INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
MESTRADO EM ENGENHARIA INFORMÁTICA




Gestão automática de atribuição e monitorização de projetos/estágios

FRANCISCO JOÃO FERREIRA SILVA FERNANDES
Outubro de 2023

POLITÉCNICO
DO PORTO

ISEP INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO

Automatic management tool for attribution and monitorization of projects/internships

Francisco João de Ferreira Silva Fernandes

A dissertation submitted in partial fulfilment of the requirements for the degree of Master of Informatics, Specialisation Area of Software Engineering

Supervisor: Drª. Piedade Barros Lopez Carvalho

Evaluation Committee:

President:

...

Members:

...

Porto, October 2023

Declaração de Integridade

Declaro ter conduzido este trabalho académico com integridade.

Não plagiei ou apliquei qualquer forma de uso indevido de informações ou falsificação de resultados ao longo do processo que levou à sua elaboração.

Portanto, o trabalho apresentado neste documento é original e de minha autoria, não tendo sido utilizado anteriormente para nenhum outro fim.

Declaro ainda que tenho pleno conhecimento do Código de Conduta Ética do P. PORTO.

ISEP, Porto, 13 de outubro de 2023

Resumo

No último ano académico, os estudantes do ISEP necessitam de realizar um projeto final para obtenção do grau académico que pretendem alcançar. O ISEP fornece uma plataforma digital onde é possível visualizar todos os projetos que os alunos se podem candidatar. Apesar das vantagens que a plataforma digital traz, esta também possui alguns problemas, nomeadamente a difícil escolha de projetos adequados ao estudante devido à excessiva oferta e falta de mecanismos de filtragem. Para além disso, existe também uma indecisão acrescida para selecionar um supervisor que seja compatível para o projeto selecionado.

Tendo o aluno escolhido o projeto e o supervisor, dá-se início à fase de monitorização do mesmo, que possui também os seus problemas, como o uso de diversas ferramentas que posteriormente levam a possíveis problemas de comunicação e dificuldade em manter um histórico de versões do trabalho desenvolvido.

De forma a responder aos problemas mencionados, realizou-se um estudo aprofundado dos tópicos de sistemas de recomendação aplicados a *Machine Learning* e *Learning Management Systems*. Para cada um desses grandes temas, foram analisados sistemas semelhantes capazes de solucionar o problema proposto, tais como sistemas de recomendação desenvolvidos em artigos científicos, aplicações comerciais e ferramentas como o ChatGPT.

Através da análise do estado da arte, concluiu-se que a solução para os problemas propostos seria a criação de uma aplicação Web para alunos e supervisores, que juntasse as duas temáticas analisadas. O sistema de recomendação desenvolvido possui filtragem colaborativa com factorização de matrizes, e filtragem por conteúdo com semelhança de cossenos. As tecnologias utilizadas no sistema centram-se em Python no *back-end* (com o uso de TensorFlow e NumPy para funcionalidades de *Machine Learning*) e Svelte no *front-end*. O sistema foi inspirado numa arquitetura em microsserviços em que cada serviço é representado pelo seu próprio contentor de Docker, e disponibilizado ao público através de um domínio público.

O sistema foi avaliado através de três métricas: performance, confiabilidade e usabilidade. Foi utilizada a ferramenta *Quantitative Evaluation Framework* para definir dimensões, fatores e requisitos (e respetivas pontuações). Os estudantes que testaram a solução avaliaram o sistema de recomendação com um valor de aproximadamente 7 numa escala de 1 a 10, e os valores de precision, recall, false positive rate e F-Measure foram avaliados em 0.51, 0.71, 0.23 e 0.59 respetivamente. Adicionalmente, ambos os grupos classificaram a aplicação como intuitiva e de fácil utilização, com resultados a rondar o 8 numa escala de 1 em 10.

Palavras-chave: Sistema de recomendação, Filtragem colaborativa, Filtragem por conteúdo, Monitorização de projeto, Estudante, Supervisor

Abstract

In the last academic year, students at ISEP need to complete a final project to obtain the academic degree they aim to achieve. ISEP provides a digital platform where all the projects that students can apply for can be viewed. Besides the advantages this platform has, it also brings some problems, such as the difficult selection of projects suited for the student due to the excessive offering and lack of filtering mechanisms. Additionally, there is also increased difficulty in selecting a supervisor compatible with their project.

Once the student has chosen the project and the supervisor, the monitoring phase begins, which also has its issues, such as using various tools that may lead to potential communication problems and difficulty in maintaining a version history of the work done.

To address the mentioned problems, an in-depth study of recommendation systems applied to Machine Learning and Learning Management Systems was conducted. For each of these themes, similar systems that could solve the proposed problem were analysed, such as recommendation systems developed in scientific papers, commercial applications, and tools like ChatGPT.

Through the analysis of the state of the art, it was concluded that the solution to the proposed problems would be the creation of a web application for students and supervisors that combines the two analysed themes. The developed recommendation system uses collaborative filtering with matrix factorization and content-based filtering with cosine similarity. The technologies used in the system are centred around Python on the backend (with the use of TensorFlow and NumPy for Machine Learning functionalities) and Svelte on the frontend. The system was inspired by a microservices architecture, where each service is represented by its own Docker container, and it was made available online through a public domain.

The system was evaluated through performance, reliability, and usability. The Quantitative Evaluation Framework tool was used to define dimensions, factors, and requirements (and their respective scores). The students who tested the solution rated the recommendation system with a value of approximately 7 on a scale of 1 to 10, and the precision, recall, false positive rate, and F-Measure values were evaluated at 0.51, 0.71, 0.23, and 0.59, respectively. Additionally, both groups rated the application as intuitive and easy to use, with ratings around 8 on a scale of 1 to 10.

Keywords: Recommendation system, Collaborative filtering, Content-based filtering, Project monitoring, Student, Supervisor

Table of Contents

List of Figures

List of Tables

List of Code

Acronyms and Symbols

List of Acronyms

AHP	*Analytical Hierarchy Process*
AI	*Artificial Intelligence*
AMQP	*Advanced Messaging Queue Protocol*
ANN	*Artificial Neural Network*
API	*Application Programming Interface*
BMF	*Biased Matrix Factorization*
CI	*Consistency Index*
CPU	*Central Processing Unit*
CR	*Consistency Ratio*
CRISP-DM	*Cross-Industry Standard Process for Data Mining*
CSS	*Cascading Style Sheets*
DL	*Deep Learning*
DM	*Data Mining*
DTO	*Data Transfer Object*
E2E	End-To-End
FAST	*Functional Analysis and System Technique*
FFE	*Fuzzy Front-End*
FN	*False Negative*
FP	*False Positive*
FURPS	*Functionality, Usability, Reliability, Performance, and Security*
GPU	*Graphics Processing Unit*
HTTP	*Hypertext Transfer Protocol*
HTTPS	*Hypertext Transfer Protocol Secure*

ISEP	*Instituto Superior de Engenharia do Porto*
JWT	*JSON Web Token*
KDD	*Knowledge Discovery in Databases*
KNN	*K-Nearest Neighbour*
LMS	*Learning Management System*
MAE	*Mean Absolute Error*
MF	*Matrix Factorization*
ML	*Machine Learning*
MOODLE	Modular Object Oriented Dynamic Learning Environment
MSE	*Mean Square Error*
NPD	*New Product Development*
ORM	*Object Relational Mapping*
QEF	*Quantitative Evaluation Framework*
REST	*Representational State Transfer*
RI	*Random Index*
RMSE	*Root Mean Square Error*
RS	*Recommendation System*
SciPro	*Supporting the Scientific Process*
SEMMA	*Sample, Explore, Modify, Model and Assess*
TF-IDF	*Term Frequency - Inverse Document Frequency*
TN	*True Negative*
TP	*True Positive*
TPU	*Tensor Processing Unit*
UC	*Use Case*

1 Introduction

In the Introduction chapter, the thought process behind the problem to be studied, the main objectives to be accomplished and the researched methodology to be utilized are explained.

Taking that into consideration, this chapter describes the context of *Instituto Superior de Engenharia do Porto* (ISEP), namely the project subject that exists in all computer engineering courses, the problem inherent to that subject, the goals of this project, the steps and research methodology to be applied, the project planning and a brief summary of what was analysed during the state of the art. Lastly, a general perspective of what was explored in this dissertation is shown.

1.1 Context

ISEP is an education and research engineering school located in Porto and was founded in 1852. ISEP offers a broad spectrum of engineering courses, such as civil, chemical, electrical, mechanical, computer engineering, and others.

Referring to the computer engineering courses, all bachelor and master students need to develop final projects in the last semester/year of their academic period. The students are free to choose whichever theme they want to explore and develop, as long as it meets the standards proposed by ISEP. Students also have the opportunity to choose their preferred teacher to supervise their work. To assist in the selection process of choosing the most appealing project for the student and the subsequent steps, ISEP provides a web application for students, professors and external organizations[1]. Inside the project's platform, it is possible to find various proposals that cover a panoply of themes, making it accessible for the student to find the most interesting project. In case some student wants to develop their own project, or he already established a project with another entity that is not already registered, the platform supports a mechanism to submit external proposals. After a student manifests their interest in a particular proposal, it needs to go through an acceptance and formalization process in which the advisor, supervisor, head of the curricular unit and ISEP presidency participate.

1 - https://projetos.dei.isep.ipp.pt/

1.2 Problem

Since the selection process of projects is manual, the enormous amount of proposals available in the platform originates several potential problems:

- Given the high offer of project opportunities, the student inadvertently overlooks any proposals that could be interesting to him. In addition, the choice of other less attractive projects due to not having noticed other proposals becomes more predominant;
- The manual process requires a detailed reading of the project/internship proposals, which takes copious amounts of time to analyse them. This can lead to the student not having enough time to evaluate all the projects that are of interest to him;
- Students who are not sure what they intend to develop in the final project of the course may feel lost in the immensity of opportunities, which could lead them to not make the right decision and to have an eventual bad experience during the duration of the project.

After choosing the internship proposal the student wants, the communication process begins between the entity that created the proposal and the student. As companies and/or ISEP advisors prefer different methods of communication and given the number of people interested in these projects, communication with these entities becomes quite complex. In certain cases, the student ends up not receiving any kind of response to the interest in the proposal.

In addition, considering the large number of documents that need to be completed by students and/or companies and formalized by the competent entities of ISEP, the process of filling, formalizing and approving them does not respect a universal sequence, which may lead to delays and disorganization in the delivery of the documents.

The supervision of the project/internship by the advisor and supervisor is also an aspect in which some disorganization is noted. The different means of communication adopted by the advisor and supervisor to communicate with the student, combined with the delivery of the written report to those entities on different platforms, means that there is no organized monitoring of the project and no way to access a history of report versions.

1.3 Objectives

The context and problem stated in the Sections 1.1 and 1.2 confirm inefficiency in the selection and monitoring process of internships and projects. By analysing the points stated in those sections, some research questions were raised:

- What is the best approach for helping a student in the decision-making process of choosing the most interesting project?
- What are the main points of interest for a given student that have a significant impact on the decision-making process of a certain project?
- How to maximize time efficiency in the selection and monitoring phases?

Adopting a more automated process would minimize the time spent by all stakeholders during the internship/project operation. Students would be able to get the majority of their recommended proposals by filling in their details, grades and preferences. Supervisors on the other hand would get notified of each potential candidate and have a unique space where they could control the project process (from acceptance to final delivery).

To achieve that, an automatic recommendation system (RS) (based on machine learning (ML) principles) that is capable of filtering project proposals based on the interests and knowledge of the students, so that only those proposals that may be of interest are shown was developed. These proposals must contain a summary of the project so that it is possible to analyse them more quickly and effectively. The choice of potential projects must be complemented with the automatic suggestion of potential supervisors to speed up the process of choosing and formalizing the project.

In addition to creating a RS for project/internship proposals, a management mechanism for these proposals is also created. The new monitoring system should incorporate a section where it is possible for students and supervisors to be able to communicate with each other and share formal documents. Those documents could be reports and other relevant files for the project.

Both implemented solutions are available via Application Programming Interface (API) gateway and act independently from external platforms like ISEP's project platform. This decision was made because the main purpose of that decision is to make a solution capable of being used for every study area (not exclusively for computer engineering) and in any university (regardless of the country).

1.4 Research Methodology

The design science methodology (also known as design and creation) was used to answer the questions stated in Section 1.3. This research technique consists of 5 process steps described in the "Design Science Research Process Model" (Vaishnavi, Kuechler and Stacey, 2021):

- **Awareness of Problem:** A problem worth investigating and solving is delved into extensively, in order to develop a research proposal out of it;
- **Suggestion:** The ideas (design) are established in order to solve the problem;
- **Development:** The solution is developed in this phase;
- **Evaluation:** The solution created in the development phase is studied, evaluated and analysed according to the criteria defined in the Awareness of Problem phase. Depending on the results gathered, another iteration of design science methodology could be initiated (starting in the Suggestion phase);
- **Conclusion:** The research methodology ends when results are satisfactory enough to prove that the developed solution works.

The design science approach is essentially a synonym of what software engineers perform on a regular basis: a potential solution is defined given an identified problem, the solution is implemented and the results are evaluated and registered (to improve in next iterations of the project).

1.5 Planning

The Gantt diagram available in Figure 1 settles the planning proposed for the development of this thesis project.

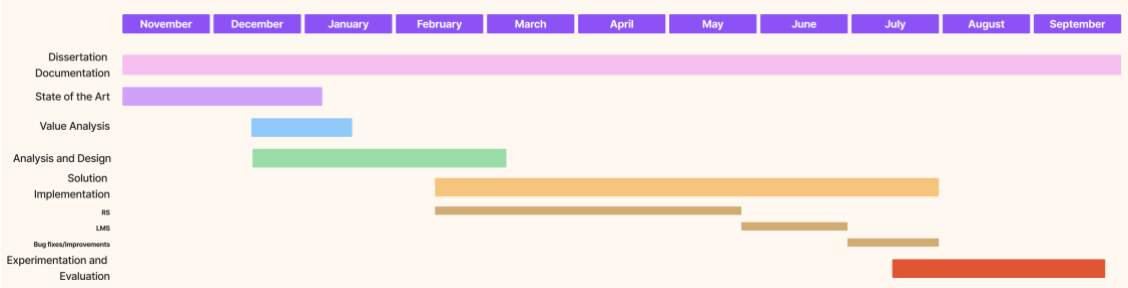


Figure 1 - Gantt Diagram

The project development started in November 2022. Throughout the first months until February 2023, the state of the art was investigated using numerous scientific sources, the value analysis of this project was explored and a first version of the analysis and design of the product was created. The following months until the final delivery (October 2023) were primarily dedicated to the development of the solution and individual experimentation and evaluation. In the experimentation and evaluation section, system trials and questionnaires were given to potential users (students and supervisors). Those questionnaires serve two primary purposes: verify if all requirements arising from the problem analysis were correctly obtained, and feedback on what would be beneficial to improve in the future.

Project development was organized using the agile methodology, which divided the project requirements into three iteration sprints: one that centred on developing the RS, other that centred on developing the LMS, and the last one focused on minor improvements and bug fixes.

1.6 State of the Art Summary

During the chapter of the state of the art, the concept of ML was researched with more emphasis on RSs. ML projects can utilize data mining procedures to discover patterns and relationships in large datasets. Data mining can be integrated popular frameworks such as KDD, CRISP-DM and SEMMA. KDD is the most traditional one, while CRISP-DM is seen as the most

popular data mining framework since it can be easily integrated with SCRUM, and SEMMA is more utilized with SAS products.

There are several types of RSs that use different filtering techniques to obtain the best results. The most traditional filtering techniques are content-based filtering and collaborative filtering. However, to have the best features of both and mitigate their disadvantages, those techniques can be combined in an hybrid approach. Examples of hybrid approaches studied in the state of the art chapter are knowledge-based, utility-based, fuzzy and demographic-based filtering. Those algorithms can be divided into four main types: supervised, semi-supervised, unsupervised and reinforcement. RSs can also be implemented with the use of DL, which is capable of recognizing more complex patterns in the data than traditional ML algorithms. Several algorithms were analysed in detail for both ML and DL, such as KNN, MF, Decision Tree and ANN. There is no correct answer as to which is the best algorithm for RS, because it is solely dependent on the type of filtering technique to be used and the main purpose of the RS. Natural language processing (NLP) is also analysed since there are RSs that use this technique (in conjunction with DL) to obtain more tailored recommendations. Term Frequency - Inverse Document Frequency (TF-IDF) is a NLP algorithm that can be used without DL.

To determine if the RS is producing accurate results, ML applications have an evaluation phase. Evaluating a RS can be done in several different environments: offline with local data, user studies where a selected number of people test the application, and online which is tested by regular users. Besides that, those algorithms are evaluated through property-based metrics, namely prediction accuracy. Accuracy is measured with equations like Root Mean Square Error (RMSE) and Mean Square Error (MAE), or classified with a confusion metric to measure precision, recall, false positive rate, and F-Measure.

There are a vast number of programming languages and frameworks to develop RSs. Python was analysed because of it being one of the most popular programming languages for ML applications for its reliability and ease of code. There are several ML frameworks for Python including Scikit-Learn, TensorFlow, Keras and PyTorch. All of those frameworks were compared in a separate section because they offer different types of programming styles and features.

The other part of the state of the art is allocated to the LMS. LMS is a concept that represents systems that allow the possibility of digital learning and offer several features that can be adapted to specific use cases (UCs). There are several LMS platforms that are either open-source or paid, including Moodle, Blackboard, Canvas and Brightspace. However, since those platforms offer a large number of functionalities, the case of creating a system with custom selected features with a certain programming language is valid and relevant depending on the UC.

The final section of the state of the art is dedicated to the analysis of related work, where both RSs and LMSs were analysed. The main investigated applications were course, thesis topic, internship and supervisor RSs, which are important themes to consider when developing this dissertation's RS. On the other hand, it was possible to find one commercial LMS for thesis monitoring, the SciPro System, a system developed at the University of Stockholm. Besides that,

another complete solution described in a research paper is eThesis, which was developed at the University of Piraeus in Greece. Both LMS solutions have an handful of features that enhance the supervision and monitoring experience of supervisors. Even though the dissertation's project is not as complex as those applications, they serve as a base of inspiration to improve the project's functionality.

1.7 Document Structure

This dissertation is composed of seven chapters: introduction, state of the art, value analysis, analysis and design, implementation, evaluation and experimentation, and conclusion.

The introduction chapter states the context, problem and objectives to be achieved from exploring the results of developing a solution that answers the raised questions.

The state of the art chapter analyses the practices being used in the field of RSs and LMS, the technologies that are commonly used in these areas, as well as the most relevant work found for both themes.

The value analysis chapter explores the value subjacent to this dissertation's project, by taking a closer look on the opportunities to be explored, how the functionalities work with each other and project decisions.

The analysis and design chapter explains the thought process behind the decisions made on what the solution consists in terms of functional and non-functional requirements and the coding architecture that it is based on.

The implementation chapter describes the technical aspect of the system by analysing the front-end and back-end and their individual modules separately. Application deployment to production is also explained.

The evaluation and experimentation chapter states the metrics that will be evaluated extensively, by specifying the tools and methodology of evaluation. Quantitative Evaluation Framework (QEF) was used to specify the dimensions, factors and requirements to test. Besides that, all tests made to the application (unit, system and end-to-end (E2E) tests) and their respective results were analysed.

The conclusion chapter makes a general overview about the objectives achieved during the implementation, evaluation and experimentation of the application and answers the research questions raised in Section 1.3. Besides that, it points limitations about the developed project, as well as future work to take into consideration.

There are also several appendix sections that include relevant images (such as use case diagrams, and questionnaires to students and supervisors) and code samples to justify points made throughout the dissertation, as well as the abstract of a delivered conference article based on this report.

2 State of the Art

In the State of the Art chapter, contextualization of the project is described, to introduce both themes to be explored in the chapter: RSs and LMS. For the RS theme, an explanation of what ML consists of is suggested, with further details about data mining techniques, RS filtering techniques and evaluation strategies, ML learning methodologies and algorithms, NLP and frameworks to develop RSs. For the LMS theme, an explanation of how those systems were introduced and their main features, as well as the most popular solutions to implement them are discussed. At the end of the chapter, commercial and scientific solutions that mimic what was developed in the dissertation's project are explored.

2.1 Context

Computer engineering students at ISEP are meant to work on a project or to be an intern at a company for the final year/semester of their academic duties. The project to be developed is intended to mark a vital point in the student's professional career since it is a way to experience the area the student intends to explore in the future. A study conducted on eleven undergraduate students (out of 58 students) that enrolled on a 13-week internship showed that these experiences can adjust "students' expectations and priorities for their future work environment" (Odio, Sagas and Kerwin, 2014).

In addition to selecting the proposal, the student needs to identify the most suitable supervisor for the project. The monitoring procedure is of extreme importance, since "good communication between students and their supervisor is the most important element of supervision" (Hassan, Ahmad and Abiddin, 2009). Supervision needs to "accommodate the students' demands and expectations with the supervisor's availability, experience and knowledge" (Yuanyuan Fan, Ana Evangelista and Hadi Harb, 2021), which improves the quality of the work produced. There is a vast number of steps to acknowledge when choosing a supervisor. In general, it is important that supervisors hold at least "the same qualification level

of the student's course" (Yuanyuan Fan, Ana Evangelista and Hadi Harb, 2021), are specialized in the area the student wants to explore and have the availability to be supervisors on top of their professional endeavours.

ISEP offers a web platform for students, professors and companies to help in the whole project process. Although this mechanism of choosing and monitoring a project is predominantly done online, and some intrinsic methods are automatic, there are still several manual procedures that have detrimental consequences for all stakeholders:

- The project or internship selection is chosen from the proposals found on ISEP's project website. Due to a large number of offers at the student's disposal, not only the student spends a vast amount of time looking at offers that are not of his interest, but also, predictably, the student also misses potential proposals that could align with what he wants to work on;
- For those students who are not convinced about what project they want to develop, the selection task becomes increasingly more difficult, as they do not have a reference point to search for the best proposal for them;
- By having a large number of projects to analyse, the project selection procedure is likely to experience delays that could affect the delivery dates of important documents;
- Students do not have access to a list of all the available teachers (and their respective specializations) to supervise their work. That makes it difficult to choose the most suitable teacher for the project.

After choosing the project to work on during the academic semester or year, the communication process begins between the entity that created the proposal and the student. Depending on the institution that proposed the project, the communication process to formalize the application can be a smooth experience or the complete opposite of that. The worst case scenario for a student would be not being given an answer back from his interest in the proposal. Besides that, having different communication routes between all involved parties can lead to loss of useful data, unavailability to contact people and increased difficulty to accompany the developed work.

During the project development phase, the student and ISEP's supervisor communicates regularly to check the report's grammar and spelling issues, as well as its content. Depending on the supervisor, the monitorization of the report can be done using different platforms such as cloud service providers (e.g.: Dropbox, Google Drive, Microsoft OneDrive, and others), chat services (e.g.: Microsoft Teams, Slack, and others) or email. The problem inherent to using the previously described tools is the difficulty for both the student and supervisor to track notes and corrections made by each one. Thus, it can be problematic to track a report's version history and see if improvements were made.

To tackle the problems described in the previous paragraphs, ML and LMS are explored with detail in Sections 2.2 and 2.3. The rise and evolution of ML technology and constant evolution

and adoption of LMS solutions during the most recent years will allow a system capable of automatizing and monitoring projects to be created.

2.2 Machine Learning

One of the techniques to automatize the attribution and monitoring of projects is ML. ML is one technology derived from "(…) artificial intelligence (AI) movement of the 1950s (…)" (Bi *et al.*, 2019) and its main purpose is to create its own intelligence "(…) without being directly programmed" by a human (Bi *et al.*, 2019). Over the last decade, similar to other technologies, ML suffered an exponential evolution that allows it to be used in multiple situations. According to a journal article published in the Science Magazine, a task that may be suitable for ML is expected to have certain key criteria (Brynjolfsson and Mitchell, 2017):

- Provide a large data set that is applicable to the domain area in cause;
- Learn a "(…) function that can map well-defined classification inputs with well-defined prediction outputs". That function cannot change swiftly over time;
- Precise decision without questioning and "(…) long chains of logic and reasoning". Since optimal answers is not a guaranteed outcome, it needs to have "tolerance for error".

2.2.1 Data Mining Techniques

Even though it is important that a certain task respect the key criteria stated previously, the process of building a new ML system can be based on data mining (DM) frameworks that help organize those types of projects. Despite DM and ML being seen as two different concepts, DM can intercept other "(…) fields, such as artificial intelligence, statistics, database systems, machine learning" (Gullo, 2015).

DM is the term that corresponds to the analysis of large amounts of data to understand unknown patterns and gather useful information(Gullo, 2015).The most traditional methodology associated to DM is the **Knowledge Discovery in Databases** (KDD). KDD "is the process of using DM methods to extract what is deemed knowledge according to the specification of measures and thresholds (…)" in an original and non-modified dataset (Ana Azevedo and Manuel Filipe Santos, 2008). KDD follows several steps to obtain knowledge (Gullo, 2015):

- **Selection:** create a dataset from the original data;
- **Pre-processing:** filter and rearrange data that is suitable for the task;
- **Transformation:** reduce and project data "(…) in order to derive a representation suitable for the specific task to be performed";
- **Data Mining:** process of extracting unknown patterns from the data;
- **Interpretation/Evaluation:** evaluate the patterns found by DM, extracting its underlying knowledge.

Over the years, other DM frameworks were popularized. Examples of such are **Cross-Industry Standard Process for Data Mining** (CRISP-DM) and **Sample, Explore, Modify, Model and Assess** (SEMMA).

The CRISP-DM framework is a project sponsored by the European Commission that is designed to make projects "(…) less costly, more reliable, more repeatable, more manageable (…)", while being faster to develop (Rüdiger Wirth and Jochen Hipp, 2000). It is also considered the standard framework for DM projects (Nguyen *et al.*, 2019). Similarly to KDD, a project passes through a number of procedures, going from a more general approach to a more specific one (Rüdiger Wirth and Jochen Hipp, 2000):

- **Business Understanding:** understand the data to analyse in a business perspective;
- **Data Understanding:** identification of "(…) data quality problems (…)", unknown aspects and subsets from a specific dataset;
- **Data Preparation:** all procedures to build a final dataset;
- **Modelling:** selection and application of modelling techniques to the dataset;
- **Evaluation:** evaluation and review of all datasets that were modelled in the previous phase (in order to assess if the datasets are worth being deployed). It is possible to go back to Business Understanding phase in order to address a business problem that it is not covered in those datasets;
- **Deployment:** Generation of a final report to show to the end user.

The other framework to facilitate DM projects is SEMMA, created by the SAS Institute, that is mostly used with the enterprise solutions offered by SAS (namely SAS Enterprise Miner) (SAS, 1999). Each letter of its acronym represents a different procedure (Ana Azevedo and Manuel Filipe Santos, 2008):

- **Sample:** extract a relevant dataset from the original data;
- **Explore:** exploration of the dataset to identify unknown patterns;
- **Modify:** modification of the dataset to make relevant changes before modelling it;
- **Model:** automatic modelling of the data in order to predict "(…) a desired outcome";
- **Assess:** evaluation of the results from the model phase, by assessing the modelling performance and prevision.

2.2.1.1 Framework Comparison

By analysing the two frameworks described in Section 2.2.1, it is possible to conclude that both are similar to each other (even though CRISP-DM has one more step than the others), because they are all based on the KDD methodology (Ana Azevedo and Manuel Filipe Santos, 2008; Shafique and Qaiser, 2014). However, their implementation in real projects can vary.

On one hand, despite SEMMA's usage not being limited to SAS's enterprise software, the majority of its utilization is associated with the SAS Enterprise Miner program, making it not the most suitable option for projects outside that tech stack.

CRISP-DM on the other hand is the most popular framework for building DM projects since it can also be "(…) combined with a team coordination framework such as SCRUM" (Saltz, 2021) to solve its weaknesses, such as the use of multiple iterations and experimentations, the lack of team roles and relevant phases (Saltz, 2021). Consequently, CRISP-DM with SCRUM is suitable for teams with several roles since everyone can participate in different phases of the DM project. However, due to the extensive documentation of CRISP-DM (Chapman *et al.*, 2000) and experience needed to run it efficiently, this framework may not be convenient for small teams or individual projects.

2.2.2 Recommendation Systems

ML methodologies and algorithms can be appropriate for several applications, one of them being RSs.

Over the years, with the exponential amount of data being generated and saved in data storage, the availability of information has been at an all-time high. This leads to a "(…) problem of information overload, which has created a potential problem to many Internet users" (Isinkaye, Folajimi and Ojokoh, 2015), since it may be difficult for them to know where and what to choose online (Ferreira and Oliveira, 2012). RSs (or recommender systems) were created to "(…) solve this problem by searching through large volume of dynamically generated information to provide users with personalized content and services" (Isinkaye, Folajimi and Ojokoh, 2015). Consequently, those systems help people with their decision-making process by suggesting the most relevant results, while minimizing the time spent searching for them.

A study made by *Fernando Colmenero-Ferreira* and *Adicinéia Aparecida de Oliveira* in 2012 analysing the influence of RSs in the decision-making process on the Web concluded that the behaviour of consumers when looking for information is based on cost and benefit, hence the importance of making it a reality by helping the user with relevant suggestions (Ferreira & Oliveira, 2012). Figure 2 represents a graphic that evaluates the average time spent by strategy time, concluding that the prescription strategy (a method that keeps on recommending new

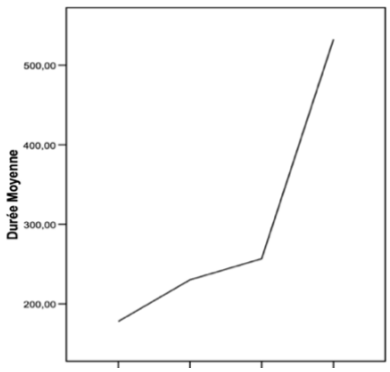


Figure 2 - Average duration of a task by strategy type
(Ferreira & Oliveira, 2012)

suggestions based on the user interests and interaction within the website) takes less than twice the amount of time of exploration strategy when searching for the same result.

Due to the relevance of this topic, most of the biggest corporations in the world have implemented some form of RS to their products, such as Netflix, Microsoft, Amazon and others (Pazzani and Billsus, 2007; Shani and Gunawardana, 2011). With a large enough dataset, a suitable learning/filtering method, a well-defined topic and a complex ML algorithm that can provide relevant results, every product can offer a RS to their end users.

RSs have intrinsically several filtering techniques implemented in order to produce the best accurate recommendations for a specific UC (Figure 3). It is also possible for developers to build their own filtering techniques based on those listed in Figure 3**Error! Reference source not found.**, by using multiple filtering techniques (hybrid filtering).

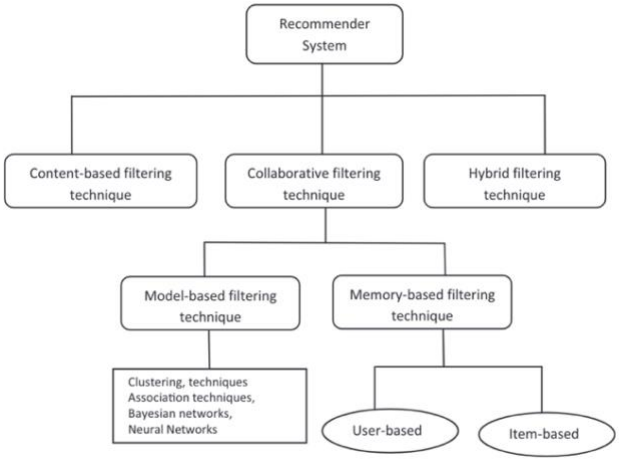


Figure 3 - Recommendation Techniques (Isinkaye, Folajimi and Ojokoh, 2015)

In the following subsections some of the most popular filtering techniques for RSs systems are analysed with more detail.

Content-Based Filtering

Content-Based filtering is a technique that analyses the connection between the user's selected items and past preferences (van Meteren and van Someren, 2000; Lops *et al.*, 2019). Taking that into consideration, this filtering technique tries to create a model "(…) based on a feature-based representation of the content of recommendable items" (Lops *et al.*, 2019). The items are analysed by their correspondent characteristics and content, and later compared with the user's previous actions. Therefore, it is possible to say that the content-based filtering algorithm "(…) recommends items that are similar to those that a user liked in the past" (Jariha and Jain, 2018).

Although this technique is based on the user's previous actions and ratings, it is not get affected from a cold start problem like collaborative filtering, because if there is no user history, the system can still make recommendations based on the content of the items themselves.

However, the recommendations based on this technique are only "(…) from a restrict theme scope (…)" (Nabizadeh *et al.*, 2013) and it only recommends items that are similar to the ones the user has seen. That makes it difficult for the system to recommend items that are somewhat outside of the user's scope but still inside its interest. It can also be less precise because recommendation can be made without user rating (B.Thorat, M. Goudar and Barve, 2015).

Collaborative Filtering

Collaborative filtering is a technique that analyses the "(…) patterns of ratings or usage without need for exogenous information about either items or users" (Koren, Rendle and Bell, 2022), which is something that contrasts with content-based filtering. In order to make recommendations, this method needs to take into consideration both the users and items of the dataset. According to Yehuda Koren, Steffen Rendle, and Robert Bell, there are two different ways for a collaborative filtering technique to make recommendations: memory-based filtering (or neighbourhood models) and model-base filtering (or latent factor models) (Isinkaye, Folajimi and Ojokoh, 2015; Koren, Rendle and Bell, 2022).

On one hand, memory-base filtering fixes on the relationships between users (user-based) or items (item-based). In user-based filtering, recommendations are made based on the preferences of similar users, while in item-based filtering, recommendations are made based on the similarity between items using user's ratings (Koren, Rendle and Bell, 2022). The main disadvantage of this approach is the use of the total or a large portion of the dataset, which can create efficiency and scalability problems (B.Thorat, M. Goudar and Barve, 2015).

On the other hand, model-base filtering makes simultaneous use of the user and items from a small set of information in the dataset and "(…) tries to explain ratings by characterizing both item and users on factors automatically inferred from user feedback" (Koren, Rendle and Bell, 2022). By using lesser data, it is considered more performant and scalable, and capable of providing better prediction accuracy than memory-based filtering (B.Thorat, M. Goudar and Barve, 2015).

Collaborative filtering is a great technique that can be both reliable and easy to implement. However problems such as cold start, sparsity and scalability are aspects to take into consideration when adopting it on a RS (B.Thorat, M. Goudar and Barve, 2015).

Hybrid Filtering

Hybrid filtering combines features of content-based and collaborative methods of filtering to create a RS suitable for a specific UC and "(…) improve recommendation accuracy" (B.Thorat, M. Goudar and Barve, 2015). Besides those advantages, combining several filtering methods in different ways to produce a reliable recommender system also mitigates some of the problems associated with using single filtering methods.

According to Gediminas Adomavicius and Alexander Tuzhilin, there are several approaches to design a hybrid filtering method (Adomavicius and Tuzhilin, 2005):

- Implementing both collaborative and content-based methodologies to combine their predictions;
- Select several features from either one of the filtering methods and add those to their implementation;
- Build a "(…) general unifying model (…)" that uses both filtering methods.

Using those references as a guideline, several hybrid implementations can be created. For instance, by only considering the user's past preferences, content-based filtering could integrate with collaborative filtering methodologies to solve that disadvantage (B.Thorat, M. Goudar and Barve, 2015; Lops *et al.*, 2019).

Examples of hybrid approaches relevant for this dissertation include knowledge-based filtering, utility-based filtering, fuzzy filtering and demographic-based filtering (Burke, 2000; Tarus, Niu and Mustafa, 2018; Lahoud *et al.*, 2022). Those approaches are explained with more detail in subsequent sections.

Knowledge-Based Filtering

Knowledge-Based filtering is a technique that uses the knowledge gathered from a particular domain to make questions to users about their preferences within that domain. Those RSs need to "(…) employ three types of knowledge: knowledge about the users, knowledge about the items and knowledge about the matching between the item and user's need" (Tarus et al., 2018). Taking into consideration the user's answers, the system tries to generate recommendations (Burke, 2000):

Based on such information, the system can pursue a knowledge-based approach to generating a recommendation, by reasoning about what products meet the user's requirements.

This type of hybrid filtering does not experience typical problems of content-based and collaborative filtering techniques such as cold start or rating sparsity problem due to the use of domain knowledge (Tarus, Niu and Mustafa, 2018).

Utility-Based Filtering

Utility-Based filtering is a technique that evaluates the utility of a particular item to a given user. By evaluating the user preferences, it is possible to calculate the utility of an item and the likelihood of that item being chosen by the user (Deng, 2015).

Just like knowledge-based filtering, it is not affected by problems from other filtering methodologies such as cold start or rating sparsity. However, since this technique often uses multi-attribute utility theory (the user decision making process is made of several attributes instead of a few), it makes it difficult to develop a reliable RS with "(…) little user effort" (Deng, 2015).

Fuzzy Filtering

Fuzzy filtering is a technique that takes into consideration the "(…) uncertainty, impreciseness and vagueness in item features and user's behaviour" (Jain and Gupta, 2018). There are cases where the user does not know how to quantify certain item ratings with an exact value. Instead, the user tries to quantify with a range of values (or labels) the item rating, without being so precise.

As with any other hybrid filtering technique, the fuzzy logic can be applied with other techniques to enhance its performance and effectiveness. Research conducted to fuzzy logic within RSs showed significant changes when applied with clustering algorithms, to improve "(…) the determination of similar users (…)" by analysing the fuzzy characteristics between users (Jain and Gupta, 2018).

Demographic-Based Filtering

Demographic-Based filtering is a technique that analyses the demographic information of a user (such as physical attributes, age, status, educational background, and other similar attributes) and compares to other users with similar demographic characteristics, in order to make recommendations. This type of filtering is very similar to collaborative filtering due to the fact that both are capable of "(…) identifying cross-genre niches, enticing the users to jump outside the familiar, and their ability to improve themselves over time" (Al-Shamri, 2016).

2.2.3 Learning Methodologies

ML methodologies can be divided into four main learning types (Ayodele, 2010; Sarker, 2021):

- **Supervised Learning:** Algorithms of this category are trained on a dataset, so that they can map the "(…) inputs to desired outputs" (Ayodele, 2010). The information inside the datasets is connected correctly, so that it is possible to make predictions about unseen examples (based on the training set). Inside this category, algorithms are grouped in two types:
    - Classification: Algorithms that associate labels (outputs) to input variables (inputs). They are normally used for more generic problems, for instance to see if an email is considered spam or not;
    - Regression: Even though classification and regression algorithms have the same core principals, regression algorithms are used to predict a continuous quantity (output) based on input variables (inputs). An application of these algorithms would be to try predicting the height of an individual based on his other physical statistics.

- **Unsupervised Learning**: Algorithms of this category need to learn by analysing similarities inside the dataset. The main objective is to learn how to model the structure

and distribution of the dataset without being told what the data represents. Inside this category, algorithms are grouped in two types:

  - Clustering: Algorithms that try to discover trends or patterns in data by grouping "(…) data points in large datasets without concern for the specific outcome" (Sarker, 2021);
  - Association: Algorithms that use a "(…) rule-based machine learning approach (…)" (Sarker, 2021) to discover patterns in data. It is used in several industries and its main objective is to find similarities in points that are often picked together. For instance, an application of this algorithm is the likelihood of someone buying a given product based on their shopping cart.

- **Semi-Supervised Learning:** Algorithms of this category are a mixture of both supervised and unsupervised learning algorithms because they operate "(…) on both labelled and unlabelled data" (Sarker, 2021). As a result, it tries to take the best characteristics of both worlds.
- **Reinforcement Learning:** Algorithms of this category learn through trial and error by not making the same mistakes made in previous actions. Consequently, these algorithms do not learn through datasets, but learn by interacting with the environment instead.

The previous described methodologies (with the exception of reinforcement learning) can be programmed with traditional ML techniques, yet they are "(…) limited in their ability to process natural data in their raw form" (LeCun, Bengio and Hinton, 2015). Classic ML techniques learn to make predictions based on the features inside datasets, which requires more engineering effort to design a model that could extract information from raw data. To mitigate that and be able to learn automatically without much effort to the development team, the concept of **Deep Learning** (DL) was originated.

DL is a subfield of ML that consists of several representation learning methods (process of learning patterns within the data) with multiple layers of abstraction (LeCun, Bengio and Hinton, 2015). Therefore, DL is capable of recognizing more complex patterns in the data than traditional ML methods. DL is known to be used in several functions such as image and speech recognition, NLP and machine translation, however there are studies that analyse the use of DL in other ML activities such as RSs (Khan *et al.*, 2021). The most popular DL techniques are "(…) Autoencoder, Deep Belief Network, Convolutional Neural Network, Recurrent Neural Network, Recursive Neural Network and Direct Deep Reinforcement Learning" (Shinde and Shah, 2018).

Although DL seems to be the obvious decision to the majority of ML related problems, it has some disadvantages compared to traditional methods. In order to use DL effectively, a very large dataset needs to be provided, which may not be suitable for smaller to medium sized projects (Shinde and Shah, 2018). Besides that, since DL uses more complex abstraction layers, it may be more difficult to assess the decision-making process behind the produced result. Lastly, a hardware machine with a highly capable Graphics Processing Unit (GPU) is advisable to

run the developed algorithms (Shinde and Shah, 2018), and it is something that may not be available in every project.

In conclusion, there is "no one size fits all" approach to choosing the best ML methodology and algorithms for RSs since various contexts need to be considered before. In the next subsections, some ML methods and algorithms for RSs, which belong to a single or multiple learning categories, are described.

2.2.3.1 K-Nearest Neighbour

KNN is a supervised learning ML algorithm that takes into consideration other points (neighbours) inside a model (Cover and Hart, 1967). KNN can be used in both classification and regression problems. The main purpose of this algorithm relies on the classification of a sample based on the data of the closest identical points of the model. Based on the acronym of the algorithm (KNN), the algorithm tries to classify a point based on the K ($K \in]0, +\infty[$) nearest neighbours (NN). According to a report published on the Institute of Electrical and Electronics Engineers, this method has a very low probability of error (Cover and Hart, 1967):

"Surprisingly, it will be shown that, in the large sample case, this simple rule has a probability of error which is less than twice the Bayes probability of error, and hence is less than twice the probability of error of any other decision rule, nonparametric or otherwise, based on the infinite sample set."

KNN classifies its points by measuring the "(…) the difference or similarity between two instances" (Jiang *et al.*, 2007). Even though this algorithm is classified as being simple and effective, according to Liangxiao Jiang and its colleagues at China University of Geosciences, KNN has three main flaws (although those are further discussed in the article and solved) (Jiang *et al.*, 2007):

- The distance being an Euclidean distance;
- Neighbourhood size being assigned as an input parameter;
- Simple voting mechanism being the only method for probability estimation.

2.2.3.2 Matrix Factorization

MF is a ML algorithm that decomposes a matrix composed of information of a dataset into lower-dimensional matrices. Therefore, it can be used in multiple applications, one of them being RSs. In fact, this technique is used in several big corporation such as Amazon and Netflix for their recommender systems (Koren, Bell and Volinsky, 2009). According to Yehuda Koren, MF is applied in RSs by a two-dimensional matrix, one that represents users and the other that represents their items of interest (Koren, Bell and Volinsky, 2009). It is then considered a variant of latent factor models, which transform both the user and item in some latent factor (Koren, Rendle and Bell, 2022). These types of algorithms are mathematically more complex than other algorithm options, however they are consequently more accurate (Mongia *et al.*, 2020).

Since it can address both supervised and unsupervised problems, it is possible that this technique can use additional information, such as user behaviour, to help in the recommendation process (Koren et al., 2009, 2022).

MF is a versatile methodology since it can be implemented in several different formats. The most popular methods include the non-negative MF, Singular Value Decomposition, LU factorization, Cholesky factorization, QR factorization, and other implementations (Ng and Tan, 2021).

2.2.3.3 Decision Tree

Decision Tree is a mechanism that can be used in both regression and classification algorithms (supervised learning). It was one of the first mechanisms to be adopted to electronic form in the 20th century and it is considered a general-purpose methodology since it can be used in a wide spectrum of tasks including "(…) data mining, knowledge discovery, machine learning, and artificial intelligence" (de Ville, 2013).

As the name suggests, decision trees resemble a tree-like shape when seeing it on a diagram (Figure 4). Therefore, it can be seen as a "(…) recursive structure for expressing classification rules" (Quinlan, 1990).

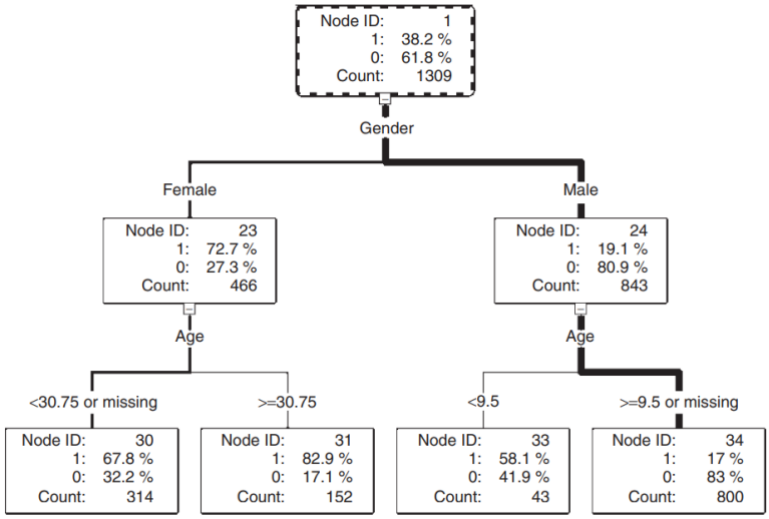


Figure 4 - Example of a decision tree (de Ville, 2013)

The tree is built by analysing the training set, splitting it into several small partitions and placing tests that connect leaves (branches) from different levels. Those tests decide if the new data meets the requirements needed for advancing to the next level. The training set is divided until "(…) every part comprise totally or predominantly of sample from one class", which means that each leaf represents tinier portions of the total dataset the higher the tree level is (Anuradha and Gupta, 2014).

This algorithm tries to predict new data by navigating through the tree and stopping when it reaches a final leaf or if the stopping criteria were already met (Quinlan, 1990; de Ville, 2013).

While it is a simple algorithm to implement, it comes with performance issues when tree complexity is increased. Since it needs to execute every test within the tree, it has a computational cost consequently (Quinlan, 1990).

2.2.3.4 Artificial Neural Network

ANN is a ML methodology that was inspired by how the human brain functions. Human brains are capable of learning on their own because of how neurons work inside the brain. ANN try to replicate the same principle by simulating a network of model neurons (Krogh, 2008):

"By applying algorithms that mimic the processes of real neurons, we can make the network 'learn' to solve many types of problems."

Like the other ML techniques and algorithms explored in Sections 2.2.3.1, 2.2.3.2 and 2.2.3.3, ANN has multiple applications besides RSs, including face recognition, anomaly identification on medical images, control of electronic devices and appliances, and other functions (da Silva *et al.*, 2017).

At its core, ANN are composed of multiple layers (da Silva *et al.*, 2017):

- Input: This type of layer is responsible for receiving information and it is the only type of layer that does not have artificial neurons, since its main purpose is to act as a conduit for the data that is passed to other layers. Before passing it, the information is first normalized to produce "(…) better numerical precision for the mathematical operations performed by the network";
- Hidden: This type of layer has artificial neurons that are responsible for identifying patterns and features in the data. It is on these layers that most of the internal processing occurs;
- Output: This layer has artificial neurons that are responsible for producing the final network output with the processing that occurred in previous layers (such as hidden layers). Those layers can have different architectures depending on the problem to tackle. These architectures can be: single-layer feedforward, multilayer feedforward, recurrent and mesh networks.

As stated previously, most layers are composed of artificial neurons. Artificial neurons mimic the behaviour of biological neurons by receiving an input from other neurons or external sources and then performing a computation on that input. Then the results obtained are sent to other neurons in the next layer, and the process repeats.

However, artificial neurons have a more complex structure behind them, and according to Ivan Nunes da Silva and co., neurons are composed by seven elements: input signals, synaptic weights, linear aggregator, activation threshold, activation potential, activation function and output signal (da Silva *et al.*, 2017).

2.2.4 Natural Language Processing

Although NLP is a field of ML and AI that analyses the content of human written language, it can be also used to enhance RSs. Items that are evaluated in a RS can have important information that is not being processed by traditional algorithms, such as user reviews or product descriptions, both of which can hide relevant details. In order to improve NLP's efficacy, it is necessary to pre-process the input sentences by removing stop words, removing prefixes and suffixes to obtain the root word, and/or other techniques.

NLP techniques are applied to process that information by using offline (e.g.: using text reviews in collaborative filtering, or product description in content-based filtering) or online inputs (e.g.: talking directly to the system to retrieve more custom recommendations, like the ChatGPT tool explained in Section 2.4.1.6) (Shalom, Roitman and Kouki, 2022). Most NLP state-of-the-art algorithms apply DL techniques under the hood to better analyse intrinsic patterns in the text. Examples of that are DeepCoNN or TransNets algorithms (Catherine and Cohen, 2017; Zheng, Noroozi and Yu, 2017; Shalom, Roitman and Kouki, 2022). However, NLP is not exclusive to DL and there are also techniques that can be used in conjunction with ML applications, such as the TF-IDF algorithm (Renuka, Raj Kiran and Rohit, 2021).

TF-IDF is a text similarity algorithm that is responsible for measuring the importance of a given word, sentence or expression in a collection of documents. The text is represented as a vector by taking into consideration the two main parts that characterizes the algorithm (Lan, 2022):

- Term Frequency: Determines how frequent the word or expression appears in the document. The more frequent it appears, the more relevant it becomes;
- Inverse Document Frequency: Determines how frequent a word or expression appears in a collection of documents. The more frequent it appears in several documents, the less relevant it becomes.

By calculating the TF-IDF vector it is possible to assess the similarity of texts by using algorithm such as cosine similarity (Lan, 2022).

2.2.5 Evaluation

In order to assess which algorithms perform the best in a particular RS, evaluations are used in different settings with several property-based evaluation metrics. Depending on the evaluation settings, some properties might have different results or are impossible to measure. For instance, the trust property of a RS is impossible to assess in an offline evaluation environment (Gunawardana, Shani and Yogev, 2022):

"It is unclear how to measure trust in an offline experiment, because trust is built through an interaction between the system and a user."

Therefore, it is advisable that several different implementations are considered depending on the evaluation setting being utilized. In the next subsections, several evaluation settings and metrics are analysed with more detail.

2.2.5.1 Evaluation Settings

There are three ways to test RSs: offline, user studies and online. Each one of them has its own advantages, disadvantages, and its own set of difficulties. In this section, a brief explanation of those settings are presented according to Asela Gunawardana, Guy Shani and Sivan Yogev (Gunawardana, Shani and Yogev, 2022).

Offline

Offline experimentation uses a selected dataset of users or rating items to "(…) simulate the behaviour of users that interact with a recommendation system", since it is assumed that the data collected in the dataset is similar to the user behaviour once the RS is online (Gunawardana, Shani and Yogev, 2022). Hence, it can be seen as a disadvantage of this testing setting because there is no possibility to reproduce user interaction with the system, which is a major component to consider.

User Studies

User studies are used as a complement to offline experimentations since it is difficult or almost impossible to simulate user behaviour and interaction with it. These studies are handed to a selected group of people where the main objective is to track their interaction with the RS when doing several proposed tasks. Besides that, questions can be asked before, during, and after the task completion to collect additional relevant data.

Even though it is an exceptional way to test user interaction, it comes with time efficiency and cost problems. One of the reasons is that it is expensive to pay for people to test the RS. It is then advisable to be cautious about the results, since they can be biased "(…) to try and satisfy the person or company conducting the experiment" (Gunawardana, Shani and Yogev, 2022).

However, even if the majority of users were volunteers, the time constraint would still be a problem since the same tasks would need to be executed several times in order to take plausible conclusions.

Online

Online experimentation can be seen as the most complete test setting since it is tested by real users with as much freedom as they want, since they can perform whichever task they desire. However, such tests can be detrimental to the final product since a bad RS might lead people to stop using it in a non-testing environment. It is recommended to conduct an extensive offline experimentation set in order to mitigate those risks.

2.2.5.2 Property-Based Evaluation Metrics

Several scientific studies suggest that a RS is evaluated and measured by the accuracy of prediction and coverage (Isinkaye, Folajimi and Ojokoh, 2015; Raghuwanshi and Pateriya, 2019) since those properties can be measured through different equations on "(…) filtering technique, features of data set, and the task of recommendation system" (Raghuwanshi and Pateriya, 2019).

However, according to several other studies, there are a wide number of evaluation properties to classify a RS. The listed properties in those scientific articles are: user preference, prediction accuracy, coverage, confidence, trust, novelty, serendipity, diversity, utility, risk, robustness, privacy, adaptability, scalability (Wu, He and Yang, 2012; Gunawardana, Shani and Yogev, 2022).

It is worth mentioning that a RS is not evaluated by all the properties referred in the last paragraph, since some of them can be traded off. Hence, it is important to assess which properties are the most valuable ones to measure depending on the RS domain.

In the next subsection, the metrics associated to the prediction accuracy are analysed with more depth, since they can be mathematically calculated and classified, in contrast with the other general properties.

Prediction Accuracy

Prediction accuracy measures how well the RS can generate accurate recommendations to the end users. It is often the property that most developers tend to prioritize since it is assumed that a user continues to use the system depending on its prediction accuracy (Gunawardana, Shani and Yogev, 2022).

There are several metrics to assess prediction accuracy, which are RMSE and MAE. The lower the value of these metrics, the better the RS is at predicting (Isinkaye, Folajimi and Ojokoh, 2015; Raghuwanshi and Pateriya, 2019; Gunawardana, Shani and Yogev, 2022).

MAE is an evaluation metric that measures the deviation of recommendation from user's specific value. It is calculated by the absolute difference between predictions ($\hat{r}_{ui}$) and actual values (r_{ui}). N stands for the number of ratings inside the dataset and ui represents the pair user-item in a RS.

$$MAE = \frac{\sum_{(u,i) \in N} |\hat{r}_{ui} - r_{ui}|}{N}$$

RMSE is an evaluation metric that penalizes large errors even further, even if it only happens on only one rating (Gunawardana, Shani and Yogev, 2022). It is calculated by the square root of the average of the difference between predictions ($\hat{r}_{ui}$) and actual values (r_{ui}). N stands for the number of ratings inside the dataset and ui represents the pair user-item in a RS.

$$RMSE = \sqrt{\frac{\sum_{(u,i) \in N} (\hat{r}_{ui} - r_{ui})^2}{N}}$$

Besides measuring prediction accuracy with equation metrics, it is also possible to evaluate prediction accuracy through classification. These metrics are computed in a confusion metric with actual and prediction values, and those are classified based on four indicators (Raghuwanshi and Pateriya, 2019; Gunawardana, Shani and Yogev, 2022) (Table 1).

Table 1 - Classification of the possible result of a recommendation of an item to a user

	Recommended	**Not Recommended**
Used	True Positive (TP)	False Negative (FN)
Not Used	False Positive (FP)	True Negative (TN)

Examples of classification metrics are precision, recall, false positive rate and F-Measure.

Precision calculates the relevance of predictions to the user, by calculating the fraction between the useful recommended items and all recommended items retrieved by the system. The higher the precision value is, the better the RS is at predicting.

$$Precision = \frac{TP}{TP + FP}$$

In addition, recall also calculates the relevance of predictions to the user but in a different manner, since it analyses the quantity of useful recommended items to the number of all useful items. The higher the recall value is, the better the RS is at predicting.

$$Recall = \frac{TP}{TP + FN}$$

False positive rate is the exact opposite of recall, since it analyses the quantity of recommended items that were not useful to the user in comparison to the number of total non-useful items. The higher the false positive rate value is, the worse the RS is at predicting.

$$False\ Positive\ Rate = \frac{FP}{FP + TN}$$

The final metric of classification is F-measure, which is a simplification method that englobes both precision and recall metrics.

$$FMeasure = \frac{2 * Precision * Recall}{Precision + Recall}$$

2.2.6 Technologies

With the evolution of ML in recent years, many programming languages have been adapted to allow programmers and developers to create ML applications with them. These types of applications are harder to develop since besides having to know how to program, the developer also needs to know other essential areas such as math, information theory and statistics (Wan *et al.*, 2020). In addition, developers need to take into consideration that software engineering

practices change between ML and non-ML applications. Those changes happen during the whole software engineering process, including requirement engineering, design, development and testing (Wan *et al.*, 2020).

It is then important to choose a programming language that can help developers accelerate the coding process. One of those programming languages is Python, which according to several sources is one of the most popular programming languages for ML and the most used one for ML programs (Srinath, 2017; Nagpal and Gabrani, 2019; Cohen, 2021).

In this section, a selected number of Python frameworks are analysed in greater depth. In the end, a comparison between all the analysed technologies is made to discover which coding framework is most suited depending on the project in hands.

2.2.6.1 Python

Python is an object-oriented, high-level, multi-purpose programming language created by Guido van Rossom that is becoming one of the most popular programming languages, and one of the most popular options for ML applications (Nagpal and Gabrani, 2019) (Srinath, 2017). One of its strengths is the fact that it is a uncomplicated programming language, that has "(…) a very simple and elegant syntax" and better readability than the likes of C++, Java or C#, making it a good choice for beginners (Srinath, 2017).

Compared to other languages such as Java and R, Python is pointed out as the one with the highest value of run time speed for ML tasks. However, it is also the one that has the least amount of memory utilization and the faster to make a ML program, due to the simpler syntax (Johnson and Chandran, 2021).

By having high popularity, Python has a considerably sized developer community and offers a wide spectrum of open-source frameworks for ML, including Scikit-Learn (built on top of NumPy, SciPy and matplotlib), and DL frameworks such as TensorFlow, Keras and PyTorch (Raschka, Patterson and Nolet, 2020) (Nagpal and Gabrani, 2019). With the amount of proven ML frameworks existent in this programming language and the fact that it is well suited for most environments (because of being written in C, C++ and other languages), Python is being seen as a replacement option to other alternatives on the market, such as MATLAB and R (Nagpal and Gabrani, 2019).

Scikit-Learn

Scikit-Learn is a ML framework built with Python that is suitable to all ML learning methodologies types described in the beginning of Section 2.2.3. (Kramer, 2016). It is one of the most popular Python ML libraries that is being used by several corporations on their commercial products. One of its advantages relies on being able to solve some of the efficiency issues from Python by allowing developers to implement algorithms in C language and integrate them with a tool called Cyphon (Kramer, 2016). Other programming languages libraries can also be implemented in Scikit-Learn, such as Fortran and C++ (Hao and Ho, 2019).

Scikit-Learn projects have an implicit structure to be followed (Hao and Ho, 2019):

The Scikit-learn package covers four main topics related to machine learning. They are data transformation, supervised learning, unsupervised learning, and model evaluation and selection.

Since this framework is built on top of other well established Python libraries such as NumPy, SciPy and matplotlib, it offers methods for various phases of ML projects. Firstly, it is possible to import datasets using auxiliary functions from NumPy (Kramer, 2016).

The next step is to use one ML learning methodology. Due to the fact that the algorithms explained in Section 2.2.3 are used in several commercial solutions and are quite popular, Scikit-Learn includes implementations of those algorithms as packages to import (Hao and Ho, 2019).

The last part in a Scikit-Learn project is the model evaluation and selection. The training and test set originated in the previous step can be checked through the execution of cross-validation method (Kramer, 2016; Hao and Ho, 2019).

TensorFlow

TensorFlow is a ML library created by Google and released in November of 2015 as an open-source project that is highly versatile due to allowing developers to program in various programming languages when needed (Low-level Python API, C++, Java and Swift). TensorFlow can run in both desktop and/or mobile environments and is also considered the "(…) most widely used deep learning library in the industry (…)" (Géron, 2022).

TensorFlow is mostly used for large ML projects because of the support for GPU and Tensor Processing Units (TPU) usage and heavy computation, however it can be used for other ML tasks (Géron, 2022). Figure 5 shows what a typical TensorFlow project architecture looks like.

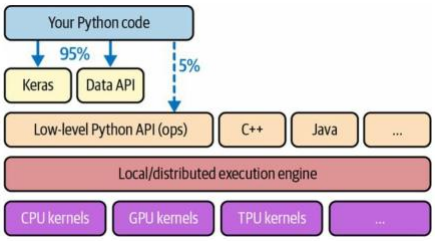


Figure 5 - Tensorflow Architecture (Géron, 2022)

TensorFlow as a wide spectrum of applications and can be seen as one of the most complete technologies at the date of writing of this dissertation for ML and DL tasks. This library was successfully deployed across many production projects across several computer science areas, including "(...) speech recognition, computer vision, robotics, information retrieval, natural language processing, geographic information extraction, and computational drug discovery" (Abadi *et al.*, 2016).

As discussed in the scientific paper "A Tour of TensorFlow" by Peter Goldsborough, TensorFlow represents ML algorithms as computational graphs, because it is an uncomplicated method to visualize and comprehend its representation (Goldsborough, 2016). Nodes are referred to as

operations, which can be "(…) a mathematical equation, a variable or constant, a control flow directive, a file I/O operation or even a network communication port" (Goldsborough, 2016), while edges are referred to as tensors that represent data flow between operations. Computational graph execution is performed in a specific session, which evaluates tensors. The execution process starts in a particular session, which subsequently sends that information to the master process that delegates the task to one or several set of workers divided into one or more devices (Figure 6).

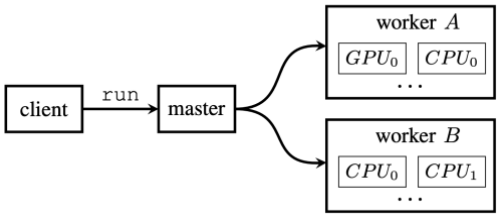


Figure 6 - Computational Graph Execution Example (Goldsborough, 2016)

Keras

Keras is a DL API created by François Chollet and released in March of 2015 as an open-source project that works exclusively with TensorFlow since version 2.4. Due to that fact, Keras cannot be used as a standalone API and needs to be used inside a TensorFlow project, however in previous versions of the API, it was compatible with other ML libraries (Géron, 2022). Since it is a DL library, it allows to "(…) train, evaluate, and execute all sorts of neural networks" (Géron, 2022).

Figure 5 shows that since TensorFlow allows low-level code to personalize the developed algorithms, Keras can be a more useful tool for more generic DL algorithms, since it is seen as a high-level library.

Every Keras code starts with a model instantiation. The simplest model that Keras provide is the Sequential model, which is "(…) a linear pipeline (…) of neural network layers" (Gulli and Pal, 2017), though it may not be the best option for more complex structures. After defining the model, it is possible to add layers (e.g. Dense layers) to it and assign its neurons specific weights. In order to transform the linear model into a non-linear model that is capable of discovering non-linear patterns in the data, the neural network layers need an activation function attached to them with the goal of introducing non-linearity into the output of a neuron. The most common activation functions in Keras are sigmoid and rectified linear unit (Gulli and Pal, 2017).

PyTorch

PyTorch is a ML and DL library that was developed and maintained by Meta (previously Facebook) that was created as a faster alternative to NumPy by enabling GPU utilization and a DL platform that provides "(...) maximum stability and speed" (Ketkar and Moolayil, 2021). Since it is an alternative to NumPy, developers familiar with that framework have a smoother transition going with PyTorch. It is also worth mentioning that like the other libraries analysed,

it supports C++ algorithms to improve performance against Python (Ketkar and Moolayil, 2021). Performance is something that is greatly considered by PyTorch, because even though the simplicity of creating ML algorithms is one of its biggest advantages, performance is not ditched to achieve that (Paszke *et al.*, 2019):

"To be useful, PyTorch needs to deliver compelling performance, although not at the expense of simplicity and ease of use. Trading 10% of speed for a significantly simpler to use model is acceptable; 100% is not. Therefore, its implementation accepts added complexity in order to deliver that performance."

In fact, even though PyTorch is a Python framework, it contains a substantial amount of non-Python code in form of C++ and NVIDIA's programming language CUDA (Stevens, Antiga and Viehmann, 2020). The framework adds several optimizations such as a C++ core, separate control and data flow, custom caching tensor allocator, multiprocessing and reference counting.

PyTorch is increasing in popularity in the last few years as a common ML and DL framework, and it can be due to the fact that it respects the 'everything is just a program' philosophy (Paszke *et al.*, 2019). What that means is that models, optimizers and data loaders in PyTorch are usually expressed as Python classes, promoting good code quality and object-oriented patterns (Paszke *et al.*, 2019).

2.2.6.2 Technology Comparison

A brief objective comparison of all the frameworks analysed in Section 2.2.6.1 is represented in Table 2.

Table 2 - Framework Objective Comparison

	SCIKIT-LEARN + SURPRISE	**PYTORCH**	**TENSORFLOW + KERAS**
LEVEL	High-Level	High / Low-Level	Low-Level (TensorFlow) / High-Level (Keras)
OPEN-SOURCE	Yes	Yes	Yes
CPU AND GPU UTILIZATION	Only CPU	Yes	Yes
LANGUAGE SUPPORT	Python	Python and C++	Python and C++ (with other language support for Java, Swift, Javascript and mobile support with TensorFlow Lite)
POPULARITY	3rd	2nd	1st

All of the options analysed in Section 2.2.6.1 are some of the most popular Python libraries for ML applications because they allow the implementation of the most common ML algorithms with relative ease. Besides that, since those frameworks are supported by the same programming language, they do not have a wide range of differences in terms of output. Interpreting on what was discussed, the main differences lie on the coding syntax, the level the framework allows to write algorithms and overall performance.

Firstly, it is worth mentioning that Scikit-Learn does not have the necessary specific implementations for RSs built in, so options like Surprise can be used as a more user-friendly approach (Hug, 2020). Therefore, implementing RSs with Scikit-Learn increases the learning curve for people not used to that framework (besides being considered a high-level ML framework). The same can be said for TensorFlow and Keras. The difference is that, being a low-level framework, TensorFlow allows for more complex RS implementations, while Keras is used as a higher-level alternative for more simple and common problems. Starting at TensorFlow's 2.4 version, Keras is deeply integrated with TensorFlow, providing a seamless experience.

In terms of performance, PyTorch seems to have the advantage in comparison to TensorFlow in attributes such as forward and backward measure in Central Processing Unit (CPU) and GPU (Goldsborough, 2016) as well as throughput across several different models, which PyTorch was 17% behind of the fastest framework (Stevens, Antiga and Viehmann, 2020). In spite of that, those tests were only executed in one specific hardware machine, and performance can vary depending on the running environment (which is something to take into consideration when choosing one of those technologies since certain frameworks do not work on specific devices) (Zhang, Wang and Shi, 2018). In fact, there is one study that concludes that PyTorch is more accurate than TensorFlow and Scikit-Learn, however it is much slower than the other two solutions when the number of epochs in a model increases (Gevorkyan *et al.*, 2019).

In conclusion, all of these ML and DL frameworks are capable of providing good results when implementing ML applications, and RSs are no exception. Scikit-Learn should be paired with a RS library for better development experience. PyTorch can be a better alternative for developers that have an object-oriented programming background, however by having that structure it does not give the same flexibility that frameworks like TensorFlow provide. Being the most popular Python ML framework, TensorFlow has a more robust open-source community, more language and device support, and the option of using Keras for higher-level DL algorithms.

2.3 Learning Management Systems

The other mechanism to automatize the monitorization process of projects is LMS. LMS is a digital solution that "(…) integrate a wide range of pedagogical and course administration tools" for students, teachers and school administrative staff (coates, james and baldwin, 2005). These features allow for the possibility of remote online learning (or e-learning) and can enhance the learning experience by saving time in more complex manual tasks (Cavus, 2015).

LMS is considered a versatile software since it can be implemented in several different ways and can only have the features that are applicable to the UC in cause.

There are several LMS options that institutions can choose in order to enhance their digital learning experience. Depending on the LMS, it can be classified as being an open-source technology or a proprietary software. According to Imed Bouchrika, the most popular LMS options (the ones with the most market share in the United States) are Moodle, Blackboard, Canvas and Brightspace (Bouchrika, 2022). All the stated software solutions are paid, with the exception of Moodle, which is open-source and well accepted by the community, being the one that "(…) can compete with the commercially available LMS systems" (Cavus, 2015). According to Nadire Cavus, some of the most common LMS features include (Cavus, 2015):

- Delivery of knowledge to students in various formats, including Word, PowerPoint, video, and/or audio;
- Assessment of students via homework and exams;
- Delivery of the results to students;
- Communication between student-student and student-teacher;
- Student Interaction with lessons' content;
- Scheduling and class management;
- Keeping records for students and teachers (e.g. via logs);
- Tracking student attendance records;
- Students seeing their own class times;
- E-learning content;

Although there are several LMS options available in the market, there are a select number of UCs where the implementation of those systems is not the recommended option, due to the excessive list of features that would not be utilized. For those specific cases, the development of a system containing a selected amount of features stated previously and other custom functionalities is something to take into consideration when implementing a LMS. Therefore, technology choice for implementing these types of systems is non-relevant, since modern programming languages can be used to achieve those functionalities.

2.4 Related Work

In this section, a list of related applications and systems are analysed in more depth for both systems to be developed: RS and LMS respectively.

2.4.1 Recommendation Systems

Since this topic is very specific, it was only possible to find one commercial solution (ChatGPT). However, there are several papers on this dissertation topic that explore more concrete solutions with several relevant features.

2.4.1.1 Course Recommendation System

A paper that studied several algorithms for course RSs (to help students decide which are the appropriate courses to take in their academic career) analysed and compared "(…) their performance by using a real educational data set" (Huynh-Ly Thanh-Nhan, Huu-Hoa Nguyen and Thai-Nghe, 2016).

The developed system is composed of two main applications: one web app for students and educators to make use of the RS, and one separate application for system admins to train and/or retrain the model with several different parameters. Even though it is not directly correlated to the dissertation's theme, it evaluates course recommendations based on the students' grades, which could be something to keep in mind when recommending projects to students.

The algorithms compared in the research were baseline predictors, KNN, MF and biased matrix factorization (BMF). By the end of the comparison, the prototype system was built with BMF because it produced the most accurate results (with the least RMSE of all algorithms) (Huynh-Ly Thanh-Nhan, Huu-Hoa Nguyen and Thai-Nghe, 2016).

2.4.1.2 Thesis Topic Recommendation System

Other scientific research made use of the simple additive weighting method, a method that is popularly used in multi-criteria decision making (Nurmalini and Rahim, 2017), in conjunction with methods such as K-Means Classifier and Naïve Bayesian Classifier to build RSs systems for thesis topics (Kusuma and Musdholifah, 2021).

This system was created due to the number of thesis options a student can choose, which can be overwhelming when choosing the best one. The system asks for the courses which the student attended and is interested in, and the corresponding grades. After getting the information, the system analyses the course syllabus and compare them to the thesis abstracts to see which ones it can recommend. By doing that, this system allows to not only take interests into consideration but also recommend a thesis within the student's abilities.

Using those techniques, the system was capable of achieving a "(…) score with relevance 77.33%, novelty 90.67%, serendipity 80% and increasing recommendation diversity 84%" (Kusuma and Musdholifah, 2021), resulting in an average score of 83% of the system objective, the metric used for evaluating the RS.

2.4.1.3 Internship Recommendation System

A group of the Department from Computer Science in Nigeria developed a web-based internship RS platform using a combination of several filtering methods, such as content-based, collaborative and knowledge-based filtering "(…) to suggest relevant internship organizations that match student's aspirations (…)" (Olasehinde *et al.*, 2022).

Under the hood, this system filters from student's past internship records and ratings to recommend relevant internship organizations to the students. To achieve that, the student is

asked to fill some data to generate recommendations, such as course of study, area of interest within that course and preferred local area for the internship.

This particular study does not provide an evaluation study for the recommender system nor the algorithms used, but by using an hybrid filtering approach it maximizes the recommendation efficacy while minimizing the downsides of using only content-based or collaborative filtering.

2.4.1.4 Supervisor Recommendation System

A case study conducted in the Engineering Institute of Technology in Australia that analysed the process of allocating supervisors for specific thesis used a decision tree algorithm implementation with Python (and suggested the random forest algorithm as a future research theme). The manual process described in the paper is explained as follows (Yuanyuan Fan, Ana Evangelista and Hadi Harb, 2021):

1. The course coordinator selects an important feature;
2. Supervisors that have that feature are kept;
3. Another feature is selected;
4. Steps 2 and 3 are repeated until it reaches a stopping criterion.

Supervisors are then rated according to their answers. If they match the feature, they are assigned with a 1, otherwise they are classified with a 0. Despite that, there are certain supervisors that cannot be classified directly with a 1 or a 0, so they are classified with 0/1. Depending on the stopping criterion, this algorithm recommends the most suitable supervisors for a specific thesis.

The model was tested using the `DecisionTreeClassifier` method in Python and the accuracy score generated was 0.75.

2.4.1.5 eThesis

eThesis is a LMS solution that is explored in more detail in Section 2.4.2.2. However, that system has an integrated RS (Tsatsaris and Sakkopoulos, 2021). That system has similar characteristics to what is expected to be achieved in the dissertation's project.

Upon registering in the platform, students have the opportunity to specify their interests in order for the RS to filter the best projects for them. The interests are filled in the user profile section of the webpage. Figure 7 shows all the recommended thesis for the student.

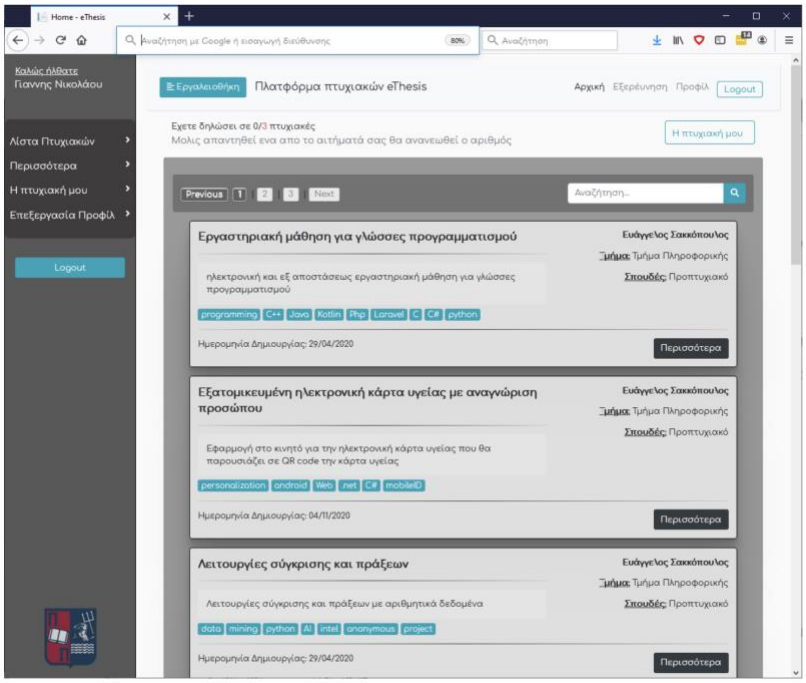


Figure 7 - Personalized Student Thesis List (Tsatsaris and Sakkopoulos, 2021)

To evaluate the usefulness and performance of the recommendations made by the RS, the system tracks user statistics. Those statistics are gathered with different methods such as what theses the students clicked, and their position in the list of recommended projects. The order of the list is based on a cosine similarity algorithm that measures "(…) similarity between documents (theses)", tags and content (Tsatsaris and Sakkopoulos, 2021). The closer the value of a project using the cosine similarity algorithm is to 1, the better it is ranked and the higher it is placed on the recommended list.

The authors of eThesis tested the performance of the RS using the precision and recall method, and the r-precision metric. Precision and recall concluded that as more theses are recommended, the more non-relevant theses are also retrieved, however the top 5 thesis have a score above 75%. In spite of that, r-precision metric indicates that the value for 10 thesis is 0.425, which is higher than any other inferior value of thesis (even though it is still considered a low value of r-precision).

2.4.1.6 ChatGPT

ChatGPT is a chatbot tool created by OpenAI that uses generative AI (way of generating "(…) new and unique content with the trained data" (Aydın and Karaarslan, 2023)) to answer questions from its users in real time. It uses NLP to understand and answer questions from humans, whether from speech or written text (Aydın and Karaarslan, 2023). Since it was trained "(…) on a massive corpus of conversational text (…)" including "(…) customer service contact transcripts, online discussions, and other sorts of spoken or written interaction", it is an application that can be used in a multitude of areas and scenarios (Aydın and Karaarslan, 2023).

ChatGPT has gained enormous popularity due to the quality and effectiveness of its answers, and the fact that has a free tier for everyone to use (at the time of writing the dissertation – October 2023).

Even though it is not a system that directly targets the project recommendation system department, it can be used for UCs such as recommending project suggestions based on the student's preferred themes.

To test its effectiveness, the author of the dissertation tested the application by asking ChatGPT project recommendations for someone who is pursuing a master's in software engineering but is unsure on what theme to explore. The interaction is shown in Figure 8, where ChatGPT was capable of recommending eight different project topics. Since this application is a chatbot, a normal conversation can be made with it by asking it to explore some of those options with even more detail for a more concrete answer.

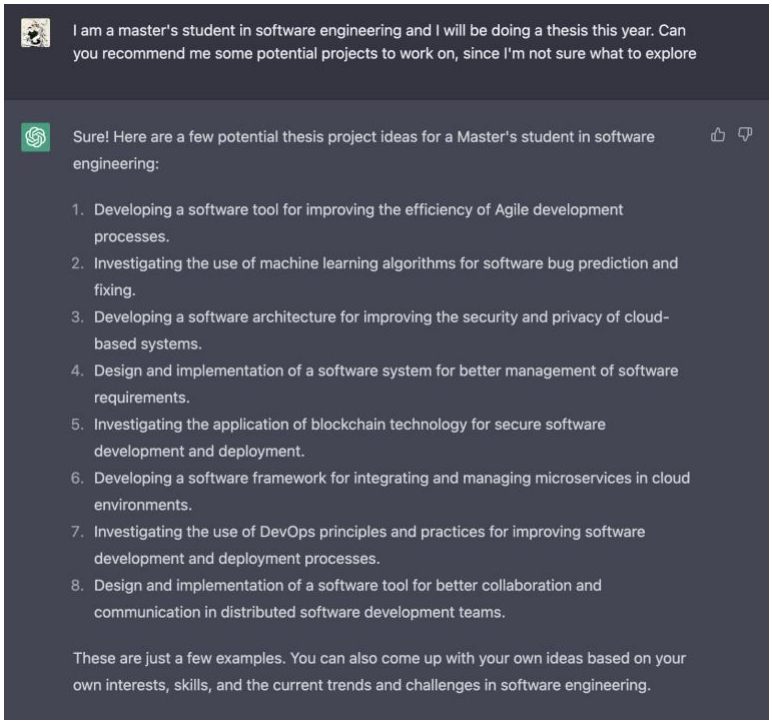


Figure 8 - ChatGPT Project Recommendations

Although the system was able to recommend several projects, it comes with some limitations. Since the application is not connected to the Internet and it was trained with datasets before 2021, it can suggest incorrect and inaccurate information on certain topics (OpenAI, 2023; Thorp, 2023). In the case represented in Figure 8, there is a possibility that the suggested projects can no longer be interesting to investigate at the time of asking the questions. Therefore, the chatbot should be used as a guideline for potential and more developed ideas and not as a definitive answer.

2.4.1.7 Recommendation System Comparison

A brief objective comparison of all the frameworks analysed in Section 2.4.1 is represented in Table 3.

Every RS analysed in section 2.4.1 targets potential topics that are relevant for the dissertation's RS.

The course RS determines the best courses for the student by analysing his grades and interests. Although the purpose of the dissertation's RS is not to suggest the best courses to the student, the principles behind it can be applied when choosing a thesis topic.

The thesis topic RS is more specific towards what is developed in this project. The recommendation process is based on a questionnaire about the student's interests. Like the course recommendation system, this one also uses student's grades to assess which projects to recommend.

Table 3 – RS Project Comparison

	COURSE RS	**THESIS TOPIC RS**	**INTERNSHIP RS**	**SUPERVISOR RS**	**ETHESIS**	**CHATGPT**
COMMERCIAL SOLUTION	No	No	No	No	No	Yes (free at the time of writing)
RECOMMENDATION INPUTS	Student's grades and interests	Student's grades and interests	Student's abilities, course of study, area of interest and preferred local area	Supervisor's answers	Student's grades and interests	User message
RECOMMENDATION FILTERING/ ALGORITHMS	BMF	K-Means Classifier and Naïve Bayesian Classifier	Hybrid Filtering	Decision Tree	Not Known	Generative AI

In comparison to the thesis topic RS, the internship RS also opts to utilize the questionnaire mechanism to obtain not only the student's interests and abilities, but also other vital properties when choosing an internship like course of study, area of interest within that course and preferred local area.

Another important RS to take into consideration is the supervisor RS. This application utilizes a different type of recommendation strategy in form of a decision tree algorithm to classify supervisors depending on the type of project to monitor.

Thus far, there were only analysed RS that tackle specific topics. However, eThesis is considered an end-to-end product that contains a RS system that shares many characteristics with the dissertation's RS. In addition to the thesis recommendation, it can generate user statistics to assess how relevant the recommendations are, and learn from them.

The last analysed project was ChatGPT, which is the only platform (out of all the compared RS) on the market. Despite the fact that ChatGPT is considered a generative AI tool, it can be used as a RS for thesis projects and themes. With the use of NLP, students can ask the chatbot about thesis themes to research.

All the analysed have relevant implementations that can serve as inspiration to come up with the best solution for the dissertation's problem. In this specific case, the conjunction of different techniques from the thesis topic RS, internship RS and supervisor RS can tackle the majority of the dissertation's problem by using the questionnaire, student's grades and abilities and supervisor allocation to the projects.

2.4.2 Learning Management Systems

Contrary to the recommendation systems section, it was possible to find production ready solutions for LMS that are specifically designed for final project and thesis processes.

2.4.2.1 Supporting the Scientific Process System

Supporting the **Sci**entific **Pro**cess (SciPro) is a system created by the The Department of Computer and Systems Sciences at Stockholm University with the purpose of maximizing the time supervisors give feedback to their students in the thesis process, while minimizing the number of administrative tasks to take into consideration.

This program was created due to students complaining by the lack of instructions for developing thesis, as well as "(…) infrequent and insufficient supervisor feedback" (Hansen and Hansson, 2015).

SciPro has a large number of features, including supervisor and student allocation to thesis in an Idea Bank, communication channels, forums and frequently asked questions section, templates for thesis, milestones, peer-reviews and seminar schedules (Figure 9).

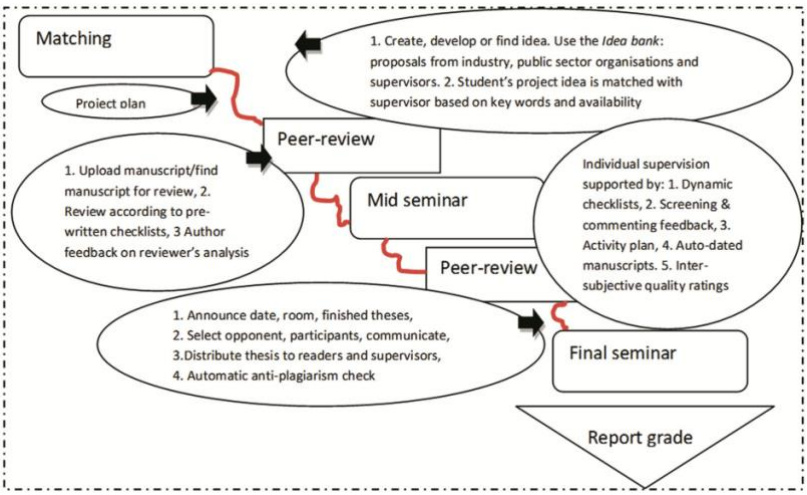


Figure 9 - SciPro Thesis Management System (Hansen and Hansson, 2015)

Even though SciPro offers communication channels for both the student and supervisor, their communication setting is up to them, because there are different communication approaches within the application.

This LMS is now a commercial product licenced by SciMind AB, and it has an exclusive partnership with the Stockholm University. There are also several studies of the implementation of SciPro in other universities, such as the University of Rwanda (Byungura, 2015). However, it's supposed price of 6000 dollars per 6 month trial is not an amount of money that many institutions can afford (Tuhkala and Kärkkäinen, 2018).

2.4.2.2 eThesis

eThesis is a solution developed at the University of Piraeus in Greece that was initially created to combat the consequences of the COVID-19 lockdown and students' working careers. Therefore, students did not have the opportunity to meet and interact directly with teachers.

This application provides a single point of contact for the whole thesis process, that includes both teachers and supervisors. In their own words, Eythymios Tsatsaris and Evangelos Sakkopoulos, the authors of eThesis, say that (Tsatsaris and Sakkopoulos, 2021):

> The eThesis SPOC proposed is an online platform for students that may search to find thesis according to their interests and for professors who see their research ideas become a reality.

Even though it is an internal software to be used in their university systems, it has an extensive list of functionalities, including a thesis topic management mechanism, grading, deadlines, bookmarks and possibility of visualizing detailed statistics and graphs (Figure 10).

Functionality — Approach	TCM [1]	TK [2]	UWA [3]	TC [4]	TH [5]	**eThesis**
Student interests in profile			*			*
Prerequisites to apply for thesis			*		*	*
Courses integration				*	*	*
Add transcript of records to solution					*	*
Shared space for professor-student						*
External third party login						*
Multiple institutions						*
Multiple programs						*
Professor in multiple programs						*
Email notifications				*	*	*
Statistics & graphs						*
Reports			*	*		*
Industry standard security						*
User profile				*	*	*
Theses topic managment (copy, move, transfer)						*
Academic year integration	*	*	*			*
Thesis grading						*
Spam prevention system			*		*	*
Deadlines					*	*
Bookmark thesis						*
Personalized student email responses						*
Ability to hide/show theses						*
Personalized proposals based on student profile						*

Figure 10 - eThesis List of Functionalities (Tsatsaris and Sakkopoulos, 2021)

In addition to the functionalities stated in Figure 10 and similarly to this dissertation's project solution, this LMS also provides an algorithm to filter projects according to their interests, and help decide which projects are the most suitable options (Tsatsaris and Sakkopoulos, 2021). Detailed information about that RS was previously analysed in Section 2.4.1.5.

There is a clear distinction between the functionalities that students and supervisors can take advantage of.

Students can authenticate using traditional credentials or with external services such as LinkedIn, Facebook, Microsoft or Google. After the authentication process, the student can use their RS to filter the most interesting proposals. As soon as the student is assigned to a new project and supervisor, the access to the "MyThesis" section becomes available, which allows the student to add notes to the project and tasks to be completed, keep track of due dates and share files with the supervisor (Tsatsaris and Sakkopoulos, 2021).

In contrast, supervisors can add, edit or delete new theses, as well as attaching tags to them to help in the recommendation process. Besides that, supervisors can add deadlines to certain parts that need to be done in the project, and do most of the student's UCs inside "MyThesis", with the capacity of adding "(…) office hours and delete tasks" (Tsatsaris and Sakkopoulos, 2021).

2.4.2.3 Learning Management System Comparison

A brief objective comparison of all the frameworks analysed in Section 0 is represented in Table 4.

Table 4 – LMS Project Comparison

	SCIPRO	ETHESIS
COMMERCIAL SOLUTION	Yes (6000$/6 month trial)	No
Nº FUNCTIONALITIES (ORDER)	2nd	1st
THESIS MONITORIZATION	Yes	Yes
DEADLINES	Yes	Yes
THESIS ALLOCATION	Yes	Yes (with the help of RS)
Nº THESIS PROCESS FEATURES (ORDER)	1st	2nd

Even though the two solutions analysed in sections 2.4.2.1 and 2.4.2.2 have different functionalities from each other, both tackle the main problem behind LMS in different ways. SciPro is more geared towards the thesis allocation process and monitorization envolving both student and supervisors, while eThesis also incorporates a RS for students to choose their preferred project to work on. Besides that, eThesis offer more general functionalities comparing to SciPro. Examples of that are possibility of statistic, graphic and report visualization, external third-party login, integrated spam prevention system, and others. However, by offering less functionalities, SciPro has a more complex flow behind its LMS (supported by Figure 9), with features such as a complete mechanism for individual supervision, a tool for detecting plagiarism, thesis distribution, and others. SciPro is also available commercially, whereas eThesis is a research solution to a problem raised by researchers in the University of Piraeus.

Although the two applications are robust implementations for a thesis LMS, the dissertation's solution is not required to have the same level of development complexity. With that, it is important to acknowledge that those features will serve as inspiration to come up with the best solution for the dissertation's problem.

3 Value Analysis

In the chapter of value analysis, the value of the dissertation project is evaluated using a different array of mechanisms. Firstly, the innovation process is explained, with more emphasis on the five core principles of the New Concept Development (NCD). After that, the value proposition of the project is represented by its correspondent canvas. Functional analysis is analysed to investigate how the functionalities work with each other. Finally, a multi-criteria decision analysis using Analytic Hierarchy Process (AHP) is implemented to determine which framework and filtering technique to use for the project.

3.1 Innovation Process

Innovation is a term associated with developing something new or a reimplementation of something that already exists but in a different manner in order to add additional value to it. When having the idea to build a new product, it is necessary to think about the predevelopment stages of the product innovation process because it is something that many institutions struggle with (Kim and Wilemon, 2002). The process can be divided in three main parts, which are the fuzzy front-end (FFE), new product development (NPD) and commercialization (Figure 11).

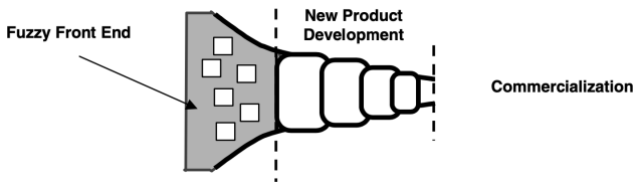


Figure 11 - Innovation Process (Koen *et al.*, 2002)

FFE is a term that represents the meaning of creating a new product concept and questioning if it is viable to invest resources into it. It is known as an experimental phase that serves as a base for the NPD when things need to be much more detailed, formal and precise. Since it is the time for experimentation, even if something goes wrong in the FFE part, the consequences of it are generally insignificant (since the project was never started) (Kim and Wilemon, 2002).

Although FFE is utilized in several companies successfully, there is not a common way of implement it, leading to "(…) difficulty of comparing FFE practices across companies" (Koen *et al.*, 2002). In order to define a common language and terminology to describe factors that were always the same for everyone, Peter A. Koen and his colleagues developed a new model called NCD (Koen *et al.*, 2002). The NCD model divides in three main parts (as suggested in Figure 12):

- Engine: represents organizational principles such as culture, business strategy and leadership, which influences the five core principles.
- Inner Spoke Area: contains the core principles of the defined FFE which are **Opportunity Identification**, **Opportunity Analysis**, **Idea Generation and Enrichment**, **Idea Selection** and **Concept Definition**.
- Influencing Factors: external factors that are possibly not controllable by the organization, such as government policies and laws, competitors, science evolution, and other attributes.

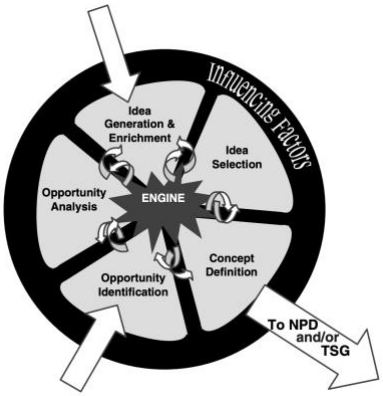


Figure 12 - New Concept Development (Koen *et al.*, 2002)

By exploring each of the main core principles of the NCD model on the dissertation's project, FFE can be followed, which translates smoothly into the development of the project. In the next subsections, NCD's principles are explored with concrete examples.

3.1.1 Opportunity Identification

Opportunity identification is the recognition of certain opportunities that might be interesting for a given institution to pursue. In case of this dissertation project, it would be the creation of an automatic attribution and monitorization tool that can serve as a potential solution for all universities, regardless of their study area.

The project that students integrate on their last semester or year is an important step in their academic and professional careers. As stated in Section 2.1, these types of experiences can leave a very impactful mark on the student's career decision. Hence the extreme importance of choosing and having the best project experience possible.

Another point in the use of the attribution process automatization is the vast number of offers that students have in their disposal. It is overwhelming for a student to search for all the options available in the platform and read their respective descriptions to see if it is a viable project for them.

Lastly, there is no universal standard for project monitoring. Although it is necessary to have a supervisor accompany them throughout the realization of the project, students and supervisors communicate and share files with other third-party programs. It may be unfair for certain people to track their history monitoring process depending on the program(s) they use.

3.1.2 Opportunity Analysis

Opportunity analysis is where the hypothesis raised during the opportunity identification phase is analysed with more depth, in order to assess if it is worth developing a solution for it.

With the vast number of project options the students have at their disposal, a lot of time is spent during the exploration of the problem and description of those proposals. Since there is not any filtering mechanism to limit the number of results shown on the screen depending on the student's preferences, the student is obligated to explore a lot more options, including ones that are not of his interest. Adding that to the analysis of the problem and description of each proposal, the student consumes a lot of time trying to try to find a project of his liking. Besides that, the probability of missing a potentially interesting opportunity among the immensity of results becomes higher the more options the platform has. The platform should be a place that helps the students choose the most compelling project possible, and not make that process even more difficult.

In addition, the deadline for accepting and formalizing the chosen proposal can be very tight, so it is vital to come up with a solution for finding the best possible project for the student with the least amount of time spent analysing proposals.

The final point for analysis is the project monitoring phase. Depending on the preferences of particular supervisors, the communication process between them and their students are held in different formats (such as chat services or email). The problem lies in the number of tools needed to monitor the work developed. For instance, supervisors may want to establish communication with email, with an optional tool like Microsoft Teams or Zoom for online calls, and cloud services for file sharing. For students, that may be a problem because they would need to get used to working with software that they are not familiar with, and they would need to use a different service every time they want to interact with their supervisors. For supervisors, it may be difficult to stay in contact with all the students they are supervising if they utilize traditional services like email or chat services for other purposes (like using those tools in a personal or work context). That may create a disorganized environment that can lead to several consequences including the inability to track the work done efficiently and the notes taken by students and supervisors respectively, and the difficulty and delay in communication with both parties. By having a universal standard for project monitoring, everything regarding the

student's project would be concentrated in one place that is distinguishable from other contexts, which could improve project monitoring quality.

3.1.3 Idea Generation and Enrichment

Idea generation and enrichment is the phase where ideas are worked, developed and matured. Depending on what was discussed in the two previous phases, ideas are "(…) built up, torn down, combined, reshaped, modified, and upgraded" (Koen *et al.*, 2002) in several iterations. Ideas can also be generated or enriched outside the bounds (influencing factors).

Considering the problems and analysis discussed in Sections 3.1.1 and 3.1.2, the ideas that were generated to tackle those points were:

- RS to help students choose their preferred project to work on the final academic semester/year;
- Advanced filtering mechanism to limit proposals shown on the screen;
- Monitoring tool to help supervisors monitor their students in a more efficient manner

3.1.4 Idea Selection

As the name suggests, idea selection consists of choosing the ideas generated in the idea generation and enrichment phase to be pursued. This is usually the most difficult part of the process, due to the fact that according to Peter A. Koen and co., the problem lies on how to choose the best ideas to follow that could give the most business value to the institution (Koen *et al.*, 2002).

Considering that this project was already defined before analysing the solutions for it, the author of this dissertation did not have any input on what the best idea was to tackle the stated problems. Therefore, the ideas to be developed are the RS and monitoring tool.

3.1.5 Concept Definition

The last element of the NCD model is the concept definition, which is also the phase to exit the NPD. In this phase, it is necessary to specify a compelling case for investment.

The purpose of this project is to help students and supervisors to have a better experience in the final project of the semester/year. The final project is extremely important for students since it may interfere with the student's career decision-making process, so students must have access to only the proposals that have the best interest to them. By building a RS that would take into consideration certain input variables, it would select just the projects worth analysing, making it more time efficient. In addition, having a monitoring tool for supervisors to monitor their student's projects would standardize the supervision process, making it more organized,

which would solve problems of missed or delayed communication with students and simplify the report history tracking process.

3.2 Value

Value to the customer was differently defined several times in literature, however, according to Albert Graf and Peter Maas, value to the customer is "(…) considered as a theoretical construct having to do with a customer perspective of provider products or services" (Graf and Maas, 2014). It is also seen as a strategy to build a competitive advantage amongst others, since satisfied customers are more likely to use the product or service again (Sánchez-Fernández and Iniesta-Bonillo, 2007).

Value can be characterized by the perceived value concept, which is how the customer sees value from the product or service that is being offered. Equally to the customer value, the perceived value is a term that has been defined differently multiple occasions, however the methods to obtain the perceived value are divided in two main approaches: one-dimensional and multi-dimensional (Sánchez-Fernández and Iniesta-Bonillo, 2007).

For this project in specific, the main benefits and sacrifices in regard to each one of the stakeholders are specified with more detail in Table 5.

Table 5 - Benefits and Sacrifices

Benefits	Sacrifices
Project recommendations allow to spend less time searching for projects	Time spent answering the questionnaire, which is necessary to have the most accurate answers Recommendations could potentially be not relevant for the student
Greater efficiency and response to all the supervisor's students	Adaptation period to the new universal supervision process

3.3 Value Proposition

Value proposition is the methodology needed to demonstrate the value of a product, service or solution. It is often used in marketing and advertisement segments, and its main purpose is to transmit the value and benefits a given product or service provides to its customers. It can also

explain how the return on the customer's investment occurs when choosing the product over other competitors (Camlek, 2011).

To have a visual representation of the value proposition, Alex Osterwalder proposed the Value Proposition canvas model, that separates the Value Proposition design into two sides: a Customer Profile side that clarifies the customer understanding, and a Value Map side that specifies how the product creates value for the customer (Kyhnau and Nielsen, 2015).

The value proposition canvas model of the dissertation's project is visualized in Figure 13. It describes the problems and features described in Section 3.1, but more geared towards the perspective of customers and the product's value.

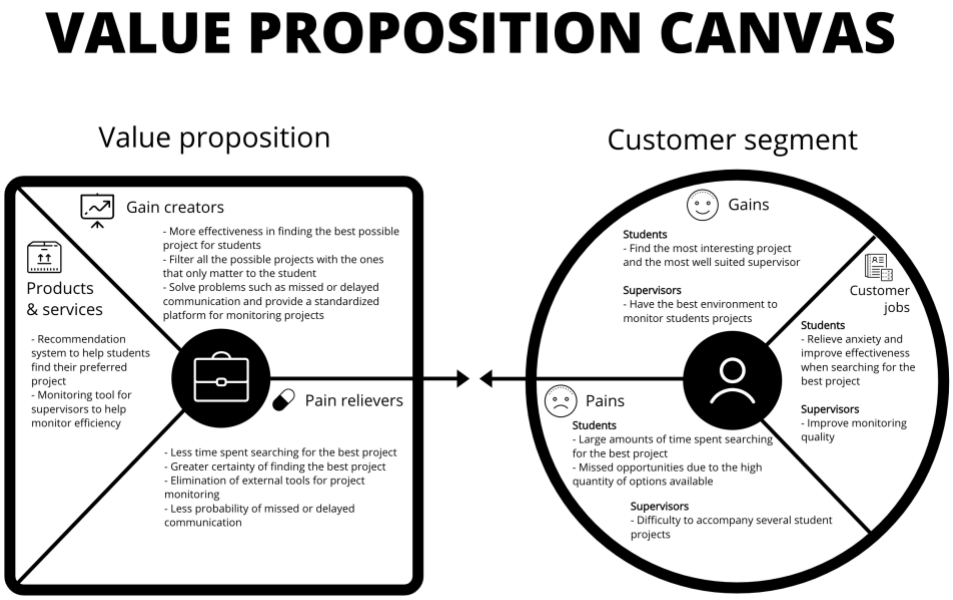


Figure 13 - Value Proposition

3.4 Functional Analysis

Once the ideas for this project are well defined, it is possible to analyse which functionalities the solution is going to have.

The outcome of this dissertation project is the result of two separate products: a RS and a monitoring tool (LMS).

The functionalities associated to the RS are the following:

- Find projects by student preferences;
- Find supervisors depending on the student's preferred projects.

Regarding the monitoring tool (LMS), the functionalities identified are:

- Communicate with students;
- Upload files and share them between stakeholders;
- See report history version.

In Section 3.4.1, a technique called Functional Analysis and System Technique (FAST) is explored in order to exemplify how those functionalities can work with each other.

3.4.1 Functional Analysis and System Technique

FAST is a technique that organizes the functionalities of a product "(…) into a How?/Why? relationship" (Borza, 2011). This methodology can be represented into a diagram, with its horizontal plane representing the **How?/Why?** (How do you do this function/Why do you do this function) relationship, and the vertical plane representing **When?** (when this function is executed, it also executes X function). The word count for function description in a FAST diagram is restricted to two words: Active Verb + Measurable Noun (Borza, 2011). FAST diagram for the project's dissertation is represented in Figure 14.

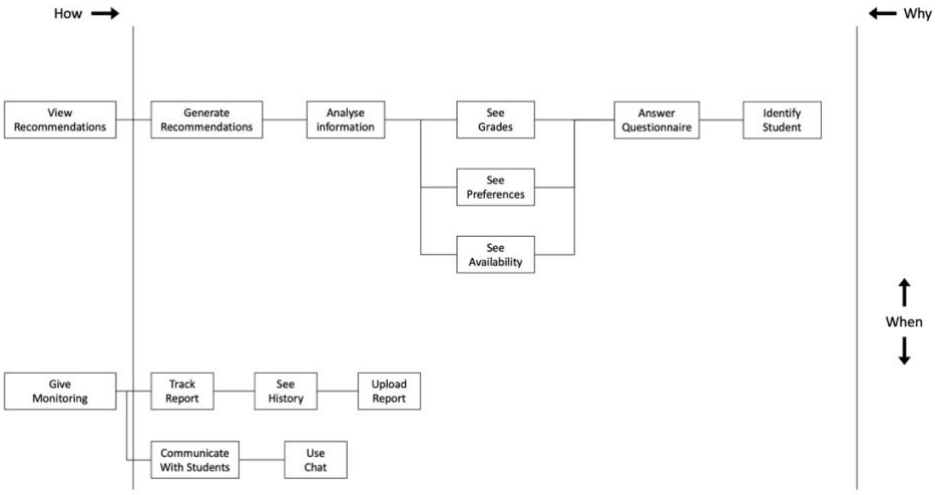


Figure 14 - FAST Diagram

Both systems to be implemented are separated due to not interacting with each other. However, monitoring tool functionalities are below the system recommendation functionalities because the student has to select his project before having supervision.

3.5 Multi-Criteria Decision Analysis

In many projects, such as this one, there are certain aspects that need to be taken into consideration before using them in development and implementation. There are a vast number of choices to choose from, and it is important to choose the best option according to the task

in hand. In this section, the AHP model is analysed by exploring each one of its phases to choose the best RS filtering technique for this dissertation project.

3.5.1 Analytic Hierarchy Process

AHP is a type of multi-criteria decision analysis model that was initially created by professor Thoma L. Saaty in 1980 and its main purpose is to help in the decision making process of complex situations (Saaty, 1988). The classic version of AHP is divided in seven phases:

1. Construction of the hierarchy tree;
2. Comparison of alternatives and criterion;
3. Definition of relative priority of each criterion;
4. Consistency evaluation of relative priorities;
5. Construction of the comparison matrix for each criterion;
6. Get the composite priority for the alternatives;
7. Selection of the alternative.

The classification of each criterion is assigned following a specific scale (Figure 15). Each value has a different definition, and even numbers connect definitions associated to odd numbers.

Intensity of Importance on an Absolute Scale	Definition	Explanation
1	Equal importance.	Two activities contribute equally to the objective.
3	Moderate importance of one over another.	Experience and judgment strongly favor one activity over another.
5	Essential or strong importance.	Experience and judgment strongly favor one activity over another.
7	Very strong importance.	An activity is strongly favored and its dominance demonstrated in practice.
9	Extreme importance.	The evidence favoring one activity over another is of the highest possible order of affirmation.
2,4,6,8	Intermediate values between the two adjacent judgments.	When compromise is needed.
Reciprocals	If activity i has one of the above numbers assigned to it when compared with activity j, then j has the reciprocal value when compared with i.	
Rationals	Ratios arising from the scale.	If consistency were to be forced by obtaining n numerical values to span the matrix.

Figure 15 - Importance Given to Each Criterion (Saaty, 1988)

3.5.1.1 AHP Demonstration

The AHP model is utilized to decide which filtering technique to use in this project. The alternatives that are analysed are the following:

- Content-Based Filtering
- Collaborative Filtering
- Hybrid Filtering

It is important to mention that since hybrid filtering is a technique that combines both content-based filtering and collaborative filtering, a large number of variations for this type of filtering are possible and it is possible to combine several of them in RSs (Kusuma and Musdholifah, 2021; Olasehinde *et al.*, 2022)

The alternatives mentioned previously are compared using the following criteria:

- Data Availability: Since the data available to be analysed by the RS is limited to what is able to get from open-source datasets, some filtering techniques can be more sensitive to low amounts of information;
- Domain Knowledge: For this RS it is important that it can recognize unknown patterns considering the domain and business logic of the project;
- Scalability: This RS is available for all students of Computer Science and Engineering, regardless of the degree they are pursuing. Therefore, it is advisable that the RS is capable of scaling with the increase of users and items, while still having good levels of performance

The first step of the APH model is to build the hierarchy tree with the alternatives and criteria. Figure 16 represents a visual representation of the AHP hierarchy tree.

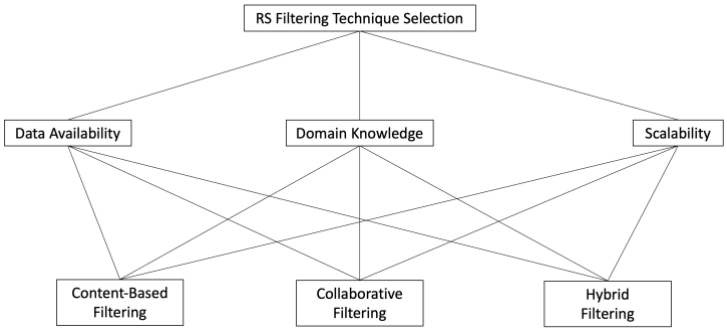


Figure 16 - AHP Hierarchy Tree

The second step of the APH model is to compare the importance of the criteria chosen considering the explored alternatives. The classification of the criteria is determined using the importance values of Figure 15. Table 6 represents the importance of each criterion in comparison to the remaining criteria.

Table 6 - Criteria Comparison Matrix

	Data Availability	Domain Knowledge	Scalability
Data Availability	1	$\frac{1}{5}$	$\frac{1}{3}$
Domain Knowledge	5	1	2
Scalability	3	$\frac{1}{2}$	1

After determining the importance of each criterion, Table 7 shows the same matrix but with normalized values. This matrix is used to determine the relative importance value of each criterion and it is determined by the sum of the elements of each column and the division of each column by that sum.

The normalized matrix is used to determine the priority vector, which analyses the relative importance of each criterion. The priority vector is calculated by the average value of the row of that criterion.

Table 7 - Normalized Criteria Comparison Matrix

	Data Availability	Domain Knowledge	Scalability	Priority Vector
Data Availability	$\frac{1}{9}$	$\frac{2}{17}$	$\frac{1}{10}$	0,11
Domain Knowledge	$\frac{5}{9}$	$\frac{10}{17}$	$\frac{3}{5}$	0,58
Scalability	$\frac{1}{3}$	$\frac{5}{17}$	$\frac{3}{10}$	0,31

The next step of the AHP model is to evaluate the consistency of the relative priorities calculated in the previous phase. For that, it is necessary to calculate the consistency ratio (CR) to determine how consistent the judgements were in comparison to large samples of random judgements. In order to calculate CR, it is first necessary to calculate the consistency index (CI), which is obtained by determining the value of λ_{max}, and determine the random index (RI).

λ_{max} is determined by multiplying the criteria comparison matrix with the priority value to obtain a new matrix, and then dividing it with the priority value. Then, the average of those values divided by the number of criterion minus 1 determines λ_{max}. That process is shown in the following calculations:

$$\begin{bmatrix} 1 & \frac{1}{5} & \frac{1}{3} \\ 5 & 1 & 2 \\ 3 & \frac{1}{2} & 1 \end{bmatrix} \begin{bmatrix} 0{,}11 \\ 0{,}58 \\ 0{,}31 \end{bmatrix} = \begin{bmatrix} 0{,}33 \\ 1{,}75 \\ 0{,}93 \end{bmatrix}$$

$$\begin{bmatrix} 0{,}33 \\ 1{,}75 \\ 0{,}93 \end{bmatrix} / \begin{bmatrix} 0{,}11 \\ 0{,}58 \\ 0{,}31 \end{bmatrix} = \begin{bmatrix} 3 \\ 3{,}02 \\ 3 \end{bmatrix}$$

$$\lambda_{max} = \frac{3 + 3{,}02 + 3}{3} \approx 3{,}01$$

CI is calculated by dividing the subtraction between λ_{max} and the number of criterion, with the number of criterion minus one:

$$CI = \frac{\lambda_{max} - n}{n - 1} \equiv CI = \frac{3{,}01 - 3}{3 - 1} = 0{,}005$$

The other variable to determine is RI. According to Thomas L. Satty, the RI values are originated "(…) simulating random reciprocal matrices of different orders" (Saaty, 1988). Those values are found in Figure 17.

n	2	3	4	5	6	7	8
R.I.	0.00	0.58	0.90	1.12	1.24	1.32	1.41

Figure 17 – Random Index Values

The value to be used in this AHP model is 0.58, since it is RI value that it is associated with matrices with order 3.

With both CI and RI determined, it is possible to calculate CR. CR is determined by the fraction between CI and RI. The value of CR is shown in the following equation:

$$CR = \frac{CI}{RI} \equiv CR = \frac{0{,}005}{0{,}58} \equiv CR = 0{,}009$$

The value of CR for this AHP exemplification is 0,009. Values of CR below 0.1 means that the results are reliable, which means that the results are consistent.

The next step of the AHP model is to compare each alternative to the criteria by building individual matrices for each criterion. The alternatives are compared with each other using the same importance values shown in Figure 15.

The first matrix to be built (and its respective normalized matrix and priority vector) is for the data availability criterion. The importance values given to each alternative are based upon the fact that low amounts of information are presented in the beginning.

Table 8 - Data Availability Comparison Matrix

	Content-Based Filtering	Collaborative Filtering	Hybrid Filtering
Content-Based Filtering	1	7	2
Collaborative Filtering	$\frac{1}{7}$	1	$\frac{1}{3}$
Hybrid Filtering	$\frac{1}{2}$	3	1

Table 9 - Normalized Data Availability Comparison Matrix

	Content-Based Filtering	Collaborative Filtering	Hybrid Filtering	Priority Vector
Content-Based Filtering	$\frac{14}{23}$	$\frac{7}{11}$	$\frac{3}{5}$	0,62
Collaborative Filtering	$\frac{2}{23}$	$\frac{1}{11}$	$\frac{1}{10}$	0,09
Hybrid Filtering	$\frac{7}{23}$	$\frac{3}{11}$	$\frac{3}{10}$	0,29

The second matrix to be built (and its respective normalized matrix and priority vector) is for the domain knowledge criterion. The importance values given to each alternative are based upon the fact that the system needs to be able to predict unknown patterns in the data according to the domain knowledge.

Table 10 - Domain Knowledge Comparison Matrix

	Content-Based Filtering	Collaborative Filtering	Hybrid Filtering
Content-Based Filtering	1	$\frac{1}{5}$	$\frac{1}{7}$
Collaborative Filtering	5	1	$\frac{1}{2}$
Hybrid Filtering	7	2	1

Table 11 – Normalized Domain Knowledge Comparison Matrix

	Content-Based Filtering	Collaborative Filtering	Hybrid Filtering	Priority Vector
Content-Based Filtering	$\frac{1}{13}$	$\frac{1}{16}$	$\frac{2}{23}$	0,08
Collaborative Filtering	$\frac{5}{13}$	$\frac{5}{16}$	$\frac{7}{23}$	0,33
Hybrid Filtering	$\frac{7}{13}$	$\frac{5}{8}$	$\frac{14}{23}$	0,59

The last matrix to be built (and its respective normalized matrix and priority vector) is for the scalability criterion. The importance values given to each alternative are based upon the fact

that the data gathered from the RS grows as more students use the system throughout the years.

Table 12 – Scalability Comparison Matrix

	Content-Based Filtering	Collaborative Filtering	Hybrid Filtering
Content-Based Filtering	1	$\frac{1}{7}$	$\frac{1}{5}$
Collaborative Filtering	7	1	2
Hybrid Filtering	5	$\frac{1}{2}$	1

Table 13 - Normalized Scalability Comparison Matrix

	Content-Based Filtering	Collaborative Filtering	Hybrid Filtering	Priority Vector
Content-Based Filtering	$\frac{1}{13}$	$\frac{2}{23}$	$\frac{1}{16}$	0,08
Collaborative Filtering	$\frac{7}{13}$	$\frac{14}{23}$	$\frac{5}{8}$	0,59
Hybrid Filtering	$\frac{5}{13}$	$\frac{7}{23}$	$\frac{5}{16}$	0,33

With the priority vectors of all alternatives and criteria, the multiplication of those priority vectors determine which filtering technique to choose. A matrix composed of all priority vectors of each alternative is multiplied by the priority vector of the criteria. The calculations are shown in the following equations:

$$\begin{bmatrix} 0{,}62 & 0{,}08 & 0{,}08 \\ 0{,}09 & 0{,}33 & 0{,}59 \\ 0{,}29 & 0{,}59 & 0{,}33 \end{bmatrix} \begin{bmatrix} 0{,}11 \\ 0{,}58 \\ 0{,}31 \end{bmatrix} = \begin{bmatrix} 0{,}14 \\ 0{,}38 \\ 0{,}48 \end{bmatrix}$$

The matrix derived from the multiplication shows that the last value (0,48) is the greatest among the other alternatives. That value is associated to the hybrid filtering, so according to the developed AHP model, it is the most suited alternative for this project.

3.6 Summary

The intrinsic value associated to this dissertation project was described using several methodologies in the chapter of value analysis.

The first tool used was described in the innovation process section with the NCD model by describing its five core principles:

- Opportunity Identification where the opportunities were recognized;
- Opportunity Analysis where the recognized opportunities were further analysed to see their potential in exploring them;
- Idea Generation and Enrichment where ideas were raised regarding the recognized opportunities;
- Idea Selection where the best ideas were selected;
- Concept Definition where a compelling case for investment is stated.

The second tool used was the development of the value proposition of the project using the canvas model, where the value proposition and customer segment were filled considering the information gathered throughout the state of the art, context and innovation process.

The third tool used was the FAST technique that has the main objective of specifying how the several functionalities of the system are organized and work with each other by analysing the how, why and when the user utilizes a certain function of the program. Both systems to be developed (RS and LMS) were represented in the FAST diagram.

The final tool used was the AHP model to determine which filtering technique to implement in the RS. By applying the seven steps of the AHP model, and determine the alternatives and criteria to be used, the filtering technique that is best suited for the RS is the hybrid filtering technique.

4 Analysis and Design

In the chapter of Analysis and Design, the description of the artefacts needed to understand the problem and what needs to be done to produce an automation tool for attribution and project monitoring is stated. The chapter is divided into two main sections: the analysis and the design of the solution. In the analysis section, a requirement analysis is made, with the concepts of domain model, actors, functional and non-functional requirements, and business process being explained. In the design section, the main focus is centred towards evaluating several proposed architectures to structure the solution's code, with visual representations in form of logical and deployment views, and studying the execution sequence of specific UCs.

4.1 Analysis

In this section, the problem is analysed in a programmatic perspective. With that in consideration, the domain model, actors, functional and non-functional requirements, and business process is presented, with written explanations and visual representations.

4.1.1 Domain Model

To understand how the solution works, it is necessary to analyse all the domain entities that intervenes in both systems. The RS produces recommendation objects based on the student's information (such as study area, specialization areas, country, locations, and other information) and the registered projects in the platform. Every project recommended have a suggested list of available supervisors (which are also chosen according to study area, specialization areas, and other information), to give the student the option to choose which one he desires. The student can then show interest in a particular project by making an application, where a recommended supervisor is also associated to. If the supervisor accepts the student's

application, the recommendation is then converted into a project. Once the student starts to work on a certain project, the student and supervisor have the opportunity to communicate with each other via chat by sending messages and share files, including the report. The system accepts multiple uploads of the report, in order to allow for version history. All domain entities were represented by a domain model diagram that can be seen on Figure 18.

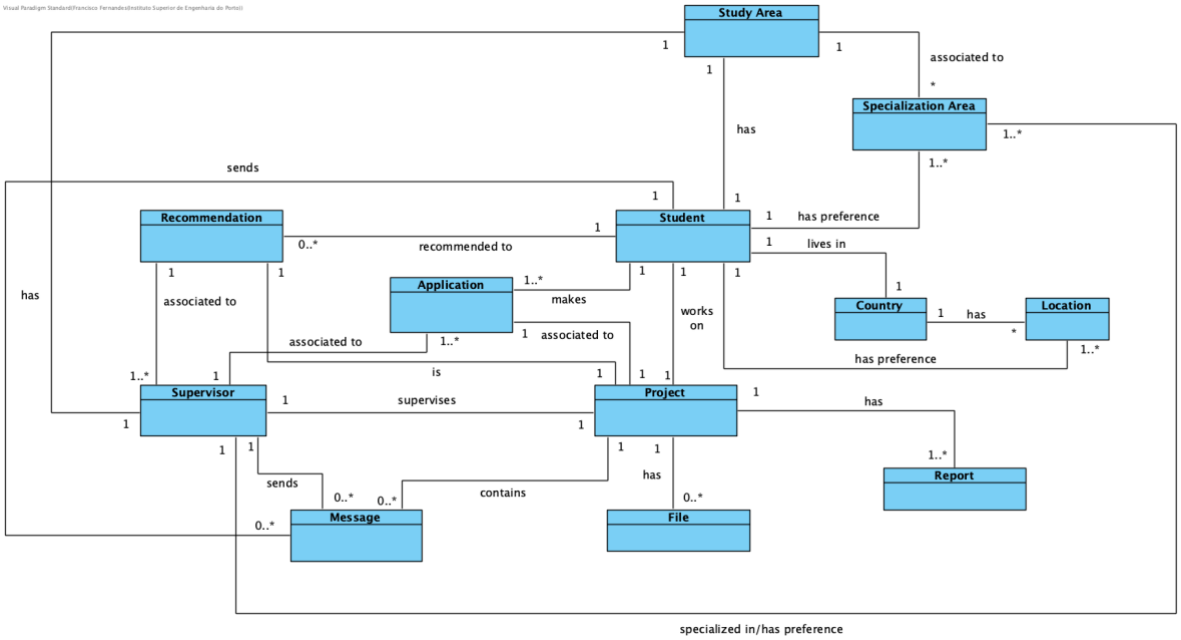


Figure 18 - Domain Model Diagram

4.1.2 Actors

Even though ISEP's project platform has several different actors that are assigned to do several tasks, both systems to be created are designed to be used by students and their respective supervisors (teachers).

Students are the main beneficiaries of the RS's UCs since the main purpose of the RS is to recommend the best possible project for the student. The recommendations are based on the gathered information from the student, which is a key element that RS needs to present the results. The student answers specific questions regarding their personal interests, what themes he would like to explore in the dissertation and past experiences. Then, the answers are collected, as well as other student information such as their academic graduation, their grades, and other useful data. That information is served as input data for the RS, which then gives the best results possible. Students can also utilize several features from the LMS to be implemented. They are able to directly communicate with their supervisor, as well as share important documents with him, such as the project's report, formalization documents and other useful files.

On the other hand, supervisors interact indirectly with the RS, since they are assigned to certain projects by the RS. RS takes into consideration the background of the teacher and availability in

order to assign him to a project. Besides that, supervisors have total access to all of the features of the LMS, to enhance their supervision experience.

Finally, there is also an admin access to the RS's training models and evaluation metrics. The admin can make adjustments to the algorithm according to certain established parameters.

4.1.3 Business Process

The purpose of the dissertation project is to make the process of the final project development more appealing and effective to both students and supervisors. That is the main reason why this project is subdivided into two different systems: one to help the student to choose the best project suited to his interests, and the other to assist supervisors during the thesis-making process and have better communication with their respective students. Although these systems help automate the project development process, they act at different times of the process.

The process starts with choosing the best project. The student starts by initiating the recommendation process, in which he is asked to fill out his interests. If those interests are already registered in the system, an option to skip the additional step is added because the RS's output will be the same (there is no need to generate the same recommendations more than once). If not, the RS generates recommendations, which can be seen by the student. The student has the opportunity to generate other recommendations by changing his interests again.

The moment when the student finds a project and gets accepted by all stakeholders (which is something that is out of the scope of this application but could be represented in the project to serve as a connection between both systems) is when the project monitoring process takes place. A general sequence of interaction between student and supervisor happens when the student uploads his report or files for review. The supervisor is notified and can give annotations to the work done. After answering back, the student then changes those details and after a certain amount of time, they can do the same process again. If that process is no longer being done, it is assumed that the project and the communication between student and supervisor has ended.

A visual representation of the outlined process is presented in Figure 19.

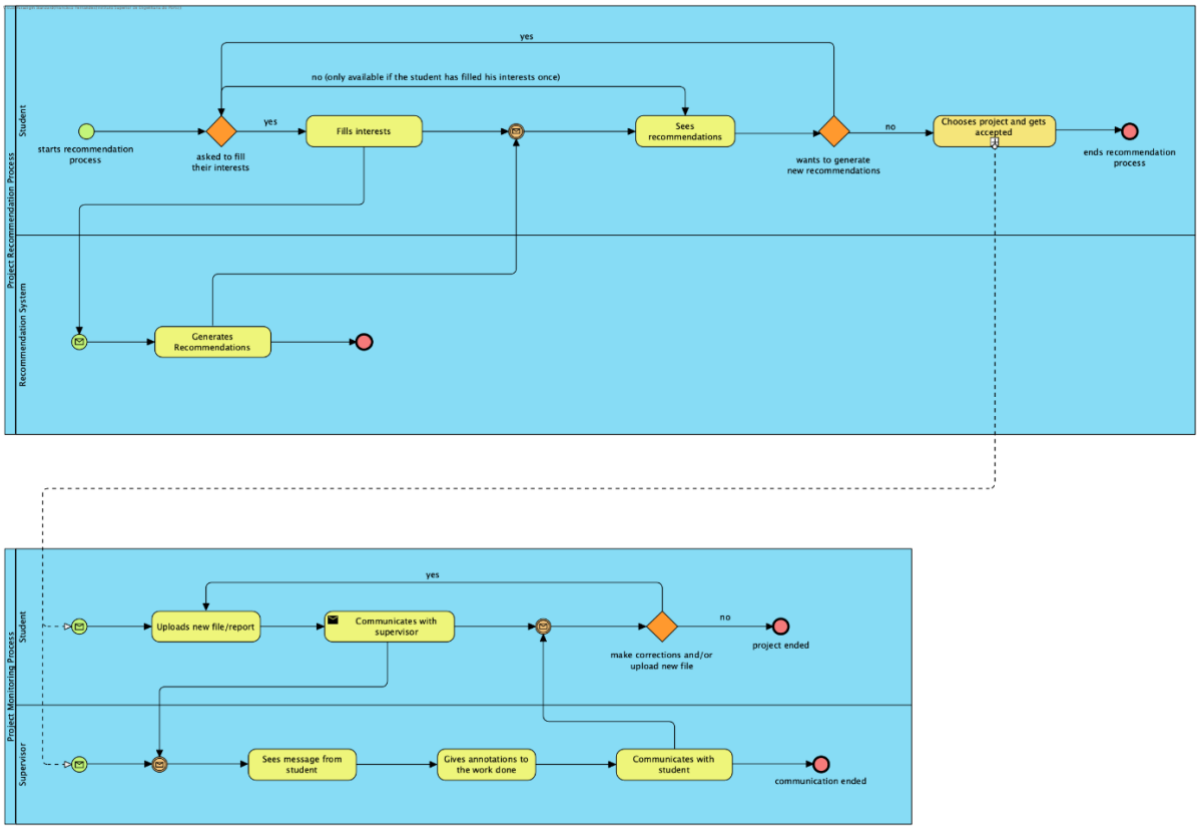


Figure 19 - Business Process Diagram

4.1.4 Functional Requirements

According to the information provided in Sections 4.1.1, 4.1.2 and 4.1.3, it was possible to raise the following UCs described in the UC diagram of Figure 20.

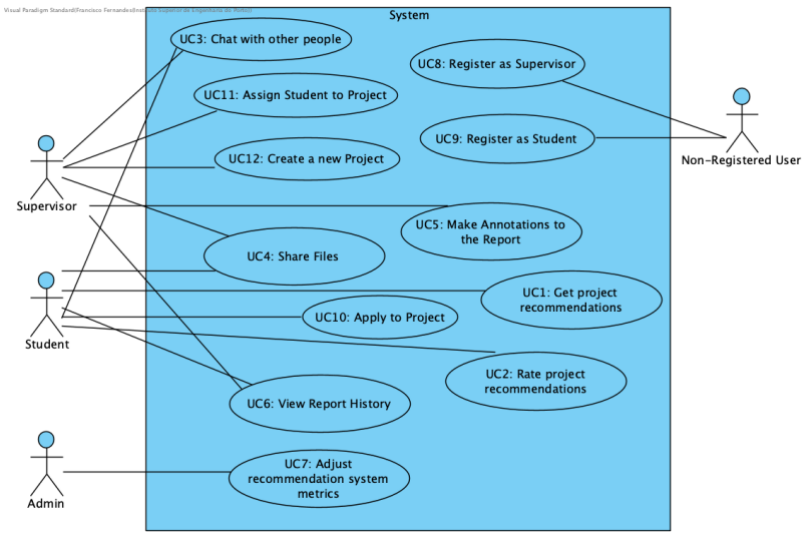


Figure 20 - Use Case Diagram

As seen on Figure 20, the UCs captured are divided by four main actors of the system: student, supervisor, non-registered user and admin. Those actors are all a hierarchy relation with a generic actor/class called User. To interact to the system, non-registered users need to either register as a student or as a supervisor. UC1, UC2, UC7, UC8, UC9, UC10, UC11 and UC12 are associated to the RS service, while the others are related to the LMS.

During the next sections and chapters, only the most important and more complex UCs are explored in more detail. Those UCs are: UC1, UC3 and UC4. Some aspects of the other UCs can be mentioned to give more context. All of the artifacts of those remaining UCs can be found in the Appendix A.

4.1.5 Non-Functional Requirements

In software engineering, non-functional requirements are visualized as restrictions there are imposed and need to be met upon project delivery (Mairiza, Zowghi and Nurmuliani, 2010). Those requirements are usually known as the words that finish with either "-ility" or "-ity" (Chung and do Prado Leite, 2009).

One model that facilitates the visualization of the most important non-functional requirements is Functionality, Usability, Reliability, Performance, and Security (FURPS), created at Hewlett-Packard. A later version of FURPS called FURPS+ was enhanced with more quality attributes (Chung and do Prado Leite, 2009).

In the following subsections, the non-functional requirements for this project are presented and discussed with more detail, according to Lawrence Chung and Julio Cesar Sampaio do Prado Leite (Chung and do Prado Leite, 2009).

4.1.5.1 Security

Security refers to the ability to apply measures and practices to protect inadvertent use of the system, and protect sensible data from unauthorized access, malicious attacks, or potential vulnerabilities.

In order to interact with most of the use cases and functionalities of the system (with the exception of login and register user), the user must be logged in. A token will ensure the user is logged in and it must be attached to every request to be authorized to do that action.

4.1.5.2 Usability

Usability refers to the way the users interact with the system, and the experience interacting with it.

The user interface of those systems needs to be easy to use, by providing a commercially-proven and intuitive design with good user experience. User efficiency when using the platform is of extreme importance due to being one of the pivotal parts of the project. System users must be able to interact with all UCs without having the need to ask for help and spend unnecessary

time to find or understand the functionality. This non-functional requirement will be evaluated through feedback and questionnaire to each of the testing users.

4.1.5.3 Reliability

Reliability refers to the ability of a software product to perform its intended functions consistently and accurately without failing.

If there is a problem with one of the systems, self-explanatory and clear errors need to be provided to the users, as well as solutions to resolve them. In the case of system failure or unavailability of one of the services, the remaining UCs need to keep work as expected due to the microservice approach.

4.1.5.4 Performance

Performance refers to the speed, efficiency and response time of the system.

There is no implicit restriction of performance for this system, however, there are minimum acceptable thresholds to be met to enhance user experience. For that reason, the RS must ensure that it produces the best results possible without affecting performance extensively. Besides that, LMS features must also be responsive and not disturb user experience. Performance tests will be used to measure request speed of those functionalities.

4.1.5.5 Supportability

Supportability refers to the ability of maintaining, updating and integrating the system with other externa sources, making it more scalable.

Maintainability of the application is other important aspect to take into consideration. Both systems need to be developed with code quality standards, implying that a testing policy (with unit and system tests) and best practices of code development are respected. The system should also allow additional functionalities to be implemented in the future.

4.2 Design

In this section, different alternatives of possible architectures for this dissertation's project are explored, as well as shown visual representations of those architectures in form of logical and deployment diagrams. Besides that, the referred UCs in Section 4.1 are detailed in text and sequence diagram formats.

4.2.1 Architecture

During the analysis described in Section 4.1, it was possible to gather that two main systems need to be designed from the ground up: the RS and the LMS. Since both tackle different domain problems and UCs and can work independently from each other, the idea of separating

both systems into individual systems is relevant. By working as independent systems, they would also have access to independent databases to store and get important information.

During this chapter, architectures are studied in a variety of different diagram formats, including logical and deployment diagrams.

4.2.1.1 Logical View

To decide the best architecture to adopt in this project, several architectures are proposed through component diagrams.

The first two architecture proposals represent the scenario of integrating the systems within an external project platform, like ISEP's project platform. The first one is represented in Figure 21 in form of a level 1 component view diagram. The main components identified from the analysis (and that are used in all the architecture proposals) are the following:

- **ISEP Platform Front-End**: Component that interacts with all stakeholders;
- **ISEP Platform Back-End**: Component that has all the functionalities that ISEP's project platform offers;
- **ISEP Platform DB**: Component that represents the database of ISEP's project platform
- **Automation Gateway**: Component that redirects the request to registered APIs depending on what the user wants to do;
- **Authentication**: Component that is responsible to handle the authentication process of the system;
- **Authentication DB**: Component that represents the database of the authentication service. It stores the data of all users registered in the system;
- **Recommender System**: Component that represents the RS, that makes recommendations based on the student's information and currently available projects;
- **Recommender DB**: Component that represents the database of the recommender system. It stores the recommendations of the students (in order to not run the recommendation algorithm for the same inputs), and other student information;
- **LMS**: Component that has all the UCs regarding the monitoring of projects;
- **LMS DB**: Component that represents the database of the LMS. It stores relevant information of the LMS, including reports and useful files that were shared between student and supervisor, and other project information;
- **Broker**: Component that retrieves relevant information from ISEP's platform database for both standalone APIs;
- **Consumer**: Component that listens to changes in the broker's component and update the databases according to the information provided.

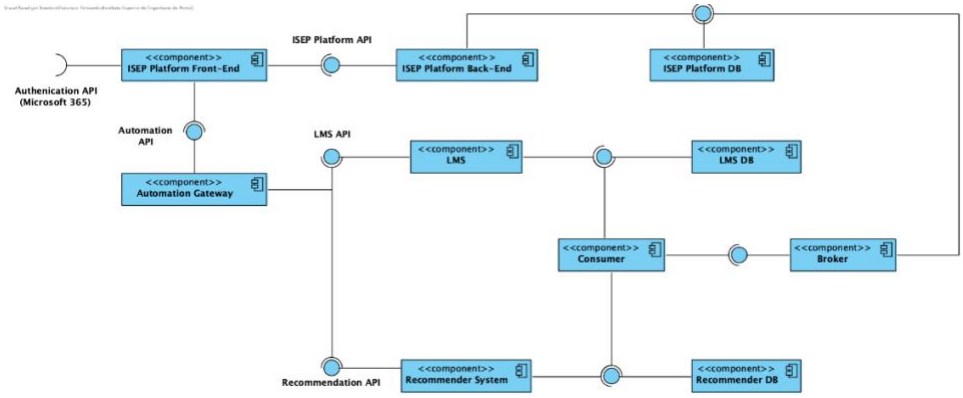


Figure 21 – Component Diagram 1

The current structure of ISEP's project platform is unknown to the author, however it is assumed that it is based on three main components: front-end, back-end and database. The authentication process is dealt by Microsoft 365, which retrieves an access token that serves as authentication proof for using the system.

To communicate with the two standalone API's, an API gateway is created to control which API to call depending on the request (Automation Gateway). Even though both APIs consume their own databases, some information from the platform is needed. The concept illustrated has a broker component that retrieves information from the database, and the consumer component then updates the databases with that data. This method of message broker allows to separate both APIs while still having the most up-to-date information from ISEP's database. However, this architecture has increased complexity, and requires more deployment structures.

The other architecture proposal to integrate with ISEP's project platform is to make the LMS and RS APIs interact directly with ISEP's back-end. The need for an automation gateway and message broker would disappear since all requests would be now processed by ISEP's backend first. In addition, the backend would have direct access to the information needed before calling one of those systems. This architecture is simpler to develop, however, it would come at the cost of high coupling. The architecture proposal can be seen in Figure 22.

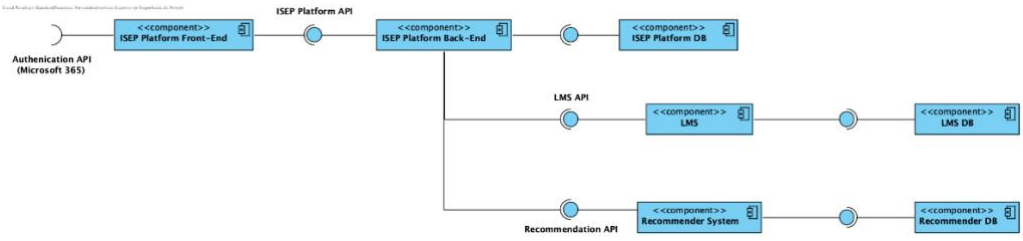


Figure 22 – Component Diagram 2

Since the main purpose of the dissertation's application is to work for every study area and university, those architecture proposals are discarded.

The other project architecture to be explored is the one that is represented in Figure 23. Contrary to the ones explored previously, this project architecture works independently from

other external platforms. By not having the information from other university platforms, the data necessary for the RS to work (on a testing environment) is retrieved from external datasets. Even though it will most likely not have the most accurate information about students and projects, it can serve as a testing example before integrating the solution with real information.

Since the application run independently, it is also necessary to build a front-end infrastructure for the users to interact with the system. The front-end will communicate with an API gateway (similarly to what was described in Figure 21) that redirects the requests to the correspondent system.

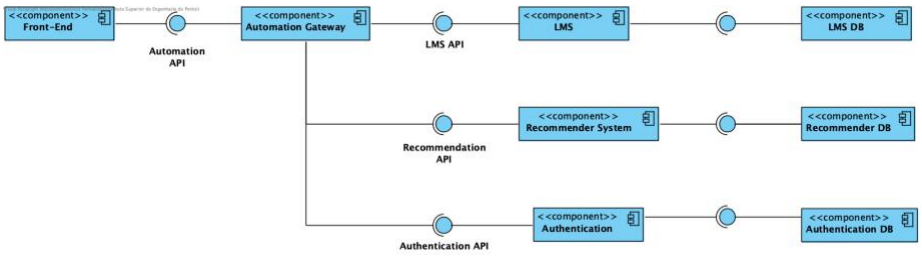


Figure 23 – Component Diagram 3

Because the systems need to work independently, the proposed architecture to be used is architecture 3. Besides that, since that architecture is based on a microservice approach, it will allow for scalability on each service if necessary.

Taking a closer look at the recommender system and LMS (represented in Figure 24 and Figure 25 respectively), the architecture developed is the controller, service, repository architecture. Each module is organized by entity, which means that depending on the module, all of the UCs and/or domain logic associated to a given entity is located in the same file. Therefore, this type of architecture simplifies the way of developing an API. The definition of those modules are the following:

- **Controller**: Module that serves as an entry point to answer outside requests. Depending on the functionality, the respective controller file calls its correspondent service file to execute the functionality asked;
- **Service**: Module that has all the business logic regarding the UCs to be performed. It is in this module that requests to other services or external services are made, as well as requests to the database via calling the correspondent repository file;
- **Repository**: Module that communicates with the database to retrieve specific information for the service module;
- **Model**: Module that represents all the domain entities identified in the domain model in Section 4.1.1;
- **Data Transfer Object (DTO)**: Module that represents the objects that are received from the request and returned to the response. These objects will be transformed into model entities during the use case execution;
- **Recommender Engine** (only applied to Recommender System): Module that represents all the logic regarding the RS model.

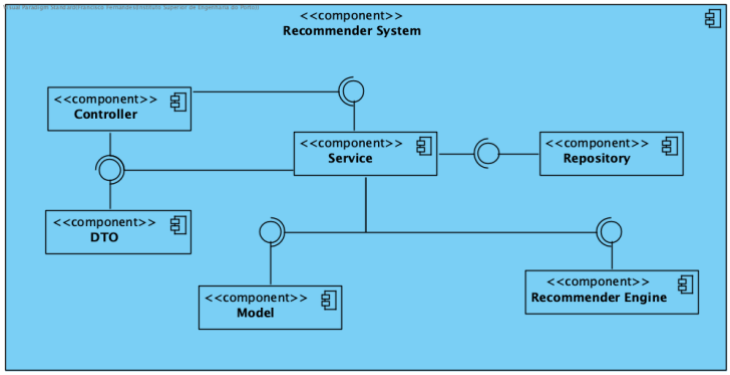


Figure 24 - Recommender System Logical View Architecture (Level 2)

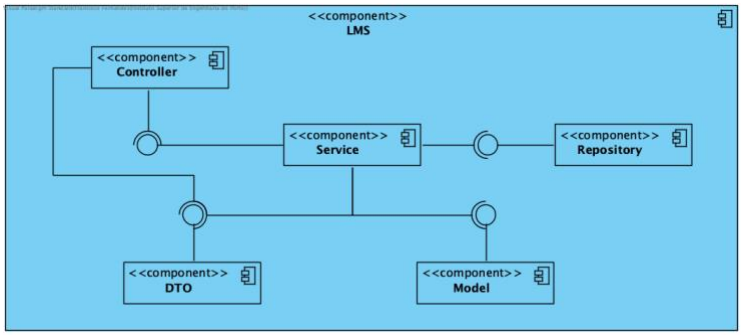


Figure 25 - LMS Logical View Architecture (Level 2)

4.2.1.2 Deployment View

According to Section 4.2.1.1, the deployment process is based on architecture 3. The deployment diagram is seen in Figure 26.

The diagram represented in Figure 26 is based on a microservice architecture, although the need for more complex concepts such as service discovery and orchestration are not needed since the system does not need to be scalable (however it allows for that option if necessary in the future). The decision to use this deployment organization is based on separating domain concept, more organization and more maintainability, since it would be easy to swap or add more APIs to this logic. Servers can communicate with each other by using asynchronous communications powered by RabbitMQ (by utilizing the Advanced Messaging Queue Protocol (AMQP)).

The automation gateway that serves as an API gateway is executed in its own dedicated Linux server. It communicates via Hypertext Transfer Protocol Secure (HTTPS) with the other APIs by redirecting the request made by the platform's front-end.

Each API to be implemented is deployed in its own dedicated Linux server and communicates with the API gateway via HTTPS.

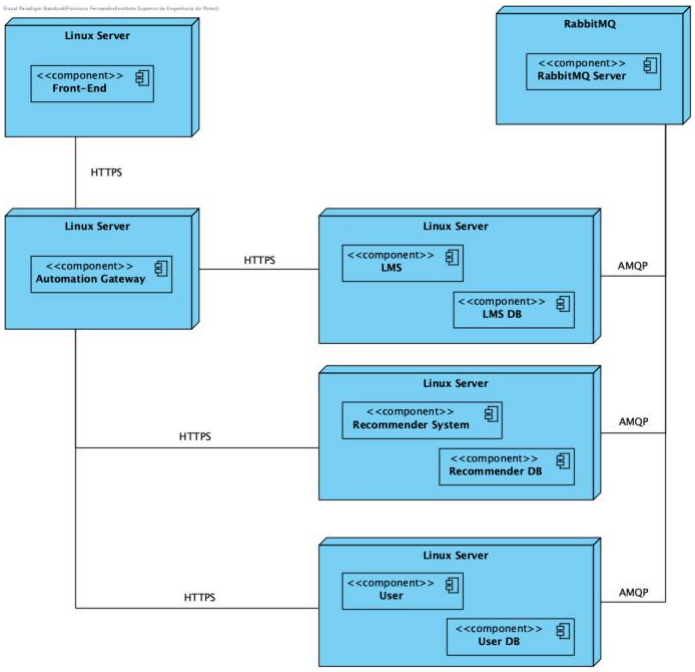


Figure 26 - Deployment Diagram

4.2.2 Use Cases

In this section, an explanation of the execution sequence of the stated UCs in Section 4.1.4 is presented, as well as the corresponding sequence diagrams as visual representations.

4.2.2.1 Get Project Recommendations

The UC starts with the student wanting to see his project recommendations, that takes into consideration his information and his interests.

For every request made (regardless of the UC), it is necessary to verify if the user is authenticated in the system. It is then necessary to attach the authentication token in the request's header section. The request is made to the automation gateway, which acts as an API gateway that redirects the request to the corresponding API.

The process starts by checking if there are any recommendations already made for the student. `findRecommendations(studentId)` only returns recommendations if it finds records registered in the database that match the student's information, otherwise it returns `null`. This methodology allows for better resource use in generating recommendations for the same inputs.

If no records were found in the database, the service calls the recommender engine to generate recommendations for the student. In order to generate recommendations, the recommender engine needs the student info and his recommendation description (if applicable). If a recommendation description is provided, the algorithm will apply NLP on top of the recommendations to sort them accordingly. After generating the recommendations, they are

saved in the database and any other recommendations found are override. Finally, the recommendations are sent back to the front-end, which are displayed to the student.

A visual representation of this UC can be in Figure 27.

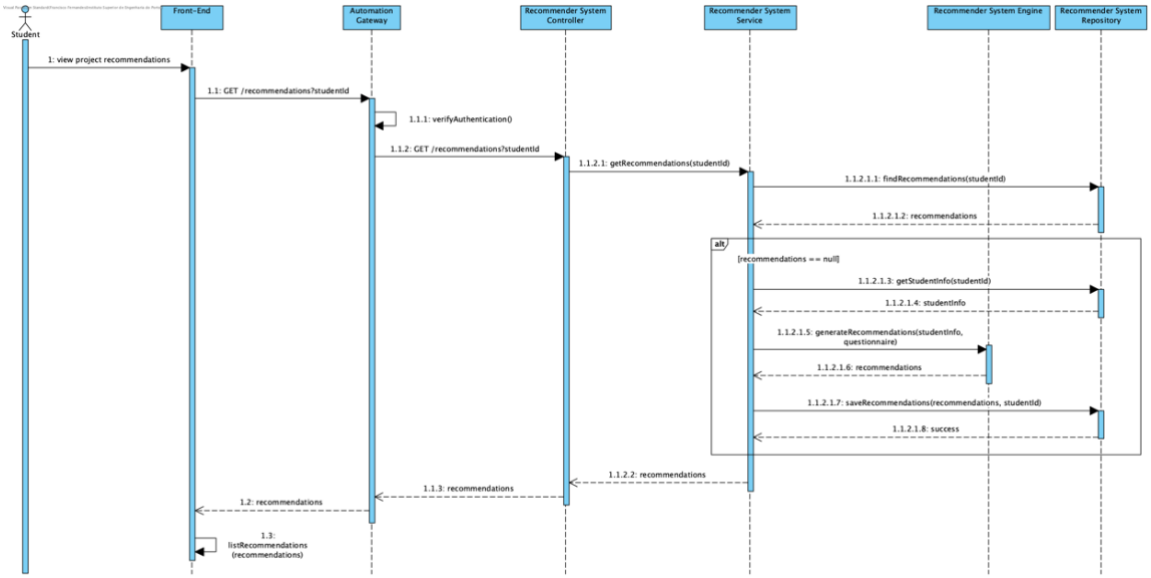


Figure 27 - Get Recommendations (UC1) Diagram

4.2.2.2 Chat With Other People

It is only possible to start this UC if a student has been assigned to a project, and it has already a supervisor attached to it, since this UC belongs to the LMS API.

This process starts by either the student or supervisor wanting to use the chat service to communicate with the counterpart. If that is not the first time both the student and supervisor communicate with each other using that service, it is necessary to retrieve the chat history between both parties. For that, a request is made to the automation gateway to retrieve that information to the platform's front-end component. The automation gateway redirects that request to the LMS API, which calls the respective service file that has the domain logic necessary to get the chat history between student and supervisor. It then calls the respective repository file to get the chat history from the database. That list of messages is retrieved to the front-end and then displayed to the user.

A visual representation of that explanation in form of a sequence diagram is found in Figure 28.

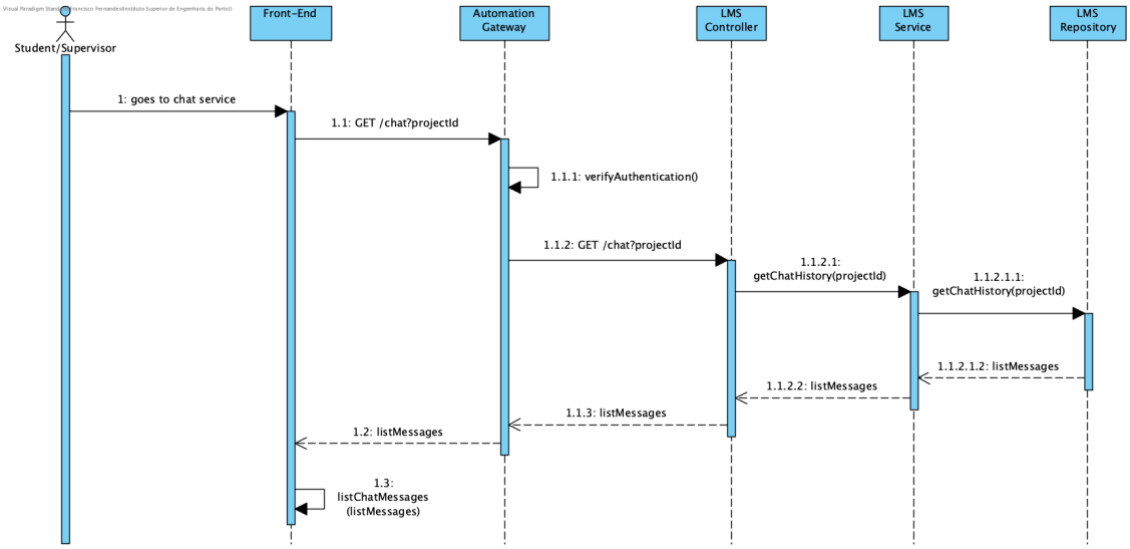


Figure 28 - Chat With Others (UC3) – Chat History Diagram

The process of getting the chat history of two people is relatively straightforward. However, the chat service needs to also allow to send and receive messages. The process of sending and receiving messages was separated in different sequence diagrams (Figure 29 and Figure 30 respectively).

If the student or supervisor wants to send a message to its respective counterpart, a request is sent with project, sender and receiver IDs as query parameters and the message in the body of the request. The automation gateway redirects the request to the LMS API, which then calls the respective service file to save the message in the database and notify the receiver that a new message was sent by the sender.

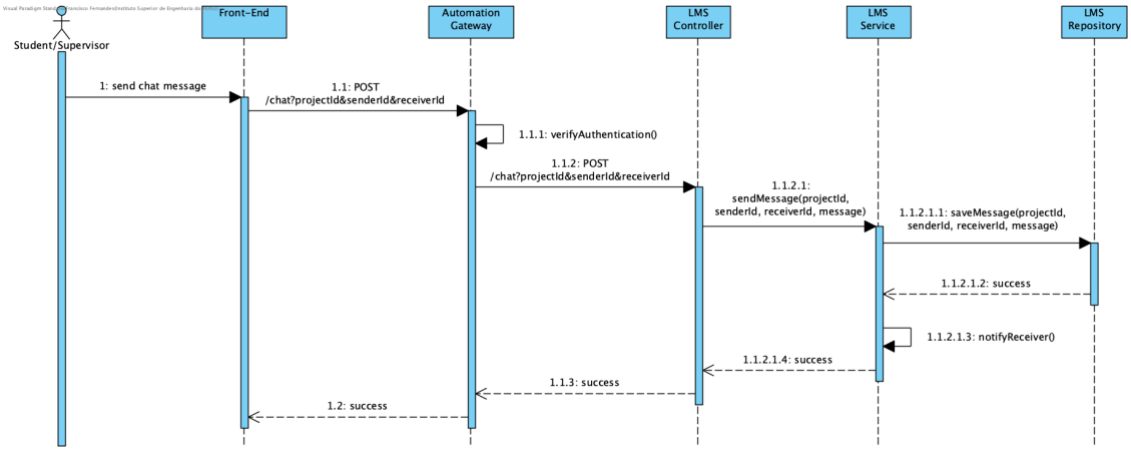


Figure 29 - Chat With Others (UC3) – Send Message Diagram

In order to receive messages in real-time, and get notifications of new messages received, an event is thrown when sending new messages, and it is then caught by an event listener. The purpose of that event listener is to receive the message sent by the event and redirect them to

the front-end component. If the user is using the chat service at that time, the front-end immediately displays the message sent.

However, there are cases where the user is not using the chat service at that precise moment or is not authenticated in the platform at all. That is why it is important to have another mechanism of notifying the receiver. Therefore, every time a message is sent, a notification email is sent to the receiver user, increasing the chances of a quicker response.

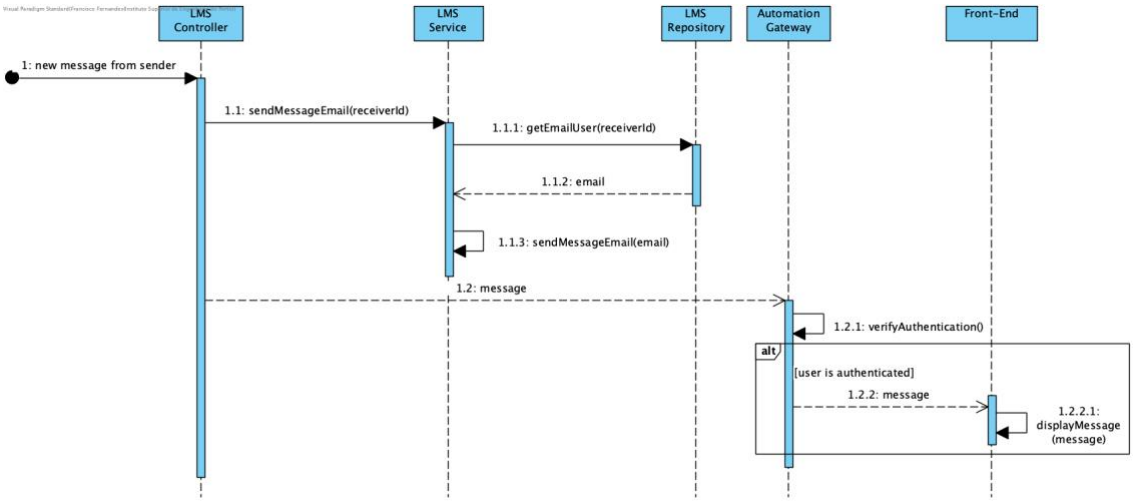


Figure 30 – Chat With Others (UC3) – Get Message Diagram

4.2.2.3 Share Files

Similarly to the UC “Chat With Others”, it is only possible to start this UC if a student has been assigned to a project, and it has already a supervisor attached to it.

The process starts when either the student or supervisor wants to upload a file (e.g. the project’s report or the formalization document) and share it with the other person. The request has attached the project and sender IDs, as well as the file in the request’s body. Since a file is sent by REST, it is mandatory to use a multipart Hypertext Transfer Protocol (HTTP) request. One form of multipart request is the multipart/form-data request, which consists of a specific request when a form is filled with “(…) information that is typed, generated by user input, or included from files that the user has selected” (Masinter, 2015).

The request is made to the automation gateway, which redirects the request to the LMS API. The controller calls the correspondent service to upload the file to the desired project. The file is saved, as well as the person who sent it. A success message is going to be returned if everything went well, and the file appears in the front-end. After that, both the student and the supervisor are able to see or download the file on whichever device they are accessing it. This behaviour is intended to be similar to what a cloud service provides.

A visual representation of that explanation in form of a sequence diagram is found in Figure 31.

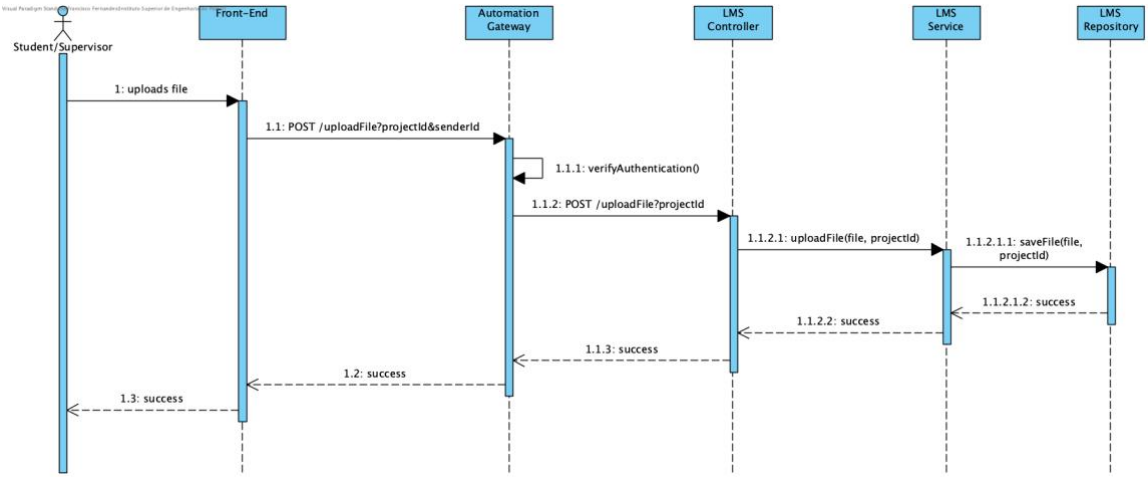


Figure 31 – Share Files (UC4) Diagram

4.3 Summary

The analysis and design chapter is subdivided into two main parts: the problem analysis and the solution design.

The analysis starts by gathering all domain entities that could be found by previous context and investigation of the problem and the respective objectives. The system has recommendations which are projects that are assigned to students and supervisors. Once both student and supervisor have a project, they can communicate with messages and share files with each other for review or guidance. The actors identified by the system are students and supervisors. Then, it is explored how the RS and LMS connect with each other in the final project's development process. Since they are separate systems, the RS is used in the beginning of the process, while the LMS is only being utilized when students and supervisors are attached to a specific project. The analysis ends with functional requirement identification with the context of previous sections in mind, as well as non-functional requirements with the help of the FURPS+ model.

The design section starts with an explanation of possible architectures for this project. Firstly, a architecture which tries to isolate the standalone systems as much as possible from the ISEP's platform is presented, in which is created a separate API gateway to redirect traffic and a consumer and provider technique to gather all the information necessary from ISEP's database and transfer over to the respective databases. However, since this project needs to function without external sources, another architecture was proposed that uses datasets as the main information source for the RS. That architecture was chosen in comparison to the other two that are more coupled to ISEP's platform. A deployment diagram using the chosen architecture was also represented to understand how the system can be deployed in production. Lastly, the most relevant UCs pointed out in the analysis section were explored with further detail, using sequence diagrams as visual representations. The UCs discussed in the design section were: UC1 – Get Project Recommendations, UC3 – Chat With Other People and UC4 – Share Files.

5 Implementation

In the chapter of Implementation, the process behind the project's dissertation is documented by analysing the back-end and the front-end separately. The back-end is explained by analysing the designed architecture and common traits amongst all services. The three services are also explained separately: authentication, recommendation and LMS. In each one of them, important files and implementations (in the case of the RS) are detailed, as well as all the endpoints and their respective functions. The front-end is also explained on the same manner, by analysing the important files of SvelteKit, folder structure and relevant implementations of third-party libraries. Screenshots of the front-end can be found inside the Appendix B. The last part of the chapter analyses the production deployment of the application.

5.1 System Architecture – Back-end

According to Figure 26, the system is supposed to be divided into a microservice architecture, in which the RS, the LMS, the authentication and the API gateway are allocated in different standalone services.

With that into consideration, four separate projects were created with Python, which is the programming language analysed in Section 2.2.6.1. Each project was built with the use of a web framework called Flask, one of the most popular frameworks for Python that gives the opportunity for developers to build full stack web applications (Ghimire, 2020). However in this case, Flask was only use as a backend framework.

Besides respecting the patterns and module structure presented in Section 4.2.1, each project have common Python files that perform specific functions:

- **`app.py`:** Each service is executed by running this file (`python app.py`). All of the service endpoints are defined, which are linked to the correspondent controller. In addition,

services and repositories are instantiated to allow for dependency injection, which minimizes dependencies between classes (coupling between objects) (Sun *et al.*, 2022);

- **`bootstrap.py`:** In a development environment, databases are filled up with the testing datasets that contains information for important entities (e.g. students, supervisors, specialization areas, study areas, etc.);
- **`init_db.py`:** Contains the information necessary to setup the database from scratch (or erase a previously built one);
- **`config.py`:** When `app.py` is executed, this file is responsible to initialize the Flask application with all the configuration variables defined in the environment file and setup the properties and relationships between entities in the database. It export important objects to be used throughout the application, such as the Flask application object, and the Object Relational Mapping (ORM) object to interact with the database.

Communication between services is essential, as some services need to exchange information with each other. Each service stores its own information in a separate PostgreSQL database. Whenever some service needs to communicate with another one, a message is sent to a queue and it will be received asynchronously by the other service that is subscribed to it. The payload of those messages is information about a certain entity. There are only two asynchronous communications between services:

- When a user is created, the authentication microservice sends the user information (excluding the password) to the recommendation microservice to store that information in the database and allow the user to complete the registration as student or supervisor;
- When a supervisor accepts an application from the student, the recommendation microservice sends the information about that project recommendation to the LMS microservice to transform that into a project, and allow for both actors to interact with the project.

The recommendation and LMS services have both a file called `auth-middleware.py` that is responsible for verifying if the user requesting is authenticated and has the proper roles to perform the desired UC.

Having discussed the common points that all of the services share, the following subsections will be dedicated to exploring each project with more detail.

5.1.1 Authentication Service

This service allocates all the logic behind the authentication process. In this case, it is responsible for logging in users in the system, as well as registering new users. As identified in the use case diagram represented in Figure 20, three actors were identified: students, supervisors and an admin. Even though all of them can authenticate in the application, it is only possible to register new students or supervisors.

When one of those actions are performed, a JSON Web Token (JWT) is returned and it is mandatory to be attached to all future requests to the system. That token serves as a method to know if the user is authenticated, and contains basic information, such as the id and role of the user.

The database of this service only saves the minimum required information to function properly. Therefore, when registering a new user (whether it is a student or a supervisor), the authentication service only saves the name, email, password and role of the user, and communicates that information to the recommendation service to store the newly created user as a student or supervisor depending on the chosen role.

5.1.1.1 Endpoints

The endpoints this service exposes are the following:

- **Authenticate in the system** (`POST /login`): Responsible for signing in a registered user. Returns a JWT if the user is registered in the system and gave the right credentials, otherwise gives the appropriate error;
- **Register in the system** (`POST /register`): Responsible for registering a new student or supervisor, and communicate with the recommendation service about the new user. Returns a JWT if the user introduces the correct information (name, email, password, and the desired role), otherwise gives the appropriate error.

5.1.2 Recommendation Service

This main function of this service is to recommend projects and the most suited supervisors to a given student. However, it has also other functionalities that are worth mentioning, such as the ability to get all the main entities of the domain model (useful when showing that data in the front-end), update information about the student or supervisor, register a new project and manage applications of students.

As mentioned in Section 5.1.1, when a new user is registered, the authentication service communicates with this one to register the user information (with the exception of the password) into this service's database. However, the registration of a student or supervisor is not completed after that, since it is necessary to supply the remaining information (e.g.: study areas, specialization areas, locations, countries, and other information) in another specific endpoint (`POST /supervisor` or `POST /student`). As soon as the information is filled, the actor entity is created and persisted to the database. From that moment, the user can interact freely with all the remaining endpoints.

In the following subsections, it will be explained the thought process behind the creation of the dataset to test the RS, as well as the RS algorithms and strategies adopted.

5.1.2.1 Data Mining

Since this problem is specific, it becomes challenging to find complete compatible datasets that can be used to train the machine learning models. The approach used to create and evaluate the dataset and machine learning models was to adopt the CRISP-DM strategy described in Section 2.2.1.

The first step was to identify which entities needed to be considered into the machine learning model. According to the Domain Model diagram in Figure 18, those entities are Recommendations, Students, Supervisors, Projects, Specialization Areas, Study Areas, Locations and Countries. After that, it was necessary to gather information about each of those entities. Some entities' information are more easily attainable than others, since there are general datasets that can be converted with minimal data processing. Examples of that are the Locations and Countries, which were retrieved from a trustable source(1). Other entities like specialization areas and study areas were collected using the ChatGPT application, and contain answers of sufficient quality for testing purposes, even though they may not be the most accurate ones. Since projects need to be as accurate as possible, they were retrieved from ISEP's current project platform.

All of the data was compiled and then organized in a Excel file, with one spreadsheet per entity (example shown in Figure 32). Each column represents a property of each entity, and each row represents an object. Some of the columns represent the IDs of another entity instead of a name, because it is easier to import the datasets (in the *bootstrap.py* file) that way. In the example of Figure 32, the IDs were randomly selected with Excel's functions to simulate real student's preferences and information. All entities were organized in this fashion, with the exception of the Recommendation's entity. For that specific entity, a R script was written to generate a CSV file capable of generating possible recommendations based on all the previous datasets (Appendix B). For every student, it generates the best five project recommendations based on their preferred specialization areas, locations, project type and academic degree. Recommendations are obtained by accessing which ones have the highest reliability percentage. Each parameter of the percentage has different weights and the value is obtained by the following equation:

$$reliabilityPercentage = 0{,}15 * m_1 + 0{,}15 * m_2 + 0{,}7 * \frac{m_3}{m_4}$$

where m_1 is equal to 1 if the student preferred project type coincides with the project type, m_2 is equal to 1 if one of the student's preferred locations are inside a range of 20km of the project location, and the split between m_3 and m_4 corresponds to how much student's preferred specialization areas are contained in the project's specialization areas.

Besides that, for every project recommendation, it recommends the best 5 supervisors that are available to supervise according to their initial and final dates, specialization areas and academic degree.

1 - https://simplemaps.com/data/world-cities

	A	B	C	D	E	F	G	H
1	id	qualification	study_area_id	preffered_areas	country	preffered_locations	preffered_project_type	user_id
2	1	masters	1	3, 1, 6, 8, 17	175	20, 86, 88	internship	1
3	2	bachelor	1	5, 9, 23, 2, 11	175	157, 52, 121	internship	2
4	3	bachelor	1	8, 16, 1, 7, 20	175	204, 109, 131	research	3
5	4	bachelor	1	16, 1, 7	175	73, 113, 69, 112	research	4
6	5	masters	1	3, 8, 10, 20, 23	175	59, 64, 231, 163, 44	internship	5
7	6	bachelor	1	18, 1, 13, 23	175	220, 76	project	6
8	7	masters	1	10, 23, 20, 9, 12	175	176, 84	research	7
9	8	masters	1	18, 16	175	171, 77, 79	research	8
10	9	bachelor	1	6, 11, 17, 13, 16	175	14, 209, 129, 140	internship	9
11	10	masters	1	12, 22	175	215, 131, 44, 97, 28	project	10
12	11	bachelor	1	19, 2, 3, 22, 23	175	159, 31	internship	11
13	12	bachelor	1	3, 14, 5	175	64, 100, 154, 94	research	12
14	13	masters	1	7, 17	175	39, 27, 88, 35, 140	project	13
15	14	phd	1	10, 23, 3, 12, 22	175	220	research	14
16	15	phd	1	18, 1, 13	175	29	internship	15
17	16	bachelor	1	12, 20	175	132, 14	project	16
18	17	phd	1	15, 7, 17, 11	175	154, 55, 133, 34	project	17
19	18	masters	1	2, 20, 10, 19	175	58, 62, 203, 146, 163	project	18
20	19	masters	1	22, 8	175	38	internship	19
21	20	bachelor	1	21, 17, 7, 12, 11	175	111	internship	20
22	21	masters	1	1, 17, 7, 19	175	232, 38, 27, 23	project	21
23	22	masters	1	15, 11, 2, 4	175	34, 216	internship	22
24	23	bachelor	1	8, 5, 15, 7	175	89	research	23
25	24	masters	1	19, 6	175	121, 63, 135, 23	project	24
26	25	masters	1	9, 3, 22, 15	175	51, 104, 144, 134, 174	project	25
27	26	masters	1	21, 14, 8	175	193, 37, 106, 132	project	26
28	27	masters	1	22, 6, 23, 21	175	163, 105, 145, 52	research	27
29	28	bachelor	1	5, 7, 21	175	136, 231	research	28
30	29	bachelor	1	15, 18, 14, 13, 23	175	86, 81, 37, 217, 190	project	29
31	30	bachelor	1	8, 18, 13, 11, 15	175	147, 2, 14, 28, 94	project	30
32	31	masters	1	22, 11, 1, 4, 14	175	123, 146, 106, 96	research	31
33	32	bachelor	1	1, 23, 20, 6	175	154, 220	project	32
34	33	masters	1	16, 2, 21	175	103, 8, 193, 225	project	33
35	34	bachelor	1	20, 9	175	172	research	34
36	35	masters	1	5, 8, 12	175	128	project	35
37	36	masters	1	5, 8, 19	175	101, 136	research	36
38	37	bachelor	1	6, 21	175	168, 215, 128	internship	37
39	38	masters	1	20, 7	175	148, 84, 18, 166, 156	internship	38
40	39	bachelor	1	11, 5	175	90	internship	39
41	40	masters	1	13, 7	175	46, 203	internship	40
42	41	bachelor	1	2, 5, 22	175	231, 90, 191	internship	41
43	42	masters	1	23, 4, 17, 19, 5	175	84, 16, 194, 215, 122	research	42
44	43	masters	1	3, 6, 19	175	29, 21, 229, 175	internship	43
45	44	bachelor	1	7, 6, 1	175	39	research	44
46	45	bachelor	1	8, 16, 17, 18, 10	175	148, 158, 90, 38, 18	project	45
47	46	masters	1	21, 19, 3, 4, 20	175	77, 119, 65	project	46
48	47	phd	1	20, 14, 2	175	159, 47, 188	project	47
49	48	bachelor	1	10, 2, 3, 16	175	201	internship	48
50	49	bachelor	1	4, 18, 23, 13	175	158, 148	internship	49
51	50	masters	1	15, 7	175	179, 91, 142, 68, 56	project	50
52	51	masters	1	3, 1, 19	175	12, 205, 164, 155	internship	51
53	52	bachelor	1	21, 15, 5, 10	175	22, 147	project	52
54	53	bachelor	1	12, 18, 8, 5, 3	175	182, 128, 219, 58, 54	internship	53
55	54	bachelor	1	20, 23	175	26	internship	54
56	55	masters	1	5, 21, 2, 6, 7	175	13, 187, 152	internship	55
57	56	bachelor	1	8, 16, 10, 4, 21	175	172, 79, 48	project	56
58	57	bachelor	1	22, 5, 4, 12	175	181	research	57
59	58	masters	1	13, 20, 5, 7, 18	175	150, 163, 205, 225	internship	58
60	59	bachelor	1	11, 21, 3	175	75, 87, 142	project	59
61	60	bachelor	1	15, 16	175	223, 28, 157, 190, 212	project	60
62	61	masters	1	23, 10, 5, 4, 9	175	32, 21, 23, 71, 78	project	61
63	62	masters	1	2, 21, 1	175	163, 102, 157, 225	research	62
64	63	masters	1	16, 3	175	9	project	63
65	64	bachelor	1	9, 10	175	81, 50, 205, 68	internship	64
66	65	masters	1	10, 15, 1, 21, 7	175	79, 195, 5, 88, 221	internship	65
67	66	bachelor	1	10, 7	175	200	research	66
68	67	bachelor	1	18, 23	175	143, 153	research	67
69	68	bachelor	1	12, 11, 1	175	141, 109, 211, 5, 12	project	68
70	69	masters	1	3, 2	175	53, 87, 49, 117, 97	project	69
71	70	bachelor	1	17, 5, 18, 10, 2	175	46	project	70

Figure 32 - Excerpt of Student's Dataset

5.1.2.2 Recommender Engine

This subsection will dive deep into the implementation process of the RS. The objective of this RS is to take into consideration previous generated recommendations and the student's properties and recommend projects based on that information. With that into consideration, it was built an hybrid filtering RS that has both collaborative and content-based filtering. The combination of the results of both techniques is expected to generate more accurate recommendations. In addition, to improve recommendation personalization, NLP was added on top of those recommendations if the user provides a project description.

For the collaborative filtering, the datasets generated in Section 5.1.2.1 need to be utilized since it is necessary to have prior data for this technique (cold-start problem). Having that data, the chosen algorithm to process it is the matrix factorization algorithm. The algorithm was written using Pandas for data organization, as well as TensorFlow and NumPy. Firstly, data is pre-processed into Pandas datasets, to easily integrate with TensorFlow and NumPy functions. Then, the user-item matrix is created. In the matrix, users represent students, while items are represented by projects. The records of that matrix are the reliability percentages of each recommendation. Every other record that does not have a recommendation associated is assigned to a 0. For that reason, the matrix is classified as non-binary because it has further information about the recommendations' quality. To train and test the model, the matrix is

divided into training and testing sets, that corresponds to 80% and 20% of the matrix data respectively.

The next step consists of decomposing the matrix to extract meaningful patterns. It factorizes the training matrix into latent factors that represent student and project preferences. In other to find the latent factors and other important variables such as the best learning rate and regularization values, the system performs a technique called “Grid Search”. That technique consists of training the model with several different values multiple times to assess which are the best values for the model. After having those values, they are then used to optimize the model by using the alternating least squares optimization algorithm. The final step is to evaluate the model by using a testing set and calculate the MAE and RMSE of those models. Since Grid Search is used to find the best values for the model, it is ensured that the error values are the smallest possible.

Collaborative filtering model is saved after training, to be then used when the recommendations endpoint is called. This algorithm is executed once the system is being started, and after that it is performed every 24 hours to accommodate new recommendations. Figure 33 illustrates the collaborative filtering process in the form of a diagram.

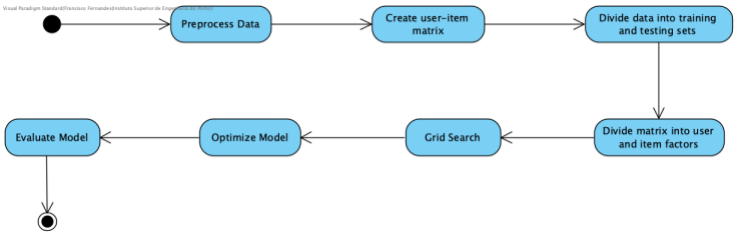


Figure 33 - Collaborative Filtering Activity Diagram

Contrary to the collaborative filtering method, the content-based filtering technique focus on the student’s data by comparing it with the available projects and supervisors. The algorithm is executed once the student’s requests to generate new recommendations. This technique uses several recommendation metrics to apply weight to each project recommendation depending on its characteristics. These metrics are applied to specialization areas, project types and location attributes. The values of each metric can be changed as specified in Figure 20 by UC7.

The algorithm starts by getting all the information about specialization areas, locations, projects and supervisors. After also retrieving the recommendation metrics from the database, the RS organizes the student’s and project’s attributes into NumPy arrays. Those attribute arrays will be then converted into decimal binary arrays (0 to 1). If the student or project has a certain specialization area, project type or location, the correspondent binary array value will be the weight of the attribute. For the student’s binary array, locations are considered when they are within a 20km radius.

Once the student’s and project’s binary arrays are built, they are compared using a distance similarity measure. In this case, the chosen algorithm is the cosine similarity. The similarity scores are then sorted from the most to less similar, and it is retrieved the first 5 results which

correspond to the most suited recommendations. For each recommendation, it is necessary to assess the best supervisors.

The same process described earlier is applied to the supervisors, however it will only be considered their specialization areas and their preferred specialization areas as attributes. Supervisors that have preference in a certain specialization area that is also present in the project will have a better score in the array in comparison to the same specialization area that is not their preferred one. Similarity distance is also calculated by cosine similarity and sorted by descent order to return the best 5 supervisors for the project recommendation. Figure 34 illustrates the content-based filtering process in the form of a diagram.

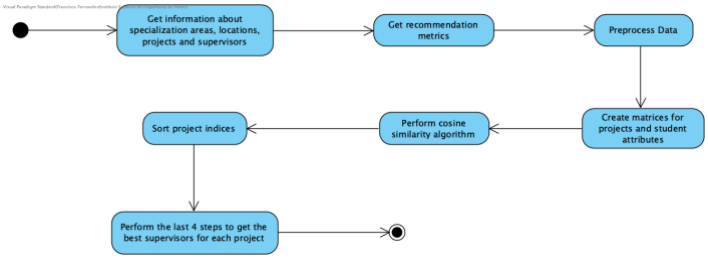


Figure 34 - Content Based Filtering Activity Diagram

When generating the recommendations, if the user provides a project description, NLP is also applied to the RS. It was decided to adopt a non DL algorithm with NLP (as discussed in Section 2.2.4) since this application does not have enough data to be trained in a DL context. Even though there are NLP implementations that have multi-language support, this processing is only applicable to English. Content-based filtering is applied firstly, and after doing the cosine similarity calculation to obtain the best recommendations, NLP is applied to the best recommendations to reorder them.

NLP compares the description provided by the student with the project's descriptions. The first step to do is perform a pre-processing function to all project descriptions and the user description, which consists of eliminating language stop words (e.g.: the, on, in, is, when, etc.), punctuation and performing stemming to convert words to their base form (e.g.: running, runs, ran → run). Those pre-processing techniques are provided by a Python package called NLTK.

With the pre-processed descriptions, TF-IDF is applied to them to transform the descriptions into numerical vectors, as explained in Section 2.2.4. Those vectors are then compared with a cosine similarity algorithm to find the most similar descriptions to the one the user wrote. The top recommendations are then reordered based on those results (with the same strategy as the content-based filtering). Figure 35 illustrates NLP in the form of a diagram.

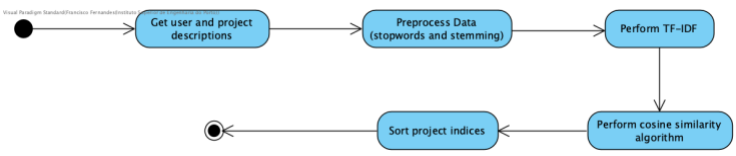


Figure 35 - Natural Language Processing Activity Diagram

5.1.2.3 Endpoints

The endpoints this service exposes are the following:

No Actor

- **Get all Study Areas** (`GET /study-area`): Responsible for retrieving all the study areas registered in the database;
- **Get all Countries** (`GET /country`): Responsible for retrieving all the countries registered in the database;
- **Get all Specialization Areas** (`GET /specialization-area`): Responsible for retrieving all the specialization areas registered in the database of a given study area;
- **Get all Locations** (`GET /location`): Responsible for retrieving all the locations registered in the database of a given country.

Student Actor

- **Register Student** (`POST /student`): Responsible for registering the remaining information associated to the user with role "student". A student must fill in his study area, preferred specialization areas, country, preferred locations, preferred type of project, and academic degree. Returns a successful message if the registration is completed, otherwise gives the appropriate error;
- **Get Student** (`GET /student`): Responsible for getting all the information regarding a given student. Returns that information if the student exists, otherwise gives the appropriate error;
- **Update Student** (`PUT /student`): Responsible for updating any information that a student desires. Returns a successful message if the update is completed, otherwise gives the appropriate error;
- **Get Recommendations** (`GET /recommendation`): Responsible for generating the recommendations for the student. Takes into consideration all the information about the student, and compares it to the trained dataset and registered data of supervisors, projects, and other relevant entities. If recommendations were generated before, it retrieves the generated recommendations from the database, otherwise it generates using the recommender engine;
- **Rate Recommendation** (`POST /recommendation/rate`): Responsible for associating a given rating given by the student to the generated recommendation (rating varies from 1 to 5);
- **Apply to a Project** (`POST /application`): Responsible for sending an application to the project to one of the recommended supervisors. Returns a successful message when applying to that project recommendation, otherwise it gives the appropriate error.

Supervisor Actor

- **Register Supervisor** (`POST /supervisor`): Responsible for registering the remaining information associated to the user with role "supervisor". A supervisor must fill in his study area, specialization areas, preferred specialization areas, academic degree, initial and final dates of supervision, and the maximum number of students that he is willing to supervise. Returns a successful message if the registration is completed, otherwise gives the appropriate error;
- **Get Supervisor** (`GET /supervisor`): Responsible for getting all the information regarding a given supervisor. Returns that information if the supervisor exists, otherwise gives the appropriate error;
- **Update Supervisor** (`PUT /supervisor`): Responsible for updating any information that a supervisor desires. Returns a successful message if the update is completed, otherwise gives the appropriate error;
- **Register a new Project** (`POST /project`): Responsible for creating a new project that can be added to the recommendation list of projects. The registered projects do not associate the supervisor, since the recommendation process suggests the best possible supervisors for each of the recommended projects. A project must have a title, description, study area, specialization areas, location, type of project, academic degree and duration (initial and final dates). Returns a successful message if the project was successfully created, otherwise gives the appropriate error;
- **Get all Applications** (`GET /application`): Responsible for retrieving all the applications students have made to a given supervisor;
- **Assign to a Project** (`POST /recommendation/assign-project`): Responsible for accepting the application of the student and assign that project to the student. As soon as the project is accepted, it can no longer be recommended to other students. The service communicates with the LMS service via AMQP to transform that recommendation into a project. Returns a successful message if the project was correctly assigned, otherwise gives the appropriate error.

Admin Actor

- **Get Recommendation Metrics** (`GET /recommendation/metrics`): Responsible for retrieving the recommendation metrics used in the recommender engine. Those metrics are specialization areas, locations and project type, and the sum of all of them is equal to one;
- **Adjust Recommendation Metrics** (`POST /recommendation/metrics`): Adjust one or more recommendation metrics. Those metrics have weights which directly influence how the recommender engine classifies each project. Returns a successful message if the sum of all recommendation metrics after the update is equal to one, otherwise gives the appropriate error.

5.1.3 LMS Service

As soon as a recommendation is converted to a project, the LMS service can be utilized. During the conversion process, the recommendation service shares the info about the student and supervisor associated to that recommendation to the LMS service via AMQP. Those entities are then saved in the LMS database.

Students and supervisors can use almost all the endpoints of that service. Both can see the files and reports associated to the project, upload or download new files, update reports and see the history of all messages exchange between them. Students are however the only ones who can upload brand new reports to the system since they are the authors of the document.

In order to upload files and/or reports to the system, it is necessary that those assets respect the supported file extensions. Uploaded files must be either images, Word, Excel, PowerPoint, PDF or text files, while reports need to be PDF files exclusively. If the files meet those criteria, they are stored as local files in the server under the `/uploads/<project_id>/files` or `/uploads/<project_id>/reports` directory (depending on the type of the uploaded file). Therefore, the database does not save the file directly, however it saves the path of the file in the system in order to be used on the download endpoint. When the user requests for the download of a file or report, the system returns the file as a response using the `send_from_directory` auxiliary function provided by Flask.

5.1.3.1 Live Chat

The service also allows for the exchange of messages between student and supervisor. To simulate a live chat experience, WebSockets must be used. WebSocket can be seen as a constant connection between the client and the server, as opposed to an HTTP request. With that, both the client and server can send updates between each other "(…) without clients needing to poll at certain intervals" (Mardan, 2018). There are libraries that assist in the process of implementing those sockets in the server and the client, and one of them is the Socket.IO library (Mardan, 2018). Socket.IO can be easily implemented in a Flask project by using the `flask_socketio` package.

Since this service does not communicate directly with the front-end counterpart, it will only expose the necessary endpoints of the Message entity. Instead, the API Gateway will have Socket.IO configuration implemented, and will communicate with the LMS service to register the messages sent by the user. Successful stored messages are then returned in the socket to the client to create the live chat experience. The Socket.IO implementation when uploading a new message can be seen in Code 1.

```
@socketio.on('upload_message')
def handle_message(message):
    session_id = request.sid
    user_data = session.get(session_id)
    token = user_data.get('authorization')
    project_id = request.args.get('projectId')
    response = requests.post(os.getenv("LMS_APP_URL") + '/project/' + str(project_id) + '/messages',
json={'message': message}, headers={'Authorization': 'Bearer ' + token})
    data = response.json()
    message = data['message']
    socketio.emit('message', message)
```

Code 1 - Socket.IO Upload Message Listener

5.1.3.2 Endpoints

The endpoints this service exposes are the following:

Student and Supervisor Actors

- **Get Project** (`GET /project`): Depending on the user executing this endpoint, it can retrieve all the projects a supervisor is currently supervising, or the project associated to a given student. If the project does not exist, gives an error;
- **Get All Files** (`GET /project/<id>/files`): Responsible for retrieving all the files associated to the project;
- **Upload File** (`POST /project/<id>/files`): Responsible for uploading a file and associate it to a given project. The file can be an image, Word, Excel, PowerPoint, PDF or text file. Returns a successful message if the file respects those formats and was uploaded, otherwise gives the appropriate error;
- **Get All Reports** (`GET /project/<id>/reports`): Responsible for retrieving all the reports associated to the project;
- **Update Report** (`PUT /project/<id>/reports/<report-id>`): Responsible for sending an updated version of a report, to let the other person know that a certain user made changes to a given report. The updated version of a report must be a PDF file. Returns a successful message if the report respects the PDF format and was uploaded, otherwise gives the appropriate error;
- **Download File** (`GET /project/<id>/files/<filename>`): Responsible for retrieving an Uniform Resource Locator (URL) containing the file to be downloaded;
- **Download Report** (`GET /project/<id>/reports/<filename>`): Responsible for retrieving an URL containing the report to be downloaded;
- **Download All Report Version History** (`GET /project/<id>/reports/<report-id>`): Responsible for retrieving an URL containing all the previous versions of a given report.
- **Get All Messages** (`GET /project/<id>/messages`): Responsible for retrieving all the messages that were exchange between the student and supervisor on a given project.

Student Actor

- **Upload Report** (`POST /project/<id>/reports`): Responsible for uploading a report and associate it to a given project. The report must be a PDF file. Returns a successful

message if the report respects the PDF format and was uploaded, otherwise gives the appropriate error;

Since the API Gateway has a Socket.IO implementation for live chat, the following endpoints are associated to that socket:

- **Register a new Message** (`POST /project/<id>/messages`): Responsible for sending and storing a message sent by a given user.

5.2 System Architecture – Front-end

Besides building the back-end infrastructure, a front-end system is also a requirement for students, supervisors and admin to interact with the system in a user-friendly manner (as identified in Section 4.1.5).

In light of that, a front-end project was created using an open-source JavaScript framework called Svelte, which is considered one of the most loved frameworks amongst developers. Besides still being a relatively new framework compared to React, Angular or Vue, it is already being used in enterprise projects and companies such as The New York Times, GitHub and GoDaddy (Bhardwaz and Godha, 2023). According to the official Svelte documentation, the most intuitive way to building Svelte web applications is with the use of their application framework called SvelteKit (Svelte, 2023a).

A new SvelteKit project was created from scratch. In order to help the creation process of the user interface, it was used a free Cascading Style Sheet (CSS) template[1] which was adapted to fulfil the application requirements and accelerate the development process. SvelteKit requires a specific folder structure when building web applications, which can be seen in Figure 36. In this case, the `src` folder contains two main directories:

- `lib`: Responsible for allocating all the library code that will be used throughout the application (such as reusable components and other sharable files). When importing files in this directory, the $lib alias is used instead of the full relative path;
- `routes:` As the name suggest, it contains the routes of the application, and it is the only mandatory directory of SvelteKit. Directories inside this folder correspond to a path segment in the URL. For example, if the directories are organized in this manner: `student/dashboard/+page.svelte`, the URL path to access to that page is going to be `FRONTEND_URL/student/dashboard` (with `FRONTEND_URL` being a representation of the application URL).
  Besides that, filename conventions must be respected, since files related to pages must start with a '+'. There are several types of files that can be used in `routes`, and the most used ones in this application are:
  - `+page.svelte`: Corresponds to a page of the application;

1 - https://adminmart.com/product/modernize-free-bootstrap-5-admin-template/

- `+page.server.ts`: Can contain a load function that loads data before the page is rendered. Since it has 'server' in the name, the file is executed in the server (by using server side rendering). The returned data from that function can be then used in `+page.svelte` and subsequent Svelte components;
- `+layout.svelte`: Corresponds to a part of the page that always visible on every page;
- `+layout.server.ts`: Same features as `+page.server.ts` but for `+layout.svelte`;
- `+server.ts`: Corresponds to an HTTP endpoint that gives the opportunity to take full control over the response.

Inside the `lib` directory, files were divided by their main functionalities:

- `components`: Contains all reusable components (`.svelte` files) that will be used throughout the application pages;
- `model`: Contains type declaration files (`.d.ts` files) that represent each of the domain's entities. Those type declarations match what is defined in the back-end and serve as type checking for the service's methods and components;
- `services`: For every entity, there is a service file which has functions that point to all of the endpoints defined in the API Gateway. Those files are also responsible for data processing before returning the response back;
- `stores`: As the name suggest, these files represent Svelte Stores, which are a global state management mechanism that can be accessed by all pages and components of the application without the use of props or event listeners;
- `utils`: Contains all auxiliary and general functions that are used more than once in the application.

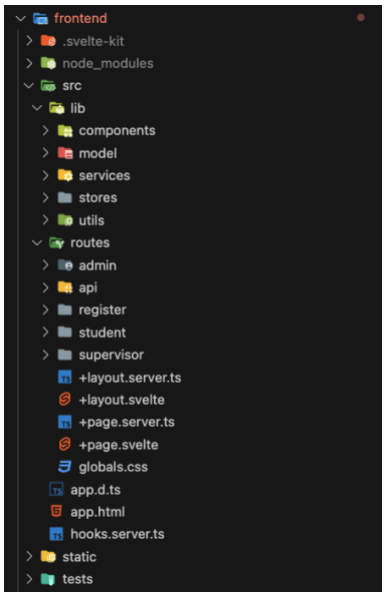


Figure 36 - Front-end Folder Structure

Touching on the authentication and authorization process, the application implements mechanisms to only allow certified users to access the application's endpoints. The first one is located in the `hooks.server.ts` file represented in Figure 36. That file implements a function that is executed every time a request in the application is made, and sees if the JWT Token is inside the cookies. If it is, it decodes the token's information and returns it. The second mechanism is implemented in the `+layout.server.ts` file. If the JWT token information is returned from `hooks.server.ts`, the layout function will ensure that the user is only redirected to the allowed page (e.g.: the student will only be able to access to `/student/dashboard` pages and not `/supervisor/dashboard` ones).

According to Figure 36, routes are organized by actors to simplify the authorization process. Each of the actors have their own dashboard, where they can interact with the allowed functionalities. There is also an `api` directory that contains HTTP endpoints (defined in `+server.ts` files) that communicate directly with the API Gateway via the services files. Those files are used to have more control over the HTTP response.

To support the live chat functionality described in Section 5.1.3.1, the front-end needs to also have a Socket.IO implementation. Code 2 reflects the connection between the front-end and API Gateway sockets, and useful listeners such as when the client connects to the socket (which sends the JWT token to the server to authorize communication) and when a message is received (adds the received message to the array of all messages). It also has a function that sends a message by emitting it to the server using the `upload_message` event.

```
<script lang="ts">
	import { PUBLIC_BACKEND_URL } from "$env/static/public";
	import type { PageData } from "./$types";
	import { io } from 'socket.io-client'

	export let data: PageData;
	let messages: MessageOutputDTO[] = data.messages

	let message = '';

	const socket = io(PUBLIC_BACKEND_URL + "?projectId=" + data.project.id, {
		transports: ['websocket', 'polling', 'flashsocket'],
		auth: {
			token: data.token
		}
	});

	socket.once('connect', () => {
		socket.emit('authorization', {
			token: data.token
		})
	})

	socket.on('message', (msg: MessageOutputDTO) => {
		messages = [...messages, msg];
	})

	const sendMessage = () => {
		socket.emit('upload_message', {
			content: message
		})
		message = '';
	}
</script>
```

Code 2 - Socket.IO Implementation Front-End

Before starting the application, the environment file needs to know which is the back-end's URL. The project uses `pnpm` has the package manager and it is necessary to execute `pnpm install` to download all the dependencies of the project, as well as Svelte and SvelteKit packages. With all

the dependencies installed, executing `pnpm dev` starts the front-end development server. However, in the production environment, the same principle made for the back-end is also applied to the front-end. A `Dockerfile` is defined to build an image of the front-end, using commands such as `pnpm build` to build a production-ready bundle for a Node.JS container. When executing the command `docker-compose up -d` in the same directory of `docker-compose.yml`, all the images of the back-end and front-end start running according to their dependencies.

Due to the fact that the application was deployed in the author's machine, and it is not accessible to the public, some screenshots of the application can be seen at Appendix B (it does not represent the entirety of the project).

5.3 Deployment

Since this system is composed of several microservices, the adopted strategy was to use Docker containers to deploy it into production. All services have their own `Dockerfile`, which is responsible to transform the service into a Docker container capable of running in a completely different environment. The root of the project contains a `docker-compose.yml` file that is responsible to orchestrate how each container is built, the running commands, the database volumes to use, and dependencies with each other. Figure 37 shows how the project is deployed in Docker by showing all of the project's containers, as well as the ports each one exposes.

The API Gateway is located in port 5000, and communicates with the other back-end services at ports 5001, 5002 and 5003. The RabbitMQ service (`message-broker-1`) exports two ports: one that serves as a communication service (5672) and other that gives access to a admin dashboard to control RabbitMQ's behaviour. The host machine and network were configured to allow for outside requests on ports 5000 and 3000 (that correspond to the API Gateway and Frontend respectively).

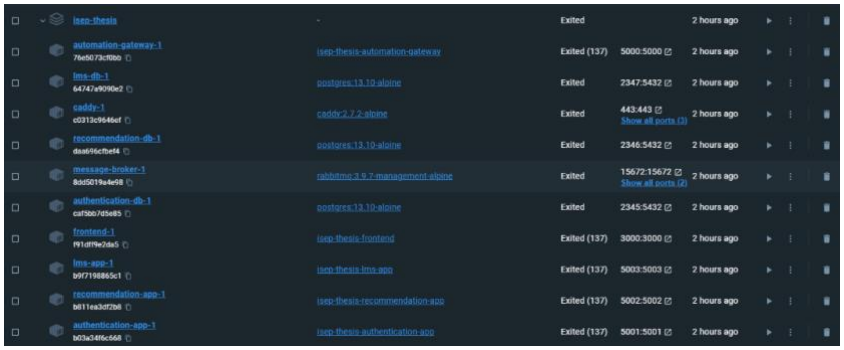


Figure 37 - Docker Containers of the Application

In Figure 37 there is also a Docker container named `caddy-1` that represents a Caddy image. Caddy is an open-source web server that supplies automatic HTTPS for secure connections. This tool was used as a reverse proxy to the local machine.

Since the system is being hosted on the author's machine and network, it was necessary to modify the network's settings to allow connections for port 443 (HTTPS default port), which is the one Caddy uses. A domain was used in conjunction with Cloudflare's DNS services to point the URLs `master-thesis.ffwork.space` and `api-master-thesis.ffwork.space` to the network's public IP address. The request will be intercepted by Caddy, and proxied to port 3000 or 5000 depending on the request URL. In this case, `master-thesis.ffwork.space` will redirect to the front-end, while `api-master-thesis.ffwork.space` will redirect to the API Gateway.

5.4 Summary

The implementation chapter dives deep into the technical aspects of the system. The chapter is divided into two main categories that symbolise both parts of the system: the front-end and the back-end. Although the back-end is divided into its three main services (authentication, recommendation and LMS), it shares common traits between all services. All of the services have their own Dockerfile (container), database, and common important Python files.

The API Gateway and frontend ports are exposed in the host machine and network. Communication between services happens using a RabbitMQ implementation and AMQP, and all services are connected to an API Gateway that communicates directly with the front-end. All services are built with Flask framework from Python, but have separate functionalities (that are all listed in detail by describing each one of the exported endpoints). The authentication service is responsible for logging in and register new users. When performing either action, a JWT token is returned and must be attached to every request made to the other services. The recommendation service is not only responsible for generating recommendations for the student, but also manage applications and register student and supervisor preferences. The process of building the recommender engine was started by doing data mining with the help of the CRISP-DM framework. Then, two algorithms were made: one collaborative filtering mechanism using Matrix Factorization, and one content-based filtering mechanism using cosine binary matrices and cosine similarity. NLP is also considered when a user enters a project description, and it was developed by using the TD-IDF mechanism. The LMS service is responsible for managing the student's project and allow for both students and supervisors to upload files and reports, and talk to each other. Those functionalities were developed with the help of Flask auxiliary methods, as well as Socket.IO for real time chat interaction.

The front-end was developed using a JavaScript framework called Svelte. Svelte was used in conjunction with SvelteKit to make a full-stack web application. SvelteKit follows a strict folder and file structure that must be complied. The project was developed using an open-source CSS template, and a Socket.IO implementation was also built to communicate directly with the back-end via the API Gateway.

The last subsection summarizes the process behind deploying the application to the public by using Caddy as a reverse proxy and a domain.

6 Experimentation and Evaluation

In the chapter of Experimentation and Evaluation, evaluation metrics are defined, and the thought process of evaluating them is described by specifying their process, tools to be used and result interpretation. Coding tests to the application are also described, and the results of each metric are analysed and discussed.

6.1 Ethics in Software Engineering

During system development of several UCs, there were some ethical concerns to be respected. The RS does not take into consideration user sensible data, and it is completely anonymous, meaning that only the main properties listed in Chapter 4 and 5 (specialization areas, preferred locations and project type) are considered for recommendations. The model is trained with previous recommendations done by the system, to ensure more rigorous results. Therefore, when creating a new account, the user is only asked to insert the minimum required data, and sensible data (e.g.: passwords) is encrypted and saved in the database. Measures to safeguard user data from unauthorized access were also implemented, as explained by port forwarding and HTTPS implementation in Section 5.3, authorization policies (with the use of role-based strategies) and various implementation techniques explained throughout Chapter 5.

Application testing was also taken into consideration, as it will be discussed during this Chapter. This system is geared towards people that are currently enrolled in an academic course or university teachers. Thus, people need to be above 18 years of age and be in one of those conditions to use the application. Students and supervisors were randomly selected to test the application to avoid result bias. Anonymity of the questionnaire results was ensured to promote honest answers and feedback. Any feedback given to the system (will) serve as continuous improvement and adjustment.

6.2 Evaluation Metrics

Evaluation metrics (or indicators) are important aspects that needed to be tested to assess if the developed system is answering all the problems stated at the beginning of the dissertation. The system is evaluated according to several evaluation metrics, including:

- **Performance**: The application, namely the RS, needs to provide good performance for all stakeholders. One of the problems identified during the analysis was the fact that students waste copious amounts of time searching for the most appealing project, which translates into a monotonous task and potentially missed opportunities. Besides that, supervisors need a performant application to adopt that system rather than their usual technological stack;
- **Reliability**: Since the application has a RS integrated, as well as other UCs, it needs to ensure that the best results are given to students and supervisors. Although project selection and monitorization is something that is not standardized and is target of criticism from both parties, the system needs to retrieve reliable information and be stable enough in order to be adopted in an university context. Besides that, the system must be written with availability and maintainability in mind;
- **Usability**: It is necessary to know if the developed functional and non-functional requirements in the system can help all stakeholders in the project development process. The RS and LMS need to have all the functionalities to solve all the problems stated throughout the dissertation, without neglecting user interface and experience topics stated in Section 4.1.5.

6.3 Evaluation Methodology

To evaluate the metrics referred to in Section 6.2, the evaluation methodology for all of them is explained, by mentioning which evaluation process to be followed, the tools needed to evaluate them, the results that are gathered and how they can be used.

6.3.1 Evaluation Process

The performance metric is evaluated by performing performance tests. The goal of the performance tests is to verify if the system is capable of having several concurrent users (students and supervisors) using the application at the same time. Besides that, other types of performance tests are directly targeted towards the RS, as previously stated in Section 4.1.5.4. Performance must be acceptable when using the RS functionalities, if not the purpose of having a RS to help students minimize their time searching for projects would be non-existent.

Information about the reliability metric is obtained by using specific methods for measuring RS accuracy. Explained with more detail in Section 2.2.4.2, prediction accuracy is one of the main evaluation properties to take into consideration when evaluating a RS. This metric calculates how accurate the generated recommendations are. The precision accuracy assesses if the RS is

producing accurate results. It is obtained by using two different methods: equations and classification. The first one addresses the MAE and RMSE of the RS. The second one classifies the RS into a confusion matrix to calculate precision, recall, false positive rate and F-Measure. Besides the prediction accuracy, availability and maintainability of the software are also checked by verifying if the system can keep working in specific scenarios and if documentation and code quality principles were adopted during implementation.

Finally, the usability metric addresses how all stakeholders experience the application by allowing them to test and verify if all the developed functional requirements were correctly implemented. Feedback on what to develop in the next iterations and potential refinements to the developed UCs are also registered. This type of interaction is recorded by handling custom questionnaires to each stakeholder and by tracking user interaction in the application. Additionally, usability is also evaluated by analysing the rating tax given amongst the developed UCs.

6.3.2 Tools

The performance metric is measured by simulating concurrent users using the same feature at the same time (e.g. several students using the RS at the same time). For that, it is used dedicated API testing tools such as Postman, that can also allow to run performance tests. Postman can simulate a determined amount of concurrent users requesting one more requests at the same time, as well as test each request independently with several different configurations.

Prediction accuracy is assessed by running the RS in a testing dataset with all the necessary information. Since the system runs independently from any third-party service, the dataset is built using other relevant datasets online. Once the RS can produce results successfully, prediction accuracy metric is calculated using the expressions revealed in Section 6.3.1.

The usability metric is measured by tracking how the user interacts with the system, which functionalities are used and not used, and how quickly a user can access certain UC or information. These type of interactions are important to know which functionalities are relevant to solve the stated problems and which ones need improvements and upgrades. To achieve that, web tracking is one tool to visualize patterns in user behaviour (Atterer, Wnuk and Schmidt, 2006). According to a study made in 2006 about user activity tracking, there are several ways of recording user behaviour such as using an HTTP proxy, client-side scripting or software to track mouse interaction, eye tracking devices and server-side tracking (Atterer, Wnuk and Schmidt, 2006). However, since testing will be done remotely using the user's own devices, it is problematic to configure those tools in different machines and it can also be seen as a privacy violation. Therefore, it will not be used to measure usability. With that in mind, questionnaires will be created (one for each stakeholder) using Microsoft Forms, which allow for several question formats and statistics.

6.3.3 Result Interpretation

All the described metrics in the Section 6.3 generate results that are translated into a statistic and graphic formats for better understanding and readability.

Firstly, the performance metric assessed by Postman (that simulates several concurrent users) concentrate on properties such as latency, response time, and how many users the system can handle before stopping to answer.

Results from the reliability metric vary depending on the calculated property, namely when mentioning prediction accuracy. The higher the values on the error properties MAE and RMSE, the less accurate the RS predictions are. However, the higher the values on the classification properties Precision and Recall, the more rigorous the RS predictions are.

The usability metric will be measured through questionnaires, which will evaluate the user satisfaction when using the application, the functionalities that need some refinement and which further feedback the users want to give to this version and future ones. With those answers, several graphs and statistics will be presented to understand the user's answers. QEF will be used as a tool to assess software quality and evaluate the retrieved data from both the questionnaires and the RS classification and error values. All the questions of both questionnaires can be seen at Appendix C and D respectively.

6.4 Testing

Testing ensures that the application is producing the correct outputs. The approach to testing the system was to build unit and system tests for the back-end, and E2E tests for the front-end. Unit tests will be used to test specific algorithm methods regarding the recommender engine and domain rules. In contrast, the system and E2E tests will provide general coverage of every use case and functionality. Regardless of the test type, successful responses are tested, as well as thrown errors to verify if the output was expected. The following subsections will be dedicated to exploring each of those types of tests in more detail.

6.4.1 Unit Tests

Unit tests aim to verify the correctness and reliability of individual components of a software application by isolating and testing small parts of an application separately. Examples of that are functions or classes.

Since this application does not have a high level of complexity in terms of domain rules, it was decided to solely use those tests to examine the RS's algorithm reliability. Even though code coverage in unit testing is something that several companies and programmers strive for, domain rules are already being tested in System and E2E testing by analysing each UC as a whole. Conversely, the RS contains complex algorithms that cannot be tested separately, and it is

necessary to check its values to ensure recommendations are being well generated. Unit tests were made for both filtering techniques:

- Collaborative Filtering: `__matrix_decomposition()`, `__optimize_model()`, and `__grid_search()`;
- Content-Based Filtering: `__create_binary_indicator_matrix()` and `__calculate_distance()`

6.4.2 System Tests

System tests encompass all modules of the back-end application. Those tests check UCs in a global manner, in this case from the controller module (where the request is received) until the repositories (where it communicates with the database).

Postman was the elected tool to build those tests. By creating one collection for each service, those requests are separated by service (which improves organization) and can be tested separately. Postman tests are then written in the "Tests" tab inside each request in JavaScript language using their embedded testing API. Figure 38 shows the tests performed for each service, which are organized by UC.

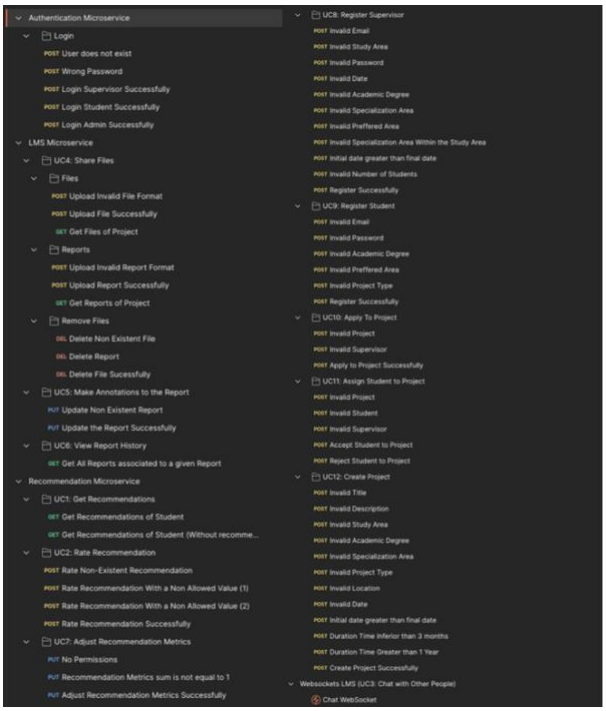


Figure 38 - List of System Tests

6.4.3 E2E Tests

E2E tests cover the whole application by testing the application flow from front-end, to the back-end and database. Those tests can simulate real user interactions with the web application,

and can also test other important aspects of the system such as error handling, user interface interactions and request orchestration.

Since the application is using SvelteKit as its front-end meta framework, it comes with Playwright integration out-of-the-box (Svelte, 2023b). Playwright is an open-source E2E tool developed by Microsoft that is capable of testing every web system regardless of the framework used in three different environments: Chromium, Firefox and WebKit. Since it is a Microsoft project, it has seamless integration with Visual Studio Code, which helps with test development. Figure 39 shows the side-menu of Visual Studio Code containing all the developed E2E tests, as well as some of their main features.

To run the Playwright tests, the whole system must be running, and a testing database must be setup to avoid any entity conflicts. Playwright tests are written in JavaScript and are located in the front-end project, under the `tests` directory. A Playwright test starts with the `test` function, which accepts a function with a `page` argument where the whole E2E test will be written. The argument allows to interact with the page as an user by clicking in page elements, typing, selecting options, and other interactions. If there is a need to check some value or property, Playwright offers a `expect` function where it is possible to write test assertions (e.g.: expect a certain property to have certain value).

Figure 39 - List of E2E Tests

From Figure 39, the get recommendations without project description test is shown at Code 3. Before executing any test to ensure they can run independently from each other, a new student/supervisor is always registered, and then logged in before each test execution. However, when registering a new student, it is not necessary to execute the login function since the new user is automatically logged in upon registration. Test executions are re-used in other test files to avoid rewriting the same code multiple times.

Prior to any test execution, Playwright simulator goes to the recommendations page and ensures it is fully loaded before running the next procedures. The test starts by clicking on the

"Generate Recommendations" button. The simulator waits until the `div` element with `data-testid` property "recommendations" is attached to the DOM and visible. If everything goes right, the `expect` function is called to check if that `div` is visible on the screen and if there are 5 recommendations inside it.

```
import { expect, test } from '@playwright/test';
import { registerStudent } from './uc9.test';
import { login } from './authentication.test';
import dotenv from 'dotenv';
import path from 'path';

dotenv.config({ path: path.resolve(__dirname, '.env.test') });

let registeredStudent = false;

test.beforeAll(async ({ page }) => {
        await registerStudent(page);
        registeredStudent = true;
});

test.beforeEach(async ({ page }) => {
        if (registeredStudent) {
                await login(page, process.env.EMAIL_STUDENT, process.env.PASSWORD);
        }
        await page.getByText('Get Recommendations').click();
        await page.waitForURL('**/student/dashboard/recommendations');
});

test('get recommendations without project description', async ({ page }) => {
        await page.getByText('Generate Recommendations').click();

        //Wait for recommendations to show up
        const recommendations = page.getByTestId('recommendations');
        await recommendations.waitFor({ state: 'attached' });

        //Check if the recommendations parent element is visible
        expect(recommendations).toBeVisible();

        //Check if the recommendations parent element has 5 child divs
        expect(recommendations.locator('div').count()).toBe(5);
});
```

Code 3 - E2E Test of Get Recommendations

6.5 Results

In this section, results for each metric will be analysed and discussed by analysing various statistics. Both the student and supervisor's questionnaires and their respective answers will be analysed alongside the QEF, which will also assess if the defined metrics were considered during implementation and their impact on software quality. Results can be seen in Appendix E.

6.5.1 User Feedback Questionnaires

In order to evaluate the system in a real world scenario, questionnaires for the main actors of the system (students and supervisors) were elaborated. A specific number of students and supervisors were allowed to fully interact with the finished product and fill the answers and feedback in the form.

The first group to test the application was the students. Since the system database contains only IT-related projects from ISEP's platform, it was necessary that the students who tested the

application had (or are pursuing) an IT degree in any school/university. Considering that restriction, it was possible to gather answers from 15 different students.

The next group was the supervisors. Because the author did not have any direct contact with IT teachers, and also due to their time constraints, it was impossible to gather the same sample size. However, 5 different supervisors were able to test the application and answer the questionnaire.

The answers from the student's[1] and supervisor's[2] questionnaires can be found in a separate Excel spreadsheet and will be used in the Subsection 6.5.2 to complement the analysed dimensions. In the following subsections, the context and motivation behind this project will be analysed inside the point of view of students and supervisors.

6.5.1.1 Context

The main purpose of this application was to make the project selection process efficient and to make the monitorization process easier for supervisors. The questionnaires' answers reflect that statement, since the current method of choosing the final project in universities averaged a rating of 3.64 (on a scale of 1 to 10) across 11 students that already participated in this process. This average rating is achieved regardless of the method used in their universities. Some of the students did not participate in any curricular project as of yet, however all participants agreed that they would use a RS if their university offered a similar tool. When asked on how they evaluate their current method of project monitoring, supervisors were satisfied (6.4 on a scale of 1 to 10) but all of them were keen to try a new method if it was available to test. It is then possible to conclude that even though supervisors use their own tools to supervise their projects, they acknowledge that improvements to their supervision's efficiency could be made.

6.5.2 Quantitative Evaluation Framework

QEF is an evaluation model that has the goal of assessing software quality by verifying if the implementation objectives were completed (Escudeiro and Bidarra, 2008). QEF models specify three main attributes:

- **Dimensions**: Represent the most important points to be achieved in software development. Dimensions in this project are represented by the metrics specified in Section 6.2;
- **Factors**: Associated to a dimension, it represents a more detailed manner to test a given dimension. Factors for each dimension in this project are designated in Section 6.3;
- **Requirements**: Associated to a factor, it represents a standard that should be strived. Each requirement has its own relevance (rw), which is a value that ranges from 0 (irrelevant) to 10 (key requirement), as well as a fulfilment (wf) that represents how well the requirement was implemented (it ranges from 0 to 100%). Requirements, as well as the full QEF model for this project, can be seen in Appendix E.

1 - https://is.gd/p1RZ46 2 - https://is.gd/BgRbIt

After identifying QEF's main attributes, system quality is discovered by determining the value of the following properties:

- **Factor Weight in Dimension (w_{ij}):**

$$w_{ij} = \frac{n_{rw \to factor}}{n_{rw \to dimension}}$$

- **Achieved Performance per Factor ($Factor$):**

$$Factor = \frac{1}{\sum rw} * \sum rw * wf$$

- **Achieved Performance per Dimension ($Dimension$):**

$$Dimension = \sum w_{ij} * Factor$$

- **Global Deviation (D):**

$$D = \sqrt{\sum \left(1 - \frac{Dimension}{100}\right)^2}$$

- **System Quality (q):**

$$q = \left(1 - \frac{D}{\sqrt{n}}\right) * 100$$

The following subsections deeply analyse the mentioned dimensions according to their requirements, the answers from both questionnaires and what was established in the QEF model at Appendix E.

6.5.3 Performance

Performance tests assess if the application is ready to accommodate a large amount of users at the same time, and still deliver responses in an adequate time. In fact, as discussed in Section 2.2.2, people will often look for what they desire based on cost and benefit, so the system need to retrieve its data (especially the RS results) in a reasonable time. Since this application is supposed to be used by students and supervisors from different institutions, it needs to be capable of supporting a certain number of concurrent users. Time constraints and concurrent users were specified in the QEF model.

To test the system, Postman will be used. Postman provides a "Performance Test" functionality that allows simulation up to 100 concurrent users on a given amount of time. Performance tests were executed on the author's computer, which is the device that is hosting the whole system. Instead of performing the requests to the localhost URL, the requests are made to `api.ffwork.space` to also them in a production environment.

The machine used for these tests has an Intel i7-11700F processor, along with a GeForce RTX 3060Ti graphics card and 16GB of RAM. Besides that, the data used for testing was the same as used for the system implementation, which is explained with more detail in Section 5.1.2.

The decision of testing only GET requests of the application lies solely on the fact that most of the time, those functionalities will be requested more often than POST's or PUT's, and also because of the fact that it would cause the database to store copious amounts of information that the computer storage cannot handle.

The following endpoints were tested:

- Get Study Areas;
- Get Locations;
- Download File;
- Get Recommendations.

It is worth noting that the following results may not be the same if they were executed in a different environment, or the project and supervisor database had more entries. Performance tests were done using the upper limit allowed on Postman (100 users) and 1 minute of test duration.

The first test consists of executing the Get Study Areas and Get Locations endpoints together since they are mostly executed at the same time when using the front-end application. Both endpoints do not return a huge amount of data (around a few kilobytes). The results of the test can be seen in Figure 40.

Figure 40 - Performance Test of Get Study Areas and Locations

The average response time of the system was 419ms, even though there were a high number of request with response times over that amount. Although 419ms represents a discrepancy of 39% compared to the 300ms desired response time, it is still a fast response time and it will possibly not be noticed by users. There are also some spikes up to 1800ms which are not ideal

when interacting with those specific features, however it could be explained by other factors such as network overload.

The next performance test performed was to the download file endpoint. The endpoint requests the download of a PDF file that has roughly 500 kilobytes of data. The results of the test can be seen in Figure 41.

Figure 41 - Performance Test of Download File

Contrary to the previous test, this one is much slower, with an average of 4935ms of response time. There is a steadily increase in response time on the first few seconds of the test, however that value stabilizes in the 4500ms range. Those values suggest that the time of that request is directly impacted by the size of the downloaded file and the user's network. If the downloaded file is bigger than the one tested, the requests would most likely have a bigger response time.

The last performance test executed was to the get recommendations endpoint. To test the recommendation system performance, the UC was altered so that new recommendations are always generated, regardless of any registered record in the database (as explained in Section 4.2.2.1). The results of the test can be seen in Figure 42.

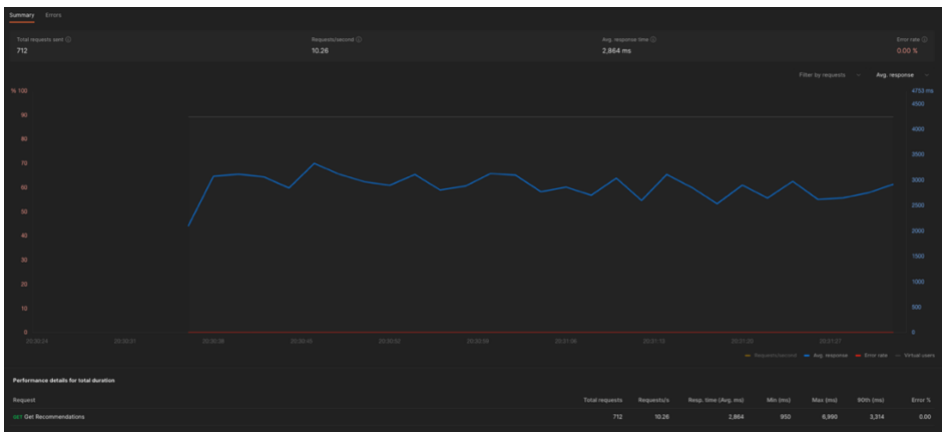


Figure 42 - Performance Test of Get Recommendations

The test report shows a 2864ms response time, which is well below the defined target of 5 seconds for each request to the RS. There are some fluctuations in response time, and the maximum registered value was almost 7 seconds, however the 90th percentile of requests situated at around 3 seconds. This results show that a capable machine with GPU acceleration contributes to a good performance of the RS. Nonetheless it is important to keep in mind that since the database has only ISEP's projects, those results could suffer changes if the tests were to be performed with a bigger dataset.

6.5.4 Reliability

The most important factor of reliability is the prediction accuracy, since it is necessary that the RS is producing the most accurate results, which therefore translates into more user satisfaction. To test this factor, the dataset will be used to assess the errors associated with the collaborative filtering part of the RS, while the answers from both the students and supervisors will assess the content-based filtering part of the RS by calculating the precision, recall, false positive rate, and F-Measure.

The recommendation system performs a matrix factorization algorithm every 24 hours when initialising the application. As explained in Section 5.1.2.2, the algorithm conducts a grid search to assess the best variables to use for this RS by optimizing the algorithm using the alternative least squares optimization. The system will calculate the loss of the algorithm by calculating the MSE with the defined learning rate and regularization values after optimization. As the iterations go by, the MSE progressively decreases. The values with the lowest MSE will be used to calculate the RMSE and MAE of the RS with the training and testing datasets. Figure 43 shows both of those error values, the left one being from the training set, and the right one being from the testing set.

Figure 43 - Error values from training and testing datasets

For the other test to evaluate the prediction accuracy of the RS, data from the student's questionnaire was used. The goal of this evaluation is to determine the quality of the generated recommendations, and also check which weight variables are the most suited for the majority of students. As discussed in Section 5.1.2.2, the algorithm will produce different results according to the weight given to each metric. To determine one variable combination that would work for most users, students tested the RS with different values:

- **Custom Settings #1: Specialization Areas**: 0.33, **Project Type**: 0.33, **Locations**: 0.33 → Equal preference in every variable;
- **Custom Settings #2: Specialization Areas**: 0.5, **Project Type**: 0.2, **Locations**: 0.3 → More preference towards specialization areas, and locations;
- **Custom Settings #3: Specialization Areas:** 0.5, **Project Type**: 0.3, **Locations**: 0.2 → More preference towards specialization areas, and project type;

The average score of all settings are mostly the same, averaging a value of 7 on a scale of 1 to 10 (custom settings #1: 7.1, custom settings #2: 6.9, custom settings #3: 7.3). However, there is a slight preference towards the custom settings #3, which prioritize specialization areas and project type. This result is proven by the average value in student's preferences when choosing a project. Students give more preference to specialization areas (8.4), followed by project type (7.7), and finally locations (6.5).

The next step of evaluation is to calculate prediction accuracy through classification. To retrieve that information, the section "Student Information" was used to assess if the preferences chosen by each student was in accordance with the generated recommendations. Additionally, each student was asked to fill some additional information on the side by specifying if the given recommendations were relevant or not (true and false positives), and if other 15 randomly selected projects that were not recommended could be relevant or not (true and false negatives) using the best settings, which in this case is the custom settings #3. Findings of this research can be found in Table 14.

Table 14 - Recommendation evaluation for each student

	Recommendation #1	#2	#3	#4	#5	#6	#7	#8	#9	#10	#11	#12	#13	#14	#15	#16	#17	#18	#19	#20
Student #1	✗	✓	✓	✓	✗	✗	✗	✗	✗	✗	✓	✗	✗	✗	✗	✗	✗	✗	✗	✓
#2	✓	✓	✓	✓	✓	✗	✓	✗	✓	✓	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗
#3	✓	✓	✓	✓	✓	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✓	✗
#4	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✓	✗	✗	✗	✗	✗	✗	✗
#5	✓	✗	✓	✗	✓	✓	✗	✗	✗	✓	✓	✓	✗	✗	✗	✗	✗	✗	✗	✗
#6	✓	✓	✓	✓	✓	✓	✓	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗
#7	✓	✓	✓	✓	✗	✓	✓	✓	✗	✗	✗	✗	✓	✗	✓	✗	✓	✗	✓	✓
#8	✗	✓	✓	✓	✓	✗	✗	✗	✗	✗	✓	✓	✗	✓	✗	✗	✗	✗	✗	✓
#9	✗	✓	✓	✓	✓	✗	✗	✗	✗	✓	✗	✗	✗	✗	✗	✗	✗	✗	✓	✗
#10	✓	✓	✓	✓	✗	✗	✗	✗	✓	✗	✗	✓	✗	✗	✓	✗	✓	✗	✗	✓
#11	✓	✓	✓	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗
#12	✗	✓	✗	✓	✗	✓	✗	✓	✗	✗	✗	✓	✗	✗	✗	✗	✓	✗	✗	✗
#13	✓	✓	✓	✗	✓	✓	✓	✓	✗	✗	✓	✗	✗	✓	✗	✗	✗	✗	✗	✓
#14	✓	✗	✗	✓	✓	✓	✗	✗	✗	✓	✓	✗	✗	✗	✗	✓	✓	✗	✓	✗
#15	✓	✓	✗	✓	✓	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓	✓	✗	✗	✗	✓

The results from each student will be used to classify the prediction accuracy using the equations stated in Section 2.2.5.2. From the testing settings where it was possible to gather 300 different recommendation evaluations (20 recommendations for each student), precision, recall, false positive rate and F-Measure have the following values:

$$Precision = \frac{TP}{TP + FP} = \frac{53}{53 + 52} \cong 0{,}51$$

$$Recall = \frac{TP}{TP + FN} = \frac{53}{53 + 22} \cong 0{,}71$$

$$False\ Positive\ Rate = \frac{FP}{FP + TN} = \frac{52}{52 + 173} \cong 0{,}23$$

$$FMeasure = \frac{2 * Precision * Recall}{Precision + Recall} = \frac{2 * \ 0{,}51 * 0{,}71}{0{,}51 + 0{,}71} \cong 0{,}59$$

Precision of the RS sits at around 0.51 or 51%, which means from all the supposed recommendations, the system is only recommending good projects half of the time. In comparison to most of the tools analysed during Chapter 2, these results may indicate less optimal recommendations for students. However, and also taking into consideration the recall value of 0.71, it is possible to assume that a possible solution to increase precision is to increase the number of recommendations from 5 to 10 or 15, or utilize the description functionality to obtain more tailored ones (by utilizing a more complex and developed NLP algorithm), since the data points out that some false positives are being left out. Another aspect that can be concluded is that project recommendations for students are more complex than just relying on their characteristics, since some projects that perfectly match the student's interests may not be a good recommendation due to other factors such as the project description or theme. Nevertheless, this tool serves as an auxiliary mechanism to filter projects and get the most suited projects more quickly. F-Measure value that encompasses precision and recall sits at 0.59. This value could have been higher if the precision value was also higher, however in counterpart it also indicates that the RS does not miss a significant number of possible recommendations.

False positive rate sits at 0.23 or 23%, which means only a small amount of random projects could be a good recommendation for the student. However, it is important to point out that this does not represent the whole dataset of the system (due to the amount of projects students needed to evaluate), so this results must be analysed with caution. Still, a small false positive rate is a result every RS should be striving for.

In terms of availability, as discussed in Section 5.3, the system is available worldwide over an HTTPS connection by accessing the URL: `master-thesis.ffwork.space`. Even though the whole system composed of several microservices is running on a single machine, the application is capable of maintaining some degree of functionality if there is one or more microservices not working properly or offline.

An example of that is the authentication microservice. If the user is already logged in the system, he can interact with the whole application as normal since each microservice implements a JWT decryption function capable of identifying the user. If one of the other microservices (recommendation or LMS) starts malfunctioning or goes offline, the user is capable of interacting with the UCs of the microservice that is running properly. However, UCs that require communication between microservices (e.g. assign student to project) will not work as expected. Those asynchronous messages passed to RabbitMQ will be received by the consumer once the microservice is online again.

Having that into consideration, data is not lost unless the RabbitMQ service stops working. There is also data replication between databases of each microservice, and automatic backups

of the recommendation dataset, however a more concrete backup strategy should have been stated to avoid complete data loss.

The final factor to evaluate is maintainability. The main pillars of maintainability in a software engineering context are documentation, tests and code quality and organization. The system was produced after the development of the analysis and architectural design specified in Chapter 4. Besides that, Chapter 5 explains the thought process, tools, API reference and procedures made during system implementation, which also serves as further documentation. In terms of code documentation, every microservice offers a Swagger integration that specifies each API endpoint the service offers, but it was not further explored due to development time restrictions.

Testing coverage is detailed in Section 6.4, where unit, system and E2E tests are written. Those tests cover the whole application by evaluating important methods of the RS, all the UC's and the application flow from front-end, to back-end and database.

Code quality and organization is respected by following the directory rules specified in Chapter 5, as well as reutilizing common methods and components. Both the front-end and back-end implement specific features that help enhance maintainability. On one hand, the front-end follows the development guidelines from SvelteKit by organizing pages into the `routes` directory, and following a service-based approach (e.g. one file for one domain entity) where requests of a given entity are made to the back-end and processed right after. On the other hand, the back-end follows a controller, service, repository architecture with dependency injection to isolate each module to reduce coupling and facilitate unit and/or integration testing.

Finally, the whole application code is stored in GitHub, a cloud-based storage service that works with Git, allowing for version control, issue tracking and pull requests, which enhances the development process.

6.5.5 Usability

Across both questionnaires, opinions about usability were extremely positive overall. Most students and supervisors pointed that the interaction between UCs and overall user experience were friendly and easy to use. Figure 44 shows the overall results of two questions from the "Application and Evaluation Feedback" section of the questionnaire, while Figure 45 indicates the satisfaction rate when using the application.

Students

23. On a scale of 1 to 10, how do you evaluate the developed functionalities in terms of usefulness?

15 Respostas

ID	Nome	Pontuação
1	anonymous	9
2	anonymous	8
3	anonymous	9
4	anonymous	7
5	anonymous	8
6	anonymous	8
7	anonymous	9
8	anonymous	9
9	anonymous	8
10	anonymous	2
11	anonymous	10
12	anonymous	10
13	anonymous	8
14	anonymous	8
15	anonymous	10

24. On a scale of 1 to 10, how user friendly do you consider this application?

15 Respostas

ID	Nome	Pontuação
1	anonymous	8
2	anonymous	7
3	anonymous	8
4	anonymous	7
5	anonymous	9
6	anonymous	9
7	anonymous	9
8	anonymous	9
9	anonymous	9
10	anonymous	10
11	anonymous	9
12	anonymous	7
13	anonymous	9
14	anonymous	10
15	anonymous	10

Supervisors

10. On a scale of 1 to 10, how do you evaluate the developed functionalities in terms of usefulness?

5 Respostas

ID	Nome	Pontuação
1	Maria Mota	7
2	Luiz Faria	9
3	anonymous	5
4	anonymous	9
5	anonymous	8

11. On a scale of 1 to 10, how user friendly do you consider this application?

5 Respostas

ID	Nome	Pontuação
1	anonymous	7
2	anonymous	9
3	anonymous	3
4	anonymous	7
5	anonymous	8

Figure 44 - Questionnaire Results of Usability

Students

22. Do you think this recommendation system can be useful for you and others?

26. Do you think this system has all the necessary functionalities to be used in other universities and study areas?

27. If your university gave you the opportunity to use a similar system, would you use it?

Supervisors

13. Do you think this system has all the necessary functionalities to be used in other universities and study areas?

Mais Detalhes

14. Do you think this system can be a substitute to all of your tools for project monitoring and communication with the student?

Mais Detalhes

15. If your university gave you the opportunity to use a similar system, would you use it? (0 ponto)

Mais Detalhes

Figure 45 - Satisfaction Rate of the Application

On a scale of 1 to 10, the average rating of application's usability for students is 8.2 (with only one negative review), while the average rating of user experience and ease of use is 8.7. This results in an application usefulness success rate of nearly 100%. Therefore, students would want to use it in an academic context. Supervisors' ratings on the other hand are slightly inferior, yet still averaging 7 on a scale of 1 to 10 (7.6 on usability and 6.8 on ease of use).

Besides those values, there was a clear tendency to choose the RS as the most useful feature of the application (with 9 students choosing it) followed by the live chat functionality (with 5 students preferring it). The complete opposite happened with the supervisors, since there was not one clear functionality that stood over the rest.

The questionnaire has a dedicated field to introduce missing functionalities and further feedback. Since part of the evaluation process was made during the final development phases of the system, some minor feedback suggestions were implemented right away, such as making an accordion component for reports such as the latest version is at the highest position. However, most of the feedback and missing functionalities are discussed with more detail in the future work section of Chapter 7.

Even though the questionnaire prints a good impression about the application's usability, there are some aspects that should also be discussed. As discussed in Section 5.2, the application's front-end adapted an open-source CSS Template which offered aesthetically pleasing objects and good user experience. This template was then adapted to include several custom objects and support for responsive design, in order to be compatible with smaller devices such as smartphones. One of those custom objects is the notification, that pops up every time certain action was successfully completed or an error occurred during execution. The only usability feature not developed in the system is the inclusion of a Frequently Asked Questions section. For first time users, it could help use the system more effectively, however it was decided not to do it due to prioritization of other features. Besides that, due to the small number of features, it would not offer a large benefit (in this current state).

Accessibility is also an important factor to take into consideration, since in an academic context, there are both students and supervisors that suffer from various types of disabilities. To accommodate those variables, the system was developed by utilizing the correct HTML DOM elements (e.g.: using `<p>` for paragraphs, `<h1>` for first headings, `<button>` for clickable buttons, etc.) and properties (e.g.: all images have `alt` attribute defined, which will be shown if the image cannot be displayed). By using the correct HTML elements and properties, it becomes easier for other devices such as screen readers to understand the application's content. A big enough font size was also used throughout the website to avoid readability problems, and correctly adapted to various screen sizes.

Another accessibility need was to allow some UCs to be used with only a keyboard to make it more intuitive to use, such as the login, register, update profile and live chat. All of those functionalities have input texts and can be saved by clicking the "Enter" key.

6.6 Summary

The experimentation and evaluation chapter explores the three evaluation metrics that were tested throughout the system: performance, reliability, and usability. For every metric, details about the methods and tools were examined. For the performance metric, performance tests were performed using Postman. For the reliability metric, RMSE and MAE values for the collaborative filtering part of the RS algorithm were determined, and classification of the RS and other important information was obtained with the data from the questionnaires. For the usability metric, the feedback and answers to the questionnaires from both students and supervisors were used.

There is also a dedicated section for testing, that explores the three types of testing done throughout the application: unit, system and E2E tests. Unit tests were solely done to the machine learning methods to ensure the calculations were producing correct values. System tests to every use case of the system (which encompasses the back-end and databases) were developed using Postman. Finally, E2E tests were performed by using Playwright as the testing tool by checking the whole application flow (from front-end, to the back-end and databases).

To help determine if the metrics were successfully implemented and check software quality, it was decided to use QEF where each metric (dimension) is divided into factors, and each factor has its requirements evaluated from 0 to 100 to assess if it was fully completed.

For the performance dimension, the executed performance tests with 100 concurrent users to the RS were below 5 seconds, while the remaining UCs tested were slightly above the targeted values but does not translate in a slower user interaction.

For the reliability metric, more precisely the prediction accuracy factor, precision, recall, false positive rate and F-Measure values were calculated by analysing 300 different recommendation evaluations from 15 different students. Precision of the RS is 0.51, while the recall is 0.71 and F-Measure is 0.59. Although the precision value is not a very high value and it could be improved with more recommendations shown to the students, the RS still serves as a tool to filter projects more quickly and effectively. False positive rate value is around 0.23, which shows that the RS does not fail to recommend a lot of projects. The availability factor was evaluated by determining if the system was available through a public domain, and testing the system with certain microservices down and its asynchronous communication. Lastly, the maintainability factor was evaluated by checking if code quality patterns and designed architecture were respected. Both of those factors were greatly achieved.

For the usability metric, data from the questionnaire from students and supervisors was used to assess the ease of use and accessibility of the application. Feedback from both actors indicated that the application's front-end has a user friendly interface and experience with support for mobile devices, as well as legible font size and keyboard-only functionalities to improve productivity and avoid the use of the mouse peripheral.

7 Conclusion

In the conclusion chapter, an overview about the system's implementation and evaluation will be made, and the research questions made during the Introduction chapter will be answered. Additionally, limitations and future work to the application will be stated.

7.1 Objective Analysis

The hypothesis raised in this thesis culminated in a system capable of aggregating the selection process of academic projects and monitoring the student's selected project. As a greenfield system, the whole architecture, structure and implementation was designed and developed from scratch. Besides that, the data used for the RS was retrieved from a low number of sources, due to the difficulty in finding academic projects that could suit the application's purpose.

After finishing the application's implementation, experimentation and evaluation was done with the help of QEF and its definition of dimensions, factors and requirements. By having those metrics defined, it was possible to compare the defined requirement attributes with the feedback retrieved from students and supervisors, as well as the results from tests and implementation.

According to the results gathered in Section 6.4, the developed system successfully integrated a solution that unites the selection and monitoring processes of academic projects, making it a unique solution that is more suited to solve the stated problems than the ones described in Section 2.4. In fact, this RS consider more variables than the analysed solutions (specialization areas, project type, location and project description with NLP) and could have a similar rate of precision if small adjustments were made to the RS (as discussed in Section 6.4). It also includes a project monitoring dashboard which, although includes a smaller number of functionalities than SciPro and eThesis, contains the necessary UCs to be used for every university and study area (which is something the prior tools lack).

With that, it is possible to determine if the questions raised during Chapter 1 were answered:

- What is the best approach for helping a student in the decision-making process of choosing the most interesting project?

According to what was explored during the Chapter 2, there is a correlation between time spent and getting the most accurate results according to the user's preferences when considering large amounts of information. Generally, it is necessary to spend more time searching for the best possible results. Even though it is possible to reduce the number of results by specifying certain filter properties, it does not give a concrete representation on all user preferences, since some of them may be outside of only checking item's properties. That is why the RS option was the one chosen for this application, since it can recommend projects in terms of importance of each variable and other aspects, such as the project description.

As demonstrated by the feedback received from the students, the current methods used for project selection were heavily criticized (as seen in Figure 46), and they were willing to experiment a RS that would give project options depending on their preferences.

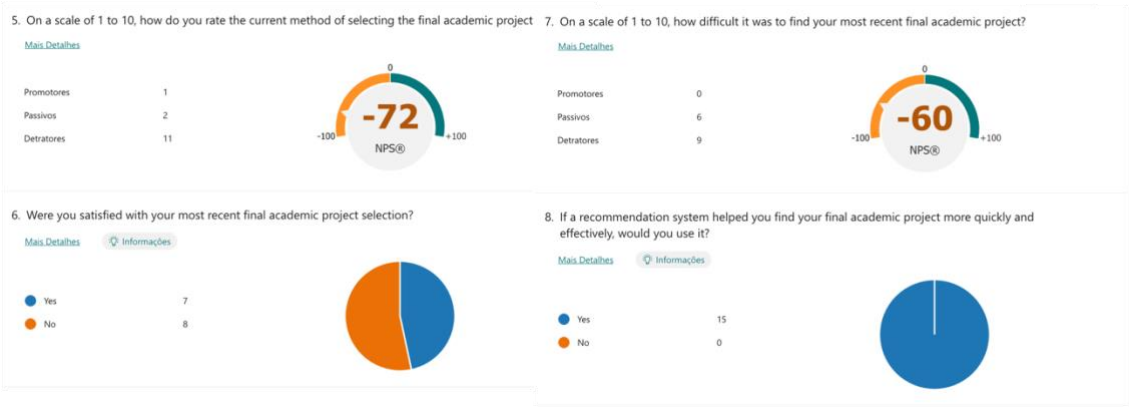


Figure 46 - Project Selection Satisfaction

Even though the results of the RS are slightly below the expectations set in the QEF model, satisfaction is very positive across the majority of students. However, as noticed during the Section 6.5.4, the RS can still improve by discovering hidden patterns in the projects, such as using a more complex NLP algorithm that filters the best projects according to what the student wrote as his preferred project description. Nonetheless, the main purpose of finding the most suited projects in the least amount of time is achieved by this current RS implementation (as seen in Figure 45).

- What are the main points of interest for a given student that have a significant impact on the decision-making process of a certain project?

By analysing the different related projects in Section 2.4 and taking into consideration the principles analysed during the Chapter 2, it was possible to come up with several attributes that can be applied to this domain. During the analysis and design of the system, three main

attributes shared between students and projects were identified: specialization areas, locations and project types. Since this application is meant to support all students and projects regardless of their study area, university and location, those attributes are seen as common variables, and other specific project preferences (e.g. languages the project is written for IT students) were not considered (which would improve RS accuracy for certain students).

By looking at the student's questionnaire results at Section 6.5.4, they give more preference to specialization areas, followed by project type, and locations. The results also say that students prefer a RS that benefits more specialization areas and project type. However, project descriptions also play a big role in finding the most suitable projects. Analysing the Table 14 where students were asked to say if a given project would be considered a good or a bad recommendation, there were some projects that do not match all the student's preferences, yet they are still recommendations due to their descriptions. That is the reason why it was also included an optional field for students to introduce their preferred project description, which will be processed by an NLP algorithm.

- How to maximize time efficiency in the selection and monitoring phases?

Time efficiency during the project selection process was already discussed in the first research question. By having a RS, students can have a list of possible project recommendations without having to search between multiple project proposals, and read each of the projects' descriptions carefully. Even though recommendations cannot be considered accurate and definitive answers, they can be regarded as a general guideline towards finding the optimal project. Besides that, for every recommendation there is also a recommended set of supervisors that are suited to supervise the project. That is a relevant point, because besides having to search for the best project, students need also to find a suitable supervisor and most of the time they do not know the teacher's specialization areas and if they are capable of supervising their project.

The project monitoring process is best maximized when there is one centralized application where all the functionalities lie, as seen in the related systems analysed in Section 2.4. Depending on the supervisor, different applications will be used to communicate and monitor the student's work. An example of that would be to use Dropbox as a cloud service to store files and the report, Microsoft Teams as the communication tool and email as a last resource if communication between both parties failed for some reason. The use of several different tools for monitoring several students may lead to loss data, miscommunication, and lack of report version history. As seen in the Chapter 6, supervisors pointed out in the questionnaire that their current project monitoring process could see some improvements, and a system that aggregates everything in one place would be a good solution for that problem. However, all of them pointed out that the current version would need more functionalities and refinement to be ready to be used in an academic context.

7.2 Limitations

During system's implementation and evaluation, some limitations were encountered.

One of those limitations lies in the developed NLP algorithm. The NLP algorithm is implemented by using TF-IDF, which verifies which project description is more similar to the one the student wrote. However, it fails to determine other important aspects such as the positioning of the words in the sentence's context, semantics, and the ability to identify synonyms. TF-IDF was the chosen mechanism to implement NLP due to its simplistic implementation and the ability to work with less available data. If other more complex techniques were used in this system (e.g. deep learning), the system would be able to identify other hidden patterns and be more accurate.

Another limitation found was during system's trial by both students and supervisors. Due to the late testing stage (from August to September), many possible candidates were unavailable to participate in the trial phase because of vacations, or professional and/or personal reasons. As mentioned in Section 6.5.4, it was more challenging to contact teachers because of not having a direct communication line with them. Therefore, the number of supervisors testing the application was considerably lower than the number of students, which ultimately impacted the obtained results.

The last limitation of the system is present in certain UCs. The project recommendations' ratings (UC2) does not integrate with the current implementation of the RS (even though it is fully implemented). Students can rate their generate recommendations from 1 to 5, however those results do not impact the subsequent generated recommendations. Also, the chat functionality (UC3) does not send an email for unseen sent messages. It was not possible to develop those concepts due to time constraints, and other more important UCs were prioritized instead.

7.3 Future Work

Despite the positive feedback from students and supervisors, as well as the fulfilment of the proposed objectives, there is always room for improvement in a software context.

The first aspect to point out is that even though the application's RS was tested with real IT projects, it was not evaluated with other study areas, due to the lack of projects in the dataset. Despite the assumption that the same results would be applied to other study areas due to the attributes being the same, it needs further testing to test that hypothesis. Therefore, a larger dataset with more projects would be necessary. Besides that, due to the fact that this RS is not of higher complexity (since it uses simpler algorithms and techniques), a result comparison between the presented approach and a new RS with more sophisticated algorithms and/or DL capabilities could be done to investigate if there are significant improvements to the recommendation accuracy.

The other point focuses on the application's front-end. Students said that the application’s usability offers ease of use and prioritizes a seamless user experience, but at the same time some of them gave feedback notes and missing functionalities to take into consideration in future iterations of the system. Examples of those features are a page of instructions and/or frequently asked questions page, group chats, calendar, and others. Small feedback notes given during the evaluation period were implemented right away in order to make the system more robust. On the other hand, supervisors gave more diverse feedback due to the fact that each of them have their own way of monitoring projects and it is difficult to please everyone. Examples of missing features are the enhancement of the notification system, calendar to list important dates, events and appointments, or a list (history) of conversations with timestamps. Besides that, the majority of supervisors agreed that limiting the report format to PDF is not a good approach, since each one has their preferences regarding document editing.

References

Abadi, M. *et al.* (2016) 'TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems'. Available at: https://arxiv.org/abs/1603.04467 (Accessed: 8 January 2023).

Adomavicius, G. and Tuzhilin, A. (2005) 'Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions', *IEEE Transactions on Knowledge and Data Engineering*, 17(6), pp. 734–749. Available at: https://doi.org/10.1109/TKDE.2005.99 (Accessed: 5 January 2023).

Al-Shamri, M.Y.H. (2016) 'User profiling approaches for demographic recommender systems', *Knowledge-Based Systems*, 100, pp. 175–187. Available at: https://doi.org/10.1016/j.knosys.2016.03.006 (Accessed: 6 January 2023).

Ana Azevedo and Manuel Filipe Santos (2008) *KDD, SEMMA AND CRISP-DM: A PARALLEL OVERVIEW*. Available at: http://hdl.handle.net/10400.22/136 (Accessed: 18 December 2022).

Anuradha and Gupta, G. (2014) 'A self explanatory review of decision tree classifiers', in *International Conference on Recent Advances and Innovations in Engineering (ICRAIE-2014)*. IEEE, pp. 1–7. Available at: https://doi.org/10.1109/ICRAIE.2014.6909245 (Accessed: 13 January 2023).

Atterer, R., Wnuk, M. and Schmidt, A. (2006) 'Knowing the user's every move', in *Proceedings of the 15th international conference on World Wide Web*. New York, NY, USA: ACM, pp. 203–212. Available at: https://doi.org/10.1145/1135777.1135811 (Accessed: 12 February 2023).

Aydın, Ö. and Karaarslan, E. (2023) 'Is ChatGPT Leading Generative AI? What is Beyond Expectations?', *SSRN Electronic Journal* [Preprint]. Available at: https://doi.org/10.2139/ssrn.4341500 (Accessed: 7 February 2023).

Ayodele, T.O. (2010) 'Types of machine learning algorithms', *New advances in machine learning*, 3, pp. 19–48. Available at: https://www.intechopen.com/books/3752 (Accessed: 28 December 2022).

Bhardwaz, S. and Godha, R. (2023) 'Svelte.js: The Most Loved Framework Today', in *2023 2nd International Conference for Innovation in Technology (INOCON)*, pp. 1–7. Available at: https://doi.org/10.1109/INOCON57975.2023.10101104.

Bi, Q. *et al.* (2019) 'What is Machine Learning? A Primer for the Epidemiologist', *American Journal of Epidemiology* [Preprint]. Available at: https://doi.org/10.1093/aje/kwz189 (Accessed: 18 December 2023).

Borza, J. (2011) 'FAST diagrams: The foundation for creating effective function models', *Trizcon Detroit* [Preprint]. Available at:

http://new.aitriz.org/documents/TRIZCON/Proceedings/2011-06_FAST-Diagrams-The-Foundation-for-Creating-Effective-Function-Models.pdf (Accessed: 15 January 2023).

Bouchrika, I. (2022) *51 LMS Statistics: 2023 Data, Trends & Predictions*, *Research.com*. Available at: https://research.com/education/lms-statistics (Accessed: 12 January 2023).

Brynjolfsson, E. and Mitchell, T. (2017) 'What can machine learning do? Workforce implications', *Science*, 358(6370), pp. 1530–1534. Available at: https://doi.org/10.1126/science.aap8062 (Accessed: 18 December 2022).

B.Thorat, P., M. Goudar, R. and Barve, S. (2015) 'Survey on Collaborative Filtering, Content-based Filtering and Hybrid Recommendation System', *International Journal of Computer Applications*, 110(4), pp. 31–36. Available at: https://doi.org/10.5120/19308-0760 (Accessed: 29 December 2022).

Burke, R. (2000) 'Integrating Knowledge-based and Collaborative-filtering Recommender Systems', *Proceedings of the Workshop on AI and Electronic Commerce*, pp. 69–72. Available at: https://www.researchgate.net/publication/2418883_Integrating_Knowledge-based_and_Collaborative-filtering_Recommender_Systems (Accessed: 2 January 2023).

Byungura, J.C. (2015) 'E-learning management system for thesis process support from a supervisor perspective: The case of SciPro System at University of Rwanda'. Available at: http://urn.kb.se/resolve?urn=urn:nbn:se:hj:diva-27573 (Accessed: 14 January 2023).

Camlek, V. (2011) 'How to spot a real value proposition', *Information Services & Use*, 30(3–4), pp. 119–123. Available at: https://doi.org/10.3233/ISU-2010-0615 (Accessed: 15 February 2023).

Catherine, R. and Cohen, W. (2017) 'TransNets: Learning to Transform for Recommendation', in *Proceedings of the Eleventh ACM Conference on Recommender Systems*. New York, NY, USA: ACM, pp. 288–296. Available at: https://doi.org/10.1145/3109859.3109878.

Cavus, N. (2015) 'Distance Learning and Learning Management Systems', *Procedia - Social and Behavioral Sciences*, 191, pp. 872–877. Available at: https://doi.org/10.1016/j.sbspro.2015.04.611 (Accessed: 9 January 2023).

Chapman, P. *et al.* (2000) *CRISP-DM 1.0*. Available at: https://www.kde.cs.uni-kassel.de/wp-content/uploads/lehre/ws2012-13/kdd/files/CRISPWP-0800.pdf (Accessed: 28 December 2022).

Chung, L. and do Prado Leite, J.C.S. (2009) 'On Non-Functional Requirements in Software Engineering', in, pp. 363–379. Available at: https://doi.org/10.1007/978-3-642-02463-4_19 (Accessed: 22 January 2023).

coates, H., james, R. and baldwin, G. (2005) 'A Critical Examination Of The Effects Of Learning Management Systems On University Teaching And Learning', *Tertiary Education and*

Management, 11(1), pp. 19–36. Available at: https://doi.org/10.1007/s11233-004-3567-9 (Accessed: 9 January 2023).

Cohen, S. (2021) 'The evolution of machine learning: past, present, and future', in *Artificial Intelligence and Deep Learning in Pathology*. Elsevier, pp. 1–12. Available at: https://doi.org/10.1016/B978-0-323-67538-3.00001-4 (Accessed: 23 February 2023).

Cover, T. and Hart, P. (1967) 'Nearest neighbor pattern classification', *IEEE Transactions on Information Theory*, 13(1), pp. 21–27. Available at: https://doi.org/10.1109/TIT.1967.1053964 (Accessed: 28 December 2022).

Deng, F. (2015) 'Utility-based Recommender Systems Using Implicit Utility and Genetic Algorithm', in *Proceedings of the 2015 International Conference on Mechatronics, Electronic, Industrial and Control Engineering*. Paris, France: Atlantis Press. Available at: https://doi.org/10.2991/meic-15.2015.197 (Accessed: 6 January 2023).

Escudeiro, P. and Bidarra, J. (2008) 'Quantitative Evaluation Framework (QEF)', *Conselho Editorial/Consejo Editorial*, p. 16.

Ferreira, F.C. and Oliveira, A.A. (2012) 'OS SISTEMAS DE RECOMENDAÇÃO NA WEB COMO DETERMINANTES PRESCRITIVOS NA TOMADA DE DECISÃO', *Journal of Information Systems and Technology Management*, 9(2). Available at: https://doi.org/10.4301/S1807-17752012000200008 (Accessed: 24 December 2022).

Géron, A. (2022) *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow*. O'Reilly Media, Inc. Available at: https://www.oreilly.com/library/view/hands-on-machine-learning/9781492032632/ (Accessed: 9 January 2023).

Gevorkyan, M.N. *et al.* (2019) 'Review and comparative analysis of machine learning libraries for machine learning', *Discrete and Continuous Models and Applied Computational Science*, 27(4), pp. 305–315. Available at: https://doi.org/10.22363/2658-4670-2019-27-4-305-315 (Accessed: 19 January 2023).

Ghimire, D. (2020) *Comparative study on Python web frameworks: Flask and Django*. Metropolia University of Applied Sciences. Available at: https://www.theseus.fi/bitstream/handle/10024/339796/Ghimire_Devndra.pdf?sequence=2&isAllowed=y (Accessed: 20 June 2023).

Goldsborough, P. (2016) 'A Tour of TensorFlow', *CoRR*, abs/1610.01178. Available at: http://arxiv.org/abs/1610.01178 (Accessed: 19 January 2023).

Graf, A. and Maas, P. (2014) 'Customer value from a customer perspective – a comprehensive review', in *Service Value als Werttreiber*. Wiesbaden: Springer Fachmedien Wiesbaden, pp. 59–87. Available at: https://doi.org/10.1007/978-3-658-02140-5_3 (Accessed: 17 February 2023).

Gulli, A. and Pal, S. (2017) *Deep learning with Keras*. Packt Publishing Ltd. Available at: https://www.packtpub.com/product/deep-learning-with-keras/9781787128422 (Accessed: 19 January 2023).

Gullo, F. (2015) 'From Patterns in Data to Knowledge Discovery: What Data Mining Can Do', *Physics Procedia*, 62, pp. 18–22. Available at: https://doi.org/10.1016/j.phpro.2015.02.005 (Accessed: 19 December 2022).

Gunawardana, A., Shani, G. and Yogev, S. (2022) 'Evaluating Recommender Systems', in *Recommender Systems Handbook*. New York, NY: Springer US, pp. 547–601. Available at: https://doi.org/10.1007/978-1-0716-2197-4_15.

Hansen, P. and Hansson, H. (2015) 'Optimizing Student and Supervisor Interaction During the SciPro Thesis Process – Concepts and Design', in, pp. 245–250. Available at: https://doi.org/10.1007/978-3-319-25515-6_23 (Accessed: 17 January 2023).

Hao, J. and Ho, T.K. (2019) 'Machine Learning Made Easy: A Review of Scikit-Learn Package in Python Programming Language', *Journal of Educational and Behavioral Statistics*, 44(3), pp. 348–361. Available at: https://doi.org/10.3102/1076998619832248 (Accessed: 4 January 2023).

Hassan, A., Ahmad, A.R. and Abiddin, N.Z. (2009) 'Research Student Supervision: An Approach to Good Supervisory Practice', *The Open Education Journal*, 2(1), pp. 11–16. Available at: https://doi.org/10.2174/1874920800902010011 (Accessed: 18 December 2022).

Hug, N. (2020) 'Surprise: A Python library for recommender systems', *Journal of Open Source Software*, 5(52), p. 2174. Available at: https://doi.org/10.21105/joss.02174 (Accessed: 19 January 2023).

Huynh-Ly Thanh-Nhan, Huu-Hoa Nguyen and Thai-Nghe, N. (2016) 'Methods for building course recommendation systems', in *2016 Eighth International Conference on Knowledge and Systems Engineering (KSE)*. IEEE, pp. 163–168. Available at: https://doi.org/10.1109/KSE.2016.7758047.

Isinkaye, F.O., Folajimi, Y.O. and Ojokoh, B.A. (2015) 'Recommendation systems: Principles, methods and evaluation', *Egyptian Informatics Journal*, 16(3), pp. 261–273. Available at: https://doi.org/10.1016/j.eij.2015.06.005 (Accessed: 24 December 2023).

Jain, A. and Gupta, C. (2018) 'Fuzzy Logic in Recommender Systems', in, pp. 255–273. Available at: https://doi.org/10.1007/978-3-319-71008-2_20 (Accessed: 6 January 2023).

Jariha, P. and Jain, S.K. (2018) 'A state-of-the-art Recommender Systems: An overview on Concepts, Methodology and Challenges', in *2018 Second International Conference on Inventive Communication and Computational Technologies (ICICCT)*. IEEE, pp. 1769–1774. Available at: https://doi.org/10.1109/ICICCT.2018.8473275 (Accessed: 6 January 2023).

Jiang, L. *et al.* (2007) 'Survey of Improving K-Nearest-Neighbor for Classification', in *Fourth International Conference on Fuzzy Systems and Knowledge Discovery (FSKD 2007)*. IEEE, pp. 679–683. Available at: https://doi.org/10.1109/FSKD.2007.552 (Accessed: 13 January 2023).

Johnson, B. and Chandran, A.S. (2021) 'Comparison between Python, Java and R progrmming language in machine learning', *Int. Res. J. Modernization Eng. Technol. Sci.*, 3(6), pp. 1–6. Available at: https://www.irjmets.com/uploadedfiles/paper/volume3/issue_6_june_2021/13609/1628083530.pdf (Accessed: 18 January 2023).

Ketkar, N. and Moolayil, J. (2021) 'Introduction to PyTorch', in *Deep Learning with Python*. Berkeley, CA: Apress, pp. 27–91. Available at: https://doi.org/10.1007/978-1-4842-5364-9_2 (Accessed: 8 January 2023).

Khan, Z.Y. *et al.* (2021) 'Deep learning techniques for rating prediction: a survey of the state-of-the-art', *Artificial Intelligence Review*, 54(1), pp. 95–135. Available at: https://doi.org/10.1007/s10462-020-09892-9 (Accessed: 4 January 2023).

Kim, J. and Wilemon, D. (2002) 'Focusing the fuzzy front–end in new product development', *R&D Management*, 32(4), pp. 269–279. Available at: https://doi.org/10.1111/1467-9310.00259 (Accessed: 11 January 2023).

Koen, P.A. *et al.* (2002) 'Fuzzy Front End: Effective Methods, Tools, and Techniques', in *The PDMA toolbook 1 for new product development*. Wiley, New York, NY, pp. 5–36. Available at: https://www.wiley.com/en-ie/The+PDMA+ToolBook+1+for+New+Product+Development-p-9780471206118 (Accessed: 11 January 2023).

Koren, Y., Bell, R. and Volinsky, C. (2009) 'Matrix Factorization Techniques for Recommender Systems', *Computer*, 42(8), pp. 30–37. Available at: https://doi.org/10.1109/MC.2009.263 (Accessed: 30 December 2023).

Koren, Y., Rendle, S. and Bell, R. (2022) 'Advances in Collaborative Filtering', in *Recommender Systems Handbook*. New York, NY: Springer US, pp. 91–142. Available at: https://doi.org/10.1007/978-1-0716-2197-4_3 (Accessed: 2 January 2023).

Kramer, O. (2016) 'Scikit-Learn', in, pp. 45–53. Available at: https://doi.org/10.1007/978-3-319-33383-0_5 (Accessed: 4 January 2023).

Krogh, A. (2008) 'What are artificial neural networks?', *Nature Biotechnology*, 26(2), pp. 195–197. Available at: https://doi.org/10.1038/nbt1386 (Accessed: 14 January 2023).

Kusuma, H.S. and Musdholifah, A. (2021) 'Recommendation System for Thesis Topics Using Content-based Filtering', *IJCCS (Indonesian Journal of Computing and Cybernetics Systems)*, 15(1), p. 65. Available at: https://doi.org/10.22146/ijccs.62716 (Accessed: 25 December 2022).

Kyhnau, J. and Nielsen, C. (2015) 'Value Proposition Design: How to create products and services customers want', *Journal of Business Models*, 3(1), pp. 81–92. Available at: https://journals.aau.dk/index.php/JOBM/article/view/1105/934 (Accessed: 2 January 2023).

Lahoud, C. *et al.* (2022) 'A comparative analysis of different recommender systems for university major and career domain guidance', *Education and Information Technologies* [Preprint]. Available at: https://doi.org/10.1007/s10639-022-11541-3 (Accessed: 6 January 2023).

Lan, F. (2022) 'Research on Text Similarity Measurement Hybrid Algorithm with Term Semantic Information and TF-IDF Method', *Advances in Multimedia*, 2022, pp. 1–11. Available at: https://doi.org/10.1155/2022/7923262.

LeCun, Y., Bengio, Y. and Hinton, G. (2015) 'Deep learning', *Nature*, 521(7553), pp. 436–444. Available at: https://doi.org/10.1038/nature14539 (Accessed: 4 January 2023).

Lops, P. *et al.* (2019) 'Trends in content-based recommendation', *User Modeling and User-Adapted Interaction*, 29(2), pp. 239–249. Available at: https://doi.org/10.1007/s11257-019-09231-w (Accessed: 1 January 2023).

Mairiza, D., Zowghi, D. and Nurmuliani, N. (2010) 'An investigation into the notion of non-functional requirements', in *Proceedings of the 2010 ACM Symposium on Applied Computing*. New York, NY, USA: ACM, pp. 311–317. Available at: https://doi.org/10.1145/1774088.1774153 (Accessed: 22 January 2023).

Mardan, A. (2018) 'Real-Time Apps with WebSocket, Socket.IO, and DerbyJS', in *Practical Node.js*. Berkeley, CA: Apress, pp. 307–330. Available at: https://doi.org/10.1007/978-1-4842-3039-8_9.

Masinter, L. (2015) *Returning Values from Forms: multipart/form-data*. Available at: https://www.rfc-editor.org/rfc/rfc7578 (Accessed: 10 January 2023).

van Meteren, R. and van Someren, M. (2000) 'Using content-based filtering for recommendation', in *Proceedings of the machine learning in the new information age: MLnet/ECML2000 workshop*, pp. 47–56. Available at: http://users.ics.forth.gr/~potamias/mlnia/paper_6.pdf (Accessed: 2 January 2023).

Mongia, A. *et al.* (2020) 'Deep latent factor model for collaborative filtering', *Signal Processing*, 169, p. 107366. Available at: https://doi.org/10.1016/j.sigpro.2019.107366 (Accessed: 13 January 2023).

Nabizadeh, A.H. *et al.* (2013) 'Recommendation Systems: a review', *International Journal of Computational Engineering Research*, 3(5), pp. 47–52. Available at: https://www.researchgate.net/profile/Amir-Hossein-Nabizadeh/publication/325074466_Recommendation_Systems_a_review/links/5f3d029a299bf13404ceecba/Recommendation-Systems-a-review.pdf (Accessed: 6 January 2023).

Nagpal, A. and Gabrani, G. (2019) 'Python for Data Analytics, Scientific and Technical Applications', in *2019 Amity International Conference on Artificial Intelligence (AICAI)*. IEEE, pp. 140–145. Available at: https://doi.org/10.1109/AICAI.2019.8701341 (Accessed: 3 January 2023).

Ng, W.S. and Tan, W.W. (2021) 'Some properties of various types of matrix factorization', *ITM Web of Conferences*, 36, p. 03003. Available at: https://doi.org/10.1051/itmconf/20213603003 (Accessed: 13 January 2023).

Nguyen, G. *et al.* (2019) 'Machine Learning and Deep Learning frameworks and libraries for large-scale data mining: a survey', *Artificial Intelligence Review*, 52(1), pp. 77–124. Available at: https://doi.org/10.1007/s10462-018-09679-z.

Nurmalini, N. and Rahim, R. (2017) 'Study approach of simple additive weighting for decision support system', *Int. J. Sci. Res. Sci. Technol*, 3(3), pp. 541–544. Available at: https://www.researchgate.net/profile/Robbi-Rahim/publication/316470807_Study_Approach_of_Simple_Additive_Weighting_For_Decision_Support_System/links/590007464585156502 9f4d2a/Study-Approach-of-Simple-Additive-Weighting-For-Decision-Support-System.pdf (Accessed: 20 January 2023).

Odio, M., Sagas, M. and Kerwin, S. (2014) 'The Influence of the Internship on Students' Career Decision Making', *Sport Management Education Journal*, 8(1), pp. 46–57. Available at: https://doi.org/10.1123/smej.2013-0011 (Accessed: 17 December 2022).

Olasehinde, O. *et al.* (2022) 'Design and Implementation of a Web-Based Internship Placement Recommendation System: A Case Study of Federal Polytechnic, Ile-Oluji, Nigeria', *International Journal of Scientific & Engineering Research*, 13, pp. 398–409. Available at: https://www.researchgate.net/profile/Olayemi-Olasehinde/publication/359403105_Design_and_Implementation_of_a_Web-Based_Internship_Placement_Recommendation_System_A_Case_Study_of_Federal_Polytechnic_Ile-Oluji_Nigeria/links/623a67922708166c05437592/Design-and-Implementation-of-a-Web-Based-Internship-Placement-Recommendation-System-A-Case-Study-of-Federal-Polytechnic-Ile-Oluji-Nigeria.pdf (Accessed: 26 December 2022).

OpenAI (2023) *ChatGPT General FAQ*. Available at: https://help.openai.com/en/articles/6783457-chatgpt-general-faq (Accessed: 1 February 2023).

Paszke, A. *et al.* (2019) 'PyTorch: An Imperative Style, High-Performance Deep Learning Library', in H. Wallach et al. (eds) *Advances in Neural Information Processing Systems*. Curran Associates, Inc. Available at: https://proceedings.neurips.cc/paper/2019/file/bdbca288fee7f92f2bfa9f7012727740-Paper.pdf (Accessed: 19 January 2023).

Pazzani, M.J. and Billsus, D. (2007) 'Content-Based Recommendation Systems', in *The Adaptive Web*. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 325–341. Available at: https://doi.org/10.1007/978-3-540-72079-9_10 (Accessed: 25 December 2022).

Quinlan, J.R. (1990) 'Decision trees and decision-making', *IEEE Transactions on Systems, Man, and Cybernetics*, 20(2), pp. 339–346. Available at: https://doi.org/10.1109/21.52545 (Accessed: 13 January 2023).

Raghuwanshi, S.K. and Pateriya, R.K. (2019) 'Recommendation Systems: Techniques, Challenges, Application, and Evaluation', in, pp. 151–164. Available at: https://doi.org/10.1007/978-981-13-1595-4_12 (Accessed: 11 January 2023).

Raschka, S., Patterson, J. and Nolet, C. (2020) 'Machine Learning in Python: Main Developments and Technology Trends in Data Science, Machine Learning, and Artificial Intelligence', *Information*, 11(4), p. 193. Available at: https://doi.org/10.3390/info11040193 (Accessed: 3 January 2023).

Renuka, S., Raj Kiran, G.S.S. and Rohit, P. (2021) 'An Unsupervised Content-Based Article Recommendation System Using Natural Language Processing', in, pp. 165–180. Available at: https://doi.org/10.1007/978-981-15-8530-2_13.

Rüdiger Wirth and Jochen Hipp (2000) 'CRISP-DM: Towards a Standard Process Model for Data Mining', *Proceedings of the 4th international conference on the practical applications of knowledge discovery and data mining*, 1, pp. 29–39. Available at: http://www.cs.unibo.it/~danilo.montesi/CBD/Beatriz/10.1.1.198.5133.pdf (Accessed: 20 December 2022).

Saaty, T.L. (1988) 'What is the Analytic Hierarchy Process?', in *Mathematical Models for Decision Support*. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 109–121. Available at: https://doi.org/10.1007/978-3-642-83555-1_5 (Accessed: 16 January 2023).

Saltz, J.S. (2021) 'CRISP-DM for Data Science: Strengths, Weaknesses and Potential Next Steps', in *2021 IEEE International Conference on Big Data (Big Data)*. IEEE, pp. 2337–2344. Available at: https://doi.org/10.1109/BigData52589.2021.9671634 (Accessed: 22 December 2022).

Sánchez-Fernández, R. and Iniesta-Bonillo, M.Á. (2007) 'The concept of perceived value: a systematic review of the research', *Marketing Theory*, 7(4), pp. 427–451. Available at: https://doi.org/10.1177/1470593107083165 (Accessed: 17 February 2023).

Sarker, I.H. (2021) 'Machine Learning: Algorithms, Real-World Applications and Research Directions', *SN Computer Science*, 2(3), p. 160. Available at: https://doi.org/10.1007/s42979-021-00592-x (Accessed: 29 December 2022).

SAS (1999) *Data Mining Software, Model Development and Deployment, SAS Enterprise Miner | SAS*. Available at: https://www.sas.com/en_us/software/enterprise-miner.html (Accessed: 28 December 2022).

Shafique, U. and Qaiser, H. (2014) 'A Comparative Study of Data Mining Process Models (KDD, CRISP-DM and SEMMA) ', *International Journal of Innovation and Scientific Research*, 12(1), pp. 217–222. Available at: https://d1wqtxts1xzle7.cloudfront.net/88729354/IJISR-14-281-04-libre.pdf?1658155393=&response-content-disposition=inline%3B+filename%3DA_Comparative_Study_of_Data_Mining_Proce.pdf&Expires=1677349371&Signature=TFNTnqxu52~qhXPps61LQBk~nI5jQ4Io2JBX9mPTgjCor0n8DiZqr2c785A-XArGKtGCAwOquuLOQAUONlCWt8fHC6G4n7681w7WZ-EtGhX43BQ9EOHzl8Rpy5-dHgRu7Hkg-NsVs3M04AjWzYFoURyxwocQF0YeyI5NffMUWs2IBKJ0kg~Lx1Ewo-1qfsKJxWjaPpPNi7l~GTyWUZ6KRzkfQZgvvVGf6YP3xvgs89uLZjmsr6X2OTV~DAERWx5suO8l8SJ3RPz-q~vaYWrTLacoOAZ5lY34eU9rnXHrNuBH2uF2Qc2UGxGuboIv4jZwhv3vf5iZksLb30WZa9QEiA__&Key-Pair-Id=APKAJLOHF5GGSLRBV4ZA (Accessed: 21 December 2022).

Shalom, O.S., Roitman, H. and Kouki, P. (2022) 'Natural Language Processing for Recommender Systems', in *Recommender Systems Handbook*. New York, NY: Springer US, pp. 447–483. Available at: https://doi.org/10.1007/978-1-0716-2197-4_12.

Shani, G. and Gunawardana, A. (2011) 'Evaluating Recommendation Systems', in *Recommender Systems Handbook*. Boston, MA: Springer US, pp. 257–297. Available at: https://doi.org/10.1007/978-0-387-85820-3_8.

Shinde, P.P. and Shah, S. (2018) 'A Review of Machine Learning and Deep Learning Applications', in *2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA)*. IEEE, pp. 1–6. Available at: https://doi.org/10.1109/ICCUBEA.2018.8697857 (Accessed: 4 January 2023).

da Silva, I.N. *et al.* (2017) *Artificial Neural Networks*. Cham: Springer International Publishing. Available at: https://doi.org/10.1007/978-3-319-43162-8 (Accessed: 14 January 2023).

Srinath, K.R. (2017) 'Python–the fastest growing programming language', *International Research Journal of Engineering and Technology*, 4(12), pp. 354–357. Available at: https://d1wqtxts1xzle7.cloudfront.net/61651202/IRJET-V4I126620200101-109100-1nbuifw-libre.pdf?1577884565=&response-content-disposition=inline%3B+filename%3DPython_The_Fastest_Growing_Programming_L.pdf&Expires=1677349493&Signature=QsdWLOQnwg-xGhScCuHs5cvNw4E8KuOwFFEs2FVNjqG8zvyci9Lkt2Uk9rX8TD3nB6xBNjyS20jAzS5eifDI2sJLVKfTVHEurseA49DZ9WlcuESSteIJN5fJ2VXyT6WURWZwLH5NWZe0JY~ZHXsyO~LagpqJL9lIiYs1OX6uwlIGdEFN9aqfONOQG5ppvzoRQxaS7K3GkyvmnF-XCi46E8fsUmT5Jjjdg3JzioVo4JNYoaosfyGPb1HOO4bjuIcLZbbHwmUtgsCKuMDu~Okb-j61BC79K95m-e6YQdcXfRRpXVwsqN2f2ilk-pSEIUI5tHLILRBmb8hNLSwoAmEISQ__&Key-Pair-Id=APKAJLOHF5GGSLRBV4ZA (Accessed: 3 January 2023).

Stevens, E., Antiga, L. and Viehmann, T. (2020) *Deep learning with PyTorch*. Manning Publications. Available at: https://www.manning.com/books/deep-learning-with-pytorch (Accessed: 19 January 2023).

Sun, P. *et al.* (2022) 'Measuring Impact of Dependency Injection on Software Maintainability', *Computers*, 11(9). Available at: https://doi.org/10.3390/computers11090141.

Svelte (2023a) *Introduction Docs - Svelte*. Available at: https://svelte.dev/docs/introduction (Accessed: 23 June 2023).

Svelte (2023b) *Project Files - Svelte*. Available at: https://kit.svelte.dev/docs/project-structure#project-files-tests (Accessed: 30 June 2023).

Tarus, J.K., Niu, Z. and Mustafa, G. (2018) 'Knowledge-based recommendation: a review of ontology-based recommender systems for e-learning', *Artificial Intelligence Review*, 50(1), pp. 21–48. Available at: https://doi.org/10.1007/s10462-017-9539-5 (Accessed: 6 January 2023).

Thorp, H.H. (2023) 'ChatGPT is fun, but not an author', *Science*, 379(6630), pp. 313–313. Available at: https://doi.org/10.1126/science.adg7879 (Accessed: 10 February 2023).

Tsatsaris, E. and Sakkopoulos, E. (2021) 'Personalized Academic Thesis Management', in *2021 12th International Conference on Information, Intelligence, Systems & Applications (IISA)*. IEEE, pp. 1–8. Available at: https://doi.org/10.1109/IISA52424.2021.9555563 (Accessed: 17 January 2023).

Tuhkala, A. and Kärkkäinen, T. (2018) 'Using Slack for computer-mediated communication to support higher education students' peer interactions during Master's thesis seminar', *Education and Information Technologies*, 23(6), pp. 2379–2397. Available at: https://doi.org/10.1007/s10639-018-9722-6 (Accessed: 17 January 2023).

Vaishnavi, V., Kuechler, W. and Stacey, P. (2021) *Design Science Research in Information Systems*. Available at: http://desrist.org/desrist/content/design-science-research-in-information-systems.pdf (Accessed: 15 December 2022).

de Ville, B. (2013) 'Decision trees', *Wiley Interdisciplinary Reviews: Computational Statistics*, 5(6), pp. 448–455. Available at: https://doi.org/10.1002/wics.1278 (Accessed: 13 January 2023).

Wan, Z. *et al.* (2020) 'How does Machine Learning Change Software Development Practices?', *IEEE Transactions on Software Engineering*, pp. 1–1. Available at: https://doi.org/10.1109/TSE.2019.2937083 (Accessed: 23 February 2023).

Wu, W., He, L. and Yang, J. (2012) 'Evaluating recommender systems', in *Seventh International Conference on Digital Information Management (ICDIM 2012)*. IEEE, pp. 56–61. Available at: https://doi.org/10.1109/ICDIM.2012.6360092 (Accessed: 12 January 2023).

Yuanyuan Fan, Ana Evangelista and Hadi Harb (2021) 'An automated thesis supervisor allocation process using machine learning', *Global Journal of Engineering Education*, 23(1), pp. 20–30. Available at: http://www.wiete.com.au/journals/GJEE/Publish/vol23no1/03-Evangelista-A.pdf (Accessed: 18 December 2022).

Zhang, X., Wang, Y. and Shi, W. (2018) ‘pCAMP: Performance Comparison of Machine Learning Packages on the Edges’, in *USENIX workshop on hot topics in edge computing (HotEdge 18)*. Available at: https://www.usenix.org/system/files/conference/hotedge18/hotedge18-papers-zhang.pdf (Accessed: 19 January 2023).

Zheng, L., Noroozi, V. and Yu, P.S. (2017) ‘Joint Deep Modeling of Users and Items Using Reviews for Recommendation’, in *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*. New York, NY, USA: ACM, pp. 425–434. Available at: https://doi.org/10.1145/3018661.3018665.

Appendix A – Sequence Diagrams

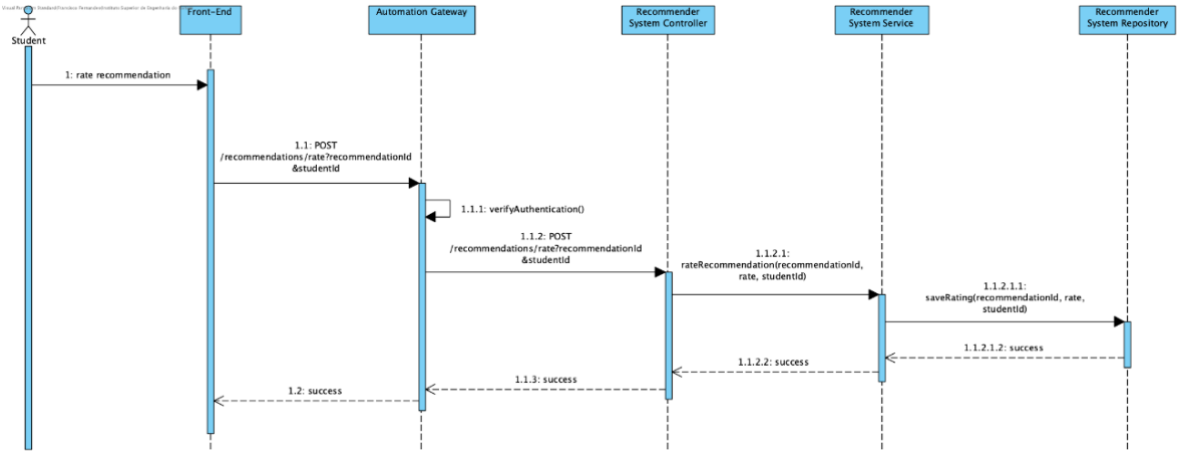


Figure A 1 - Rate Recommendations Diagram

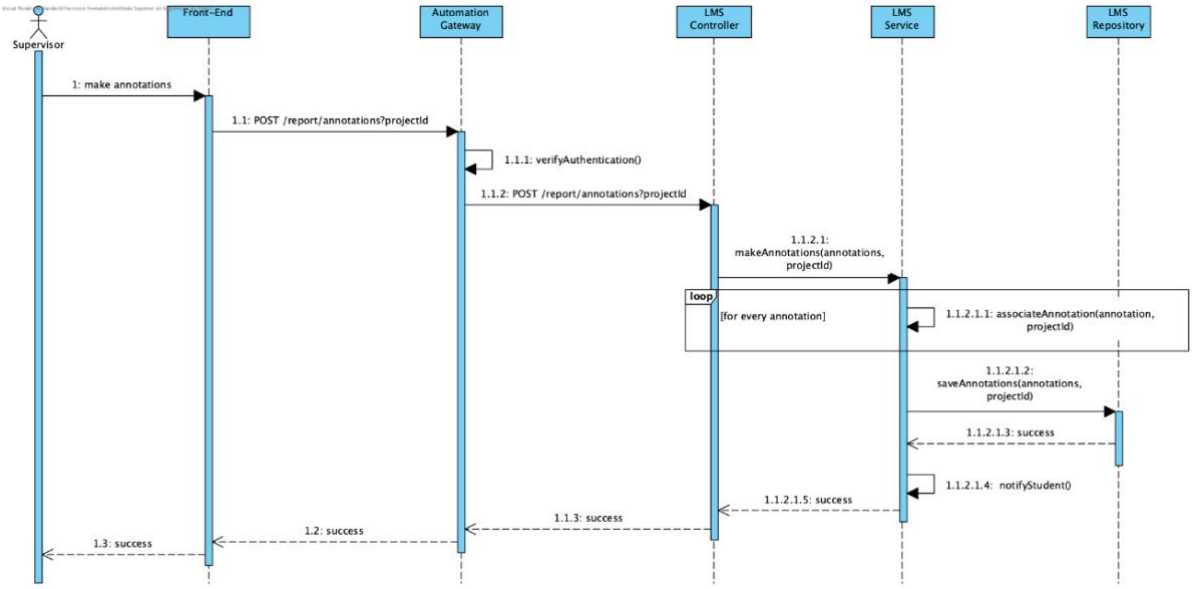


Figure A 2 - Make Annotations to the Report Diagram

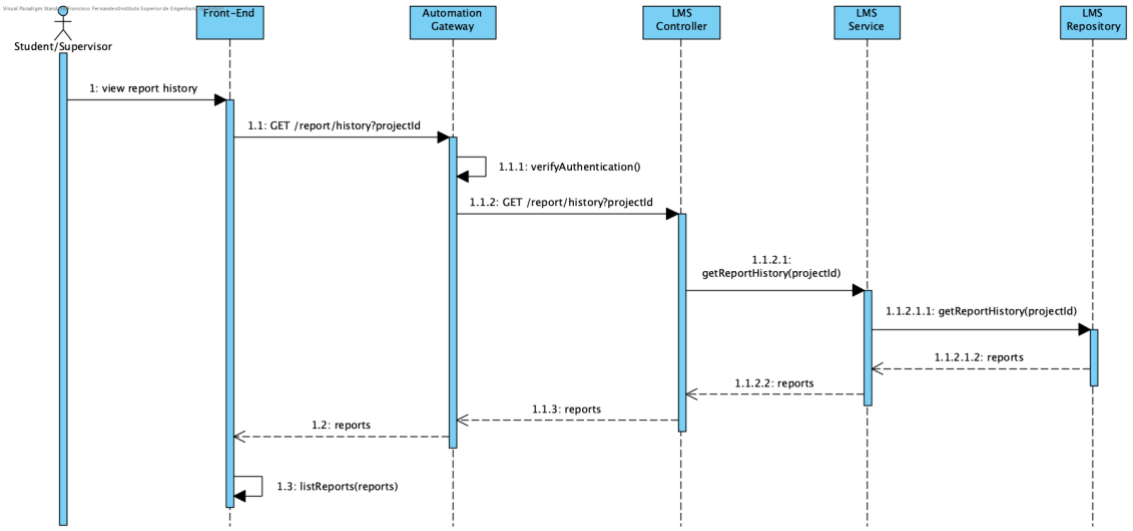


Figure A 3 - View Report History Diagram

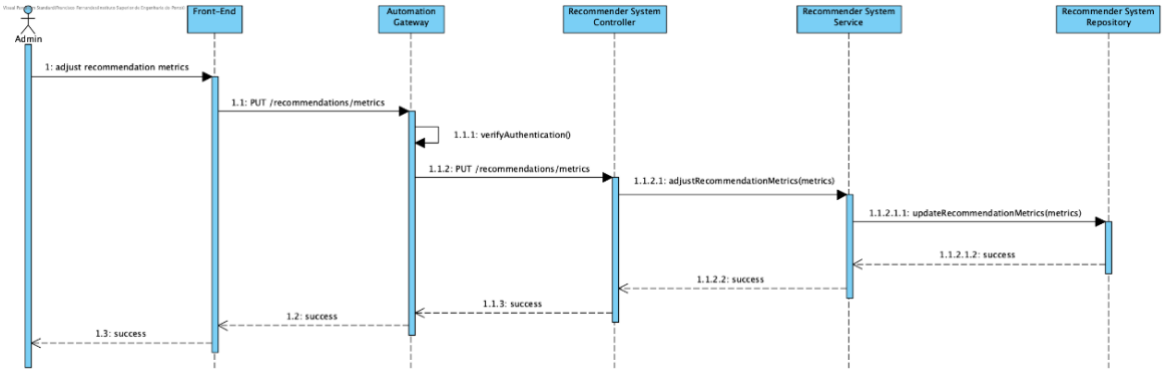


Figure A 4 - Adjust Recommendation System Metrics Diagram

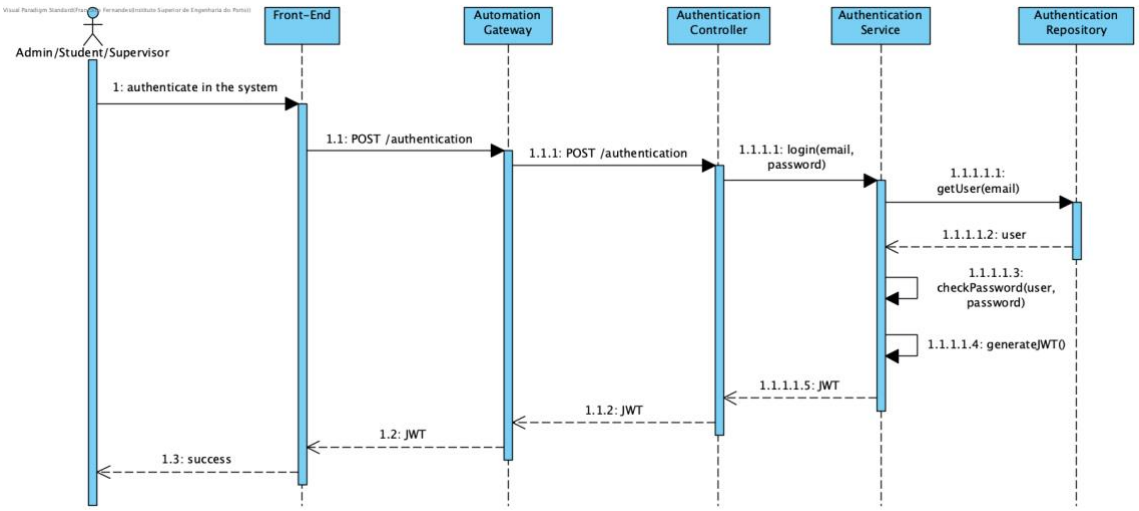


Figure A 5 - Authenticate in the System Diagram

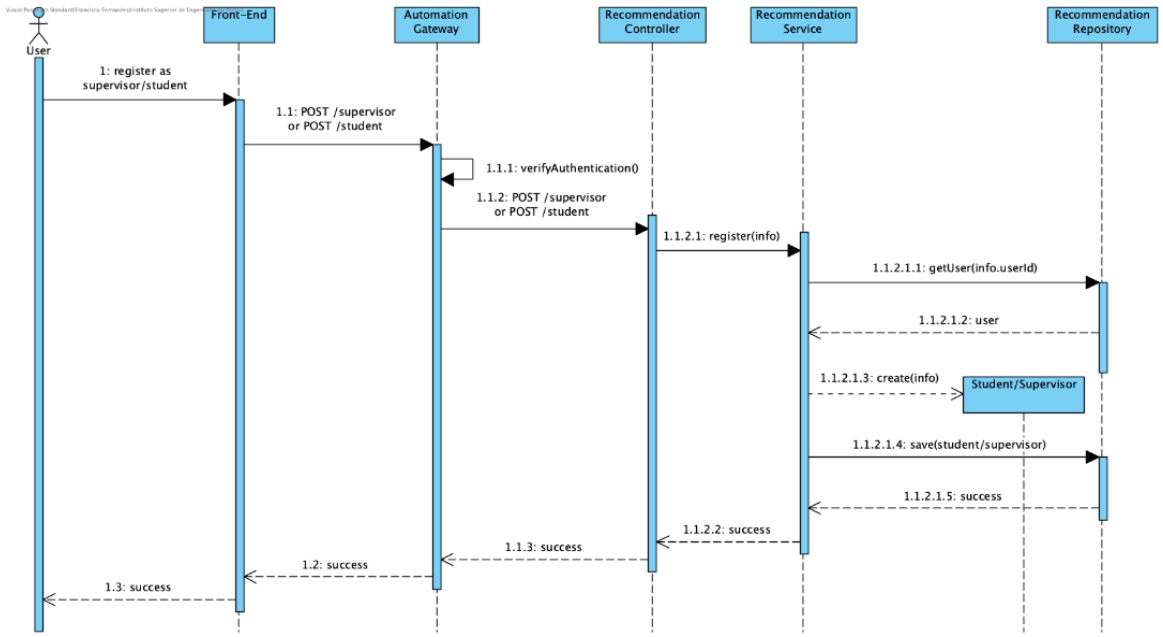


Figure A 6 - Register as a Supervisor/Student

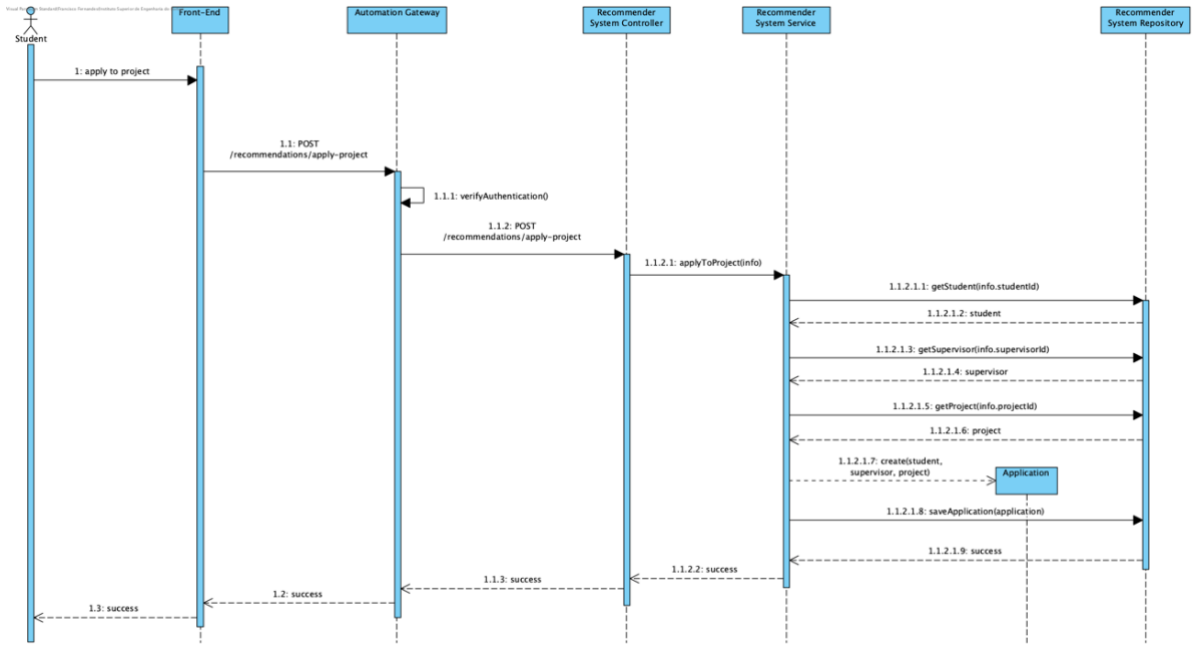


Figure A 7 - Apply to Project

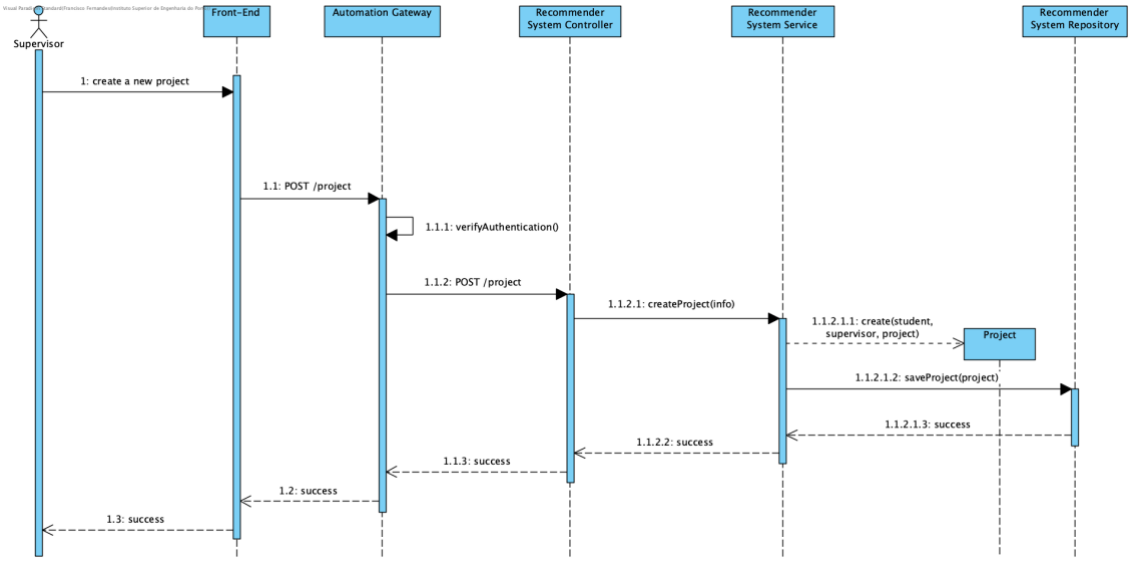


Figure A 8 - Create a new Project

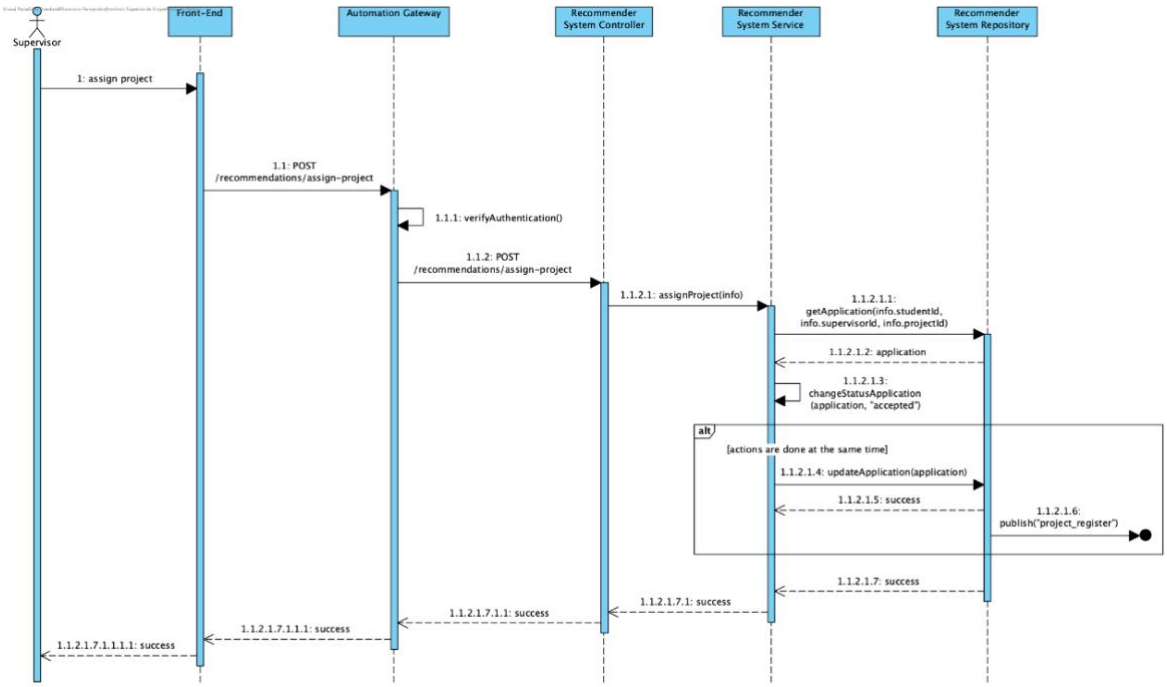


Figure A 9 - Assign Project

Appendix B – Screenshots and Code

```
library("stringr")

# Import CSV files into R

supervisors <- read.csv("./csv/supervisors.csv", header = TRUE, sep = ";")
students <- read.csv("./csv/students.csv", header = TRUE, sep = ";")
specialization_areas <- read.csv("./csv/specialization_areas.csv", header = TRUE, sep = ";")
study_areas <- read.csv("./csv/study_areas.csv", header = TRUE, sep = ";")
locations <- read.csv("./csv/locations.csv", header = TRUE, sep = ";")
projects <- read.csv("./csv/projects.csv", header = TRUE, sep = ";")

# Create CSV File for recommendations
recommendations <- data.frame(
  id = integer(),
  student_id = integer(),
  project_id = integer(),
  supervisor_ids = integer(),
  reliability_percentage = numeric(),
  stringsAsFactors = FALSE
)

id <- 1

#Loop over projects and fill recommendations
for (i in 1:nrow(projects)) {
  project_id <- projects[i,]$id
  project_qualification <- projects[i,]$qualification
  project_initial_date <- as.Date(projects[i,]$initial_date, "%d/%m/%Y")
  project_final_date <- as.Date(projects[i,]$final_date, "%d/%m/%Y")
  project_location <- projects[i,]$location_id
  project_type <- projects[i,]$project_type

  #Transform project_specialization_areas list of strings into list of integers
  project_specialization_areas <- lapply(str_split(projects[i,]$specialization_area_ids, ","), function(x) as.integer(x))[[1]]

  # Initialize list of supervisors
  list_supervisors_project <- list()

  # Search for supervisors that are interested and available for the project
  for(k in 1:nrow(supervisors)){
    supervisor_preffered_areas <- lapply(str_split(supervisors[k,]$preffered_area_ids, ","), function(x) as.integer(x))[[1]]
    supervisor_initial_date <- as.Date(supervisors[k,]$initial_date, "%d/%m/%Y")
    supervisor_final_date <- as.Date(supervisors[k,]$final_date, "%d/%m/%Y")

    if(length(intersect(project_specialization_areas, supervisor_preffered_areas)) > 0){
        if(supervisor_initial_date <= project_initial_date && supervisor_final_date >= project_final_date){
          list_supervisors_project <- c(list_supervisors_project, supervisors[k,]$id)
        }
    }
  }

  # Choose 5 random supervisors from the list
  list_supervisors_project <- sample(list_supervisors_project, min(5, length(list_supervisors_project)), replace = FALSE)

  if(length(list_supervisors_project) > 0){
    supervisor_ids <- paste(list_supervisors_project, collapse = ",")

    #Get all students that are interested in the project
    for(j in 1:nrow(students)){
      student_specialization_areas <- lapply(str_split(students[j,]$preffered_areas, ","), function(x) as.integer(x))[[1]]
      student_common_specialization_areas <- intersect(project_specialization_areas, student_specialization_areas)

      if(length(student_common_specialization_areas) > 0){
        student_qualification <- students[j,]$qualification
        if(tolower(student_qualification) == tolower(project_qualification)){
          student_project_type <- students[j,]$preffered_project_type
          student_locations <- lapply(str_split(students[j,]$preffered_locations, ","), function(x) as.integer(x))[[1]]

          # Calculate reliability percentage of the student considering the project type and the location. It must be a value between 0 and 1, and a
value with a large amounts of decimals
          reliability_percentage <- 0.0
          if(student_project_type == project_type){
            reliability_percentage <- reliability_percentage + 0.15
          }

          if(project_location %in% student_locations){
            reliability_percentage <- reliability_percentage + 0.15
          }

          reliability_percentage <- reliability_percentage + 0.7 * length(student_common_specialization_areas) / length(project_specialization_areas)
          print(project_id)
          student_id <- students[j,]$id
          recommendations <- rbind(recommendations, data.frame(id, project_id, student_id, supervisor_ids, reliability_percentage, stringsAsFactors =
FALSE))
          id <- id + 1
        }
      }
    }
  }
}

# Write recommendations to CSV file, with delimiter ";"
write.table(recommendations, file = "./csv/recommendations.csv", sep = ";", row.names = FALSE)
```

Code B 1 - R Script to Generate Recommendations (Dataset)

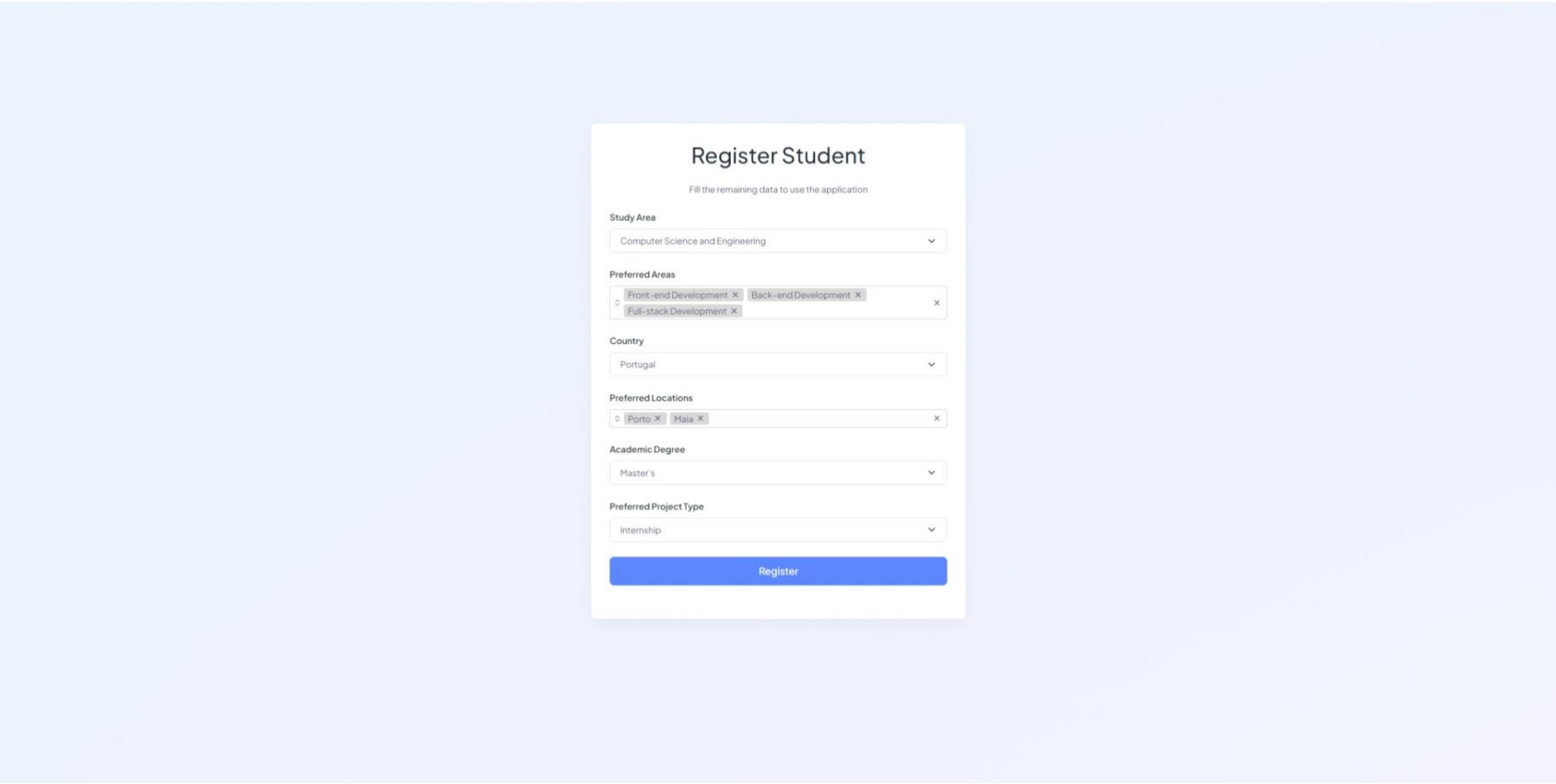


Figure B 1 - Register a New Student

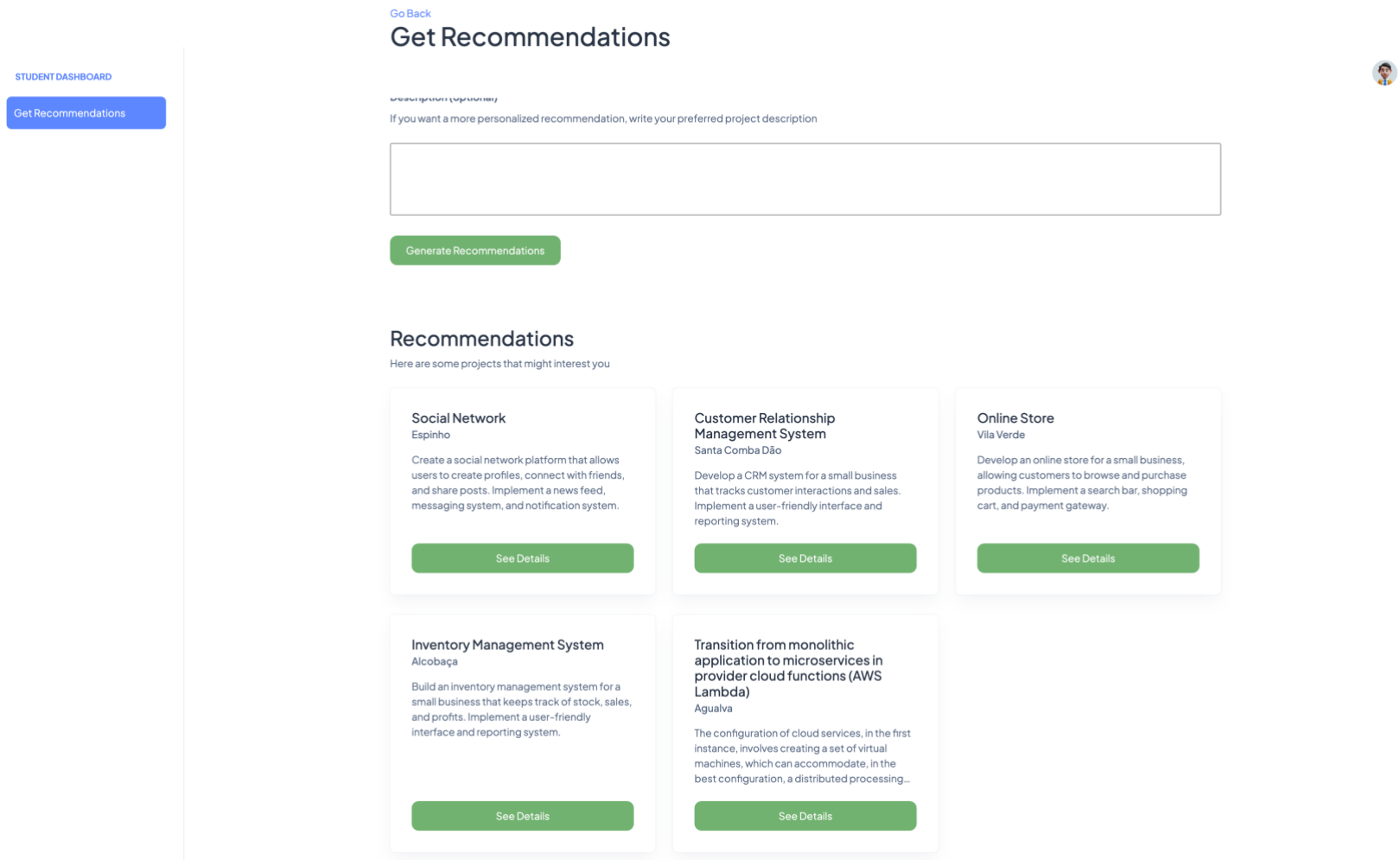


Figure B 2 - Student Recommendations

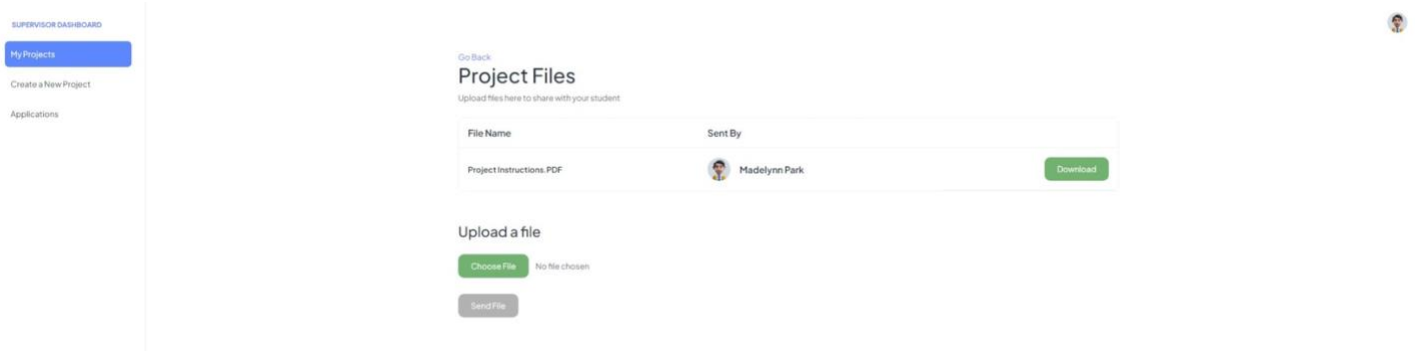
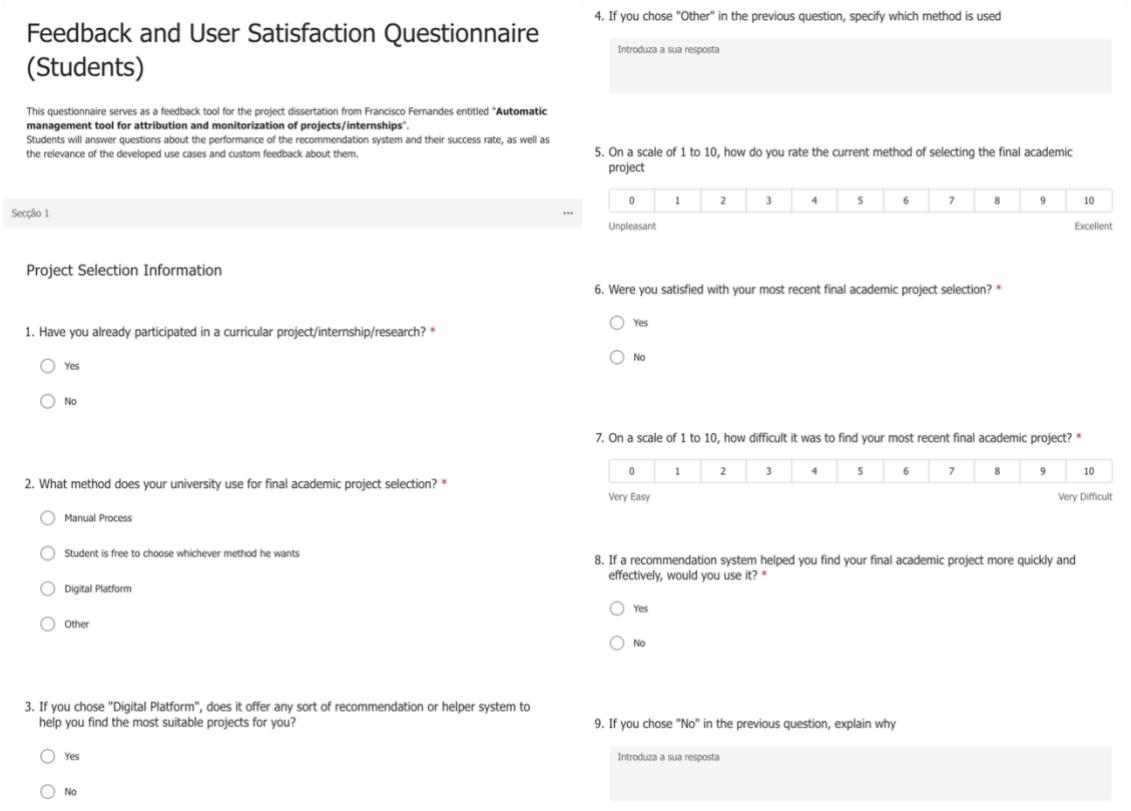


Figure B 3 - Upload Files

Figure B 4 - Chat With Student/Supervisor

Appendix C – Student Questionnaire



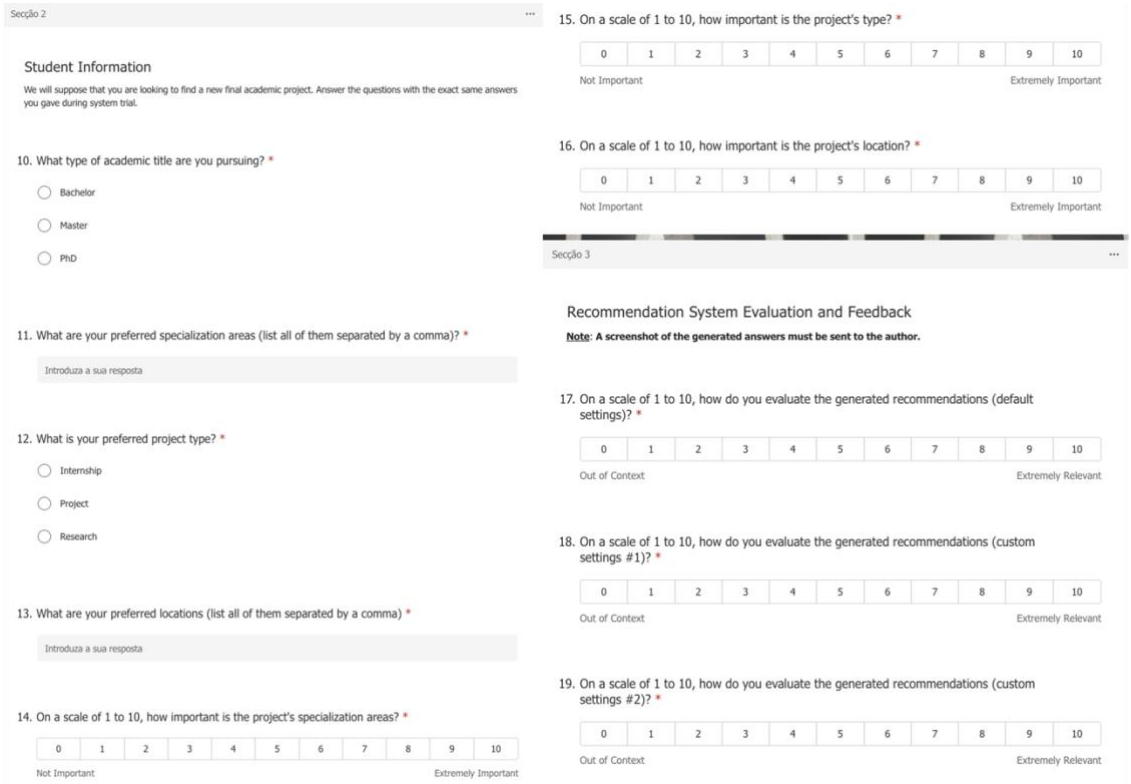
Feedback and User Satisfaction Questionnaire (Students)

This questionnaire serves as a feedback tool for the project dissertation from Francisco Fernandes entitled "**Automatic management tool for attribution and monitorization of projects/internships**".
Students will answer questions about the performance of the recommendation system and their success rate, as well as the relevance of the developed use cases and custom feedback about them.

Secção 1

Project Selection Information

1. Have you already participated in a curricular project/internship/research? *
   - Yes
   - No
2. What method does your university use for final academic project selection? *
   - Manual Process
   - Student is free to choose whichever method he wants
   - Digital Platform
   - Other
3. If you chose "Digital Platform", does it offer any sort of recommendation or helper system to help you find the most suitable projects for you?
   - Yes
   - No
4. If you chose "Other" in the previous question, specify which method is used
   Introduza a sua resposta
5. On a scale of 1 to 10, how do you rate the current method of selecting the final academic project
   0 1 2 3 4 5 6 7 8 9 10
   Unpleasant — Excellent
6. Were you satisfied with your most recent final academic project selection? *
   - Yes
   - No
7. On a scale of 1 to 10, how difficult it was to find your most recent final academic project? *
   0 1 2 3 4 5 6 7 8 9 10
   Very Easy — Very Difficult
8. If a recommendation system helped you find your final academic project more quickly and effectively, would you use it? *
   - Yes
   - No
9. If you chose "No" in the previous question, explain why
   Introduza a sua resposta



Secção 2

Student Information

We will suppose that you are looking to find a new final academic project. Answer the questions with the exact same answers you gave during system trial.

10. What type of academic title are you pursuing? *
    - Bachelor
    - Master
    - PhD
11. What are your preferred specialization areas (list all of them separated by a comma)? *
    Introduza a sua resposta
12. What is your preferred project type? *
    - Internship
    - Project
    - Research
13. What are your preferred locations (list all of them separated by a comma) *
    Introduza a sua resposta
14. On a scale of 1 to 10, how important is the project's specialization areas? *
    0 1 2 3 4 5 6 7 8 9 10
    Not Important — Extremely Important
15. On a scale of 1 to 10, how important is the project's type? *
    0 1 2 3 4 5 6 7 8 9 10
    Not Important — Extremely Important
16. On a scale of 1 to 10, how important is the project's location? *
    0 1 2 3 4 5 6 7 8 9 10
    Not Important — Extremely Important

Secção 3

Recommendation System Evaluation and Feedback

Note: **A screenshot of the generated answers must be sent to the author.**

17. On a scale of 1 to 10, how do you evaluate the generated recommendations (default settings)? *
    0 1 2 3 4 5 6 7 8 9 10
    Out of Context — Extremely Relevant
18. On a scale of 1 to 10, how do you evaluate the generated recommendations (custom settings #1)? *
    0 1 2 3 4 5 6 7 8 9 10
    Out of Context — Extremely Relevant
19. On a scale of 1 to 10, how do you evaluate the generated recommendations (custom settings #2)? *
    0 1 2 3 4 5 6 7 8 9 10
    Out of Context — Extremely Relevant

20. Did you describe your preferred project description? *

- ◯ Yes
- ◯ No

21. If you chose "Yes" in the previous section, on a scale of 1 to 10, how much it improve the recommendations? *

0	1	2	3	4	5	6	7	8	9	10

Nothing — A lot

22. Do you think this recommendation system can be useful for you and others? *

- ◯ Yes
- ◯ No

Secção 4 ...

Application Evaluation and Feedback

23. On a scale of 1 to 10, how do you evaluate the developed functionalities in terms of usefulness?

0	1	2	3	4	5	6	7	8	9	10

Not Useful — Very Useful

24. On a scale of 1 to 10, how user friendly do you consider this application?

0	1	2	3	4	5	6	7	8	9	10

Not User Friendly — Very User Friendly

25. Which functionality was the most helpful for you? *

Introduza a sua resposta

26. Do you think this system has all the necessary functionalities to be used in other universities and study areas? *

- ◯ Yes
- ◯ No

27. If your university gave you the opportunity to use a similar system, would you use it? *

- ◯ Yes
- ◯ No

28. Is there any missing functionality you think is necessary for this type of system?

Introduza a sua resposta

29. If you want to add feedback, place your answer here

Introduza a sua resposta

Appendix D – Supervisor Questionnaire

Feedback and User Satisfaction Questionnaire (Supervisors)

This questionnaire serves as a feedback tool for the project dissertation from Francisco Fernandes entitled "**Automatic management tool for attribution and monitorization of projects/internships**".
Supervisors will answer questions about the performance of the system in general as well as the relevance of the developed use cases and custom feedback about them.

Secção 1

Initial Context

1. Have you already supervised any curricular project/internship/research? *
   - ◯ Yes
   - ◯ No
2. Does your university offer any platform for keeping track of student projects? *
   - ◯ Yes
   - ◯ No
3. If you chose "No" in the previous question, do you use any other external tool for project monitoring and communication with the student?
   - ◯ Yes
   - ◯ No
4. If you chose "Yes" in the previous question, list the tools you use for project monitoring and communication with the student (list all of them separated by a comma)

   Introduza a sua resposta
5. On a scale of 1 to 10, how do you rate your current method of tracking student projects (in terms of usefulness, effectiveness, etc.)

0	1	2	3	4	5	6	7	8	9	10

   Unpleasant — Excelent
6. If a system that englobes both the project monitoring and communication with the student ever existed, would you use it? *
   - ◯ Yes
   - ◯ No

Secção 2

Supervisor Information

You will register as a supervisor and provide your personal preferences. Answer the questions with the exact same answers you gave during system trial.

7. What is your title? *
   - ◯ Bachelor
   - ◯ Master
   - ◯ PhD
8. What are your specialization areas (list all of them separated by a comma)? *

   Introduza a sua resposta
9. What are your preferred specialization areas (list all of them separated by a comma)? *

   Introduza a sua resposta

ecção 3

Application Evaluation and Feedback

10. On a scale of 1 to 10, how do you evaluate the developed functionalities in terms of usefulness? *

0	1	2	3	4	5	6	7	8	9	10

    Not Useful — Very Useful
11. On a scale of 1 to 10, how user friendly do you consider this application? *

0	1	2	3	4	5	6	7	8	9	10

    Not User Friendly — Very User Friendly
12. Which functionality was the most helpful for you? *

    Introduza a sua resposta
13. Do you think this system has all the necessary functionalities to be used in other universities and study areas? *
    - ◯ Yes
    - ◯ No
14. Do you think this system can be a substitute to all of your tools for project monitoring and communication with the student? *
    - ◯ Yes
    - ◯ No
15. If your university gave you the opportunity to use a similar system, would you use it? *
    - ◯ Yes
    - ◯ No
16. Is there any missing functionality you think is necessary for this type of system? *

    Introduza a sua resposta
17. If you want to add feedback, place your answer here

    Introduza a sua resposta

Appendix E – Quantitative Evaluation Framework

q	D	Q_i	Dimension	Q_j	W_{ij} (Factor Weight *j* in Dim *i*) [0,1]	Factor	rw_{jk} (requirement weight *k* in Factor *j*) {2, 4, 6, 8, 10}	Requirement	wf_k % requirement fulfillment *k*) [0,100]
92%	0,18	91,111	Performance	100	0,60	Concurrency	8	PC01 - The system should be able to accomodate at least 50 concurrent requests to the recommendation system	100
							6	PC02 - The system should not exceed a 20% increase in response time when concurrent requests to the recommendation system occur	100
							6	PC03 - The system's performance should remain consistent with increasing concurrency up to 100 concurrent users.	100
				77,7778	0,40	Efficiency	8	PE01 - Interacting with the overall system (with all use cases in general) should not exceed 300ms of response time	50
							10	PE02 - The system should deliver the results of the recommendation system within 5 seconds	100
		90,19	Reliability	87,069	0,40	Prediction Accuracy	10	RPA01 - RMSE of the recommendation system should be below 0.5	100
							10	RPA02 - MAE of the recommendation system should be lower than RMSE and also be below 0.5	100
							10	RPA03 - Precision of the recommendation system should be equal or higher than 0.7	50
							10	RPA04 - Recall of the recommendation system should be equal or higher than 0.7	100
							8	RPA05 - False Positive Rate of the recommendation system should not be higher than 0.3	100
							10	RPA06 - F1-Score of the recommendation system should be equal or higher than 0.7	75
				83,3333	0,20	Availability	10	RA01 - The system must remain functional if one of the microservices is offline or not working properly (some use cases cannot be affected).	100
							6	RA02 - The system must be available for everyone using a secure connection	100
							8	RA03 - The system must save periodical backups in case of system failure and corruption.	50
				96,7391	0,40	Maintainability	6	RM01 - The system is well documented	75
							10	RM02 - The system has test coverage for the most important methods of the recommendation system	100
							8	RM03 - The system has test coverage for the remaining use cases	100
							8	RM04 - The system has test coverage for the whole application flow (starting from the front-end and finishing in the back-end and database)	100
							10	RM05 - The system was developed respecting the analysis of the problem, design and architecture	100
							4	RM06 - The system is hosted in a cloud-based service that allows version control with Git	100
		87,171	Usability	89,4737	0,63	Ease of Use	10	UEU01 - The interface of the system offers good user experience	100
							8	UEU02 - All functionalities of the system can be easily accessed in the dashboard	100
							8	UEU03 - The interface of the system gives warnings and errors when something unexpected happened	100
							8	UEU04 - The system was designed to support mobile devices	100
							4	UEU05 - The system provides a "Frequently Asked Questions" section in case the user has some doubt	0
				83,3333	0,38	Accessability	8	UA01 - The system contains a font size big enough to not cause reading problems	100
							6	UA02 - The system respects the correct HTML DOM elements and their respective accessability properties	50
							4	UA03 - Some functionalities of the system can be interacted with only a keyboard	100

Appendix F – Article Abstract (DeLTA 2023)

Automatic management tool for attribution of projects/internships

Francisco Fernandes[1][0009−0008−3881−4380] and Piedade Carvalho[2]

[1] Institute of Engineering, Polytechnic of Porto, Rua Dr. António Bernardino de Almeida, 431, 4200-072 Porto, Portugal
1180964@isep.ipp.pt

[2] GILT – Games Interaction and Learning Technologies, Institute of Engineering, Polytechnic of Porto, Rua Dr. António Bernardino de Almeida, 431, 4200-072 Porto, Portugal
pbc@isep.ipp.pt

Abstract. Students at ISEP need to develop a final project in the last semester/year of their academic period, however, it is difficult for them to find the most suited project for their interests. A theory of implementing a recommendation system that takes into consideration several relevant variables of students, supervisors, and projects is studied. Recommendation systems are based on filtering techniques, such as content-based, collaborative, and hybrid filtering. Related work on the field is analysed by studying the recommendation systems' approaches of several scientific papers in separate themes such as courses, thesis topics, internships, and supervisors. Furthermore, other complete and commercial solutions are also dissected. With that information gathered, a possible solution is provided. The solution consists of a recommendation system that combines hybrid filtering techniques and natural language processing for more personalized project recommendations. In the end, an evaluation process is presented to ensure optimal prediction accuracy levels, by calculating the root mean square error, mean absolute error, and a confusion matrix.

Keywords: Recommendation System · Machine Learning · Natural Language Processing