



**NOVA**  
NOVA SCHOOL OF  
SCIENCE & TECHNOLOGY

DEPARTMENT OF  
COMPUTER SCIENCE

JOÃO MIGUEL TAVARES MARTINS

BSc

# DEVELOPMENT OF A WEB PLATFORM FOR SURGICAL ONCOLOGISTS IN PORTUGAL

MASTER IN COMPUTER SCIENCE

NOVA University Lisbon  
July, 2022



# DEVELOPMENT OF A WEB PLATFORM FOR SURGICAL ONCOLOGISTS IN PORTUGAL

**JOÃO MIGUEL TAVARES MARTINS**

BSc

**Adviser:** Rafael Costa

*Assistant Professor and Researcher, NOVA University Lisbon*

**Co-adviser:** Pedro Barahona

*Full Professor, NOVA University Lisbon*

## Examination Committee

**Chair:** João Ricardo Viegas da Costa Seco

*Full Professor, NOVA University Lisbon*

**Members:** Jorge Carlos Ferreira Rodrigues da Cruz

*Assistant Professor, NOVA University Lisbon*

Rafael Costa

*Assistant Professor and Researcher, NOVA University Lisbon*

## **Development of a Web Platform for Surgical Oncologists in Portugal**

Copyright © João Miguel Tavares Martins, NOVA School of Science and Technology, NOVA University Lisbon.

The NOVA School of Science and Technology and the NOVA University Lisbon have the right, perpetual and without geographical boundaries, to file and publish this dissertation through printed copies reproduced on paper or on digital form, or by any other means known or that may be invented, and to disseminate through scientific repositories and admit its copying and distribution for non-commercial, educational or research purposes, as long as credit is given to the author and editor.

## ACKNOWLEDGEMENTS

First of all, I would like to thank my advisor Professor Rafael Costa and co-advisor Professor Pedro Barahona for entrusting me with this dissertation and for their continued guidance, support, suggestions and patience.

I would also like to thank my college, Faculdade de Ciências e Tecnologia da Universidade Nova de Lisboa, specially Departamento de Informática, for providing all the conditions needed to achieve my goal of obtaining a BSc and now a MSc. Next, I'd like to thank colleagues Daniel Gonçalves and Leonardo Alexandre from IST for their availability and collaboration in the development of the dissertation.

Thirdly, my thanks goes to my family for their support, assistance and understanding in the final of this chapter of my academic life. Special thanks to my mother for trying to calm me down in the most difficult moments and for her wise words that gave me strength.

Last, but definitely not least, I thank my dear friends Antero Pires, Sérgio André, Luís Simões, João Oliveira, Miguel Arnaud, Soraia Cruz, André Teixeira, Duarte Saraiva and André Grossinho for their encouragement, patience, help and unconditional support on the most difficult moments of this dissertation and of course, for sharing great moments with me over this last years. A great special thanks to Antero for all his support, coaching and guidance throughout the development of most of the code for the platform.

To all of you, thank you so much for what you did for me.

## ABSTRACT

In an age of enormous access to clinical data and rapid technological development, ensuring that physicians have computational tools to navigate a sea of information and improve health outcomes is vital. A major advance in medical practice is the incorporation of Clinical Decision Support Systems (CDSSs) to assist and support the healthcare team in clinical decision making, thus improving the quality of decisions and overall patient care, while minimizing costs.

Postsurgical complications of cancer surgery are hard to predict, although there are several traditional risk scores available. However, there is an urgent need to improve peri-operative risk assessment to reduce the growing postoperative burden in the Portuguese population. Understanding the individual risks of performing surgical procedures is essential to customizing preparatory, intervention, and aftercare protocols to minimize post-surgical complications. This knowledge is essential in oncology, given the nature of the interventions, the fragile profile of patients with comorbidities and drug exposure, and the possible recurrence of cancer.

This thesis aims to develop an user-friendly web platform to support the collaboration and manage clinical data among oncologists at the Portuguese Institute of Oncology, Porto. The work integrates both a database to register/store the clinical data of cancer patients in a structured format, visualization tools and computational methods to calculate a specific risk score of postoperative outcomes for the Portuguese population. The platform named *IPOscore* will not only to manage the clinic data but also offer a predictive healthcare system, as an valuable instrument for the oncologists.

**Keywords:** Web platform, database, decision support systems, data management, risk prediction, cancer, post-surgical complications

## RESUMO

Numa época de grande acesso a dados e rápido desenvolvimento tecnológico, garantir que os médicos tenham as ferramentas de apoio à decisão clínica para se deslocar em um mar de informação para encontrar o que é mais relevante para as necessidades dos pacientes é vital para otimizar os resultados de saúde. Um grande avanço na prática médica é a incorporação de Sistemas de Apoio à Decisão Clínica (CDSSs) para auxiliar e apoiar a equipe de saúde na tomada de decisão clínica, melhorando assim a qualidade das decisões e o atendimento geral ao paciente, minimizando custos.

As complicações pós-operatórias da cirurgia do cancro ainda são difíceis de prever, embora existam muitos scores de risco destinados a fazer tais previsões. Compreender os riscos individuais de realizar procedimentos cirúrgicos é essencial para personalizar os protocolos preparatórios, de intervenção e pós-atendimento para minimizar as complicações pós-cirúrgicas. Esse conhecimento é fundamental em oncologia, dada a natureza das intervenções, o perfil frágil dos pacientes com comorbidades e exposição a drogas e a possível recorrência do cancro.

Este trabalho propõe a construção duma plataforma web de fácil utilização para apoiar a colaboração e dispor uma gestão de dados clínicos entre oncologistas. O trabalho integra uma base de dados para registrar / armazenar os dados clínicos, fisiológicos e biopatológicos de pacientes com cancro num formato estruturado e métodos computacionais para calcular um grau de risco específico de complicações pós-operatórias para a população portuguesa. A plataforma denominada *IPOscore* servirá para a gestão de dados clínicos, mas também oferecerá um sistema preditivo e preventivo, como uma ferramenta de apoio à decisão médica no contexto clínico diário.

**Palavras-chave:** Gestão de dados, previsão de risco, cancro, Base de Dados, Platform Web, complicações pós-cirúrgicas, sistema de apoio á decisão

# CONTENTS

<b>List of Figures</b>	<b>viii</b>
<b>List of Tables</b>	<b>x</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Context . . . . .	1
1.2 Objectives . . . . .	2
1.3 Contributions . . . . .	3
1.4 Document Organization . . . . .	4
<b>2 Background</b>	<b>5</b>
2.1 Cancer and Healthcare Informatics . . . . .	6
2.2 Clinical Decision Support Systems . . . . .	7
2.2.1 Traditional Risk Score Systems . . . . .	8
2.3 Predictive Machine Learning models . . . . .	9
2.4 Related Work . . . . .	10
2.4.1 ACS NSQIP Surgical Risk Calculator . . . . .	11
2.4.2 iManageCancer . . . . .	12
2.4.3 Clinical Calculators by MD Anderson Cancer Center . . . . .	13
2.4.4 Surgical Outcome Risk Tool (SORT) . . . . .	14
2.4.5 Cleveland Clinic Risk Calculator Library . . . . .	14
2.4.6 Predict tool . . . . .	15
2.4.7 Breast Cancer Recurrence Score Estimator . . . . .	15
<b>3 Methodology</b>	<b>17</b>
3.1 Overview . . . . .	18
3.2 Dataset description . . . . .	18
3.3 Database . . . . .	19
3.4 Architecture . . . . .	22
3.4.1 Web Platform Architecture . . . . .	22

3.5	Tools and Technology . . . . .	22
3.5.1	Client-Side . . . . .	23
3.5.2	Server-Side . . . . .	23
3.6	Implementation . . . . .	25
3.6.1	Web Interface and user experience . . . . .	25
3.6.2	Angular overview . . . . .	26
3.6.3	Client structure . . . . .	28
3.7	Content web server . . . . .	29
3.7.1	Main web server . . . . .	29
3.7.2	Integration Process of the modules and Secondary web servers . . . . .	30
3.8	Security . . . . .	32
3.8.1	JSON Web Token . . . . .	33
3.8.2	Implementation of the JWT . . . . .	34
<b>4</b>	<b>Results and Discussion</b>	<b>39</b>
4.1	IPOscore web-interface and modules integration . . . . .	40
4.1.1	Login and Register . . . . .	41
4.1.2	Role Management . . . . .	43
4.1.3	Search and View Data Patient . . . . .	44
4.1.4	Add New Patient . . . . .	45
4.1.5	Update Patient's Data . . . . .	46
4.1.6	Data Exploration and Feature Selection Modules . . . . .	48
4.1.7	Score Prediction Module . . . . .	52
4.2	Continuous Evaluation in the clinical context . . . . .	55
<b>5</b>	<b>Conclusions</b>	<b>56</b>
5.1	Conclusions . . . . .	57
5.2	Limitations and Future Work . . . . .	57
	<b>Bibliography</b>	<b>59</b>
	<b>Appendices</b>	
<b>A</b>	<b>Appendix - Dataset Variables</b>	<b>64</b>



## LIST OF FIGURES

1.1	Workflow of the IPOscore Web-Platform implementation. . . . .	4
3.1	Schematic representation of the database tables. . . . .	21
3.2	Overview of the platform architecture. . . . .	22
3.3	Scheme of the angular architecture (adapted from [4].) . . . . .	28
3.4	Structure of the components in the frontend . . . . .	29
3.5	Example of the implemented interaction between Python and Java programming language. . . . .	31
3.6	Signature example for the JWT. . . . .	34
3.7	User Registration and User Login processes. adapted from [51] . . . . .	35
3.8	Spring Security/JWT classes. adapted from [51] . . . . .	35
3.9	Angular front-end access with Interceptor. adapted from [51] . . . . .	37
4.1	Schematic overview of the modules integrated in the IPOscore platform. . . . .	40
4.2	Screenshot of the IPOscore main user interface. Representative image of the basic Home tab information. . . . .	41
4.3	Screenshot of the Initial State for the platform . . . . .	41
4.4	Screenshot of the Login page for the platform. . . . .	42
4.5	Screenshot of the Register new user panel in the platform. . . . .	42
4.6	Overview of all the users registered in the platform. . . . .	43
4.7	Update form for the users component. . . . .	44
4.8	Screenshot of the patients tab' initial state. . . . .	45
4.9	Representative image of a typical search and associated results table. . . . .	45
4.10	Screenshot of the initial state, showing in detail on of the pages of the form for adding a new Patient to the database. . . . .	46
4.11	Screenshot of the Final State showing details from the review process before executing the adding new Patient component. . . . .	46
4.12	Screenshot of the initial datatable with some patient's personal information . . . . .	47
4.13	Screenshot of the secondary state showing details for the patients update for personal information. . . . .	48

4.14	Screenshot of the third state showing details for the new case form . . . . .	48
4.15	Screenshot of the data exploration visualization modele - including histograms, violin charts and parallel coordinates plots. Each blue line in the parallel coordinates plot represents a patient. . . . .	49
4.16	Screenchot of the Feature Ranking module - single data type. . . . .	50
4.17	Screenchot of the Feature Ranking module - multiple data type. . . . .	51
4.18	Printscreen of the risk score module for the surgical complication outcome.	52
4.19	Printscreen of the final step in risk score module for the complication outcome. The blue dots represent all the pacients in the system and the red dots represent the patient whose input data was entered. . . . .	53
4.20	Printscreen of the risk score module for "death in months". The blue dots represent all the pacients in the system and the red dots represent the patient whose input data was entered. . . . .	53
4.21	Printscreen of the first step in the risk score module for the "ICU days" outcome.	54
4.22	Printscreen of the final step in the risk score module for the "ICU days" outcome. The blue dots represent all the pacients in the system. . . . .	54

# LIST OF TABLES

A.1 Target cohort study: input variables and output variables (star-marked).	
† Nominal values assumed to have an explicit ordering for value disclosure. . . . .	65

## LIST OF LISTINGS

3.1	example of basic metadata for the Patients' Angular component (*.css, *.spec.ts, *.html and *.ts) . . . . .	29
-----	---	----



# INTRODUCTION

The scope of the thesis is the development of an user-friendly web platform to support the collaboration and manage clinical data among oncologists at Instituto Português de Oncologia do Porto (IPO-Porto). It includes the development of both a database to register/store the clinical, physiological and biopathological data of cancer patients in a structured format and the integration of visualization tools and computational methods to calculate a specific risk score of postoperative complications for the Portuguese population.

## 1.1 Context

Cancer has a major impact on society across the whole world, being the second leading cause of death globally has accounted an estimated of 9.6 million deaths by 2018. [39] In countries with strong health systems, the survival rates for the various types of cancers are gradly improving thanks to more accessible quality treatment and early detection techniques. Meanwhile, in low-income and middle-income countries, since they're less prepared to manage this burden, a large number of cancer patients around do not have access to quality diagnosis and treatment.

The World Health Organization (WHO) provide a Cancer Country Profile [40] in order to synthesize the current status of cancer control not only for each their member state (194 in total) but fot their Regions as well (6 in total). For example, in Portugal's case, checking the profile created by WHO, by 2018 there were 58,199 total number of test cases and 28,960 total number of cancer deaths. The most occurrant types of cancer were breast cancer with 12.0%, colorectum with 17.6%, prostate with 11.4%, and lung with 9.1% incidences.

This disease strikes differently depending on the gender. Lung, prostate, colorectal, stomach and liver cancer are the most common types of cancer in men, while breast, colorectal, lung, cervical and thyroid cancer are the most common among women. With that in mind, checking the WHO's Cancer Country Profile for Portugal, for the most occurrant types of cancer the total percentage of deaths by Breast cancer was 6.0%, for Colorectum was 14.7%, for Prostate was 6.5%, and for Lung was 16.1%. With those numbers we can see that lung cancer, besides not being the most occurrant, is the type

cancer that has bigger mortality rate, with 19.6% of the cases being caused by tobacco. In this same profile it is possible to see that the estimate of the total number of lung cancer cases per year in 2018 was 5284 cases. But is also possible see check an estimate for future occurrences, where by 2040 the total number of cases per year is estimated to be 6404 [40].

Cancer treatment's goal is to achieve a cure for cancer, allowing everyone to live a normal life span. This may or may not be possible, depending on the patient specific situation [52]. If a cure isn't possible, the treatments may be used to shrink the cancer or slow the growth of it to allow the patient to live symptom free for as long as possible. Many cancer treatments are available [53]. All treatment options will depend on several factors, such as the type and stage of the cancer, general health, and the patients preferences. Together, both the doctor and the patient can weigh the benefits and risks of each cancer treatment to determine which is best for them. Cancer treatment may include options like: i) Surgery, ii) Chemotherapy, iii) Radiation therapy, iv) Targeted therapy, v) Immunotherapy and vi) Hormone therapy. Other treatments may be available to other patients, depending on their type of cancer.

Overall, the growing access to the health data (e.g. clinical, molecular and demographic) of cancer patients in national cancer institutes offer unique opportunities for understanding and anticipating individual responses to treatments including surgical procedures. However, these heterogeneous data are often dispersed on record sheets, which makes their use and integration difficult for the purposes of computational modeling in the context of precision medicine.

This thesis integrates the IPOScore project in collaboration with the Portuguese Institute of Oncology, Porto, Portugal (IPO-Porto). The general goals of the project are: 1) create a database store and (or) manage clinical, physiological and biopathological data of cancer patients in a structured format, 2) build computational approaches for clinical data exploration and visualization, 3) develop data-based models of risk prediction for postoperative outcomes in cancer patients and 4) integrate all these in a tool to support clinical decisions in the oncology domain.

In the context of this project, it was thought that a collaborative web platform would be a great solution for hosting the main components of the project. A platform that includes tools for clinical prognosis and risk stratification (through the use of Machine learning techniques) and computational methods for data analysis and visualization.

## 1.2 Objectives

The primary goal of this thesis is to develop a web-based platform to assist clinical decision-making in the surgical oncology domain. This system includes four main modules: 1) a database to store/manage the clinical data collected in a structured format, 2) visualization facilities, 3) data analysis tools, and 4) predictive models to calculate postsurgical outcomes, namely: i) existence of postoperative complications, ii) severity of complications, iii) number of days in the intermediate care unit, iv) if the patient requires intensive care unit,

v) the number of days in the intensive care unit, vi) total number of days in the hospital, vii) one-year mortality after surgery and viii) time of death after surgery. These last three modules were developed in the context of the IPOscore project by the team members.

### 1.3 Contributions

In the context of this thesis, several contributions were proposed. The following list covers the main steps.

- Development of a database for management of clinical data in standard structure
  - **Database designing**, in order to classify data and identify interrelationships. During this process is where it is determined what data must be stored and how.
  - **Database normalization**, consists in dividing larger tables into smaller tables and links them using relationships. This technique was used to eliminate redundant (useless) data and to ensure that the data is stored logically.
  - **Data preprocessing and cleaning**, was applied to get familiar with the data. It consists in preparing the data for analysis (or other process) by removing or modifying data that is corrupted or inaccurate, incorrect, incomplete, irrelevant, duplicated, or improperly formatted from the data set, table or database.
  - **Database population**, in order to import the designated dataset to fill all the tables.
  - **Data entry**, refers to the process of inputting data or information to the database.
  - **Data validation**, which refers to the process of ensuring data have undergone data cleansing to ensure they have quality. In short, to make sure they are both correct and useful.
- Web-based Online Platform implementation
  - **Different queries for the data**, in form of mapping routes to all the database tables.
  - **Data Manipulation**, through the use of *SQL commands* to retrieve/update/delete information from the database.
  - **Data Visualization**, through the implementation of views in order to show all the information stored in the database.
  - **Security**, in the form of access to the platform. Making it safe where only the admin users may have full control of the application
- Integration of data exploration, visualization and risk score modules.
  - **Data Manipulation**, retrieve/update/delete information from the database.



- **Data Visualization**, through implementation of views in order to show patient information (and more) stored in the database.
- **Security**, just like in the Web Platform’s security component, where different accesses will grant you different types of action, but tighter.

The web-platform implementation is the main contribution of this dissertation. The developed work hopes to improve healthcare delivery by enhancing medical decisions with targeted clinical knowledge, patient information, and other health information.

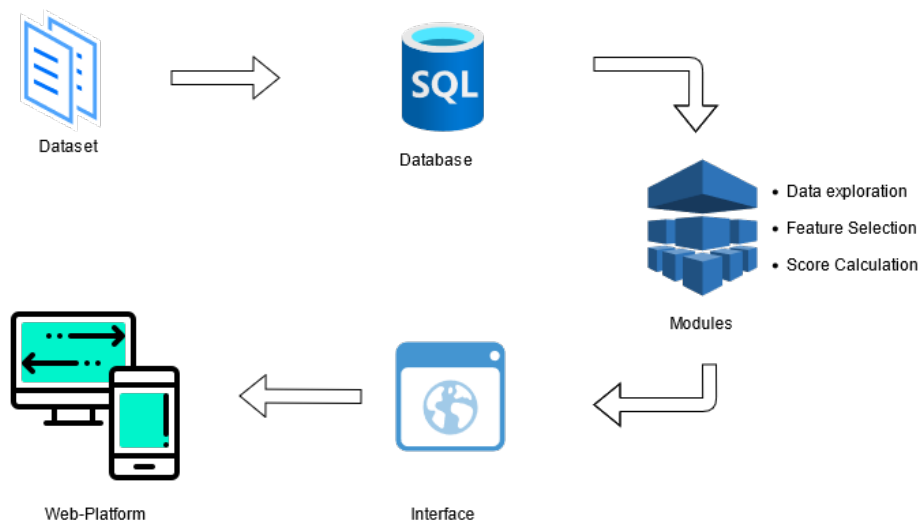


Figure 1.1: Workflow of the IPOscore Web-Platform implementation.

## 1.4 Document Organization

In addition to this chapter, the rest of the document is organized as follows:

- **Chapter 2**, addresses the related work and presents some of the major concepts and terms relating to this document.
- **Chapter 3**, details the implementation of the proposed solution, describing the experimental work done, the system’s components and how major drawbacks were overcome.
- **Chapter 4**, presents the obtained results from the implementaion process. This chapter proposes also guidelines for the use of the platform.
- **Chapter 5**, presents the final conclusions, the future work, as well as a list of limitations.

## BACKGROUND

## CONTENTS

2.1	Cancer and Healthcare Informatics . . . . .	6
2.2	Clinical Decision Support Systems . . . . .	7
2.3	Predictive Machine Learning models . . . . .	9
2.4	Related Work . . . . .	10

## 2.1 Cancer and Healthcare Informatics

For decades, cancer became one of the most deadly and devastating diseases worldwide. It is mainly caused by genetic and non-genetic changes (or a combinations of both) induced by environmental factors that prompt the activation or inactivation of specific genes leading to abnormal cell growth or neoplastic transformations [38].

Cancer' is a generic term used to describe a group of at least a hundred diseases that occur when malignant forms of abnormal cell growth develop in one or more body organs.

Certain forms of cancer result in visible growths called tumors, while others, such as leukemia, do not. Most of the body's cells have specific functions and fixed lifespans. While it may sound like a bad thing, cell death is part of a natural and beneficial phenomenon called apoptosis [47]. A cell receives instructions to die so that the body can replace it with a newer cell that functions better. Cancerous cells lack the components that instruct them to stop dividing and to die. So, as a result, they build up in the body, using oxygen and nutrients that would usually nourish other cells. Cancerous cells can form tumors, impair the immune system and cause other changes that prevent the body from functioning regularly.

With a certain patient and type of cancer, the treatment and diagnosis of it may vary significantly. For example, in solid tumours surgery is the standard option for treatment (usually in early stages of detection), that is oftenly combined with radiation therapy and systematic therapy.

An electronic health record (EHR) is a digital version of a patient's paper sheet. EHRs are real-time, patient-centered records that make information available instantly and securely to authorized users [24]. Although an EHR contains patient medical and treatment histories, an EHR system is built to go beyond the standard clinical data collected at a service provider's office and may even have a broader view of a patient's care.

In recent years, healthcare is a constant topic of discussion and health informatics has played a major role in trying to ensure everyone is provided with the best possible quality of health care [28]. Health informatics is about getting the right information in the easiest way possible. As the concept of health informatics consists of the use of technology, it is a branch of significant importance. Health informatics combines the fields of medicine, information technology, information science and technology where it is most important goal is to provide effective health care to patients.

Healthcare databases are systems in which healthcare providers regularly enter clinical and laboratory data [16]. Electronic health records (EHRs) are one of the most commonly used forms of health care databases. Practitioners enter routine clinical and laboratory data into EHRs during routine practice as a record of patient care. With this rapid growth in the use of technology in the healthcare industry, there is a large amount of data generation. Therefore, it becomes a requirement that they collect information for ease of use.

This will greatly assist health professionals in the performance of their duties. Health database organizations can play a central role in health care delivery and research, but

they will need to take important steps to ensure that private patient information remains private.

Before the 1960s, all medical records were kept on paper and in filing systems. All the diagnostics, laboratory reports, visit notes and medication instructions were written and maintained using work sheets attached to the patient's medical record. These records were labeled using only the patient's last name, the patient's last social security number digits, or some other medical record numbering system. The records were then archived and retrieved from shelves made especially for storing vertical folders [41].

Around 1971, Lockheed Corporation created a company that became known as Eclipsys Corporation featuring computerized entry of medical orders [17]. At the same time, the Veterans Administration became one of the first major health systems to fully implement a computerized patient registration system, allowing for the ordering of medications, procedures, x-rays, requests for patient assistance, special diets and laboratory tests.

## 2.2 Clinical Decision Support Systems

Patient information or data collection is a problem of the past, the main obstacle in recent years has been the necessary tools to evaluate available data.

Decision support systems help gather and analyze data, synthesizing it in order to produce comprehensive information reports[54]. In this way, as an informational application, a decision support system differs from a common operations application, whose function is only to collect data. As a computer program, decision support systems (DSS) are used to support determinations, judgments and courses of action in an organization or company. A DSS examines and analyzes large amounts of data, compiling comprehensive information that can be used to solve problems and make decisions.

Computerized clinical decision support systems, or CDSS, represent a paradigm shift in healthcare today [48]. Clinical decision support systems are quickly becoming essential tools for healthcare providers as the volume of available data increases alongside their responsibility to deliver value-based care.

Clinical decision support (CDS) provide timely information, usually at the point of care, to help inform decisions about a patient's care. CDS tools and systems help clinical teams by taking over some routine tasks, warning of potential problems, or providing suggestions for the clinical team and patient to consider[18]. They also can effectively improve patient outcomes and lead to higher-quality health care.

These type of tools are designed to help shift through enormous amounts of digital data to suggest next steps for treatments, alert providers to available information they may not have seen, or catch potential problems, such as dangerous medication interactions[1]. CDS tools are often integrated into the electronic health record (EHR) to streamline workflows and take advantage of existing data sets, although many organizations are still facing significant challenges when it comes to creating intuitive, user-friendly, and effective protocols for alarms, alerts, and decision-making pathways.[57]

Clinical Decision Support systems can be used on a variety of platforms (such as the Internet, personal computers, electronic medical record networks, handheld devices, or written materials) [25]. Planning for a new health information technology (IT) system to support electronically-based CDS includes a number of key steps, such as identifying the needs of users and what the system is expected to do, deciding whether to purchase a commercial system or build the system, designing the system for a clinic's specific needs, planning the implementation process, and determining how to evaluate how well the system has addressed the identified needs.

The utilization of health information exchanges and creation of governance standards has the potentials to aid health organizations in improving care quality [26]. Quality care provision has been hindered in many communities across the United States for many reasons; these include mismanagement, financial constraints, poor research and lack of active patient participation or contribution to care in general.

### **2.2.1 Traditional Risk Score Systems**

Developing clinical risk scores [27] and predictive models has become increasingly popular over the years but unfortunately, prediction research, specially the development of prediction models or clinical risk scores, has proven to be quite prone to error. None the less, an accurate risk stratification would facilitate informed patient consent and identify those individuals who may benefit from specific perioperative interventions. The ideal clinical risk scoring system would be objective, accurate, economical, simple to perform, based entirely on information available preoperatively, and suitable for patients undergoing both elective and emergency surgery.

In the beginning, only two indicators formed the basis when calculating a risk score: age and gender. Nowadays, newer models still use these two demographic details but also include many other additional indicators [19].

When selecting a method for risk scoring, several aspects of a model need to be considered: the risk factors or indicators used to make the prediction, underlying data integrity, methodology preference, and resource capabilities.

Traditionally, the most commonly used additional indicators are diagnostic information and disease status and this information can be usually found in medical records, claims, hospital discharges or even in prescription drugs data. This is one of the reasons why data is very important. Standardized and accurate data are helpful for the creation of these types of predictive models in order to risk score successfully.

For example, the ARISCAT Score [11] was developed to predict the risk of in-hospital postoperative pulmonary complications - as defined as the occurrence of respiratory failure, respiratory infection, pleural effusion, atelectasis, pneumothorax, bronchospasm or aspiration pneumonitis - after surgery with general, neuraxial or regional anesthesia.

The POSSUM scale [14] (Physiological and Operative Severity Score for the enUmeration of Mortality and Morbidity) is a scoring system that is used to predict risk-adjusted

mortality and morbidity rates in a wide variety of surgical procedures. In this prospective study, the validity of the POSSUM scale was evaluated in patients undergoing laparotomy in a general surgery department. POSSUM and a modified version P-POSSUM have been evaluated in various groups of surgical patients for the accuracy of predicting mortality.

Risk scoring is beneficial both for its predictive capabilities and its application to population health management but, selecting the proper risk scoring model for a specific population can be complicated [19]. To find a model that is the best fit for an organization goals, first it is needed to establish a set of indicators that will be used to define an individual's risk. Then, ensure data quality and standardization techniques followed by creating a meaningful methodological approach that best fits the goals of risk scoring. Finally, consider an organization's resource capabilities and set realistic expectations of what can be achieved.

### **2.3 Predictive Machine Learning models**

In the past decades, a continuous evolution related to cancer research has been performed. Scientists have applied different methods, such as screening in early stage, in order to find types of cancer before they cause symptoms [29]. Moreover, they have developed new strategies for the early prediction of cancer treatment outcome. With the advent of new technologies in the field of medicine, large amounts of cancer data have been collected and are available to the medical research community.

However, the accurate prediction of a disease outcome is one of the most interesting and challenging tasks for physicians. As a result, Machine Learning (ML) methods have become a popular tool for medical researchers. These techniques can discover and identify patterns and relationships between them, from complex datasets, while they are able to effectively predict future outcomes of a cancer type.

Given the significance of personalized medicine and the growing trend on the application of ML techniques, we here present a review of studies that make use of these methods regarding the cancer prediction and prognosis. In these studies prognostic and predictive features are considered which may be independent of a certain treatment or are integrated in order to guide therapy for cancer patients, respectively [43].

As branch of Artificial Intelligence, Machine learning relates the problem of learning from data samples to the general concept of inference [8]. Every learning process consists of two phases: (i) estimation of unknown dependencies in a system from a given dataset and (ii) use of estimated dependencies to predict new outputs of the system. ML also has been proven to be an interesting area in biomedical research with many applications, where an acceptable generalization is obtained by searching through an n-dimensional space for a given set of biological samples, using different techniques and algorithms.

There are two main common types of ML methods known: (i) supervised learning and (ii) unsupervised learning [23]. In supervised learning a labeled set of training data is used to estimate or map the input data to the desired output. In contrast, under the

unsupervised learning methods no labeled examples are provided and there is no notion of the output during the learning process. As a result, it is up to the learning scheme/model to find patterns or discover the groups of the input data. In supervised learning this procedure can be thought as a classification problem. The classification task refers to a learning process that categorizes the data into a set of finite classes. Two other common ML tasks are regression and clustering. In the case of regression problems, a learning function maps the data into a real-value variable. Subsequently, for each new sample the value of a predictive variable can be estimated, based on this process. Clustering is a common unsupervised task in which one tries to find the categories or clusters in order to describe the data items. Based on this process each new sample can be assigned to one of the identified clusters concerning the similar characteristics that they share.

When applying a ML method, data samples constitute the basic components. Every sample is described with several features and every feature consists of different types of values [29]. Furthermore, knowing in advance the specific type of data being used allows the right selection of tools and techniques that can be used for their analysis. Some data-related issues refer to the quality of the data and the preprocessing steps to make them more suitable for ML. Data quality issues include the presence of noise, outliers, missing or duplicate data and data that is biased-unrepresentative. When improving the data quality, typically the quality of the resulting analysis is also improved. In addition, in order to make the raw data more suitable for further analysis, preprocessing steps should be applied that focus on the modification of the data. A number of different techniques and strategies exist, relevant to data preprocessing that focus on modifying the data for better fitting in a specific ML method. Among these techniques some of the most important approaches include (i) dimensionality reduction (ii) feature selection and (iii) feature extraction. There are many benefits regarding the dimensionality reduction when the datasets have a large number of features. ML algorithms work better when the dimensionality is lower. Additionally, the reduction of dimensionality can eliminate irrelevant features, reduce noise and can produce more robust learning models due to the involvement of fewer features. In general, the dimensionality reduction by selecting new features which are a subset of the old ones is known as feature selection. Three main approaches exist for feature selection namely embedded, filter and wrapper approaches. In the case of feature extraction, a new set of features can be created from the initial set that captures all the significant information in a dataset. The creation of new sets of features allows for gathering the described benefits of dimensionality reduction.

## **2.4 Related Work**

In this section, the focus is the available web-based platforms and risk calculators in the cancer domain and/or surgical complications.

### 2.4.1 ACS NSQIP Surgical Risk Calculator

In 1994, the National Surgical Quality Improvement Program (NSQIP) [9] was created out of a need for the Department of Veterans Affairs (VA) to monitor and improve the quality of surgical care across all VA medical centers, since their previous program (NVASRS - National VA Surgical Risk Study) was a great success.

The ACS (American College of Surgeons) NSQIP surgical risk calculator is a web-based decision aid and informed consent tool (based on reliable multi-institutional clinical data) that helps provide customized risks estimates for patients (and doctors) who are preparing for surgery. ACS NSQIP data improves a hospital's ability to focus on preventable complications. Because it was developed by surgeons who understand the reality of the operating room, this tool helps hundreds of hospitals across the United States to assess the quality of their surgical programs with unparalleled accuracy and measurably improve surgical results.

This surgical risk calculator, uses about 20 patient variables (like age, ASA class, height, weight) and the planned procedure (CPT code) to help predict the chance that patients will have any of 18 different outcomes possible within 30 days after the given intervention. The outcomes include:

- Serious complication (cardiac arrest, myocardial infarction, pneumonia, progressive renal insufficiency, acute renal failure, PE, DVT, return to the operating room, deep incisional SSI, organ space SSI, systemic sepsis, unplanned intubation, UTI, wound disruption)
- Any complication (superficial incisional SSI, deep incisional SSI, organ space SSI, wound disruption, pneumonia, unplanned intubation, PE, ventilator > 48 hours, progressive renal insufficiency, acute renal failure, UTI, stroke, cardiac arrest, myocardial infarction, DVT, return to the operating room, systemic sepsis)
- Pneumonia
- Cardiac Complication (cardiac arrest or MI)
- Surgical Site Infection (SSI)
- Urinary Tract Infection (UTI)
- Venous Thromboembolism (VTE)
- Renal Failure (progressive renal insufficiency or acute renal failure)
- Colon Ileus (Conditionally displayed based on the selected Procedure)
- Colon Anastomotic Leak (Conditionally displayed based on the selected Procedure)
- Readmission
- Return to OR
- Death



- Discharge to Nursing or Rehab Facility
- Predicted Length of Hospital Stay
- T Vascular LEO Wound
- T Pancreatectomy Delayed Gastric
- T Proctectomy Ileus

This risk score was built with data collected from over 4.3 million operations (procedures or interventions) provided by 780 American hospitals that participate in the NSQIP from 2013 to 2017. As long as the information entered is the most complete and accurate, the risk calculator will provide the most precise risk information. However, the estimate can still be calculated even if some of the patient information is missing or unknown.

#### **2.4.2 iManageCancer**

The *iManageCancer* project [50] is an optimal and specific approach to a clinical decision support system and to patient guidance in the cancer domain.

This project is a research collaboration composed of nine partners with clinical, industrial and academic background from five European countries (Germany, Greece, Netherlands, Italy and The UK) with the purpose of facilitating efficient management and self-management of cancer according to available clinical knowledge, patient data and the healthcare delivery model, in order to support patients and their healthcare providers in making informed decisions on the treatment choices and in managing side effects of their cancer therapy.

*iManageCancer* provides cancer patients and their physicians with a comprehensive platform of interconnected mobile tools to empower patients and to support them in the management of their disease in collaboration with their doctors. The back end of the *iManageCancer* platform contains an intelligent personal health record (iPHR) application and also comprises the central decision support unit (CDSU). The CDSU offers dedicated disease management services to the end users in combination with the apps *iManageMyHealth* and *iSupportMyPatients*.

The main purpose of this CDSS is that the domain experts design care flow diagrams, workflows for formal disease management (based on clinical guidelines), knowledge on care pathways and an organisational model for integrated healthcare with the patient being the co-manager of his health in the centre of it. The term '*care flow*' here is used in the sense that the care flow diagrams describe the management of certain aspects of the disease of the patient in an outpatient setting.

In this case, the care flow is represented as a complex formal process diagram with integrated predictive models from a model repository framework (MRF). These diagrams are personalised for a specific patient in individual care flow plans and executed by the CDSU of the *iManageCancer* platform, in the so-called *Care Flow Engine* (CFE). This engine guides the patient but, optionally, also the healthcare team through the management

of the disease and related co-morbidities. The engine does this by issuing tasks and recommendations to the patient and to the different members of the healthcare team, and by controlling the execution of the care flow plan based on the results of tasks and monitored health status of the patient.

Both patients and their physicians, who are registered on the iManageCancer platform, can utilize this management services with the help of two individual apps.

The iManageMyHealth app for the patient downloads and processes the individual Care Flow tasks issued for himself or herself by the CFE. The tasks given for patients are typically health enquiries and measurement tasks but also information providing tasks with the recommendations obtained from the CFE. Then, the results are sent back to the CFE for further assessment and control. Similarly, the iSupportMyPatients app for physicians downloads and processes tasks for the patient's physician. The both apps receive also notifications from the CFE about tasks assigned to health professionals or patients.

The iSupportMyPatients app for physicians, receives the list of available care flows. Then, it visualises the tasks from the CFE for all patients who enabled data sharing in their intelligent personal health record (iPHR) with their respective physician. The physician then can observe in the list of his patients the number of currently pending tasks that he shall execute. After selecting a patient or his pending tasks, the physician is forwarded to a page for managing care flows and tasks.

The *iManageCancer* platform with its various tools is currently being evaluated in two clinical pilots with adult and paediatric cancer patients in Italy and Germany. With this CDSU, a powerful clinical decision support framework has been provided and integrated in the *iManageCancer* platform to support patients and their physicians in the management of their cancer. These implemented management services offered by the framework focus on chemotherapies and their side effects. In the mean time, this dynamic nature of the system raises new challenges from the regulatory perspective. It requires a thorough risk analysis before new models and care flows are deployed in the system. Then the models need to be validated before they can be used as part of a care flow diagram in the clinical routine. As it goes without saying, this framework is also applicable to other chronic diseases and their mobile management.

### **2.4.3 Clinical Calculators by MD Anderson Cancer Center**

The University of Texas MD Anderson Cancer Center [56] is one of the world's most respected centers devoted exclusively to cancer patient care, research, education and prevention. It is the largest cancer center in the U.S., also being one of the original three comprehensive cancer centers in the country and, as all cancer centers, they strive to the help eliminate cancer through programs that integrate patient care, research and prevention.

MD Anderson offers clinical tools and resources, like clinical calculators or clinical

practice algorithms, for physicians to accompany the patients through their process, or disease, and make informed decisions about cancer treatments. The calculators were created by their faculty members in conjunction with peer-reviewed journals to predict treatment outcomes, survival and response to specific cancer treatments.

These clinical calculators focus on 4 different types of cancer: (i) breast, (ii) colorectal, (iii) esophageal and (iv) pancreatic. Amongst these categories, some of the calculators try to help predict further complications prior to surgery or treatment. For example, the Chemotherapy Response Calculator for Breast Cancer or Disease-free survival after breast surgery [55] helps, as the name refers, to calculate the probability of disease-free survival after breast cancer. This takes in consideration variables like the histologic type or the histologic grade of the cancer or even some surgical data like the pathologic tumor size, to give a probability of an outcome of the patient being disease free up to 5 or 10 years.

#### **2.4.4 Surgical Outcome Risk Tool (SORT)**

The SORT [44] is a risk prediction tool that has been developed and validated in a UK population (and now also New Zealand and Australia) of patients undergoing inpatient non-cardiac non-neurological surgery, which estimates the risk of death within 30 days of inpatient surgery.

It has a number of potential advantages over existing risk prediction tools, including parsimony (few variables) that combined with clinicians' assessment ends up being both excellent in terms of discrimination and also well-calibrated. The fact that all data should also be available pre-operatively and that because no blood or radiological results are required, it should be possible for family doctors and preoperative assessment teams to calculate it, before the results of preoperative investigations may be available.

#### **2.4.5 Cleveland Clinic Risk Calculator Library**

A committee of national experts, led by a Cleveland Clinic [13] researcher, has established first-of-its-kind guidelines to promote more accurate and individualized cancer predictions, guiding more precise treatment and leading to improved patient survival rates and outcomes.

These new guidelines are changing the traditional approach of cancer staging methods for cancer treatment. The new risk calculators – which will complement the existing staging system – will enable physicians to more accurately and precisely determine the best treatment for individual patients.

The American Joint Committee on Cancer (AJCC), which is responsible for periodically evaluating and updating the cancer stages, invited a group of top healthcare statistical experts from across the country to form the Precision Medicine Core (PMC). They acknowledged that cancer stages are imperfect, and it is committed to enhancing the system with more prognostic, statistically based risk calculators.

This group also discussed characteristics necessary for developing a quality risk model in

cancer patients. The emphasis centered on performance metrics, implementation clarity, and clinical relevance.

The following cancers will be the first to be evaluated for existing prediction models: breast, colon, prostate, lung, melanoma, and head and neck cancer, with the goal being to comprehensively include all cancers in the future. The formulas must predict overall survival or death from a particular type of cancer and have to pass all 16 criteria.

#### **2.4.6 Predict tool**

*Predict* is a tool developed by a collaboration between the Cambridge Breast Unit from the Department of Oncology of the University of Cambridge and the ECRIC (UK's Eastern Cancer Information and Registration Centre) that helps show how breast cancer treatments, after surgery, might improve survival rates [10]. Once the details about the patient and their cancer have been entered, the tool will show how different treatments would be expected to improve survival rates up to 15 years after diagnosis.

The tool applies to women who have undergone early breast cancer surgery and is different from the other treatments they must undergo.

*Predict's* data, on which it is based, did not include information on the presence of bilateral disease. The performance of the model in women with bilateral breast cancer is unknown. The benefits of applied chemotherapy were taken from the Early Breast Cancer Trialist Collaborative Group meta-analysis of adjuvant chemotherapy trials.

*Predict* has been tested to ensure that its produced estimates are as accurate as current knowledge can give. *Predict* was originally developed using data from more than 5,000 women with breast cancer. Their predictions were then tested with data from 23,000 other women around the world to make sure they gave the best possible estimate.

This tool only asks for certain information about cancer. The inputs you request are those for which we have enough information to predict how they affect survival rates. It does not differentiate between types of surgery (for example, mastectomy or lumpectomy) nor does it ask about lifestyle factors such as smoking or exercise. These will affect survival, but at the moment we cannot say how much.

Even though it produces good estimates, *Predict* can't say whether an individual patient will survive or not. It can only provide the average survival rate for people in the past of a similar age and with similar cancers.

#### **2.4.7 Breast Cancer Recurrence Score Estimator**

Researchers led by scientists at the Johns Hopkins Kimmel Cancer Center claim to have developed a free web app that could take some of the guesswork out of decisions to order an additional and expensive molecular test to assess the risk of recurrence in women with early-stage breast cancer.

The *Breast Cancer Recurrence Score Estimator* is designed to help physicians decide when the oncotypeDX® test (test developed for patients in early stages of breast cancer) is likely

to offer information that complements, rather than replicates, what is already available from high quality pathology information [12].

The scientists at Johns Hopkins cancer center say that determining a patient's risk for recurrence is a major factor in determining the need for chemotherapy and anti-hormone medications after surgery to remove an early-stage tumor.

The investigators also looked at what would happen if doctors at Johns Hopkins used this app, rather than their own intuition, to decide when to request the molecular test. They said that their analysis in 939 patients suggested that by using the app, they would have ordered 297 tests, instead of the 299 tests actually ordered.

The app is based on extracted information from medical records of patients (1113 patients, precisely) treated at five different hospitals in the United States, including the Johns Hopkins Hospital, for stage 1 or 2 breast cancer.

In this dissertation a web-based platform was developed in order to help oncology doctors manage cancer patients data in a standard format. Throughout the following chapters, its architecture, components and implementation are detailed.

## METHODOLOGY

## CONTENTS

3.1	Overview . . . . .	18
3.2	Dataset description . . . . .	18
3.3	Database . . . . .	19
3.4	Architecture . . . . .	22
3.5	Tools and Technology . . . . .	22
3.6	Implementation . . . . .	25
3.7	Content web server . . . . .	29
3.8	Security . . . . .	32

In this dissertation a web-based platform was developed in order to help oncology doctors. This chapter explains in detail the proposed solution and its implementation. There is a section that describes the work done before the implementation, that led to the development of the database and platform prototypes and finally how it all was integrated.

### 3.1 Overview

Imagine the case where a doctor is at his place of work, jointly or individually, and wants to find out information about a patient that he's treating. Instead of having to search through countless record sheets, if the latter had access to a platform on the computer or via smartphone to consult the patient's data it would facilitate his research, saving costs and time for future treatments, for the patient and for him. This is where the *IPOScore* project comes in.

Through the access of the *IPOScore* platform, the doctor would be able to view all the patient's information and could even compare it's case with other similar cases. Based on some characteristics of this patient, he can also calculate a risk score for postsurgical outcomes and identify discriminative surgical risk patterns to be able to classify the high risk patient for surgical complications.

This thesis is inserted in the context of the *IPOScore* project. The *IPO-Porto* provided a anonymized dataset with a prospective cohort of cancer patients who undertook surgery and that wore treated at the center, with and without post-surgical complications. The proposed solution consists of two main components:

1. Database to store and (or) manage clinical data in standard format.
2. Web-based online platform that integrates all previous modules ("Data Exploration", "Feature selection" and "Risk Score") developed in the context of the *IPOscore* project.

### 3.2 Dataset description

The dataset provided by *IPO-Porto* fits into the class of multivariable data, where each variable, monitored (between 2016 to 2018), has a different distribution.

The cohort contains 847 patients with 138 variables (33 binary, 45 nominal, 35 numerical, 8 ordinal, 4 date and 13 free-text). Moreover, the data contains information pertaining to the demographic and physiological patient characteristics, cancer location and histopathological determinants, surgical procedures, post-surgical outcomes and risk scores variables from ARISCAT, P-POSSUM, ACS NSQIP, and Charlson Index. The *IPO-Porto* Ethics Committee approved the analysis of the anonymized data (CES IPO:91/019).

The patients included in this study were selected because they had co-morbidities or because the surgery to be performed was complex, which advocated that the immediate postoperative be monitored in the HDU. The current variable distribution in the dataset can be grouped in order to create different tables.

The postsurgical complication is defined as a deviation from the ideal postoperative course, which is deemed clinically connected to the surgery prior, requiring any intervention, and happening within the first 90 days after the surgery.

See the full list of variables in the dataset at [Appendix A](#).

### 3.3 Database

Before data can be processed, stored and analyzed, it needs to be prepared, so it can be read by algorithms. Raw data needs to undergo ETL - extract, transform, load - to get to the data warehouse for processing.

Data processing occurs when data is collected and translated into usable information. Usually performed by a data scientist or team of data scientists, it's important that the data processing is done correctly as not to have a negatively affect to the data output. Also, data processing starts with data in its raw form and converts it into a more readable format (graphs, documents, etc.), giving it the form and context necessary to be interpreted by computers and utilized by employees throughout an organization.

For the construction of the database, the dataset first underwent through data processing which includes several stages like data collection (previously made by IPO-Porto experts), data preparation, data input, data storage, etc. Throughout these stages, the data was cleaned, processed, interpreted and finally stored for usage. Properly stored data is a necessity for compliance with data protection legislation and after that, it can be quickly and easily accessed by members of the organization when needed.

From the analysis made to the provided dataset it was clear that the variables already came grouped together. With those agrupations, eleven different tables were created:

1. Application for UCI Admission
2. Patients
3. Internment characteristics
4. Surgery Data
5. P-POSSUM
6. ACS Risk Calculator
7. ARISCAT
8. Post Surgical complications
9. Hospital Discharge
10. Preoperative Comorbidities
11. CHARLSON

The dataset provided by IPO-Porto, and described in section [3.2](#), came in a Excel format. Here, all the eleven tables above are used, managed and stored in a MySQL [\[37\]](#) database



## CONTENTS

---

with all the patients' data. It had all the variables structured and categorized in a way that it can help understand what each variable represents (see Figure 3.1).

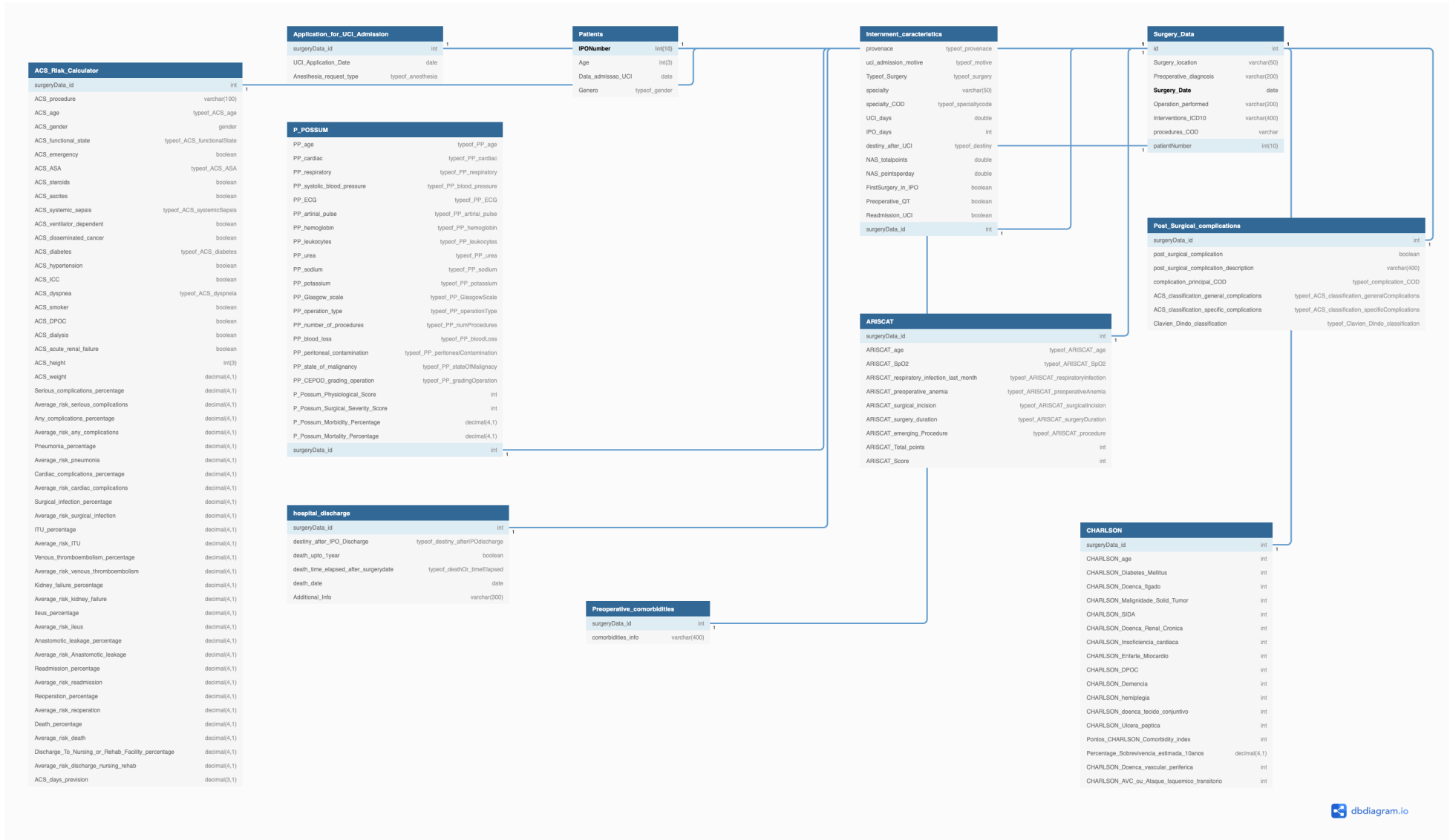


Figure 3.1: Schematic representation of the database tables.

## 3.4 Architecture

### 3.4.1 Web Platform Architecture

It is proposed that the IPOscore web platform be implemented following a REST architecture, where there will be one or more clients (front-end) and a server (back-end), with an API that will handle the communication between both.

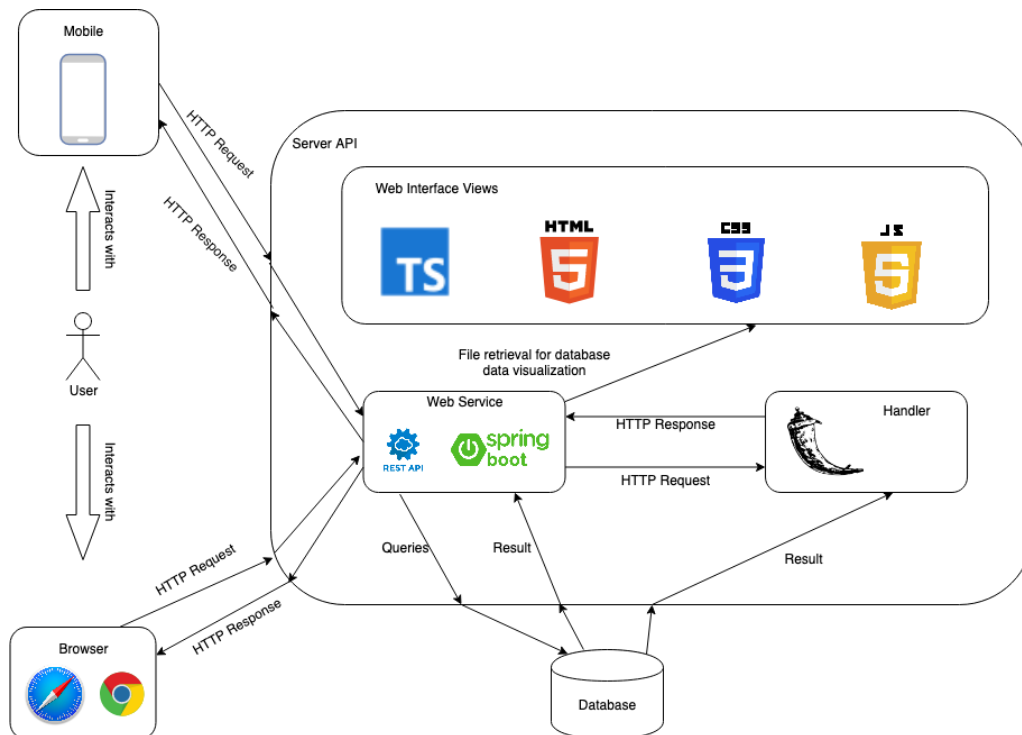


Figure 3.2: Overview of the platform architecture.

The proposed platform architecture (see 3.2) consists of two main components: 1) the back-end server, represented by Web Service (Spring Boot [7](version 2.2.5)) and the Python handlers (Flask) and 2) the front-end web represented by the Web interface views. The form of communication, represented by the arrows, between the *REST API* of the IPOscore and the various clients, is outlined, this communication is made through the web, using REST requests.

The users of the IPOscore platform are the various physicians, who access the web application, implemented in Angular, through the browser.

## 3.5 Tools and Technology

The *IPOscore* platform is *web based* and the choice of technologies to use is made easier in some aspects, as it is known at the outset that at least *Hyper Text Markup Language (HTML)* [35] (HTML5), *Cascading Style Sheets (CSS)*[34] (CSS3) and *JavaScript* [36] (ES2015) in the client implementation, as they are the standard languages used in the development of

web applications, with no great alternatives. In addition to these, they will also be used in the implementation of the *Angular* client with usage of *Typescript* [33] (version 4.4.3). On the server-side, the final choice for the database implementation was *MySQL*[37] (version 8.0.20).

### 3.5.1 Client-Side

#### 3.5.1.1 Angular

*Angular* [5] is an open-source web application framework developed by Google for building client applications by creating efficient and sophisticated single-page apps (SPAs) in HTML and either JavaScript or a language like TypeScript that compiles to JavaScript. The framework consists of several libraries, some of them core and some optional.

Using *Typescript*[33] as its main programming language, *Angular* was used to create the whole client side of the application.

### 3.5.2 Server-Side

#### 3.5.2.1 MySQL

Developed, distributed and supported by Oracle Corporation, *MySQL* [37] is the most popular open source database management system. *MySQL* is a Relational Database Management System based on SQL (Structured Query Language), the most common standardized language used to access databases. A relational database stores data in separate tables, instead of putting all the data in a large pantry. The structures are organized in physical files optimized for speed. *MySQL* is also based on a client-server model, where its core is *MySQL Server*.

*MySQL Server*, which handles all the database instructions (to add, process and access data), is also available as a separate program in a networked client-server environment and as library that can be embedded (or even linked) into separate applications. *MySQL* was originally developed to handle large databases more quickly. Even though it's typically installed only on one machine, it's able to send the database to multiple locations as users are able to access it via different *MySQL* client interfaces. These interfaces send SQL statements to the server and then display the results.

#### 3.5.2.2 Liquibase

*Liquibase* [30] is an independent open source database solution for tracking, managing, applying and deploying changes to the schema of the database. The integration of *Liquibase* into the code version management system (applying also CI / CD efforts) can synchronize database versions with application versions, for faster and better quality software release cycles.

To manage changes to the database, *Liquibase* uses scripts denominated change sets. These scripts can be stored in a variety of different formats, including XML, JSON and

SQL. One single master file shows all files in the change set.

Liquibase also keeps track of which change sets already have been executed. For example, upon trying to issue an update command to the database Liquibase analyzes the current state of the master file and identifies which changes occurred. Then it performs the rest of the changes, offering the most recent revision of the defined database structures. Liquibase was used to create the structure and all the tables of the database. Firstly, a XML file is created with the properties of the connection to the database: i) Name of the file that keeps the log changes; ii) url to the database, iii) username, iv) password and v) the driver/connector type of the database (in this case, the technology used: MySQL).

### 3.5.2.3 Spring boot

The Spring [7] framework is an open source Java platform that provides comprehensive infrastructure support for developing robust Java applications very easily and very rapidly. It's used to create a micro Service. Micro Service is an architecture that allows developers to independently develop and deploy services. Each running service has its own process and this achieves the lightweight model for supporting business applications. Spring boot was used in order to create the back-end server, a microservice to communicate with the front-end server to send and retrieve information between both parties.

### 3.5.2.4 Flask

Flask [49] is a micro web framework written in Python. It is classified as a microframework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. Flask offers suggestions, but doesn't enforce any dependencies or project layout. However, Flask supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for object-relational mappers, form validation, upload handling, various open authentication technologies and several common framework related tools.

## 3.6 Implementation

In this section, it's described in detail the web-based platform, with each subsections focusing on its major components and features. The last section explains in detail the webserver(s).

For more details about the implementations and all the source code can be consulted at GitHub repository: [https://github.com/joaomartins-git/IP0score\\_app](https://github.com/joaomartins-git/IP0score_app)

### 3.6.1 Web Interface and user experience

As with any application that is intended to be marketed, it is important to take some care with the presentation of the GUI, to ensure that it fits what is intended and that users will have a good experience when using it. Otherwise, this application is unlikely to succeed. Therefore, during the implementation of the GUI for IPOscore, attempts were made to always find solutions that would provide the end user with the best possible user experience. We tried to implement an appealing interface, which could attract potential new users of the application.

#### 3.6.1.1 Aesthetic aspect

In order not to overwhelm users with information, it is imperative that the system has a minimalist design, showing only the information they need at all times. Information that is of little relevance or rarely needed should be presented only in the background, leaving room for the information that really matters.

With the IPOscore platform an effort was made to show only the truly essential information. Although the database has dozens of properties associated with controls, many of them are omitted much of the time. The type and amount of information shown varies depending on the current selection of elements.

#### 3.6.1.2 Adapt the system to users

It is essential that the system gives users a sense of familiarity, using terms and phrases that are not too complex and with which all users are comfortable, as well as adopting aspects of situations that users would expect to encounter in the real world. The IPOscore platform, whenever possible, tries to reproduce scenarios that users are used to and does not use technical or complex terms, making it easy for anyone to use.

#### 3.6.1.3 Application components

The application is splitted in two main parts: the side navigation bar (which is composed by 7 different tabs) and the top navigation bar. It doesn't look like much but since the

application is a SPA, it interacts with the user dynamically, rewriting the current web page with new data from the web server, instead of the usual method of the browser loading entire new pages. A SPA (Single-Page Application) is a web application capable of dynamically rewriting parts of the page, without needing to reload it to reflect a change. Whenever there is a change in the data, the update is done automatically, without the need for user action.

These are all components of the client, however, the server also has a structure that allows it to meet all the needs of the client (see Figure 3.4 to check all the different tabs available in the platform).

In order to separate the implementation logic of each component of the **IPOscore platform** and to allow easy application extensibility, several templates were created. Each of these templates represents a component, or part of a component, of the IPOscore platform.

With this separation, it becomes easy to add new types of elements in the future, just create a new template for that element and include it in the project. It is also possible to use the same template over and over again, eliminating the need to repeat the code.

### 3.6.2 Angular overview

In order to understand all that was implemented to build the web-based platform, firstly is crucial to understand how Angular works. The architecture of an Angular application depends on certain fundamental concepts [2]. The basic building blocks of the Angular structure are angular components that are organized into *NgModules*[3]. *NgModules* collects related code into functional sets; an Angular app is defined by a set of *NgModules* and an application always has at least one root module that allows bootstrapping, and typically has many more feature modules.

- Components define *views*, which are sets of screen elements that Angular can choose and modify according to program logic and data.
- Components use *services*, which provide specific functionality not directly related to views. Service providers can be injected into components as dependencies, making their code modular, reusable, and efficient.

Modules, components, and services are classes that use *decorators*. These decorators mark their type and provide metadata that tells Angular how to use them.

- The metadata for a component class associates it with a model that defines a view. A template combines standard HTML with Angular directives and linking markup that allows Angular to modify the HTML before rendering it for display.
- The metadata of a class of service provides the information Angular needs to make it available to components through dependency injection (DI).

Understanding the angular overview is crucial, because it allowed to know the angular structure, and how it works, in order to build the platform. The metadata for the components helped understand where to get the major building blocks that it needs to create and present itself, as well as its view. Components also associate a template to themselves, either directly with inline code or by reference and together with its template it describes a view.

### 3.6.2.1 Angular NgModules

*Angular NgModules* differs from and complements JavaScript modules (ES2015) [2]. An NgModule declares a compilation context for a set of components dedicated to an application domain, workflow, or even a set of closely related features. An NgModule can also combine its components with associated code, such as services, to form functional units. Every Angular application has a root module, *AppModule*, which provides the bootstrap mechanism that launches the application. An application usually contains many functional modules.

Just like JavaScript modules, NgModules can import functionality from other NgModules and allow their own functionality to be exported and used by other NgModules. An example of this is the use of the router service in your app, there you import the NgModule router. Organizing your code into separate functional modules helps you manage the development and design of complex applications for reuse. In addition, this technique allows you to take advantage of slow loading, that is, loading modules on demand, to minimize the amount of code to be loaded at startup.

### 3.6.2.2 Angular Components

Every Angular application has at least one component, *the root component*, that connects a hierarchy of components with the Page Document Object Model (DOM). Each component defines a class that contains data and application logic and is associated with an HTML template that defines a view to be displayed in a target environment. The `@Component()` decorator identifies the class immediately below it as a component and provides the associated component-specific template and metadata.

A *template* combines HTML with Angular markup that can modify HTML elements before they are displayed. Template directives provide program logic, and binding markup connects your application data and the DOM. There are two types of data binding:

- Event association enables your application to respond to user input in the target environment by updating application data.
- Property binding allows you to interpolate calculated values from your application's data into HTML code.

Before a view is displayed, Angular evaluates directives and resolves link syntax in the model to modify HTML and DOM elements based on your program's data and logic.



Angular supports two-way data binding, which means that changes to the DOM, such as user choices, are also reflected in the program data.

### 3.6.2.3 Angular Services

For data or logic that is not associated with a specific view and that you want to share between components, you create a service class. A class of service definition is immediately preceded by the `@Injectable()` decorator. This decorator provides the metadata that allows other providers to be injected as dependencies into your class. *Dependency Injection (DI)* allows you to keep your component classes simple and efficient. They do not retrieve data from the server, validate user input, or connect directly to the console; they delegate these tasks to the services (see figure 3.3 for an overview of all the components together).

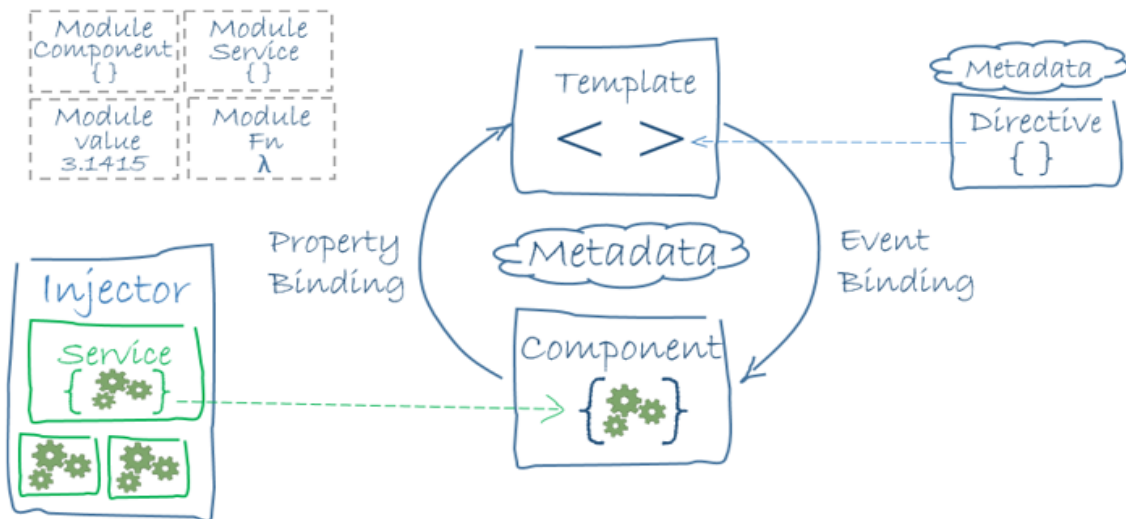


Figure 3.3: Scheme of the angular architecture (adapted from [4].)

### 3.6.3 Client structure

In order to separate the implementation logic of each component of the *IPOscore platform* and to allow easy application extensibility, several templates were created. Each of these templates represents a component, or part of a component, of the *IPOscore platform*. This was possible thanks to the dynamism ensured by Angular, as the components are Typescript-specific files or some more common ones like HyperText Markup Language and Cascading Style Sheets (\*.css, \*.html and \*.ts), their usual structure is exemplified in Listing 3.1.

With this separation, it becomes easy to add new types of elements in the future, just create a new template for that element and include it in the project. As said above, components can be reused over and over again, eliminating the need to repeat code.

The component files were also organized in folders, in order to allow easy categorization. The structure of these folders can be seen in figure 3.4

```
import { Component, OnInit } from '@angular/core';

@Component({
  selector: 'app-new-comp',
  templateUrl: './new-comp.component.html',
  styleUrls: ['./new-comp.component.css']
})

export class NewCompComponent implements OnInit {

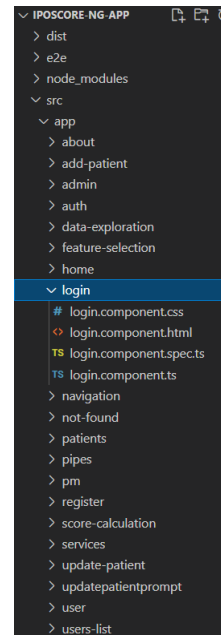
  constructor () {

  }

  ngOnInit(): void {

  }

}
```



Listing 3.1: example of basic metadata for the Patients' Angular component (\*.css, \*.spec.ts, \*.html and \*.ts)

Figure 3.4: Structure of the components in the frontend

Every single one of these components (that are organized in folders) are essential templates for the functioning of the *IPOscore platform* because each one of them represents a functioning feature crucial to the platform.

## 3.7 Content web server

### 3.7.1 Main web server

The web server is a very simple content hub, where the latest information and multimedia contents are stored. The IPOscore platform downloads, and uploads, such data from the webserver via a REST API, that communicates with a MySQL database, whose diagram can be seen in Figure 3.1.

REST API is implemented through Spring [7] (version 2.2.5), which executes specific database queries and returns data processed in a JSON file format, that in turn, will be parsed by the app. Given the prototype nature of this implementation, no backend interface was developed (data must be inserted into the database manually) and only the necessary API calls were implemented. However, both the database structure and the API implementation are flexible, giving the possibility to be easily extended. Calls implemented for the Patients table are as follows:

- @GetMapping("/patientsInfo") - Lists all the patients available, their personal information (personal identifier, age and sex) and all his/hers clinical data (like surgery date, number of days in UCI, the operation performed, etc) used internally by the app
- @PostMapping("/patientsInfo") - Adds a new patient to the database.
- @PutMapping("/patientsInfo/id") - Given a certain ID, it updates the information of that patient in its table.
- @DeleteMapping("/patientsInfo/id") - Given a certain ID, it deletes that patient from the table and consequently from the database.

This process explained above was repeated eleven times, due to the fact that there are eleven different tables that constitute the whole database. So, with that said, all the tables in the platforms' database have at least four different types of possible calls to the API (GET, POST, PUT and DELETE). But sometimes only these four API calls aren't enough to do some certain features/tasks in the platform, so it ends up need some more specific ones like:

- @GetMapping("/patientsInfo/id") - Given a certain ID, lists all of the patients information if he/she exists in the system.
- @GetMapping("/patientsInfo/gender/gender") - Lists all the patients available in the database that matches the given GENDER.
- and so on and so forth for other types of calls to the API.

### 3.7.2 Integration Process of the modules and Secondary web servers

As mentioned before, the IPOscore platform is part of a project that uses ML algorithms to predict postsurgical outcomes in cancer patients. These techniques used, helped developing predictive models (prediction tool) to predict the different surgical outcomes. These two modules were developed by the team members of the IPOscore project. Further information and details such as features selection, hyperparameter optimization and the models' performance results are described in [15]. The best predictive models resulting from this previous study were serialized for usage in the IPOscore platform.

This module was developed using Python [45] (version 3.7.0) because they bring benefits like simplicity and consistency, access to great libraries and frameworks for ML, flexibility, platform independence, and a wide community.

Firstly, the code needed to be analysed in order to try to split, through definition of tasks, the integration process of all the components in the final prototype.

The first approach taken was through the usage of *Spring Python* [46]. Why this approach? Well, first of all, the platform's backend is all written using *Spring Boot*, so the approach here wasn't too far-fetched. Second, *Spring Python* was created with the intent to provide utilities to support any Python application, even if this one is web-based so it seemed a pretty good first choice. Unfortunately it ended up being a bust due to the fact that its usage was too complex because it almost required a code rewrite, and that is not what was intended. The same happened in the second approach.

In the second approach taken, *Jython* was used. *Jython* [20], is a Java implementation of Python, which means that it's 100% Java that compiles its code and works just like Java even though it combines expressive power with clarity. But the issue here was practically the same as before. Not so much in a complex way, but more on the time consuming way, that similar to *Spring Python*, the code itself need to be rewritten in order just to fit/function in the platform that was being designed.

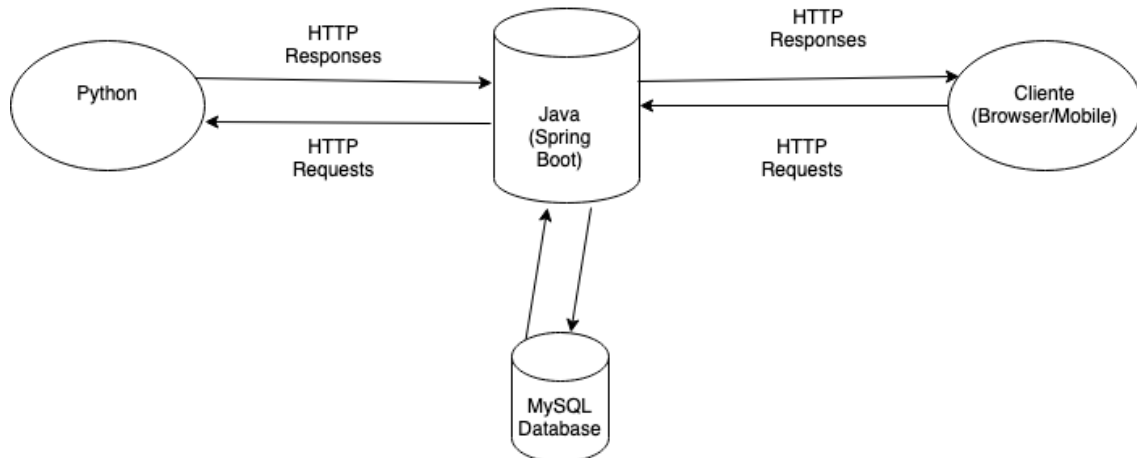


Figure 3.5: Example of the implemented interaction between Python and Java programming language.

Since both approaches were unsuccessful, there is only one solution that works certainly and for that, instead of thinking as integrating components the thought process shifted towards *microservice* depicted in Figure 3.5.

This microservice is a sort of a handler, in a way that that it handles all the HTTP requests and HTTP responses between the Spring boot backend and the Python microservice. Since there are two components let's separate this process in two phases.

### 3.7.2.1 Phase One

The integration of the "Data Exploration" and "Feature Selection" modules was easier from the "Score Calculation" module. This because its structure, as a whole (and as standalone application), was pretty similar to the IPOscore platform (it had a good logic separation between the frontend and the backend). In terms of frontend, for this component, the integration was relatively easy due to the fact that it was already desinged/implemented using HTML and JavaScript terminology so ended up being the case of "copy-paste" the onto the platform's code. The hard part was the integration of its backend server which was designed and implemented using Flask (see section 3.5.2.4) and executes specific machine learning techniques in order to return data processed, similiar to the platform's *Spring Boot* backend, in a JSON file format, that in turn will be parsed by the app to show graphical data of the variables to help discover discriminative patterns.

### 3.7.2.2 Phase Two

With phase one completed, and with the "Data Exploration" and "Feature Selection" modules fully integrated, the attention shifted towards the "Risk Score" component. Similar to the previous modules (see section 3.7.2.1), this one was also implemented using Flask, but using a slightly different framework named *Plotly Dash* [42]. This framework is usually seeked with the intent of building analytic and data visualization applications that offer highly custom user interfaces.

Similar to phase one, the intent of the backend aspect of this component is to treat it like a microservice. But since the true purpose of this component is to serve as/be like a calculator as whole so that oncologists can try to estimate certain surgery-related complications, there is no need for this backend to return the data in a JSON file format. So, with this in mind, how should this problem be addressed?

The solution, and simplest approach, found was to directly embed the Dash application [42] (version v5.0) in the already existing web application, through the use of *iframe* element in the HTML code. The way this works is to treat it like a window to the other website. On the platform's code it's possible to define the size of this window and all the environment around it.

## 3.8 Security

With the rising popularity of single page applications, mobile applications, and RESTful API services, the way web developers write back-end code has changed significantly. With technologies like Angular, we are no longer spending much time building markup, instead we are building APIs that our front-end applications consume. Our back-end is more about business logic and data, while presentation logic is moved exclusively to the front-end or mobile applications. These changes have led to new ways of implementing authentication in modern applications.

Authentication is one of the most important parts of any web application. For decades, cookies and server-based authentication were the easiest solution. However, handling authentication in modern Mobile and Single Page Applications can be tricky, and demand a better approach. One of the best known solutions to authentication problems for APIs is the JSON Web Token (JWT) [6].

### 3.8.1 JSON Web Token

A JSON Web Token is used to send information that can be verified and trusted by means of a digital signature [6]. It comprises a compact and URL-safe JSON object, which is cryptographically signed to verify its authenticity, and which can also be encrypted if the payload contains sensitive information.

Scenarios where JSON Web Tokens are useful:

- **Authorization:** the most common scenario for using JWT. Single Sign On is a feature that widely uses JWT;
- **Information exchange:** Because JWTs can be signed, JSON Web Tokens are a good way of securely transmitting information between parties.

JSON Web Tokens consist of 3 parts:

- **Header;**
- **Payload;**
- **Signature;**

The **header** typically consists of two parts: the type of the token, which is JWT, and the signing algorithm being used, such as HMAC SHA256 or RSA.

The second part of the token is the payload, which contains the claims. Claims are statements about an entity (typically, the user) and additional data. There are three types of claims: registered, public, and private claims.

- **Registered claims:** These are a set of predefined claims which are not mandatory but recommended, to provide a set of useful, interoperable claims. Some of them are: iss (issuer), exp (expiration time), sub (subject), aud (audience), and others.
- **Public claims:** These can be defined at will by those using JWTs. But to avoid collisions they should be defined in the IANA JSON Web Token Registry or be defined as a URI that contains a collision resistant namespace.
- **Private claims:** These are the custom claims created to share information between parties that agree on using them and are neither registered or public claims.

To create the signature part you have to take the encoded header, the encoded payload, a secret, the algorithm specified in the header, and sign that. For example, if you want to use the HMAC SHA256 algorithm, the signature will be created in the way shown in Figure 3.6.

```
HMACSHA256(  
  base64UrlEncode(header) + "." +  
  base64UrlEncode(payload),  
  secret)
```

Figure 3.6: Signature example for the JWT.

The signature is used to verify the message wasn't changed along the way, and, in the case of tokens signed with a private key, it can also verify that the sender of the JWT is who it says it is.

Putting all together, the output is three Base64-URL strings separated by dots that can be easily passed in HTML and HTTP environments, while being more compact when compared to XML-based standards such as SAML.

When accessing a protected route or resource, the user agent should send the JWT, typically in the Authorization header using the Bearer schema.

### 3.8.2 Implementation of the JWT

The application is built from frontend (Angular) to backend (Spring Boot), which allows users to register, login account. This application is secured with JWT (JSON Web Token) authentication and Spring Security. Then, depending on the role of current User (user, project manager or admin), this system accepts what he can access. The diagram below (Figure 3.7) show how the system handles User Registration and User Login processes:

#### 3.8.2.1 Spring Boot back-end with Spring Security

The following diagram represents the Spring Security/JWT classes, which are separated into 3 layers: HTTP, Spring Security and REST API (see Figure 3.8). It shows how the all the different types of components interact with each other.

In this three layers we can find the following components:

- SecurityContextHolder provides access to the SecurityContext.
- SecurityContext holds the Authentication and possibly request-specific security information.

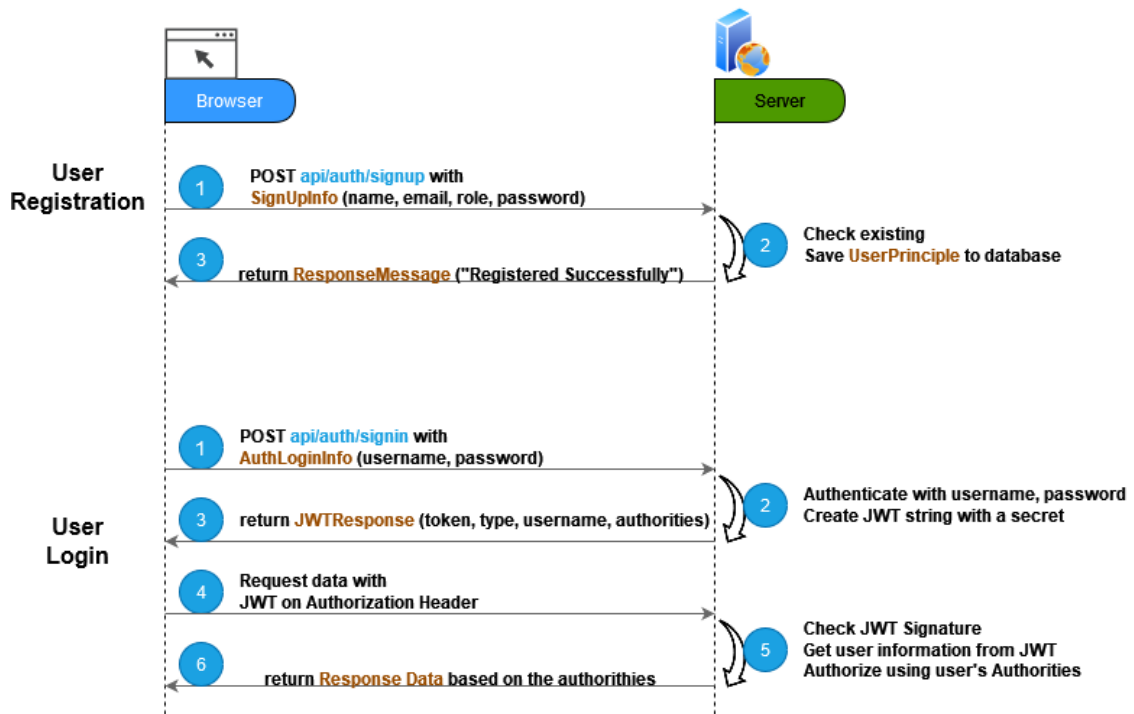


Figure 3.7: User Registration and User Login processes. adapted from [51]

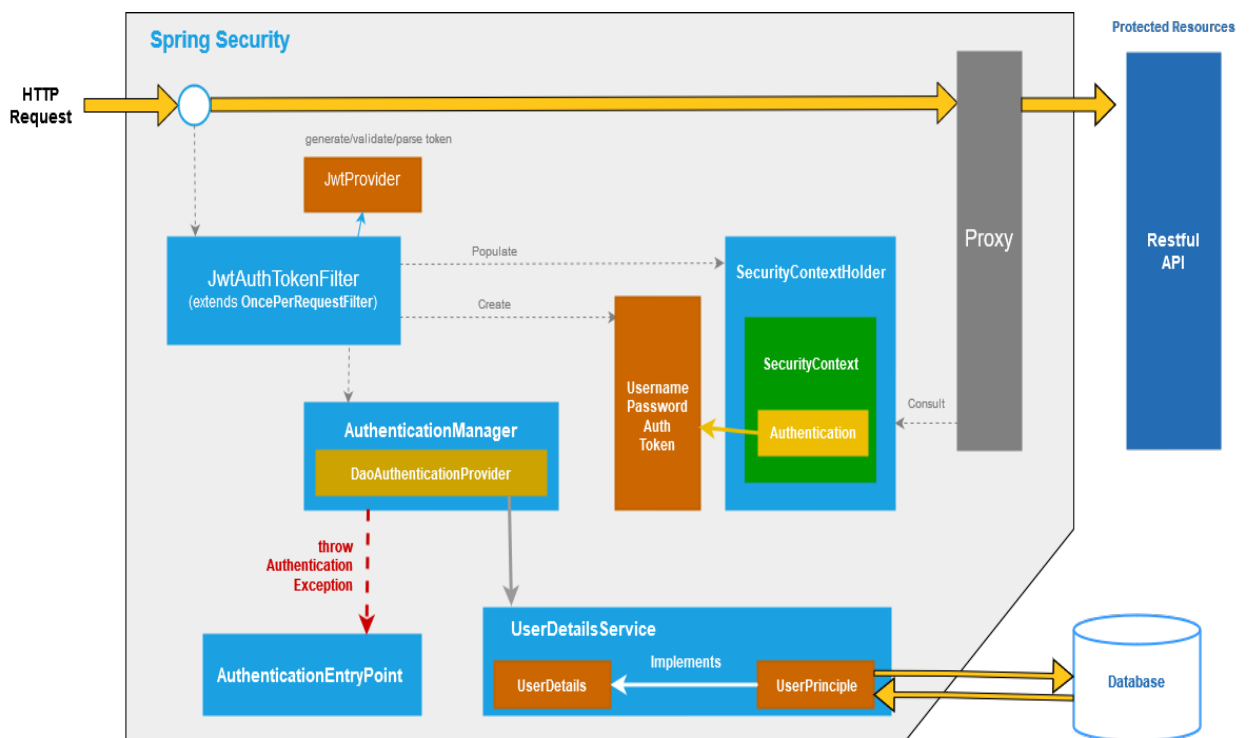


Figure 3.8: Spring Security/JWT classes. adapted from [51]



- Authentication represents the principal which includes GrantedAuthority that reflects the application-wide permissions granted to a principal.
- UserDetails contains necessary information to build an Authentication object from DAOs or other source of security data.
- UserDetailsService helps to create a UserDetails from a String-based username and is usually used by AuthenticationProvider.
- JwtAuthTokenFilter (extends OncePerRequestFilter) pre-processes HTTP request, from Token, create Authentication and populate it to SecurityContext.
- JwtProvider validates, parses token String or generates token String from UserDetails.
- UsernamePasswordAuthenticationToken gets username/password from login Request and combines into an instance of Authentication interface.
- AuthenticationManager uses DaoAuthenticationProvider (with help of UserDetailsService and PasswordEncoder) to validate instance of UsernamePasswordAuthenticationToken, then returns a fully populated Authentication instance on successful authentication.
- SecurityContext is established by calling SecurityContextHolder.getContext().setAuthentication(... ) with returned authentication object above.
- AuthenticationEntryPoint handles AuthenticationException.
- Access to Restful API is protected by HTTPSecurity and authorized with Method Security Expressions.

The presented components/steps were fully implemented and integrated in IPOscore to add spring security on the backend server in order to provide security and privacy of data.

### 3.8.2.2 Angular front-end with Interceptor

The built Angular Client allows users to register, login account. And depending on the role of current User (user, project manager or admin), this system accepts what he can access. Here, the Angular HTTP Interceptor is needed to add JWT Token Based for Security authentication (Figure 3.9).

For the Front-end interaction with the Interceptor, we can find the following components:

- app.component is the parent component that contains routerLink and router-outlet for routing. It also has an authority variable as the condition for displaying items on navigation bar.

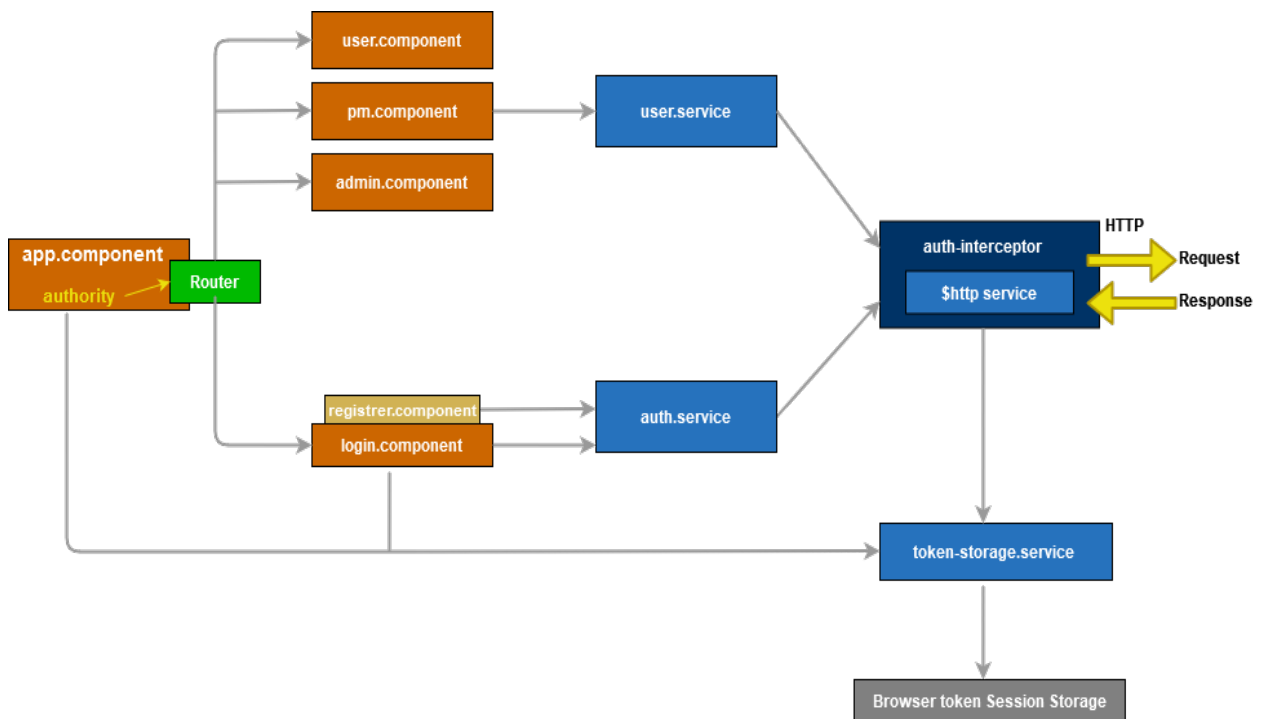


Figure 3.9: Angular front-end access with Interceptor. adapted from [51]

- user.component, pm.component, admin.component correspond to Angular Components for User Board, PM Board, Admin Board. Each Board uses user.service to access authority data.
- register.component contains User Registration form, submission of the form will call auth.service.
- login.component contains User Login form, submission of the form will call auth.service and token-storage.service.
- user.service gets access to authority data from Server using Angular HttpClient (http service).
- auth.service handles authentication and signup actions with Server using Angular HttpClient (http service).
- every HTTP request by http service will be inspected and transformed before being sent to the Server by auth-interceptor (implements HttpInterceptor).
- auth-interceptor check and get Token from token-storage.service to add the Token to Authorization Header of the HTTP Requests.
- token-storage.service manages Token inside Browser's sessionStorage.

## CONTENTS

---

All the previous described components/steps were fully integrated on the IPOscore platform in order to provide security and privacy of the users, complementing what was already made with Spring Security for the backend server.

## RESULTS AND DISCUSSION

### CONTENTS

4.1	IPOscore web-interface and modules integration . . . . .	40
4.2	Continuous Evaluation in the clinical context . . . . .	55

Evaluation by potential users is an important role in a new kind of approach, as it has a crucial part deciding whether the concept is on the right direction or needs to be rethought. Considering the population monitored at IPO-Porto as a case study, the proposed solution was applied to comprehensively understand what a Decision Support System is, how it works and how it can change the peoples lives in ways that they weren't even expecting. This chapter is organized as follows. First, an overview of the developed components. Secondly, the integration process with the whole project components is presented. This way, the progress is more easily tracked and the impact of each change can be measured and discussed.

Different modules/tabs have been used for designing the platform and are discussed as follow (see Figure 4.1 for the overall view of the modules).

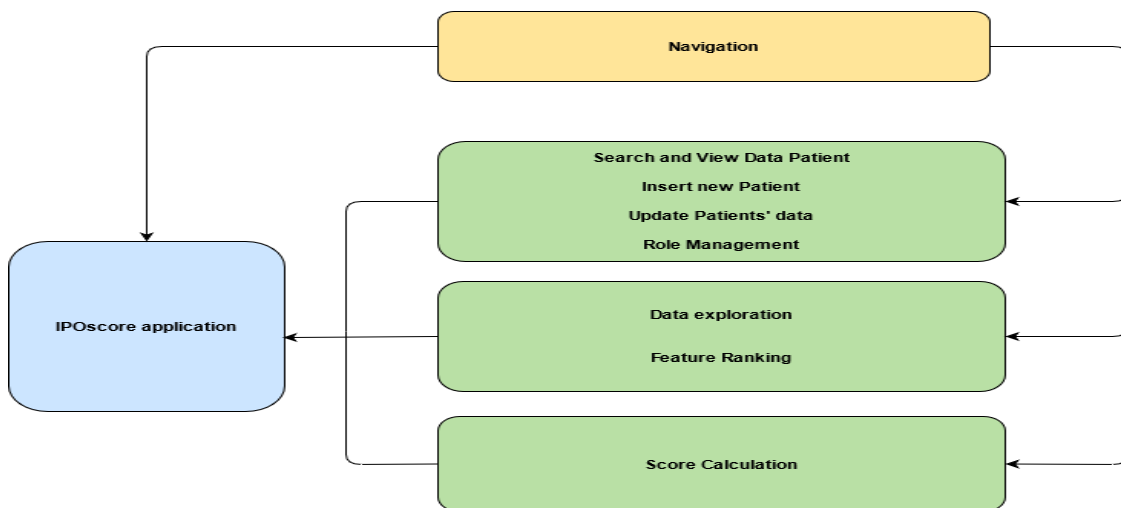


Figure 4.1: Schematic overview of the modules integrated in the IPOscore platform.

## 4.1 IPOscore web-interface and modules integration

As mentioned before, it is imperative that the web-based platform meets some requirements. In terms of platform wise, it is split in 7 different tabs (discarding the **about** component cause it is merely a disclaimer on the usage of the platform) as we can see in Figure 4.2.

The first 4 tabs relate to aspects dedicated only to the interaction of the platform with the database, that is, interactions that only either fetch or insert data onto the database. The last three tabs are components derived from the integration with two other modules (data exploratio/feature ranking?????) that make the clinical decision support system as a whole platform. Most of these components are divided into two states: an initial and a final state, that is, a state before execution an action and the state after that execution. Some of them have an extra state regarding to an action on the data.

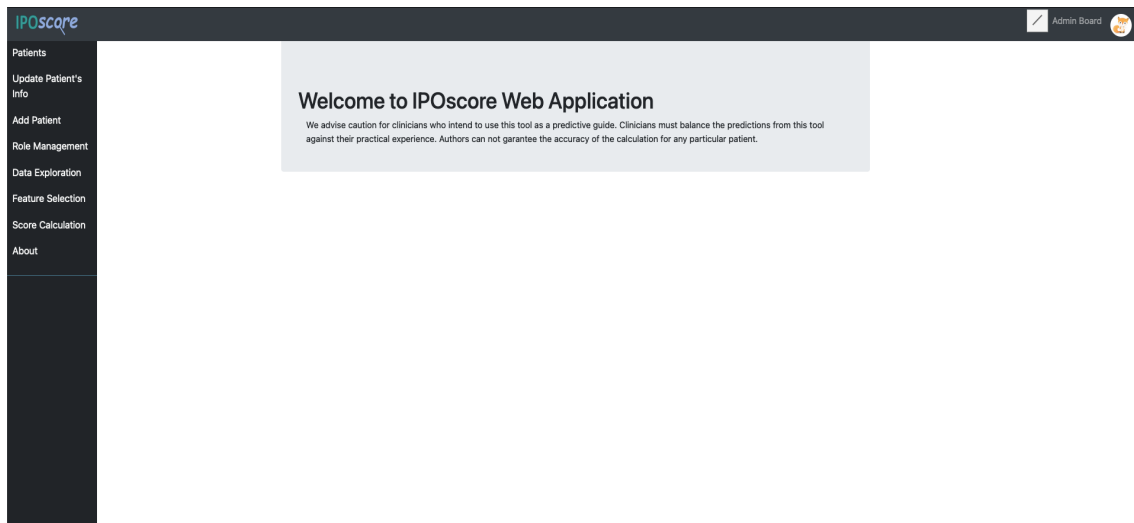


Figure 4.2: Screenshot of the IPOscore main user interface. Representative image of the basic Home tab information.

### 4.1.1 Login and Register

Logins are an important involved in keeping user accounts safe and accessible because they're a security measure designed to prevent unauthorized access to confidential and sensitive data. In the platform there are 3 different types of user: Admin, Project manager and User. The *Admin* role can perform any task in the platform, whether it is related to data input/deletion. It has full control of the platform. The *Project Manager* role is similar to the *Admin* role but it can't change any type of users information, like: username, roles, etc. The *User* role, is pretty simple. The user is only allowed to use the search and visualization of Patients data component and the last 3 components/tabs corresponding to the *Data Exploration*, *Feature Selection* and *Score Calculation* tabs.

Here, in this component, there are 3 different states:

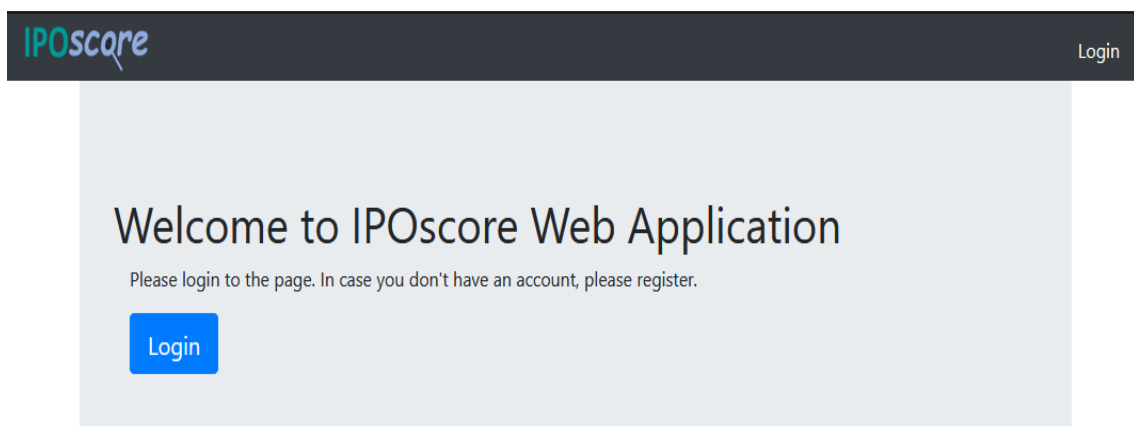
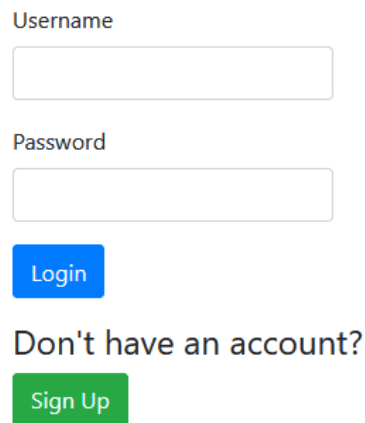


Figure 4.3: Screenshot of the Initial State for the platform

- The stationary state which represents a neutral state of the platform. Here, when the user tries to access the platform, he/she is faced with this initial panel as shown in Figure 4.3, where it is possible for the user to do one of the two options: either leave the platform or log in to it.



Username

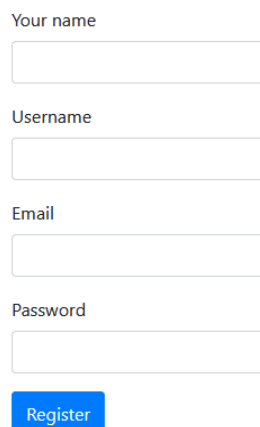
Password

Login

Don't have an account?  
Sign Up

Figure 4.4: Screenshot of the Login page for the platform.

- After the initial process, and stationary/initial state of the component, the user faces another choice. In this secondary state, they insert their credentials to access the platform.  
Here, the user also has the possibility to choose to register to the platform if they're not already registered (Figure 4.4).



Your name

Username

Email

Password

Register

Figure 4.5: Screenshot of the Register new user panel in the platform.

- Finally, where the user can simply fill the form to register to the platform (Figure 4.5). The register form is a simple form, where the user has to insert their full name (or just first and last names), a username that will be the identifier throughout the whole time they'll be using the platform, a valid email and a password (which will be encrypted in order to assure a high level of security in terms of data protection needs).

#### 4.1.2 Role Management

As any good web-based platform that also manages a lot of users, the IPOscore platform does it too. The user management is important because not all of the users have the same role (not everybody in the hospital is a doctor). We also have nurses that will use the platform as well).

As described before, the platform has 3 different roles: **Admin**, **Project manager** and **User**. These 3 have some specific, and in some cases limited, actions that they can do in the platform. Whereas a plain user can only take advantage of the visualization of Patients data component and the last 3 components/tabs corresponding to the *Data Exploration*, *Feature Selection* and *Score Calculation* tabs. With that in mind, it was better to have some sort of management over the users of the platform.

In this component, there are separated into 3 states:

Showing 6 entries

Search:

id	name	username	email	Role	Action
2	Adam	adamtest	adamtest@gmail.com	<ul style="list-style-type: none"> <li>• ROLE_USER</li> <li>• ROLE_PM</li> </ul>	Update Delete
3	Thomas	thomastest	thomastest@gmail.com	<ul style="list-style-type: none"> <li>• ROLE_ADMIN</li> </ul>	Update Delete
5	Antero Pires	anttails	anttails@gmail.com		Update Delete
6	Joana Filipa Santos	joanafs	joanafilipa@gmail.com	<ul style="list-style-type: none"> <li>• ROLE_USER</li> </ul>	Update Delete
9	ana paula	apmartins	apm@gmail.com	<ul style="list-style-type: none"> <li>• ROLE_USER</li> </ul>	Update Delete

Showing 1 to 5 of 5 entries

Previous 1 Next

Figure 4.6: Overview of all the users registered in the platform.

- The initial state where a datatable with a list of all the users that are enrolled in the platform is displayed. In this table it is possible to observe all the information of the users, their full name, email, platform username and role. In addition to this, there's also a column of the table that has two buttons: one for updating the users information and another to delete the user from the system. Finally, it is possible to filter the number of users to be viewed as it is also possible to search for a specific users name, user name, email or even role.



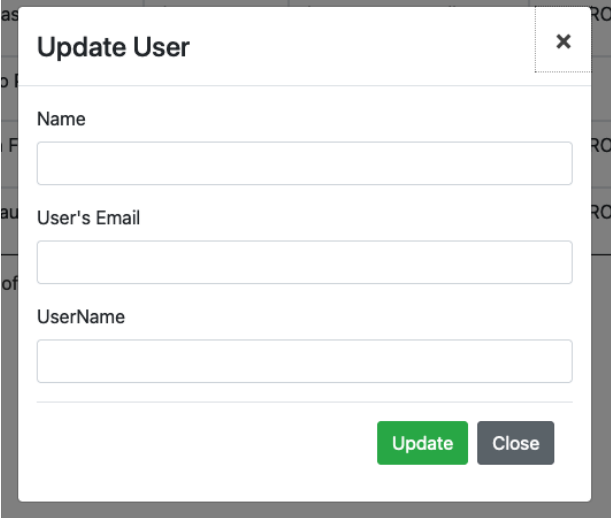


Figure 4.7: Update form for the users component.

- The secondary state, where after selecting the action to update, the admin (or admins) of the platform can change/update the user data. For the time being, the admin(s) can change data such as email, username or even your own name displayed on the platform or the users role in the platform.
- Finally, the tertiary state is where the admins have the possibility to remove users from the platform/system.

### 4.1.3 Search and View Data Patient

When the project was presented there were some specific and basic features that needed to be implemented in the the platform, like seeing the data of all or one patient, updating their information and adding new patients to the database. On this approach, the first step taken was to build the visualization of the patients data.

The component was separated into 2 different states:

- The initial state for the patients data visualization component is where the selected variables to visualize are chosen first. After that, it is possible to eventually apply some filters to restrict the data to be displayed (Figure 4.8).
- After the process of choosing what to see and how to filter the data, it is to possible to observe all the records corresponding to the selections made in the initial state (Figure 4.9).  
Also, in this particular state, the possibility to recreate the steps from the initial state is still up, so that if you want to visualize more or less variables and / or possibly apply one or more filters again to the variables in case it is necessary to restrict the data (reformulate the search).

Choose attributes to display:

Filter data:

First Set of filters (&&)

---

Second Set of filters (||)

---

Third Set of filters (|| and &&)

Attributes:	Relations:	Value:	Conjunct:
<input type="text"/>	<input type="text"/>	<input type="text"/>	
<input type="text"/>	<input type="text"/>	<input type="text"/>	or <input type="text"/>
<input type="text"/>	<input type="text"/>	<input type="text"/>	or <input type="text"/>

Figure 4.8: Screenshot of the patients tab' initial state.

Choose attributes to display:

TestNumber Age Surgery Location IPOP Days

Filter data:

First Set of filters (&&)

Attributes:	Relations:	Value:
Age	>	80
Gender	=	M

Second Set of filters (||)

Third Set of filters (|| and &&)

TestNumber	Age	IPOP Days	Surgery Location
41938339	81	• 4	• PULMÃO
42496477	85	• 24	• CAVIDADE ORAL
43875720	83	• 18	• CAVIDADE ORAL
46012630	88	• 6	• TECIDO CONJUNTIVO E PARTES MOLES DO TORAX
46285976	82	• 77	• ESTOMAGO

Figure 4.9: Representative image of a typical search and associated results table.

#### 4.1.4 Add New Patient

One thing that was also demanded from the platform was the adition of new patients to the database. All components are all pretty similiar, but in this one most of the thought processes was a bit different. This is due to the fact that there are a lot of variables that are part of the spectrum of the IPOscore project, and since the user's experience and good platform use are two important aspects of the project, it had to be done in a way that it would be easy to understand what is suppose to be done.

So, to add new patients this component was separated into 2 steps:

- The initial state, where the insertion form is filled out with the various fields corresponding to the data of the new patient to be inserted. In this state, is mandatory for the user to fill all the dropdown boxes with the information of the patient or else it woun't be possible to insert the new patient (Figure 4.10).

Figure 4.10: Screenshot of the initial state, showing in detail on of the pages of the form for adding a new Patient to the database.

Figure 4.11: Screenshot of the Final State showing details from the review process before executing the adding new Patient component.

- The final state where a review form is shown with all the information entered about the new patient.

This state serves, to review the information of the patient to be inserted. If there is any incorrect information, it is always possible to go back on the form to correct these variable(s) before submission (Figure 4.11).

#### 4.1.5 Update Patient’s Data

As described in subsection 3.6.3, there are some components that have one more specific state, and this is one of them. So, this component is a bit complex. Complex in a way that

it is possible to not only see the a table with the patients identifier, but to take actions on it. In this tab there are 4 separated states:

- Initial state where a datatable with a list of all patients, where on column represents the patients' unique identifier and one other column with two different actions/buttons: "Update patient's" information or "Delete" the patients from the system. In this state it is possible to filter the number of records to be viewed as it is also possible to search for a specific patient identifier (see Figure 4.12).

(Actions on the patients data could both count as a single state, where only one action or the other would be done. However, for explanatory reasons, it is better to consider two different states.)

Test Number	Age	Gender	IPOP Days	UCI Days	Action
475689	74	F	12	1	Update New Case Delete
1855029	73	M	7	1.87	Update New Case Delete
2820858	86	F	7	0.66	Update New Case Delete
3336607	61	M	13	1	Update New Case Delete
4087193	91	F	8	2	Update New Case Delete

Figure 4.12: Screenshot of the initial datatable with some patient's personal information

- The second state, where after selecting the action to update, it is possible to change/update the patients variables such as age or sex. This because clinical data may change rapidly and one of the purposes of the project is to maintain the data in a standard and clean way (Figure 4.13).
- The third state, similar to the secondary state, is where it is possible add a new case of study to that patient. This is important because since there are machine learning components in the platform, it will help populate the tests for better outcomes and also to keep track of the patients clinical changes throughout this process (Figure 4.14).
- And finally, the last state is where the user has the possibility to remove patients from the database.

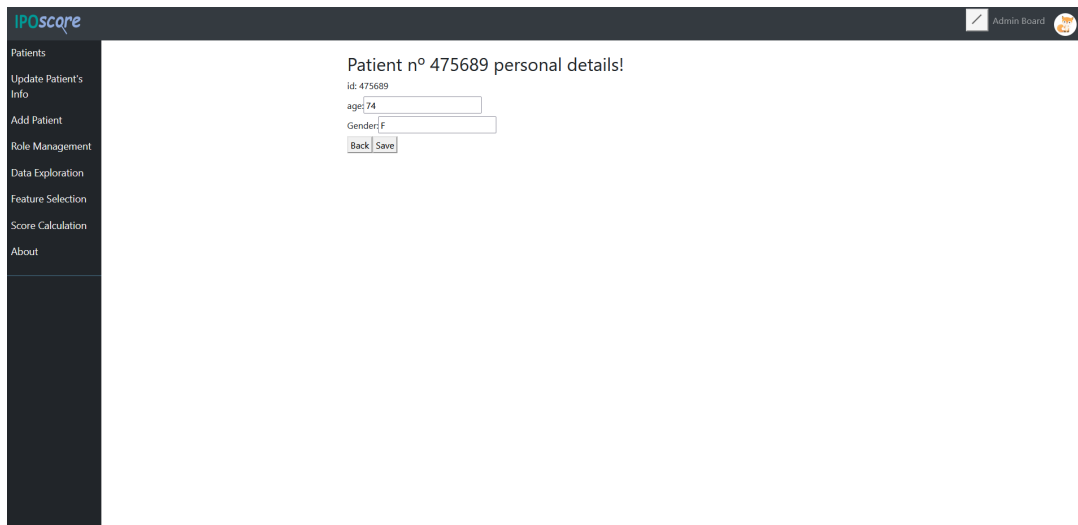


Figure 4.13: Screenshot of the secondary state showing details for the patients update for personal information.

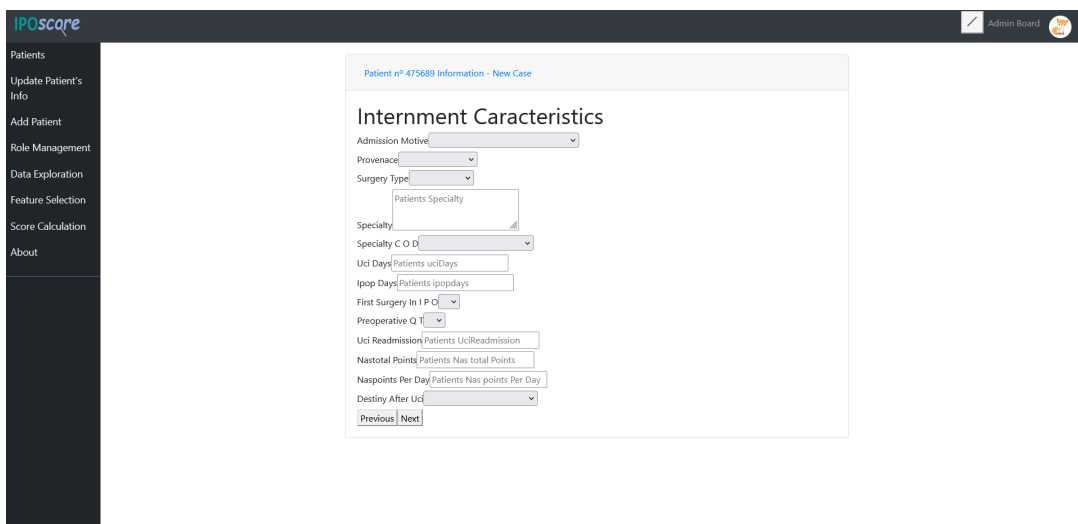


Figure 4.14: Screenshot of the third state showing details for the new case form

### 4.1.6 Data Exploration and Feature Selection Modules

After showing and explaining the main components developed for the platform, let's move on to the components related to the integration part, that is, those that were made to be integrated in the final platform. Firstly, the data exploration component will be addressed. *Data exploration* tab provides various plots to offer a better understanding of the patients' data and characteristics of the dataset.

As mentioned in section 3.2, the dataset contains a considerable amount of missings, with 11 variables reaching at least 75% missing values. The data also has 47 variables where a single value occurs for at least 70% observations. The variables can be clustered in accordance with clinical data, patient characteristics, demographic, surgical procedures, and risk scores.

Here, this component only has 1 state. By that it means that all that has to be done is to select the type of data wished to be seen (data type like categorical, binary, numeric continuous and numeric discrete data), the attribute you want to explore (it could be any patient-related attribute like age, provenance, etc), the type of visualization (it can be either violin charts, histograms, box plots or parallel coordinates) and the class label (which can only be post surgical complications or the clavien-dindo scale) (see Figure 4.15).

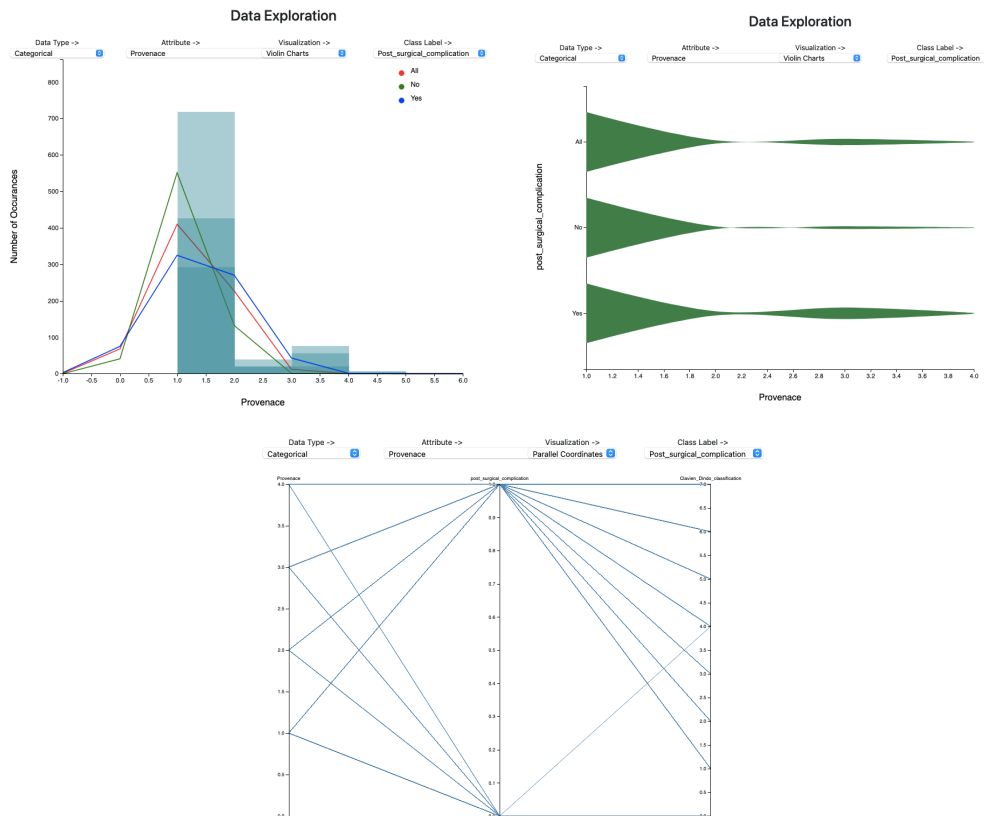


Figure 4.15: Screenshot of the data exploration visualization model - including histograms, violin charts and parallel coordinates plots. Each blue line in the parallel coordinates plot represents a patient.

Now that the data exploration component is already understood, it is possible to better understand the impact each variable might have in the output patterns (such as frequent occurrence in patterns) by making feature ranking tests. The *Feature ranking* tab displays an ordered bar plot of the most statistically significant features (scoring function which usually measures feature-relevance).

As shown in Figures 4.16 and 4.17, here it is possible to either display a single data type to "rank" (like binary data for example), with the possibility to cycle through all data types with a simple click on the left or right arrows, or either see the full display of bar charts that represent all the types. Similar to the data exploration component it is also possible to select the class label (to either post surgical complications or the clavien dindo scale) but in this particular case, the scoring function to rank the features is available for

selection as well (it has 3 statistical tests available: Chi2[22], F-Regression[31] and ANOVA [21])



Figure 4.16: Screenshot of the Feature Ranking module - single data type.

## 4.1. IPOSCORE WEB-INTERFACE AND MODULES INTEGRATION

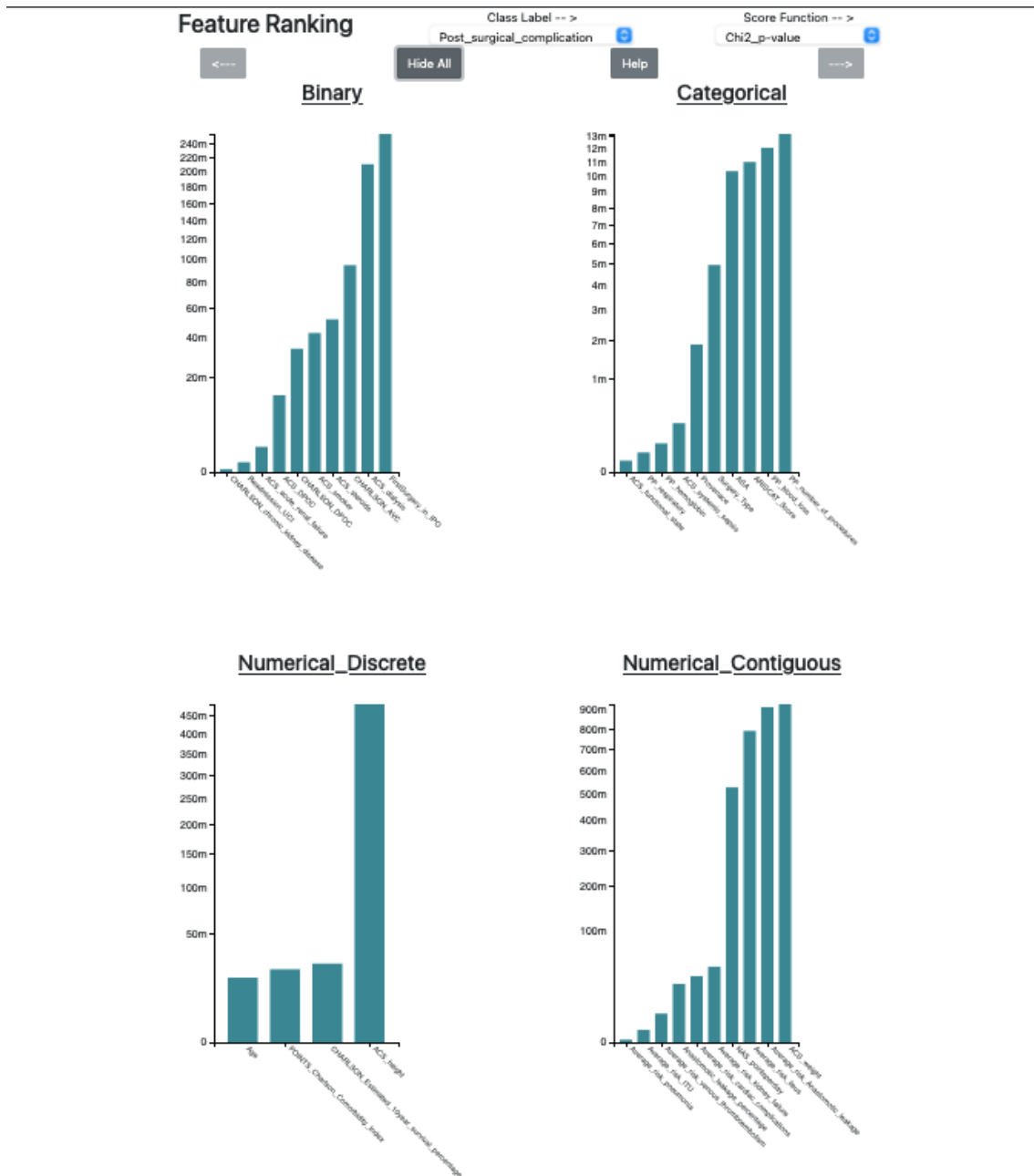


Figure 4.17: Screenshot of the Feature Ranking module - multiple data type.



#### 4.1.7 Score Prediction Module

It is known that surgical complications has been for a long time a topic of research. Being able to predict what will be the state of a patient after a certain surgical procedure has always been important and the methods used to do so have been improving. From using medical intuition into risk scores that can predict the probability of complications, their type, severity and other relevant outcomes. As mentioned before, these risk scores are based on Machine Learning techniques whose models were developed earlier, and whose detailed information about them can be found at [15].

What is represented here in this component is a risk score based on machine learning (ML) algorithms, that can predict several surgical outcome in cancer patients, by giving some pre-surgical variables and some patient characteristics.

Here in this component, there are 2 separated states:

Figure 4.18: Printsreen of the risk score module for the surgical complication outcome.

- The initial state where the user firstly selects the type outcome of interest, where they can choose between seven different outcomes: i) postoperative complications, ii) severity, iii) days in the ICU, iv) days in IPOP, v) death within one year after surgery, vi) death in months and vii) NAS points (Nursing Activity Score). As seen in Figure 4.18, after selecting the outcome, the only thing left for the user to do is to insert some inputs (like, group age, weight, number of operations made, etc.) and click calculate.
- The final state (Figure 4.19), where the user can observe the outcome from the prediction made in the initial state. Here is presented the result, which depending on the outcome selected can give different outputs (like for instance, in the case of the postoperative complications it returns low or high probability of complications

## 4.1. IPOSCORE WEB-INTERFACE AND MODULES INTEGRATION

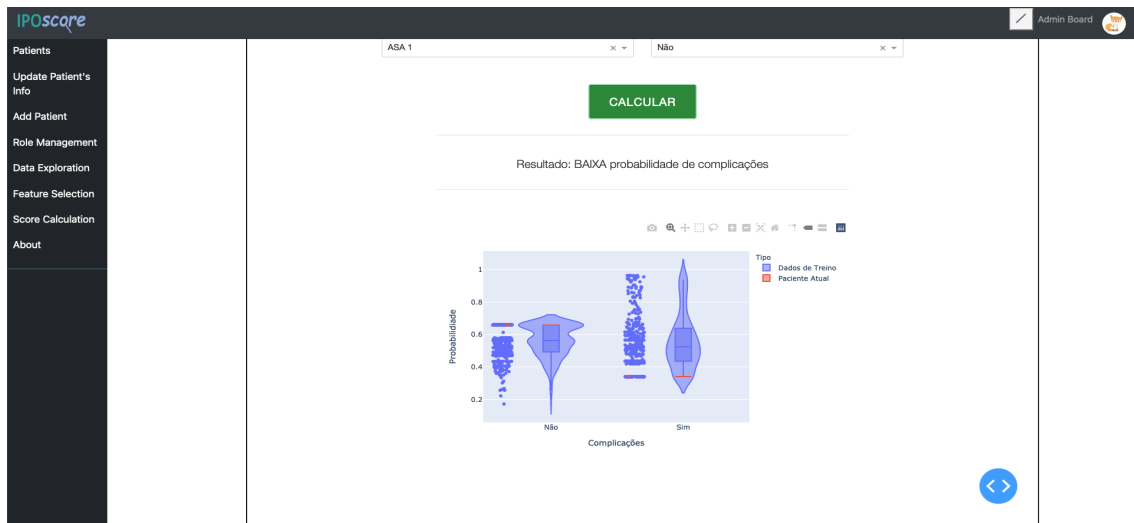


Figure 4.19: Printscren of the final step in risk score module for the complication outcome. The blue dots represent all the patients in the system and the red dots represent the patient whose input data was entered.

according to the data entered), and a plot with all the training data "used" to help predict this outcome as well as the data from the patient.

As mentioned above, this tool can help predict some different types of outcomes. Figures 4.20, 4.21 and 4.22 are some examples of the behavior of this tool when it comes to calculate possible risk outcomes.

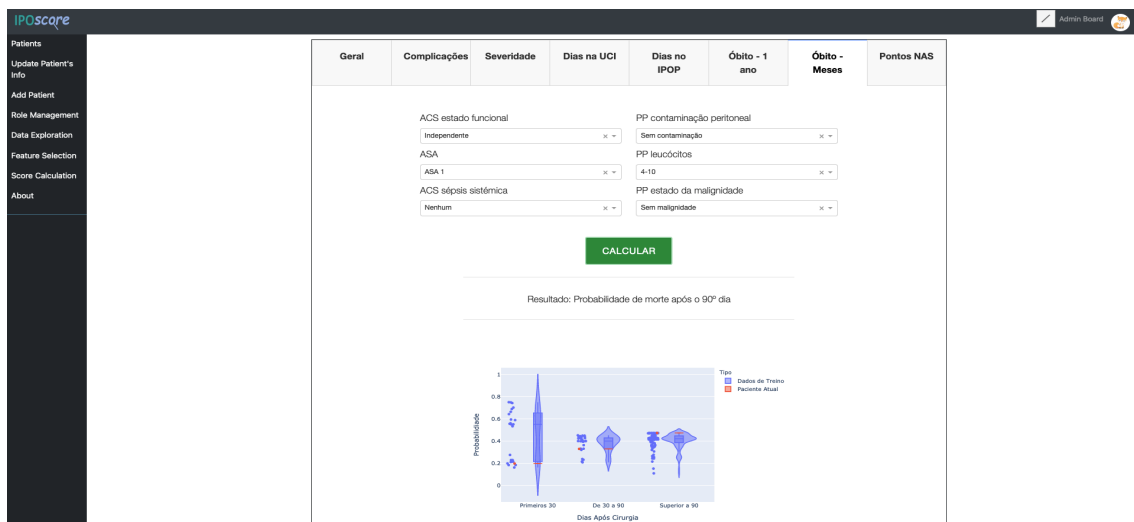


Figure 4.20: Printscren of the risk score module for "death in months". The blue dots represent all the patients in the system and the red dots represent the patient whose input data was entered.

# CONTENTS

The screenshot shows the IPOScore application interface. On the left is a dark sidebar with the following menu items: Patients, Update Patient's Info, Add Patient, Role Management, Data Exploration, Feature Selection, Score Calculation, and About. The main content area has a top navigation bar with tabs: Geral, Complicações, Severidade, **Dias na UCI**, Dias no IPOP, Óbito - 1 ano, Óbito - Meses, and Pontos NAS. Below the tabs, there are several input fields for clinical variables:

- PP nº procedimentos: Um
- ARISCAT duração cirurgia: <2 horas
- ACS sépsis sistêmica: Nenhum
- ARISCAT infecção respiratória último mês: Sim
- ARISCAT procedimento emergente: Não
- ARISCAT anemia pré-operativa: Não
- ACS insuficiência renal aguda: Não

A green button labeled "CALCULAR" is centered below the input fields. Below the button, the results are displayed: "Resultado: Internamento até 1 dia" and "Previsão: [0.52458479] ± 1".

Figure 4.21: Printscren of the first step in the risk score module for the "ICU days" outcome.

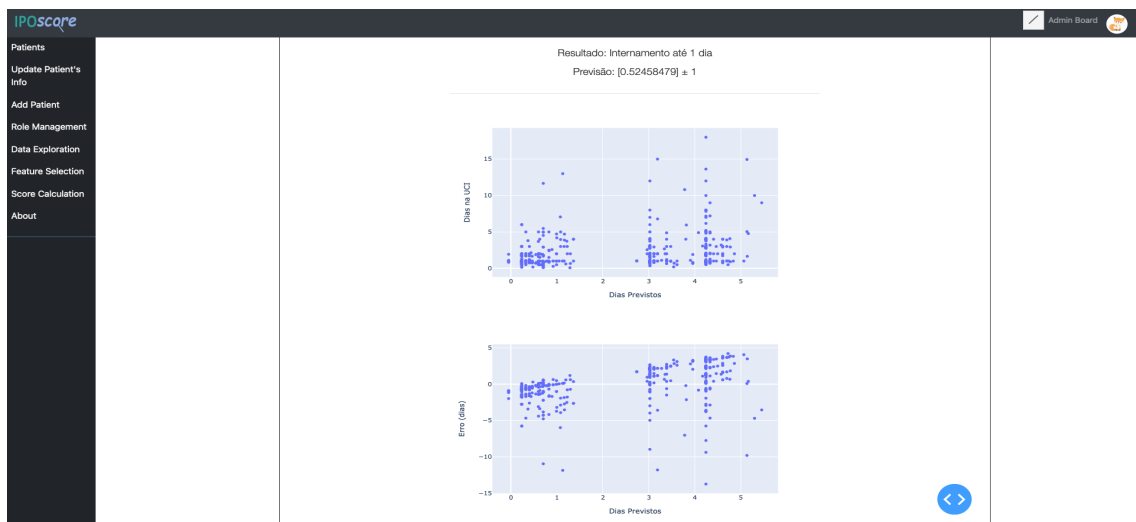


Figure 4.22: Printscren of the final step in the risk score module for the "ICU days" outcome. The blue dots represent all the patients in the system.

## **4.2 Continuous Evaluation in the clinical context**

Throughout the development process, we tried to adopt a work methodology that would allow regular reactions and opinions, of the features already developed, by some of the future users of IPOscore Platform that was being implemented. For this, the monthly IPOscore platform presentation sessions were used, since it would be difficult and not very fruitful to be gathering people just to introduce them to the results.

These presentation sessions were held monthly but not on every month. These are informal presentations, only for IPO-Porto physicians, with the aim of publicizing the changes and improvements added to the platform, during that month, by the medics who were incharge of the project.

Several of these sessions were used to make demonstrations and to present the status of the implementation of the IPOscore platform, showing and discussing the implemented features and also some that were still intended to be implemented. On several occasions, questions were raised by those present about the options taken and suggestions were made to improve features or even new features that could be implemented.

## CONCLUSIONS

## CONTENTS

5.1	Conclusions . . . . .	57
5.2	Limitations and Future Work . . . . .	57

This final chapter reflects on the realized work and discusses the feedback given by the users' evaluation. Then, wraps up with recommendations and guidelines for clinical decision support system applications and future work.

## 5.1 Conclusions

This thesis presented the IPOscore platform that is expected to support oncologists at IPO-Porto. This web-based platform integrates a new way to manage/store clinical, physiological and biopathological data of cancer patients in a structured format, visualization tools and predictive models to calculate a risk score of postoperative outcomes for the Portuguese population.

The final platform is a valuable resource to oncologists because not only it help predict several postoperative outcomes (like the number of days in ICU, postoperative complications, etc.), it also is the first web-based platform of its kind that was developed specific for the portuguese population. This is important step due to the fact that most applications of its kind are designed in other countries and for its population, using previously known data from those patients to build predictive models.

The platform yields unique properties of interest like: i) ensuring full access control, privacy and security; ii) a database for storing and retrieval of the patients data in a standard format; iii) exploration and visualization features, to explore and visualize the data; and iv) a risk prediction tool to help doctors in decision making.

## 5.2 Limitations and Future Work

When it come to these type of applications, there are always some drawbacks associated to the implementation choices.

One of these drawbacks is the fact that by treating the "external" components (the one's that were intregrated onto the platform) as a *Microservice*, it ends up having more than a single backend server. To be more specific, there are 3 "backend servers": one for the platform as a whole (which handles data retrieval, patient insertion/removal, user management, etc), one server to handle the components for the Data Exploration and Feature Selection modules and finally one server for the Score Calculation.

This is a big drawback, because it's like its consuming 3 times the memory or processing time of the application as a whole. Even though that the really "heavy" server is the platform's one and not the auxiliary servers for the other "external" components, it can end up affecting the performance of the platform.

A way to solve this drawback would be to migrate all the backend's servers in one, preferably to a single Flask backend since the 2 modules that were integrate to the platform have Flask backends.

Another drawback is the use of the *iframe* element. Although its use is very convinient to load a program directly into static page, it can actually end up slowing down the website

(in this case, the platform) or sometimes even causing security issues (fortunately this is not the case due to the fact that security was implemented onto the platform) and even end up not working because there are some browsers that specifically block the use of iframes. If the whole code of the platform ends up migrating towards Python, this problem can be solved "easily".

Mobile was topic addressed just as the platform was. Unfortunately, its completion was delayed due to programming problems. Even though this platform is web-based, and it's possible to use it in your mobile device, it had programming problems like stability or even compatibility (between Android and iOS). In a near future this can be solved by making a native mobile app (for both operating systems) instead of it being just web-based.

In the future the platform will be updated with data from new patients. Another update will be migrating all the current Java code to Python code in order to keep it consistent with the machine learning components. As the dataset size increases, the ML algorithms will be also retrained.

## BIBLIOGRAPHY

- [1] E. Adams. *Clinical decision support system as a risk assessment tool to aid in earlier diagnosis of pancreatic cancer*. URL: <https://rucore.libraries.rutgers.edu/rutgers-lib/47325/> (cit. on p. 7).
- [2] Angular. *Angular - Introduction to Angular concepts*. URL: <https://angular.io/guide/architecture> (cit. on pp. 26, 27).
- [3] Angular. *Angular Introduction to modules*. URL: <https://angular.io/guide/architecture-modules> (cit. on p. 26).
- [4] Angular. *Introduction to Angular concepts*. URL: <https://angular.io/guide/architecture> (cit. on p. 28).
- [5] Angular.io. *One framework. Mobile and desktop*. URL: <https://angular.io/> (cit. on p. 23).
- [6] auth0. *Introduction to JSON Web Tokens*. URL: <https://jwt.io/> (cit. on p. 33).
- [7] S. Boot. *Spring makes Java simple; modern; productive; reactive; cloud-ready*. URL: <https://spring.io/projects/spring-boot> (cit. on pp. 22, 24, 30).
- [8] J. Brownlee. *14 Different Types of Learning in Machine Learning*. URL: <https://machinelearningmastery.com/types-of-learning-in-machine-learning/> (cit. on p. 9).
- [9] A.R. Calculator. *ACS NSQIP Surgical Risk Calculator*. URL: <https://riskcalculator.facs.org/RiskCalculator/index.jsp> (cit. on p. 11).
- [10] U. o. C. D. o. O. Cambridge Breast Unit, U. E. C. Information, and R. C. (ECRIC). *Predict Breast Cancer*. URL: <https://breast.predict.nhs.uk/tool> (cit. on p. 15).
- [11] D. Canet. *ARISCAT Score for Postoperative Pulmonary Complications*. URL: <https://www.mdcalc.com/ariscat-score-postoperative-pulmonary-complications#use-cases> (cit. on p. 8).
- [12] J. H. K. C. Center. *Breast Cancer Recurrence Score Estimator*. URL: <http://www.breastrecurrenceestimator.onc.jhmi.edu/> (cit. on p. 16).



## BIBLIOGRAPHY

---

- [13] D. o. Q. H. S. Cleveland Clinic. *Cleveland Clinic Risk Calculator Library*. URL: <https://riskcalc.org/> (cit. on p. 14).
- [14] D. Copeland. *POSSUM for Operative Morbidity and Mortality Risk*. URL: <https://www.mdcalc.com/possum-operative-morbidity-mortality-risk> (cit. on p. 8).
- [15] L. L. S. e. R. S. C. Daniel Gonçalves Rui Henriques. *On the predictability of post-operative complications for cancer patients: a Portuguese cohort study*. URL: <https://bmcmeginformdecismak.biomedcentral.com/articles/10.1186/s12911-021-01562-2> (cit. on pp. 30, 52).
- [16] H. C. Databases. *the importance of Health Care Database Organizations and how do we benefit from them*. URL: <https://www.ncbi.nlm.nih.gov/books/NBK236556/> (cit. on p. 6).
- [17] S. Doyle-Lindrud. *The evolution of the electronic health record*. URL: <https://pubmed.ncbi.nlm.nih.gov/25840379/> (cit. on p. 7).
- [18] R. C. E. Parimbelli S.Wilk. *A Review of AI and Data Science Support for Cancer Management*. URL: <https://www.medrxiv.org/content/10.1101/2020.08.07.20170191v3> (cit. on p. 7).
- [19] M. Emily Sokol. *5 Successful Risk Scoring Tips to Improve Predictive Analytics*. URL: [https://healthitanalytics.com/features/5-successful-risk-scoring-tips-to-improve-predictive-analytics?\\_\\_cf\\_chl\\_jschl\\_tk\\_\\_=pmd\\_BwLDvVXmNxUSaWEB\\_FdXALncnlfUX3.r0kQBAJ3C3eo-1631270458-0-gqNtZGzNAmWjcnBszQj9](https://healthitanalytics.com/features/5-successful-risk-scoring-tips-to-improve-predictive-analytics?__cf_chl_jschl_tk__=pmd_BwLDvVXmNxUSaWEB_FdXALncnlfUX3.r0kQBAJ3C3eo-1631270458-0-gqNtZGzNAmWjcnBszQj9) (cit. on pp. 8, 9).
- [20] P. S. Foundation. *Jython*. URL: <https://www.jython.org/> (cit. on p. 31).
- [21] S. K. Gajawada. *ANOVA for Feature Selection in Machine Learning*. URL: <https://towardsdatascience.com/anova-for-feature-selection-in-machine-learning-d9305e228476> (cit. on p. 50).
- [22] S. K. Gajawada. *Chi2 for Feature Selection in Machine Learning*. URL: <https://towardsdatascience.com/chi-square-test-for-feature-selection-in-machine-learning-206b1f0b8223> (cit. on p. 50).
- [23] S. R. Guruvayur and R. Suchithra. *A DETAILED STUDY ON MACHINE LEARNING TECHNIQUES FOR DATA MINING*. URL: <https://ieeexplore.ieee.org/document/8300900> (cit. on p. 9).
- [24] E. Health. *History about EHR. Creation and importance of these records*. URL: <https://www.elationhealth.com/clinical-ehr-blog/history-ehrs/> (cit. on p. 6).
- [25] A. for Healthcare Research and M. Quality Rockville. *Clinical Decision Support*. URL: <https://www.ahrq.gov/cpi/about/otherwebsites/clinical-decision-support/index.html> (cit. on p. 8).

- [26] M. Jason S. ShapiroMD and M. Farzad MostashariMD. *Using Health Information Exchange to Improve Public Health*. URL: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3052326/> (cit. on p. 8).
- [27] O. A. Journals. *Most clinical risk scores are useless*. URL: <https://academic.oup.com/ndt/article/32/5/752/3819536/> (cit. on p. 8).
- [28] A. T. Kabakus. "The importance of informatics for health care industry". In: (). URL: [https://www.researchgate.net/publication/304367329\\_The\\_Importance\\_of\\_Informatics\\_for\\_Health\\_Care\\_Industry](https://www.researchgate.net/publication/304367329_The_Importance_of_Informatics_for_Health_Care_Industry) (cit. on p. 6).
- [29] K. P. E. Konstantina Kourou Themis P. Exarchos. *Machine learning applications in cancer prognosis and prediction*. URL: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4348437/> (cit. on pp. 9, 10).
- [30] Liquibase. *Track, version, and deploy database changes*. URL: <https://docs.liquibase.com/home.html> (cit. on p. 23).
- [31] S. Loukas. *How To Perform Feature Selection for Regression Problems*. URL: <https://towardsdatascience.com/how-to-perform-feature-selection-for-regression-problems-c928e527bbfa> (cit. on p. 50).
- [32] J. M. Lourenço. *The NOVAthesis L<sup>A</sup>T<sub>E</sub>X Template User's Manual*. NOVA University Lisbon. 2021. URL: <https://github.com/joaomlourenco/novathesis/raw/master/template.pdf> (cit. on p. ii).
- [33] Microsoft. *TypeScript is JavaScript with syntax for types*. URL: <https://www.typescriptlang.org/> (cit. on p. 23).
- [34] Mozilla. *CSS: Cascading Style Sheet*. URL: <https://developer.mozilla.org/en-US/docs/Web/CSS> (cit. on p. 22).
- [35] Mozilla. *HTML: HyperText Markup Language*. URL: <https://developer.mozilla.org/en-US/docs/Web/HTML> (cit. on p. 22).
- [36] Mozilla. *JavaScript*. URL: <https://developer.mozilla.org/en-US/docs/Web/JavaScript> (cit. on p. 22).
- [37] MySQL. *MySQL: why should we use it?* URL: <https://www.mysql.com/why-mysql/> (cit. on pp. 19, 23).
- [38] W. H. Organization. *Cancer*. URL: <https://www.who.int/news-room/fact-sheets/detail/cancer> (cit. on p. 6).
- [39] W. H. Organization. *Cancer. What is it? How does it affect the world? An overview on this disease*. 2018. URL: <https://www.who.int/health-topics/cancer> (cit. on p. 1).
- [40] W. H. Organization. *Portugal's Cancer Country Profile, created by the World Health Organization*. 2020. URL: [https://www.who.int/cancer/country-profiles/PRT\\_2020.pdf](https://www.who.int/cancer/country-profiles/PRT_2020.pdf) (cit. on pp. 1, 2).

## BIBLIOGRAPHY

---

- [41] P. Patel. *Electronic Health Record*. URL: <https://medium.com/fuzzycloud/electronic-health-record-c31e7fdb1be1> (cit. on p. 7).
- [42] Plotly. *Dash Enterprise 5.0*. URL: <https://plotly.com/dash/> (cit. on p. 32).
- [43] K. E. Polley M.-Y.C. Freidlin B. *Statistical and practical considerations for clinical evaluation of predictive biomarkers*. URL: <https://pubmed.ncbi.nlm.nih.gov/24136891/> (cit. on p. 9).
- [44] K. Protopapa and N. Neil Smith. *Surgical Outcome Risk Tool v2 (SORT)*. URL: <http://www.sortsurgery.com/index.php?> (cit. on p. 14).
- [45] Python. *Learn Python Programming*. URL: <https://pythonbasics.org/> (cit. on p. 31).
- [46] S. Python. *Welcome to Spring Python's documentation!* URL: <https://docs.spring.io/spring-python/1.2.x/sphinx/html/> (cit. on p. 31).
- [47] C. Rachel Nall MSN. *What to know about cancer*. URL: <https://www.medicalnewstoday.com/articles/323648> (cit. on p. 6).
- [48] D. C. B. Reed T. Sutton David Pincock. *An overview of clinical decision support systems: benefits, risks, and strategies for success*. URL: <https://www.nature.com/articles/s41746-020-0221-y> (cit. on p. 7).
- [49] A. Ronacher. *Flask: Web Development, one drop at a time*. URL: <https://flask.palletsprojects.com/en/2.0.x/> (cit. on p. 24).
- [50] F. Schera. *iManageMyHealth and iSupportMyPatients: mobile decision support and health management apps for cancer patients and their doctors*. URL: <https://ecancer.org/en/journal/article/848-imanagemyhealth-and-isupportmypatients-mobile-decision-support-and-health-management-apps-for-cancer-patients-and-their-doctors/> (cit. on p. 12).
- [51] R. Shaikh. *Spring Boot Security + JWT 'Hello World' Example*. URL: <https://dzone.com/articles/spring-boot-security-json-web-tokenjwt-hello-world> (cit. on pp. 35, 37).
- [52] A. C. Society. *Cancer Treatment Types*. URL: <https://www.cancer.org/treatment/treatments-and-side-effects/treatment-types.html> (cit. on p. 2).
- [53] M. C. Staff. *Cancer*. URL: <https://www.mayoclinic.org/tests-procedures/cancer-treatment/about/pac-20393344> (cit. on p. 2).
- [54] D. S. Systems. *Decision Support Systems (DSS), what are they and what is their purpose*. URL: <https://www.investopedia.com/terms/d/decision-support-system.asp> (cit. on p. 7).
- [55] U. of Texas MD Anderson Cancer Center. *Chemotherapy Response Calculator for Disease-free survival after breast surgery*. URL: <http://www3.mdanderson.org/app/medcalc/index.cfm?pagename=jsonconvert1> (cit. on p. 14).

- [56] U. of Texas MD Anderson Cancer Center. *Clinical Calculators*. URL: <https://www.mdanderson.org/for-physicians/clinical-tools-resources/clinical-calculators.html> (cit. on p. 13).
- [57] P. P. Yu. *Knowledge bases, clinical decision support systems, and rapid learning in oncology*. URL: <https://pubmed.ncbi.nlm.nih.gov/25715002/> (cit. on p. 7).

A

## APPENDIX - DATASET VARIABLES

Table A.1: Target cohort study: input variables and output variables (star-marked).  
 † Nominal values assumed to have an explicit ordering for value disclosure.

domain	variable	type	min <sup>†</sup>	max <sup>†</sup>	domain	variable	type	min <sup>†</sup>	max <sup>†</sup>
PATIENT DATA	ICU admission date	date	-	-		ACS procedure	text	-	-
	age	numerical	22	97		ACS age	categorical	<65	>85
	genre	binary	female	male		ACS functional status	categorical	independent	total dependency
	pre-operative comorbidities	text	-	-		ACS emergency	binary	no	yes
INTERMENT	provenance	categorical	operating room	other	ACS RISK	ACS ASA	categorical	healthy patient	dying patient
	reason for ICU admission	categorical	post-operative	septic shock		ACS steroids	categorical	no	yes
	surgery type	binary	elective surgery	urgent		ACS ascites	binary	no	yes
	specialty	categorical	thoracic	other		ACS systemic sepsis	categorical	none	septic shock
	days at the ICU*	numerical	0.1	18.0		ACS ventilation dependency	binary	no	yes
	days at IPOP*	numerical	1	280		ACS disseminated cancer	binary	no	yes
	destination after ICU*	categorical	home	other hospital unit		ACS diabetes	categorical	no	insulin
	NAS points*	numerical	2.1	995		ACS hypertension	binary	no	yes
	surgery recurrence	binary	no	yes		ACS ICC	binary	no	yes
	preoperative QT	binary	no	yes		ACS dyspnea	categorical	no	when resting
SURGERY DATA	ICU readmission*	binary	no	yes	ACS smoker	binary	no	yes	
	ASA	categorical	healthy patient	dying patient	ACS DPOC	binary	no	yes	
	location	categorical	abdomen	vulva	ACS dialysis	binary	no	yes	
	preoperative diagnosis	text	-	-	ACS acute renal failure	binary	no	yes	
	surgery date	date	-	-	ACS height	numerical	137	193	
	surgery class	text	-	-	ACS weight	numerical	36	169	
	interventions: ICD10	text	-	-	serious complications (%)	numerical	1	73.4	
	procedures: COD	text	-	-	any complication (%)	numerical	1.4	77.4	
	post-surgical complication*	binary	no	yes	pneumonia (%)	numerical	0	40.1	
	complication description*	text	-	-	cardiac complications (%)	numerical	0	27.1	
POST SURGICAL COMPLICATION	main complication: COD*	text	-	-	surgical infection (%)	numerical	0.1	31.4	
	secondary complications: COD*	text	-	-	ITU (%)	numerical	0.1	17.3	
	ACS general complications*	binary	serious	any complications	venous thromboembolism (%)	numerical	0	10.9	
	ACS specific complications*	categorical	pneumonia	discharge: post-acute care	renal failure (%)	numerical	0	21.6	
	Clavien-Dindo classification*	categorical	0	7	anastomotic leak (%)	numerical	0.9	9.3	
ADMISSION REQUEST	anesthesia request date	date	-	-	readmission (%)	numerical	1.3	41.3	
	anesthesia type	categorical	no request	request: complex surgery	reoperation (%)	numerical	0.4	29.0	
	ARISCAT age	categorical	<51	>80	death (%)	numerical	0	96.5	
ARISCAT	ARISCAT SpO2	numerical	>95%	<91%	discharge to nursing/rehab (%)	numerical	0.3	92.1	
	ARISCAT resp. infection	binary	no	yes	ACS forecast of hosp. days	numerical	0	30	
	ARISCAT preoperative anemia	binary	no	yes	PP age	categorical	<61	>70	
	ARISCAT surgical incision	categorical	peripheral	intrathoracic	PP respiratory	categorical	without dyspnea	resting dyspnea/fibrosis	
	ARISCAT surgery duration	categorical	<2	>3	PP ECG	categorical	normal ECG	other abnormal rhythms	
CHARLSON	ARISCAT emerging procedure	binary	no	yes	PP systolic blood pressure	categorical	110-130mmHg	<90mmHg	
	ARISCAT score	categorical	<26	>44	PP arterial pulse	categorical	50-80 bpm	<40 or >120 bpm	
	Charlson age	categorical	<50	> 79	PP hemoglobin	categorical	13-16 g/dl	<10 or >18 g/dl	
	Charlson diabetes mellitus	categorical	none	damaged organ	PP leukocytes	categorical	4-10	>20 or <3	
	Charlson liver disease	categorical	none	medium to severe	PP urea	categorical	<7.6	>15	
	Charlson malignity	categorical	none	solid and metastasized	PP sodium	categorical	>135 mmol/l	<126 mmol/l	
	Charlson AIDS	binary	no	yes	PP potassium	categorical	3.5-5 mmol/l & <2.9 or >5.9 mmol/l		
	Charlson chronic kidney disease	binary	no	yes	PP glasgow scale	categorical	15	<9	
	Charlson heart failure	binary	no	yes	PP type of surgery	categorical	minor surgery	major/complex surgery	
	Charlson myocardial infarction	binary	no	yes	PP number of procedures	categorical	1	>3	
	Charlson COPD	binary	no	yes	PP blood loss	categorical	<100 mls	>1000 mls	
	Charlson peripheral vascular	binary	no	yes	PP peritoneal contamination	categorical	no contamination	intestine/pus/blood	
	Charlson AVC/transient ischemic	binary	no	yes	PP state of malignancy	categorical	not malignant	malignant with metastasis	
	Charlson dementia	binary	no	yes	POSSUM physiological score	numerical	12	61	
	Charlson hemiplegia	binary	no	yes	PP CEPOD-classification	categorical	elective	emergency surgery (<2h)	
	Charlson connective tissue	binary	no	yes	PP cardiac	categorical	no heart failure	jugular venous pressure	
	Charlson peptic ulcer	binary	no	yes	PP surgical severity score	numerical	9	38	
	Charlson comorbidity index	numeric	0	13	POSSUM physiological score	numerical	12	61	
	% estimated 10-year survival	numeric	0	100	PP CEPOD-classification	categorical	elective	emergency surgery (<2h)	
	destination after IPO*	categorical	home	other hospital unit	PP cardiac	categorical	no heart failure	jugular venous pressure	
death up to 1 year*	binary	no	yes	PP surgical severity score	numerical	9	38		
surgery-to-death time (<1year)*	categorical	<30 days	90 to 365 days	PP % morbidity	numerical	9.3	100		
				PP % mortality	numerical	0.4	99.5		



2022 Platform for Sustainable Growth