophe, imperative to fix before the product is launched.

6.3.1 Results

Five users, who had never before seen or interacted with the user interface, participated in our study. These users are all between the ages of twenty-four and thirty-five, with three of them being female

Table 6.2: Duration in minutes of the time taken for each user to complete each task

	Task 1	**Task 2**	**Task 3**	**Task 4**
User 1	01.18	01.32	01.40	00:20
User 2	01.20	02.16	01:25	00:25
User 3	01.00	02.11	01.22	00.23
User 4	01.08	02.30	01.30	00.21
User 5	01.06	01.27	01.12	00.15
Average	01.10	01.59	01.26	00.21

Table 6.3: Average points and percentage given by users to the user interface based on the heuristics

Heuristic	**Average points on 1 to 5 scale**	**Average percentage**
Visibility of system status	4.8	96%
Match between system and the real world	5	100%
User control and freedom	3.6	72%
Recognition rather than recall	4.8	96%
Consistency and standards	5	100%
Aesthetic and minimalist design	5	100%

and the other two male. All participants except one had completed a master's degree, while the other had completed a bachelor's degree. Four users considered themselves familiarized when dealing with technological platforms, while the other one considered himself or herself an expert.

Table 6.2 displays the duration, measured in minutes, taken by each user to complete each task, along with the average duration for each task. Each of these tasks was designed to be a simple task, and according to Nielsen Norman Group [23] simple tasks should take about one minute for users to complete. Analyzing the average time taken to complete each task, we can see that this average is a little over the one-minute mark. However, while conducting these tasks, the users had to read at the same time the guide provided to them, which contained the description of each task. Besides that, task two had the highest average time, since it involved filling in a form with around nineteen mandatory fields, which context the users did not fully understand, although options were provided for these fields. Taking all of this into consideration, we deem that the average duration taken by the users to complete each task is adequate, which implies that our user interface is simple and easy to use.

In table 6.3 we can analyze the ratings the users gave the user interface on a scale of one to five, with one being fully disagreeing and five fully agreeing, based on the provided heuristics. It is possible to observe that in the eyes of the users, the user interface was in accordance with all heuristics except one, the "User control and freedom" heuristic. When comparing these results to the feedback from the users' issues with the platform, we can understand why. Three users reported the need for a button to go back other than the one provided by the browser or a button that would take them to the home page on all screens. Of these users, all of them rated this issue a two on the severity scale. This means this issue is of low priority, being a minor usability issue. In conclusion, although the issue identified by the users should be fixed, it is not a critical problem that impairs the use of the user interface. Besides that, the requirements of the remainder of the heuristics were met, making this user interface an intuitive and user-friendly platform.

7

Conclusion

Contents

7.1 Summary and Conclusions 63
7.2 System Limitations and Future Work 65

In this chapter, we make a summary of the work done and draw some final conclusions. The limitations of the work are then described, along with suggestions for what should be done as additional research in the future.

7.1 Summary and Conclusions

This document presented an overview of the development and implementation of a database for CMU Portugal and a user interface so users can list, create, modify, and delete records in this database without resorting to SQL queries. The problem definition was outlined, highlighting the need for an efficient and reliable database to manage the CMU Portugal's data, which integrated its various data sources.

A small chapter, chapter 2, is dedicated to explaining and giving context to what CMU Portugal is and the initiatives it is involved in.

The background chapter, chapter 3, provided insights into the CMU Portugal's data, including data sources originating from Excel files and the CMU Portugal website. In this section, a brief description of CMU Portugal's data quality problems is also presented.

The related work of this paper, chapter 4, focuses on presenting an analysis and discussion of various tools that could be used in the processes of choosing a database management system, analyzing and integrating the CMU Portugal's data, and developing a user interface for the database. Specifically, MySQL was identified as the chosen database management system, the Pentaho Data Integration tool and its plug-in DataCleaner were chosen for the profiling, cleaning, and migrating of the data, and HTML, CSS, Javascript, and PHP were utilized for the user interface development.

In chapter 5 we present the design and implementation of the database for CMU Portugal and its user interface following a requirements-gathering process. An entity-relationship and relational model are developed in order to represent the data in a logical and structured way, detailing its distinct objects and the interconnections between them. Data profiling, cleaning, and migration techniques are employed to identify CMU Portugal data's inaccuracies and inconsistencies, fix these issues, and migrate the cleaned data into the database. Additionally, the user interface's logic and key functionalities are described, such as searching, adding, updating, and removing student records. These features aim to enhance the accessibility of the database for its intended users.

The work evaluation chapter, chapter 6, focuses on quality assurance in data migration, database performance, and user testing. Testing techniques, such as completeness and type correspondence tests are used to verify the accuracy and integrity of the migrated data when comparing it with the source data. Database performance testing was conducted to assess the efficiency and responsiveness of the system, ensuring optimal performance under various load conditions. User testing was also performed in order to analyze if the developed user interface for the database is intuitive and simple and easy for the users to utilize.

Overall, the implementation of the database for CMU Portugal and its user interface addressed the problem at hand and provided a favorable and user-friendly solution for managing CMU Portugal's data. The goal of creating a database exempt from the data issues coming from CMU Portugal's various sources that integrated both Excel files and data coming from the website was achieved. The requirements needed for the users' interaction with the database, i.e the user interface, and the website were met, with the database not yet serving as a source to the website but meeting the requirements to do so. Through completeness and type correspondence tests to analyze the data quality in the migration of CMU Portugal's data, a performance evaluation of the database through measuring the duration and retrieved records of queries, and user testing, the developed work demonstrated its effectiveness in meeting the needs of CMU Portugal and its users. The paper highlighted the importance of data quality, database performance, and an intuitive user interface to ensure a reliable and efficient system.

7.2 System Limitations and Future Work

This chapter explores further areas in which the system can be developed and improved. We will focus on identifying the main system's limitations, identifying areas that require additional attention, and presenting suggestions for future enhancements that can lead to the maximization of the system's effectiveness, functionality, and user satisfaction.

Our system's biggest limitation is the fact that only data related to the students participating in CMU Portugal's educational program has been profiled, cleaned, and migrated to the database. Therefore, as future work, it would be essential to profile, clean, and migrate to the database the data included in the "AlumniPositions", "Faculty", "Projects", "Visiting Faculty", and "Visiting Students" Excel tables. Nevertheless, there is the advantage that the database structure and code have already been developed considering all the data coming from all CMU Portugal's existing sources, and some data quality issues have already been outlined.

Besides that, it is necessary to add the database as a source for CMU Portugal's website. This way the website is always consistent with the CMU Portugal's database, and the users only need to alter or insert new data in the database, which would update the website, removing the need to manipulate data in both the website and the database.

Furthermore, the following additional features would allow to enhance the system's capabilities and user experience:

- Security measures: There could exist an authentication step before entering the user interface, where it is possible to change database records. The CMU Portugal team also suggested a feature to allow non-authenticated users, such as guests who are not part of the CMU Portugal team, to be able to interact with certain parts of the user interface but not be able to create, remove, or modify the data records nor access sensitive information about the students.

- Add a functionality in the user interface that allows the users to insert, alter, or remove institutions, such as universities, research centers, or companies: In the case of universities, allow the users to change schools, departments, and programs associated with the universities. At this time, when inserting a student, users are allowed to affiliate students with institutions that already exist in the database or insert a new institution that does not exist in the database. Newly inserted institutions will appear as an option after being inserted in the database for the first time. However, there have been times when the CMU Portugal, has wanted to change, for example, the abbreviation of a school, such as changing the abbreviation of ”IST” to ”Técnico”. With the way the system is implemented now, CMU Portugal's team members would have to change every student with this school abbreviation one at a time to complete this action. With the proposed functionality, users could have access to all institutions associated with CMU Portugal and freely alter their names or abbreviations, and in the case of the universities, add schools, add new departments, add new programs, add existing programs to other departments, and so forth. This would improve the users' flexibility in this area when using the user interface and probably also reduce errors since users would have a section where they could view only the institutions affiliated with CMU Portugal and correct errors if needed.

Bibliography

[1] CMUPortugal, “CMU portugal annual report 2018/2019,” 2019. [Online]. Available: https://www.cmuportugal.org/wp-content/uploads/2020/09/Relatorio_CMU.pdf

[2] D. D. Brad Williams and H. Stern, *Professional WordPress: Design and Development*. Wrox, 2013.

[3] kinsta, “A beginner’s guide to wordpress database: What it is and how to access it,” visited on 2023-04-24. [Online]. Available: https://kinsta.com/knowledgebase/wordpress-database/

[4] H. K. Abraham Silberschatz and S. Sudarshan, *Database System Concepts*. McGraw-Hill, 2010.

[5] S. A. Mostafa, S. W. AbuSalim, and M. Z. Saringat, “A comparative study of data management systems,” *Journal of Soft Computing and Data Mining*, vol. 1, no. 1, p. 10–16, Mar. 2020. [Online]. Available: https://publisher.uthm.edu.my/ojs/index.php/jscdm/article/view/6826

[6] T. P. G. D. Group, “PostgreSQL documentation website,” visited on 2023-04-28. [Online]. Available: https://www.postgresql.org/docs/current/intro-whatis.html

[7] Oracle, “Introduction to oracle database,” visited on 2023-04-28. [Online]. Available: https://docs.oracle.com/en/database/oracle/oracle-database/19/cncpt/introduction-to-oracle-database.html

[8] DB-Engines, “Db-engines ranking of relational dbms,” visited on 2023-04-28. [Online]. Available: https://db-engines.com/en/ranking/relational+dbms

[9] R. Poljak, P. Poščić, and D. Jakšić, “Comparative analysis of the selected relational database management systems,” in *2017 40th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, 2017, pp. 1496–1500. [Online]. Available: https://doi.org/10.23919/MIPRO.2017.7973658

[10] F. Naumann, “Data profiling revisited,” *SIGMOD Rec.*, vol. 42, no. 4, p. 40–49, feb 2014. [Online]. Available: https://doi.org/10.1145/2590989.2590995

[11] S. C. Fernandes, “Integrating approximate duplicate detection into pentaho data integration,” 2019.

[12] R. Mukherjee and P. Kar, “A comparative review of data warehousing etl tools with new trends and industry insight,” in *2017 IEEE 7th International Advance Computing Conference (IACC)*, 2017. [Online]. Available: https://doi.org/10.1109/IACC.2017.0192

[13] M. Badiuzzaman Biplob, G. A. Sheraji, and S. I. Khan, “Comparison of different extraction transformation and loading tools for data warehousing,” in *2018 International Conference on Innovations in Science, Engineering and Technology (ICISET)*, 2018, pp. 262–267. [Online]. Available: https://doi.org/10.1109/ICISET.2018.8745574

[14] A. Singh and J. Singh, “A comparative review of extraction, transformation and loading tools,” 06 2018.

[15] S. Sridevi, “User interface design,” *International Journal of Computer Science and Information Technology Research*, vol. 2, no. 2, pp. 415–426, 2014.

[16] S. Trent, M. Tatsubori, T. Suzumura, A. Tozawa, and T. Onodera, “Performance comparison of php and jsp as server-side scripting languages,” in *Middleware 2008*, V. Issarny and R. Schantz, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 164–182.

[17] D. Ghimire, “Comparative study on python web frameworks: Flask and django,” 2020.

[18] F. Matthes, C. Schulz, and K. Haller, “Testing & quality assurance in data migration projects,” in *2011 27th IEEE International Conference on Software Maintenance (ICSM)*, 2011, pp. 438–447. [Online]. Available: https://doi.org/10.1109/ICSM.2011.6080811

[19] K. Haller, “Towards the industrialization of data migration: Concepts and patterns for standard software implementation projects,” in *International Conference on Advanced Information Systems Engineering*, 2009.

[20] R. Han, L. K. John, and J. Zhan, “Benchmarking big data systems: A review,” *IEEE Transactions on Services Computing*, vol. 11, no. 3, pp. 580–597, 2018. [Online]. Available: https://doi.org/10.1109/TSC.2017.2730882

[21] D. G. Manuel J. Fonseca, Pedro Campos, *Introdução ao Design de Interfaces*. FCA, 2017.

[22] D. Dowding and J. A. Merrill, “The development of heuristics for evaluation of dashboard visualizations,” *Applied clinical informatics*, vol. 9, no. 03, pp. 511–518, 2018.

[23] Nielsen Norman Group, “Powers of 10: Time Scales in User Experience,” visited on 2023-05-21. [Online]. Available: https://www.nngroup.com/articles/powers-of-10-time-scales-in-ux/?fbclid=IwAR0qsOSKooC3wN98YY1RUL5p2Qww_DviAyBeOsdy1IwuouwdBidaFDv7-4w

A

SQL code of the database

In this appendix, the code of the database developed for CMU Portugal is presented below:

Listing A.1: SQL code of the database for CMU Portugal

```
DROP DATABASE IF EXISTS cmu;
CREATE DATABASE IF NOT EXISTS CMU;

DROP TABLE IF EXISTS person CASCADE;
DROP TABLE IF EXISTS student CASCADE;
DROP TABLE IF EXISTS affiliated CASCADE;
DROP TABLE IF EXISTS dualdegree CASCADE;
DROP TABLE IF EXISTS withdraw CASCADE;
DROP TABLE IF EXISTS alumni CASCADE;
DROP TABLE IF EXISTS university CASCADE;
DROP TABLE IF EXISTS school CASCADE;
DROP TABLE IF EXISTS department CASCADE;
DROP TABLE IF EXISTS program CASCADE;
DROP TABLE IF EXISTS associatedTo CASCADE;
DROP TABLE IF EXISTS enrolledIn CASCADE;
```

```sql
DROP TABLE IF EXISTS affiliatedVisits CASCADE;

create table person (
   id integer not null,
   name varchar(50) not null,
   fullName varchar(80),
   gender varchar(10),
   email varchar(50) ,
   email2 varchar(50),
   constraint pk_person primary key(id),
   UNIQUE(email),
   UNIQUE(email2));

create table student (
   id integer not null,
   researchArea varchar(70) not null,
   status varchar(30) not null,
   degreeType varchar(70) not null,
   enrollDate date,
   enrollYear integer not null,
   academicEnrollYear varchar(10) not null,
   expectedGradYear integer,
   endYear integer,
   nationality varchar(50) not null,
   nationality2 varchar(50),
   nationalityArea varchar(50) not null,
   scholarshipNr varchar(50),
   researchTopic varchar(300),
   introText text,
   constraint pk_student primary key(id),
   constraint fk_student foreign key (id) references person(id) ON DELETE CASCADE ON UPDATE
       CASCADE);

create table profile (
   id integer not null,
   type varchar(20) not null,
   url varchar(250) not null,
   constraint pk_profile primary key(type, id),
  constraint fk_profile foreign key (id) references student(id) ON DELETE CASCADE ON UPDATE CASCADE);

create table position (
  id integer not null,
  positionName varchar(50) not null,
  startDate year not null,
  country varchar(30),
  dateVerified year not null,
  source varchar(80),
  positionWeblink varchar(250),
  constraint pk_position primary key(positionName, id),
  constraint fk_position foreign key (id) references student(id) ON DELETE CASCADE ON UPDATE CASCADE);

create table institution (
  instName varchar(80) not null,
  instNameAbbrv varchar(30),
  instType varchar(50),
  constraint pk_institution primary key(instName));

create table school (
  schoolNameAbbrv varchar(50) not null,
  schoolName varchar(80) not null,
  instName varchar(50) not null,
  constraint pk_school primary key(schoolNameAbbrv, instName),
  constraint fk_school foreign key (instName) references institution(instName) ON DELETE CASCADE ON UPDATE CASCADE);

create table department (
  deptName varchar(50) not null,
  schoolNameAbbrv varchar(50) not null,
  instName varchar(50) not null,
  constraint pk_department primary key(deptname, schoolNameAbbrv, instName),
  constraint fk_department foreign key (schoolNameAbbrv, instName) references school(schoolNameAbbrv, instName) ON DELETE CASCADE ON UPDATE CASCADE);

create table program (
  progName varchar(80) not null,
  progNameAbbrv varchar(30),
  schoolNameAbbrv varchar(50) not null,
  instName varchar(50) not null,
  constraint pk_program primary key(progName, schoolNameAbbrv, instName),
  constraint fk_program foreign key (schoolNameAbbrv, instName) references
      school(schoolNameAbbrv, instName) ON DELETE CASCADE ON UPDATE CASCADE);

create table enrolledIn (
  id integer not null,
  progName varchar(80) not null,
  schoolNameAbbrv varchar(50) not null,
  instName varchar(50) not null,
  enrollType varchar(10) not null,
  constraint pk_program primary key(id, progName, schoolNameAbbrv, instName),
  constraint fk_enrolledIn_student foreign key (id) references student(id) ON DELETE CASCADE
      ON UPDATE CASCADE,
  constraint fk_enrolledIn_program foreign key (progName, schoolNameAbbrv, instName)
      references program(progName, schoolNameAbbrv, instName) ON DELETE CASCADE ON UPDATE
      CASCADE);

create table associatedToDept (
  id integer not null,
  deptName varchar(50) not null,
  schoolNameAbbrv varchar(50) not null,
  instName varchar(50) not null,
  constraint pk_associatedToDept primary key(id),
  constraint fk_associatedToDept_person foreign key (id) references person(id) ON DELETE
      CASCADE ON UPDATE CASCADE,
  constraint fk_associatedToDept_dept foreign key (deptName, schoolNameAbbrv, instName)
      references department(deptName, schoolNameAbbrv, instName) ON DELETE CASCADE ON UPDATE
      CASCADE);

create table affiliatedWithInst (
  id integer not null,
  instName varchar(80) not null,
  constraint pk_affiliatedWithInst primary key(id, instName),
  constraint fk_affiliatedWithInst_person foreign key (id) references person(id) ON DELETE CASCADE ON UPDATE CASCADE,
  constraint fk_affiliatedWithInst_inst foreign key (instName) references institution(instName) ON DELETE CASCADE ON UPDATE CASCADE);

create table worksUnder (
  id integer not null,
  schoolNameAbbrv varchar(50) not null,
  instName varchar(50) not null,
  constraint pk_worksUnder primary key(id),
  constraint fk_worksUnder_student foreign key (id) references student(id) ON DELETE CASCADE ON UPDATE CASCADE,
  constraint fk_worksUnder_school foreign key (schoolNameAbbrv, instName) references school(schoolNameAbbrv, instName) ON DELETE CASCADE ON UPDATE CASCADE);

create table facultyResearcher (
  id integer not null,
  type varchar(30) not null,
  affiliation varchar(30),
  advisingStatus varchar(50) not null,
  constraint pk_program primary key(id));

create table advisedBy (
  student_id integer not null,
  person_id integer not null,
  advisingType varchar(10) not null,
  constraint pk_advisedBy primary key(student_id, person_id),
  constraint fk_advisedBy_student foreign key (student_id) references student(id) ON DELETE CASCADE ON UPDATE CASCADE,
  constraint fk_advisedBy_person foreign key (person_id) references person(id) ON DELETE CASCADE ON UPDATE CASCADE);

create table thesis (
  thesisId integer not null,
  thesisTitle varchar(300) not null,
  url varchar(250),
  constraint pk_thesis primary key(thesisId));

```

```sql
create table writes(
  id integer not null,
  thesisId integer not null,
  constraint pk_writes primary key(id),
  constraint fk_writes_id foreign key (id) references student(id) ON DELETE CASCADE ON
      UPDATE CASCADE,
  constraint fk_writes_title foreign key (thesisId) references thesis(thesisId) ON DELETE
      CASCADE ON UPDATE CASCADE);

create table visitor (
  id integer not null,
  enrollYear year not null,
  startDate date not null,
  endDate date not null,
  constraint pk_visitor primary key(id),
  constraint fk_visitor foreign key (id) references person(id) ON DELETE CASCADE ON UPDATE
      CASCADE);

create table visitingStudent (
  id integer not null,
  sponsorPT varchar(50) not null,
  academicQualification varchar(30) not null,
  constraint pk_visitingStudent primary key(id),
  constraint fk_visitingStudent foreign key (id) references visitor(id) ON DELETE CASCADE ON
      UPDATE CASCADE);

create table visitingFaculty (
  id integer not null,
  position varchar(60) not null,
  constraint pk_visitingFaculty primary key(id),
  constraint fk_visitingFaculty foreign key (id) references visitor(id) ON DELETE CASCADE ON
      UPDATE CASCADE);

create table hostedBy (
  visitor_id integer not null,
  fac_id integer not null,
  constraint pk_hostedBy primary key(visitor_id),
  constraint fk_hostedBy_visitor foreign key (visitor_id) references visitor(id) ON DELETE CASCADE ON UPDATE CASCADE,
  constraint fk_hostedBy_faculty foreign key (fac_id) references facultyResearcher(id) ON DELETE CASCADE ON UPDATE CASCADE);

create table comesFrom (
  id integer not null,
  schoolNameAbbrv varchar(50) not null,
  instName varchar(50) not null,
  constraint pk_comesFrom primary key(id),
  constraint fk_comesFrom_visitor foreign key (id) references visitor(id) ON DELETE CASCADE ON UPDATE CASCADE,
  constraint fk_comesFrom_school foreign key (schoolNameAbbrv, instName) references school(schoolNameAbbrv, instName) ON DELETE CASCADE ON UPDATE CASCADE);

create table isStudent (
  student_id integer not null,
  visStudent_id integer not null,
  constraint pk_isStudent primary key(student_id),
  constraint fk_isStudent_student foreign key (student_id) references student(id) ON DELETE CASCADE ON UPDATE CASCADE,
  constraint fk_isStudent_visStudent foreign key (visStudent_id) references visitingStudent(id) ON DELETE CASCADE ON UPDATE CASCADE);

create table isFaculty (
  fac_id integer not null,
  visFac_id integer not null,
  constraint pk_isFaculty primary key(fac_id),
  constraint fk_isFaculty_fac foreign key (fac_id) references facultyResearcher(id) ON DELETE CASCADE ON UPDATE CASCADE,
  constraint fk_isFaculty_visFac foreign key (visFac_id) references visitingFaculty(id) ON DELETE CASCADE ON UPDATE CASCADE);

create table project (
  id integer not null,
  projType varchar(10) not null,
  projYear year not null,
  title varchar(150) not null,
  titleAbbrv varchar(20) not null,
  startDate date,
  endDate date,
  referenceNr varchar(50),
  constraint pk_project primary key(id));

create table keyword (
  id integer not null,
  name varchar(30) not null,
  constraint pk_keyword primary key(name,id),
  constraint fk_keyword foreign key (id) references project(id) ON DELETE CASCADE ON UPDATE CASCADE);

create table participates (
  student_id integer not null,
  proj_id integer not null,
  constraint pk_participates primary key(student_id, proj_id),
  constraint fk_participates_student foreign key (student_id) references student(id) ON DELETE CASCADE ON UPDATE CASCADE,
  constraint fk_participates_project foreign key (proj_id) references project(id) ON DELETE CASCADE ON UPDATE CASCADE);

create table worksOn (
  facRes_id integer not null,
  proj_id integer not null,
  position varchar(50),
  constraint pk_worksOn primary key(facRes_id, proj_id),
  constraint fk_worksOn_facRes foreign key (facRes_id) references facultyResearcher(id) ON DELETE CASCADE ON UPDATE CASCADE,
  constraint fk_worksOn_project foreign key (proj_id) references project(id) ON DELETE CASCADE ON UPDATE CASCADE);

create table deptInvolvedIn (
  id integer not null,
  deptName varchar(50) not null,
  schoolNameAbbrv varchar(50) not null,
  instName varchar(50) not null,
  constraint pk_deptInvolvedIn primary key(id,deptName, schoolNameAbbrv, instName),
```

```sql
  constraint fk_deptInvolvedIn_proj foreign key (id) references project(id) ON DELETE CASCADE ON UPDATE CASCADE,
  constraint fk_deptInvolvedIn_dept foreign key (deptName, schoolNameAbbrv, instName) references department(deptName, schoolNameAbbrv, instName) ON DELETE CASCADE ON UPDATE CASCADE);

create table schoolInvolvedIn (
  id integer not null,
  schoolNameAbbrv varchar(50) not null,
  instName varchar(50) not null,
  role varchar(50) not null,
  constraint pk_deptInvolvedIn primary key(id, schoolNameAbbrv, instName),
  constraint fk_schoolInvolvedIn_proj foreign key (id) references project(id) ON DELETE CASCADE ON UPDATE CASCADE,
  constraint fk_schoolInvolvedIn_school foreign key (schoolNameAbbrv, instName) references school(schoolNameAbbrv, instName) ON DELETE CASCADE ON UPDATE CASCADE);

create table instInvolvedIn (
  id integer not null,
  instName varchar(80) not null,
  role varchar(50) not null,
  constraint pk_instInvolvedIn primary key(id, instName),
  constraint fk_instInvolvedIn_proj foreign key (id) references project(id) ON DELETE CASCADE ON UPDATE CASCADE,
  constraint fk_instInvolvedIn_inst foreign key (instName) references institution(instName) ON DELETE CASCADE ON UPDATE CASCADE);
```

B

Database performance's queries

In this appendix, we detail the queries developed to test the database's performance. These queries are presented below:

Listing B.1: Database performance's queries

```sql
-- QUERY 1 - get the list of students presented in the user interface
WITH studentPT AS (SELECT * FROM person NATURAL JOIN student NATURAL JOIN enrolledIn WHERE
    enrollType = 'PT'),
dualdegreeCMU AS (SELECT * FROM student NATURAL JOIN enrolledIn NATURAL JOIN
    associatedToDept),
affiliatedDept AS (SELECT * FROM student NATURAL JOIN associatedToDept WHERE degreeType =
    'affiliated')
SELECT studentPT.name, studentPT.researchArea, studentPT.status, studentPT.degreeType,
    studentPT.enrollYear, studentPT.endYear, studentPT.progName AS 'PT Program',
    studentPT.schoolNameAbbrv AS 'PT school', studentPT.instName AS 'PT University',
    dualdegreeCMU.progName AS 'CMU Program', dualdegreeCMU.schoolNameAbbrv AS 'CMU School',
    dualdegreeCMU.deptname AS 'CMU Department'
FROM studentPT
INNER JOIN
```

```
dualdegreeCMU
ON studentPT.id = dualdegreeCMU.id
UNION
SELECT studentPT.name, studentPT.researchArea, studentPT.status, studentPT.degreeType,
    studentPT.enrollYear, studentPT.endYear, studentPT.progName AS 'PT Program',
    studentPT.schoolNameAbbrv AS 'PT school', studentPT.instName AS 'PT University', null,
    affiliatedDept.schoolNameAbbrv AS 'CMU School', affiliatedDept.deptname AS 'CMU
    Department'
FROM studentPT
INNER JOIN
affiliatedDept
ON studentPT.id = affiliatedDept.id;

-- QUERY 2 - get the personal information of all students
SELECT fullName AS 'Full Name', gender, email, nationality, nationalityArea AS 'Nationality
    Area'
FROM person natural join student;

-- QUERY 3 - get the academic information of all students
SELECT degreeType AS 'Degree Type', status, researchArea AS 'Research Area', enrollDate AS
    'Enroll Date', academicEnrollYear AS 'Academic Enroll Year', enrollYear AS 'Enroll
    Year', endYear AS 'End Year', endYear - enrollYear AS 'Nr of Enrollments'
FROM person natural join student;

-- QUERY 4 - get the enrollment information of a student's profile on the user interface
SELECT enrollType, progName, progNameAbbrv, schoolName, null, deptName, instName,
    instNameAbbrv
FROM person natural join enrolledIn natural join program natural join school natural join
    institution natural join associatedToDept
WHERE enrollType='CMU'AND name= 'Alexandre Ligo'
UNION
SELECT enrollType, progName, progNameAbbrv, schoolName, schoolNameAbbrv, null, instName,
    instNameAbbrv
from person natural join student natural join enrolledIn natural join program natural join
    school natural join institution
WHERE enrollType='PT' AND degreeType= 'Dual Degree' AND name= 'Alexandre Ligo';

-- QUERY 5 - get the student's advisors of a student's profile on the user interface
```

```sql
WITH student AS ( SELECT id FROM person WHERE name = 'Alessandro Giordano')
SELECT advisingType AS 'Advsing Type', p.name
FROM student as s, advisedBy as a, person as p
WHERE s.id = a.student_id and a.person_id = p.id
ORDER BY advisingType;

-- QUERY 6 - get the number of students with the degree type "Dual Degree" grouped by their enrollment year and current status
SELECT enrollYear AS "Enrollment Year", status AS "Status", COUNT(id) AS "Nr of Students"
FROM student
WHERE degreeType = 'Dual Degree'
GROUP BY enrollYear, status
ORDER BY enrollYear;

-- QUERY 7 - get the number of students with the degree type "Dual Degree" grouped by their enrollment year and study program at CMU
SELECT enrollYear AS "Enrollment Year", progNameAbbrv AS "Doctoral Program CMU", COUNT(id) AS "Dual degree PhD Students"
FROM student NATURAL JOIN enrolledIn NATURAL JOIN program
WHERE enrollType = 'CMU' AND degreeType = 'Dual Degree'
GROUP BY enrollYear, progNameAbbrv
ORDER BY enrollYear;

-- QUERY 8 - get the number of students with the degree type "Dual Degree" grouped by their their enrollment year and department at CMU
SELECT enrollYear AS "Enrollment Year", deptName AS "Department CMU", COUNT(id) AS "Dual degree PhD Students"
FROM person NATURAL JOIN student NATURAL JOIN associatedToDept
WHERE degreeType = 'Dual Degree'
GROUP BY enrollYear, deptName
ORDER BY enrollYear;

-- QUERY 9 - get the number of students with the degree type "Dual Degree" grouped by their university in Portugal
SELECT schoolNameAbbrv AS "Institution PT", COUNT(id) AS "Dual degree PhD Students"
FROM student NATURAL JOIN enrolledIn
WHERE enrollType = 'PT' AND degreeType = 'Dual Degree'
GROUP BY schoolNameAbbrv;
```

```sql

-- QUERY 10 - get the number of students with the degree type "Dual Degree" grouped by their gender
SELECT gender AS Gender, COUNT(id) AS "Dual degree PhD Students"
FROM person NATURAL JOIN student
WHERE degreeType = 'Dual Degree'
GROUP BY gender;

-- QUERY 11 - get the number of students with the degree type "Dual Degree" grouped by their nationality area
SELECT nationalityArea AS Nationality, COUNT(id) AS "Dual degree PhD Students"
FROM person NATURAL JOIN student
WHERE degreeType = 'Dual Degree'
GROUP BY nationalityArea;

-- QUERY 12 get the number of students with the degree type "Affiliated" grouped by their enrollment year and research area
SELECT enrollYear AS "Enrollment Year", researchArea AS "Research area", COUNT(id) AS "Nr of Affiliated Students"
FROM student
WHERE degreeType = 'Affiliated'
GROUP BY enrollYear, researchArea
ORDER BY enrollYear;

-- QUERY 13 webiste: get the name, institution PT, research area and enroll year of all active Students
SELECT name, schoolNameAbbrv, researchArea, enrollYear
FROM person NATURAL JOIN student NATURAL JOIN enrolledIn
WHERE enrollType = 'PT' AND status = 'Student'
ORDER BY name;

-- QUERY 14: website get the name, institution PT, research area and enroll year of active Students with "Dual Degree" degree type
SELECT name, schoolNameAbbrv, researchArea, enrollYear
FROM person NATURAL JOIN student NATURAL JOIN enrolledIn
WHERE enrollType = 'PT' AND status = 'Student' AND degreeType = 'Dual Degree'
ORDER BY name;

```

```sql
-- QUERY 15 webiste: get the name, institution PT, research area and enroll year of all Alumni
SELECT name, schoolNameAbbrv, researchArea, enrollYear
FROM person NATURAL JOIN student NATURAL JOIN enrolledIn
WHERE enrollType = 'PT' AND status = 'Alumni'
ORDER BY name;
```

C
User Testing

In this chapter, we present the tasks the users had to follow, the questionnaire they had to answer, and the documents provided to them while filling out the questionnaire in order for the user interface to be evaluated.

C.1 User tasks

Below are tasks the users had to follow during user testing:

- **1st task: "Search Student" functionality**
  1. Search for the student with the name "Carla costa".
  2. Search all students whose institution in Portugal is "IST".
  3. Order the table by enrolment year.
  4. Access the student's profile of "Hugo Pinto".
- **2nd task: "Add Student" functionality**
  1. Add a new student with the name "Sofia Mendes", filling only the mandatory fields with the inputs that you wish and using the help of the provided options in the possible fields.

2. Confirm that the student with the name “Sofia Mendes” has been added to the list of students.

- **3rd task: “Update Student” functionality**

  1. Update the nationality, status, and host institution of the student “Sofia Mendes” with the values you wish.
  2. Confirm that the fields of the student “Sofia Mendes” have been successfully updated.

- **4th task: “Remove Student” functionality**

  1. Delete the student “Sofia Mendes”.
  2. Confirm that the student Sofia Mendes” has been successfully deleted.

C.2 Questionnaire

1. **Do you accept that the gathered data, which is treated anonymously, be used as a means to evaluate the user interface and collect statistics?**
   - Yes
   - No
2. **Gender**:
   - Male
   - Female
   - Prefer not to say
3. **Age**:
   - 18-23
   - 24-35
   - 36-45
   - $>$ 46
4. **Highest academic qualification received:**
   - High school
   - Bachelor
   - Master

- Ph.D.

5. **Level of experience with technological platforms:**

    - Inexperienced
    - Familiarized
    - Experienced

6. **Please choose a number between 1 and 5 to indicate your level of disagreement or agreement, with 1 meaning fully disagree and 5 fully agree, in terms of whether our platform adhered to the provided heuristics list (that includes the detailed description of these heuristics) for each of the completed tasks:**

Heuristic	**1st task**	**2nd task**	**3rd Task**	**4th Task**
Visibility of system status				
Match between system and the real world				
User control and freedom				
Recognition rather than recall				
Consistency and standards				
Aesthetic and minimalist design				

7. **Is there any issue you encountered in the user interface?**

    - Yes
    - No

8. **If yes, please describe the issue:**

9. **If yes, please rate the issue based on the severity scale (0 to 4) provided to you:**

10. **Additional feedback?**

Thank you!

C.3 Provided documents

Below are the documents provided to the users while they were filling in the questionnaire, already described in Section 6.3:

Detailed heuristics list:

- Visibility of system status: The system should always update the user on what is happening by providing suitable feedback in a timely manner. Important characteristics are that every screen has a title or header to describe its contents, a consistent stylistic treatment across the system is applied, a clear indication of the current location exists, and the menu-naming terminology is consistent within the users' task domain.
- Match between system and the real world: The system should speak the user's language, with words, phrases, and concepts familiar to the user, rather than system-oriented terms. It should follow real-world conventions, making information appear in a natural and logical order. Important characteristics are that section headings and subsections in each screen are ordered in a logical way, there is a natural sequence to the menu choices for a data item, and that selected colors correspond to common expectations about color codes.
- User control and freedom: Users should be free to select and sequence tasks (when appropriate), rather than having the system do this for them. Important characteristics are that there are clear exits on each document screen, all screens are accessible across the system, there is an ”undo” function, and users can easily move forward and backward.
- Recognition rather than recall: The user should not have to remember information from one part of the dialogue to the next. Instructions for the use of the system should be visible or easily retrievable whenever appropriate. Important characteristics are that prompts, cues, and messages are placed where the user is likely to be looking on the screen, white space is used to create symmetry and lead the user in the appropriate direction, have items grouped into logical zones, and headings to distinguish between zones.
- Consistency and standards: Users should not have to wonder whether different words, situations, or actions mean the same thing. Important characteristics are that formatting standards are followed consistently on all screens, each window has a title, there are no more than four to seven colors, and names are consistent, both within each tab and across the system, in grammatical style and terminology.
- Aesthetic and minimalist design:Information that is unnecessary or rarely used shouldn't be included in dialogues. Important characteristics are that only information essential to decision-

making is displayed on the screen, large objects and bold fonts are used to distinguish sections, field labels are brief, familiar, and descriptive, and the visual layout is well designed.

Severity scale:

- **0**: I completely disagree that this is a usability issue.
- **1**: Cosmetic problem, needs to be rectified only if more time is available to complete the project.
- **2**: Minor usability problem, fixing this should be given low priority.
- **3**: Major usability problem that is important to fix, should be given high priority.
- **4**: Usability catastrophe, imperative to fix before the product is launched.